ระบบควบคุมฟัซซี่
FUZZY CONTROL SYSTEM

โดย

นายกมล เกี่ยวศรีกุล

นางสาวถวิดา มณีวรรณ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

สาขาวิชาวิศวกรรมระบบควบคุม

สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง

ปีการศึกษา 2536

ปริญญานิพนธ์ ปีการศึกษา 2536

ภาควิชา วิศวกรรมระบบควบคุม

คณะ วิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง

เรื่อง ระบบควบคุมฟัซซี่

FUZZY CONTROL SYSTEM

ผู้จัดทำ 1. นายกมล เกี่ยวศรีกุล รหัสประจำตัวนักศึกษา 33100002

2. นางสาวถวิดา มณีวรรณ์ รหัสประจำตัวนักศึกษา 33100116

..... อาจารย์ที่ปรึกษา

(ผศ.ดร.จงกล งามวิวิทย์)

ระบบควบคุมฟัซซี่

FUZZY CONTROL SYSTEM

โดย	นายกมล	เกี่ยวศรีกุล	รหัสประจำตัวนักศึกษา 33100002
	Mr.Kamol	Kieawsrikul	
	นางสาวถวิดา	มณีวรรณ์	รหัสประจำตัวนักศึกษา 33100116
	Miss Thavida	Maneewarn	
อาจารย์ที่ปรึกษา	ผศ.ดร.จงกล	งามวิวิทย์	
	Assistant Professor Dr. Jongkol Ngamwiwit		

บทคัดย่อ

ในปริญญานิพนธ์ฉบับนี้ เป็นการศึกษาและประยุกต์ใช้ทฤษฎีของฟัซซี่เซต (Fuzzy set) และฟัซซี่ลอจิก (Fuzzy logic) สำหรับการนำมาสร้างตัวควบคุมฟัซซี่ (Fuzzy controller) ซึ่งเป็นตัวควบคุม (Controller) ที่ทำงานโดยใช้กฎ (Rule) อันมีลักษณะคล้ายคลึงกับภาษามนุษย์ (Human linguistic) เป็นตัวตัดสินใจในการควบคุมระบบ ทำให้ตัวควบคุมฟัซซี่ เหมาะที่จะนำมาใช้แทนการควบคุมที่เดิมมีมนุษย์เป็นผู้ควบคุมซึ่งต้องอาศัยทักษะความชำนาญนอกจากนี้ตัวควบคุมฟัซซี่ ยังเหมาะกับการควบคุมระบบที่มีความซับซ้อนเนื่องจากไม่ต้องใช้แบบจำลองคณิตศาสตร์ (Mathematical model)

นอกจากนั้น ในปริญญานิพนธ์ยังสร้างซอฟแวร์จำลองการทำงาน (Simulation software) ของตัวควบคุมฟัซซี่ โดยแสดงผลการควบคุมด้วยผลตอบสนองเชิงเวลา (Time response) ซึ่งสามารถปรับค่าฟังก์ชันระดับการเป็นสมาชิก (Membership function) และกฎการควบคุม (Control rule) ให้เหมาะสมกับระบบต่างๆ ได้ตามต้องการ และได้พัฒนาซอฟแวร์ดังกล่าวให้สามารถทำหน้าที่ตัวควบคุมฟัซซี่ บน Personal Computer ซึ่งจะทำการส่งและรับสัญญาณผ่าน A/D และ D/A เพื่อนำไปใช้ในการควบคุมกระบวนการ (Process) ซึ่งได้ถูกจำลองขึ้นด้วยเครื่อง Simulator

ABSTRACT

In this thesis is a study and implementation of fuzzy set and fuzzy logic theorem for constructing fuzzy controller. Fuzzy controller is the controller that operates by using fuzzy rule which is similar to human linguistic to be a control decision in controlling system. Thus fuzzy controller is appropriate for using instead of skilled human operator. Apart from that, fuzzy controller is suitable for controlling a complex system because it does not need to use mathematical model.

We invented a software for simulating a control action of fuzzy controller that will display a time response of the controlled system. This controller can be adjusted membership function and control rules that are suitable for the system. Furthermore, we develop a fuzzy controller software on Personal Computer that will generate and receive signal via A/D and D/A card for controlling a process that was simulated by a simulator.

กิตติกรรมประกาศ

ผู้จัดทำขอขอบคุณผู้ให้คำแนะนำและความช่วยเหลือทุกท่านดังต่อไปนี้

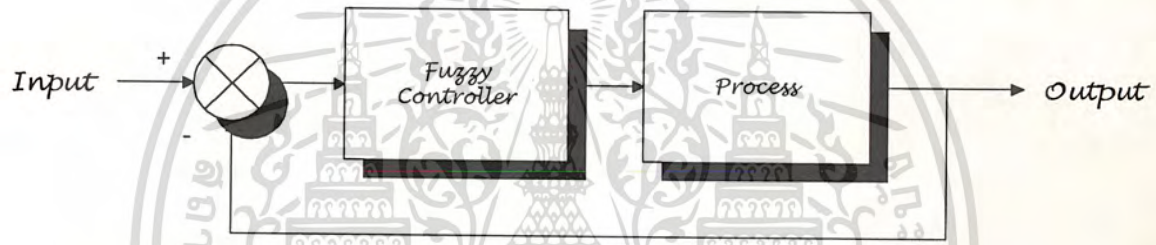
1. *ผศ.ดร.จงกล งามวิวิทย์* อาจารย์ที่ปรึกษา ภาควิชาวิศวกรรมระบบควบคุม
2. *อ.สุเธียร เกียรติสุนทร* อาจารย์ประจำภาควิชาวิศวกรรมระบบควบคุม
3. *นายสมพร ชาญประจักษ์วณิช* บัณฑิตรุ่น 28 ภาควิชาวิศวกรรมระบบควบคุม
4. *นายทรงฤทธิ์ มณีวงศ์วัฒนา* นักศึกษาชั้นปีที่ 4 ภาควิชาคอมพิวเตอร์
5. *นางสาวสุชาดา เตโชติรส* นักศึกษาชั้นปีที่ 3 ภาควิชาอิเลคทรอนิคส์

สารบัญ

บทที่ 1
บทนำ

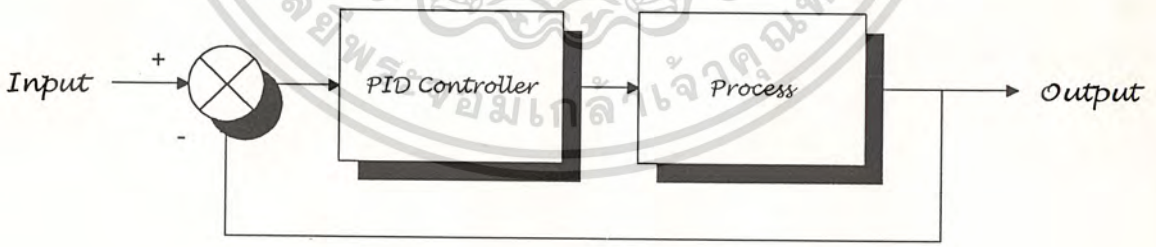
ในการควบคุมกระบวนการทางอุตสาหกรรม มีวิธีการในการควบคุมหลายวิธีซึ่งโดยปกติแล้วเรามักใช้ PID Controller ในการควบคุม แต่ในปัจจุบันกระบวนการทางอุตสาหกรรมมีความซับซ้อนมากขึ้น ซึ่งทำให้การควบคุมโดยใช้ PID Controller มีขีดจำกัดอยู่ระดับหนึ่ง และ เนื่องจากความซับซ้อนของกระบวนการ ทำให้การหารูปแบบของกระบวนการ ออกมาเป็นแบบจำลองทางคณิตศาสตร์ทำได้ลำบากจึงเริ่มมีการคิดวิธีการควบคุมแบบใหม่ๆ เพื่อแก้ปัญหาดังกล่าว ซึ่งปัจจุบันมีวิธีการในการควบคุมกระบวนการที่ได้รับการคิดค้นและพัฒนาขึ้นมาหลายวิธี ซึ่งวิธีการที่ได้รับความนิยมอย่างแพร่หลาย คือ การควบคุมโดยการเลียนแบบความคิดของมนุษย์ ได้แก่ การควบคุมโดยใช้ทฤษฎี Fuzzy Set และ Neural Network

สำหรับ Project ที่นำเสนอนี้เป็นการควบคุมกระบวนการโดยใช้ Fuzzy Controller ซึ่งอาศัยหลักการของ Fuzzy Set มาใช้ในการออกแบบ Controller ส่วนกระบวนการที่จะทำการควบคุมนั้น เราจะจำลองด้วย Transfer Function ของกระบวนการ ซึ่งแสดงให้เห็นดังรูปที่ 1



รูปที่ 1 กระบวนการควบคุมด้วย FUZZY CONTROLLER

นอกจากนี้ใน project ที่นำเสนอยังได้ทำการเปรียบเทียบกับการควบคุมโดยใช้ PID Controller ซึ่งระบบควบคุมแสดงดังรูปที่ 2



รูปที่ 2 กระบวนการควบคุมด้วย PID CONTROLLER

การควบคุมกระบวนการใน project ที่นำเสนอนี้ทุกขั้นตอนจะจำลองการทำงานด้วย Computer โดยระบบที่เราจำลองขึ้นทั้งหมดนั้นเป็นระบบ input เดียว output เดียว จากนั้นเราจะทำการนำ Fuzzy Controller ไปประยุกต์ใช้งานจริงโดยการนำ Fuzzy Controller ซึ่งเป็น Software ที่ได้ทำการสร้างขึ้น ร่วมกับ Computer Hardware ไปใช้ในระบบควบคุมป้อนกลับ (Feedback Control System) ซึ่งได้มาจากการจำลองกระบวนการ (Process) จริงขึ้นด้วย Simulator *(Automatic Control Simulator ,Model PTS-10,PACIFIC,TAIHEIYO KOGYO CO.LTD.)*

Fuzzy Control คืออะไร ?

ระบบ Fuzzy Control เป็นระบบที่เรียกว่า rule-based system ซึ่งหมายความว่าจะมี fuzzy rules หรือ กฎของ Fuzzy เป็นตัวตัดสินใจกลไกการทำงานของระบบควบคุมนั้นให้มีการปรับเปลี่ยนไปตามผลกระทบที่เกิดขึ้นในระบบ *จุดมุ่งหมายของระบบ Fuzzy Control ก็คือการให้ fuzzy rule-based system นี้เข้าไปแทนการควบคุมแบบเก่าซึ่งใช้มนุษย์เป็นผู้ควบคุม (skilled human operator)*

Fuzzy control จะมีประโยชน์ในสถานการณ์ที่

- ไม่มี mathematical model ที่ยอมรับได้สำหรับ plant นั้นๆ
- เดิมมีการใช้คนที่มีประสบการณ์เป็นผู้ควบคุมกระบวนการ ซึ่งต้องอาศัยกฎของการควบคุมที่ไม่ชัดเจนและคลุมเครือ

ทำไมจึงต้องใช้ Fuzzy Control ?

เหตุผลที่ต้องใช้ Fuzzy Control นั้นแบ่งได้เป็น 2 ส่วนใหญ่ๆ คือ เหตุผลทางทฤษฎีและเหตุผลทางปฏิบัติ

1. เหตุผลทางทฤษฎีสำหรับ Fuzzy Control

- โดยทั่วไปแล้วหลักการทางวิศวกรรมที่ดีนั้น ควรจะต้องใช้ข้อมูลที่มีอยู่อย่างมีประสิทธิภาพสูงสุด เมื่อ Mathematical model ของระบบนั้นยากเกินไปที่จะหามาได้ (ซึ่งเป็นไปได้มากในระบบที่ใช้ในการปฏิบัติจริงทั่วไป) ข้อมูลที่สำคัญที่สุดก็จะมาจาก

1) Sensor ซึ่งจะวัดค่าตัวแปรที่สำคัญออกมาเป็นตัวเลข

2) ผู้เชี่ยวชาญซึ่งจะอธิบายระบบออกมาเป็นภาษาของมนุษย์ (linguistic description) และ Control Instruction

Fuzzy Controller นั้นสามารถออกแบบโดยนำข้อมูลจากผู้เชี่ยวชาญ ซึ่งมีลักษณะไม่ชัดเจนนั้น มาใช้ได้อย่างมีระบบและมีประสิทธิภาพ. ซึ่ง Controller แบบเก่านั้นไม่สามารถนำข้อมูลแบบภาษาของมนุษย์ (linguistic description) มาใช้ได้ จึงทำให้*ในสถานการณ์ที่ข้อมูลที่สำคัญที่สุดได้มาจากผู้เชี่ยวชาญนั้น Fuzzy Controller จะเป็นทางเลือกที่ดีที่สุด*

- *Fuzzy Control เป็น model-free approach* กล่าวคือไม่ต้องการ mathematical model ของระบบเหล่านั้นก็ได้มายากขึ้นตามไปด้วย. ดังนั้น model-free approach จึงมีความสำคัญมากขึ้น. การ Control แบบเก่านั้นก็มีบ้างเหมือนกันที่เป็น model-free approach เช่น non-linear adaptive control และ PID Control

- *Fuzzy Control นั้นเป็น nonlinear controller ซึ่งถูกปรับแต่งโดยทฤษฎี universal approximation.* Fuzzy logic controller นั้น สามารถปรับแต่งให้ทำงานเป็น nonlinear control action ใดๆ ก็ได้ ดังนั้นถ้าเลือก parameter ของ Fuzzy controller ได้เหมาะสม ก็จะสามารถควบคุมระบบ nonlinear ได้

2. เหตุผลทางการปฏิบัติสำหรับ Fuzzy control

- *เข้าใจได้ง่าย.* เนื่องจาก Fuzzy control เป็นการเลียนแบบวิธีการควบคุมของมนุษย์ซึ่งหลักการของ Fuzzy control ก็สามารถที่จะเข้าใจได้ง่ายดายแม้แต่โดยบุคคลซึ่งไม่ใช่ผู้เชี่ยวชาญ. ใน 2 ทศวรรษที่ผ่านมา ทฤษฎีการควบคุมแบบเก่าได้มีการใช้ mathematical tools ที่ซับซ้อนมากขึ้นเรื่อยๆ ซึ่งก็เป็นสิ่งจำเป็นในการแก้ปัญหาที่ยาก อย่างไรก็ตามสิ่งเหล่านี้ก็ทำให้จำนวนของวิศวกรปฏิบัติงานผู้เข้าใจในทฤษฎีเหล่านี้ลดน้อยลงไปด้วย ดังนั้นวิศวกรปฏิบัติงานที่ทำงานในการออกแบบสินค้าอุปโภคบริโภค ก็มีแนวโน้มที่จะใช้วิธีซึ่งใช้งานและเข้าใจได้ง่าย ซึ่ง Fuzzy control ก็นับว่าเป็นวิธีการหนึ่งนั่นเอง

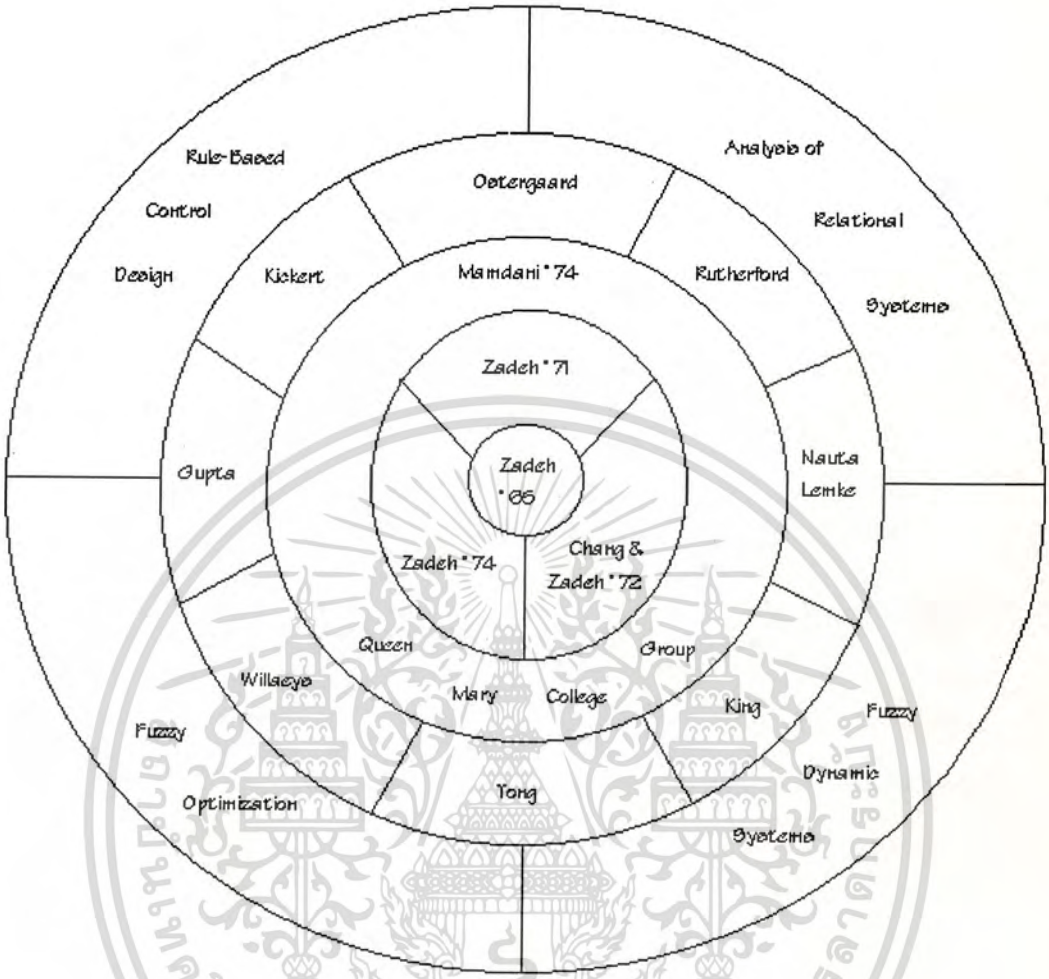
- *Implement ได้ง่าย.* ระบบ Fuzzy logic ซึ่งเป็นหัวใจของ Fuzzy control สามารถ Implement ได้ไม่ยากนัก. ปัจจุบันมี Fuzzy VLSI Chip เกิดขึ้นมากมาย ซึ่งก็จะทำให้ในการสร้าง Fuzzy controller ทำได้ง่ายและรวดเร็ว

- ใช้ค่าใช้จ่ายไม่มากนักในการพัฒนา. ในทรรศนะทางการปฏิบัตินั้น ค่าใช้จ่ายในการพัฒนานับว่าเป็นปัจจัยสำคัญที่จะบ่งชี้ความสำเร็จของผลิตภัณฑ์เนื่องจาก Fuzzy control นั้นสามารถทำความเข้าใจได้ง่าย ใช้เวลาในการเรียนรู้ไม่มากนัก ทำให้ software cost ต่ำและจากการที่ Fuzzy control สามารถ Implement ได้ง่าย จึงทำให้ hardware cost ต่ำด้วยเช่นกัน. นอกจากนี้ยังมี software tools สำหรับการออกแบบ Fuzzy controller อีกด้วย. ดังนั้น Fuzzy controlจึงเป็นวิธีการที่มีอัตราส่วนระหว่างประสิทธิภาพการทำงานต่อราคาสูงมากทีเดียว

ประวัติย่อของ FUZZY CONTROL

การวิจัยทางด้าน **FUZZY CONTROL** และ **FUZZY MODELLING** มีการเจริญเติบโตขึ้นอย่างรวดเร็วนับตั้งแต่ *Professor Lotfy A. Zadeh* แห่ง University of Berkeley ได้เสนอหลักการทางคณิตศาสตร์ของทฤษฎี **FUZZY SET** ขึ้นมาเป็นครั้งแรกในปี ค.ศ. 1965 . ทฤษฎี FUZZY SET และ **FUZZY ALGORITHM** ได้นำไปประยุกต์ใช้ในการควบคุมเป็นครั้งแรกโดยใช้ควบคุมเครื่องทำความร้อน (Steam engine) โดย *Professor Mamdani* แห่ง London University ในปี ค.ศ. 1974 .มีการอภิปรายถึงสถานะปัจจุบันของผลงานวิจัยที่เกี่ยวกับ **FUZZY CONTROLLER** โดย *Mamdani* และ *Sembi* ในปี ค.ศ. 1980 . *Willaeys* , *Malvache* และ *Hammad* ได้เสนอวิธีในการสร้าง FUZZY CONTROLLER ขึ้นมาจากพื้นฐานของ FUZZY MODEL ของ CONTROL PROCESS (1977). แนวคิดของ FUZZY ALGORITHM ได้ถูกนำไปใช้ในการควบคุมความเร็วของ D.C. Motor โดย *Willaeys*, *Malvache* และ *Mangin* (1977). *Willaeys*, *Malvache* (1978,1979) , *Braae*, *Rutherford* (1979) ได้ทำการศึกษาถึงอิทธิพลของพารามิเตอร์ต่างๆ ในการสร้าง FUZZY CONTROLLER MODEL. *Braae*, *Rutherford* (1979) ได้เสนอ **Linguistic analysis** ของ FUZZY CONTROLLER MODEL. Application ของ FUZZY CONTROL สำหรับระบบซึ่งเป็น Multidimensional, nonlinear และ มี deadtime ได้ถูกสร้างขึ้นโดย *Mamdani* และ *Procyk* (1979). Algorithm สำหรับตรวจสอบ FUZZY ได้รับการเสนอจาก *Czogala* และ *Pedrycz* (1981) . และ *Tong* (1979) ได้ทำการเสนอแนวทางสำหรับการสร้าง FUZZY MODEL ซึ่งประกอบไปด้วย **Verbalisation, Fuzzyfication** และ **Identification.**

ทฤษฎี FUZZY ได้นำไปประยุกต์ใช้ในการควบคุมกระบวนการอุตสาหกรรมอย่างแพร่หลายในทศวรรษที่ 80 อย่างแพร่หลายและได้รับการพัฒนาอย่างต่อเนื่องมาจนปัจจุบัน.



The Growth of Fuzzy Control

วัตถุประสงค์ของปริญญานิพนธ์

1. ศึกษาหลักการของการควบคุมกระบวนการด้วย Fuzzy Controller ซึ่งอาศัยหลักการของทฤษฎี Fuzzy set และ Fuzzy logic
2. ทำการจำลองการทำงาน (Simulation) ของ Fuzzy Controller บน PC computer
3. เปรียบเทียบระบบควบคุมป้อนกลับ (Feedback Control System) ที่ทำการควบคุมด้วย Fuzzy Controller และ PID Controller
4. ศึกษาการเลือกใช้กฎพื้นฐาน (Rules) และ ฟังก์ชันระดับการเป็นสมาชิก (Membership function) ให้เหมาะสมกับกระบวนการ (Process)
5. ศึกษาแนวทางในการนำ Fuzzy Controller ไปประยุกต์ใช้งานจริงโดยการนำ Fuzzy Controller ซึ่งเป็น Software ที่ได้ทำการสร้างขึ้น ร่วมกับ PC Hardware ไปใช้ในระบบควบคุมป้อนกลับ (Feedback Control System) ซึ่งได้มาจากการจำลองกระบวนการ (Process) จริงขึ้นด้วย Simulator *(Automatic Control Simulator ,Model PTS-10, PACIFIC,TAIHEIYO KOGYO CO.LTD.)*

บทที่ 2
ทฤษฎี

ระบบควบคุมฟัซซี่ (FUZZY CONTROL) มีพื้นฐานแนวความคิดมาจากทฤษฎีฟัซซี่ลอจิก (FUZZY LOGIC) และทฤษฎีฟัซซี่เซต (FUZZY SET). *ฟัซซี่ลอจิก (FUZZY LOGIC)* มีพื้นฐานอยู่บนทฤษฎีฟัซซี่เซต (FUZZY SET) ซึ่งจะช่วยให้สามารถอธิบายการปฏิบัติการและกฎการควบคุมของระบบเป็นคำพูดได้ชัดเจนขึ้น หลักสำคัญของ *ทฤษฎีฟัซซี่เซต (FUZZY SET)* คือ ยอมรับสมาชิกที่มีลักษณะตามเซตเพียงบางส่วนเข้ามาเป็นสมาชิก ซึ่งแตกต่างจากทฤษฎีเซตดั้งเดิม (CRISP SET). *ทฤษฎีเซตดั้งเดิม (CRISP SET)* จะเน้นชัดเจนเลยว่าเป็นสมาชิกของเซตหรือไม่เท่านั้น ไม่มีการเป็นสมาชิกของเซตเพียงบางส่วน. ต่อไปเราจะกล่าวถึงรายละเอียดของทฤษฎีข้างต้นและการนำทฤษฎีดังกล่าวมาประยุกต์ใช้ในการควบคุมระบบตามลำดับ.

2.1 คลิซเซต (CRISP SET)

ทฤษฎีคลิซเซต (CRISP SET) เป็นทฤษฎีเซตที่เราคุ้นเคยกันดีโดยหลักการพื้นฐานของคลิซเซต (CRISP SET) มีดังนี้

1. การเป็นสมาชิกของเซต

" ถ้า x เป็นสมาชิกของเซต A " เราจะใช้สัญลักษณ์

$$x \in A$$

" ถ้า x ไม่เป็นสมาชิกของเซต A " เราจะใช้สัญลักษณ์

$$x \notin A$$

ซึ่งในคลิซเซต (CRISP SET) นี้เราจะระบุชัดเจนว่าสมาชิกของ Universe เป็นสมาชิกของเซต A หรือไม่ เราสามารถแทนค่าระดับการเป็นสมาชิก (Membership function) ซึ่งแทนด้วยสัญลักษณ์ μ_A ดังนี้

$\mu_A(x) = 1$ ก็ต่อเมื่อ $x \in A$

$\mu_A(x) = 0$ ก็ต่อเมื่อ $x \notin A$

2. การเป็นสับเซต

" ถ้าสมาชิกทุกตัวของเซต A เป็นสมาชิกของเซต B จะเรียกว่า เซต A เป็นสับเซตของเซต B "
เราใช้สัญลักษณ์การเป็นสับเซตดังนี้

$$A \subseteq B$$

3. การเท่ากันของเซต

" ถ้าสมาชิกทุกตัวของเซต A เท่ากับสมาชิกทุกตัวของเซต B จะเรียกว่า เซต A เท่ากับ เซต B "
เราใช้สัญลักษณ์การเท่ากันดังนี้

$$A = B$$

4. การไม่เท่ากันของเซต

" ถ้าสมาชิกทุกตัวของเซต A ไม่เท่ากับสมาชิกทุกตัวของเซต B จะเรียกว่า เซต A ไม่เท่ากับ เซต B "
เราใช้สัญลักษณ์การไม่เท่ากันดังนี้

$$A \neq B$$

5. การเป็นสับเซตแท้

" ถ้าสมาชิกทุกตัวของเซต A เป็นสับเซตของเซต B และ เซต A ไม่เท่ากับ เซต B จะเรียกว่า เซต A เป็นสับเซตแท้ของเซต B "

เราใช้สัญลักษณ์การเป็นสับเซตแท้ดังนี้

$$A \subset B$$

6. การปฏิบัติการเบื้องต้นของเซต ได้แก่

- *Complement* เป็นการสร้างเซตใหม่โดยสมาชิกของเซตใหม่จะไม่เป็นสมาชิกของ เซต A เลย

  Complement A = A′
- *Union* เป็นการสร้างเซตใหม่โดยสมาชิกของเซตใหม่จะมีสมาชิกของทุกเซตรวมอยู่
- *Intersection* เป็นการสร้างเซตใหม่โดยสมาชิกของเซตใหม่จะมีสมาชิกซึ่งเป็นสมาชิกของทุกเซต

2.2 ฟัซซี่เซต (FUZZY SET)

ทฤษฎีฟัซซี่เซต (FUZZY SET) เป็นการรวมสมาชิกของหลายๆ คลิซเซต (CRISP SET) ที่มีอยู่จริงและพอจะมีลักษณะเข้ารวมกลุ่มได้ ฟัซซี่เซต (FUZZY SET) ยอมรับการเป็นสมาชิกของสมาชิกที่มีลักษณะตามเซตเพียงบางส่วน ซึ่งจะมีการเปลี่ยนแปลงทีละน้อยระหว่างการมีคุณสมบัติของการเป็นสมาชิกอย่างครบถ้วน กับ ไม่มีคุณสมบัติของการเป็นสมาชิกเลย แม้ว่าจะไม่พร้อมกันก็ตาม จึงสามารถใช้ฟัซซี่เซต (FUZZY SET) ไปทำงานกับระบบคลิซเซตได้

1. การเป็นสมาชิกของฟัซซี่เซต

ในระบบคลิซเซต (CRISP SET) จะกำหนดเพียงว่า x เป็นสมาชิกของเซต A หรือไม่เป็นสมาชิกของเซต A โดยแทนด้วยฟังก์ชันระดับการเป็นสมาชิก (Membership function) เป็น 0 หรือ 1 แต่ฟัซซี่เซต (FUZZY SET) จะยอมรับสมาชิกที่มีลักษณะที่ถูกเพียงบางส่วนและผิดเพียงบางส่วนไม่มีขอบเขตแน่นอน การประยุกต์ใช้ทฤษฎีฟัซซี่เซต (FUZZY SET) จะต้องแสดงค่าระดับ (DEGREE) ซึ่งเป็นค่าที่เป็นไปได้ที่จะเป็นสมาชิกของเซตหรือฟังก์ชันระดับการเป็นสมาชิก (Membership function) โดยเราใช้สัญลักษณ์ μ แทนค่าระดับความเป็นสมาชิกซึ่งมีค่าระหว่าง 0 ถึง 1 แสดงดังนี้

$$\mu_A(X) \rightarrow [\,0,1\,]$$

หมายความว่า ระดับการเป็นสมาชิก (Grade of membership หรือ Degree of membership) ของ X ในฟัซซี่เซต A อยู่ในช่วงทั้งหมดจาก 0 ถึง 1 เมื่อประยุกต์เข้ากับฟัซซี่ลอจิก (FUZZY LOGIC) ค่า μ จะถูกเรียกว่า ค่าความจริง ที่แสดงค่าระดับขอบเขตของเซตคือ

$$0 \leq X \leq 1$$

โดยถ้าค่าสูงแสดงว่ามีความเป็นสมาชิกมาก , ถ้าค่าต่ำแสดงว่ามีความเป็นสมาชิกน้อย , ค่า 0 จะหมายถึงไม่เป็นสมาชิกเลยและ 1 คือเป็นสมาชิกอย่างสมบูรณ์

ขอเน้นว่าฟังก์ชันระดับการเป็นสมาชิก (Membership function) ของฟัซซี่เซต (FUZZY SET) ถึงแม้จะมีค่าระหว่าง 0 ถึง 1 แต่อย่านำไปสับสนกับความน่าจะเป็น (Probability). ฟัซซี่เซต (FUZZY SET) เป็นรูปแบบหนึ่งของความไม่แน่นอน (Uncertainly) และโดยธรรมชาติ ฟัซซี่เซต (FUZZY SET) เป็นศาสตร์ที่ไม่มีความเกี่ยวข้องกับสถิติ

2. นิยามที่ฟัซซี่เซต (FUZZY SET) ขยายมาจากคลิซเซต (CRISP SET)

- ฟัซซี่เซต (FUZZY SET) จะว่าง ก็ต่อเมื่อ ฟังก์ชันระดับการเป็นสมาชิก (Membership function) ของฟัซซี่เซต (FUZZY SET) นั้นเป็น 0 ตลอดทั้ง X

- ฟัซซี่เซต (FUZZY SET) A และ B จะเท่ากัน ก็ต่อเมื่อ $\mu_A(X) = \mu_B(X)$ สำหรับทุก ๆ x ใน X เขียนแทนด้วย A = B

- Subset หรือ Containment นิยามโดย

$$A \subset B \leftrightarrow \mu_A(X) \leq \mu_B(X)$$

- Complement ของฟัซซี่เซต (FUZZY SET) A เขียนแทนด้วย A′ นิยามโดย

$$\mu_{A'}(X) = 1 - \mu_A(X)$$

- Union ของฟัซซี่เซต (FUZZY SET) A และ B ซึ่งมีฟังก์ชันระดับการเป็นสมาชิก (Membership function) $\mu_A(X)$ และ $\mu_B(X)$ ตามลำดับสามารถแทนด้วย ฟัซซี่เซต (FUZZY SET) C โดยฟังก์ชันระดับการเป็นสมาชิก (Membership function) ของฟัซซี่เซต (FUZZY SET) C กำหนดโดย

$$\mu_C(X) = \text{Max}[\mu_A(X), \mu_B(X)] \quad \text{หรืออาจเขียนย่อเป็น}$$

$$\mu_C = \mu_A \vee \mu_B(X)$$

- Intersection ของฟัซซี่เซต (FUZZY SET) A และ B ซึ่งมีฟังก์ชันระดับการเป็นสมาชิก (Membership function) $\mu_A(X)$ และ $\mu_B(X)$ ตามลำดับสามารถแทนด้วย ฟัซซี่เซต (FUZZY SET) C โดยฟังก์ชันระดับการเป็นสมาชิก (Membership function) ของฟัซซี่เซต (FUZZY SET) C กำหนดโดย

$$\mu_C(X) = \text{Min}[\mu_A(X), \mu_B(X)] \quad \text{หรืออาจเขียนย่อเป็น}$$

$$\mu_C = \mu_A \wedge \mu_B(X)$$

อย่างไรก็ตามไม่มีหลักตายตัวว่า การ Union ต้องใช้ Max Operation, การ Intersection ต้องใช้ Min Operation หรือการ Complement ต้องใช้ $\mu_{A'}(X) = 1 - \mu_A(X)$ ดังนั้นเพื่อให้ general มากขึ้นจึงมีนิยามสัจพจน์ (Axiom) ปฏิบัติการแต่ละชนิดดังนี้

- **Complement C** : [0,1] → [1,0]

  Axiom C1 : C(0) = 1 และ C(1) = 0 หมายความว่าสามารถครอบคลุมคลิซเซต (CRISP SET)

  Axiom C2 : สำหรับทุก ๆ a,b ∈ [0,1] ถ้า a < b แล้ว C(a) ≥ C(b) หมายความว่า C เป็น Monotonic nonincreasing

ทุก ๆ การปฏิบัติที่จะถือว่าเป็นการ Complement ได้อย่างน้อยที่สุดจะต้องสอดคล้องกับสัจจพจน์ทั้งสอง

- **Union U** : [0,1] × [0,1] → [0,1]

  Axiom U1 : U(0,0) = 0 ; U(0,1) = U(1,0) = U(1,1) = 1 นั่นหมายความว่า จะต้องสามารถครอบคลุมคลิซเซต (CRISP SET)

  Axiom U2 : U(a,b) = U(b,a) ; มีคุณสมบัติการสลับที่

  Axiom U3 : ถ้า a ≤ a' และ b ≤ b' แล้ว U(a,b) ≤ U(a',b') หมายความว่า U เป็น Monotonic

  Axiom U4 : U(U(a,b),c) = U(a,U(b,c)) ; มีคุณสมบัติการจัดหมู่

ทุก ๆ การปฏิบัติที่จะถือว่าเป็นการ Union ได้อย่างน้อยที่สุดจะต้องสอดคล้องกับสัจจพจน์ทั้งสี่

- **Intersection I** : [0,1] × [0,1] → [0,1]

  Axiom I1 : I(1,1) = 1 ; I(0,1) = I(1,0) = I(0,0) = 0 นั่นหมายความว่า จะต้องสามารถครอบคลุมคลิซเซต (CRISP SET)

  Axiom I2 : I(a,b) = I(b,a) ; มีคุณสมบัติการสลับที่

  Axiom I3 : ถ้า a ≤ a' และ b ≤ b' แล้ว I(a,b) ≤ I(a',b') หมายความว่า I เป็น Monotonic

  Axiom I4 : I(I(a,b),c) = I(a,I(b,c)) ; มีคุณสมบัติการจัดหมู่

ทุก ๆ การปฏิบัติที่จะถือว่าเป็นการ Intersection ได้อย่างน้อยที่สุดจะต้องสอดคล้องกับสัจจพจน์ทั้งสี่

ตัวอย่างการปฏิบัติการ Union และ Intersection นอกเหนือจาก Max และ Min Operation แสดงดังตาราง

