การควบคุมหุ่นยนต์เดินตามเส้นโค้ง โดยใช้ฟัซซี่ลอจิกคอนโทล

FUZZY LOGIC CONTROL FOR CURVE LINE TRACKING ROBOT CONTROL

โดย

ณัฐพล จิระบวรภิญโญ

NATPOL JEERABORVORNPINYO

อาจารย์ที่ปรึกษา

ผศ.ดร.พรฤดี เนติโสภากุล

รายงานนี้เป็นส่วนหนึ่งของวิชาโครงการพัฒนาระบบงาน
หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
คณะเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ภาคฤดูร้อน ปีการศึกษา 2551

FUZZY LOGIC CONTROL FOR
CURVE LINE TRACKING ROBOT CONTROL

NATPOL JEERABORVORNPINYO

A SYSTEM DEVELOPMENT PROJECT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF SCIENCE PROGRAM IN INFORMATION TECHNOLOGY

FACULTY OF INFORMATION TECNOLOGY

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

SUMMER / 2008

หัวข้อ	การควบคุมหุ่นยนต์เดินตามเส้นโค้ง โดยใช้ฟัซซี่ลอจิกคอนโทล
นักศึกษา	นายณัฐพล จิระบวรภิญโญ
รหัสนักศึกษา	49066512
ปริญญา	วิทยาศาสตรมหาบัณฑิต
สาขาวิชา	เทคโนโลยีสารสนเทศ
แขนงวิชา	วิทยาการสารสนเทศ
ปีการศึกษา	2551
อาจารย์ที่ปรึกษา	ผศ.ดร. พรฤดี เนติโสภากุล

บทคัดย่อ

ปัจจุบันเทคโนโลยีด้านหุ่นยนต์ได้พัฒนาไปไกล ซึ่งหุ่นยนต์ในปัจจุบันนี้มีความสามารถที่หลากหลาย แต่การจะทำให้หุ่นยนต์นั้นสามารถทำงานสิ่งต่างๆ ที่เรากำหนดให้หุ่นยนต์นั้นทำ ก็ต้องมีระบบควบคุม เพื่อให้กับหุ่นยนต์นั้นสามารถทำงานตามที่เรากำหนดได้ โดยระบบควบคุมสำหรับใช้ในหุ่นยนต์นั้นก็ใช้ความรู้ทางคณิตศาสตร์ เพื่อมาคำนวณหาค่าพารามิเตอร์ต่างๆ ที่ใช้อยู่ในตัวหุ่นยนต์ใน และได้มีการพัฒนาระบบควบคุมให้อยู่ในรูปแบบอันโนมัติ เช่น ฟัซซี่ลอจิกคอนโทล , นิวรอนเน็ตเวิร์ก , จิเนติกอัลกอริทึ่ม หรือ ปัญญาประดิษฐ์ เป็นต้น

การจะนำระบบควบคุมอัตโนมัติเหล่านี้ไปใช้งานได้นั้น ก็ต้องทำการสร้างฮาร์ดแวร์ ซึ่งก็คือตัวหุ่นยนต์ ขึ้นมาก่อน แล้วทำการติดตั้งซอฟต์แวร์ที่ได้ของระบบควบคุมอัตโนมัติลงไปอีก จึงจะสามารถทำการทดสอบการทำงานของระบบควบคุมอัตโนมัตินี้ได้ ว่ามีการทำงานเป็นเช่นไร ซึ่งการสร้างฮาร์ดแวร์ นั้นต้องมีค่าใช้จ่ายที่สูง ให้ไม่สามารถพัฒนาความรู้ด้านหุ่นยนต์ไปได้ไกล เพราะเนื่องจากต้องใช้การลงทุนที่ค่อนข้างมากในการสร้างหุ่นยนต์แต่ละตัว

ทางไมโครซอฟต์ได้ออกซอฟต์แวร์ชื่อว่า Microsoft Robotics Developer Studio 2008 เพื่อใช้สำหรับออกแบบและจำลองการทำงานของหุ่นยนต์ เพื่อให้บุคคลทั่วไปที่มีเครื่องคอมพิวเตอร์สามารถเรียนรู้และพัฒนาเทคโนโลยีด้านหุ่นยนต์ได้

ซึ่งในโครงงานนี้ได้ใช้ระบบควบคุมอัตโนมัติ คือ ฟัซซี่ลอจิกคอนโทล ประยุกต์ใช้กับการควบคุมหุ่นยนต์ให้เดินตามเส้นโค้ง โดยใช้ซอฟต์แวร์ Microsoft Robotics Developer Studio 2008 สำหรับในการจำลองการทำงานของโปรแกรม

Title	Fuzzy Logic Control for Curve Line Tracking Robot Control
Student	Mr. Natpol Jeeraborvornpinyo
Student ID.	49066512
Degree	Master of Science
Programme	Information Science
Academic Year	2008
Advisor	Asst. Prof. Dr. Pornrudee Netisopakul

ABSTRACT

Nowadays, there is an increasing technology development in robot invention. Even though, robots are currently capable of doing various things, it still needs a specific software so as to get robots follow orders. To implement this kind of software requires a great deal of mathematics knowledge; otherwise, parameter calculation might be a threat for developers. Robotic control software now have automatic function such as Fuzzy Logic Control, Neuron Network , Genetic Algorithms or artificial intelligence, to name a few.

Hardware or robot structure needs to be built before planting automatic control software into robot; then software function can be tested afterward. However, developing hardware requires a significant amount of investment and this is the reason why robot development is still in the early age.

Microsoft launches Microsoft Robotics Developer Studio 2008 software created for simulation process. This Microsoft software is developed for those who are interested in robotic developer.

Fuzzy Logic Control is applied to this project in order to make robots walk along the curve line and Microsoft Robotics Developer Studio 2008 software is used for simulation program.

กิตติกรรมประกาศ

โครงงานฉบับนี้สำเร็จได้อย่างดี ด้วยคำแนะนำและคำปรึกษาจาก ผศ.ดร. พรฤดี เนติโสภากุล ขอกราบขอบพระคุณอาจารย์เป็นอย่างสูง

ขอกราบขอบพระคุณคณาจารย์คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ที่ได้ประสิทธิ์ประสาทวิชาให้กับข้าพเจ้า

ขอขอบคุณเพื่อนๆ พี่ๆ น้องๆ ในคณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยี พระจอมเกล้าเจ้าคุณทหารลาดกระบัง ทุกคนที่ให้คำแนะนำต่างๆ และคอยให้กำลังใจเสมอมา

สุดท้ายนี้ข้าพเจ้าขอกราบขอบพระคุณ บิดา มารดา และครอบครัวของข้าพเจ้าที่เป็นกำลังใจและให้การสนับสนุนในทุกเรื่อง ทำให้ข้าพเจ้าสามารถพัฒนาระบบงานนี้สำเร็จลุล่วงด้วยดี

ณัฐพล จิระบวรภิญโญ

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

รูปที่ หน้า

สารบัญรูป (ต่อ)

รูปที่ หน้า

สารบัญรูป (ต่อ)

รูปที่ หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันเทคโนโลยีด้านหุ่นยนต์ได้มีการพัฒนาไปอย่างมาก ซึ่งการจะให้หุ่นยนต์ทำงานได้นั้นก็ต้องมีการนำชุดคำสั่งต่างๆ ใส่เข้าไปในส่วนควบคุมของตัวหุ่นยนต์ เพื่อให้หุ่นยนต์สามารถทำงานตามที่เราได้กำหนดไว้ ตามแต่ละจุดประสงค์ของหุ่นยนต์แต่ละตัว โดยพื้นฐานที่หุ่นยนต์ต้องสามารถทำได้ คือ การเดิน เช่น เดินหน้า, เดินถอยหลัง, หมุนซ้าย หรือ หมุนขวา เป็นต้น

แต่ในการควบคุมหุ่นยนต์ที่มีความซับซ้อนขึ้นมาอีก คือ การนำการเดินมาประยุกต์ใช้กับสถานการณ์ต่างๆ เช่น การเดินขึ้นบันได, การเดินตามสิ่งของ หรือ การเดินตามเส้นแบบต่างๆ เป็นต้น ซึ่งการเดินเหล่านี้สามารถใช้ชุดคำสั่งพื้นฐานมาประยุกต์ใช้งานรวมกับทฤษฎีระบบควบคุมอัตโนมัติ เพื่อนำไปใช้ในการควบคุมการทำงานแบบต่างๆ

ซึ่งในการเดินตามเส้นโค้งของหุ่นยนต์นั้น จะมีสองอินพุท คือ ทิศทางของเส้นโค้งและความเร็วที่เคลื่อนที่ของหุ่นยนต์ และมีสองเอาท์พุท คือ ความเร็วของมอเตอร์และทิศทางการเคลื่อนที่ของหุ่นยนต์

วิธีการควบคุมที่สามารถทำการควบคุมระบบกระบวนการที่มีลักษณะเช่นนี้ได้แก่ วิธีการ State-Space ซึ่งเป็นวิธีการที่จะต้องทำการหาแบบจำลองทางคณิตศาสตร์ (Mathematical Model) ของระบบออกมาให้ได้อย่างถูกต้อง แล้วทำการวิเคราะห์ในเรื่องของการตอบสนองทางความถี่ (Frequency Response) โดยวิเคราะห์ Bode Plot , หาฟังก์ชั่นการถ่ายโอน (Transfer Function) ของระบบซึ่งจะหาออกมาเป็นสมการทางอนุพันธ์แล้วจึงแปลงรูปมาเป็นสมการรูปแบบลาปลาซ (Laplace Transform) แล้วจึงนำมาทำเป็นฟังก์ชั่นการควบคุมแบบป้อนกลับ (Feedback Control Function) ซึ่งจะต้องทำการวิเคราะห์แนวโน้มการควบคุมด้วยวิธีการต่างๆ เช่น การวิเคราะห์แบบ Root Locus แล้วนำมาทำการหาสมการของฟังก์ชั่นการควบคุมโดยสามารถนำเสนอได้ในลักษณะของสมการเชิงเส้นของระบบหลายๆ สมการ

การใช้วิธีการ State-Space ทำออกเป็นฟังก์ชั่นการควบคุมนั้น สามารถทำการควบคุมได้ถูกต้องอย่างแท้จริงก็โดยที่จะต้องหาแบบจำลองทางคณิตศาสตร์ ทำการวิเคราะห์คำนวณ และทำการแปลงสมการออกมาได้อย่างถูกต้องในทุกขั้นตอนจริงๆ ซึ่งจะต้องทำการวิเคราะห์และคำนวณที่ยุ่งยากซับซ้อนและเสียเวลามาก และหากขาดตกบกพร่องไปแม้แต่สิ่งเดียว หรือแม้แต่ขั้นตอนเดียว ก็จะไม่สามารถทำการควบคุมระบบได้อย่างที่ต้องการ

ซึ่งฟัซซี่ลอจิกคอนโทลเป็นระบบควบคุมที่สามารถใช้กับระบบกระบวนการที่มีความซับซ้อนเป็นอย่างมากและสามารถสร้างแบบจำลองทางคณิตศาสตร์ได้เป็นอย่างง่าย (นันทวัช อัชชารกูร. 2543 : 8)