Reference	Fuzzy Unions	Fuzzy Intersections	Range of Parameter
Schweizer & Sklar [1961]	$1-\max[0,(1-a)^{-p}+(1-b)^{-p}-1)]^{1/p}$	$\max(0,a^{-p}+b^{-p}-1)^{-1/p}$	$p\in(-\infty,\infty)$
Hamacher [1978]	$\frac{a+b-(2-\gamma)ab}{1-(1-\gamma)ab}$	$\frac{ab}{\gamma+(1-\gamma)(a+b-ab)}$	$\gamma\in(0,\infty)$
Frank [1979]	$1-\log_s[1+\frac{(s^{1-a}-1)(s^{1-b}-1)}{s-1}]$	$\log_s[1+\frac{(s^{a}-1)(s^{b}-1)}{s-1}]$	$s\in(0,\infty)$
Yager [1980]	$\min[1,(a^w+b^w)^{1/w}]$	$1-\min[1,((1-a)^w+(1-b)^w)^{1/w}]$	$w\in(0,\infty)$
Dubois & Prade [1980]	$\frac{a+b-ab-\min(a,b,1-\alpha)}{\max(1-a,1-b,\alpha)}$	$\frac{ab}{\max(a,b,\alpha)}$	$\alpha\in(0,1)$
Dombi [1982]	$\frac{1}{1+[(\frac{1}{a}-1)^{-\lambda}+(\frac{1}{b}-1)^{-\lambda}]^{-1/\lambda}}$	$\frac{1}{1+[(\frac{1}{a}-1)^{\lambda}+(\frac{1}{b}-1)^{\lambda}]^{1/\lambda}}$	$\lambda\in(0,\infty)$

2.3 ฟัซซี่ลอจิก (FUZZY LOGIC)

ฟัซซี่ลอจิก (FUZZY LOGIC) จะมีการกระทำทางลอจิกเพื่อรวมค่าลอจิกต่างๆ ให้เป็นค่าฟัซซี่ลอจิก (FUZZY LOGIC) คล้าย ตัวแปรในระบบลอจิก 2 ระดับ โดยจะต้องมีกำหนดลักษณะความหมายให้แตกต่างกันแต่ใช้คำสั่งทางลอจิกที่เหมือนกันคือ AND, OR และ NOT ความหมายต่างๆ ในระบบฟัซซี่ (FUZZY SYSTEM) ถูกกำหนดโดย L.A. Zadeh ผู้คิดค้นระบบฟัซซี่ลอจิก (FUZZY LOGIC)

การ AND ของฟัซซี่ลอจิก

ตามคำจำกัดความของ Zadeh คือ ค่าความจริงที่น้อยที่สุด (Minimum) นั่นคือสำหรับค่าฟัซซี่ A และ B

$$\mu(A \text{ AND } B) = \min(\mu A, \mu B)$$

การ OR ของฟัซซี่ลอจิก

ตามคำจำกัดความของ Zadeh คือ ค่าความจริงเป็นค่ามากที่สุด(Maximum)

$$\mu(A \text{ OR } B) = \max(\mu A, \mu B)$$

การ NOT ของค่าฟัซซี่ลอจิก

$$\mu(\text{ NOT } A) = 1 - \mu A$$

ซึ่งการกระทำทั้ง 3 นี้ เป็นสมดุลของการกระทำในลอจิก 2 ระดับ สำหรับค่า μ มีค่าอยู่ระหว่าง 0 ถึง 1

2.4 การประยุกต์ใช้งานของฟัซซี่ลอจิก (FUZZY LOGIC APPLICATION)

โดยทั่วไปจะมีลักษณะโครงสร้างพื้นฐานแบ่งได้ 3 ส่วนหลักคือ

- *~ การเปลี่ยนคลิซเซตเป็นฟัซซี่เซต (FUZZIER)*
- *~ กฎการวินิจฉัย (INFERENCE MECHANISM THAT EMPLOYS RULES)*
- *~ การเปลี่ยนฟัซซี่เป็นคลิซ (DEFUZZIFIER)*

ในการใช้กับระบบจะต้องเปลี่ยนให้อยู่ในรูปของฟัซซี่โดเมน (FUZZY DOMAIN) เคลื่อนย้ายประมวลผลข้อมูลแล้วเปลี่ยนกลับให้อยู่ในรูป คลิซโดเมน (CRISP DOMAIN) ตามเดิมซึ่งเหมือนกับการกระทำทางอนาลอก (ANALOG) คือ เปลี่ยนให้อยู่ในรูปของโดเมนความถี่ (FREQUENCY DOMAIN) จากข้อมูลในโดเมนเวลา (TIME DOMIAN) เพราะว่ากระบวนการในโดเมนความถี่ (FREQUENCY DOMAIN) จะง่ายกว่าโดเมนเวลา (TIME DOMIAN) ในระบบฟัซซี่ (FUZZY SYSTEM) กฎพื้นฐานสามารถอธิบายการทำงานของระบบในรูปของฟัซซี่ (FUZZY) ได้ง่าย ดังนั้นเราจะเปลี่ยนค่าอินพุทในรูปของคลิซ (CRISP) ไปอยู่ในฟัซซี่โดเมน (FUZZY DOMAIN) มากกว่าจะเปลี่ยนจากฟัซซี่ไปอยู่ในรูปคลิซโดเมน (CRISP DOMAIN)

กฎการวินิจฉัย(INFERENCE)

สำหรับกฎการวินิจฉัยค่าอินพุทและค่าความจริง จะสนับสนุนเงื่อนไขสำหรับการสร้างส่วนกำหนดรูปแบบฟัซซี่ที่ช่วงปกติฟัซซี่คอมไพเลอร์ จะสุ่มค่าของอินพุทและนำมาวินิจฉัยค่า เพื่อให้ได้ผลออกมาทางเอาท์พุทของระบบตามทฤษฎี ระบบจะรวมเอาค่าที่เป็นไปได้ทั้งหมดของอินพุท เพื่อนำมาวินิจฉัยและประมวลผล แต่จริงๆ แล้วการครอบคลุมค่าเหล่านี้ไม่จำเป็นในการใช้งานปกติ

กฎการวินิจฉัยนั้นได้มาจากการสร้างความสัมพันธ์ระหว่าง input และ output ขึ้นมา เพื่อนำไปสู่ output ที่ต้องการ ซึ่งความสัมพันธ์ระหว่าง input และ output ก็คือ RULE นั่นเอง เช่น เมื่อให้ input ของ controller คือ error(er) และ change of error(ce) ของระบบป้อนกลับและให้ output ของ controller เป็น input ของ process หรือ control input (ci) เราจะสามารถสร้างความสัมพันธ์ได้ดังนี้คือ

IF er=LN AND ce=LN THEN ci=LP

OR

IF er=SN AND ce=SN THEN ci=SP

OR

.

.

.

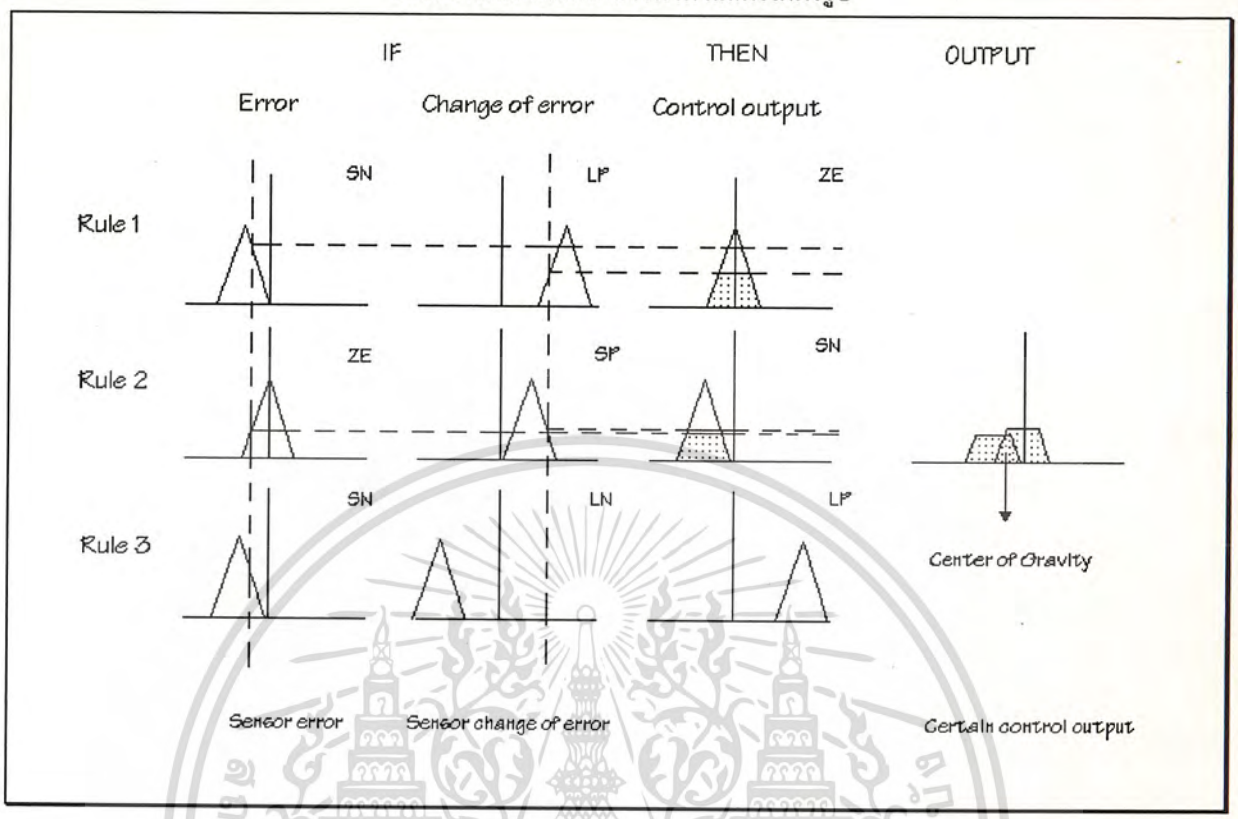
หมายเหตุ : LN = Large Negative

SN = Small Negative

LP = Large Positive

SP = Small Positive

จากความสัมพันธ์ดังกล่าวนำมาแสดงได้ดังรูป



รูปที่ 3 แสดงการ INFERENCE

ซึ่งเมื่อนำ rule ทั้งหมดมารวมเข้าด้วยกัน ก็จะได้รูป Membership Function ของ control input โดยจะนำไปหาค่าของ control input ที่ต้องการได้ต่อไป

วิธีการ DEFUZZIFIER

มีเทคนิคและวิธีการในการปลี่ยนฟัซซี่เป็นคลิช (DEFUZZIER) มีอยู่หลายเทคนิค ซึ่งจะกล่าวถึงเป็นบางเทคนิคดังนี้

1. เทคนิค Maximizer เลือกค่าสูงสุดจากหลายๆ แบบมาเพียงหนึ่ง

เป็นการใช้ค่าสูงสุดของค่าระดับการเป็นสมาชิก จากการกระทำหลายๆ แบบ แล้วเลือกกระทำเพียงหนึ่งรูปแบบ ถ้าหากเกิดการกระทำที่มีค่า μ สูงสุดเท่ากัน 2 อย่าง จะต้องใช้รูปการแก้ปัญหาอีกลักษณะหนึ่ง คือ ใช้ค่าเฉลี่ยของค่าเอาท์พุท หรือเลือกการกระทำที่สัมพันธ์กับค่าระดับของระบบพื้นฐาน ถึงแม้เทคนิค Maximizer จะเป็นวิธีที่ง่ายที่สุด แต่ก็ไม่มีประสิทธิภาพเท่าที่ควร

2. ทฤษฎีค่าน้ำหนักเฉลี่ย

จะใช้ค่าเฉลี่ยของการกระทำหลังจากการกำหนดค่าระดับของระดับการเป็นสมาชิกไว้ล่วงหน้าแล้ว เป็นวิธีที่ง่ายและใช้การคำนวณเพียงเล็กน้อย แต่ก็ยังให้ค่าที่ไม่ค่อยชัดเจน เช่นเดียวกับเทคนิค Maximizer ที่เกิดความไม่ชัดเจนก็เพราะว่าค่าเอาท์พุทของฟังก์ชันระดับการเป็นสมาชิก (MEMBERSHIP FUNCTION) มีค่าเอาท์พุทมากกว่าหนึ่งค่าต่อค่า μ ที่กำหนดให้ ค่าเอาท์พุทฟังก์ชันของการเป็นสมาชิก มีลักษณะคล้ายกับรูปปิระมิดหรือปิระมิดตัดยอด ถ้า $\mu = 0.5$ ค่าเอาท์พุทมาจากค่าฟังก์ชันของขอบสัญญาณทั้งด้านขาขึ้นและขาลง ถ้า $\mu = 1$ จะมีค่าตรงกับช่วงของสัญญาณที่เกิดขึ้นทั้งหมด

วิธีการกำจัดความไม่ชัดเจนสามารถทำได้ด้วยกระบวนการแดมปิ้งค่าในฟังก์ชันเอาท์พุท ด้วยค่าที่แน่นอนให้กลับไปอยู่ในฟังก์ชันอินพุท ซึ่งเป็นวิธีที่น่าเบื่อ และ ไม่สามารถใช้ค่าที่เป็นค่าตรงข้ามของฟังก์ชันอินพุทในการวิเคราะห์ด้วย

3. กรรมวิธีค่าศูนย์กลาง

เป็นการแสดงค่าเอาท์พุทที่สัมพันธ์กับค่าจุดศูนย์กลางมวลของเอาท์พุท ในระดับที่ทำงานเพราะว่าเราไม่ใช้ค่าขอบของฟังก์ชันระดับการเป็นสมาชิก และจะไม่เกิดความไม่ชัดเจนอีกต่อไปกรรมวิธีค่าศูนย์กลางเป็นการคำนวณที่แน่นอนและเป็นการแก้ไขข้อบกพร่องที่เกิดขึ้นในวิธีอื่นๆ

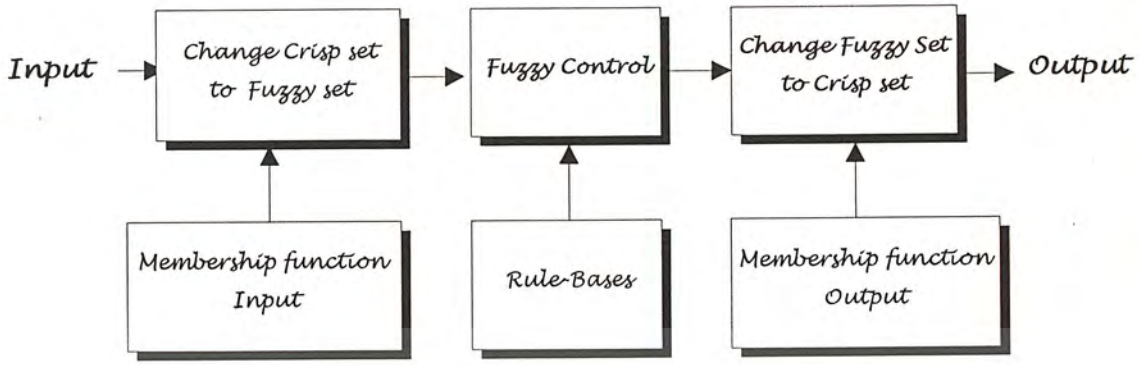
ค่าศูนย์กลางมักจะสัมพันธ์กับค่าเอาท์พุทหนึ่งค่า ผลที่ได้จะเป็นการกระทำอย่างหยาบๆ ภายในช่วงของเอาท์พุท กฎทั่วๆไปจะต้องมีการปฏิบัติเพียงจุดเดียวของแต่ละระบบ ในกรณีที่ต้องทำพร้อมๆกันหลายกฎ ในการกระทำหนึ่งครั้ง จะต้องเกิดการซ้อนทับกัน (OVERLAP) ของค่าฟังก์ชันอินพุทของระดับการเป็นสมาชิก เพื่อแก้ไขความไม่ต่อเนื่องของเอาท์พุท ถึงแม้ว่าจะมีข้อบกพร่องแต่ก็เป็นวิธีที่ดีที่สุดในการรวมกัน (COMBINATION) และการแปลงค่าฟัซซี่กลับคืน (DEFUZZIFICATION)

กรรมวิธีนี้จะรวมค่าเอาท์พุทของการกระทำหลายๆค่า เป็นค่าค่าเดียวสำหรับใช้ในระบบและค่าเอาท์พุทค่าเดียวนี้เป็นค่าน้ำหนักเฉลี่ยของศูนย์กลาง (CENTROID) ของแต่ละฟังก์ชันระดับการเป็นสมาชิก

4. กรรมวิธีซิงเกิลตัน : สังเคราะห์เอาเอาท์พุทเดียว

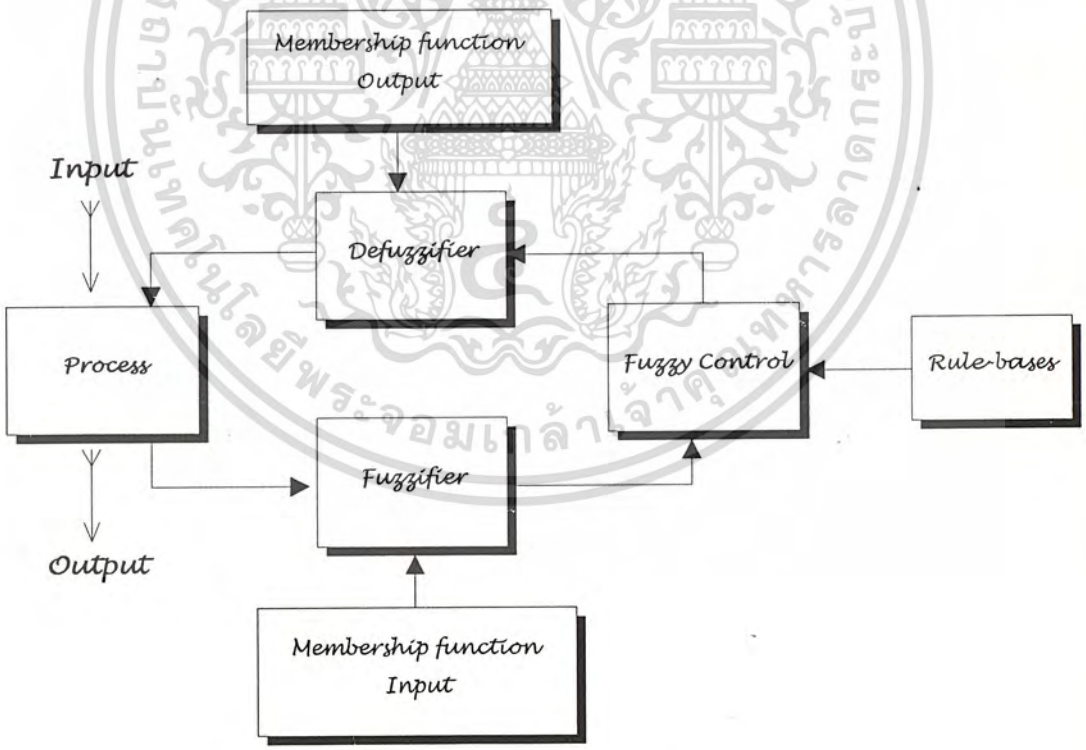
เป็นเทคนิคพิเศษของแบบกรรมวิธีค่าศูนย์กลาง หรือจะเรียกอีกอย่างหนึ่งคือ กรรมวิธี REMAINING COMBINATION/DEFUZZIFICATION วิธีนี้เป็นการนำค่าเอาท์พุทของและฟัซซี่เซตมาใช้ใหม่ เป็นค่าเอาท์พุทค่าเดียวโดยใช้ค่าน้ำหนักเฉลี่ยจากการกระทำรวมกันหลายๆอย่าง วิธีนี้ได้ค่าความถูกต้องน้อยกว่ากรรมวิธีค่าศูนย์กลาง และยังคงต้องการการซ้อนทับกันของอินพุทฟังก์ชัน เพื่อเป็นการหลีกเลี่ยงความไม่ต่อเนื่องของเอาท์พุท ด้วยหลักการและการคำนวณที่ไม่ยุ่งยาก ประกอบกับยังไม่มีใครคิดค้นวิธีใหม่และดีกว่านี้ กรรมวิธีนี้จึงน่าที่จะใช้แทนกรรมวิธีค่าศูนย์กลางได้ดีที่สุด

โครงสร้างพื้นฐานของระบบฟัซซี่ลอจิกแสดงดังรูป



รูปที่ 4 โครงสร้างพื้นฐานของระบบฟัซซี่ลอจิก

สำหรับ project ที่นำเสนอจะใช้หลักการ Deffuzzifier กรรมวิธีค่าศูนย์กลาง ซึ่งระบบควบคุมป้อนกลับสามารถแสดงดังรูป



รูปที่ 5 โครงสร้างพื้นฐานของระบบควบคุมป้อนกลับด้วย Fuzzy Controller

หลักการหา Fuzzy Control Rule

ในการออกแบบ fuzzy controller นั้น ปัญหาสำคัญอันหนึ่งคือการหา fuzzy control rule ที่เหมาะสม ซึ่งวิธีที่สามารถนำมาใช้ได้จริงในงาน process control อยู่ 3 วิธีด้วยกัน คือ

1. จากความรู้ และประสบการณ์ของผู้เชี่ยวชาญ

Fuzzy controller ส่วนมากจะได้รับการออกแบบขึ้นโดยอ้างอิงกับ ความรู้และประสบการณ์ของวิศวกรควบคุม ซึ่งอันที่จริงแล้วก็สามารถกล่าวได้ว่า Fuzzy control นั้นเป็น application ที่ใช้ได้จริงอันแรกทางด้านระบบผู้เชี่ยวชาญ (Expert system) เลยทีเดียว

การออกแบบด้วยวิธีนี้เหมาะกับระบบที่ผู้ควบคุม ควบคุมเป็นหลักสำคัญ Control rule นั้นก็จะได้มาจากการ list วิธีการควบคุมต่างๆ โดยวิศวกรควบคุม แต่ข้อเสียของวิธีนี้ก็คือ ส่วนมากผู้ควบคุมจะไม่สามารถถ่ายทอดความรู้ที่มีออกมาได้เป็นลายลักษณ์อักษร โดยเฉพาะเมื่อระบบมีความซับซ้อนมาก

2. จากการจำลองลักษณะการทำงานของผู้ควบคุม

เมื่อทักษะของผู้ควบคุมเป็นสิ่งสำคัญ จึงจำเป็นมากที่จะหา Fuzzy control rule มาจากการจำลองลักษณะการทำงานของผู้ควบคุม. ซึ่งวิธีนี้สามารถเป็นไปได้ที่จะแปลงการทำงานของผู้ควบคุมมาให้เป็น input และ output ของ controller.

วิธีนี้ก็จะค่อนข้างคล้ายคลึงกับการจำลองกระบวนการ แต่ว่าการจำลองการทำงานของผู้ควบคุมนั้นจะง่ายกว่าการจำลองกระบวนการ เพราะ input ของระบบจะหาได้ง่ายกว่า แต่ในสถานการณ์จริงนั้นก็ควรจะรวมวิธีที่ 1 และ 2 เข้าด้วยกัน

3. จากการจำลองกระบวนการ

ในวิธีแรกนั้นจะมีพื้นฐานมาจากความคิดคร่าวๆ ที่เกี่ยวกับคุณลักษณะของกระบวนการ เช่น output เพิ่มขึ้นเมื่อ input เพิ่มขึ้น, กระบวนการที่มี timelag ฯลฯ. ส่วนวิธีที่ 2 จะใช้เฉพาะตัวแปรที่มีให้แก่ผู้ควบคุมกระบวนการ ซึ่งทั้ง 2 วิธีนี้จะใช้ได้ดีเฉพาะในกรณีที่ผู้ควบคุมมีบทบาทสำคัญในการควบคุมกระบวนการเท่านั้น

แต่ถ้าไม่ต้องการขึ้นอยู่กับผู้ควบคุมและต้องการให้ผลการควบคุมดีกว่าการควบคุมด้วยผู้ควบคุม ก็จะมีการออกแบบอีกวิธีหนึ่งคือ การจำลองกระบวนการ ซึ่งจะเป็นวิธีที่ซับซ้อนกว่า ซึ่งการออกแบบด้วยวิธีนี้ได้มีการศึกษาวิจัยมาหลายครั้งด้วยกัน. การจำลองกระบวนการในที่นี้จะเป็นการแสดงคุณลักษณะของ process ออกมาด้วย fuzzy set โดยพิจารณาจาก input,ตัวแปรสถานะ และ output

มีแนวความคิดสองทางด้วยกันในการออกแบบ Fuzzy controller จากแบบจำลอง fuzzy

-ทางแรก คือ การออกแบบให้ Control rule นั้น ทำการ compensate ลักษณะที่ไม่ต้องการของกระบวนการเพื่อให้ได้ตามเป้าหมายที่ต้องการ

-ทางที่สอง คือ เป็นไปตามทฤษฎีของ Optimal Control ซึ่งจะให้โครงสร้างและ parameter ของ control rule ทำให้ระบบที่มี Fuzzy controller ควบคุมอยู่นั้นเป็นไปตามเป้าหมายที่ต้องการ.

2.5 ทฤษฎีการสุ่มสัญญาณ (Sampling Theorem)

ทฤษฎีการสุ่มสัญญาณของแชนนอน (Shannon's Sampling theorem) กล่าวว่า

If ω_s , defined as $2\pi/T$, where T is the samping peroid ,is greater than $2\omega_1$, or

$$\omega_s > 2\omega_1$$

where ω_1 is the highest-frequency component present in the continous-time signal $x(t)$, then the signal $x(t)$ can be reconstructed completely from the sampled signal $x^*(t)$

ในวิทยานิพนธ์ฉบับนี้ เรามีวิธีการเลือกค่า sampling time ดังนี้

จากสมการระบบอันดับสอง (Second order system)

$$G(s) = \frac{\omega_n}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

และ $$\omega_d = \omega_n\sqrt{1-\zeta^2}$$

ซึ่งโดยปกติแล้วระบบทั่วไปจะมีค่า ω_1 ไม่เกิน ω_d

และเพื่อความถูกต้องเราจะไม่เลือกค่า ω_s เพียงแค่ 2 เท่าของ ω_1

แต่เราจะเลือกใช้ ω_s ประมาณ 10 เท่าของ ω_1

$$\omega_s \approx 10\omega_d$$

จาก Folding frequency or Nyquist frequency

$$\omega_N = \frac{1}{2}\omega_s = \frac{\pi}{T}$$

ดังนั้นเราสามารถเลือกค่า sampling time ที่เหมาะสมจาก

$$T \le \left[\frac{2\pi}{\omega_s} = \frac{2\pi}{10\omega_d} = \frac{2\pi}{10\omega_n\sqrt{1-\zeta^2}} \right]$$

ตัวอย่างเช่น

ระบบอันดับสอง (Second order system) $$G(s) = \frac{1}{s^2 + s + 1}$$

จะได้ว่า ω_n=1 และ ζ=0.5

ดังนั้นเราควรจะใช้ค่า sampling time $$T \le \frac{2\pi}{(10)\sqrt{1-0.5^2}}$$

$$T \le 0.7255 \text{ sec.}$$

บทที่ 3
โปรแกรมจำลองการทำงานของระบบควบคุมป้อนกลับด้วย FUZZY CONTROLLER

วัตถุประสงค์

- แสดงผลตอบสนองของระบบต่าง ๆที่ได้ถูกกำหนดโดยผู้ใช้โดยเลือกใช้ CONTROLLER มาควบคุมได้ 2 ชนิด คือ

~ FUZZY CONTROLLER

~ PID CONTROLLER

- สามารถเปรียบเทียบผลการตอบสนองระหว่าง FUZZY CONTROLLER กับ PID ได้ รวมทั้งเปรียบเทียบกับระบบป้อนกลับเดิมที่ไม่ได้ผ่าน CONTROLLER ได้ด้วย

- สามารถปรับค่า CONTROL RULE และ MEMBERSHIP FUNCTION ได้อย่างสะดวก

ALGORITHM ของ FUZZY CONTROLLER

1. รับค่า MEMBERSHIP FUNCTION โดยเรียกจาก file *. mem ที่มีอยู่หรือจะแก้จาก default ที่มีอยู่ก็ได้รับค่าสูงสุดของ error , change of error และ control input เพื่อกำหนด range ของ error,change of error และ control input สำหรับ coarse และ fine membership function

2. รับค่า rule โดยเรียกจาก file *. rul ที่มีอยู่หรือจะแก้ไขจาก default ที่มีอยู่ก็ได้

3. นำค่าจาก membership function และ rule ที่มีอยู่มาสร้างเป็น Fuzzy controller coarse table และ Fine table ซึ่งได้จากการนำ error และ change of error จากตาราง membership function มา inference กัน และ defuzzification โดยการหา center of gravity ออกมาเป็น control input แล้วนำมาบรรจุไว้ในตาราง Fuzzy controller coarse table และ Fuzzy controller fine table ดังตัวอย่าง

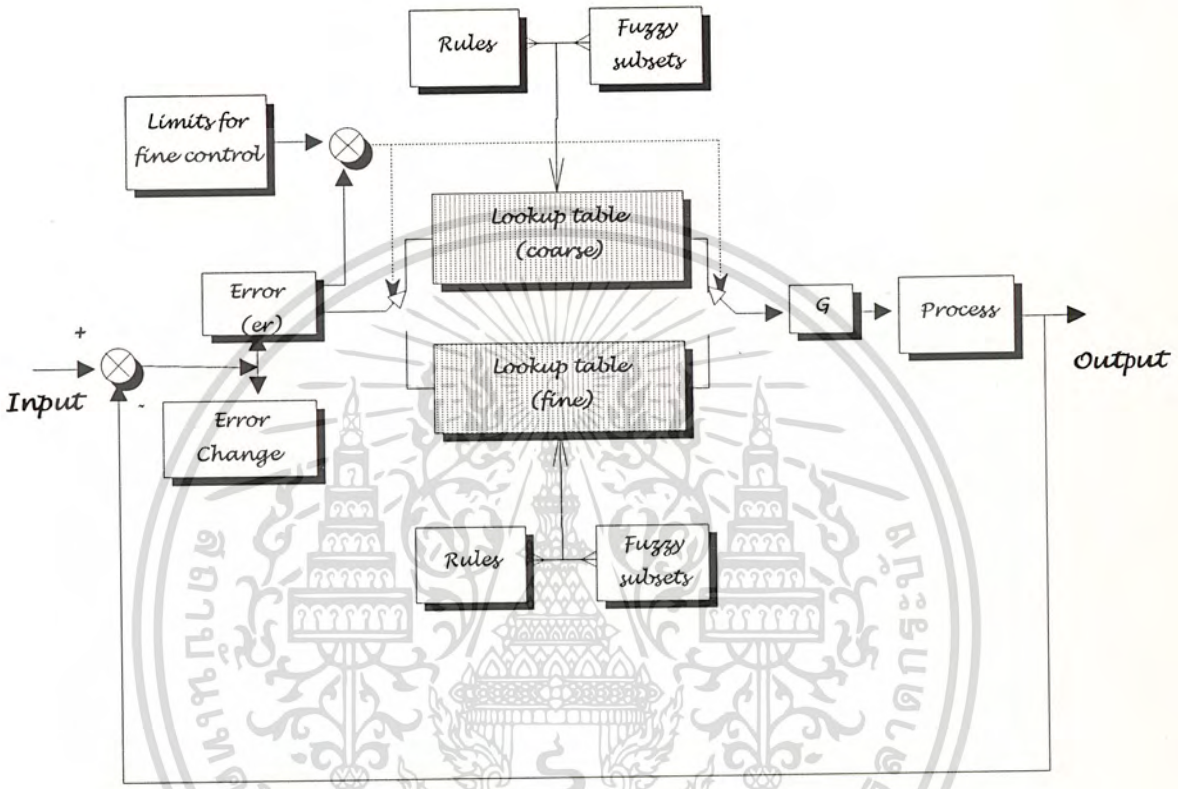
Change of error

error	-5	-4	-3	-2	-1	0	1	2	3	4	5
-5	-5	-5	-4	-4	-3	-1	0	1	1	2	2
-4	-5	-5	-5	-4	-3	-1	0	1	1	2	2
-3	-5	-5	-5	-4	-3	-1	0	1	2	2	2
-2	-5	-5	-4	-4	-3	-1	1	2	3	3	3
-1	-5	-5	-4	-3	-2	0	1	2	3	3	4
0	-5	-4	-4	-2	-2	0	1	3	4	4	4
1	-4	-4	-3	-2	-1	0	2	3	4	4	4
2	-4	-4	-3	-2	-1	1	2	3	5	4	5
3	-3	-4	-2	-1	0	1	2	3	5	4	5
4	-2	-3	-2	-1	0	1	3	3	5	5	5
5	-2	-3	-1	-1	0	1	3	3	5	5	5

LOOKUP TABLE

4. รับค่า error และ change of error จากกระบวนการมา แล้วนำมาเทียบกับค่าในตาราง Fuzzy controller coarse table ก็จะได้ค่า control input ที่เหมาะสม จากนั้นจึงส่งออกไปให้เป็น input ของ process และเมื่อ error มีค่าต่ำกว่าขอบเขตของ fine error ก็ให้ switch มาใช้ตาราง Fuzzy controller fine table

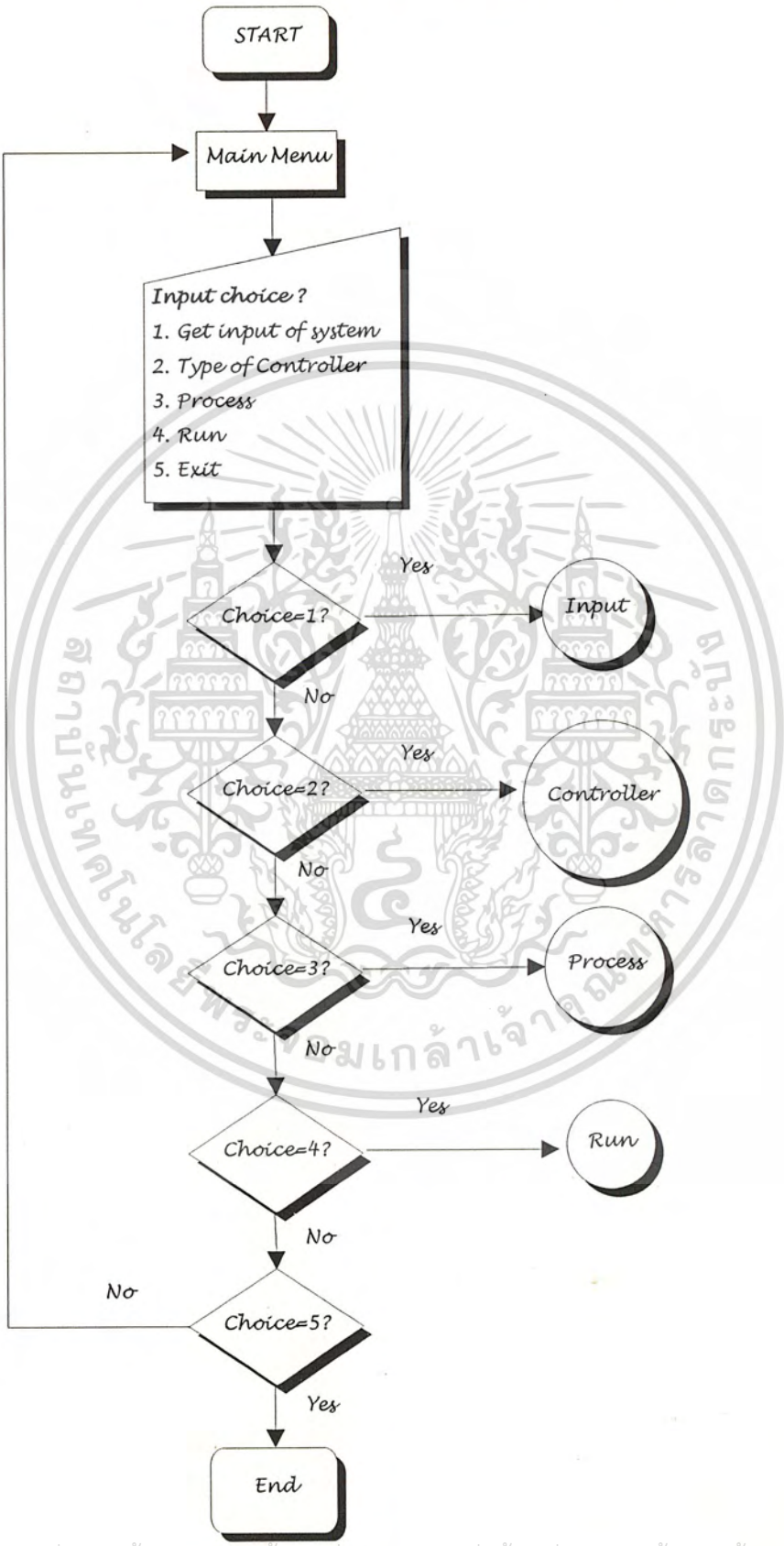
Block diagram แสดงการทำงานแสดงดังรูป

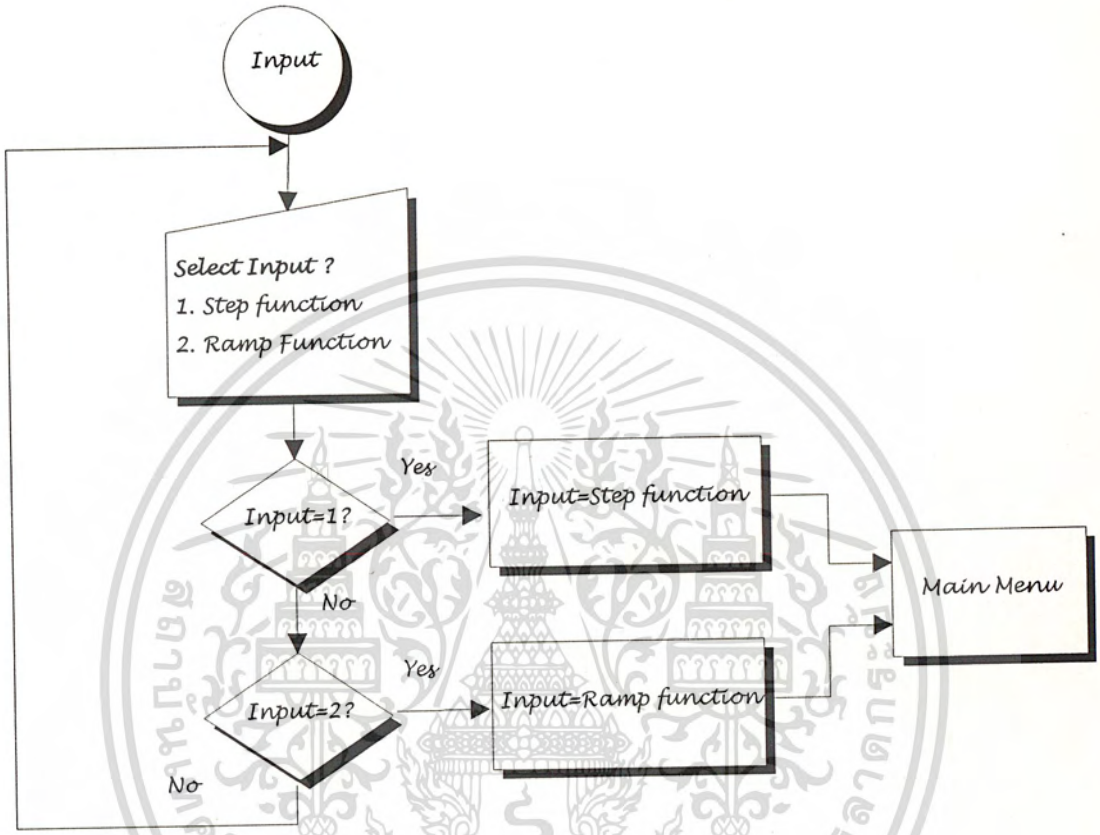


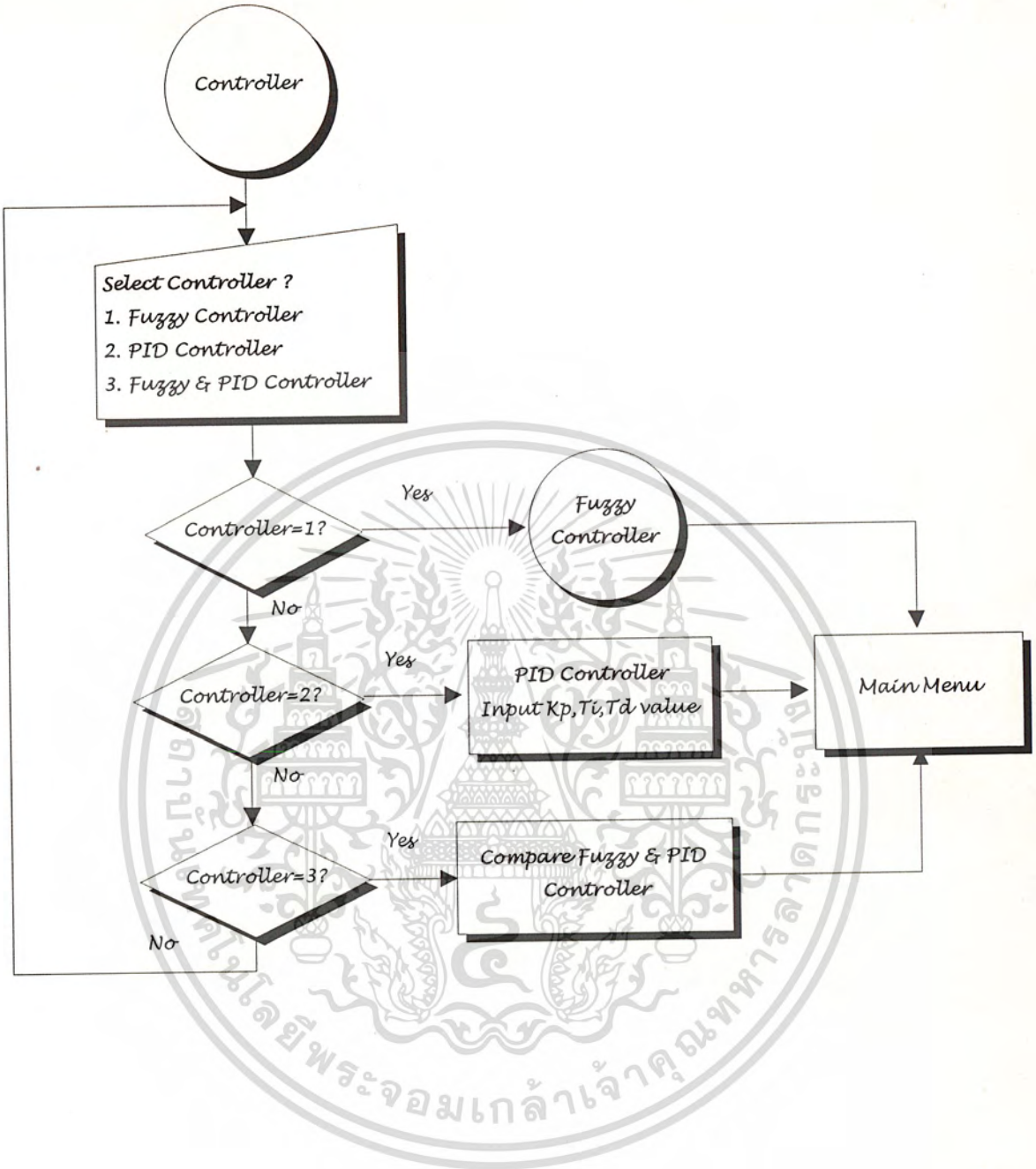
รูปที่ 6 Block diagram แสดงการทำงานการควบคุมด้วย Fuzzy Controller

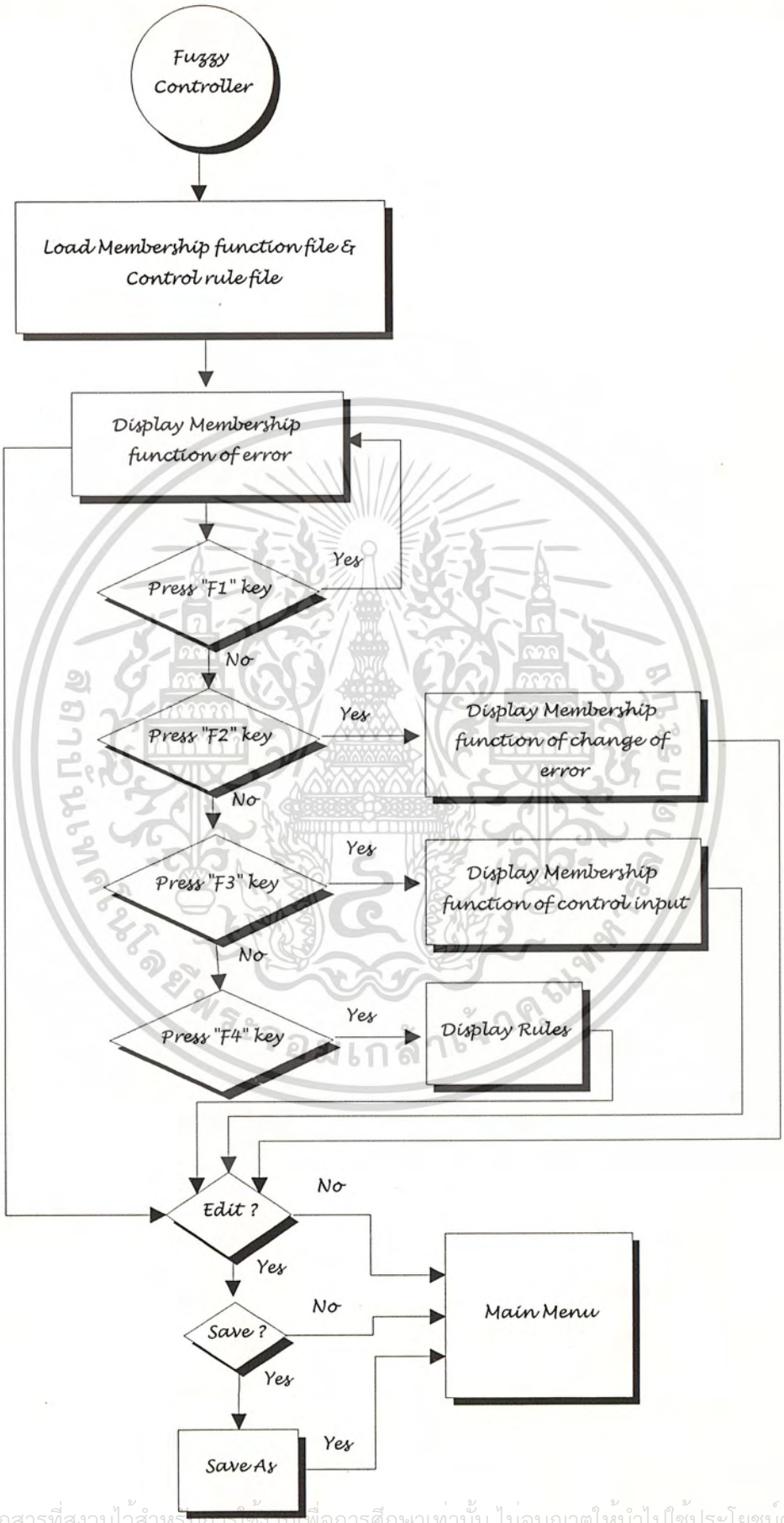
หมายเหตุ : โปรแกรมที่ได้พัฒนาขึ้นมานี้นับได้ว่าเป็นโปรแกรมที่ทำให้สามารถแก้ไข rule และ membership function ได้สะดวกและเห็นภาพได้ชัดเจนแต่ยังมีส่วนที่ต้องปรับปรุงอีกหลายส่วน เช่น ควรให้มีการยืดหยุ่นstate(LN,LP,...etc.) ได้มากกว่านี้และควรให้มีการเพิ่มจำนวน input ของ Fuzzy controller ให้ได้มากกว่า 2 input

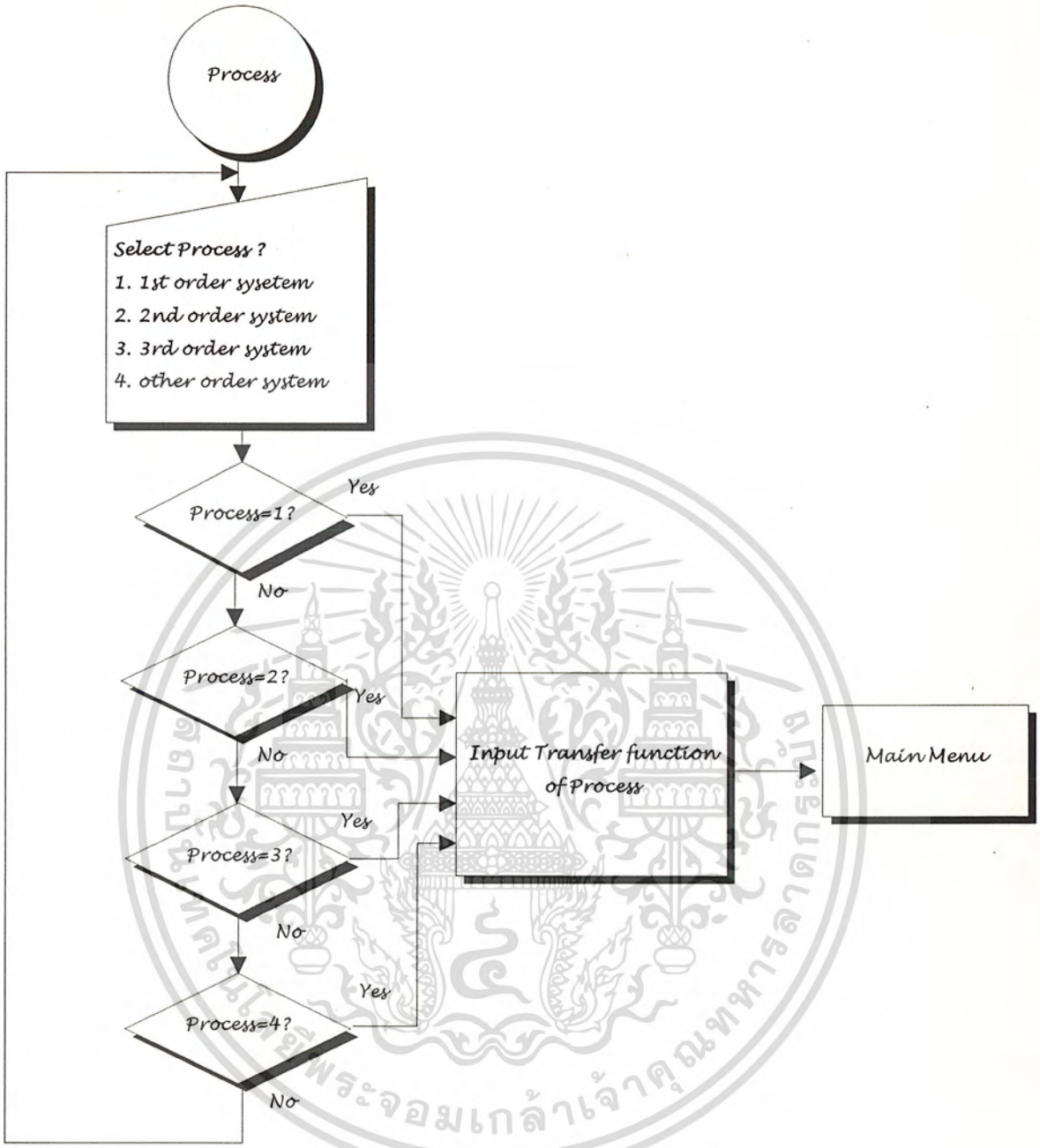
FLOWCHART แสดงการทำงานของโปรแกรมจำลองการทำงาน
ของระบบควบคุมป้อนกลับด้วย FUZZY CONTROLLER

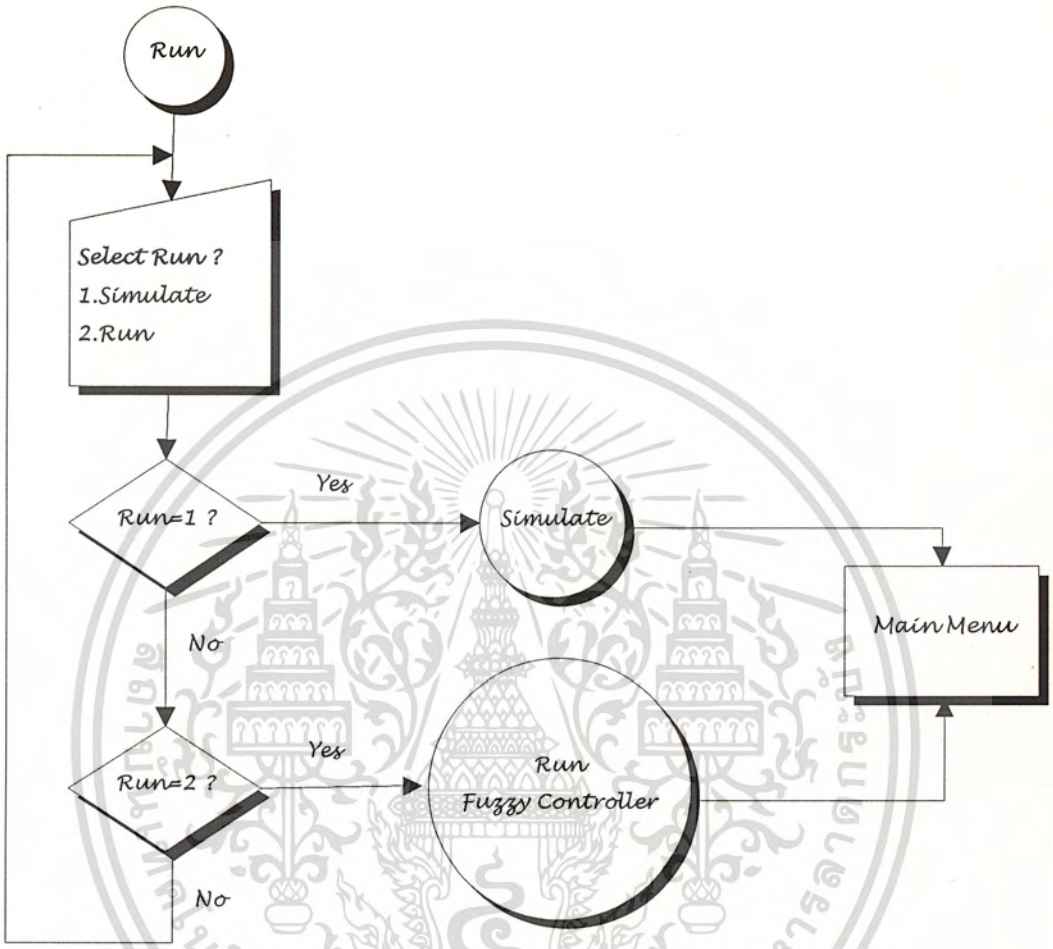


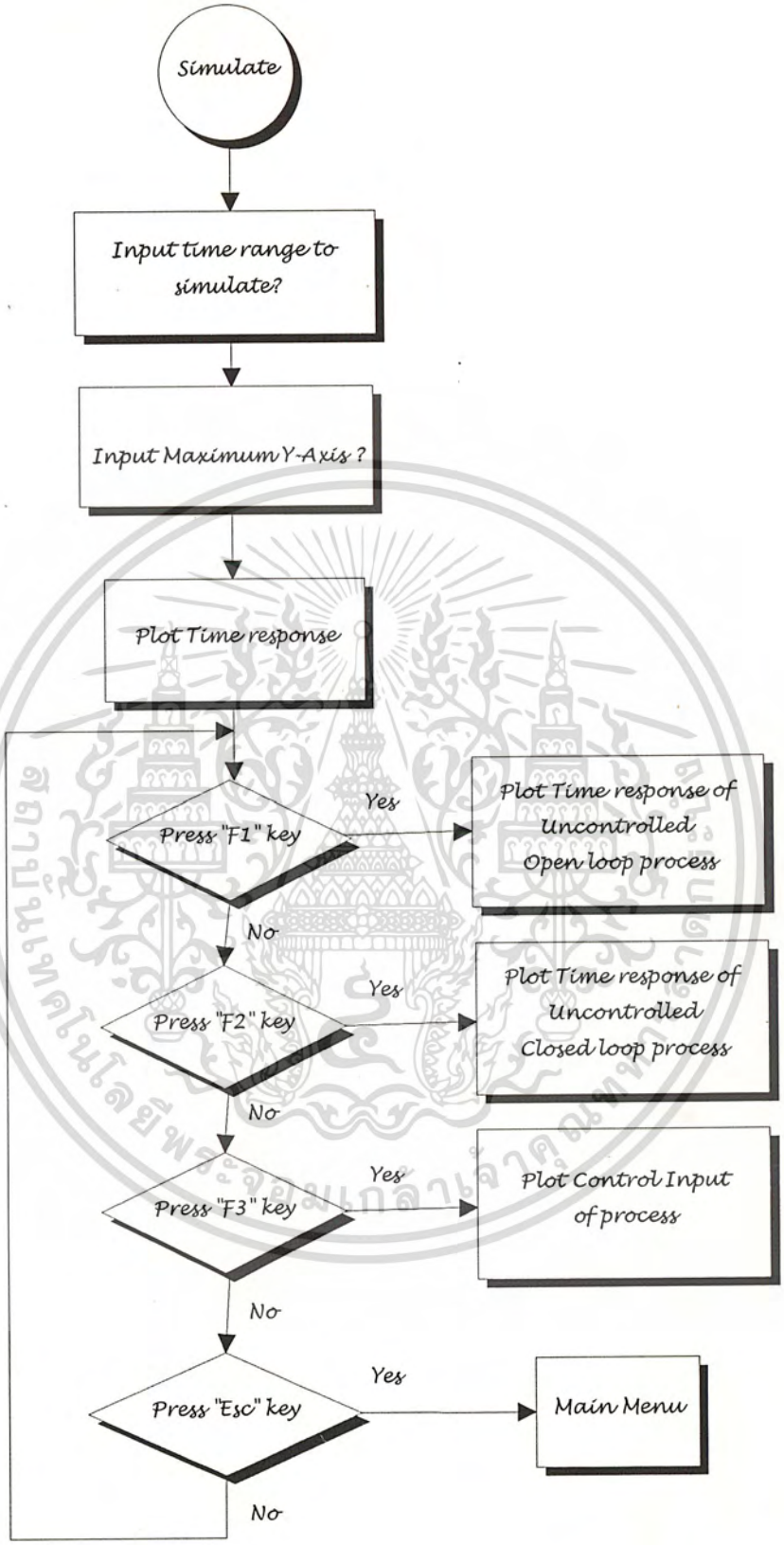


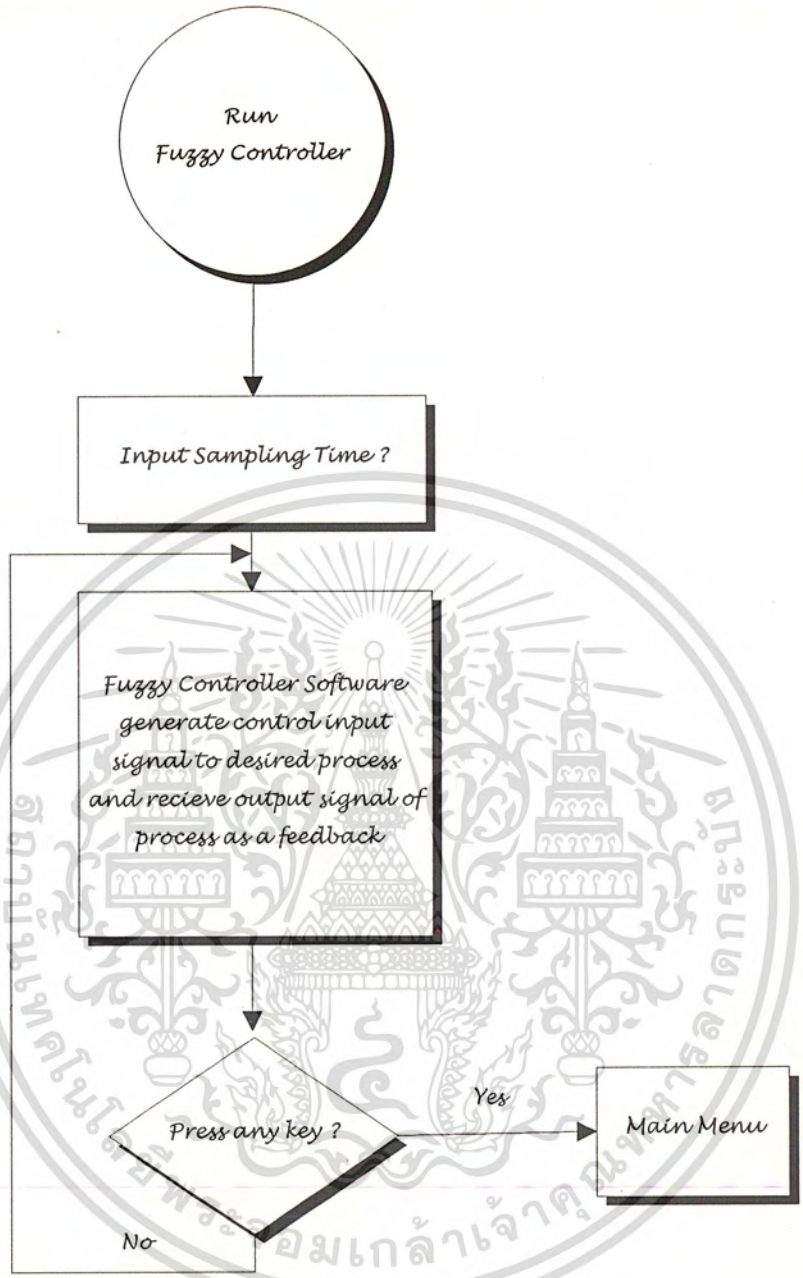












บทที่ 4
การทดลอง

4.1 การทดลองที่ 1

Transfer function ของ process คือ $\frac{1}{s^2+s+1}$

Pole	-0.5±0.87j
Zero	-

Open loop control system

Overshoot	Rise Time	S.S. Offset	S.S. Osc
1.15	2.3	0	No

หมายเหตุ ในทุกการทดลอง S.S. Offset หมายถึง Steady state offset
และ S.S. Osc หมายถึง การแกว่ง (Oscillate) ที่ Steady state

1. ปรับค่า Fuzzy controller ได้เป็นขั้นตอนดังนี้

1.1 เริ่มจากการใช้ค่าที่กำหนดได้เป็น DEFAULT

coarse membership function	DEFAULT
fine membership function	DEFAULT
control rule	DEFAULT

จากผลตอบสนองที่ได้พบว่าในช่วง transientจะมี overshoot ที่ค่อนข้างสูง และ ในช่วง steady state ก็จะไม่มีการสั่นของผลตอบสนองทั้งยังเข้าสู่ setpoint ได้อย่างถูกต้องโดยไม่มี offset

1.2 ทำการปรับปรุงโดยการปรับปรุง control rule จะได้ผลการทดลองดังแสดงในตาราง

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
Default	1.5	1.2	1.27	1.6	0.01	Small
		1.05	1.27	1.6	0	No
	1.3	1.1	1.24	1.7	0	No
		1.05	1.24	1.7	0	No
	1.1	1.05	1.2	2	0	No
		1.01	1.2	2	0	No
tst1	1.5	1.2	1.23	1.7	0.01	Small
		1.05	1.23	1.7	0	No
	1.3	1.1	1.21	1.9	0	No
		1.05	1.21	1.9	0	No
	1.1	1.05	1.18	2.1	0	No
		1.01	1.18	2.1	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S Offset	S.S. Osc
tst2	1.5	1.2	1.27	1.6	0.01	Small
		1.05	1.27	1.6	0	No
	1.3	1.1	1.25	1.7	0	No
		1.05	1.25	1.7	0	No
	1.1	1.05	1.2	2	0	No
		1.01	1.2	2	0	No
tst3	1.5	1.2	1.27	1.5	0.01	Small
		1.05	1.27	1.5	0.01	Very Small
	1.3	1.1	1.25	1.7	0.01	Small
		1.05	1.25	1.7	0.01	Very Small
	1.1	1.05	1.2	2	0.01	Very Small
		1.01	1.2	2	0	No
tst4	1.5	1.2	1.05	3.7	0.04	Small
		1.05	1.05	3.7	0	No
	1.3	1.1	1.1	2.8	0	No
		1.05	1.1	2.8	0	No
	1.1	1.05	1.13	2.4	0	No
		1.01	1.13	2.4	0	No
tst5	1.5	1.2	1.15	2.2	0.01	Small
		1.05	1.15	2.2	0.01	Very Small
	1.3	1.1	1.15	2.25	0.01	Small
		1.05	1.15	2.25	0.01	Very Small
	1.1	1.05	1.15	2.25	0.01	Very Small
		1.01	1.15	2.3	0	No
tst6	1.5	1.2	1.27	1.6	-0.03	Medium
		1.05	1.27	1.6	-0.02	No
	1.3	1.1	1.25	1.7	-0.03	Very Small
		1.05	1.25	1.7	-0.02	No
	1.1	1.05	1.21	2.1	-0.02	No
		1.01	1.21	2.1	0	No
tst7	1.5	1.2	1.15	2.25	0	No
		1.05	1.15	2.25	0	No
	1.3	1.1	1.15	2.3	0	No
		1.05	1.15	2.3	0	No
	1.1	1.05	1.15	2.3	0	No
		1.01	1.15	2.3	0	No
tst8	1.5	1.2	1.25	1.7	0.01	Small
		1.05	1.25	1.7	0	No
	1.3	1.1	1.21	1.9	0	No
		1.05	1.21	1.9	0	No
	1.1	1.05	1.2	2.1	0	No
		1.01	1.2	2.1	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
tst9	1.5	1.2	1.26	1.5	0.01	Medium
		1.05	1.26	1.5	0.01	Small
	1.3	1.1	1.24	1.7	0.01	Small
		1.05	1.24	1.7	0.01	Small
	1.1	1.05	1.21	2	0.01	Small
		1.01	1.21	2	0	No
t10	1.5	1.2	1.16	2.2	0	No
		1.05	1.16	2.2	0	No
	1.3	1.1	1.15	2.25	0	No
		1.05	1.15	2.25	0	No
	1.1	1.05	1.15	2.3	0	No
		1.01	1.15	2.3	0	No
t11	1.5	1.2	1.16	2.2	0	No
		1.05	1.16	2.2	0	No
	1.3	1.1	1.15	2.25	0	No
		1.05	1.15	2.25	0	No
	1.1	1.05	1.15	2.3	0	No
		1.01	1.15	2.3	0	No

จะพบว่าเมื่อใช้ control rule จาก file tst4.rul จะทำให้ผลตอบสนองเชิงเวลาดีกว่า control rule อื่นๆ โดยมี overshoot ลดลงแต่ก็ทำให้ rise time มีค่ามากขึ้น

1.3 ทำการปรับปรุง membership function โดยใช้ membership function ที่เก็บไว้ใน file ชื่อtest.mem และ ใช้ control rule ซึ่งได้เก็บไว้ใน file ชื่อ tst4.rul

coarse membership function	test.mem
fine membership function	test.mem
control rule	tst4.rul

จากผลการทดลองพบว่าผลตอบสนองจะมีลักษณะเป็นที่น่าพอใจ คือ มี overshoot ขนาดเล็กลง แต่ก็ทำให้ rise time มีค่ามากขึ้น และผลตอบสนองที่ steady state ก็ค่อนข้างที่จะเรียบขึ้น ซึ่งเมื่อเปรียบเทียบกับการทดลองโดยใช้ control rule จาก file tst4.rul และ membership function จาก file default.mem แล้วจะเห็นได้ว่าผลตอบสนองมี maximum overshoot ลดลงเพียงเล็กน้อยแต่ rise time เพิ่มขึ้นพอสมควร เราจึงเลือก membership function จาก file default.mem และ control rule จาก file tst4.rul เป็น membership function และ control rule ของ fuzzy controller ที่เหมาะสม

การเปลี่ยนแปลงกฎ (rule) จะเห็นได้ชัดเจนกว่าการเปลี่ยนแปลงที่เกิดขึ้นจากการปรับเปลี่ยน membership function ดังนั้นในการทดลองต่อไปเราจะไม่พิจารณาผลตอบสนองจากการเปลี่ยนค่า membership function

2. หาค่า parameter ของ PID controller โดยวิธีของ Ziegler-Nichols

ซึ่งจะได้ค่า parameter ดังนี้คือ

Kp	Ti	Td
5.7	0.8	0.2

จะได้ค่าผลตอบสนองเชิงเวลา (Time response) แสดงดังตาราง

Overshoot	Rise time (sec.)	S.S. Offset	S.S. Osc
1.43	0.55	0	No

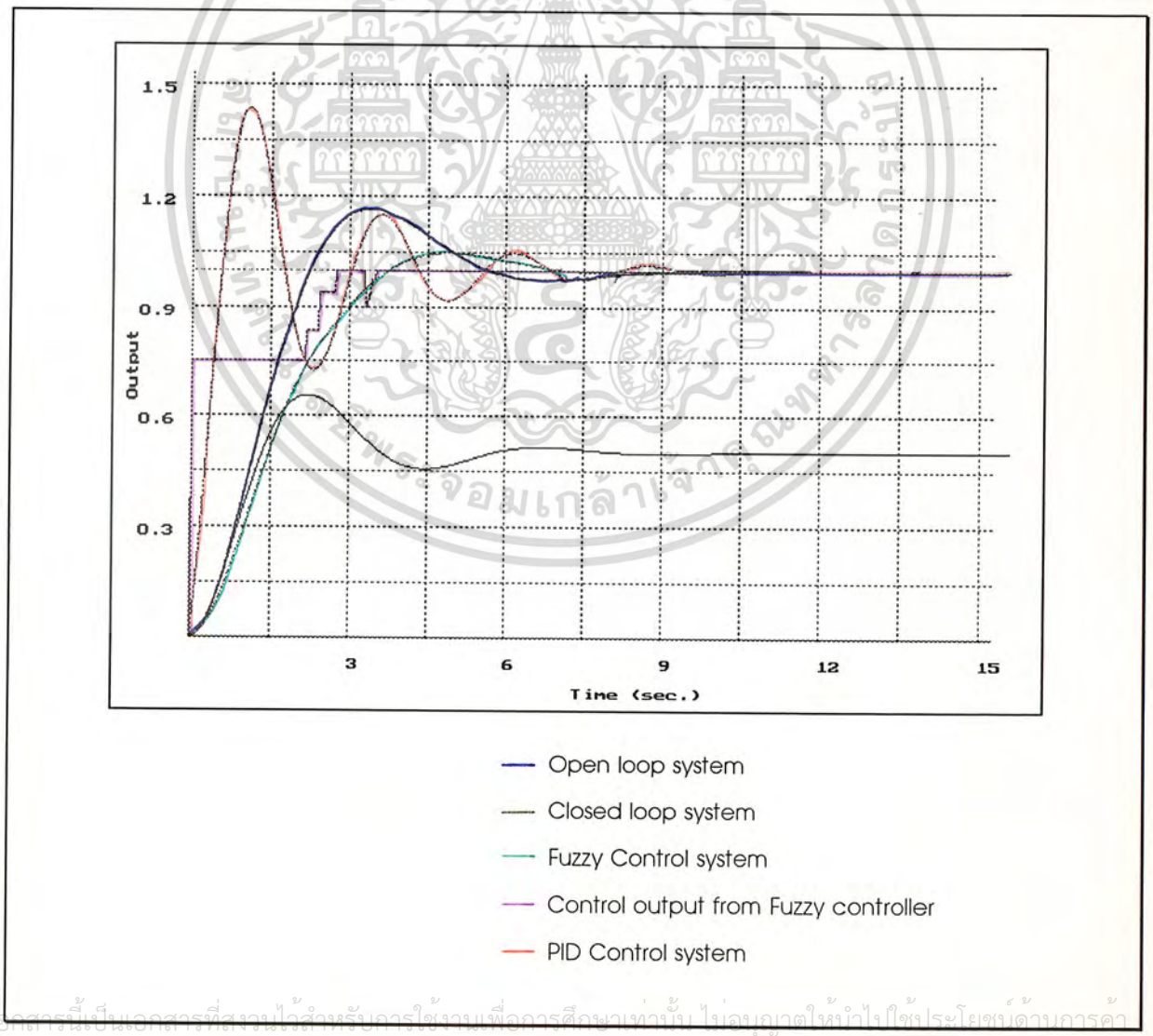
จากผลการทดลองจะเห็นว่าผลตอบสนองเป็นที่น่าพอใจ แม้ว่าจะมี overshoot ค่อนข้างสูง แต่ค่า rise time ลดลงมากและที่ steady state ก็เข้าสู่ setpoint ได้เป็นอย่างดี

3. เปรียบเทียบ Fuzzy controller ที่ดีที่สุด (จาก 1.3) กับ PID controller (จาก 2)

จะเห็นได้ว่า Fuzzy controller สำหรับระบบนี้จะค่อนข้างดีกว่า เนื่องจากทำให้เกิด overshoot น้อย และเข้าสู่เสถียรได้เร็วกว่า PID controller

Time response ของ การทดลองที่ 1

coarse membership function	default.mem
fine membership function	default.mem
control rule	tst4.rul
coarse control input	1.50
fine control input	1.05



4.2 การทดลองที่ 2

Transfer function ของ process คือ $\frac{1}{s^2+0.5s+1}$

Pole	-0.25±0.97j
Zero	-

Open loop control system

Overshoot	Rise Time	S.S. Offset	S.S. Osc
1.44	1.9	0	No

1. ปรับค่า Fuzzy controller ได้เป็นขั้นตอนดังนี้

1.1 เริ่มจากการใช้ค่าที่กำหนดได้เป็น DEFAULT

coarse membership function	DEFAULT
fine membership function	DEFAULT
control rule	DEFAULT

จากผลตอบสนองที่ได้พบว่าในช่วง transient จะมี overshoot สูง และ ในช่วง steady state ก็จะไม่มีการสั่นของผลตอบสนองทั้งยังเข้าสู่ setpoint ได้อย่างถูกต้องโดยไม่มี offset