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

การควบคุมหุ่นยนต์เดินตามเส้นโค้ง โดยใช้ฟัซซี่ลอจิกคอนโทล มีวัตถุประสงค์ในการศึกษา เพื่อเป็นกรอกและแนวทางในการศึกษาดังนี้

1. เพื่อศึกษาการกลไกการเดินตามเส้นโค้งของหุ่นยนต์
2. เพื่อศึกษาระบบควบคุมระบบอัตโนมัติ แบบฟัซซี่ลอจิกคนโทล (Fuzzy Logic Control) เพื่อนำไปประยุกต์ใช้งานกับระบบควบคุมที่มีความซับซ้อนได้อย่างมีประสิทธิภาพ
3. ออกแบบ, สร้างโปรแกรมและจำลองการทำงานการควบคุมการเดินตามเส้นโค้งของหุ่นยนต์ โดยใช้ซอฟต์แวร์ Microsoft Robotics Developer Studio 2008
4. ปรับแต่งพารามิเตอร์ต่างๆ เพื่อให้หุ่นยนต์สามารถเดินตามเส้นโค้งได้อย่างมีประสิทธิภาพ

1.3 ขอบเขตการของโครงงาน

ในโครงงานฉบับนี้จะเป็นการประยุกต์ทฤษฎีฟัซซี่ลอจิกคอนโทล (Fuzzy Logic Control) ในการควบคุมการเดินตามเส้นโค้งของหุ่นยนต์ โดยจะทำการจำลองหุ่นยนต์และสภาพแวดล้อมต่างๆ ให้เหมือนระบบจริง และจำระบบควบคุมแบบฟัซซี่เข้ามาทำการควบคุมระบบการทำงานของหุ่นยนต์ให้มีประสิทธิภาพอย่างที่ต้องการ ซึ่งใช้ซอฟต์แวร์ Microsoft Robotics Developer Studio 2008 ในการออกแบบระบบและจำลองการทำงานของระบบในรูปแบบของกราฟิก เพื่อให้ผู้ใช้สามารถทำการทดลองปรับแต่งค่าในการควบคุมให้มีการตอบสนองได้ตามต้องการและผู้ใช้สามารถนำไปศึกษาค้นคว้าเพิ่มเติมสามารถทำความเข้าใจและเรียนรู้ได้อย่างสะดวกและรวดเร็ว

1.4 ขั้นตอนของการศึกษา

โครงงานฉบับนี้ได้แบ่งเนื้อหาออกเป็น 5 บทด้วยกันคือ

บทที่ 1 กล่าวถึงความเป็นมาของโครงงาน ความมุ่งหมายและวัตถุประสงค์ ขอบเขตของการวิจัย และขั้นตอนการศึกษา

บทที่ 2 กล่าวถึงทฤษฎีพื้นฐานที่ใช้ในโครงงาน ซึ่งประกอบด้วย ทฤษฎีการควบคุมแบบฟัซซี่คอนโทล (Fuzzy Logic Control)

บทที่ 3 กล่าวถึงหลักการตัวควบคุมแบบฟัซซี่สำหรับหุ่นยนต์เดินตามเส้นโค้ง

บทที่ 4 กล่าวถึงการสร้างโปรแกรมและจำลองการทำงานของโปรแกรมสำหรับควบคุมหุ่นยนต์เดินตามเส้นโค้ง

บทที่ 5 เป็นบทสรุปผลการศึกษาและข้อเสนอแนะ

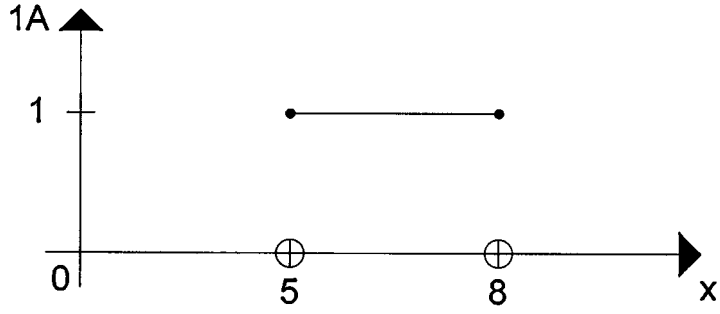
บทที่ 2
ทฤษฎีพื้นฐานที่ใช้ในการออกแบบระบบควบคุม

2.1 ระบบควบคุมแบบฟัซซี่ลอจิก

ระบบควบคุมแบบฟัซซี่ลอจิกคอนโทล (Fuzzy Logic Control System) คือ การควบคุมระบบอัตโนมัติโดยใช้หลักการของฟัซซี่ลอจิก โดยมีหลักการคือ สามารถหาค่าที่มีความคลุมเครือต่างๆ แล้วทำการคำนวณออกมาให้อยู่ในรูปแบบที่กำหนดไว้ได้ เช่น เครื่องปรับอากาศ โดยทั่วไปก็จะมีอินพุทเข้ามาคือ อุณหภูมิ ซึ่งจะสามารถบอกได้ว่าร้อนหรือเย็น โดยถ้าอากาศร้อนระบบควบคุมก็จะสั่งให้เครื่องปรับอากาศทำงาน ถ้าอากาศเย็นระบบควบคุมก็จะสั่งให้เครื่องปรับอากาศหยุดทำงาน แต่ถ้าใช้หลักการของฟัซซี่ลอจิก ก็จะมีการแบ่งอินพุทที่เข้าเป็นหลายๆ ระดับ เช่น อากาศร้อนมาก ร้อนน้อย ปานกลาง เย็น หรือเย็นมาก และทำการควบคุมเครื่องปรับอากาศให้ทำงานได้ตามค่าของอินพุท เช่น ถ้าอากาศร้อนมาก ก็ให้เครื่องปรับอากาศทำงาน 100% แต่ถ้าอากาศร้อนน้อยก็ให้เครื่องปรับอากาศทำงาน 50% เป็นต้น (นันทวัช อัชชารกูร. 2543 : 8)

2.1.1 ฟัซซี่เซ็ท (FUZZY SET)

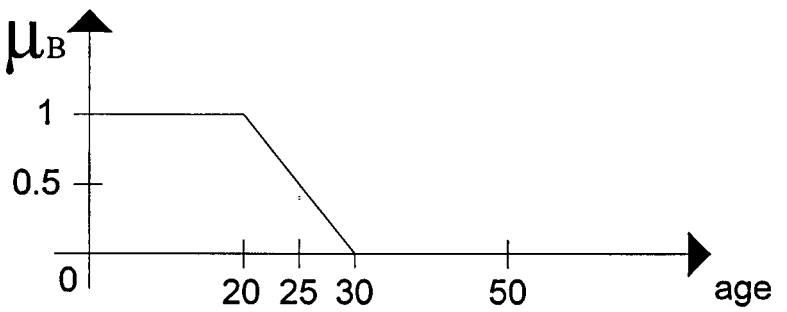
เซ็ทของค่าทางคณิตศาสตร์ โดยทั่วไปแล้วจะเป็นลักษณะอย่างง่ายๆ คือ ค่านั้นอยู่ในเซ็ทนี้หรือค่านั้นไม่อยู่ในเซ็ทนี้ เช่น เซ็ท A เป็นเซ็ทของค่าจำนวนจริงตั้งแต่ 5 ถึง 8 เขียนแทนเป็นช่วงได้เป็น A=[5,8] ดังรูป 2.1



รูปที่ 2.1 เซ็ทแบบธรรมดา (Crisp Set) ของ A=[5,8]

เซ็ทลักษณะตามรูปที่ 2.1 นั้น ถ้าค่าขององค์ประกอบใดๆ ไม่เป็นสมาชิกของเซ็ทนี้ก็จะถือว่าองค์ประกอบนั้นมีค่าทางตรรกเป็น 0 แต่ถ้าค่าขององค์ประกอบนั้นเป็นสมาชิกของเซ็ทนี้ก็จะถือว่าองค์ประกอบนั้นมีค่าทางตรรกเป็น 1 เซ็ทแบบนี้จะถือเป็นเซ็ทแบบปกติที่เรียกว่า Crisp Set ซึ่งค่า

ความเป็นสมาชิกทางตรรกที่มี 2 ค่า (0 กับ 1) นั้นจะไม่ละเอียดและไม่มีความยืดหยุ่นพอสำหรับการนำไปใช้งานในบางกรณี เช่น ถ้าบอกว่าเซ็ท B={เซ็ทของผู้เยาว์}=[อายุ0ปี,อายุ20ปี] นั้นหมายความว่าถ้าอายุ 20 ปี 1 วันก็ไม่ถือว่าเป็นผู้เยาว์แล้ว ซึ่งดูแล้วการตีความจะไม่ดีพอสำหรับกรณีนี้ ควรจะบอกว่าอายุ 0 ถึง 20 ยังมีความเป็นผู้เยาว์อยู่ พออายุมากกว่า 20 ขึ้นไปก็เริ่มมีความเป็นผู้เยาว์น้อยลง และหมดความเป็นผู้เยาว์ตอนอายุ 30 ปี ดังรูปที่ 2.2



รูปที่ 2.2 ถ้าอายุ 0 ถึง 20 ปีจะมีความเป็นผู้เยาว์เป็น 1 ถ้าอายุ 25 ปีก็จะถือว่ามีความเป็นผู้เยาว์อยู่ 0.5

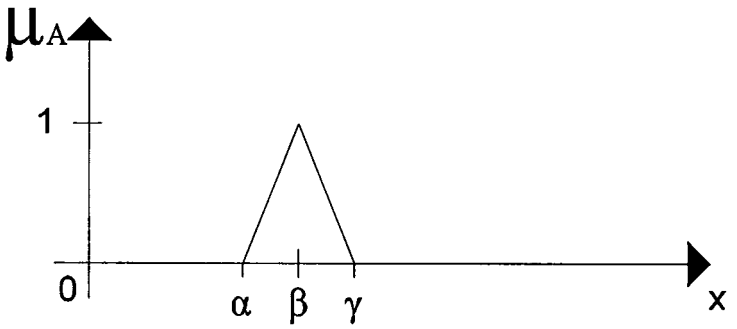
รูปแบบเซ็ทตามรูปที่ 2.2 นั้นสามารถมีค่าทางตรรกของความเป็นสมาชิกได้ตั้งแต่ค่า 0 ถึง 1 เซ็ทแบบนี้จะถือเป็นฟัซซี่เซ็ท (Fuzzy Set)

2.1.2 ฟังก์ชั่นความเป็นสมาชิก (MEMBERSHIP FUNCTION)

ฟังก์ชั่นความเป็นสมาชิกใช้เพื่อแสดงขอบเขตของค่าระดับความเป็นสมาชิกในแต่ละ ฟัซซี่เซ็ท ฟังก์ชั่นความเป็นสมาชิกของฟัซซี่เซ็ทมีด้วยกันหลายแบบและที่นิยมใช้กันได้แก่

- ฟังก์ชั่นความเป็นสมาชิกแบบสามเหลี่ยม (Triangular membership function): ฟัซซี่เซ็ทที่ใช้ฟังก์ชั่นความเป็นสมาชิกแบบนี้จะมีรูปร่างดังรูปที่ 2.3 ซึ่งค่าระดับความเป็นสมาชิก, μ_A ของฟังก์ชั่นจะนิยามได้ดังนี้

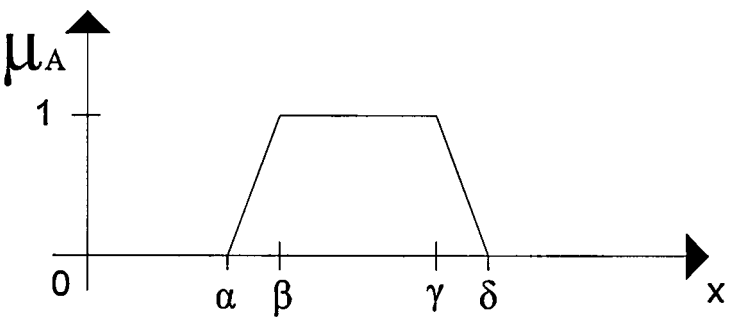
$$\mu_A(x) = \begin{cases} 0 & ; x > \alpha \\ (x-\alpha)/(\beta-\alpha) & ; \alpha \le x \le \beta \\ (\gamma-x)/(\beta-\alpha) & ; \beta \le x \le \gamma \\ 0 & ; x > y \end{cases} \quad \text{.....(2.1)}$$



รูปที่ 2.3 ฟังก์ชั่นความเป็นสมาชิกแบบสามเหลี่ยม

- ฟังก์ชั่นความเป็นสมาชิกแบบสามเหลี่ยมคางหมู (Trapezoidal membership function): ฟัซซี่เซ็ทที่ใช้ฟังก์ชั่นความเป็นสมาชิกแบบนี้จะมีรูปร่างดังรูปที่ 2.4 ซึ่งค่าระดับความเป็นสมาชิก, μ_A ของฟังก์ชั่นจะนิยามได้ดังนี้

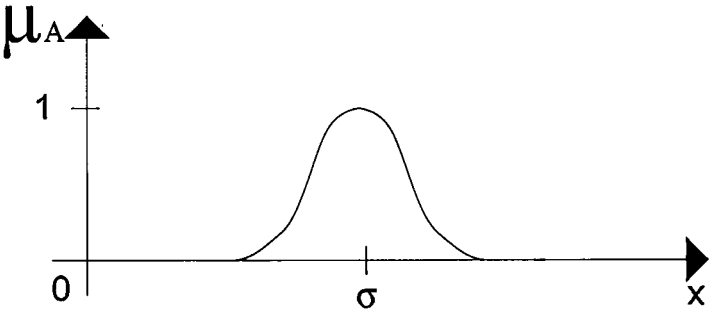
$$\mu_A(x) = \begin{cases} 0 & ; x > \alpha \\ (x-\alpha)/(\beta-\alpha) & ; \alpha \le x \le \beta \\ 1 & ; \beta \le x \le \gamma \\ (\gamma-x)/(\beta-\alpha) & ; \beta \le x \le \delta \\ 0 & ; x > \delta \end{cases} \quad \text{.....(2.2)}$$



รูปที่ 2.4 ฟังก์ชั่นความเป็นสมาชิกแบบสี่เหลี่ยมคางหมู

- ฟังก์ชั่นความเป็นสมาชิกแบบรูประฆังคว่ำ (Gaussian membership function): ฟัซซี่เซ็ทที่ใช้ฟังก์ชั่นความเป็นสมาชิกแบบนี้จะมีรูปร่างดังรูปที่ 2.5 ซึ่งค่าระดับความเป็นสมาชิก, μ_A ของฟังก์ชั่นจะนิยามได้ดังนี้

$$\mu_A(x) = \begin{cases} \exp\{-\dfrac{(x-c_1)^2}{2s_1^2}\} & ; x > \sigma \\ \exp\{-\dfrac{(x-c_2)^2}{2s_2^2}\} & ; x > \sigma \end{cases} \quad \text{.....(2.3)}$$

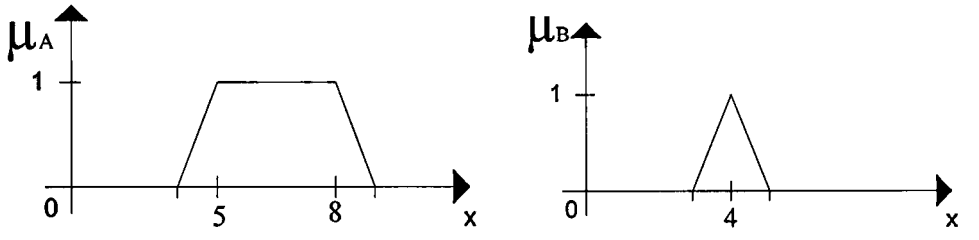


รูปที่ 2.5 ฟังก์ชั่นความเป็นสมาชิกแบบรูประฆังคว่ำ

2.1.3 การกระทำกันของฟัซซี่เซ็ท (OPERATIONS OF FUZZY SET)

การกระทำกันของฟัซซี่เซ็ทก็คล้ายกับการกระทำกันของ Crisp Set ดังตัวอย่างด้านล่าง

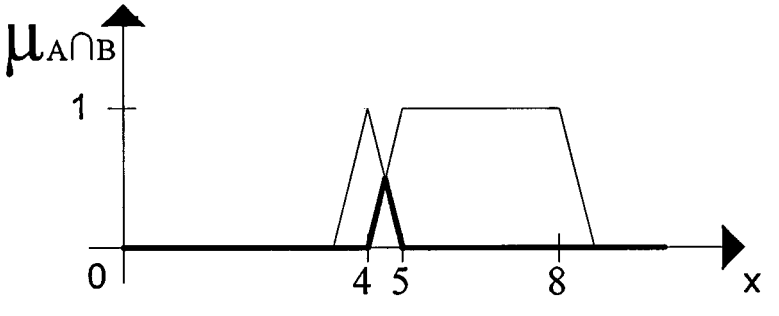
ตัวอย่าง



(a) (b)

รูปที่ 2.6 a) ฟัซซี่เซ็ท A, b) ฟัซซี่เซ็ท B

- การ Intersection หรือ AND Operation จะได้ผลลัพธ์ตามรูปที่ 2.7 ที่เป็นฟัซซี่เซ็ทเส้นหนา

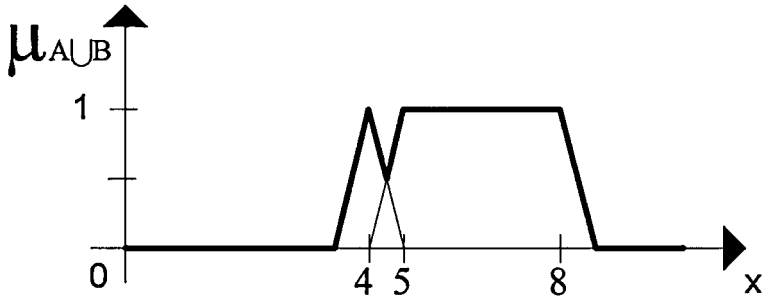


รูปที่ 2.7 ฟัซซี่เซ็ท A AND B

โดยทั่วไปการ Intersection ของฟัซซี่จะใช้ตัวกระทำการในการหาค่าต่ำสุดดังสมการ

$$A \cap B \longleftrightarrow \mu_{A \cap B}(x) = \min\{\mu_A(x), \mu_B(x)\} \quad \text{.....(2.4)}$$

- การ Union หรือ OR Operation จะได้ผลลัพธ์ตามรูปที่ 2.8 ที่เป็นฟัซซี่เซ็ทเส้นหนา

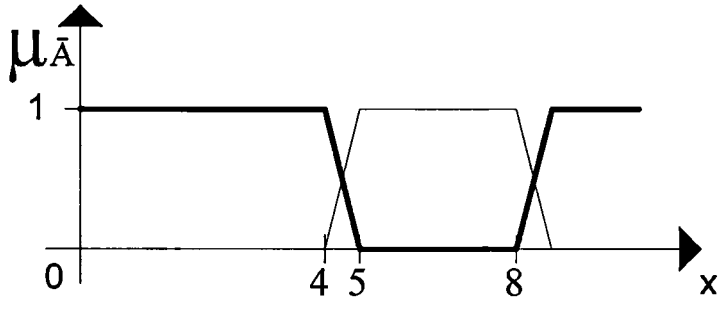


รูปที่ 2.8 ฟัซซี่เซ็ท A OR B

โดยทั่วไปการ Union ของฟัซซี่เซ็ทจะใช้ตัวกระทำการในการหาค่าสูงสุดดังสมการ

$$A \cup B \longleftrightarrow \mu_{A \cup B}(x) = \max\{\mu_A(x), \mu_B(x)\} \quad \text{.....(2.5)}$$

- การ Complement หรือ Negation จะได้ผลลัพธ์ตามรูปที่ 2.9 ที่เป็นฟัซซี่เซ็ทเส้นหนา



รูปที่ 2.9 ฟัซซี่เซ็ท NOT A

โดยทั่วไปการ Complement ของฟัซซี่เซ็ทจะใช้ตัวกระทำการในการหาค่าที่ไม่อยู่ในฟัซซี่เซ็ทดังสมการ

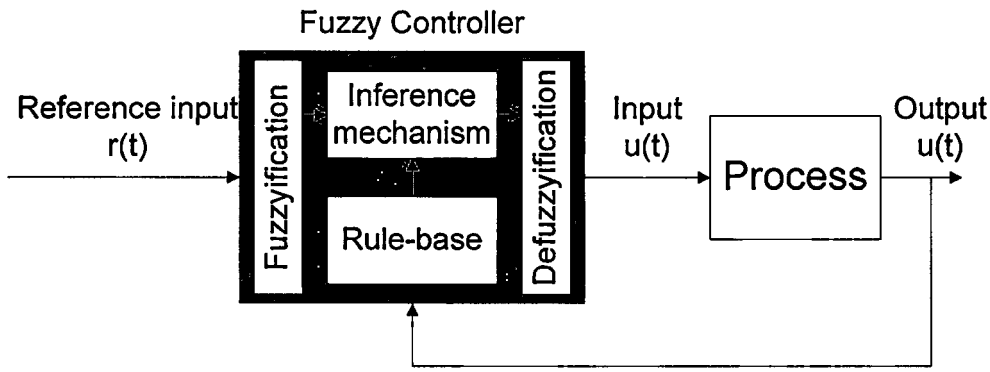
$$A' \longleftrightarrow \mu_{A'}(x) = 1 - \mu_A(x) \quad \text{.....(2.6)}$$

2.2 การควบคุมแบบฟัซซี่ (Fuzzy Control)