1.2 ทำการปรับปรุงโดยการปรับปรุง control rule จะได้ผลการทดลองดังแสดงในตาราง

coarse membership function	DEFAULT
fine membership function	DEFAULT

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
Default	1.5	1.2	1.57	1.2	0	No
		1.05	1.57	1.2	0	No
	1.3	1.1	1.53	1.5	0	No
		1.05	1.53	1.5	0.01	No
	1.1	1.05	1.48	1.7	0	No
		1.01	1.48	1.7	0	No
tst1	1.5	1.2	1.49	1.5	0	No
		1.05	1.49	1.5	0	No
	1.3	1.1	1.48	1.6	0	No
		1.05	1.48	1.6	0.01	No
	1.1	1.05	1.45	1.7	0	No
		1.01	1.45	1.7	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
tst2	1.5	1.2	1.56	1.3	0	No
		1.05	1.56	1.3	0	No
	1.3	1.1	1.54	1.4	0	No
		1.05	1.54	1.4	0	No
	1.1	1.05	1.48	1.6	0	No
		1.01	1.48	1.6	0	No
tst3	1.5	1.2	1.57	1.35	0.01	Large
		1.05	1.57	1.35	0.01	Small
	1.3	1.1	1.53	1.5	0.01	Medium
		1.05	1.53	1.5	0.01	Small
	1.1	1.05	1.48	1.65	0.01	Medium
		1.01	1.48	1.65	0	No
tst4	1.5	1.2	1.28	2.3	0	No
		1.05	1.28	2.3	0	No
	1.3	1.1	1.35	2.0	0	No
		1.05	1.35	2.0	0	No
	1.1	1.05	1.42	1.8	0	No
		1.01	1.42	1.8	0	No
tst5	1.5	1.2	1.39	1.8	0.01	Large
		1.05	1.39	1.8	0.01	Small
	1.3	1.1	1.40	1.8	0.01	Medium
		1.05	1.40	1.8	0.01	Small
	1.1	1.05	1.43	1.8	0.01	Small
		1.01	1.43	1.8	0	No
tst6	1.5	1.2	1.94	1.3	0.42	No
		1.05	1.94	1.3	0.42	No
	1.3	1.1	1.73	1.5	-0.15	Medium
		1.05	1.73	1.5	-0.15	Medium
	1.1	1.05	1.54	1.7	-0.02	No
		1.01	1.54	1.7	0	No
tst7	1.5	1.2	1.44	1.8	0	No
		1.05	1.44	1.8	0	No
	1.3	1.1	1.44	1.8	0	No
		1.05	1.44	1.8	0	No
	1.1	1.05	1.44	1.8	0	No
		1.01	1.44	1.8	0	No
tst8	1.5	1.2	1.58	1.5	0.01	Large
		1.05	1.58	1.5	0	No
	1.3	1.1	1.53	1.6	0.01	Small
		1.05	1.53	1.6	0	No
	1.1	1.05	1.48	1.7	0	No
		1.01	1.48	1.7	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S Offset	S.S. Osc
tst9	1.5	1.2	1.52	1.3	0.01	Large
		1.05	1.52	1.3	0.01	Medium
	1.3	1.1	1.50	1.5	0.01	Large
		1.05	1.50	1.5	0.01	Medium
	1.1	1.05	1.47	1.7	0.01	Medium
		1.01	1.47	1.7	0	No
t10	1.5	1.2	1.38	1.8	0	No
		1.05	1.38	1.8	0.01	No
	1.3	1.1	1.40	1.8	0.01	Small
		1.05	1.40	1.8	0	No
	1.1	1.05	1.43	1.8	0	No
		1.01	1.43	1.8	0	No
t11	1.5	1.2	1.38	1.8	0	No
		1.05	1.38	1.8	0.01	No
	1.3	1.1	1.40	1.8	0.01	Small
		1.05	1.40	1.8	0	No
	1.1	1.05	1.43	1.8	0	No
		1.01	1.43	1.8	0	No

จะพบว่าเมื่อใช้ control rule จาก file tst4.rul จะทำให้ผลตอบสนองเชิงเวลาดีกว่า control rule อื่นๆ โดยมี overshoot ลดลงแต่ก็ทำให้ rise time มีค่ามากขึ้น

2. หาค่า parameter ของ PID controller โดยวิธีของ Ziegler-Nichols

ซึ่งจะได้ค่า parameter ดังนี้คือ

Kp	Ti	Td
4.08	1.0	0.25

จะได้ค่าผลตอบสนองเชิงเวลา (Time response) แสดงดังตาราง

Overshoot	Rise time (sec.)	S.S. Offset	S.S. Osc
1.41	0.7	0	No

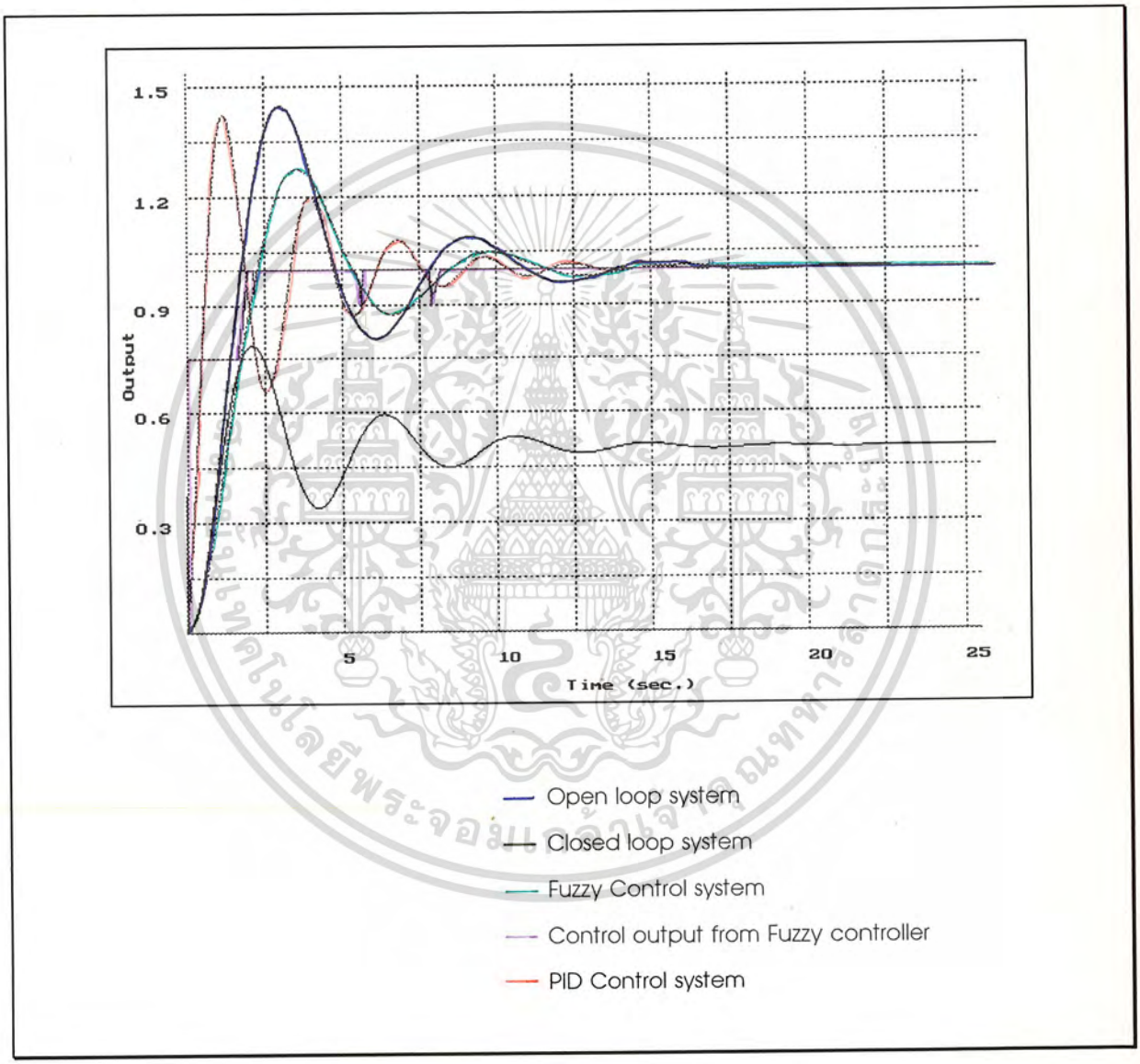
จากผลการทดลองจะเห็นว่าผลตอบสนองเป็นที่น่าพอใจ แม้ว่าจะมี overshoot ค่อนข้างสูง แต่ที่ steady state ก็เข้าสู่ setpoint ได้เป็นอย่างดี

3. เปรียบเทียบ Fuzzy controller ที่ดีที่สุด (จาก 1.3) กับ PID controller (จาก 2)

จะเห็นได้ว่า Fuzzy controller สำหรับระบบนี้จะค่อนข้างดีกว่า เนื่องจากทำให้เกิด overshoot น้อย และเข้าสู่เสถียรได้เร็วกว่า PID controller

Time response ของ การทดลองที่ 2

coarse membership function	default.mem
fine membership function	default.mem
control rule	tst4.rul
coarse control input	1.50
fine control input	1.05



4.3 การทดลองที่ 3

Transfer function ของ process คือ $\frac{1}{s^2+0.2s+1}$

Pole	-0.12±0.99j
Zero	-

Open loop control system

Overshoot	Rise Time	S.S. Offset	S.S. Osc
1.74	1.6	0	No

1. ปรับค่า Fuzzy controller ได้เป็นขั้นตอนดังนี้

1.1 เริ่มจากการใช้ค่าที่กำหนดได้เป็น DEFAULT

coarse membership function	DEFAULT
fine membership function	DEFAULT
control rule	DEFAULT

จากผลตอบสนองที่ได้พบว่าในช่วง transient จะมี overshoot สูงมาก และ ในช่วง steady state ก็จะไม่มีการสั่นของผลตอบสนองทั้งยังเข้าสู่ setpoint ได้อย่างถูกต้องโดยไม่มี offset

1.2 ทำการปรับปรุงโดยการปรับปรุง control rule จะได้ผลการทดลองดังแสดงในตาราง

coarse membership function	DEFAULT
fine membership function	DEFAULT

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
Default	1.5	1.2	1.85	1.25	0	Large
		1.05	1.85	1.25	0	Large
	1.3	1.1	1.81	1.3	0	No
		1.05	1.81	1.3	0	No
	1.1	1.05	1.76	1.5	0	No
		1.01	1.76	1.5	0	No
tst1	1.5	1.2	1.75	1.3	0	Large
		1.05	1.75	1.3	0.01	Small
	1.3	1.1	1.74	1.4	0	Large
		1.05	1.74	1.4	0	No
	1.1	1.05	1.73	1.5	0.01	No
		1.01	1.73	1.5	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
tst2	1.5	1.2	1.85	1.2	0	Large
		1.05	1.85	1.2	0	No
	1.3	1.1	1.81	1.3	0.01	Large
		1.05	1.81	1.3	0	No
	1.1	1.05	1.76	1.5	0	No
		1.01	1.76	1.5	0	No
tst3	1.5	1.2	1.86	1.2	0	Large
		1.05	1.86	1.2	0	Large
	1.3	1.1	1.82	1.3	0	Large
		1.05	1.82	1.3	0	Medium
	1.1	1.05	1.76	1.5	0	Medium
		1.01	1.76	1.5	0	No
tst4	1.5	1.2	1.53	2.0	0	No
		1.05	1.53	2.0	0	No
	1.3	1.1	1.62	1.8	0	No
		1.05	1.62	1.8	0	No
	1.1	1.05	1.70	1.6	0	No
		1.01	1.70	1.6	0	No
tst5	1.5	1.2	1.62	1.6	0	Large
		1.05	1.62	1.6	0	Medium
	1.3	1.1	1.66	1.6	0	Large
		1.05	1.66	1.6	0.01	Large
	1.1	1.05	1.70	1.6	0.01	Large
		1.01	1.70	1.6	0	No
tst6	1.5	1.2	2.35	1.2	0.42	No
		1.05	2.35	1.2	0.42	No
	1.3	1.1	2.10	1.3	0.26	No
		1.05	2.10	1.3	0.26	No
	1.1	1.05	1.85	1.5	-0.02	No
		1.01	1.85	1.5	0	No
tst7	1.5	1.2	1.73	1.6	0	Large
		1.05	1.73	1.6	0.01	No
	1.3	1.1	1.73	1.6	0	Large
		1.05	1.73	1.6	0	No
	1.1	1.05	1.73	1.6	0	No
		1.01	1.73	1.6	0	No
tst8	1.5	1.2	1.90	1.3	0	Large
		1.05	1.90	1.3	0	No
	1.3	1.1	1.84	1.4	0	Large
		1.05	1.84	1.4	0	No
	1.1	1.05	1.76	1.6	0.01	No
		1.01	1.76	1.6	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
tst9	1.5	1.2	1.78	1.2	0	Large
		1.05	1.78	1.2	0	Large
	1.3	1.1	1.76	1.3	0	Large
		1.05	1.76	1.3	0	Large
	1.1	1.05	1.74	1.5	0	Large
		1.01	1.74	1.5	0	No
t10	1.5	1.2	1.62	1.6	0	Large
		1.05	1.62	1.6	0.01	No
	1.3	1.1	1.66	1.6	0	Large
		1.05	1.66	1.6	0.01	No
	1.1	1.05	1.70	1.6	0	No
		1.01	1.70	1.6	0	No
t11	1.5	1.2	1.62	1.6	0	Large
		1.05	1.62	1.6	0.01	No
	1.3	1.1	1.66	1.6	0	Large
		1.05	1.66	1.6	0.01	No
	1.1	1.05	1.70	1.6	0	No
		1.01	1.70	1.6	0	No

จะพบว่าเมื่อใช้ control rule จาก file tst4.rul จะทำให้ผลตอบสนองเชิงเวลาดีกว่า control rule อื่นๆ โดยมี overshoot ลดลงแต่ก็ทำให้ rise time มีค่ามากขึ้น

2. หาค่า parameter ของ PID controller โดยวิธีของ Ziegler-Nichols

ซึ่งจะได้ค่า parameter ดังนี้คือ

Kp	Ti	Td
2.2	1.2	0.3

จะได้ค่าผลตอบสนองเชิงเวลา (Time response) แสดงดังตาราง

Overshoot	Rise time (sec.)	S.S. Offset	S.S. Osc
1.38	0.9	0	No

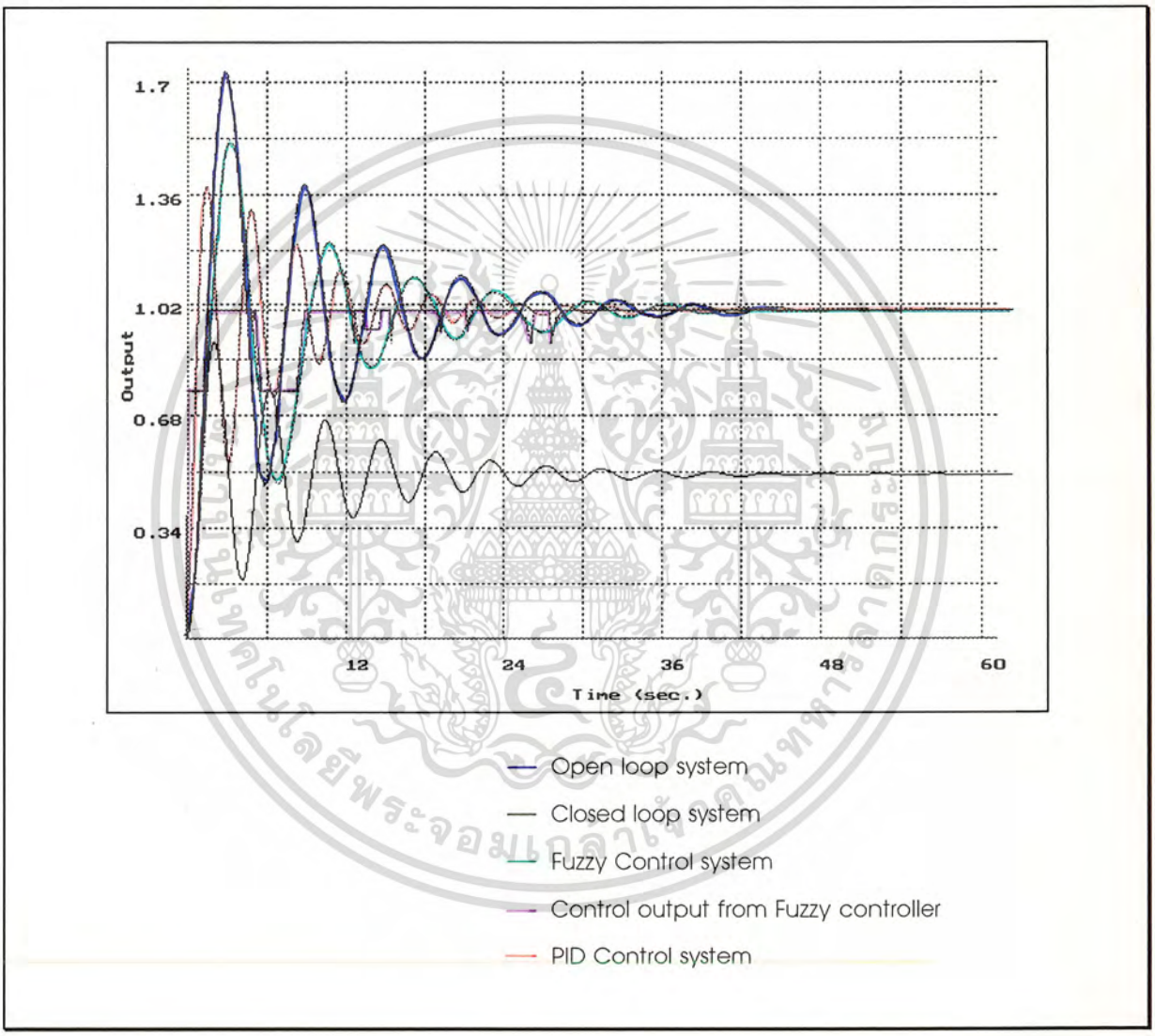
จากผลการทดลองจะเห็นว่าผลตอบสนองเป็นที่น่าพอใจมาก แม้ว่าจะมี overshoot ค่อนข้างสูง แต่ก็นับว่าลดลงจากระบบเดิมได้มากและค่า rise time ยังเร็วขึ้นอีกด้วย นอกจากนี้ที่ steady state ก็เข้าสู่ setpoint ได้เป็นอย่างดี

3. เปรียบเทียบ Fuzzy controller ที่ดีที่สุด (จาก 1) กับ PID controller (จาก 2)

จะเห็นได้ว่า PID controller สำหรับระบบนี้จะค่อนข้างดีกว่า เนื่องจากทำให้เกิด overshoot น้อย และเข้าสู่เสถียรได้เร็วกว่า Fuzzy controller

Time response ของ การทดลองที่ 3

coarse membership function	default.mem
fine membership function	default.mem
control rule	tst4.rul
coarse control input	1.50
fine control input	1.05



4.4 การทดลองที่ 4

Transfer function ของ process คือ $\frac{50}{s^2+10s+50}$

Pole	-5±5j
Zero	-

Open loop control system

Overshoot	Rise Time	S.S. Offset	S.S. Osc
1.04	0.4	0	No

1. ปรับค่า Fuzzy controller ได้เป็นขั้นตอนดังนี้

1.1 เริ่มจากการใช้ค่าที่กำหนดได้เป็น DEFAULT

coarse membership function	DEFAULT
fine membership function	DEFAULT
control rule	DEFAULT

จากผลตอบสนองที่ได้พบว่าในช่วง transient จะมี overshoot ต่ำ และ ในช่วง steady state ก็จะไม่มีการสั่นของผลตอบสนอง ผลตอบสนองเข้าสู่ setpoint ได้อย่างถูกต้องโดยไม่มี offset

1.2 ทำการปรับปรุงโดยการปรับปรุง control rule จะได้ผลการทดลองดังแสดงในตาราง

coarse membership function	DEFAULT
fine membership function	DEFAULT

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
Default	1.5	1.2	1.15	0.25	0	No
		1.05	1.15	0.25	0.01	No
	1.3	1.1	1.125	0.3	0.01	Small
		1.05	1.125	0.3	0	No
	1.1	1.05	1.09	0.35	0	No
		1.01	1.09	0.35	0	No
tst1	1.5	1.2	1.125	0.3	0.01	Small
		1.05	1.125	0.3	0	No
	1.3	1.1	1.1	0.3	0	No
		1.05	1.1	0.3	0	No
	1.1	1.05	1.06	0.3	0	No
		1.01	1.06	0.3	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
tst2	1.5	1.2	1.15	0.25	0.01	Small
		1.05	1.15	0.25	0	No
	1.3	1.1	1.125	0.3	0	No
		1.05	1.125	0.3	0	No
	1.1	1.05	1.06	0.3	0	No
		1.01	1.06	0.3	0	No
tst3	1.5	1.2	1.15	0.25	0.01	Small
		1.05	1.15	0.25	0.01	No
	1.3	1.1	1.125	0.25	0.01	Small
		1.05	1.125	0.25	0.01	No
	1.1	1.05	1.06	0.3	0.01	No
		1.01	1.06	0.3	0	No
tst4	1.5	1.2	1	1	0	No
		1.05	1	1	0	No
	1.3	1.1	1.005	0.7	0	No
		1.05	1.005	0.7	0	No
	1.1	1.05	1.03	0.5	0	No
		1.01	1.03	0.5	0	No
tst5	1.5	1.2	1.09	0.3	0.01	Small
		1.05	1.09	0.3	0.01	No
	1.3	1.1	1.07	0.3	0.01	Small
		1.05	1.07	0.3	0.01	No
	1.1	1.05	1.05	0.4	0.01	No
		1.01	1.05	0.4	0	No
tst6	1.5	1.2	1.16	0.3	0.14	No
		1.05	1.16	0.3	0.14	No
	1.3	1.1	1.1	0.3	-0.03	Small
		1.05	1.1	0.3	-0.01	No
	1.1	1.05	1.06	0.35	0	No
		1.01	1.06	0.35	0	No
tst7	1.5	1.2	1.07	0.35	0.01	Small
		1.05	1.07	0.35	0	No
	1.3	1.1	1.06	0.35	0	No
		1.05	1.06	0.35	0	No
	1.1	1.05	1.05	0.4	0	No
		1.01	1.05	0.4	0	No
tst8	1.5	1.2	1.125	0.25	0.01	Small
		1.05	1.125	0.25	0	No
	1.3	1.1	1.1	0.3	0	No
		1.05	1.1	0.3	0	No
	1.1	1.05	1.06	0.35	0	No
		1.01	1.06	0.35	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
tst9	1.5	1.2	1.14	0.25	0.01	Small
		1.05	1.14	0.25	0.01	Small
	1.3	1.1	1.125	0.25	0.01	Small
		1.05	1.125	0.25	0.01	Small
	1.1	1.05	1.09	0.3	0.01	Small
		1.01	1.09	0.3	0	No
t10	1.5	1.2	1.09	0.3	0.01	Small
		1.05	1.09	0.3	0	No
	1.3	1.1	1.06	0.35	0	No
		1.05	1.06	0.35	0	No
	1.1	1.05	1.05	0.4	0	No
		1.01	1.05	0.4	0	No
t11	1.5	1.2	1.08	0.35	0.01	Small
		1.05	1.08	0.35	0	No
	1.3	1.1	1.08	0.3	0	No
		1.05	1.08	0.3	0	No
	1.1	1.05	1.05	0.4	0	No
		1.01	1.05	0.4	0	No

จะพบว่าเมื่อใช้ control rule จาก file tst4.rul จะทำให้ผลตอบสนองเชิงเวลาดีกว่า control rule อื่นๆ โดยมี overshoot ลดลงแต่ก็ทำให้ rise time มีค่ามากขึ้น

2. หาค่า parameter ของ PID controller โดยวิธีของ Ziegler-Nichols

ซึ่งจะได้ค่า parameter ดังนี้คือ

Kp	Ti	Td
4.32	0.2	0.05

จะได้ค่าผลตอบสนองเชิงเวลา (Time response) แสดงดังตาราง

Overshoot	Rise time (sec.)	S.S. Offset	S.S. Osc
1.11	0.12	0	No

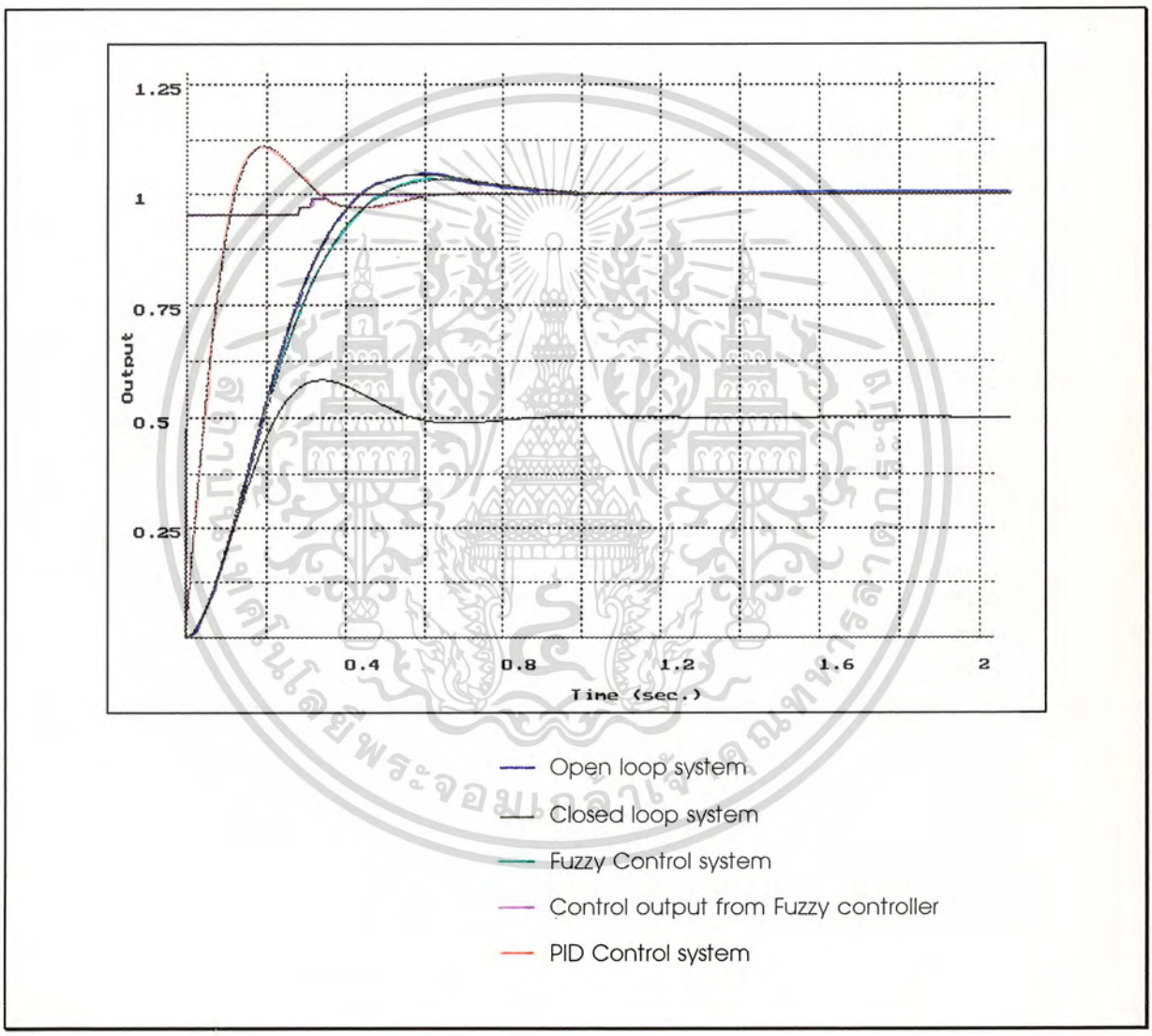
จากผลการทดลองจะเห็นว่าผลตอบสนองค่อนข้างน่าพอใจ เนื่องจากมี overshoot สูงขึ้นเล็กน้อย แต่ก็ทำให้ rise time ลดลงมาก และ ที่ steady state ก็เข้าสู่ setpoint ได้เป็นอย่างดี

3. เปรียบเทียบ Fuzzy controller ที่ดีที่สุด (จาก 1) กับ PID controller (จาก 2)

จะเห็นได้ว่า Fuzzy controller สำหรับระบบนี้จะค่อนข้างดีกว่า เนื่องจากทำให้เกิด overshoot น้อย และเข้าสู่เสถียรได้เร็วกว่า PID controller

Time response ของ การทดลองที่ 4

coarse membership function	default.mem
fine membership function	default.mem
control rule	tst4.rul
coarse control input	1.10
fine control input	1.01



4.5 การทดลองที่ 5

transfer function ของ process คือ $\frac{s+1}{s^2+s+1}$

Pole	-0.5±0.87j
Zero	-1

Open loop control system

Overshoot	Rise Time	S.S. Offset	S.S. Osc
1.28	1.1	0	No

1. ปรับค่า Fuzzy controller ได้เป็นขั้นตอนดังนี้

1.1 เริ่มจากการใช้ค่าที่กำหนดได้เป็น DEFAULT

coarse membership function	DEFAULT
fine membership function	DEFAULT
control rule	DEFAULT

จากผลตอบสนองที่ได้พบว่าในช่วง transientจะมี overshoot ลดลง และ ในช่วง steady state ก็จะมีการสั่นของผลตอบสนองเล็กน้อย แต่ผลตอบสนองก็เข้าสู่ setpoint ได้อย่างถูกต้องโดยไม่มี offset

1.2 ทำการปรับปรุงโดยการปรับปรุง control rule จะได้ผลการทดลองดังแสดงในตาราง

coarse membership function	DEFAULT
fine membership function	DEFAULT

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
Default	1.5	1.2	1.2	0.7	0.06	Small
		1.05	1.2	0.7	0.06	Small
	1.3	1.1	1.23	0.8	0	No
		1.05	1.23	0.8	0	No
	1.1	1.05	1.26	1	0	No
		1.01	1.26	1	0	No
tst1	1.5	1.2	1.17	0.8	0.06	Small
		1.05	1.17	0.8	0.06	Small
	1.3	1.1	1.23	0.9	0	No
		1.05	1.23	0.9	0	No
	1.1	1.05	1.26	1	0	No
		1.01	1.26	1	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
tst2	1.5	1.2	1.23	0.7	0.06	Small
		1.05	1.23	0.7	0.06	Small
	1.3	1.1	1.24	0.8	0	No
		1.05	1.24	0.8	0	No
	1.1	1.05	1.27	1	0	No
		1.01	1.27	1	0	No
tst3	1.5	1.2	1.21	0.7	0.06	Medium
		1.05	1.21	0.7	0.06	Medium
	1.3	1.1	1.24	0.8	0.06	Small
		1.05	1.24	0.8	0.06	Small
	1.1	1.05	1.27	1	0.01	Very Small
		1.01	1.27	1	0	No
tst4	1.5	1.2	1.25	1.6	-0.05	Medium
		1.05	1.25	1.6	0	No
	1.3	1.1	1.27	1.3	-0.05	Small
		1.05	1.27	1.3	0	No
	1.1	1.05	1.28	1.15	0	No
		1.01	1.28	1.15	0	No
tst5	1.5	1.2	1.18	1	0.06	Medium
		1.05	1.18	1	0.06	Medium
	1.3	1.1	1.23	1.1	0.06	Small
		1.05	1.23	1.1	0.06	Small
	1.1	1.05	1.26	1.1	0	No
		1.01	1.26	1.1	0	No
tst6	1.5	1.2	1.77	0.9	0.3	No
		1.05	1.77	0.9	0.3	No
	1.3	1.1	1.58	1	0.23	No
		1.05	1.58	1	0.23	No
	1.1	1.05	1.37	1	-0.02	No
		1.01	1.37	1	0	No
tst7	1.5	1.2	1.15	1.05	0.06	Small
		1.05	1.15	1.05	0.06	Small
	1.3	1.1	1.21	1.05	0	No
		1.05	1.21	1.05	0	No
	1.1	1.05	1.28	1.1	0	No
		1.01	1.28	1.1	0	No
tst8	1.5	1.2	1.18	0.8	0.06	Small
		1.05	1.18	0.8	0.06	Small
	1.3	1.1	1.26	0.95	0	No
		1.05	1.26	0.95	0	No
	1.1	1.05	1.28	1	0	No
		1.01	1.28	1	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
tst9	1.5	1.2	1.14	0.7	0.06	Medium
		1.05	1.14	0.7	0.06	Medium
	1.3	1.1	1.15	0.8	0.06	Medium
		1.05	1.15	0.8	0.06	Medium
	1.1	1.05	1.25	1	0.005	Medium
		1.01	1.25	1	0	No
t10	1.5	1.2	1.18	1	0.06	Small
		1.05	1.18	1	0.06	Small
	1.3	1.1	1.23	1.05	0	No
		1.05	1.23	1.05	0	No
	1.1	1.05	1.26	1.1	0	No
		1.01	1.26	1.1	0	No
t11	1.5	1.2	1.18	1	0.06	Small
		1.05	1.18	1	0.06	Small
	1.3	1.1	1.23	1.05	0	No
		1.05	1.23	1.05	0	No
	1.1	1.05	1.26	1.1	0	No
		1.01	1.26	1.1	0	No

จะพบว่าเมื่อใช้ control rule จาก file tst9.rul จะทำให้ผลตอบสนองเชิงเวลาดีกว่า control rule อื่นๆ โดยมี overshoot และค่า rise time ลดลง แต่การเลือก control input ก็จะมีผลกับระบบทำให้ระบบเกิดค่า offset และการแกว่งที่ steady state ด้วย

2. หาค่า parameter ของ PID controller โดยวิธีของ Ziegler-Nichols

ซึ่งจะได้ค่า parameter ดังนี้คือ

Kp	Ti	Td
118.8	0.01	0.0025

จะได้ค่าผลตอบสนองเชิงเวลา (Time response) แสดงดังตาราง

Overshoot	Rise time (sec.)	S.S. Offset	S.S. Osc
1.26	0.001	0	No

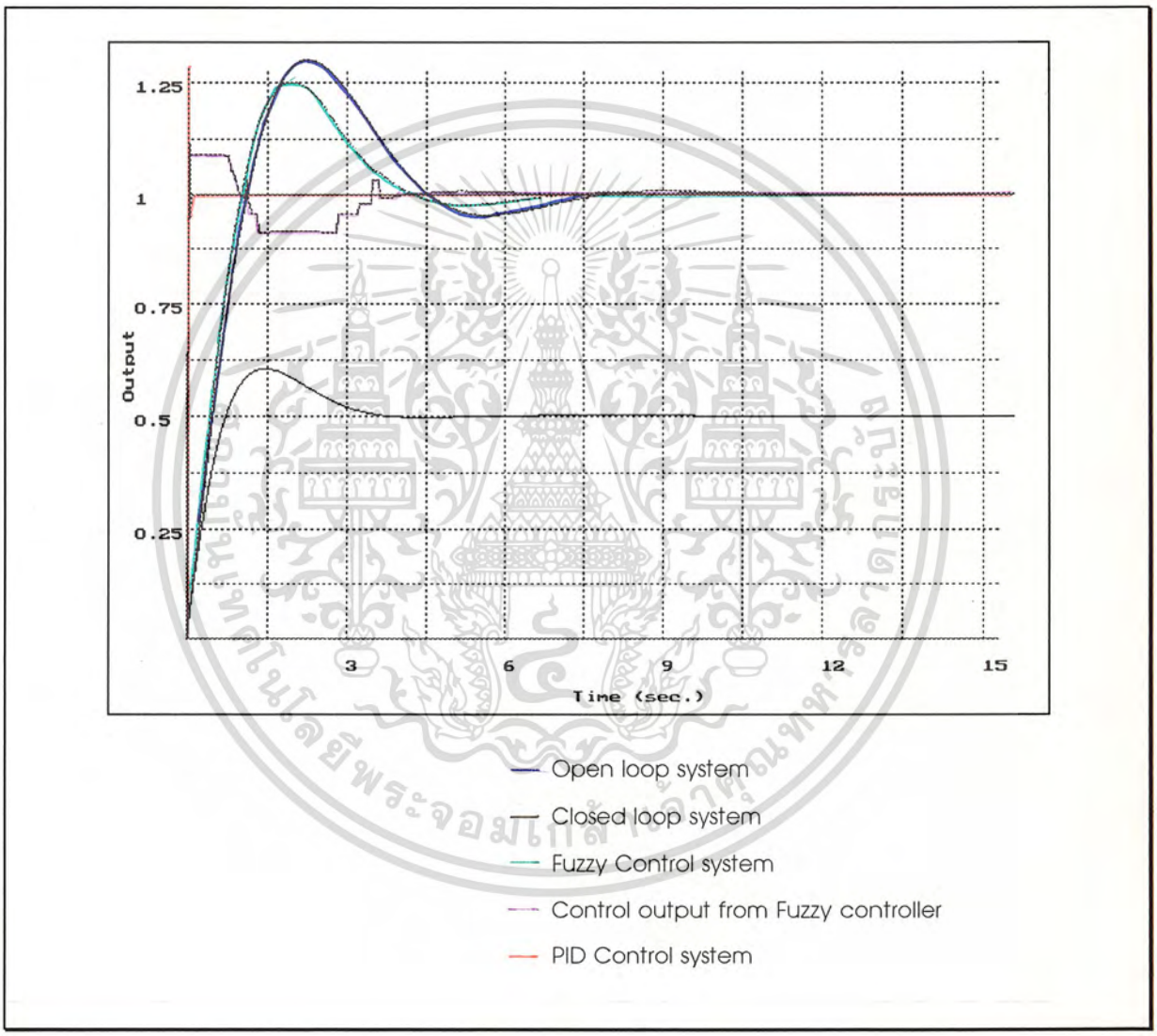
จากผลการทดลองจะเห็นว่าผลตอบสนองเป็นที่น่าพอใจมาก เพราะมีค่า overshoot ลดลงเล็กน้อย และค่า rise time ลดลงมาก นอกจากนี้ที่ steady state ก็เข้าสู่ setpoint ได้เป็นอย่างดี

3. เปรียบเทียบ Fuzzy controller ที่ดีที่สุด (จาก 1) กับ PID controller (จาก 2)

จะเห็นได้ว่า Fuzzy controller สำหรับระบบนี้จะมีค่า overshoot ต่ำกว่า แต่ PID controller จะทำให้ค่า rise time ลดลงมากและเข้าสู่ steady state ได้เร็วกว่า ถึงแม้ว่าจะมีค่า overshoot สูงกว่า แต่ก็เพียงเล็กน้อย

Time response ของ การทดลองที่ 5

coarse membership function	default.mem
fine membership function	default.mem
control rule	tst9.rul
coarse control input	1.10
fine control input	1.01



4.6 การทดลองที่ 6

Transfer function ของ process คือ $\dfrac{25s+5}{s^3+5s^2+25s+5}$

Pole	-0.208 , -2.396±4.273j
Zero	-0.2

Open loop control system

Overshoot	Rise Time	S.S. Offset	S.S. Osc
1.21	0.45	0	No

1. ปรับค่า Fuzzy controller ได้เป็นขั้นตอนดังนี้

1.1 เริ่มจากการใช้ค่าที่กำหนดได้เป็น DEFAULT

coarse membership function	DEFAULT
fine membership function	DEFAULT
control rule	DEFAULT

จากผลตอบสนองที่ได้พบว่าในช่วง transient จะมี overshoot สูงขึ้น แต่ค่า rise time ลดลง และ ในช่วง steady state จะไม่มีการสั่นของผลตอบสนองและ setpoint ได้อย่างถูกต้องโดยไม่มี offset

1.2 ทำการปรับปรุงโดยการปรับปรุง control rule จะได้ผลการทดลองดังแสดงในตาราง

coarse membership function	DEFAULT
fine membership function	DEFAULT

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
Default	1.5	1.2	1.31	0.3	0	No
		1.05	1.31	0.3	0.01	No
	1.3	1.1	1.28	0.35	0	No
		1.05	1.28	0.35	0.01	No
	1.1	1.05	1.25	0.4	0.01	No
		1.01	1.25	0.4	0	No
tst1	1.5	1.2	1.27	0.3	0	No
		1.05	1.27	0.3	0.01	No
	1.3	1.1	1.25	0.35	0	No
		1.05	1.25	0.35	0.01	No
	1.1	1.05	1.22	0.4	0.01	No
		1.01	1.22	0.4	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
tst2	1.5	1.2	1.32	0.3	0	No
		1.05	1.32	0.3	0.01	No
	1.3	1.1	1.29	0.35	0	No
		1.05	1.29	0.35	0.01	No
	1.1	1.05	1.25	0.4	0.01	No
		1.01	1.25	0.4	0	No
tst3	1.5	1.2	1.32	0.3	0.01	Large
		1.05	1.32	0.3	0.01	Medium
	1.3	1.1	1.29	0.35	0.01	Large
		1.05	1.29	0.35	0.01	Medium
	1.1	1.05	1.25	0.4	0.01	Medium
		1.01	1.25	0.4	0	No
tst4	1.5	1.2	1.11	0.63	0.045	Large
		1.05	1.11	0.63	0	No
	1.3	1.1	1.15	0.5	0	No
		1.05	1.15	0.5	0	No
	1.1	1.05	1.20	0.45	0	No
		1.01	1.20	0.45	0	No
tst5	1.5	1.2	1.20	0.45	0.01	Large
		1.05	1.20	0.45	0.01	Medium
	1.3	1.1	1.20	0.45	0.01	Large
		1.05	1.20	0.45	0.01	Medium
	1.1	1.05	1.20	0.45	0.01	Medium
		1.01	1.20	0.45	0	No
tst6	1.5	1.2	1.34	0.6	-0.02	Large
		1.05	1.34	0.6	-0.01	No
	1.3	1.1	1.30	0.3	-0.02	Large
		1.05	1.30	0.3	-0.01	No
	1.1	1.05	1.25	0.4	-0.01	No
		1.01	1.25	0.4	0	No
tst7	1.5	1.2	1.18	0.4	0	No
		1.05	1.18	0.4	0.01	No
	1.3	1.1	1.19	0.4	0	No
		1.05	1.19	0.4	0.01	No
	1.1	1.05	1.20	0.4	0.01	No
		1.01	1.20	0.4	0	No
tst8	1.5	1.2	1.31	0.3	0	No
		1.05	1.31	0.3	0.01	No
	1.3	1.1	1.26	0.375	0	No
		1.05	1.26	0.375	0.01	No
	1.1	1.05	1.23	0.4	0.01	No
		1.01	1.23	0.4	0	No

Rules File	CCI Max.	FCI Max.	Overshoot	Rise Time	S.S. Offset	S.S. Osc
tst9	1.5	1.2	1.31	0.3	0.01	Large
		1.05	1.31	0.3	0.01	Medium
	1.3	1.1	1.27	0.35	0.01	Large
		1.05	1.27	0.35	0.01	Medium
	1.1	1.05	1.24	0.4	0.01	Large
		1.01	1.24	0.4	0	No
t10	1.5	1.2	1.20	0.4	0	No
		1.05	1.20	0.4	0.01	No
	1.3	1.1	1.20	0.4	0	No
		1.05	1.20	0.4	0.01	No
	1.1	1.05	1.21	0.4	0.01	No
		1.01	1.21	0.4	0	No
t11	1.5	1.2	1.20	0.4	0	No
		1.05	1.20	0.4	0.01	No
	1.3	1.1	1.20	0.4	0	No
		1.05	1.20	0.4	0.01	No
	1.1	1.05	1.21	0.45	0.01	No
		1.01	1.21	0.45	0	No

จะพบว่าเมื่อใช้ control rule จาก file tst4.rul จะทำให้ผลตอบสนองเชิงเวลาดีกว่า control rule อื่นๆ โดยมี overshoot ลดลงแต่ก็ทำให้ rise time มีค่ามากขึ้นเล็กน้อย

2. หาค่า parameter ของ PID controller โดยวิธีของ Ziegler-Nichols

ซึ่งจะได้ค่า parameter ดังนี้คือ

Kp	Ti	Td
5.55	0.16	0.04

จะได้ค่าผลตอบสนองเชิงเวลา (Time response) แสดงดังตาราง

Overshoot	Rise time (sec.)	S.S. Offset	S.S. Osc
1.45	0.11	0	No

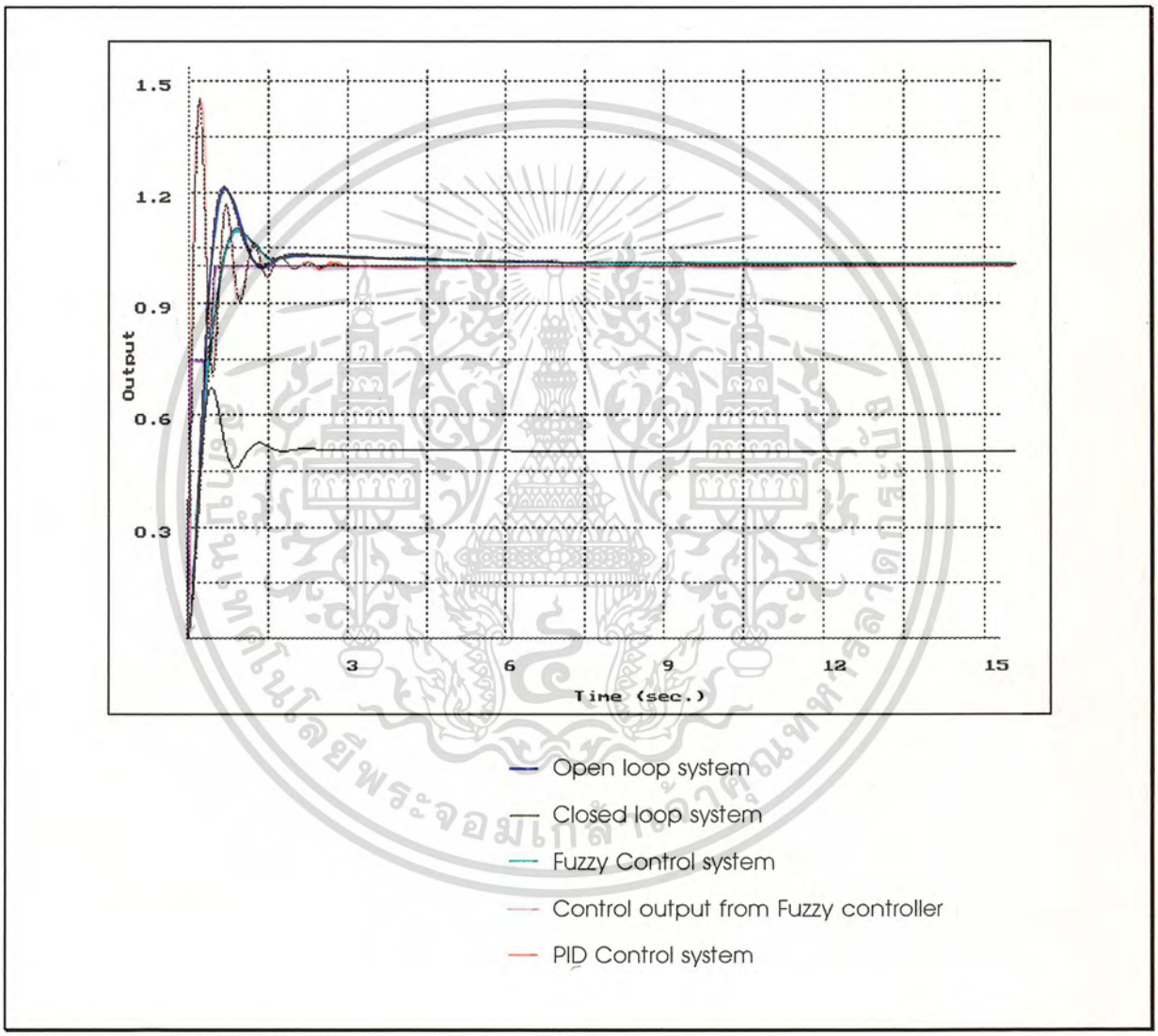
จากผลการทดลองจะเห็นว่าผลตอบสนองเป็นที่น่าพอใจ แม้ว่าจะมี overshoot ค่อนข้างสูง แต่ค่า rise time ลดลงพอสมควร และ ที่ steady state ก็เข้าสู่ setpoint ได้เป็นอย่างดี

3. เปรียบเทียบ Fuzzy controller ที่ดีที่สุด (จาก 1) กับ PID controller (จาก 2)

จะเห็นได้ว่า Fuzzy controller สำหรับระบบนี้จะค่อนข้างดีกว่า เนื่องจากทำให้เกิด overshoot น้อย และเข้าสู่เสถียรได้เร็วกว่า PID controller

Time response ของ การทดลองที่ 6

coarse membership function	default.mem
fine membership function	default.mem
control rule	tst4.rul
coarse control input	1.50
fine control input	1.20



4.7 การทดลองโดยใช้ Fuzzy controller ควบคุม Process ซึ่งจำลองการทำงานด้วย Simulator

เนื่องจากข้อจำกัดของ Simulator ทำให้เราสามารถทดลองโดยจำลอง process ได้ไม่เกินระบบอันดับสอง นอกจากนี้ Recorder ที่ใช้ในการบันทึกผลสามารถบันทึกได้เพียง 2 ช่อง ดังนั้นในการทดลอง เราจะบันทึกค่าเพียงผลตอบสนองจากการควบคุมด้วย fuzzy controller และ control input ที่ส่งออกมาจาก fuzzy controller ดังนี้

ในการทดลองเราเลือกใช้ parameter ต่างๆ ดังตาราง

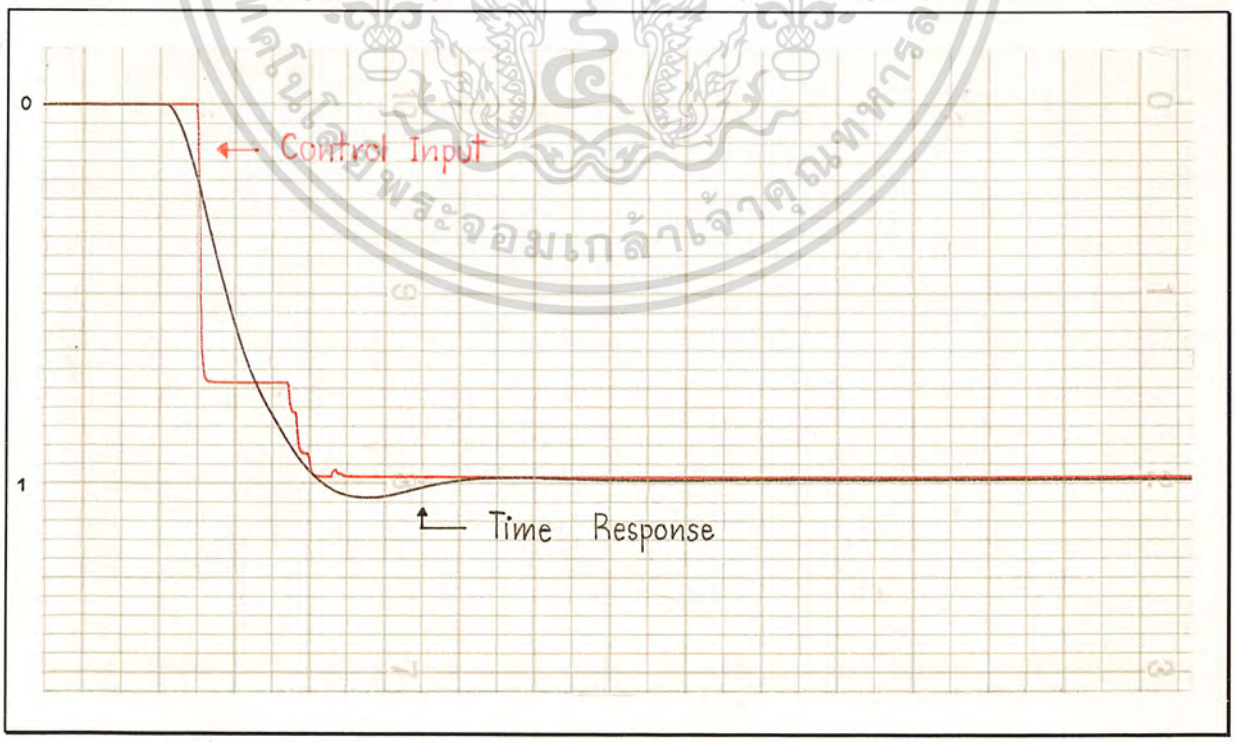
coarse membership function	default.mem
fine membership function	default.mem
control rule	tst4.rul
coarse control input	1.50
fine control input	1.05

หมายเหตุ ในการทดลองกับ Simulator เราเลือกใช้ค่า Sampling time เป็น 0.01 sec. ทั้งหมด

4.7.1 การทดลองกับ Simulator ที่ 1

Transfer function ของ process คือ $\frac{1}{s^2+s+1}$

ได้ผลตอบสนองของระบบดังนี้

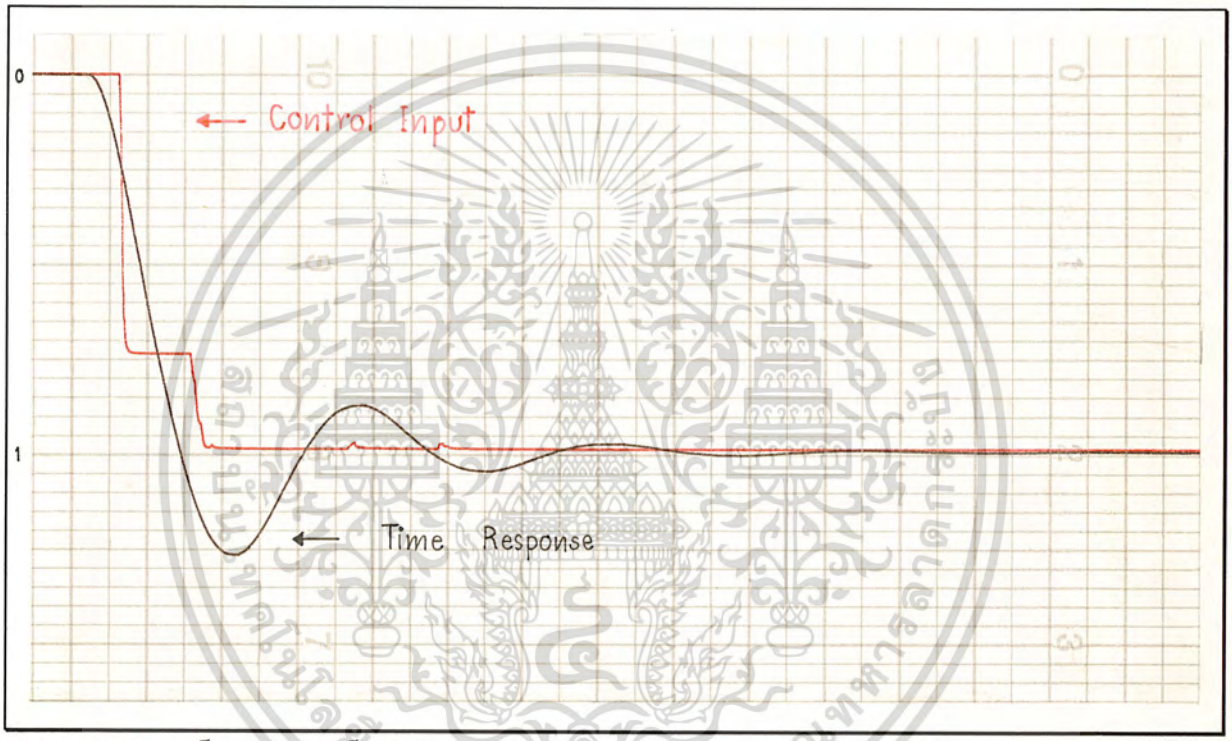


หมายเหตุ ความเร็ว Recorder เป็น 300 mm/min

4.7.2 การทดลองกับ Simulator ที่ 2

Transfer function ของ process คือ $\frac{1}{s^2+0.5s+1}$

ได้ผลตอบสนองของระบบดังนี้

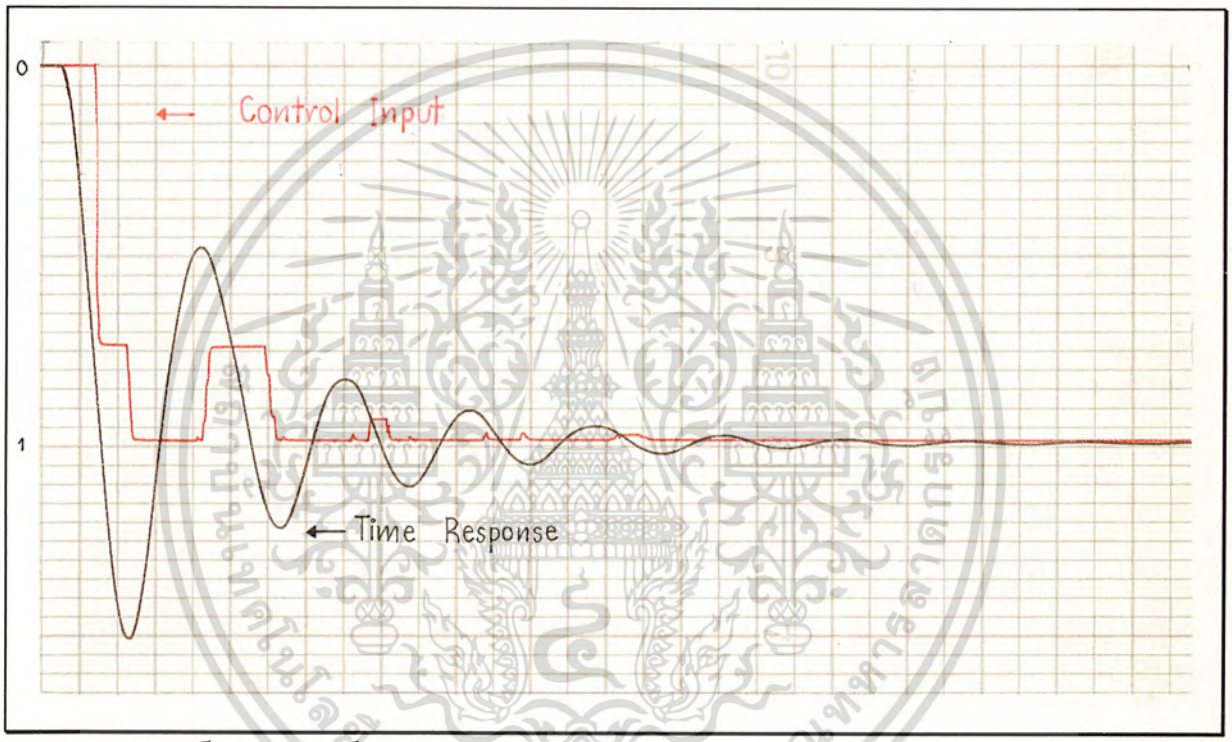


หมายเหตุ ความเร็ว Recorder เป็น 300 mm/min

4.7.3 การทดลองกับ Simulator ที่ 3

Transfer function ของ process คือ $\frac{1}{s^2+0.2s+1}$

ได้ผลตอบสนองของระบบดังนี้



หมายเหตุ ความเร็ว Recorder เป็น 150 mm/min

บทที่ 5
สรุปและวิจารณ์ผลการทดลอง

1. การกำหนดลักษณะของ membership function เช่น ลักษณะการซ้อนทับกัน (overlap) นั้นจะมีผลต่อผลตอบสนองของระบบ แต่ก็ยังไม่มากนักเมื่อเทียบกับผลที่เกิดขึ้นจาก control rule ซึ่งจะมีผลมากโดยเฉพาะกับช่วง transient ของผลตอบสนอง นอกจากนี้การปรับค่า Range of error, Change of error และ control input ก็มีผลต่อผลตอบสนองด้วยเช่นกัน จากการทดลองจะเห็นความสัมพันธ์ระหว่าง Range ของ control input และผลตอบสนอง ก็คือ ในระบบส่วนใหญ่ถ้าให้ Range ของ coarse control input กว้าง ก็จะทำให้มี overshoot สูง แต่บางระบบผลที่ได้ก็เป็นไปในลักษณะตรงกันข้าม จึงทำให้สรุปได้เพียงแต่ว่า Range ของ control input มีผลต่อค่า overshoot

2. จากผลการทดลองจะเห็นได้ว่า control rule ที่ดีที่สุดของระบบหนึ่งอาจเป็น control rule ที่ไม่ดีสำหรับอีกระบบหนึ่ง ทั้งนี้ขึ้นกับธรรมชาติของระบบนั้น ดังนั้นการเลือก control rule ที่เหมาะสมที่สุดนั้นก็ต้องทำการลองผิดลองถูกจนกว่าจะได้ control rule ที่ดีที่สุดนั่นเอง แต่หลักการคร่าวๆ ในการออกแบบ control rule สำหรับการทดลองนี้ก็คือ การ compensate ส่วนที่ไม่ต้องการให้ได้เท่ากับเป้าหมาย (setpoint) ที่ต้องการ เช่น เมื่อผลตอบสนองเป็นลบมาก (er=LN,cer=LN) ก็ให้ control input เป็นบวกมาก (ci=LP) เข้าไป แต่ถ้าผลตอบสนองที่ออกมามี overshoot สูงเกินไปก็ลด overshoot โดยการลด control input ให้เป็นบวกน้อย (ci=SP) ซึ่งก็จะทำให้ overshoot ลดลงตามต้องการ หรือการออกแบบ control rule อีกทางหนึ่งอาจได้จากการพิจารณา matrix รูปแบบดังนี้ คือ

cer / *er*	LN	SN	ZE	SP	LP
LN			LP		
SN			SP	SN	
ZE	LP	SP	ZE	SN	LN
SP		SP	SN		
LP			LN		

ซึ่งหลักการก็จะคล้ายคลึงกับวิธีแรก และจากการทดลองที่ผ่านมา จะเห็นได้ว่าช่วงรอบนอกของ matrix จะมีผลต่อผลตอบสนองในช่วง transient ส่วนรอบในมักจะมีผลต่อช่วง steady state

3. จำนวน control rule ที่มากนั้นไม่ได้หมายความว่าจะทำให้การควบคุมมีประสิทธิภาพดีขึ้นเสมอไป การกำหนดจำนวนของ control rule ที่มากเกินไปอาจทำให้การทำงานของ controller ใช้เวลามากขึ้นโดยไม่จำเป็น ดังนั้นการกำหนด rule นั้นก็ควรเริ่มจากการใช้ control rule น้อยๆก่อน แล้วจึงค่อยๆเพิ่มขึ้นเท่าที่จำเป็นต้องใช้เท่านั้น ซึ่งการทดลองที่ผ่านมาก็มักจะใช้ control rule ประมาณ 11-13 rules

4. ผลตอบสนองที่ได้จาก Fuzzy controller ส่วนใหญ่จะพบว่าที่ steady state จะไม่เรียบเลยทีเดียว แต่จะมีการแกว่งด้วย amplitude ขนาดเล็กๆ ทั้งนี้เป็นเพราะ controller พยายามที่จะปรับให้ผลตอบสนองอยู่ที่ setpoint ให้ได้นั่นเอง. การแก้ไขอาจทำได้โดยการกำหนด Range ของ fine control input ให้มีค่าน้อยๆ คือใกล้ setpoint มากๆ ก็จะทำให้ทั้งการ oscillate และ offset ในช่วง steady state หมดไป

5. จากการทดลองในบางกรณีนั้นพบว่า Fuzzy controller ยังทำงานได้ไม่ดีนักเมื่อเปรียบเทียบกับ PID controller ซึ่งสาเหตุที่อาจเป็นไปได้ก็ได้แก่ ผู้ทดลองอาจมีประสบการณ์ไม่มากพอในการออกแบบ control rule และ membership function และอาจเป็นเพราะระบบที่นำมาทดลองนี้ เป็นระบบที่ไม่ต้องอาศัยความชำนาญของมนุษย์ในการควบคุม จึงเหมาะกับ controller ที่ได้มาจากการคำนวณทางคณิตศาสตร์มากกว่า

6. จากการทดลองและการศึกษาที่ผ่านมาสามารถสรุปข้อดีข้อเสียของ Fuzzy controller ได้ดังนี้

ข้อดีของ Fuzzy controller

- ไม่ต้องการรายละเอียดที่ซับซ้อนในการออกแบบ Controller ทำให้สามารถใช้งานกับระบบที่มีแบบจำลองทางคณิตศาสตร์ที่ซับซ้อนได้ดีกว่า
- แสดงด้วยกฎที่เราคุ้นเคยกันดี
- ด้วยการเลือกใช้ set ที่แตกต่างกันของ control rule ทำให้ Fuzzy controller สามารถใช้ได้กับ input ในช่วงกว้าง
- สามารถนำความรู้และประสบการณ์ของผู้ควบคุมที่ไม่สามารถ แสดงได้ ด้วยรูปแบบทางคณิตศาสตร์มาใช้ในการควบคุมได้
- เมื่อมีการเปลี่ยนแปลงเกิดขึ้นกับระบบอย่างกระทันหัน โดยมิได้คาดหมาย Fuzzy controller ก็จะสามารถจัดการได้
- fuzzy controller นั้นง่ายต่อการทำความเข้าใจ ทั้งนี้เพราะ control rule มีลักษณะคล้ายคลึงกับภาษาของมนุษย์ และไม่ต้องอาศัยการคำนวณทางคณิตศาสตร์ที่ยุ่งยาก ดังนั้นจึงไม่จำเป็นที่จะต้องอาศัยวิศวกรที่มีความชำนาญมากนักในการใช้งาน fuzzy controller

ข้อเสียของ Fuzzy controller

- ไม่มีทฤษฎีที่ชัดเจนในการกำหนด control rule และ membership function ทำให้การออกแบบ controller ที่เหมาะสมที่สุดต้องใช้เวลามากในการออกแบบ เพราะต้องใช้วิธีลองผิดลองถูก เนื่องจาก control rule และ membership function หนึ่ง ก็จะเหมาะสมกับระบบหนึ่งๆ เท่านั้น
- ยังไม่มีทฤษฎีรองรับที่ชัดเจนในเรื่องของเสถียรภาพของระบบที่ควบคุมโดย fuzzy controller

7. จากการทดลองและการศึกษา พบว่ามีความเป็นไปได้ที่จะนำ Fuzzy controller มาใช้เป็น Universal controller ซึ่งหมายถึง controller ที่ใช้ได้ทั่วไปไม่เจาะจงเฉพาะระบบเดียวเท่านั้น ทั้งนี้จากการพัฒนา application ทาง Fuzzy control ที่ผ่านๆ มา ส่วนใหญ่จะเป็น Fuzzy controller เฉพาะสำหรับระบบหนึ่งๆ เท่านั้น ซึ่งการสร้าง Fuzzy controller ที่เป็น Universal controller นั้นข้อสำคัญจะต้องสะดวกในการปรับเปลี่ยน control rule และ membership function และควรที่จะมีหลักการในการปรับ control rule ที่ชัดเจนกว่านี้ ซึ่งต้องอาศัยการพัฒนาอีกระยะหนึ่ง

แนวทางในการพัฒนา

- การพัฒนาขั้นต่อไปจะเป็นการสร้าง application จริงของ fuzzy controller ที่มีลักษณะเป็น Universal controller โดยการใช้ไมโครคอมพิวเตอร์ หรือไมโครโปรเซสเซอร์ มาโปรแกรมให้เป็น Fuzzy controller สำหรับควบคุมกระบวนการจริงๆ
- ควรมีการหาหลักการในการหา control rule และการปรับค่า parameter ต่างๆ ที่ชัดเจนกว่าในปัจจุบัน

บรรณานุกรม

1. BEN EZZELL,*"GRAPHICS PROGRAMMING IN TURBO C++"*,ADDISON-WESLEY PUBLISHIG COMPANY,INC.
2. BORLAND INTERNATIONAL,INC.,*"TURBO C++ -GETTING STARTED,USER'S GUIDE, REFERENCE GUIDE, PROGRAMMER'S GUIDE"*
3. DIDIER DUBOIS, HENRI PRADE, *"FUZZY SETS AND SYSTEMS"*,ACADEMIC PRESS,INC.
4. GEORGE J.KLIR, TINA A.FOLGER,*"FUZZY SETS, UNCERTAINTY, AND INFORMATION"*,PRENTICE-HALL INTERNATIONAL EDITIONS.
5. KATSUHIKO OGATA,*"MODERN CONTROL ENGINEERING 2ND."*,PRENTICE-HALL INTERNATIONAL,INC.
6. KATSUHIKO OGATA,*"DISCRETE-TIME CONTROL SYSTEM"*,PRENTICE-HALL INTERNATIONAL,INC.
7. KRIS JAMSHA,*"DOS PROGRAMMING THE COMPLETE REFERENCE"*,OSBORNE/MCGRAW HILL PUBLISHIG COMPANY,INC.
8. L.A.ZADEH,*"FUZZY SETS"*,INFORMATION AND CONTROL, VOL.8, 1965
9. LI-XIN WANG,*"STABLE ADAPTIVE FUZZY CONTROL OF NONLINEAR SYSTEMS"*, IEEE TRANSACTIONS ON FUZZY SYSTEMS, VOL.1,NO.2,MAY 1993
10. MICHIO SUGENO,*"INDUSTRIAL APPLICATIONS OF FUZZY CONTROL"*,NORTH-HOLLAND ELSEVIER SCIENCE PUBLISHING COMPANY INC.
11. MOHAMMAD JAMSHIDI, NADER VADIEE, TIMOTHY J. ROSS, *"FUZZY LOGIC AND CONTROL"*, PRENTICE HALL,INC.
12. NIKOLAOS P.PAPANIKOLOPOULOS,SPYROS TZAFESTAS,*"INCREMENTAL FUZZY EXPERT PID CONTROL"*, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, VOL.37,NO.5,OCTOBER 1990
13. Y.F. LI,C.C. LAU,*"DEVELOPMENT OF FUZZY ALGORITHMS FOR SERVO SYSTEMS"*,IEEE CONTROL SYSTEMS MAGAZINE,APRIL 1989
14. YOKOGAWA ELECTRIC CORPORATION TECHNICAL INFORMATION,*"INTRODUCTION TO FUZZY CONTROL"*
15. กิตติ อินทรานนท์,ศรีรักษ์ ศรีทองชัย,*"การตัดสินใจโดยใช้ทฤษฎีฟัซซี่เซต"*,วิศวกรรมสาร, ปีที่ 46 เล่มที่ 2, กุมภาพันธ์ 2536
16. สิทธิชัย เคนบุปผา,*"ฟัซซี่ลอจิก นวัตกรรมทางเทคโนโลยีเพื่อความรู้สึกของมนุษย์"*, วารสารเซมิคอนดักเตอร์ อิเลคทรอนิคส์ ฉบับที่ 121,ตุลาคม 2535
17. เอกสารสัมนาทางวิชาการ โดย คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย กับ YOKOGAWA ELECTRIC CORPORATION (JAPAN),*"FUZZY CONTROL THEORY AND ITS APPLICATION"*

ภาคผนวก

Control Rules Files

DEFAULT.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN	LP				
SN		SP	SP		
ZE		SP	ZE	SN	
SP			SN	SN	
LP					LN

TST1.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN	SP				
SN		SP	SP		
ZE		SP	ZE	SN	
SP			SN	SN	
LP					SN

TST2.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN	LP		SP		
SN		SP	ZE		
ZE		SP	ZE	SN	
SP			ZE	SN	
LP			SN		LN

TST3.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN	LP				
SN		SP	ZE		
ZE	ZE	ZE	ZE	ZE	ZE
SP			ZE	SN	
LP					LN

TST4.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN	SN				
SN		ZE	ZE		
ZE		ZE	ZE	ZE	
SP			ZE	ZE	
LP					SP

TST5.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN			SP		
SN		SP	ZE		
ZE	SP	ZE	ZE	ZE	SN
SP			ZE	SN	
LP			SN		

TST6.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN	LP		SN		SP
SN		LN		SP	
ZE	SP		ZE		SN
SP		SN		LP	
LP	SN		SN		LN

TST7.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN			SP		
SN			SP	ZE	
ZE	LP	SP	ZE	SN	LN
SP		ZE	SN		
LP			SN		

TST8.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN		SP			
SN		SP	SP	SN	
ZE		SP	ZE	SN	
SP	LP	SP	SN		
LP		SN			

TST9.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN	LP				SP
SN		LP		ZE	
ZE	SP		ZE		SN
SP		ZE		LN	
LP	SN				LN

T10.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN			SP		
SN		SP	ZE		
ZE		SP	ZE	SN	
SP			ZE	SN	
LP			SN		

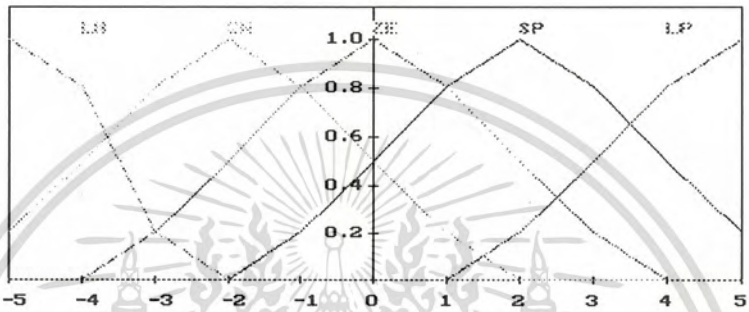
T11.RUL

CE \ ER	LN	SN	ZE	SP	LP
LN			SP		
SN		SP	ZE		
ZE	SP	SP	ZE	SN	SN
SP			ZE	SN	
LP			SN		

File DEFAULT.MEM (Error Table)