การควบคุมแบบฟัซซี่นั้นจะมีหลักการต่างจากการควบคุมตามวิธีการแบบดั้งเดิม (Conventional Control) พอสมควร ซึ่งวิธีการควบคุมแบบดั้งเดิมนั้นโดยทั่วไปจะต้องทำการหาแบบจำลองทางคณิตศาสตร์ (Mathematical Model) ขององค์ประกอบส่วนต่างๆ ของระบบกระบวนการ (Process System) แล้วนำมาทำการสร้างระบบควบคุม (Control System) ออกมาในรูปของสมการทางอนุพันธ์ แต่สำหรับวิธีการควบคุมแบบฟัซซี่นั้นจะอาศัยประสบการณ์ความชำนาญของผู้วิเคราะห์ในการที่จะถ่ายทอดออกมาเป็นค่าตัวแปรทางความคิดความรู้สึก (Linguistic Variable) ซึ่งสามารถอธิบายค่าปริมาณของตัวแปรพวกนี้ได้จากฟัซซี่เซ็ท (Fuzzy Set) โดยจะอธิบายตามตัวอย่างซึ่งเป็นการนำวิธีการควบคุมแบบฟัซซี่ไปใช้สำหรับการควบคุมการควบคุมหุ่นยนต์เดินตามเส้นโค้งในเนื้อหาบทถัดไป (นันทวัช อัชชารกูร. 2543 : 12)

2.2.1. องค์ประกอบหลักของตัวควบคุมแบบฟัซซี่เซ็ท

ตัวควบคุมแบบฟัซซี่เซ็ท (Fuzzy Controller) จะมีองค์ประกอบหลักอยู่ 4 อย่างดังรูปที่ 2.10



รูปที่ 2.10 บล็อกไดอะแกรมตัวควบคุมแบบฟัซซี่โดย Reference Input, r(t) คือ ค่าที่เราต้องการตั้งไว้เพื่อทำการควบคุมกระบวรการให้ได้ค่านี้ (Setpoint)

โดยใน คือ ตัวควบคุมแบบฟัซซี่จะมีส่วนประกอบหลักต่างๆ คือ

1. กฎเงื่อนไขพื้นฐาน (Rule-Base):- เป็นกลุ่มของกฎเงื่อนไขถ้า-แล้ว (if-then rules) ซึ่งเป็นกฎที่ตั้งจากความรู้สึกนึกคิดทางตรรกของผู้ชำนาญระบบนั้นว่า ถ้าเป็นเงื่อนไขนั้นควรจะทำอย่างไรเพื่อให้การควบคุมดี โดยกฎเงื่อนไขพื้นฐานนี้มักออกแบบมาให้อยู่ในรูปแบบดังนี้

กฎข้อที่ i : ถ้า x_1 เป็น $A_1^{(i)}$ และ...และ x_n เป็น $A_n^{(i)}$ แล้ว y เป็น $B^{(i)}$ .....(2.7)

2. ส่วนที่ทำการแปลงให้เป็นค่าทางฟัซซี่ (Fuzzification interface):- เป็นส่วนที่แปลงค่าอินพุทของตัวควบคุมหรือตัวแปรกระบวนการ (Controller inputs or Process Variable) ที่เป็นตัวเลขให้มาเป็นข้อมูลค่าทางฟัซซี่ ซึ่งให้กลไกการวิเคราะห์สามารถใช้กฎเงื่อนไขพื้นฐานในการทำการวิเคราะห์ค่าพวกนี้ได้ โดยการแปลงให้เป็นค่าทางฟัซซี่นั้นก็จะเป็นการนำค่าอินพุทที่เป็นค่าตัวเลขปกติ, x ไปเป็นอินพุทของฟังก์ชั่นความเป็นสมาชิกที่ใช้

3. กลไกการวิเคราะห์ทางฟัซซี่ (Fuzzy inference mechanism):- เป็นส่วนที่จะทำการคัดเลือกกฎเงื่อนไขพื้นฐาน เพื่อนำอินพุทที่เป็นค่าทางฟัซซี่มาทำการวิเคราะห์ออกมาเป็นเอาท์พุทค่าทางฟัซซี่ โดยกลไกการวิเคราะห์ก็มีใช้กันอยู่หลายวิธี ได้แก่

   - การใช้ตัวดำเนินการแบบหาค่าน้อยที่สุด (Min-Operation of fuzzy implication)

     $$\mu_{A \longrightarrow B}(x, y) = \min\{\mu_A(x), \mu_B(y)\} \quad \text{.....(2.8)}$$

   - การใช้ตัวดำเนินการแบบผลคูณ (Product-Operation of fuzzy implication)

     $$\mu_{A \longrightarrow B}(x, y) = \mu_A(x) \cdot \mu_B(y)\} \quad \text{.....(2.9)}$$

   - การวิเคราะห์โดยใช้กฎเชิงเลขคณิต (Arithmetic Rule of fuzzy implication)

     $$\mu_{A \longrightarrow B}(x, y) = \min\{1, 1 - \mu_A(x) + \mu_B(y)\} \quad \text{.....(2.10)}$$

   - การวิเคราะห์โดยใช้กฎค่าสูงสุดของค่าต่ำสุด (Maxmin Rule of fuzzy implication)

     $$\mu_{A \longrightarrow B}(x, y) = \max\{\min\{\mu_A(x), \mu_B(y)\}, 1 - \mu_A(x)\} \quad \text{.....(2.11)}$$

   - การวิเคราะห์โดยใช้กฎบูลีน (Boolean Rule of fuzzy implication)

     $$\mu_{A \longrightarrow B}(x, y) = \max\{1 - \mu_A(x), \mu_B(y) \quad \text{.....(2.12)}$$

   - การวิเคราะห์โดยใช้กฎของ Goguen (Goguem's Rule of fuzzy implication)

$$\mu_{A \rightarrow B}(x, y) = \begin{cases} 1 & ; \mu_A(x) \leq \mu_B(y) \\ \dfrac{\mu_B(y)}{\mu_A(x)} & ; \mu_A(x) > \mu_B(y) \end{cases} \quad \text{.....(2.13)}$$

4. ส่วนที่ทำการแปลงออกมาเป็นค่าเพื่อส่งออกไปสั่งการควบคุม (Defuzzification interface):- จะทำการคำนวณแปลงค่าทางฟัซซี่จากการวิเคราะห์ของกลไกการวิเคราะห์ออกมาเป็นค่าผลลัพธ์ที่เป็นตัวเลขและส่งออกไปสั่งการควบคุมอินพุทของกระบวนการ โดยมีอยู่ด้วยกันหลายวิธีได้แก่

   - วิธีการหาค่ากลางของพื้นที่ (Center of area defuzzification) หรืออาจเรียกว่าการหาจุดศูนย์ถ่วง (center of gravity, COG) หรือ แบบเซนทรอยด์ (centroid)

$$\text{output} = \frac{\sum_i b_i \int \mu_{(i)}}{\sum_i \int \mu_{(i)}} \quad \text{.....(2.14)}$$

   โดย b_i คือ ค่าจุดศูนย์กลางของฟัซซี่เซ็ทย่อยแต่ละรูปในฟัซซี่เซ็ทรวม

   $\int \mu_{(i)}(x)$ คือ พื้นที่ใต้กราฟของฟังก์ชั่นความเป็นสมาชิกของฟัซซี่เซ็ทย่อยแต่ละรูป ในฟัซซี่เซ็ทรวมฟังก์ชั่นความเป็นสมาชิกรูปกราฟสามเหลี่ยมที่มีความเป็นสมาชิกสูงสุดเป็น 1 ซึ่งจะถูกตัดออกโดยค่าระดับความเป็นสมาชิก จะมีค่าเป็น:-

$$w\left(h - \frac{h^2}{2}\right) \quad \text{.....(2.15)}$$

   โดย w คือ ค่าความกว้างของฐานของรูปกราฟสามเหลี่ยมของฟัซซี่เซ็ทฟังก์ชั่นความเป็นสมาชิก

   h คือ ค่าความสูงของรูปกราฟสามเหลี่ยมที่ถูกตัดยอดออก ซึ่งก็คือค่าระดับความเป็นสมาชิกของฟัซซี่เซ็ทย่อยแต่ละรูปนั่นเอง

   - วิธีการหาจุดศูนย์กลางเฉลี่ย (Center-Average) ซึ่งสามารถคำนวณได้ง่ายกว่าวิธีการ COG โดยสามารถคำนวณค่าออกได้จากสมการ:-

$$output = \frac{\sum_i b_i \int \mu_{premise(i)}}{\sum_i \int \mu_{premise(i)}} \quad \text{.....(2.16)}$$

โดย $\mu_{premise(i)}$ คือ ค่าระดับความเป็นสมาชิกของฟัซซี่เซ็ทผลลัพธ์ย่อยแต่ละรูป

วิธีการหาค่าเอาท์พุทผลลัพธ์ จากการหาจุดศูนย์กลางของรูปกราฟของฟัซซี่เซ็ทฟังก์ชั่นความเป็นสมาชิกผลลัพธ์รวม (Defuzzification) ทั้งสองวิธีนี้ จะคำนวณค่าออกมาได้แตกต่างกัน ซึ่งการจะพิจารณาว่าวิธีการไหนจะดีกว่ากันในการนำไปใช้ควบคุมระบบกระบวนการของเรานั้นจะต้องขึ้นอยู่กับการทดสอบจำลองการทำงาน (simulation) และการนำไปใช้งาน (implementation) ต่อไป

2.3 คำสั่งต่างๆ ที่ใช้ในเครื่องมือ Visual Programming Language 2008 มีดังนี้

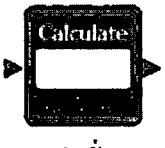
- คำสั่ง Data ใช้สำหรับกำหนดค่าในรูปแบบต่างๆ ดังตารางที่ 2.1

รูปที่ 2.11 คำสั่ง Data

ตารางที่ 2.1 ประเภทของข้อมูล

VPL Type	Description
bool	Boolean values: true, false
byte	8 bit unsigned integer (0 to 255)
sbyte	8 bit signed integer (-128 to 127)
char	character
decimal	fixed point decimal number (fixed precision number)
double	double precision (64-bit) floating point number (approx 14 significant digits)
float	single precision (32-bit) floating point number (approx 7 significant digits)
int	32 bit signed integer
uint	32 bit unsigned integer
long	64 bit signed integer
ulong	64 bit unsigned integer
short	16 bit signed integer (-32768 to 32767)
ushort	16 bit unsigned integer (0 to 65535)
string	character string (text)

- คำสั่ง Calculate ใช้สำหรับคำนวณค่าตัวแปรที่เข้ามา โดยมีรูปแบบการกระทำกันของ Calculate ดังตารางที่ 2.2

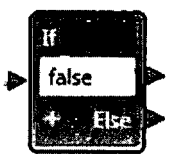


รูปที่ 2.12 คำสั่ง Calculate

ตารางที่ 2.2 การกระทำกันของค่าตัวแปรของคำสั่ง Calculate

Command	Description
+	add
-	subtract/minus
*	multiply
/	divide
%	mod (The modulus operation returns the remainder after a division)
&&	AND
||	OR
!	NOT

- คำสั่ง If ใช้สำหรับกำหนดเงื่อนไขของต่างๆ ของข้อมูลหรือตัวแปรที่เข้ามา เพื่อทำการเลือกเอาท์พุทตามที่กำหนดจากอินพุทได้ โดยมีรูปแบบการกระทำกันของ If ดังตารางที่ 2.3