ERROR TABLE

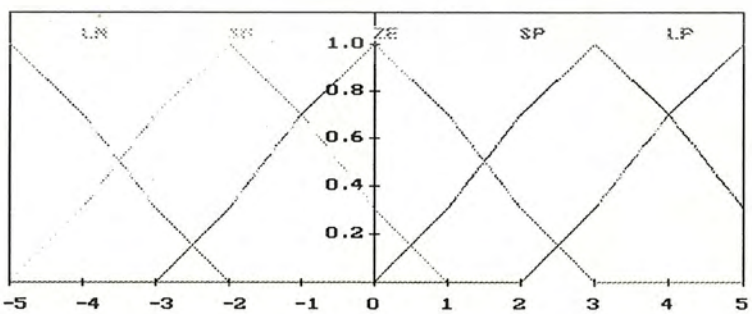
	-5	-4	-3	-2	-1	0	1	2	3	4	5
LN	1.0	0.8	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
SN	0.2	0.5	0.8	1.0	0.8	0.5	0.2	0.0	0.0	0.0	0.0
ZE	0.0	0.0	0.2	0.5	0.8	1.0	0.8	0.5	0.2	0.0	0.0
SP	0.0	0.0	0.0	0.0	0.2	0.5	0.8	1.0	0.8	0.5	0.2
LP	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.5	0.8	1.0



File DEFAULT.MEM (Change of error Table)

CHANGE OF ERROR TABLE

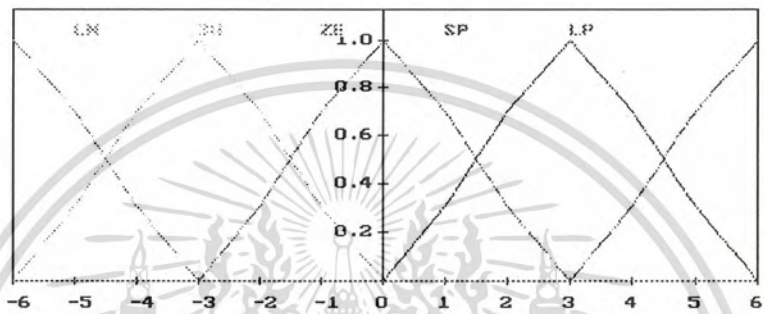
	-5	-4	-3	-2	-1	0	1	2	3	4	5
LN	1.0	0.7	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
SN	0.0	0.3	0.7	1.0	0.7	0.3	0.0	0.0	0.0	0.0	0.0
ZE	0.0	0.0	0.0	0.3	0.7	1.0	0.7	0.3	0.0	0.0	0.0
SP	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.7	1.0	0.7	0.3
LP	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.7	1.0



File DEFAULT.MEM (Control input Table)

CONTROL INPUT TABLE

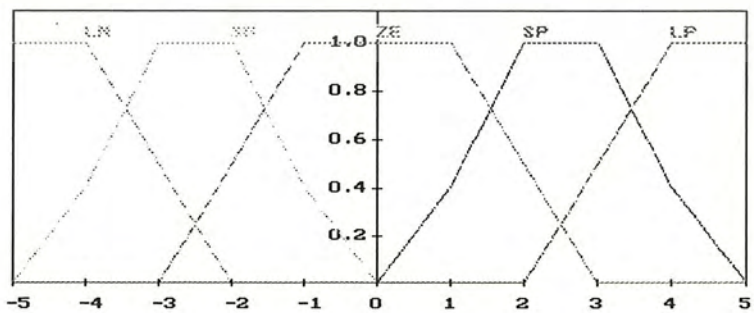
	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
LN	1.0	0.7	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
SN	0.0	0.3	0.7	1.0	0.7	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ZE	0.0	0.0	0.0	0.0	0.3	0.7	1.0	0.7	0.3	0.0	0.0	0.0	0.0
SP	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.7	1.0	0.7	0.3	0.0
LP	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.7	1.0



File TEST.MEM (Error Table)

ERROR TABLE

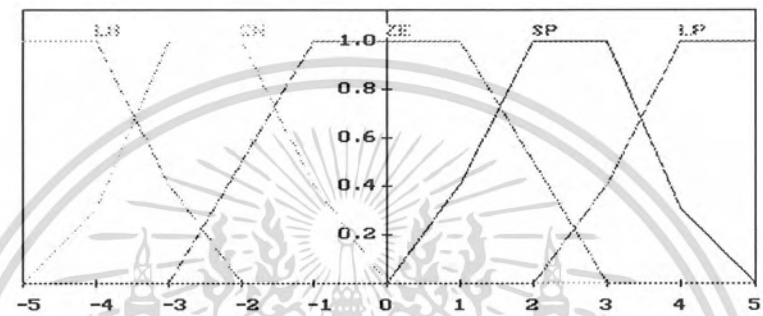
	-5	-4	-3	-2	-1	0	1	2	3	4	5
LN	1.0	1.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
SN	0.0	0.4	1.0	1.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0
ZE	0.0	0.0	0.0	0.5	1.0	1.0	1.0	0.5	0.0	0.0	0.0
SP	0.0	0.0	0.0	0.0	0.0	0.0	0.4	1.0	1.0	0.4	0.0
LP	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	1.0	1.0



File TEST.MEM (Change of error Table)

CHANGE OF ERROR TABLE

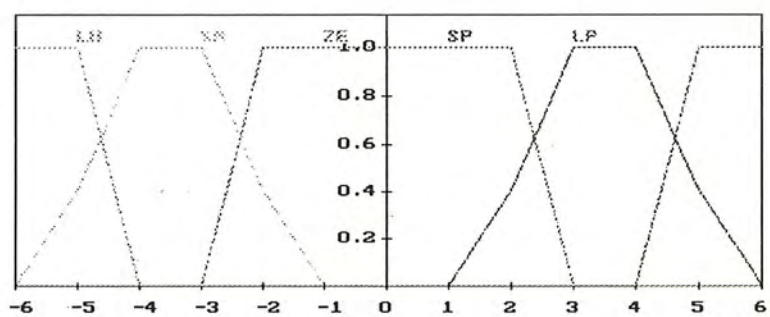
	-5	-4	-3	-2	-1	0	1	2	3	4	5
LN	1.0	1.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
SN	0.0	0.3	1.0	1.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0
ZE	0.0	0.0	0.0	0.5	1.0	1.0	1.0	0.5	0.0	0.0	0.0
SP	0.0	0.0	0.0	0.0	0.0	0.0	0.4	1.0	1.0	0.3	0.0
LP	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4	1.0	1.0



File TEST.MEM (Control input Table)

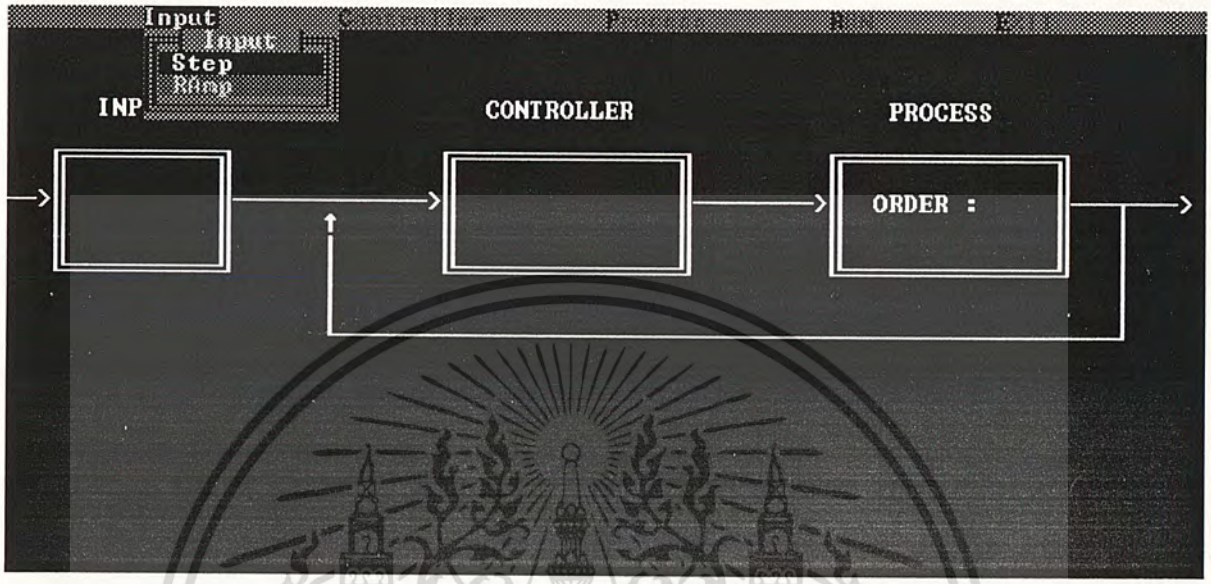
CONTROL INPUT TABLE

	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
LN	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
SN	0.0	0.4	1.0	1.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ZE	0.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0
SP	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4	1.0	1.0	0.4	0.0
LP	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0



คู่มือการใช้โปรแกรม Fuzzy Controller Simulation

1. เมื่อเริ่มต้นเข้ามาสู่โปรแกรมจะมี Pull-down menus ดังภาพ



การใช้งานนั้นเราต้องป้อนค่าต่างๆ ให้ครบก่อนที่จะ run โปรแกรม โดยจะป้อนค่าใดก่อนหรือหลังก็ได้ ซึ่งใน Menu INPUT นั้นยังไม่ได้พัฒนาให้สามารถใช้ input เป็น Ramp function ได้ จึงเลือกได้เฉพาะ input ที่เป็น Step function

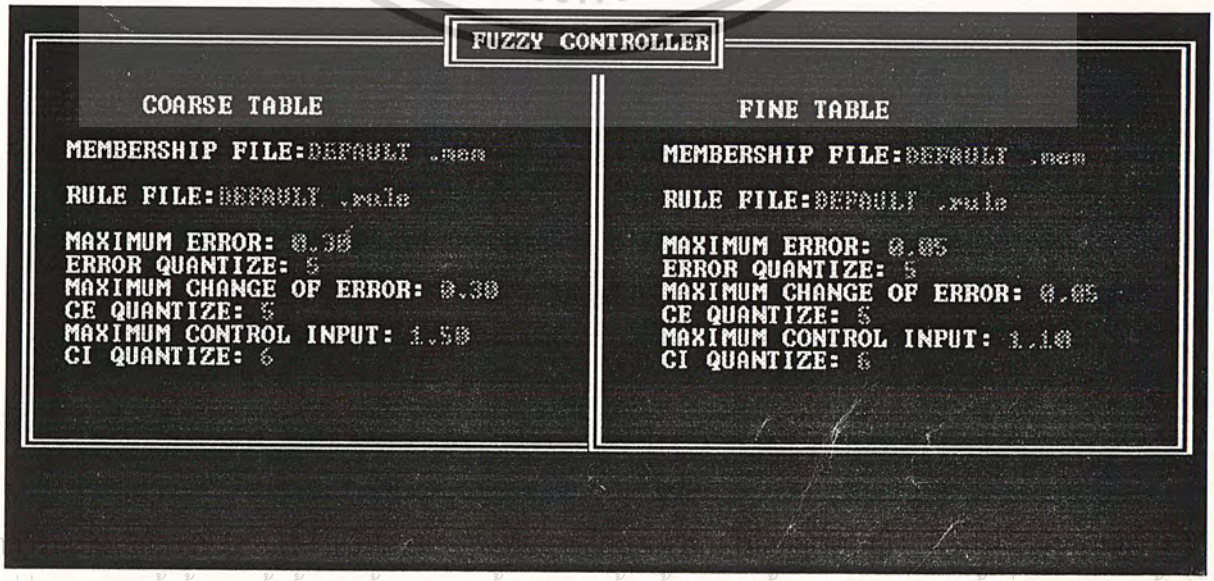
2. Menu CONTROLLER สามารถเลือกได้ 3 แบบคือ

1.FUZZY CONTROLLER

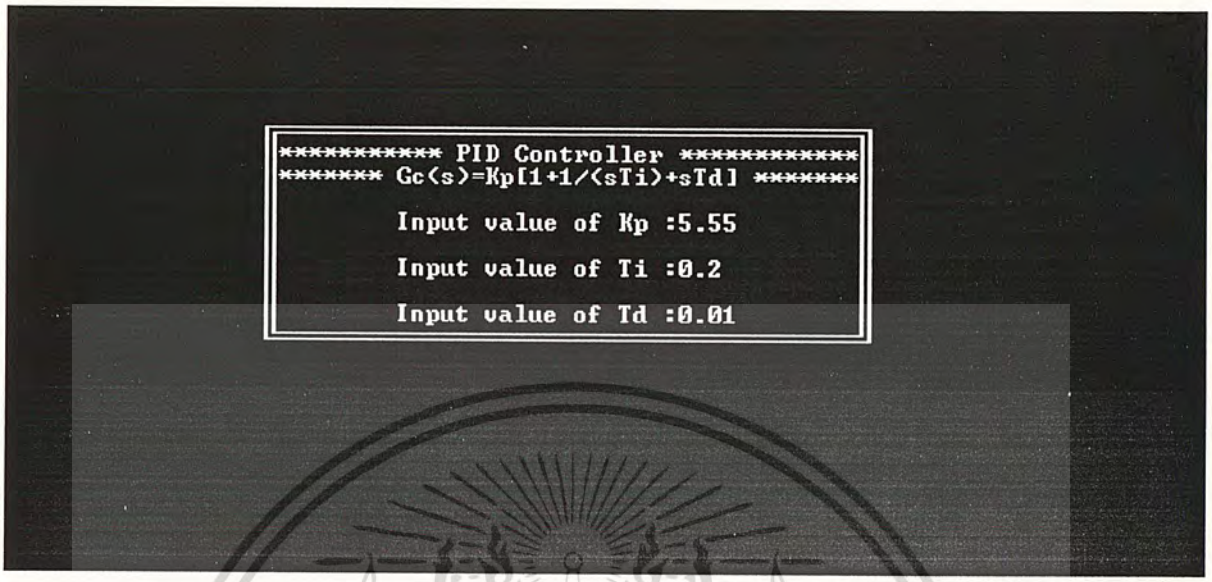
2.PID CONTROLLER

3.FUZZY CONTROLLER AND PID CONTROLLER

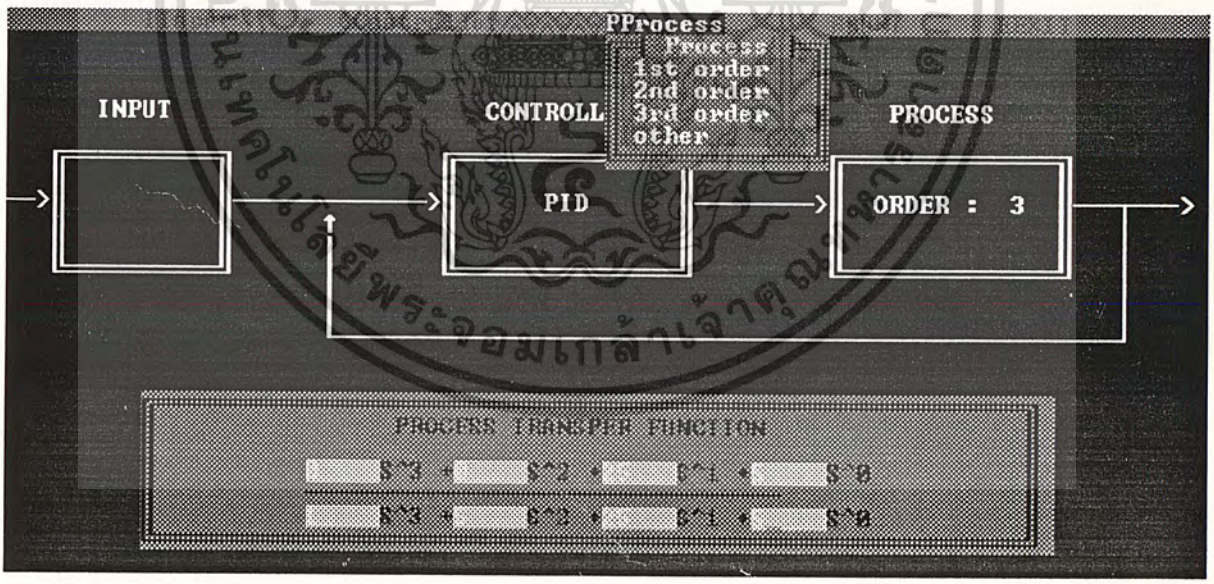
ซึ่งถ้าเลือกเข้าสู่ FUZZY CONTROLLER ก็จะมี parameter ต่างๆ ให้ป้อนดังภาพ



ถ้าเลือกเข้าสู่ PID CONTROLLER ก็จะมี parameter ต่างๆ ให้ป้อนดังภาพ



3. Menu PROCESS จะสามารถเลือกระบบอันดับต่างๆ ได้ และ จะปรากฏหน้าต่างสำหรับการป้อนค่าสัมประสิทธิ์ของ numerator และ denominator ดังภาพ



4. Menu **RUN** จะสามารถเลือกได้ 2 แบบ คือ

1. SIMULATE เป็นการจำลองผลตอบสนองของระบบบน personal computer ซึ่งเมื่อเลือกการ RUN แบบนี้จะต้องป้อนค่าสัมประสิทธิ์ของ numerator และ denominator ของ process ไว้เรียบร้อย โปรแกรมก็จะให้ป้อนค่าเวลาสูงสุดที่จะทำการ simulate และ ค่าสูงสุดของผลตอบสนอง จากนั้นก็จะ plot ผลตอบสนองออกมาทางจอภาพ

2. RUN เป็นการใช้ personal compute ทำหน้าที่ Fuzzy controller จริง ๆ ซึ่งเมื่อเลือกการ RUN แบบนี้ค่าสัมประสิทธิ์ของ numerator และ denominator ของ process ไม่จำเป็นต้องป้อนค่าเอาไว้เพราะ process จะได้มาจากการปรับค่าที่ Simulator โปรแกรมจะให้ป้อนค่า Sampling time และทำงานจนกว่าผู้ใช้จะกด key ใด ๆ

5. Menu **EXIT** ใช้เมื่อต้องการออกจากโปรแกรม

```
/*****************************************************************/
/*                                                               */
/*                    FUZZY CONTROL SYSTEM PROJECT               */
/*                                                               */
/*  By Kamol   Kieawsrikul Code 33100002 Control Engineering KMITL  */
/*     Thavida Maneewarn   Code 33100116 Control Engineering KMITL  */
/*                                                               */
/*****************************************************************/

#define MAX 25
#define NUM_STATE 5
#define BACKGND BLUE
#define LN 0
#define SN 1
#define ZE 2
#define SP 3
#define LP 4
#define BL 5
#define SCALEMAX 550

#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <stdarg.h>
#include <ctype.h>
#include <bios.h>
#include <dir.h>
#include <dos.h>
#include <string.h>
#include <math.h>
#include "..\work\back.h"
#include "..\work\fuzzy.h"
#include "..\work\response.h"
#include "..\work\graph.h"
#include "..\work\sim.h"
#include "..\work\run.h"
#include "..\work\tf.h"
#include "..\tcu\tcu.h"

/*****************************************************************/
/*          Declare static variables in program                  */
/*****************************************************************/


int  input = 0, controller = 0, process = 0;
int  *cursor;
int  th, tw, midx, midy, txpos, typos;
int  block, top, bottom, left, right, startx, starty, endx, endy;
int  curx, cury, cur_part, next_part, s_key, m_edit, r_edit, stop,
                q_max, load;
int  xpos[16];
int  ypos[16];
float old, new;
char str[3];
char cmem_file[10]={"DEFAULT"};
char crul_file[10]={"DEFAULT"};
char fmem_file[10]={"DEFAULT"};
char frul_file[10]={"DEFAULT"};
char *state[7] = {"LN","SN","ZE","SP","LP","  "};
int  r_max = 9;
int  tr_max ;
int  rule[5][5] = {
     {LP, BL, BL, BL, BL},
     {BL, SP, SP, BL, BL},
     {BL, SP, ZE, SN, BL},
     {BL, BL, SN, SN, BL},
     {BL, BL, BL, BL, LN}};
int  cqer_max = 5;
int  cqce_max = 5;
int  cqci_max = 6;
int  fqer_max= 5, fqce_max= 5, fqci_max=6;
int  qer_max, qce_max, qci_max;
float  c_er[MAX][MAX] = {
        /*-5    -4    -3    -2    -1     0     1     2     3     4     5*/
  /*LN*/ {1.0, 0.8, 0.2, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
  /*SN*/ {0.2, 0.5, 0.8, 1.0, 0.8, 0.5, 0.2, 0.0, 0.0, 0.0, 0.0},
  /*ZE*/ {0.0, 0.0, 0.2, 0.5, 0.8, 1.0, 0.8, 0.5, 0.2, 0.0, 0.0},
  /*SP*/ {0.0, 0.0, 0.0, 0.0, 0.2, 0.5, 0.8, 1.0, 0.8, 0.5, 0.2},
  /*LP*/ {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2, 0.5, 0.8, 1.0}};
float  c_ce[MAX][MAX] = {
        /*-5    -4    -3    -2    -1     0     1     2     3     4     5*/
  /*LN*/ {1.0, 0.7, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
  /*SN*/ {0.0, 0.3, 0.7, 1.0, 0.7, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0},
  /*ZE*/ {0.0, 0.0, 0.0, 0.0, 0.3, 0.7, 1.0, 0.7, 0.3, 0.0, 0.0},
  /*SP*/ {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3, 0.7, 1.0, 0.7, 0.3},
  /*LP*/ {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3, 0.7, 1.0}};
float  c_ci[MAX][MAX] = {
       /*-6   -5   -4   -3   -2   -1    0    1    2    3    4    5    6*/
  /*LN*/ {1.0,0.7,0.3,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0},
  /*SN*/ {0.0,0.3,0.7,1.0,0.7,0.3,0.0,0.0,0.0,0.0,0.0,0.0,0.0},
  /*ZE*/ {0.0,0.0,0.0,0.0,0.0,0.3,0.7,1.0,0.7,0.3,0.0,0.0,0.0},
  /*SP*/ {0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.3,0.7,1.0,0.7,0.3},
  /*LP*/ {0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.3,0.7},
float  f_er[MAX][MAX] = {
        /*-5    -4    -3    -2    -1     0     1     2     3     4     5*/
  /*LN*/ {1.0, 0.8, 0.2, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
  /*SN*/ {0.2, 0.5, 0.8, 1.0, 0.8, 0.5, 0.2, 0.0, 0.0, 0.0, 0.0},
  /*ZE*/ {0.0, 0.0, 0.2, 0.5, 0.8, 1.0, 0.8, 0.5, 0.2, 0.0, 0.0},
  /*SP*/ {0.0, 0.0, 0.0, 0.0, 0.2, 0.5, 0.8, 1.0, 0.8, 0.5, 0.2},
  /*LP*/ {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2, 0.5, 0.8, 1.0}};
float  f_ce[MAX][MAX] = {
        /*-5    -4    -3    -2    -1     0     1     2     3     4     5*/
  /*LN*/ {1.0, 0.7, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
  /*SN*/ {0.0, 0.3, 0.7, 1.0, 0.7, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0},
  /*ZE*/ {0.0, 0.0, 0.0, 0.0, 0.3, 0.7, 1.0, 0.7, 0.3, 0.0, 0.0},
  /*SP*/ {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3, 0.7, 1.0, 0.7, 0.3},
  /*LP*/ {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3, 0.7, 1.0}};
```



```
  /*SN*/ {0.0, 0.3, 0.7, 1.0, 0.7, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0},
  /*ZE*/ {0.0, 0.0, 0.0, 0.3, 0.7, 1.0, 0.7, 0.3, 0.0, 0.0, 0.0},
  /*SP*/ {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3, 0.7, 1.0, 0.7, 0.3},
  /*LP*/ {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3, 0.7, 1.0}};
float  f_ci[MAX][MAX] = {
        /*-6   -5   -4   -3   -2   -1    0    1    2    3    4    5    6*/
  /*LN*/ {1.0,0.7,0.3,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0},
  /*SN*/ {0.0,0.3,0.7,1.0,0.7,0.3,0.0,0.0,0.0,0.0,0.0,0.0,0.0},
  /*ZE*/ {0.0,0.0,0.0,0.0,0.0,0.3,0.7,1.0,0.7,0.3,0.0,0.0,0.0},
  /*SP*/ {0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.3,0.7,1.0,0.7,0.3,0.0},
  /*LP*/ {0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.3,0.7,1.0}};
float  f_table[MAX][MAX], c_table[MAX][MAX];
float  cmax_er = 0.3, cmax_ce = 0.3, cmax_ci = 1.50;
float  fmax_er= 0.05, fmax_ce=0.05, fmax_ci=1.10;
float  num[MAX],denom[MAX],newnum[MAX+2],newdenom[MAX+2];
float  Kp,Ti,Td;
float  ci_dat[SCALEMAX];
int    order;
int    MaxX,MaxY,gapX,gapY;
int    count,number,check;
float  x[MAX];

/*****************************************************************/
/*            Main program for FUZZY CONTROL SYSTEM              */
/*****************************************************************/


int main(void)
{
/*****************************************************************/
/*                  Declare Menu variables                       */
/*****************************************************************/

   char *headings[] = { "IInput", "CController", "PProcess",
                        "RRUN", "EEXIT", NULL };
   TCU_MENU          input_m,
                     controller_m,
                     process_m,
                     run_m ,
                    *m[list[5];
   TCU_FORM          form;
   TCU_FIELD_VALUE   frm_val;
   int               message,
                     menu,
                     option,
                     r1,r2;
   char              msg[26];
   TCU_PULLDOWN      pd;

   char              *input_options[] =
                { "SStep", "RRAmp", NULL };

   char              *controller_options[] =
                { "Fuzzy", "PID", "Fuzzy_PID", NULL };

   char              *process_options[] =
                { "1st order","2nd order","3rd order","other", NULL };
   char              *run_options[] =
                { "Simulate","Run", NULL};
/*****************************************************************/
/*                  Initialize pulldown menu                     */
/*****************************************************************/

   tcu_set_mouse_mode (1);

   r1 = tcu_define_menu (&input_m,
                         "Input",
         tcu_colour_attrib (YELLOW, CYAN),       /* Title */
         tcu_colour_attrib (BLUE, CYAN),         /* Box */
         tcu_colour_attrib (WHITE, MAGENTA),     /* Options */
         tcu_colour_attrib (WHITE, BLUE),        /* Selected */
         tcu_colour_attrib (LIGHTGRAY, MAGENTA), /* Deselected */
         TCU_ESC_ESC | TCU_ESC_CNTL_C | TCU_ESC_PGUP | TCU_ESC_FUNC,
         TCU_BOX_DOUBLE,
         input_options,
         tcu_colour_attrib (LIGHTRED, MAGENTA));

   r1 = tcu_define_menu (&controller_m,
                         "Controller",
         tcu_colour_attrib (YELLOW, CYAN),       /* Title */
         tcu_colour_attrib (BLUE, CYAN),         /* Box */
         tcu_colour_attrib (WHITE, MAGENTA),     /* Options */
         tcu_colour_attrib (WHITE, GREEN),        /* Selected */
         tcu_colour_attrib (LIGHTGRAY, MAGENTA), /* Deselected */
         TCU_ESC_ESC | TCU_ESC_CNTL_C | TCU_ESC_PGUP | TCU_ESC_FUNC,
         TCU_BOX_DOUBLE,
         controller_options,
                         0);

   r1 = tcu_define_menu (&process_m,
                         "Process",
         tcu_colour_attrib (YELLOW, CYAN),       /* Title */
         tcu_colour_attrib (BLUE, CYAN),         /* Box */
         tcu_colour_attrib (WHITE, MAGENTA),     /* Options */
         tcu_colour_attrib (WHITE, GREEN),        /* Selected */
         tcu_colour_attrib (LIGHTGRAY, MAGENTA), /* Deselected */
         TCU_ESC_ESC |  TCU_ESC_CNTL_C | TCU_ESC_PGUP | TCU_ESC_FUNC,
         TCU_BOX_DOUBLE,
         process_options,
                         0);

   r1 = tcu_define_menu (&run_m,
                         "RUN",
         tcu_colour_attrib (YELLOW, CYAN),       /* Title */
         tcu_colour_attrib (BLUE, CYAN),         /* Box */
         tcu_colour_attrib (WHITE, MAGENTA),     /* Options */
         tcu_colour_attrib (WHITE, GREEN),        /* Selected */
         tcu_colour_attrib (LIGHTGRAY, MAGENTA), /* Deselected */
         TCU_ESC_ESC |  TCU_ESC_CNTL_C | TCU_ESC_PGUP | TCU_ESC_FUNC,
```

```
        TCU_BOX_DOUBLE,
        run_options,
                0);


   mlist[0] = &input_m;
   mlist[1] = &controller_m;
   mlist[2] = &process_m;
   mlist[3] = &run_m;
   mlist[4] = NULL;

   tcu_define_pulldown    (&pd,
                           tcu_colour_attrib (WHITE,CYAN),
                           tcu_colour_attrib (MAGENTA,CYAN),
                           tcu_colour_attrib (YELLOW,BLUE),
                           headings,
                           tcu_colour_attrib (BLUE, CYAN),
                            mlist);

   tcu_load_form (&form, "pulldown");

   tcu_get_field_id (&form, "menu", &menu);
   tcu_get_field_id (&form, "option", &option);
   tcu_get_field_id (&form, "message", &message);

   tcu_set_field_mode (&form, menu, TCU_FORM_NOENTER);
   tcu_set_field_mode (&form, option, TCU_FORM_NOENTER);
   tcu_set_field_mode (&form, message, TCU_FORM_CONFIRM);

   tcu_display_form (&form, 1, 2);

   tcu_set_menu_option (&input_m, 2, 0);
   tcu_set_menu_option (&run_m, 1, 0);
   tcu_set_menu_option (&run_m, 2, 0);

/*****************************************************************/

/*                  Create user interface screen                 */

/*****************************************************************/

   Background();

   do {
      tcu_read_pulldown_selection (&pd, &r1, &r2);
      if (r1) {
         frm_val.v_int = r1;
         tcu_put_field (&form, menu, &frm_val);
         frm_val.v_int = r2;
         tcu_put_field (&form, option, &frm_val);
         if (r2)
            msg[0] = '\0';
         else
            strcpy (msg, "No sub-menu available");
         frm_val.v_string = msg;
         tcu_put_field (&form, message, &frm_val);

         switch(r1){
             case 1: if(r2 == 1){
                        input = 1;
                        gotoxy(7,9);
                        cprintf("STEP");
                       }
                     /*else
                        input = ramp*/
                     break;
             case 2: switch(r2){
                          case 1: controller = 1;
                                  gotoxy(36,9);
                                  cprintf("FUZZY");
                                  Fuzzy_controller();
                                  break;
                          case 2: controller = 2;
                                  gotoxy(37,9);
                                  cprintf("PID");
                                  PID_controller();
                                  break;
                          case 3: controller = 3;
                                  gotoxy(34,9);
                                  cprintf("FUZZY-PID");
                                  Fuzzy_controller();
                                  PID_controller();
                                  }
                     Background();
                     break;
             case 3: switch(r2){
                          case 1: order = 1;
                                  gotoxy(68,9);
                                  cprintf("1");
                                  break;
                          case 2: order = 2;
                                  gotoxy(68,9);
                                  cprintf("2");
                                  break;
                          case 3: order = 3;
                                  gotoxy(68,9);
                                  cprintf("3");
                                  break;
                          case 4: order = Getorder();
                          }
                     Getinput();
                     process = 1;
                     break;
             case 4: switch(r2){
                      case 1: sim(controller);
                              Background();
                              break;
                      case 2: run();
                              Background();
                              break;
                             }
                     break;
         }
```



```
         if(input!=0 && controller!=0)
            tcu_set_menu_option(&run_m, 2, 1);
         if(input!=0 && controller!=0 && process!=0)
            tcu_set_menu_option (&run_m, 1, 1);
      }
   }while (r1 != 5);

   tcu_remove_pulldown(&pd);
   tcu_remove_form (&form);
   textbackground(BLACK);
   clrscr();
   return(0);
}
/*****************************************************************/

/*                          END OF PROGRAM                       */

/*****************************************************************/
```

back.h

```
void draw_block(int, int, int, int);
void Background(void);

void Background(void)
{ extern int input,controller,process,order;
  int cnt;

  textbackground(BLUE);
  textcolor(WHITE);
  clrscr();
  draw_block(4,7,15,12);
  draw_block(30,7,46,12);
  draw_block(56,7,71,12);

  gotoxy(7,5);
  cprintf("INPUT");
  if(input == 1){
     gotoxy(7,9);
     cprintf("STEP");
     }
  gotoxy(33,5);
  cprintf("CONTROLLER");
  if(controller!=0){
     gotoxy(36,9);
     switch(controller){
            case 1: cprintf("FUZZY");
                    break;
            case 2: cprintf(" PID");
                    break;
            case 3: gotoxy(34,9);
                    cprintf("FUZZY-PID");
            }
     }
  gotoxy(60,5);
  cprintf("PROCESS");
  gotoxy(59,9);
  cprintf("ORDER : ");
  if(process!=0)
     cprintf("%d",order);

  gotoxy(1,9);
  for(cnt=1;cnt<3;cnt++)
      putch(0xC4);
  putch(0x3E);
  gotoxy(16,9);
  for(cnt=16;cnt<29;cnt++)
      putch(0xC4);
  putch(0x3E);
  gotoxy(47,9);
  for(cnt=47;cnt<55;cnt++)
      putch(0xC4);
  putch(0x3E);
  gotoxy(72,9);
  for(cnt=72;cnt<79;cnt++)
      putch(0xC4);
  putch(0x3E);
  gotoxy(75,9);
  putch(0xC2);
  for(cnt=10;cnt<15;cnt++){
      gotoxy(75,cnt);
      putch(0xB3);
      }
  gotoxy(75,15);
  putch(0xD9);
  gotoxy(22,15);
  putch(0xC0);
  for(cnt=22;cnt<74;cnt++)
      putch(0xC4);
  gotoxy(22,10);
  putch(0x18);
  for(cnt=11;cnt<15;cnt++){
      gotoxy(22,cnt);
      putch(0xB3);
      }

}

void draw_block(int left,int top,int right,int bottom)
{ int cnt1;

  gotoxy(left,top);
  putch(0xC9);
  for(cnt1=left+1; cnt1<right; cnt1++)
      putch(0xCD);
  putch(0xBB);
  putch('\n');
  for(cnt1=top+1; cnt1<bottom; cnt1++){
      gotoxy(left, cnt1);
      putch(0xBA);
      gotoxy(right, cnt1);
      putch(0xBA);
      putch('\n');
     }
  gotoxy(left,bottom);
  putch(0xC8);
  for(cnt1=left+1; cnt1<right; cnt1++)
      putch(0xCD);
  putch(0xBC);
}
```

```
/*****************************************************************/
/*                        Define constant                        */
/*****************************************************************/

#define MAX 25
#define NUM_STATE 5
#define BACKGND BLUE
#define LN 0
#define SN 1
#define ZE 2
#define SP 3
#define LP 4
#define BL 5
/*****************************************************************/
/*                     Define routines in fuzzy.h                */
/*****************************************************************/

void Initialize(void);
void gprintxy(int, int,char *,...);
void erasech(int, int, int);
void make_cursor(int);
void move_cursor(int, int);
void draw_table(void);
void default_table(void *);
void draw_member_graph(void);
void default_member_graph(void *);
void Show_fuzzy_table(void *);
void edit_member(void *);
void edit_graph(void *);
void draw_rule(void);
void default_rule(void *);
void ruleprint(int, int, int);
void edit_rule(void *);
void save_mem_file(void *, void *, void *, char);
void save_rule_file(void *);
void load_mem(char *, char);
void load_rule(char *);
void view_fuzzy(void);
float get_float(void);
int  get_int(void);
int  Membership(char);
void Title(void);

/*****************************************************************/
/*                        Extern variables                       */
/*****************************************************************/

extern int  *cursor;
extern int  th, tw, midx, midy, MaxX, MaxY, txpos, typos;
extern int  block, top, bottom, left, right, startx, starty,
            endx, endy;
extern int  curx, cury, cur_part, next_part, s_key, m_edit, r_edit,
            stop, q_max, load;
extern int  xpos[16];
extern int  ypos[16];
extern float old, new;
extern char str[3];
extern char cmem_file[10],crul_file[10],fmem_file[10],frul_file[10];
extern char *state[7];
extern int  r_max, tr_max;
extern int  rule[5][5];
extern int  cqer_max, cqce_max,cqci_max,fqer_max, fqce_max, fqci_max;
extern int  qer_max, qce_max, qci_max;
extern float  c_er[MAX][MAX], c_ce[MAX][MAX],c_ci[MAX][MAX];
extern float  f_er[MAX][MAX], f_ce[MAX][MAX], f_ci[MAX][MAX];
extern float  cmax_er,cmax_ce,cmax_ci;
extern float  fmax_er, fmax_ce, fmax_ci;

/*****************************************************************/
/*   Routine Fuzzy_controller : user interface screen for input  */
/*                              Fuzzy-controller parameter such  */
/*                              as name of membership and  rule  */
/*                              file and maximum value of error, */
/*                              change of error, control input   */
/*****************************************************************/

void Fuzzy_controller(void)
{
  char  fname[10];
  char  t_fname[10];
  char  *type;
  float f_buff;
  int   i_buff;
  int  cnt;

  clrscr();
  Title();
  getch();
  m_edit=0;
  view_fuzzy();
  textcolor(WHITE);
  gotoxy(xpos[0],ypos[0]);
  type = ".mem";
  for(cnt=0;cnt<10;cnt++)
      fname[cnt] = '';
  gets(fname);
  if(fname[0] != ''){
     gotoxy(xpos[0],ypos[0]);
     cprintf("          ");
     gotoxy(xpos[0],ypos[0]);
     cprintf("%s",fname);
     sprintf(t_fname,"%s",fname);
     strcat(fname,type);
```



```
     load_mem(fname,'c');
     if(load==1){
        sprintf(cmem_file,"%s",t_fname);
        gotoxy(xpos[3],ypos[3]);
        cprintf("%d",cqer_max);
        gotoxy(xpos[5],ypos[5]);
        cprintf("%d",cqce_max);
        gotoxy(xpos[7],ypos[7]);
        cprintf("%d",cqci_max);
       }
     else{
        gotoxy(xpos[0],ypos[0]);
        textcolor(CYAN);
        cprintf("DEFUALT");
       }
   }

  textcolor(WHITE);
  gotoxy(xpos[1],ypos[1]);
  type = ".rule";
  for(cnt=0;cnt<10;cnt++)
      fname[cnt] = 0;
  gets(fname);
  if(fname[0] != ''){
     gotoxy(xpos[1],ypos[1]);
     cprintf("          ");
     gotoxy(xpos[1],ypos[1]);
     cprintf("%s",fname);
     sprintf(t_fname,"%s",fname);
     strcat(fname,type);
     load_rule(fname);
     if(load==1){
        gotoxy(xpos[9],ypos[9]);
        cprintf("          ");
        sprintf(crul_file,"%s",t_fname);
        sprintf(frul_file,"%s",t_fname);
        gotoxy(xpos[9],ypos[9]);
        cprintf("%s",frul_file);
       }
     else{
        gotoxy(xpos[1],ypos[1]);
        textcolor(CYAN);
        cprintf("DEFAULT");
       }
   }
  textcolor(WHITE);
  gotoxy(xpos[2],ypos[2]);
  f_buff = get_float();
  if(f_buff!=0)
     cmax_er = f_buff;

  gotoxy(xpos[3],ypos[3]);
  i_buff = get_int();
  if(i_buff!=0)
     qer_max = i_buff;
  else
     qer_max = cqer_max;

  gotoxy(xpos[4],ypos[4]);
  f_buff = get_float();
  if(f_buff!=0)
     cmax_ce = f_buff;

  gotoxy(xpos[5],ypos[5]);
  i_buff = get_int();
  if(i_buff!=0)
     qce_max = i_buff;
  else
     qce_max = cqce_max;

  gotoxy(xpos[6],ypos[6]);
  f_buff = get_float();
  if(f_buff!=0)
     cmax_ci = f_buff;

  gotoxy(xpos[7],ypos[7]);
  i_buff = get_int();
  if(i_buff!=0)
     qci_max = i_buff;
  else
     qci_max = cqci_max;

  Membership('c');

  view_fuzzy();
  textcolor(WHITE);
  gotoxy(xpos[8],ypos[8]);
  type = ".mem";
  for(cnt=0;cnt<10;cnt++)
      fname[cnt] = 0;
  gets(fname);
  if(fname[0] != ''){
     gotoxy(xpos[8],ypos[8]);
     cprintf("          ");
     gotoxy(xpos[8],ypos[8]);
     cprintf("%s",fname);
     sprintf(t_fname,"%s",fname);
     strcat(fname,type);
     load_mem(fname,'f');
     if(load==1){
        sprintf(fmem_file,"%s",t_fname);
        gotoxy(xpos[11],ypos[11]);
        cprintf("%d",fqer_max);
        gotoxy(xpos[13],ypos[13]);
        cprintf("%d",fqce_max);
        gotoxy(xpos[15],ypos[15]);
        cprintf("%d",fqci_max);
       }
     else{
        gotoxy(xpos[8],ypos[8]);
        textcolor(CYAN);
        cprintf("DEFUALT");
       }
   }
```

```
  textcolor(WHITE);
  gotoxy(xpos[10],ypos[10]);
  f_buff = get_float();
  if(f_buff!=0)
     fmax_er = f_buff;

  gotoxy(xpos[11],ypos[11]);
  i_buff = get_int();
  if(i_buff!=0)
     qer_max = i_buff;
  else
     qer_max = fqer_max;

  gotoxy(xpos[12],ypos[12]);
  f_buff = get_float();
  if(f_buff!=0)
     cmax_ce = f_buff;

  gotoxy(xpos[13],ypos[13]);
  i_buff = get_int();
  if(i_buff!=0)
     qce_max = i_buff;
  else
     qce_max = fqce_max;

  gotoxy(xpos[14],ypos[14]);
  f_buff = get_float();
  if(f_buff!=0)
     fmax_ci = f_buff;

  gotoxy(xpos[15],ypos[15]);
  i_buff = get_int();
  if(i_buff!=0)
     qci_max = i_buff;
  else
     qci_max = fqci_max;

  Membership('f');
  view_fuzzy();
  getch();

  textbackground(BLACK);
  clrscr();
}
/*****************************************************************/

/*  Routine Membership: routine for show or edit fuzzy membership-  */

/*                      table and graph for error, change of error  */

/*                      and control input                           */

/*****************************************************************/

int Membership(char cof)
{
  float er[MAX][MAX],ce[MAX][MAX],ci[MAX][MAX];
  int   t_rule[5][5];
  int   cnt1, cnt2;

  union inkey{
              char ch[2];
              int i;
              } c;

  if(!m_edit){
     switch(cof){
         case 'c': for(cnt1=0; cnt1<MAX; cnt1++)
                   for(cnt2=0; cnt2<MAX; cnt2++){
                     er[cnt1][cnt2] = c_er[cnt1][cnt2];
                     ce[cnt1][cnt2] = c_ce[cnt1][cnt2];
                     ci[cnt1][cnt2] = c_ci[cnt1][cnt2];
                     }
                   break;
         case 'f': for(cnt1=0; cnt1<MAX; cnt1++)
                   for(cnt2=0; cnt2<MAX; cnt2++){
                     er[cnt1][cnt2] = f_er[cnt1][cnt2];
                     ce[cnt1][cnt2] = f_ce[cnt1][cnt2];
                     ci[cnt1][cnt2] = f_ci[cnt1][cnt2];
                     }
                   qer_max = fqer_max;
                   qce_max = fqce_max;
                   qci_max = fqci_max;
                 }
     }
  else{
       for(cnt1=0; cnt1<MAX; cnt1++)
       for(cnt2=0; cnt2<MAX; cnt2++){
          er[cnt1][cnt2] = 0;
          ce[cnt1][cnt2] = 0;
          ci[cnt1][cnt2] = 0;
        }
      }
  for(cnt1=0;cnt1<5;cnt1++)
  for(cnt2=0;cnt2<5;cnt2++)
      t_rule[cnt1][cnt2] = rule[cnt1][cnt2];
  tr_max = r_max;
  textbackground(BLUE);
  clrscr();
  Initialize();
  cleardevice();
  tw = textwidth("W");
  th = textheight("W");
  MaxX = getmaxx();
  MaxY = getmaxy();
  midx = getmaxx()/2;
  midy = getmaxy()/2;
  make_cursor(RED);
  settextjustify(LEFT_TEXT,TOP_TEXT);
  setbkcolor(BACKGND);
  cur_part = 0;
  next_part = 0;
  stop  = 0;
  s_key = 0;
  r_edit= 0;
```



```
  q_max = qer_max;
  Show_fuzzy_table(er);
  while(!stop){
  if(cur_part != next_part){
     cleardevice();
     switch(next_part){
            case 0: cur_part = 0;
                    q_max = qer_max;;
                    Show_fuzzy_table(er);
                    break;
            case 1: cur_part = 1;
                    q_max = qce_max;
                    Show_fuzzy_table(ce);
                    break;
            case 2: cur_part = 2;
                    q_max = qci_max;
                    Show_fuzzy_table(ci);
                    break;
            case 3: cur_part = 3;
                    q_max = 3;
                    draw_rule();
                    default_rule(t_rule);
            }
     s_key = 0;
     }
  while(!s_key){
      while(!bioskey(1));
      c.i = bioskey(0);
      if(c.ch[0]==0){
        switch(c.ch[1]){
      /*up arrow*/ case 72 :if(cury!=0){
                            move_cursor(curx,cury-1);
                            cury = cury-1;
                            }
                           break;
    /*down arrow*/ case 80 :if(cury!=endy){
                            move_cursor(curx,cury+1);
                            cury = cury+1;
                            }
                           break;
  /*right arrow*/ case 77 :if(curx!=endx){
                            move_cursor(curx+1,cury);
                            curx = curx+1;
                            }
                           else
                            if(cury != endy){
                             move_cursor(0,cury+1);
                             curx = 0;
                             cury = cury+1;
                             }
                           break;
    /*left arrow*/ case 75 :if(curx!=0){
                            move_cursor(curx-1,cury);
                            curx = curx-1;
                            }
                           else
                            if(cury != 0){
                             move_cursor(endx,cury-1);
                             curx = endx;
                             cury = cury-1;
                             }
                           break;
                 /*F1*/ case 59 :next_part = 0;
                           s_key = 1;
                           break;
                 /*F2*/ case 60 :next_part = 1;
                           s_key = 1;
                           break;
                 /*F3*/ case 61 :next_part = 2;
                           s_key = 1;
                           break;
                 /*F4*/ case 62 :next_part = 3;
                           s_key = 1;
      }
     }
    else{
        switch(c.ch[0]){
          /*enter*/ case '\r':if(curx!=endx){
                             move_cursor(curx+1,cury);
                             curx = curx+1;
                             }
                            else
                             if(cury!=endy){
                                 move_cursor(0,cury+1);
                                 curx = 0;
                                 cury = cury+1;
                               }
                             else{
                                 move_cursor(0,0);
                                 curx = 0;
                                 cury = 0;
                                 }


                           break;
    /*space bar*/ case ' ' :if(cur_part!=3){
                            if(curx!=endx){
                              move_cursor(curx+1,cury);
                              curx = curx+1;
                             }
                             else{
                              move_cursor(0,cury+1);
                              curx = 0;
                              cury = cury+1;
                             }
                           }
                           else{
                             r_edit = 1;
                             setcolor(YELLOW);
                             edit_rule(t_rule);
                             if(curx != endx){
                                move_cursor(curx+1,cury);
                                curx = curx+1;
                                }
```

```
                        else
                           if(cury != endy){
                              move_cursor(0,cury+1);
                              cury = cury+1;
                              curx = 0;
                             }
                           else{
                              move_cursor(0,0);
                              curx = 0;
                              cury = 0;
                             }
                        }
                      break;
      /*0 or 1*/ case '0' :
                 case '1' :if(cur_part!=3){
                        for(cnt1=0;cnt1<3;cnt1++)
                            str[cnt1] = 0;
                        m_edit = 1;
                        str[0] = c.ch[0];
                        setcolor(YELLOW);
                        gprintxy(startx+2+curx*block,
                                 starty+1+cury*3*tw,str);
                        switch(cur_part){
                            case 0:edit_member(er);
                                   edit_graph(er);
                                   break;
                            case 1:edit_member(ce);
                                   edit_graph(ce);
                                   break;
                            case 2:edit_member(ci);
                                   edit_graph(ci);
                                   break;
                        }
                        if(curx != endx){
                          move_cursor(curx+1,cury);
                          curx = curx+1;
                          }
                        else
                          if(cury != endy){
                            move_cursor(curx,cury+1);
                            cury = cury+1;
                            }
                        }
                       else
                        putchar(0x07);
                       break;
                 case 27  :stop = 1;
                           s_key = 1;
                           break;
                 default  :putchar(0x07);
          }
        }
      }
    }
  if(m_edit)
     save_mem_file(er,ce,ci,cof);
  if(r_edit)
     save_rule_file(t_rule);
  m_edit=0;
  r_edit=0;
  stop = 0;
  free(cursor);
  closegraph();
  return(0);
}

/*****************************************************************/

/*  Routine Erasestr: for erase string in graphics mode          */

/*****************************************************************/

void erasestr(int xloc, int yloc, char *str)
{
  struct textsettingstype textinfo;
  int    xdim, ydim;
  void   *textimage;
  gettextsettings(&textinfo);
  switch (textinfo.direction)
  {
    case HORIZ_DIR: xdim = textwidth(str);
                    ydim = textheight(str);
                    yloc--; break;
    case VERT_DIR : ydim = textwidth(str);
                    xdim = textheight(str);
                    yloc++;
  }
  switch(textinfo.horiz)
  {
    case   LEFT_TEXT:                  break;
    case CENTER_TEXT: xloc -= xdim/2;  break;
    case  RIGHT_TEXT: xloc -= xdim;
  }
  switch (textinfo.vert)
  {
    case BOTTOM_TEXT: yloc -= ydim;    break;
    case CENTER_TEXT: yloc -= ydim;    break;
    case    TOP_TEXT: ;
  }
  while (xloc<0) { xloc++; xdim--; }
  while (yloc<0) { yloc++; ydim--; }
  textimage = malloc(imagesize(xloc, yloc, xdim, ydim));
  getimage(xloc, yloc, xloc+xdim, yloc+ydim, textimage);
  putimage(xloc, yloc, textimage, XOR_PUT);
  free(textimage);
}

/*****************************************************************/

/*  Routine gprintxy : print text to screen in graphics mode     */

/*****************************************************************/

void gprintxy(int xloc, int yloc, char *fmt, ...)
{
```



```
  va_list argptr;
  char    str[140];
  struct  textsettingstype  textinfo;
  va_start(argptr, fmt);
  vsprintf(str, fmt, argptr);
  gettextsettings(&textinfo);
  erasestr(xloc, yloc, str);
  outtextxy(xloc, yloc, str);
  va_end(argptr);
}

/*****************************************************************/

/*  Routine erasech : erase character stream in graphics mode    */

/*****************************************************************/

void erasech(int xloc, int yloc,int num)
{
  int  cnt1, xdim, ydim;
  void *textimage;

  for(cnt1=0,xdim=0;cnt1<num;cnt1++)
     xdim = xdim+textwidth("W");
  ydim = textheight("W");
  textimage = malloc(imagesize(xloc, yloc, xdim, ydim));
  getimage(xloc, yloc-1, xloc+xdim, yloc+ydim, textimage);
  putimage(xloc, yloc, textimage, XOR_PUT);
  free(textimage);
}

/*****************************************************************/

/*  Routine move_cursor : for move cursor to any position in     */

/*                        graphics mode                          */

/*****************************************************************/

void move_cursor(int new_curx, int new_cury)
{
  int xpos,ypos;
  int xdest,ydest;

  xpos  = startx+curx*block;
  ypos  = starty+cury*3*tw;
  xdest = startx+new_curx*block;
  ydest = starty+new_cury*3*tw;
  putimage(xpos,ypos,cursor,XOR_PUT);
  putimage(xdest,ydest,cursor,XOR_PUT);

}

/*****************************************************************/

/*  Routine make_cursor : for making a cusor                     */

/*****************************************************************/

void make_cursor(int color)
{

  setcolor(color);
  bar(20,40,20+tw,40+th);
  cursor = malloc(imagesize(20, 40, tw, th));
  getimage(20,40,20+tw,40+th,cursor);
  putimage(20,40,cursor,XOR_PUT);
}

/*****************************************************************/

/*  Routine draw_table : Draw outline of membership function table */

/*****************************************************************/

void draw_table(void)
{
  div_t frac;
  int   cnt1, cnt2, x, y;
  char  key_str[80];

  setcolor(YELLOW);
  switch(cur_part){
        case 0: gprintxy(midx-50,25,"ERROR  TABLE");
                rectangle(midx-54,20,midx+50,35);
                break;
        case 1: gprintxy(midx-80,25,"CHANGE OF ERROR TABLE");
                rectangle(midx-84,20,midx+88,35);
                break;
        case 2: gprintxy(midx-80,25,"CONTROL INPUT TABLE");
                rectangle(midx-85,20,midx+77,35);
        }

  frac = div(MaxX-80,2*q_max+2);
  block = frac.quot;
  if (block<30)
     block = 30;
  setcolor(WHITE);
  x = midx-(q_max+1)*block;
  for(cnt1=0; cnt1<q_max*2+2; cnt1++){
      y = 50;
      for(cnt2=0; cnt2<NUM_STATE+1; cnt2++){
          rectangle(x,y,x+block,y+3*tw);
          y = y+3*tw;
         }
      x = x+block;
      }
  x = midx-q_max*block+10;
  y = 50+10;
  for(cnt1=-q_max; cnt1<=q_max; cnt1++){
      gprintxy(x,y,"%d",cnt1);
      x = x+block;
      }
  x = midx-(q_max+1)*block+10;
  y = 50+30;
```

```
  for(cnt1=0; cnt1<NUM_STATE; cnt1++){
      gprintxy(x,y,state[cnt1]);
      y = y+3*tw;
      }
  setfillstyle(SOLID_FILL, CYAN);
  bar(0,MaxY-30,MaxX,MaxY);
  setcolor(BLUE);
  sprintf(key_str,"F1 ERROR   F2 CHANGE OF ERROR   F3 CONTROL INPUT

                      F4 RULE    Esc EXIT ");
  outtextxy(45,MaxY-20,key_str);
}

/*************************************************************/

/*  Routine default_table:Show default value of membership function */

/*************************************************************/

void default_table(void *d_table)
{
  int   cnt1, cnt2, x, y;
  float (*table)[MAX][MAX];

  table = d_table;
  setcolor(YELLOW);
  x = midx-q_max*block+10;
  for(cnt1=0; cnt1<q_max*2+1; cnt1++){
      y = 50+30;
      for(cnt2=0; cnt2<NUM_STATE; cnt2++){
          gprintxy(x,y,"%3.1f",(*table)[cnt2][cnt1]);
          y = y+3*tw;
          }
      x = x+block;
      }
  startx = midx-q_max*block+8;
  starty = 79;
  putimage(startx, starty, cursor, XOR_PUT);
  curx = 0;
  cury = 0;
  endx = q_max*2;
  endy = NUM_STATE-1;
}

/*************************************************************/

/*  Routine draw_member_graph : draw graph axis for membership     */

/*                              function graph                     */

/*************************************************************/

void draw_member_graph(void)
{
  int x, y, cnt1;

  setcolor(WHITE);
  top = 230;
  left = midx-q_max*block;
  bottom = 400;
  right = midx+q_max*block;
  rectangle(left, top, right, bottom);
  line(midx,bottom,midx,top);
  x = left;
  for(cnt1=0; cnt1<=q_max*2; cnt1++){
      line(x,bottom-2,x,bottom+2);
      gprintxy(x-4,bottom+10,"%d",cnt1-q_max);
      x = x+block;
      }
  y = bottom;
  for(cnt1=0; cnt1<5; cnt1++){
      y = y-30;
      line(midx-3,y,midx+3,y);
      gprintxy(midx-30,y-4,"%3.1f",0.2+cnt1*0.2);
      }
}

/*************************************************************/

/*  Routine default_member_graph : draw line of membership funtion  */

/*                                 in membership function graph    */

/*************************************************************/

void default_member_graph(void *d_table)
{
  float (*table)[MAX][MAX];
  int   color, xpos1, ypos1, xpos2, ypos2;
  int   cnt1, cnt2;

  table = d_table;
  color = 2;
  for(cnt1=0; cnt1<NUM_STATE; cnt1++){
      setcolor(color);
      for(cnt2=0; cnt2<q_max*2; cnt2++){
          xpos1 = left+cnt2*block;
          ypos1 = bottom-(*table)[cnt1][cnt2]*150;
          xpos2 = left+(cnt2+1)*block;
          ypos2 = bottom-(*table)[cnt1][cnt2+1]*150;
          line(xpos1,ypos1,xpos2,ypos2);
          }
      gprintxy(left+(2*cnt1+1)*block, top+10, state[cnt1]);
      color = color+2;
      }
}

/*************************************************************/

/*  Routine Show_fuzzy_table : show fuzzy membership table and     */

/*                             graph on screen                     */

/*************************************************************/
```



```
void Show_fuzzy_table(void *t_table)
{
  float  (*table)[MAX][MAX];
  table = t_table;

  setcolor(WHITE);
  draw_table();
  draw_member_graph();
  default_table((*table));
  default_member_graph((*table));
}

/*************************************************************/
/*  Routine edit_member:for edit data in membership function table */
/*************************************************************/
void edit_member(void *e_table)
{
  float (*table)[MAX][MAX];
  char  temp;

  table = e_table;
  while(getch() != '.')
       putchar(0x07);
  str[1] = '.';
  temp = ' ';
  if(str[0] == '0'){
       while(isdigit(temp)==0){
            temp = getch();
            if(isdigit(temp)==0)
              putchar(0x07);
            }
       str[2] = temp;
     }
  else{
       while(temp != '0'){
            temp = getch();
            if(temp != '0')
              putchar(0x07);
            }
       str[2] = temp;
     }
  gprintxy(startx+2+curx*block,starty+1+cury*3*tw,"%3s",str);
  putimage(startx+curx*block, starty+cury*3*tw, cursor, XOR_PUT);
  old = (*table)[cury][curx];
  (*table)[cury][curx] = atof(str);
  new = (*table)[cury][curx];
  while(getch() != '\r')
       putchar(0x07);
}
/*************************************************************/

/*  Routine edit_graph : For edit line in graph when data in       */

/*                       membership function table was changed     */

/*************************************************************/

void edit_graph(void *e_table)
{
  int o_xmid, o_ymid, o_xleft, o_yleft, o_xright, o_yright;
  int n_xmid, n_ymid;
  float (*table)[MAX][MAX];

  table = e_table;
     o_xmid   = left+curx*block;
     o_ymid   = bottom-old*150;
  if(curx != 0){
     o_xleft  = left+(curx-1)*block;
     o_yleft  = bottom-(*table)[cury][curx-1]*150;
     }
  if(curx != q_max*2+1){
     o_xright = left+(curx+1)*block;
     o_yright = bottom-(*table)[cury][curx+1]*150;
     }
  n_xmid   = left+curx*block;
  n_ymid   = bottom-new*150;
  setcolor(BACKGND);
  if(curx != 0)
    line(o_xleft,o_yleft,o_xmid,o_ymid);
  if(curx != q_max*2)
    line(o_xmid,o_ymid,o_xright,o_yright);
  setcolor(2+cury*2);
  if(curx != 0)
    line(o_xleft,o_yleft,n_xmid,n_ymid);
  if(curx != q_max*2)
    line(n_xmid,n_ymid,o_xright,o_yright);
}

/*************************************************************/

/*  Routine draw_rule : draw outline of rule table                 */

/*************************************************************/

void draw_rule(void)
{
  extern char *state[7];
  int  cnt1, cnt2, x, y;
  char key_str[80];


  block = 6*tw;
  setcolor(YELLOW);
  gprintxy(midx-2*tw,25,"RULE");
  rectangle(midx-20,20,midx+19,35);
  setcolor(WHITE);
  gprintxy(midx-tw,50+6*th,"ER");
  gprintxy(midx-4*block,50+18*th,"CE");
  x = midx-3*block;
  for(cnt1=0;cnt1<6;cnt1++){
      y = 50+9*th;
      for(cnt2=0;cnt2<6;cnt2++){
          rectangle(x,y,x+block,y+3*th);
          y = y+3*th;
```

```
        }
      x = x+block;
     }
  x = midx-2*block+2*tw;
  y = 50+10*th;
  for(cnt1=0;cnt1<5;cnt1++){
      gprintxy(x,y,state[cnt1]);
      x = x+block;
     }
  y = 50+13*th;
  for(cnt1=0;cnt1<5;cnt1++){
      gprintxy(midx-3*block+tw,y,state[cnt1]);
      y = y+3*th;
      }
  setfillstyle(SOLID_FILL, CYAN);
  bar(0,MaxY-30,MaxX,MaxY);
  setcolor(BLUE);
  sprintf(key_str,"F1 ERROR   F2 CHANGE OF ERROR   F3 CONTROL INPUT

                   F4 RULE    Esc EXIT ");
  outtextxy(45,MaxY-20,key_str);
  setcolor(CYAN);
  outtextxy(midx-20,MaxY-50,crul_file);
}

/*****************************************************************/

/*  Routine default_rule : show default control rule             */

/*****************************************************************/

void default_rule(void *tr_table)
{
  int cnt1, cnt2, x, y;
  int (*r_table)[5][5];

  r_table = tr_table;
  setcolor(YELLOW);
  y = 50+13*th;
  for(cnt1=0;cnt1<5;cnt1++){
      x = midx-2*block+2*tw;
      for(cnt2=0;cnt2<5;cnt2++){
          ruleprint(x,y,(*r_table)[cnt1][cnt2]);
          x = x+block;
      }
      y = y+3*th;
  }
  startx = midx-2*block+2*tw-2;
  starty = 50+13*th-1;
  putimage(startx,starty,cursor,XOR_PUT);
  curx = 0;
  cury = 0;
  endx = 4;
  endy = 4;
}
/*****************************************************************/
/*  Routine ruleprint : print rule to screen                     */
/*****************************************************************/
void ruleprint(int xloc,int yloc, int rt)
{
  switch(rt){
    case LN : gprintxy(xloc,yloc,"LN");
              break;
    case SN : gprintxy(xloc,yloc,"SN");
              break;
    case ZE : gprintxy(xloc,yloc,"ZE");
              break;
    case SP : gprintxy(xloc,yloc,"SP");
              break;
    case LP : gprintxy(xloc,yloc,"LP");
              break;
    case BL : gprintxy(xloc,yloc,"  ");
  }
}

/*****************************************************************/

/*  Routine edit_rule : edit rule                                */

/*****************************************************************/

void edit_rule(void *tr_table)
{
  int cnt;
  int (*r_table)[5][5];
  char key;

  r_table = tr_table;
  cnt = 0;
  key = ' ';
  while(key!='\r'){
       if(cnt==NUM_STATE+1){
          cnt = 0;
        }
       if(key==' '){
          gprintxy(startx+2+curx*block,starty+1+cury*3*th,state[cnt]);

          cnt++;
        }
       else
         putchar(0x07);
       key = getch();
  }
  switch(cnt){
         case 1 : (*r_table)[cury][curx] = LN;
                  break;
         case 2 : (*r_table)[cury][curx] = SN;
                  break;
         case 3 : (*r_table)[cury][curx] = ZE;
                  break;
         case 4 : (*r_table)[cury][curx] = SP;
                  break;
         case 5 : (*r_table)[cury][curx] = LP;
                  break;
         case 6 : (*r_table)[cury][curx] = BL;
```



```
        }
   putimage(startx+curx*block, starty+cury*3*tw, cursor, XOR_PUT);
}

/*****************************************************************/

/* Routine save_mem_file : save most recent edit data in membership */

/*                         function table into file              */

/*****************************************************************/

void save_mem_file(void *t_er, void *t_ce, void *t_ci, char cof)
{

   FILE  *in;
   struct ffblk ffblk;
   float (*er)[MAX][MAX], (*ce)[MAX][MAX], (*ci)[MAX][MAX];
   char  fname[10] ;
   char  *type = ".mem";
   char  s;
   int   cnt1, cnt2;

   er = t_er;
   ce = t_ce;
   ci = t_ci;

   setfillstyle(SOLID_FILL, GREEN);
   bar(150,200,470,270);
   setcolor(WHITE);
   outtextxy(170,210," MEMBERSHIP FILE HAS BEEN MODIFIED");

   outtextxy(200,230," SAVE CHANGE ?<y/n> : ");
   s = getche();
   if(s == '\r'|| s == 'y'){
      setcolor(YELLOW);
      outtextxy(380,230,"Y");
      setcolor(WHITE);
      outtextxy(230,245,"SAVE AS:");
      setcolor(YELLOW);
      outtextxy(350,245,".mem");
      for(cnt1=0;cnt1<10;cnt1++)
         fname[cnt1] = '';
      cnt1 = 0;
      while(s != '\r'){
          s  = getch();
          if(s != '\r'){
             fname[cnt1] = s;
             outtextxy(300,245,fname);
          }
          cnt1++;
          }
      if(cnt1!=1){
         if(cof=='c')
            sprintf(cmem_file,"%s",fname);
         else
            sprintf(fmem_file,"%s",fname);
         strcat(fname,type);
         }
      if(!findfirst(fname,&ffblk,0))
         in = fopen(fname,"w+t");
      else
         in = fopen(fname,"wt");
      fprintf(in,"%d",qer_max);
      putc(' ',in);
      if(cof=='c')
         cqer_max = qer_max;
      else
         fqer_max = qer_max;
      fprintf(in,"%d",qce_max);
      putc(' ',in);
      if(cof=='c')
         cqce_max = qce_max;
      else
         fqce_max = qce_max;
      fprintf(in,"%d",qci_max);
      putc(' ',in);
      if(cof=='c')
         cqci_max = qci_max;
      else
         fqci_max = qci_max;
      for(cnt1=0;cnt1<MAX;cnt1++)
      for(cnt2=0;cnt2<MAX;cnt2++){
          fprintf(in,"%3.1f",(*er)[cnt1][cnt2]);
          putc(' ',in);
          if(cof=='c')
             c_er[cnt1][cnt2] = (*er)[cnt1][cnt2];
          else
             f_er[cnt1][cnt2] = (*er)[cnt1][cnt2];
        }
      for(cnt1=0;cnt1<MAX;cnt1++)
      for(cnt2=0;cnt2<MAX;cnt2++){
          fprintf(in,"%3.1f",(*ce)[cnt1][cnt2]);
          putc(' ',in);
          if(cof=='c')
             c_ce[cnt1][cnt2] = (*ce)[cnt1][cnt2];
          else
             f_ce[cnt1][cnt2] = (*ce)[cnt1][cnt2];
        }
      for(cnt1=0;cnt1<MAX;cnt1++)
      for(cnt2=0;cnt2<MAX;cnt2++){
          fprintf(in,"%3.1f",(*ci)[cnt1][cnt2]);
          putc(' ',in);
          if(cof=='c')
              c_ci[cnt1][cnt2] = (*ci)[cnt1][cnt2];
          else
              f_ci[cnt1][cnt2] = (*ci)[cnt1][cnt2];
        }
      fclose(in);
    }
}

/*****************************************************************/
/*  Routine save_rule_file : save most recent edit rule into file */
```

```
/*****************************************************************/

void save_rule_file(void *tr_table)
{
   FILE *in;
   struct ffblk ffblk;
   int  (*r_table)[5][5];
   char  fname[10] = "";
   char  *type = ".rul";
   char  s;
   int   cnt1, cnt2,count=0;

   r_table = tr_table;
   for(cnt1=0;cnt1<5;cnt1++)
   for(cnt2=0;cnt2<5;cnt2++)
       if((*r_table)[cnt1][cnt2]!=BL)
          count++;
   tr_max=count;
   setfillstyle(SOLID_FILL, GREEN);
   bar(150,200,470,270);
   setcolor(WHITE);
   outtextxy(170,210,"   RULE FILE HAS BEEN MODIFIED");

   outtextxy(200,230," SAVE CHANGE ?<y/n> : ");
   s = getche();
   if(s == '\r'|| s == 'y'){
      setcolor(YELLOW);
      outtextxy(380,230,"Y");
      setcolor(WHITE);
      outtextxy(230,245,"SAVE AS:");
      setcolor(YELLOW);
      outtextxy(350,245,".rule");
      cnt1 = 0;
      while(s != '\r'){
            s  = getch();
            if(s != '\r'){
               fname[cnt1] = s;
               outtextxy(300,245,fname);
              }
            cnt1++;
            }
      if(cnt1!=1){
         sprintf(crul_file,"%s",fname);
         sprintf(frul_file,"%s",fname);
         strcat(fname,type);
         }
      if(!findfirst(fname,&ffblk,0))
         in = fopen(fname,"r+t");
      else
         in = fopen(fname,"wt");
      fprintf(in,"%d",tr_max);
      putc(' ',in);
      r_max = tr_max;
      for(cnt1=0;cnt1<5;cnt1++)
      for(cnt2=0;cnt2<5;cnt2++){
          fprintf(in,"%d",(*r_table)[cnt1][cnt2]);
          putc(' ',in);
          rule[cnt1][cnt2] = (*r_table)[cnt1][cnt2];
         }
      fclose(in);
      }
}

/*****************************************************************/

/*  Routine load_rule : Load rule from user indicated rule file to  */

/*                      array of integer that contain control rule  */

/*****************************************************************/

void load_rule(char *fname)
{
   extern int rule[5][5];
   FILE *out;
   struct ffblk ffblk;
   int  cnt1,cnt2;

   if(!findfirst(fname,&ffblk,0)){
      load = 1;
      out = fopen(fname,"r+t");
      if(out == NULL)
         out = fopen(fname, "rt");
      fscanf(out,"%d",&r_max);
      for(cnt1=0;cnt1<5;cnt1++)
      for(cnt2=0;cnt2<5;cnt2++)
          fscanf(out,"%d",&rule[cnt1][cnt2]);
      fclose(out);
     }
   else{
      load = 0;
      window(25,22,60,24);
      textbackground(GREEN);
      cprintf("%s, file does'nt exist",fname);
      getch();
      textbackground(BLUE);
      clrscr();
      window(1,1,80,25);
     }
}

/*****************************************************************/

/*  Routine load_mem : Load data from user indicated file to array  */

/*                     of integer that contain data in membership   */

/*                     function table                               */

/*****************************************************************/

void load_mem(char *fname, char cof)
{
   FILE  *out;
```



```
   struct ffblk ffblk;
   char   *buffer = "";
   int    cnt1, cnt2;

   if(!findfirst(fname,&ffblk,0)){
      load = 1;
      out = fopen(fname,"r+t");
      if(out == NULL)
         out = fopen(fname,"rt");
      switch(cof){
          case 'c': fscanf(out,"%d",&cqer_max);
                    fscanf(out,"%d",&cqce_max);
                    fscanf(out,"%d",&cqci_max);
                    for(cnt1=0;cnt1<MAX;cnt1++)
                    for(cnt2=0;cnt2<MAX;cnt2++){
                        fscanf(out,"%s",buffer);
                        c_er[cnt1][cnt2] = atof(buffer);
                        }
                    for(cnt1=0;cnt1<MAX;cnt1++)
                    for(cnt2=0;cnt2<MAX;cnt2++){
                        fscanf(out,"%s",buffer);
                        c_ce[cnt1][cnt2] = atof(buffer);
                        }
                    for(cnt1=0;cnt1<MAX;cnt1++)
                    for(cnt2=0;cnt2<MAX;cnt2++){
                        fscanf(out,"%s",buffer);
                        c_ci[cnt1][cnt2] = atof(buffer);
                        }
                    break;
          case 'f': fscanf(out,"%d",&fqer_max);
                    fscanf(out,"%d",&fqce_max);
                    fscanf(out,"%d",&fqci_max);
                    for(cnt1=0;cnt1<MAX;cnt1++)
                    for(cnt2=0;cnt2<MAX;cnt2++){
                        fscanf(out,"%s",buffer);
                        f_er[cnt1][cnt2] = atof(buffer);
                        }
                    for(cnt1=0;cnt1<MAX;cnt1++)
                    for(cnt2=0;cnt2<MAX;cnt2++){
                        fscanf(out,"%s",buffer);
                        f_ce[cnt1][cnt2] = atof(buffer);
                        }
                    for(cnt1=0;cnt1<MAX;cnt1++)
                    for(cnt2=0;cnt2<MAX;cnt2++){
                        fscanf(out,"%s",buffer);
                        f_ci[cnt1][cnt2] = atof(buffer);
                        }
      }
     fclose(out);
    }
   else{
      load = 0;
      window(25,22,60,24);
      textcolor(WHITE);
      textbackground(GREEN);
      cprintf("   %s, file does'nt exist",fname);
      getch();
      textbackground(BLUE);
      clrscr();
      window(1,1,80,25);
     }
}

/*****************************************************************/

/*  Routine view_fuzzy : Show Background of user interface screen   */

/*                       of fuzzy controller show-edit              */

/*****************************************************************/

void view_fuzzy(void)
{

  textbackground(BLUE);
  clrscr();
  textcolor(WHITE);
  draw_block(2,2,40,20);
  draw_block(40,2,79,20);
  draw_block(30,1,48,3);
  textcolor(YELLOW);
  gotoxy(31,2);
  cprintf(" FUZZY CONTROLLER");
  gotoxy(10,5);
  cprintf("COARSE TABLE");
  gotoxy(50,5);
  cprintf("FINE TABLE");
  gotoxy(5,7);
  cprintf("MEMBERSHIP FILE:");
  xpos[0] = wherex();
  ypos[0] = wherey();
  gotoxy(5,9);
  cprintf("RULE FILE:");
  xpos[1] = wherex();
  ypos[1] = wherey();
  gotoxy(5,11);
  cprintf("MAXIMUM ERROR: ");
  xpos[2] = wherex();
  ypos[2] = wherey();
  gotoxy(5,12);
  cprintf("ERROR QUANTIZE: ");
  xpos[3] = wherex();
  ypos[3] = wherey();
  gotoxy(5,13);
  cprintf("MAXIMUM CHANGE OF ERROR: ");
  xpos[4] = wherex();
  ypos[4] = wherey();
  gotoxy(5,14);
  cprintf("CE QUANTIZE: ");
  xpos[5] = wherex();
  ypos[5] = wherey();
  gotoxy(5,15);
  cprintf("MAXIMUM CONTROL INPUT: ");
  xpos[6] = wherex();
```

```
  ypos[6] = wherey();
  gotoxy(5,16);
  cprintf("CI QUANTIZE: ");
  xpos[7] = wherex();
  ypos[7] = wherey();

  gotoxy(45,7);
  cprintf("MEMBERSHIP FILE:");
  xpos[8] = wherex();
  ypos[8] = wherey();
  gotoxy(45,9);
  cprintf("RULE FILE:");
  xpos[9] = wherex();
  ypos[9] = wherey();
  gotoxy(45,11);
  cprintf("MAXIMUM ERROR: ");
  xpos[10] = wherex();
  ypos[10] = wherey();
  gotoxy(45,12);
  cprintf("ERROR QUANTIZE: ");
  xpos[11] = wherex();
  ypos[11] = wherey();
  gotoxy(45,13);
  cprintf("MAXIMUM CHANGE OF ERROR: ");
  xpos[12] = wherex();
  ypos[12] = wherey();
  gotoxy(45,14);
  cprintf("CE QUANTIZE: ");
  xpos[13] = wherex();
  ypos[13] = wherey();
  gotoxy(45,15);
  cprintf("MAXIMUM CONTROL INPUT: ");
  xpos[14] = wherex();
  ypos[14] = wherey();
  gotoxy(45,16);
  cprintf("CI QUANTIZE: ");
  xpos[15] = wherex();
  ypos[15] = wherey();

  textcolor(CYAN);
  gotoxy(xpos[0],ypos[0]);
  cprintf("%s .mem",cmem_file);
  gotoxy(xpos[1],ypos[1]);
  cprintf("%s .rule",crul_file);
  gotoxy(xpos[2],ypos[2]);
  cprintf("%3.2f",cmax_er);
  gotoxy(xpos[3],ypos[3]);
  cprintf("%d",cqer_max);
  gotoxy(xpos[4],ypos[4]);
  cprintf("%3.2f",cmax_ce);
  gotoxy(xpos[5],ypos[5]);
  cprintf("%d",cqce_max);
  gotoxy(xpos[6],ypos[6]);
  cprintf("%3.2f",cmax_ci);
  gotoxy(xpos[7],ypos[7]);
  cprintf("%d",cqci_max);

  gotoxy(xpos[8],ypos[8]);
  cprintf("%s .mem",fmem_file);
  gotoxy(xpos[9],ypos[9]);
  cprintf("%s .rule",frul_file);
  gotoxy(xpos[10],ypos[10]);
  cprintf("%3.2f",fmax_er);
  gotoxy(xpos[11],ypos[11]);
  cprintf("%d",fqer_max);
  gotoxy(xpos[12],ypos[12]);
  cprintf("%3.2f",fmax_ce);
  gotoxy(xpos[13],ypos[13]);
  cprintf("%d",fqce_max);
  gotoxy(xpos[14],ypos[14]);
  cprintf("%3.2f",fmax_ci);
  gotoxy(xpos[15],ypos[15]);
  cprintf("%d",fqci_max);
}

/*****************************************************************/

/*  Routine get_float : get float from user in graphics mode     */

/*****************************************************************/

float get_float(void)
{ int  cnt;
  char fl_temp[5];
  char temp;

  for(cnt=0;cnt<5;cnt++)
      fl_temp[cnt] = 0;
  cnt =0;
  temp = ' ';
  while(temp != '\r'){
       temp = getche();
       if(temp != '\r')
          fl_temp[cnt] = temp;
       cnt++;
       }
  if(cnt!=1)
     return(atof(fl_temp));
  else
     return(0);
}

/*****************************************************************/

/*  Routine get_int : get integer from user in graphics mode     */

/*****************************************************************/

int get_int(void)
{
  char i_temp[2];

  i_temp[1] = 0;
  i_temp[0] = getche();
  if(i_temp[0]!='\r'){
```



```
     m_edit = 1;
     return(atoi(i_temp));
     }
  else
     return(0);
}
/*****************************************************************/

/*  Routine Title : Show title of fuzzy controller on screen     */

/*****************************************************************/

void Title(void)
{
    gotoxy(29,10);
    cprintf("FUZZY CONTROLLER");
    gotoxy(25,23);
    cprintf("Press any key to continue");
}
```