รูปที่ 2.13 คำสั่ง If

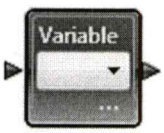
ตารางที่ 2.3 การกระทำกันของค่าตัวแปรของคำสั่ง If

Command	Description
= or ==	equals
!= or <>	not equals
<	less than
>	greater than
<=	less than or equals
>=	greater than or equals

- คำสั่ง Merge ใช้สำหรับรวมข้อมูลหรือตัวแปรต่างๆ เพื่อส่งข้อมูลเหล่านี้ไปยังคำสั่งถัดไป

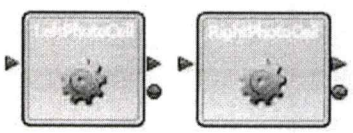
รูปที่ 2.14 คำสั่ง If

- คำสั่ง Variable เป็นตัวแปรที่ใช้สำหรับเก็บข้อมูลต่างๆ โดยมีประเภทของข้อมูลเช่นเดียวกับคำสั่ง Data ดังตารางที่ 2.1



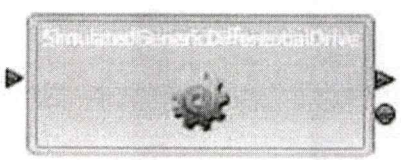
รูปที่ 2.15 คำสั่ง Variable

- คำสั่ง PhotoCell เป็นฟังก์ชั่นสำหรับใช้อ่านค่าของสีต่างๆ ของทางด้านซ้ายและขวา



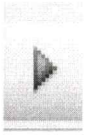
รูปที่ 2.16 คำสั่ง Photo Cell

- คำสั่ง SimulatedGenericDifferentialDrive เป็นฟังก์ชั่นสำหรับควบคุมการทำงานของมอเตอร์ โดยสามารถกำหนดค่าต่างๆ คือ ความเร็ว, กำลัง, ทิศทางหรือแรงบิดเป็นต้น



รูปที่ 2.17 คำสั่ง SimulatedGenericDifferentialDrive

- คำสั่ง Run ใช้สำหรับจำลองการทำงานของโปรแกรมที่ได้ทำการสร้างขึ้นมา



รูปที่ 2.17 คำสั่ง Run

บทที่ 3
หลักการควบคุมหุ่นยนต์เดินตามเส้นโค้งโดยใช้ฟัซซี่ลอจิกคอนโทล

แนวทางในการควบคุมหุ่นยนต์เดินตามเส้นโค้งในเนื้อหาส่วนนี้จะอธิบายเป็นตัวอย่างอย่างง่ายที่ประยุกต์สำหรับการควบคุมมอเตอร์ด้านซ้ายและขวา ซึ่งทำหน้าขับเคลื่อนและกำหนดทิศทางของหุ่นยนต์ โดยถือเป็นเอาท์พุทตัวหลักของระบบ

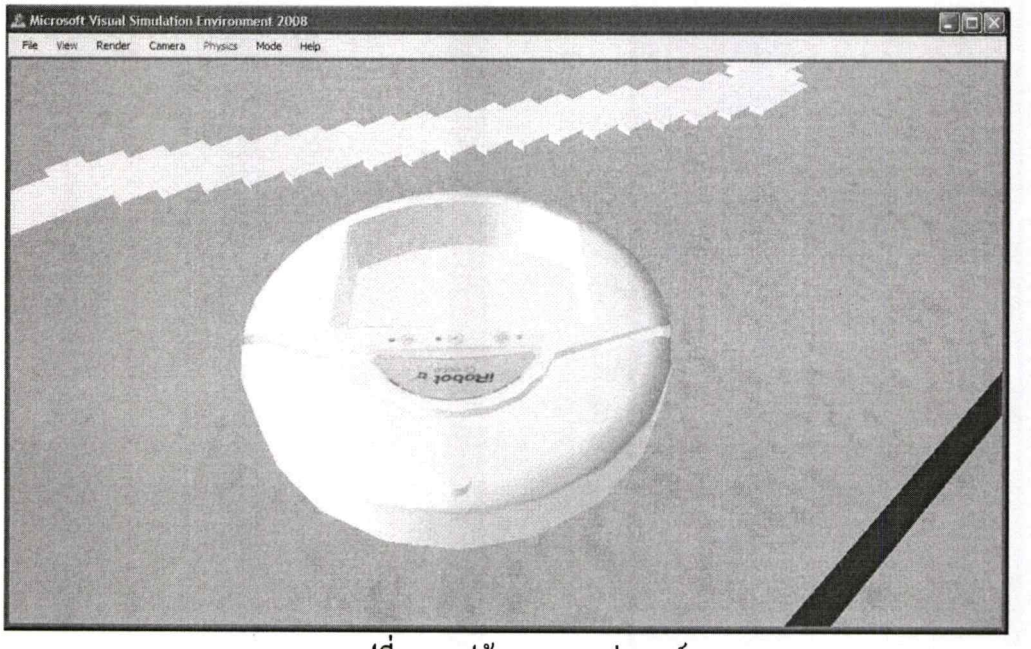
3.1 หลักการในควบคุมหุ่นยนต์ให้เดินตามเส้นโค้ง

ทำการหาความสัมพันธ์ระหว่างการควบคุมมอเตอร์ซ้ายและขวาของตัวหุ่นยนต์กับทิศทางการเคลื่อนที่และความเร็วของตัวหุ่นยนต์ เพื่อให้หุ่นยนต์สามารถเดินตามเส้นโค้งที่กำหนดได้โดยไม่หลุดออกไปนอกเส้นทาง

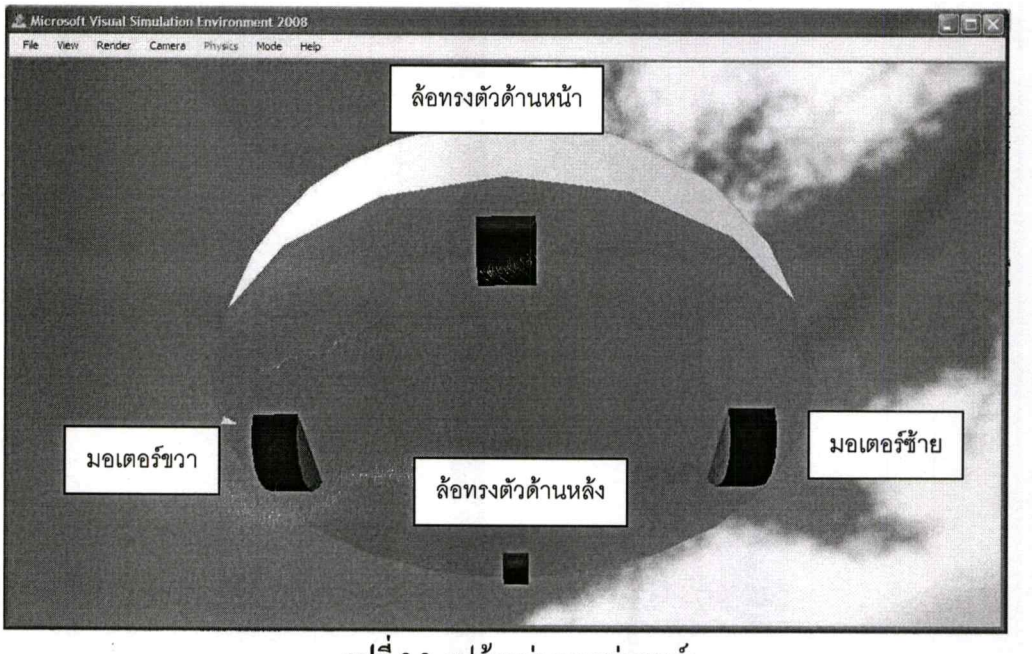
3.1.1 ศึกษาหลักการทำงานสำหรับควบคุมหุ่นยนต์ให้เดินตามเส้นโค้ง

ทำการศึกษาการทำงานของหุ่นยนต์เดินตามเส้นโค้งจาก www.ProMRDS.com โดยโปรแกรมชื่อว่า FollowLine และใช้ซอฟต์แวร์ของ Microsoft Robotics Develop Studio 2008 โดยใช้เครื่องมือ Visual Programming Language 2008 ในการอิมพลีเมนต์ ซึ่งตัวหุ่นยนต์ สภาพแวดล้อมและค่าพารามิเตอร์ต่างๆ เป็นดังนี้

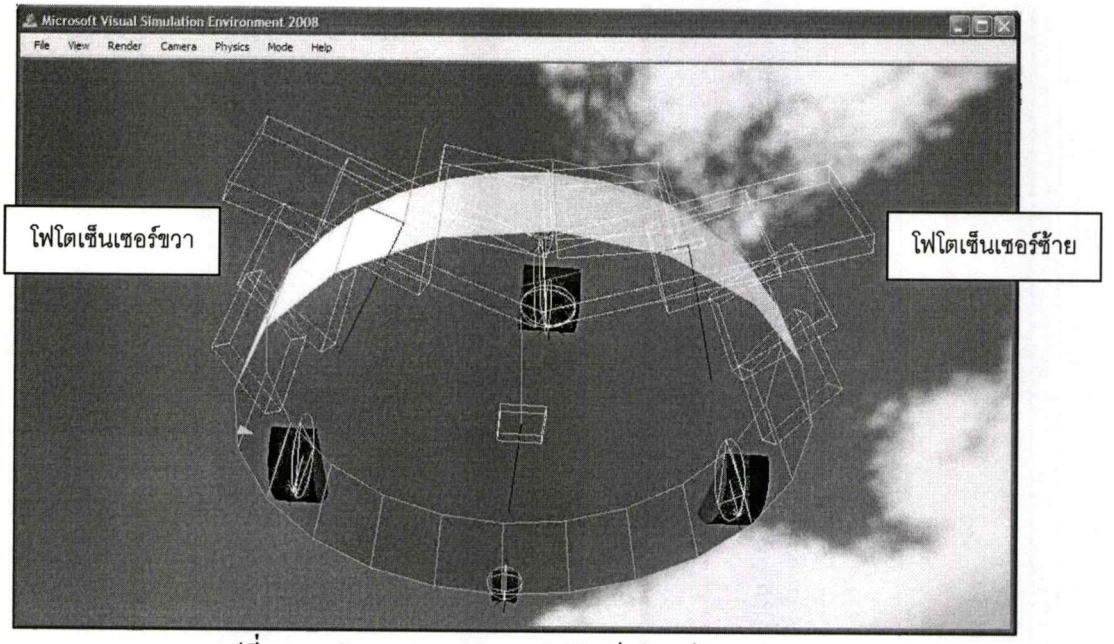
- ตัวหุ่นยนต์ ประกอบด้วยโฟโตเซ็นเซอร์ (Foto Sensor) จำนวน 2 ตัว ติดตั้งอยู่ใต้บริเวณด้านหน้าซ้ายและขวาของตัวหุ่นยนต์, มีมอเตอร์สำหรับควบคุมล้อจำนวน 2 ตัว ติดตั้งอยู่ด้านซ้ายและขวาอย่างละ 1 ตัว ซึ่งเป็นมอเตอร์แบบที่สามารถควบคุมองศาในการหมุน ความเร็วหรือแรงบิดได้ และมีล้อสำหรับทรงตัวจำนวน 2 ล้อติดตั้งอยู่ที่ด้านหน้าและหลังของตัวหุ่นยนต์ ดังรูปที่ 3.1, 3.2 และ 3.3



รูปที่ 3.1 รูปด้านบนของหุ่นยนต์

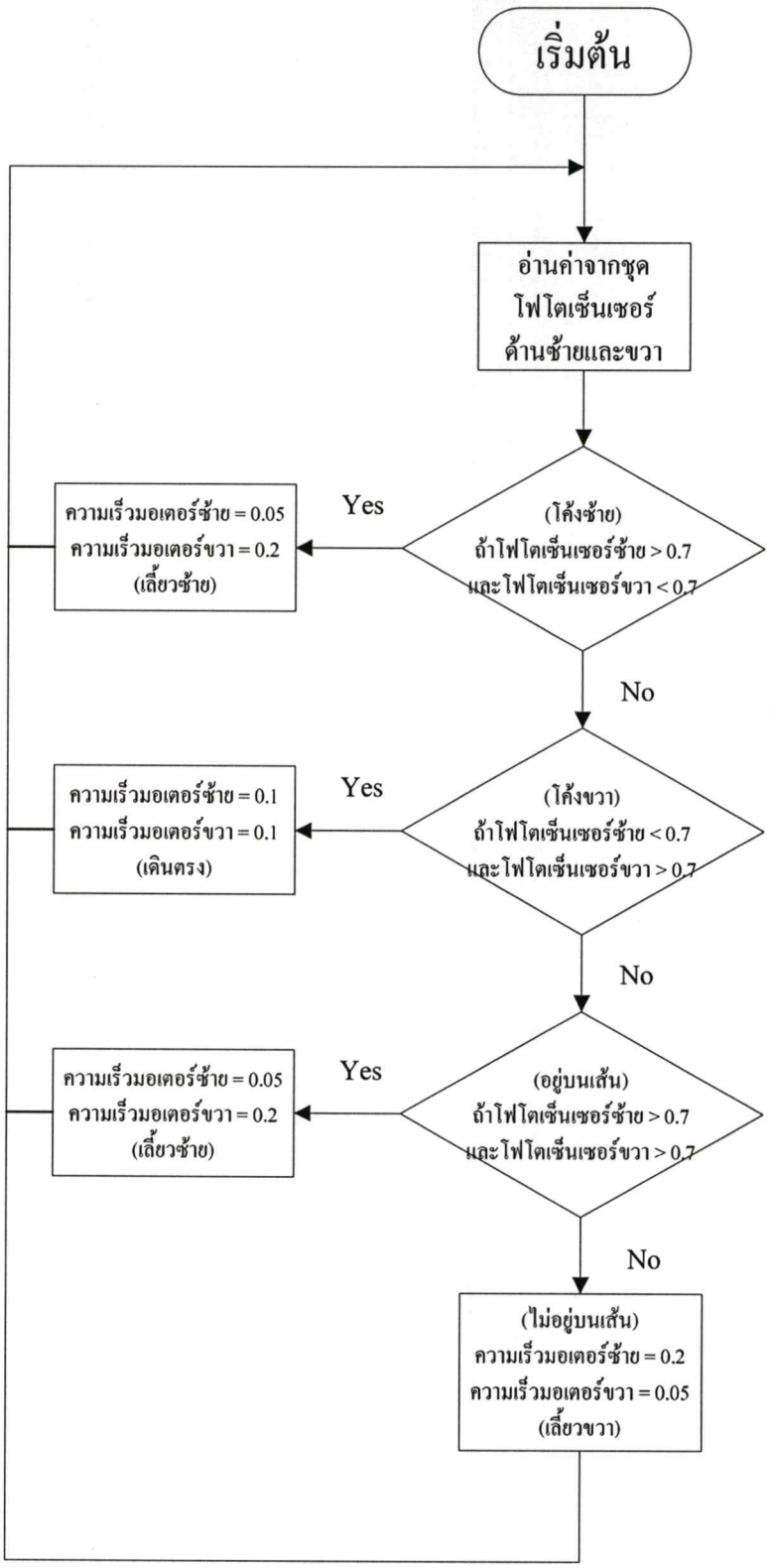


รูปที่ 3.2 รูปด้านล่างของหุ่นยนต์



รูปที่ 3.3 รูปแสดงตำแหน่งของโฟโตเซ็นเซอร์(Foto Sensor)

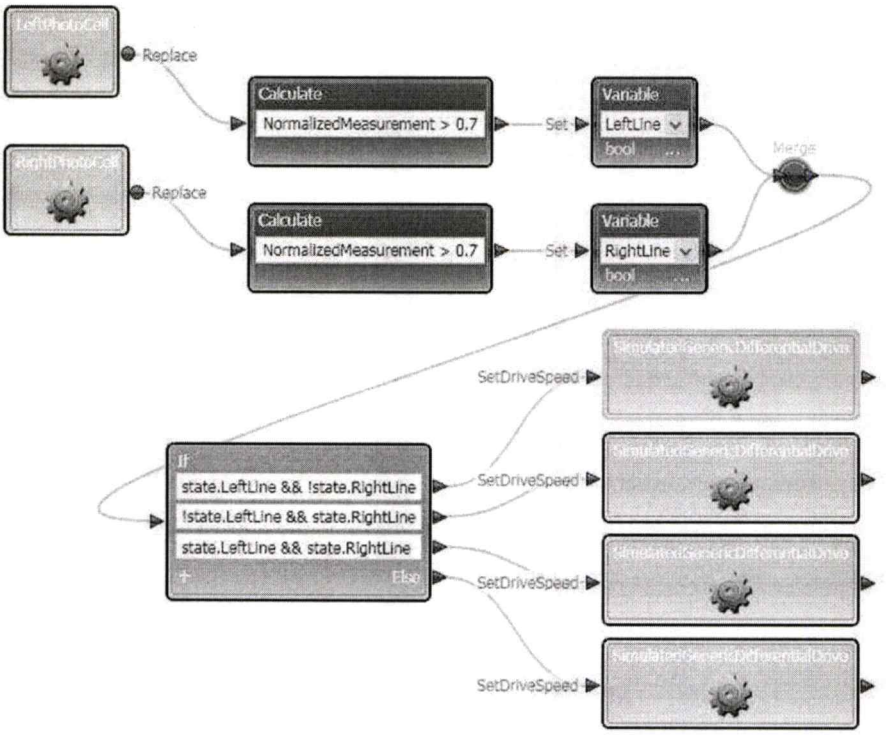
- สภาพแวดล้อมที่จะให้หุ่นยนต์ทำงานนั้นจะเป็นพื้นที่ราบ โดยมีเส้นสีขาวและพื้นสีเทา เพื่อให้หุ่นยนต์เดิมตามเส้นนี้ไปยังจุดกำหนดไว้ ซึ่งในการเดินตามเส้นนี้ต้องใช้อุปกรณ์การตรวจจับเส้นสีขาว คือ โฟโตเซ็นเซอร์ (Foto Sensor)
- อินพุทของระบบ จะใช้โฟโตเซ็นเซอร์ (Foto Sensor) จำนวน 2 ตัวติดตั้งไว้ที่ด้านหน้าของหุ่นยนต์ เพื่อทำหน้าที่ตรวจจับว่าเส้นโค้งกับตัวหุ่นยนต์นั้นมีทิศทางเป็นอย่างไร โดยใช้หลักการความแตกต่างของสีทั้ง 2 โดยเมื่อโฟโตเซ็นเซอร์ (Foto Sensor) ตรวจพบสีเทาจะให้เอาท์พุทออกมาเป็นค่า 0.628 , เมื่อตรวจพบขอบสีขาวจะให้เอาท์พุทมาเป็นค่า 0.88 และเมื่อตรวจพบสีขาวจะให้เอาท์พุทมาเป็นค่า 0.919 โดยทำการอ่านค่าจากโฟโตเซ็นเซอร์ (Foto Sensor) ทั้ง 2 ด้าน โดยโฟโตเซ็นเซอร์ด้านใดมีค่ามากกว่า 0.7 ก็แสดงว่าตรวจพบเส้นโค้งไปทางด้านนั้น
- เอาท์พุทของระบบ คือความเร็วของมอเตอร์ด้านซ้ายและขวา ซึ่งมีหน่วยเป็นเมตรต่อวินาที (m/s) โดยเมื่อต้องการเลี้ยวไปทางใดก็ให้มอเตอร์ด้านนั้นมีความเร็วน้อยกว่ามอเตอร์อีกด้านหนึ่ง
- ขั้นตอนการทำงานของโปรแกรม FollowLine ดังรูปที่ 3.4



รูปที่ 3.4 โฟลวชาร์ตแสดงขั้นตอนการทำงานของโปรแกรม FollowLine

- จากโฟลวชาร์ตแสดงขั้นตอนการทำงานของโปรแกรม FollowLine นำมาอิมพลีเมนต์ด้วย Visual Programming Language 2008 ดังรูปที่ 3.5 ซึ่งการทำงานของโปรแกรม โดยเริ่มต้นจากโฟโตเซ็นเซอร์ (Foto Sensor) ทั้งด้านซ้ายและขวาทำการอ่านค่าสีที่ตำแหน่งนั้นๆ นำค่าที่ได้มาทำการคำนวณว่ามีค่ามากกว่า 0.7 หรือไม่ ถ้าโฟโตเซ็นเซอร์ (Foto Sensor) ด้านซ้ายอ่านค่าสีได้มากกว่า 0.7 ให้ทำการกำหนดตัวแปร LeftLine ให้มีค่าเป็นจริง และถ้าโฟโตเซ็นเซอร์ (Foto Sensor) ด้านซ้ายอ่านค่าสีได้น้อยกว่าหรือเท่ากับ 0.7 ให้ทำการกำหนดตัวแปร LeftLine นั้นให้มีค่าเป็นเท็จและถ้าโฟโตเซ็นเซอร์ (Foto Sensor) ด้านขวาอ่านค่าสีได้มากกว่า 0.7 ให้ทำการกำหนดตัวแปร RightLine ให้มีค่าเป็นจริง และถ้าโฟโตเซ็นเซอร์ (Foto Sensor) ด้านขวาอ่านค่าสีได้น้อยกว่าหรือเท่ากับ 0.7 ให้ทำการกำหนดตัวแปร RightLine นั้นให้มีค่าเป็นเท็จ นำตัวแปรทั้ง 2 ตัวไปทำการตรวจสอบเงื่อนไขดังต่อไปนี้