graph.h

```
/*****************************************************************/
/*                                                               */
/*                Header file for Graphics routine               */
/*                                                               */
/*****************************************************************/


/*****************************************************************/
/*                      Initial Graphic device                   */
/*****************************************************************/

void Initialize(void)
{
int Graphmode;
int Graphdriver;
int Errorcode = 0;
  Graphdriver = DETECT;
  initgraph( &Graphdriver, &Graphmode, "");
  Errorcode = graphresult();
  if ( Errorcode != grOk )
  {  printf("Graphics System Error: %s\n",
            grapherrormsg(Errorcode));
     exit(1);
  }
}


/*****************************************************************/
/*                   Create Background of Graph                  */
/*****************************************************************/

void graph1(float setpoint,float Ymax,float Time_range,int con_type)
{
extern int MaxX, MaxY, gapX, gapY;

int cnt1;
double time_number,output_number;
char time_char[3],output_char[3];
    setbkcolor(WHITE);
    setcolor(DARKGRAY);
    rectangle(0,0,MaxX,MaxY);
    rectangle(4,4,MaxX-4,MaxY-4);
    setcolor(RED);
    line(MaxX-40,MaxY-60,60,MaxY-60);
    line(60,MaxY-60,60,40);
    setcolor(DARKGRAY);
    setlinestyle(DOTTED_LINE,0,1);
         for(cnt1=1;cnt1<=10;cnt1++){
             line(60+cnt1*gapX,40,60+cnt1*gapX,MaxY-60);
             line(60,MaxY-60-cnt1*gapY,MaxX-40,MaxY-60-cnt1*gapY);
             }
    setcolor(CYAN);
    line(60,(MaxY-60-(gapY*10/Ymax*setpoint)),MaxX-40,
               (MaxY-60-(gapY*10/Ymax*setpoint)));
    setcolor(BLUE);
    settextstyle(TRIPLEX_FONT,HORIZ_DIR,1);
    switch(con_type){
       case 1: outtextxy(MaxX/5,15,"System Response with
                                    FUZZY Controller");
               break;
       case 2: outtextxy(MaxX/5,15,"System Response with
                                    PID Controller");
               break;
       case 3: outtextxy(MaxX/5-10,15,"Compare System Response
                              between FUZZY and PID Controller");
       }
    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
    outtextxy(MaxX/2,MaxY-25,"Time (sec.)");
    settextstyle(DEFAULT_FONT,VERT_DIR,1);
    outtextxy(25,MaxY/2-25,"Output");
    settextstyle(DEFAULT_FONT,HORIZ_DIR,0);
    for(cnt1=2;cnt1<=10;cnt1=cnt1+2){
    output_number=cnt1*(Ymax/10);
        gcvt(output_number,3,output_char);
        outtextxy(25,MaxY-60-cnt1*gapY,output_char);
        time_number=cnt1*(Time_range/10);
        gcvt(time_number,3,time_char);
        outtextxy(60+cnt1*gapX,MaxY-45,time_char);
        }
    moveto(60,MaxY-60);
}


/*****************************************************************/
/*                      Plot Response of system                  */
/*****************************************************************/

void plotgraph(float value_1,float value,int time,
               float Ymax,int color)
{   extern int MaxY, gapY ;

    setcolor(color);
    setlinestyle(SOLID_LINE,0,1);
    line((time-1)+60,(MaxY-60-(gapY*10/Ymax*value_1)),
               time+60,(MaxY-60-(gapY*10/Ymax*value)));
}
```

```
/*****************************************************************/
/*                                                               */
/*              Header file for finding Solution of              */
/*      Differential equation by Runge-Kutta order 4th method    */
/*                                                               */
/*****************************************************************/

extern float x[MAX];

/*****************************************************************/
/*        Function to prepare Alpha for Function Runge           */
/*****************************************************************/

void Prepare_Alpha(float *source1,float *source2,float *dest,
                   int order)
{ float *s1,*s2,*d1,*d2; /* source1 is a,source2 is b,dest is alpha */

  float alpha_temp;
  int   ct1,ct2;
     s1=source1+order;
     d1=dest;
     (*d1)=(*s1);
     for(ct1=1;ct1<=order;ct1++){
         s1=source1+(order-ct1);
         d1=dest+ct1;
         alpha_temp=0;
         for(ct2=1;ct2<=ct1;ct2++){
             s2=source2+(order-ct2);
             d2=dest+(ct1-ct2);
             alpha_temp+=(*s2)*(*d2);
             }
         (*d1)=(*s1)-alpha_temp;
         }
}


/*****************************************************************/
/*       Function to prepare state equation for Function Runge   */
/*****************************************************************/

float f(float *source1,float *source2,float *x,int order,
        int equation_index,float input)
{ float sum;  /* source1 is alpha , source2 is b */
  int ct1;
    if(equation_index!=order)
       return(x[equation_index]+(source1[equation_index])*input);
    else{ sum=0;
          for(ct1=0;ct1<order;ct1++)
              sum-=(source2[ct1])*(x[ct1]);
          return(sum+(source1[order])*input);
        }
}


/*****************************************************************/
/*                Runge-Kutta funtion to find x                  */
/*****************************************************************/

float Runge(float *source1,float *source2,float *dest,int order,
            float input,float h)
{ float factor;                 /* source1 is alpha,source2 is b */
  float Kutta[5][MAX],xxx[500];  /* dest is x */
  int equation_index,ct3,ct2,ct1;
     for(ct1=1;ct1<=order;ct1++)  /* clear K[0][n] */
         Kutta[0][ct1]=0;
     for(ct3=1;ct3<=4;ct3++){
         if(ct3==4) factor=1;           /* set factor of Kutta */
           else factor=0.5;
         for(ct2=0;ct2<order;ct2++)   /* find medium value */
             xxx[ct2]=dest[ct2]+factor*Kutta[ct3-1][ct2+1];
         for(equation_index=1;equation_index<=order;equation_index++)
             Kutta[ct3][equation_index]=
                h*f(source1,source2,xxx,order,equation_index,input);
         }
     for(ct1=0;ct1<order;ct1++){
         dest[ct1]+=(Kutta[1][ct1+1]+2*Kutta[2][ct1+1]+
                    2*Kutta[3][ct1+1]+Kutta[4][ct1+1])/6;
         }
     return(dest[0]);
}


/*****************************************************************/
/*                Funtion to find Output Response                */
/*****************************************************************/

float Response(float *numer,float *denom,float input,
               float Time_range,int order)
{ float Alpha[MAX+1],temp,h;
  float *tempnumer2,*tempdenom1,*tempdenom2;
  int   ct,precision=10,j;
    tempdenom1=denom+order;
    tempnumer2=numer;
    tempdenom2=denom;
    if(((*tempdenom1)!=1)||((*tempdenom1)!=0))
        for(j=0;j<=order;j++){
            (*tempdenom2)/=(*tempdenom1);
            tempdenom2++;
```



```
            (*tempnumer2)/=(*tempdenom1);
            tempnumer2++;
            }
    h=Time_range/(550*precision);    /* Iteration scale */
    Prepare_Alpha(numer,denom,Alpha,order);
    for(ct=0;ct<=precision;ct++)
            temp=Runge(Alpha,denom,x,order,input,h);
    return(temp);
}
```

```
#define vinmax  10
#define voutmax 5

void run(void)
{
union REGS inregs, outregs;
struct SREGS segs;

void interrupt handler(void);

unsigned old_seg, old_offset;    /*original handler Seg:0fs*/

extern float num[MAX],denom[MAX];
extern float fmax_er,fmax_ce,fmax_ci,cmax_er,cmax_ce,cmax_ci;
extern float f_er[MAX][MAX], f_ce[MAX][MAX], f_ci[MAX][MAX];
extern float c_er[MAX][MAX], c_ce[MAX][MAX], c_ci[MAX][MAX];
extern float f_table[MAX][MAX], c_table[MAX][MAX];
extern int   MaxX,MaxY,gapX,gapY;
extern int   fqer_max, fqce_max, fqci_max;
extern int   cqer_max, cqce_max, cqci_max;
extern int   order;
extern int   count,number,check;
int    t,cnt;
int    vin, vout;
int    buffer[5];
float  setpoint;
float errork_1,errork,output;
double errtablef,cerrtablef,errtable,cerrtable;
float  out,ce,ci;
float  stime;
unsigned char lowbyte;
unsigned char hibyte;


/*****************************************************************/

/*              Initialize Fuzzy table and some parameters       */

/*****************************************************************/

     Fuzzy_table(f_er,f_ce,f_ci,fqer_max,fqce_max,fqci_max,f_table);
     Fuzzy_table(c_er,c_ce,c_ci,cqer_max,cqce_max,cqci_max,c_table);
     errork_1=0;
     output = 0;
     setpoint = 1;
     ci = setpoint;
     count = 0;

     window(20,20,60,23);
     textcolor(BLUE);
     textbackground(CYAN);
     draw_block(1,1,40,3);
     gotoxy(2,2);
     cprintf("Enter sampling time(sec) : ");
     cscanf("%f",&stime);
     window(1,1,80,25);

     clrscr();
     draw_block(30,11,40,15);
     draw_block(40,11,50,15);
     gotoxy(33,12);
     cprintf("vin");
     gotoxy(33,14);
     cprintf("vout");
     number = (int)(stime*18.2);

  /* Save the original int 1CH vector */
     inregs.x.ax = 0x351C;
     intdosx(&inregs, &outregs, &segs);
     old_seg = segs.es;
     old_offset = outregs.x.bx;

  /* Assign the interrupt to our handler */
     inregs.x.ax = 0x251C;
     inregs.x.dx = (unsigned) FP_OFF(handler);
     segs.ds = (unsigned) FP_SEG(handler);

  /* Disable and enable interrupt before
       and after changing its int handler*/
     disable();
     intdosx(&inregs, &outregs, &segs);
     enable();

 do{
    if(count==0 && check==0){
/*****************************************************************/

/*                  inport from A/D                              */

/*****************************************************************/

        outportb(0x22B,0x01);
        for(cnt=0;cnt<5;cnt++){
            outportb(0x22A,0x00);
            outportb(0x22C,0x00);
            do {
               hibyte = inportb(0x225);
               } while(hibyte > 0x0F);
               lowbyte = inport(0x224);
               buffer[cnt] = (int)hibyte*256+(int)lowbyte - 2048;
             }
        outportb(0x22B,0x00);

/*****************************************************************/

/*              convert DEC to input value in voltage            */

/*****************************************************************/

        vin = buffer[4];
        output = (float)vinmax*(float)vin/2047;

/*****************************************************************/
/*                  Fuzzy Controller                             */
```



```
/*****************************************************************/

        errork=output-setpoint;
        ce=errork-errork_1;
        if(fabs(errork) > fmax_er)
            errtablef=modf(errork*cqer_max/cmax_er,&errtable);
        else
            errtablef=modf(errork*fqer_max/fmax_er,&errtable);
        if(fabs(errtablef)>=0.5)
        if(errtable>0)
            errtable=errtable+1;
        else
            errtable=errtable-1;
        if(errtable>cqer_max)
        errtable=cqer_max;
        else if(errtable<(-cqer_max))
            errtable=(-cqer_max);
        if(fabs(errork) > fmax_er)
            cerrtablef=modf(ce*cqce_max/cmax_ce,&cerrtable);
        else
            cerrtablef=modf(ce*fqce_max/fmax_ce,&cerrtable);
        if(fabs(cerrtablef)>=0.5)
            if(cerrtable>0)
            cerrtable=cerrtable+1;
        else
            cerrtable=cerrtable-1;
        if(cerrtable>cqce_max)
            cerrtable=cqce_max;
        else if(cerrtable<(-cqce_max))
            cerrtable=(-cqce_max);
    if(fabs(errork)>fmax_er){
            out=c_table[errtable+cqer_max][cerrtable+cqce_max];
        ci = setpoint+out*(cmax_ci-setpoint)/cqci_max;
        }
    else{
        out=f_table[errtable+fqer_max][cerrtable+cqce_max];
        ci= setpoint+out*(fmax_ci-setpoint)/fqci_max;
        }
    errork=errork_1;
/*****************************************************************/

/*              convert output value to DEC                      */

/*****************************************************************/

    vout = 4095*ci/voutmax;
    if(ci==0)
        vout = 1;

/*****************************************************************/

/*                  outport to D/A                               */

/*****************************************************************/

    outport(0x226,vout);
    gotoxy(43,12);
    cprintf("%5.4f",output);
    gotoxy(43,14);
    cprintf("%5.4f",ci);
    }
    check++;
   }while(!kbhit());

  /* Restore interrupt vector assign interrupt to our handler */
    inregs.x.ax = 0x251C;
    inregs.x.dx = (unsigned) old_offset;
    segs.ds = (unsigned) old_seg;

  /* Disable and enable interrupt before
       and after changing its int handler*/
    disable();
    intdosx(&inregs, &outregs, &segs);
    enable();
}
void interrupt handler (void)
{
    if (count++ == number){
        count = 0;
        check = 0;
    }
}
```

```
/*****************************************************************/
/*                                                               */
/*                  Header file for RUN System                   */
/*                                                               */
/*****************************************************************/


/*****************************************************************/
/*       Create fuzzy table for fuzzy controller                 */
/*****************************************************************/

void Fuzzy_table(void *t_er, void *t_ce, void *t_ci, int qer_max
                 ,int qce_max, int qci_max ,void *t_table)
{
  extern int    r_max ;
  extern int    rule[5][5] ;

  float (*er)[MAX][MAX], (*ce)[MAX][MAX], (*ci)[MAX][MAX];
  float (*table)[MAX][MAX];
  int cnt1, cnt2, cnt3, cnt4, cnt5;
  float w, u1, u2, g;
  float mo[MAX], o[MAX];

      er = t_er;
      ce = t_ce;
      ci = t_ci;
      table = t_table;

  for(cnt1=0; cnt1<qer_max*2+1; cnt1++){
  for(cnt2=0; cnt2<qce_max*2+1; cnt2++){
        /*************************************************/
        /*         clear inference output array          */
        /*************************************************/
        for (cnt3=0; cnt3<qci_max*2+1; cnt3++){
            o[cnt3] = 0;
            }
        /*************************************************/
        /*                Inference                      */
        /*************************************************/
        for(cnt3=0; cnt3<NUM_STATE-1; cnt3++){
    for(cnt4=0; cnt4<NUM_STATE-1; cnt4++){
         if(rule[cnt4][cnt3] != BL){
             if((*er)[cnt3][cnt1] < (*ce)[cnt4][cnt2])
                 w = (*er)[cnt3][cnt1];
             else
                 w = (*ce)[cnt4][cnt2];
         for(cnt5=0; cnt5<qci_max*2+1; cnt5++){
             if(w > (*ci)[rule[cnt4][cnt3]][cnt5])
                mo[cnt5] = (*ci)[rule[cnt4][cnt3]][cnt5];
             else
                mo[cnt5] = w;
          }

             for(cnt5=0; cnt5<qci_max*2+1; cnt5++){
                 if (o[cnt5] < mo[cnt5])
                     o[cnt5] = mo[cnt5];
                 }
         }
    }
    }
        /*************************************************/
        /*     Defuzzification by calculate center of gravity      */
        /*************************************************/
        u1 = 0;
        u2 = 0;
        for(cnt3=0; cnt3<qci_max*2+1; cnt3++){
            u1 = u1+o[cnt3];
            u2 = u2+o[cnt3]*cnt3;
            }
        if(u1==0)
           g = 0;
        else
           g = u2/u1-qci_max;
        /*************************************************/
        /*                create fuzzy table             */
        /*************************************************/
        (*table)[cnt1][cnt2] = g;
        }
        }
}


/*****************************************************************/
/*Caculate and update control input of process with FUZZY Controller*/
/*****************************************************************/


void fuzzy(int tmax,int input,float Ymax)
{
extern float num[MAX],denom[MAX];
extern float fmax_er,fmax_ce,fmax_ci,cmax_er,cmax_ce,cmax_ci;
extern float f_er[MAX][MAX], f_ce[MAX][MAX], f_ci[MAX][MAX];
extern float c_er[MAX][MAX], c_ce[MAX][MAX], c_ci[MAX][MAX];
extern float f_table[MAX][MAX], c_table[MAX][MAX];
extern float ci_dat[SCALEMAX];
```



```
extern int   MaxX,MaxY,gapX,gapY;
extern int   fqer_max, fqce_max, fqci_max;
extern int   cqer_max, cqce_max, cqci_max;
extern int   order;
int    t,cnt;
float  setpoint,errork_1,errork,output,output_1;
double erftablef,cerftablef,erftable,cerftable;
float out,ce,ci;
    Fuzzy_table(f_er,f_ce,f_ci,fqer_max,fqce_max,fqci_max,f_table);
    Fuzzy_table(c_er,c_ce,c_ci,cqer_max,cqce_max,cqci_max,c_table);
    for(cnt=0;cnt<MAX;cnt++)
        x[cnt] = 0;
    setpoint=input;
    errork_1=0;
    output_1 = 0;
    ci_dat[0] = 0;

    for(t=0;t<=SCALEMAX;t++){
        if(t==0)
          output=Response(&num[0],&denom[0],input,tmax,order);
        else{
             errork=output-setpoint;
             ce=errork-errork_1;
             if(fabs(errork) > fmax_er)
               erftablef=modf(errork*cqer_max/cmax_er,&erftable);
             else
               erftablef=modf(errork*fqer_max/fmax_er,&erftable);
             if(fabs(erftablef)>=0.5)
                if(erftable>0)
                    erftable=erftable+1;
                else
                    erftable=erftable-1;
             if(erftable>cqer_max)
                erftable=cqer_max;
             else if(erftable<(-cqer_max))
                    erftable=(-cqer_max);
             if(fabs(errork) > fmax_er)
              cerftablef=modf(ce*cqce_max/cmax_ce,&cerftable);
             else
              cerftablef=modf(ce*fqce_max/fmax_ce,&cerftable);
             if(fabs(cerftablef)>=0.5)
               if(cerftable>0)
                  cerftable=cerftable+1;
               else
                  cerftable=cerftable-1;
             if(cerftable>cqce_max)
               cerftable=cqce_max;
             else if(cerftable<(-cqce_max))
                    cerftable=(-cqce_max);
             if(fabs(errork)>fmax_er){
              out= c_table[erftable+cqer_max][cerftable+cqce_max];
              ci = setpoint+out*(cmax_ci-setpoint)/cqci_max;
              }
                else{
             out= f_table[erftable+fqer_max][cerftable+fqce_max];
             ci = setpoint+out*(fmax_ci-setpoint)/fqci_max;
             }
           output=Response(&num[0],&denom[0],ci,tmax,order);
           errork=errork_1;
        }
       plotgraph(output_1,output,t,Ymax,GREEN);
       output_1 = output;
       ci_dat[t] = ci;
      }
}

/*****************************************************************/
/*                Subroutine for PID Controller                  */
/*****************************************************************/

void PID_controller(void)
{
char   chk = 'y';
extern float Kp,Ti,Td;

    textcolor(WHITE);
    textbackground(BLUE);
    clrscr();
    draw_block(18,6,58,15);
    gotoxy(19,7);
    cprintf("********** PID Controller **********");
    gotoxy(19,8);
    cprintf("******* Gc(s)=Kp[1+1/(sTi)+sTd] *******");
    gotoxy(19,10);
    cprintf("          Input value of Kp :");
    cscanf("%f",&Kp);
    if(Kp==0){
       getch();
       window(20,20,60,23);
       textcolor(BLUE);
       textbackground(CYAN);
       draw_block(1,1,40,3);
       gotoxy(2,2);
       cprintf("Are you sure that Kp = 0 ?<y/n> : ");
       chk = getch();
       textcolor(WHITE);
       textbackground(BLUE);
       clrscr();
       window(1,1,80,25);
       if(chk!='y' && chk!='Y'){
          do{
           gotoxy(19,10);
           cprintf("          Input value of Kp :");
           cscanf("%f",&Kp);
          }while(Kp==0);
         }
    }
    gotoxy(19,12);
    cprintf("          Input value of Ti :");
    cscanf("%f",&Ti);
    gotoxy(19,14);
    cprintf("          Input value of Td :");
```

```
    cscanf("%f",&Td);
}

void PID_TF(int order)
{
int cnt1;
extern float Kp,Ti,Td;
extern float num[MAX],denom[MAX],newnum[MAX+2],newdenom[MAX+2];

    for(cnt1=0;cnt1<=order+2;cnt1++){
        newnum[cnt1]=0;
        newdenom[cnt1]=0;
        }
    if ((Kp!=0)&&(Ti!=0)&&(Td!=0)) {
        for(cnt1=0;cnt1<=order+1;cnt1++){
            if(cnt1==order+1)
                newnum[cnt1]=Kp*Ti*Td*num[cnt1-2];
            else if(cnt1==order)
                newnum[cnt1]=Kp*Ti*Td*num[cnt1-2]+Kp*Ti*num[cnt1-1];
            else if(cnt1==1)
                newnum[cnt1]=Kp*Ti*num[0]+Kp*num[1];
            else if(cnt1==0)
                newnum[cnt1]=Kp*num[0];
            else
                newnum[cnt1]=Kp*Ti*Td*num[cnt1-2]+Kp*Ti*num[cnt1-1]
                             +Kp*num[cnt1];
            }
        for(cnt1=0;cnt1<=order+1;cnt1++){
            if(cnt1==0)
                newdenom[cnt1]=newnum[cnt1];
            else
                newdenom[cnt1]=newnum[cnt1]+Ti*denom[cnt1-1];
            }
        }
    else if ((Kp!=0)&&(Ti==0)&&(Td!=0)) {
             for(cnt1=0;cnt1<=order;cnt1++){
                 if(cnt1==order)
                     newnum[cnt1]=Kp*Td*num[cnt1-1];
                 else if(cnt1==0)
                     newnum[cnt1]=Kp*num[0];
                 else
                     newnum[cnt1]=Kp*Td*num[cnt1-1]+Kp*num[cnt1];
                 }
             for(cnt1=0;cnt1<=order;cnt1++)
             newdenom[cnt1]=newnum[cnt1]+denom[cnt1];
             }
    else if ((Kp!=0)&&(Ti!=0)&&(Td==0)) {
              for(cnt1=0;cnt1<=order;cnt1++){
                  if(cnt1==order)
                      newnum[cnt1]=Kp*Ti*num[cnt1-1];
                  else if(cnt1==0)
                      newnum[cnt1]=Kp*num[0];
                  else
                      newnum[cnt1]=Kp*Ti*num[cnt1-1]+Kp*num[cnt1];
                  }
              for(cnt1=0;cnt1<=order+1;cnt1++){
                  if(cnt1==order+1)
                      newdenom[cnt1]=denom[cnt1-1];
                  else if(cnt1==0)
                      newdenom[cnt1]=newnum[cnt1];
                  else
                      newdenom[cnt1]=newnum[cnt1]+Ti*denom[cnt1-1];
                  }
              }
    else if ((Kp!=0)&&(Ti==0)&&(Td==0))  {
              for(cnt1=0;cnt1<=order-1;cnt1++)
                  newnum[cnt1]=Kp*num[cnt1];
              for(cnt1=0;cnt1<=order;cnt1++){
                  if(cnt1==order)
                      newdenom[cnt1]=denom[cnt1];
                  else
                 newdenom[cnt1]=newnum[cnt1]+denom[cnt1];
                  }
              }
}

void pid(float tmax,float setpoint,float Ymax)
{
float  output, output_1;
int t,cnt,order_temp;
extern float newnum[MAX+2],newdenom[MAX+2];
extern float Kp;
extern int order;

   for(cnt=0;cnt<MAX;cnt++)
     x[cnt] = 0;
   output_1 = 0;
   moveto(60,MaxY-60);
   if(Ti==0)
      order_temp = order;
   else
      order_temp = order+1;
   for(t=0;t<550;t++){
     if(Kp != 0)
        output=Response(&newnum[0],&newdenom[0],setpoint,
                        tmax,order_temp);
     else
        output=0;
     plotgraph(output_1,output,t,Ymax,RED);
     output_1 = output;
     }
}

void run_close(int tmax,float setpoint,float Ymax)
{
extern float num[MAX],denom[MAX];
extern int   order;
float  cl_denom[MAX];
float  output, output_1;
int t,cnt;

 for(cnt=0;cnt<MAX;cnt++)
     cl_denom[cnt] = num[cnt] +denom[cnt];
 for(cnt=0;cnt<MAX;cnt++)
     x[cnt] = 0;
```



```
 output_1 = 0;
 moveto(60,MaxY-60);
 for(t=0;t<550;t++){
     output=Response(&num[0],&cl_denom[0],setpoint,tmax,order);
     plotgraph(output_1,output,t,Ymax,BLUE);
     output_1 = output;
     }
}

void run_open(int tmax,float setpoint,float Ymax)
{
extern float num[MAX],denom[MAX];
extern int   order;
float  output, output_1;
int t,cnt;

 for(cnt=0;cnt<MAX;cnt++)
     x[cnt] = 0;
 output_1 = 0;
 moveto(60,MaxY-60);
 for(t=0;t<550;t++){
     output=Response(&num[0],&denom[0],setpoint,tmax,order);
     plotgraph(output_1,output,t,Ymax,BROWN);
     output_1 = output;
     }
}

/*****************************************************************/

/*                  Subroutine for Run simulate System           */

/*****************************************************************/


void sim(int con_type)
{
extern int input;
extern float cmax_er,cmax_ce,cmax_ci;
extern float ci_dat[SCALEMAX];
extern int   MaxX,MaxY,gapX,gapY,stop;
int Time_range,t;
float setpoint,Ymax;
union inkey{
            char ch[2];
            int i;
            } c;

      stop = 0;
      input = 1;
      setpoint = input;
      window(20,20,60,22);
      textcolor(BLUE);
      textbackground(CYAN);
      draw_block(1,1,40,3);
      gotoxy(2,2);
      cprintf("Input the maximum time to simulate:");
      cscanf("%d",&Time_range);
      clrscr();
      draw_block(1,1,40,3);
      gotoxy(2,2);
      cprintf("Input the maximum Y-Axis:");
      cscanf("%f",&Ymax);
      window(1,1,80,25);
      Initialize();
      MaxX=getmaxx();
      MaxY=getmaxy();
      gapX=((MaxX-40)-60)/10;
      gapY=((MaxY-60)-40)/10;
      switch(con_type){
            /* FUZZY CONTROLLER */
            case 1: graph1(setpoint,Ymax,Time_range,1);
                    fuzzy(Time_range,setpoint,Ymax);
                    break;
            /* PID CONTROLLER */
            case 2: graph1(setpoint,Ymax,Time_range,2);
                    PID_TF(order);
                    pid(Time_range,setpoint,Ymax);
                    break;
            /* Compare FUZZY & PID CONTROLLER */
            case 3: graph1(setpoint,Ymax,Time_range,3);
                    fuzzy(Time_range,setpoint,Ymax);
                    PID_TF(order);
                    pid(Time_range,setpoint,Ymax);
                  }
     while(!stop){
       while(!bioskey(1));
          c.i = bioskey(0);
          if(c.ch[0]==27)
             stop = 1;
          switch(c.ch[1]){
                case 59: run_open(Time_range,setpoint,Ymax);
                         break;
                case 60: run_close(Time_range,setpoint,Ymax);
                         break;
                case 61: for(t=0;t<SCALEMAX-1;t++){
                             if(ci_dat[t]<1.5)
                             plotgraph(ci_dat[t],ci_dat[t+1],t,
                                       Ymax,MAGENTA);
                           }
                }
      }
     closegraph();
}
```

table.h

```
/*****************************************************************/
/*        Create fuzzy table for fuzzy controller                */
/*****************************************************************/

void Fuzzy_table(void *t_er, void *t_ce, void *t_ci, int qer_max
                ,int qce_max, int qci_max ,void *t_table)
{
  extern int    r_max ;
  extern int    rule[MAX][3] ;

  float (*er)[MAX][MAX], (*ce)[MAX][MAX], (*ci)[MAX][MAX];
  float (*table)[MAX][MAX];
  float cnt1, cnt2, cnt3, cnt4;
  float w, u1, u2, g;
  float mo[MAX], o[MAX];

      er = t_er;
      ce = t_ce;
      ci = t_ci;
      table = t_table;

      for(cnt1=0; cnt1<qer_max*2+1; cnt1++){
      for(cnt2=0; cnt2<qce_max*2+1; cnt2++){
         /*************************************************/
         /*         clear inference output array          */
         /*************************************************/
         for (cnt3=0; cnt3<qci_max*2+1; cnt3++){
             o[cnt3] = 0;
             }
         /*************************************************/
         /*               Inference                       */
         /*************************************************/
         for(cnt3=0; cnt3<r_max; cnt3++){
             if((*er)[rule[cnt3][0]][cnt1]<(*ce)[rule[cnt3][1]][cnt2])
                 w = (*er)[rule[cnt3][0]][cnt1];
              else
                 w = (*ce)[rule[cnt3][1]][cnt2];
              for(cnt4=0; cnt4<qci_max*2+1; cnt4++){
                 if(w > (*ci)[rule[cnt3][2]][cnt4])
                    mo[cnt4] = (*ci)[rule[cnt3][2]][cnt4];
                 else
                    mo[cnt4] = w;
                 }
              for(cnt4=0; cnt4<qci_max*2+1; cnt4++){
                 if (o[cnt4] < mo[cnt4])
                     o[cnt4] = mo[cnt4];
                 }
            }
         /*************************************************/
         /*     Defuzzification by calculate center of gravity     */
         /*************************************************/
         u1 = 0;
         u2 = 0;
         for(cnt3=0; cnt3<qci_max*2+1; cnt3++){
             u1 = u1+o[cnt3];
             u2 = u2+o[cnt3]*cnt3;
            }
         if(u1==0)
            g = 0;
         else
            g = u2/u1-qci_max;
         /*************************************************/
         /*               create fuzzy table              */
         /*************************************************/
         (*table)[cnt1][cnt2] = g;
         }
         }
}
```

1

tf.h

```
extern float num[MAX],denom[MAX];
extern int   order;
/*****************************************************************/
/*             Input coeficient of Transfer function             */
/*****************************************************************/
int Getorder(void)
{
   gotoxy(67,9);
   cscanf("%d",&order);
   return(order);
}

void Getinput(void)
{
int cnt1, start, x;

   for(cnt1=0;cnt1<MAX;cnt1++){
       num[cnt1] = 0;
       denom[cnt1] = 0;
      }
   window(10,18,70,24);
   textbackground(CYAN);
   textcolor(BLUE);
   clrscr();
   draw_block(1,1,60,7);
   gotoxy(18,2);
   cprintf("PROCESS TRANSFER FUNCTION");
   start = 15-order;
   gotoxy(start,4);
   for(cnt1=order;cnt1>0;cnt1--){
       cprintf("     ");
       cprintf("S^%d +",cnt1);
      }
       cprintf("    S^0");
   gotoxy(start,5);
   for(cnt1=0;cnt1<8*(order+1);cnt1++)
       putch(0xC4);
   gotoxy(start,6);
   for(cnt1=order;cnt1>0;cnt1--){
       cprintf("     ");
       cprintf("S^%d +",cnt1);
      }
       cprintf("    S^0");
   textcolor(YELLOW);
   textbackground(WHITE);
   x = start;
   for(cnt1=order;cnt1>=0;cnt1--){
       gotoxy(x,4);
       cprintf("     ");
       x = x+10;
      }
   x = start;
   for(cnt1=order;cnt1>=0;cnt1--){
       gotoxy(x,6);
       cprintf("     ");
       x = x+10;
      }
   x = start;
   for(cnt1=order;cnt1>=0;cnt1--){
       gotoxy(x,4);
       cscanf("%f",&num[cnt1]);
       x = x+10;
      }
   x = start;
   for(cnt1=order;cnt1>=0;cnt1--){
       gotoxy(x,6);
       cscanf("%f",&denom[cnt1]);
       x = x+10;
      }
   getch();
   textbackground(BLUE);
   clrscr();
   window(1,1,80,25);
}
```

1