  1. ถ้า LeftLine เป็นจริงและ RightLine เป็นเท็จ ให้กำหนดค่าความเร็วมอเตอร์ด้านซ้ายเท่ากับ 0.05 m/s และความเร็วมอเตอร์ด้านขวาเท่ากับ 0.2 m/s
  2. ถ้า LeftLine เป็นเท็จและ RightLine เป็นจริง ให้กำหนดค่าความเร็วมอเตอร์ด้านซ้ายเท่ากับ 0.1 m/s และความเร็วมอเตอร์ด้านขวาเท่ากับ 0.1 m/s
  3. ถ้า LeftLine เป็นจริงและ RightLine เป็นจริง ให้กำหนดค่าความเร็วมอเตอร์ด้านซ้ายเท่ากับ 0.05 m/s และความเร็วมอเตอร์ด้านขวาเท่ากับ 0.2 m/s
  4. ถ้าค่าของ LeftLine และ RightLine ไม่อยู่ในเงื่อนไขทั้ง 3 ข้อให้กำหนดค่าความเร็วมอเตอร์ด้านซ้ายเท่ากับ 0.2 m/s และความเร็วมอเตอร์ด้านขวาเท่ากับ 0.05 m/s

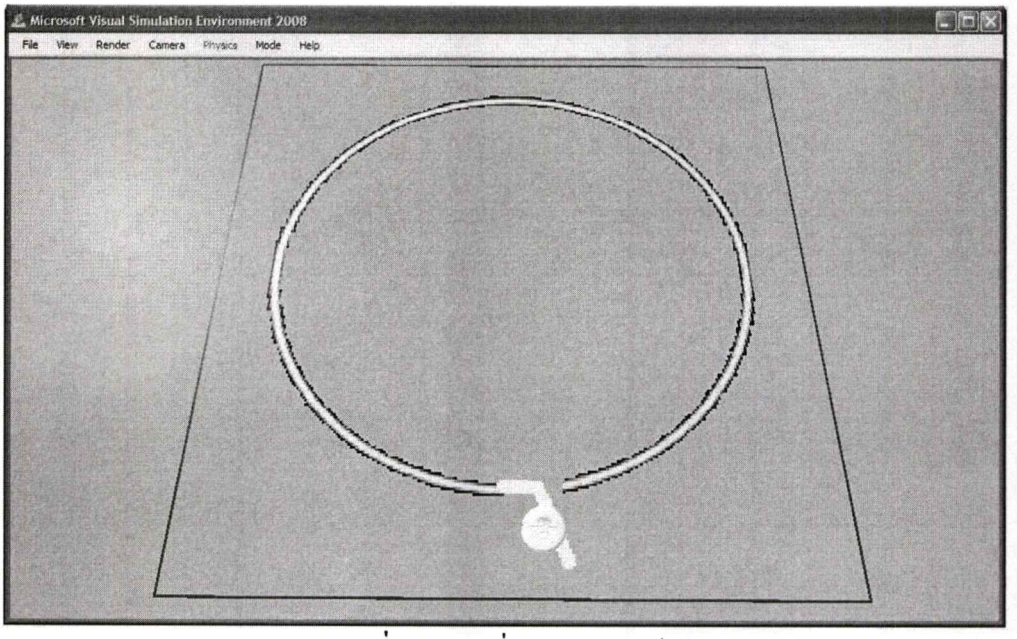


รูปที่ 3.5 โปรแกรม FollowLine จาก www.ProMRDS.com

3.1.2 ทดสอบการทำงานของโปรแกรมจาก WWW.PROMRDS.COM กับเส้นโค้งรูปแบบต่างๆ

เพื่อทดสอบว่าโปรแกรม FollowLine สามารถนำไปใช้กับการเดินตามเส้นโค้งแบบใดได้บ้าง โดยใช้แผนที่สำหรับทดสอบด้วยกันทั้งหมด 3 แผนที่คือ 1.แผนที่รูปวงกลมวนซ้าย, 2.แผนที่รูปวงกลมวนขวา และ 3.แผนที่ไม่มีรูปแบบเฉพาะ แล้วทำการจับเวลาการเดินของหุ่นยนต์จากจุดเริ่มต้นไปยังจุดสิ้นสุด ซึ่งมีผลการทดสอบเป็นดังนี้

- แผนที่รูปวงกลมวนซ้าย พบว่าโปรแกรม FollowLine สามารถเดินตามเส้นโค้งวงกลมวนซ้ายได้ แต่ไม่สามารถเดินให้อยู่ในเส้นทางได้ตลอดเวลา ซึ่งได้ผลการทดสอบออกมาดังตารางที่ 3.1

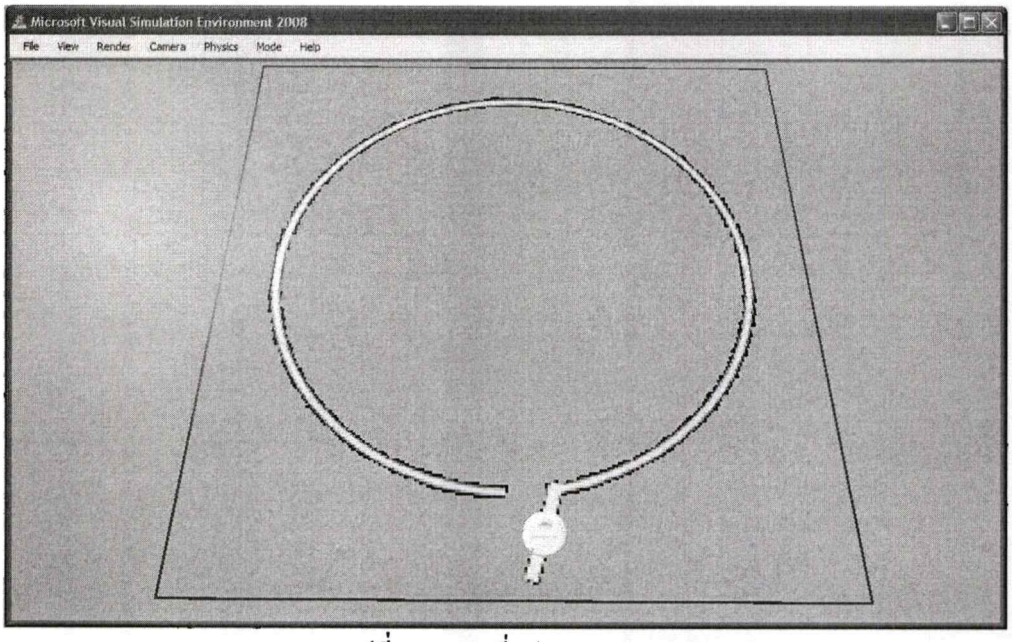


รูปที่ 3.6 แผนที่รูปวงกลมวนซ้าย

ตารางที่ 3.1 ผลการทดสอบโดยใช้แผนที่วงกลมวนซ้าย

ครั้งที่	โปรแกรม FollowLine	
	เวลา (นาที)	หมายเหตุ
1	2:35:06	หลุดจากเส้น 2 ครั้ง
2	2:35:09	หลุดจากเส้น 2 ครั้ง
3	2:35:03	หลุดจากเส้น 2 ครั้ง
4	2:36:04	หลุดจากเส้น 2 ครั้ง
5	2:35:08	หลุดจากเส้น 2 ครั้ง
เฉลี่ย	2:35:18	

- แผนที่รูปวงกลมวนขวา พบว่าโปรแกรม FollowLine ไม่สามารถเดินตามเส้นโค้งวงกลมวนขวาได้ ซึ่งได้ผลการทดสอบออกมาดังตารางที่ 3.2

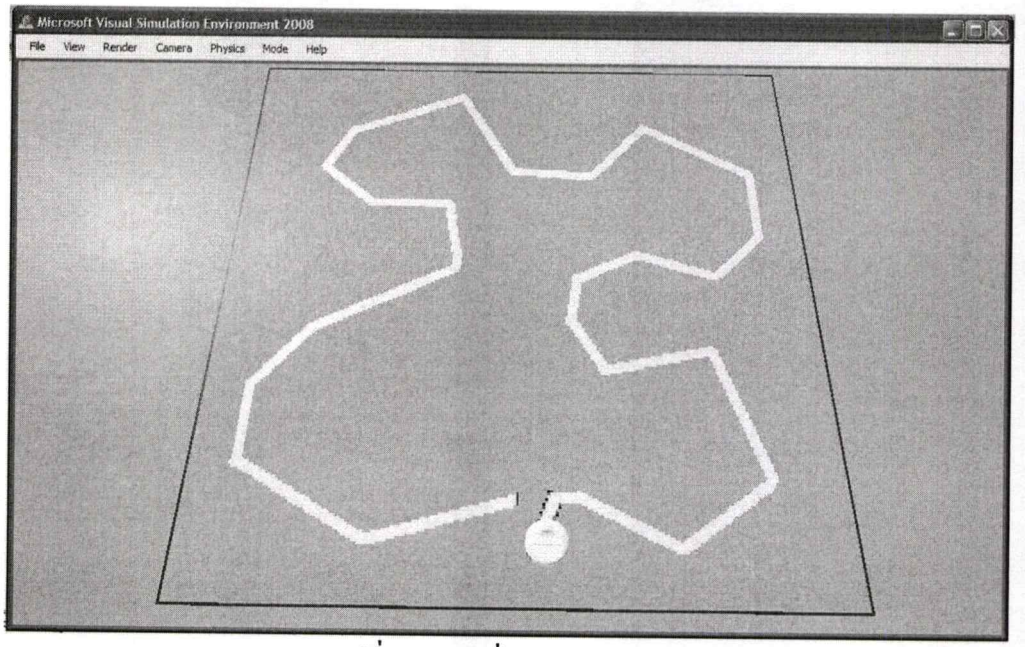


รูปที่ 3.7 แผนที่รูปวงกลมวนขวา

ตารางที่ 3.2 ผลการทดสอบโดยใช้แผนที่วงกลมวนขวา

ครั้งที่	โปรแกรม FollowLine	
	เวลา (นาที)	หมายเหตุ
1	-	เดินออกนอกเส้นทาง
2	-	เดินออกนอกเส้นทาง
3	-	เดินออกนอกเส้นทาง
4	-	เดินออกนอกเส้นทาง
5	-	เดินออกนอกเส้นทาง
เฉลี่ย	-	

- แผนที่ไม่มีรูปแบบเฉพาะ พบว่าโปรแกรม FollowLine สามารถเดินตามเส้นโค้งไม่มีรูปแบบเฉพาะได้ แต่ไม่สามารถเดินให้อยู่ในเส้นทางได้ตลอดเวลา ซึ่งได้ผลการทดสอบออกมาดังตารางที่ 3.3



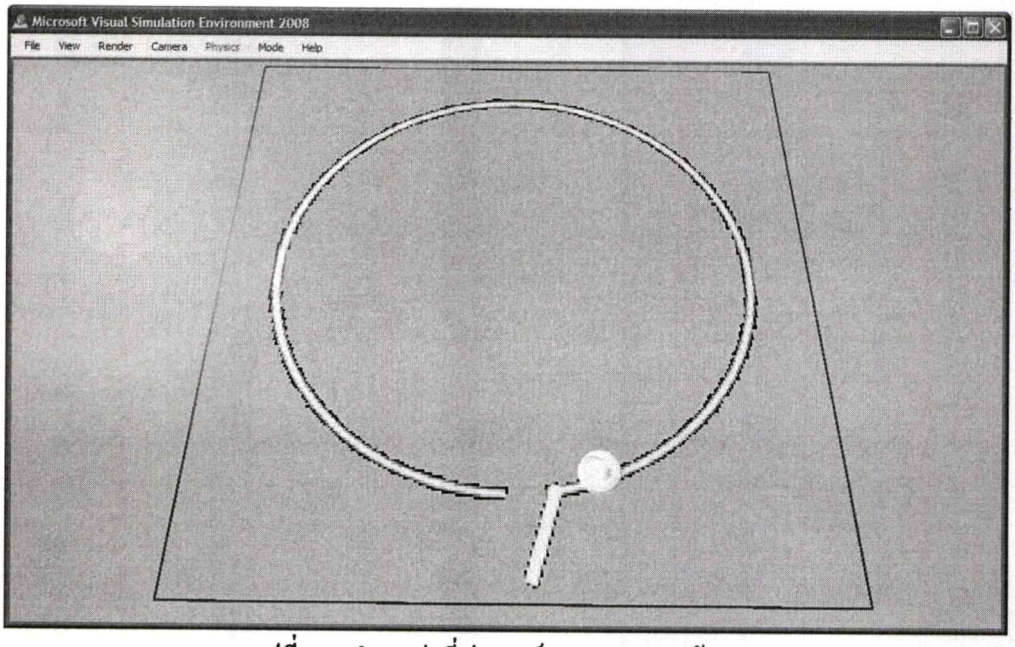
รูปที่ 3.8 แผนที่ไม่มีรูปแบบเฉพาะ

ตารางที่ 3.3 ผลการทดสอบโดยใช้แผนที่ไม่มีรูปแบบเฉพาะ

ครั้งที่	โปรแกรม FollowLine	
	เวลา (นาที)	หมายเหตุ
1	3:19:01	หลุดจากเส้น 3 ครั้ง
2	3:05:02	หลุดจากเส้น 2 ครั้ง
3	3:21:04	หลุดจากเส้น 2 ครั้ง
4	3:05:08	หลุดจากเส้น 1 ครั้ง
5	3:08:03	หลุดจากเส้น 1 ครั้ง
เฉลี่ย	3:11:40	

สรุปผลการศึกษาการทำงานของโปรแกรม FollowLine พบว่าสามารถเดินตามแผนที่ 1 และ 3 ได้ แต่การเดินนั้นยังพบว่า มีการหลุดออกนอกเส้นทาง ซึ่งหุ่นยนต์ก็สามารถเดินวนกลับมาสู่เส้นทาง แต่โปรแกรมนี้ไม่สามารถเดินตามแผนที่ 2 ซึ่งมีลักษณะเป็นวงกลมวนขวาได้ เนื่องจากหลังจากช่วงเริ่มต้นที่เป็นเส้นตรงนั้น หุ่นยนต์เดินเป็นเส้นตรงไปเรื่อยๆ เพราะ โฟโตเซ็นเซอร์ (Foto Sensor) ด้านขวานั้นตรวจพบเส้นสีขวาอยู่ แต่โฟโตเซ็นเซอร์ (Foto Sensor) ด้านซ้ายไม่พบเส้นสีขวา จนกระทั่งหุ่นยนต์เดินหลุดออกจากเส้นทาง ก็จะทำการเลี้ยวขวา เพื่อหันกลับมายัง

เส้นทาง เมื่อหุ่นยนต์ทำการหันกลับมาจนพบเส้นสีขวาแล้ว แต่หุ่นยนต์อยู่ในตำแหน่งขนานกับเส้นทาง ซึ่งตำแหน่งนี้เองโฟโตเซ็นเซอร์ (Foto Sensor) ด้านขวาและซ้ายจะตรวจไม่พบเส้นสีขวา และสั่งให้หุ่นยนต์เดินเลี้ยวขวา จนกระทั่งโฟโตเซ็นเซอร์ (Foto Sensor) ด้านซ้ายก็ตรวจพบเส้นสีขวา แต่โฟโตเซ็นเซอร์ (Foto Sensor) ด้านขวาไม่พบเส้นสีขวา จึงสั่งให้หุ่นยนต์เดินเลี้ยวซ้าย แต่เนื่องจากองศาการเลี้ยวนั้นกว้างไป ทำให้หุ่นยนต์ไม่สามารถเลี้ยวซ้ายกลับเข้าเส้นทางได้ก่อนที่โฟโตเซ็นเซอร์ (Foto Sensor) ด้านซ้ายจะหลุดออกจากเส้นสีขวา ทำให้หุ่นยนต์กลับมาเลี้ยวขวาอีกครั้งหนึ่ง ซึ่งหุ่นยนต์ได้หลุดออกนอกเส้นทางเรียบร้อยแล้ว และก็จะทำการเลี้ยวขวาไปเรื่อยๆ เนื่องจากไม่พบเส้นทางเดิน



รูปที่ 3.9 ตำแหน่งที่หุ่นยนต์หลุดออกจากเส้นทาง

3.2 แนวคิดในการพัฒนาโปรแกรม FollowLine ด้วยฟัซซี่ลอจิกคอนโทล

หลังจากที่ได้ทำการศึกษาโปรแกรม FollowLine พบว่าโปรแกรม FollowLine ยังคงมีข้อผิดพลาดในการเดินตามเส้นโค้งในบ้างรูปแบบอยู่ เพราะฉะนั้นในโครงงานนี้จึงได้นำฟัซซี่ลอจิกคอนโทล เข้ามาประยุกต์ใช้กับโปรแกรม FollowLine เพื่อแก้ไขข้อผิดพลาดบ้างอย่างของโปรแกรม FollowLine โดยมีขั้นตอนดังนี้

3.2.1 กำหนดฟัซซี่เซ็ทของอินพุทและเอาท์พุท

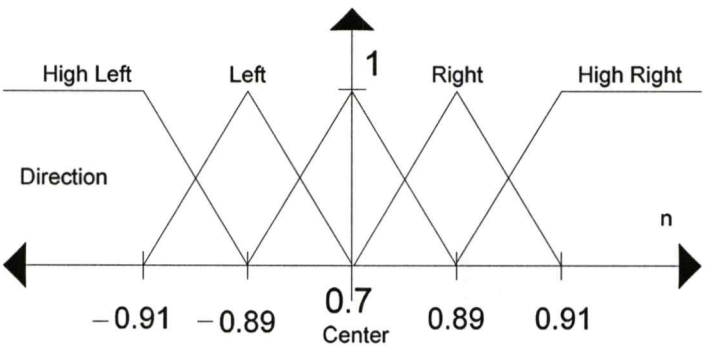
ในที่นี้จะใช้ฟังก์ชั่นความเป็นสมาชิกแบบสามเหลี่ยม โดยแบ่งฟัซซี่เซ็ทในส่วนของทิศทางดังนี้

- ไปทางซ้ายมาก:- High left, HL
- ไปทางซ้าย:- Left, L
- ตรงกลาง:- Center, C
- ไปทางขวา:- Right, R
- ไปทางขวามาก:- High right, HR

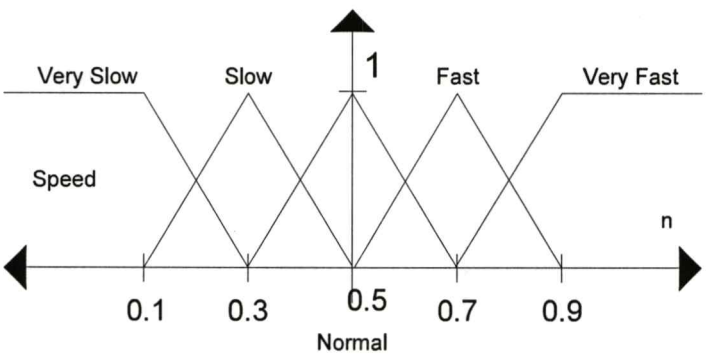
และในส่วนของความเร็วของหุ่นยนต์และมอเตอร์เป็นดังนี้

- ช้ามาก:- Very Slow, VS
- ช้าน้อย:- Slow, S
- ปานกลาง:- Normal, N
- เร็วน้อย:- Fast, F
- เร็วมาก:- Very Fast, VF

กำหนดฟัซซี่เซ็ทของอินพุทซึ่งเป็นตัวแปรจากกระบวนการในที่นี้คือ ทิศทางและความเร็วของหุ่นยนต์



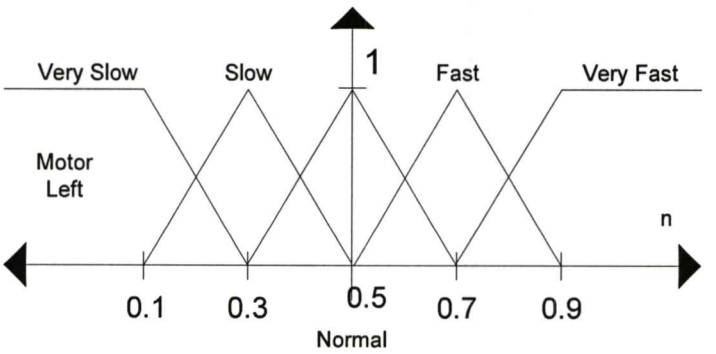
(a)



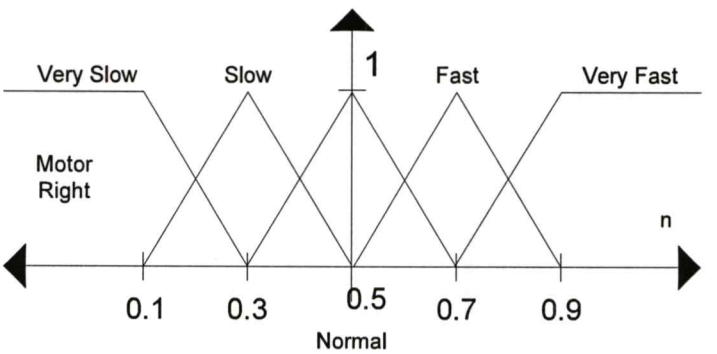
(b)

รูปที่ 3.10 (a) ฟัซซี่เซ็ททิศทางของหุ่นยนต์และ (b) ฟัซซี่เซ็ทความเร็วของหุ่นยนต์

กำหนดฟัซซี่เซ็ทของเอาท์พุทความเร็วของมอเตอร์ทั้งซ้ายและขวาที่เป็นตัวควบคุมทิศทางและความเร็วของหุ่นยนต์



(a)



(b)

รูปที่ 3.11 (a) ฟัซซี่เซ็ทของความเร็วมอเตอร์ด้านซ้าย (b) ฟัซซี่เซ็ทของความเร็วมอเตอร์ด้านขวา

3.2.2 สร้างกฎเงื่อนไขพื้นฐาน (RULE-BASE)

มีรูปแบบ ถ้า (if) เหตุพื้นฐาน (premise) แล้ว (then) ผลลัพธ์สืบเนื่อง (Consequence) สามารถเขียนกฎเงื่อนไขตามตัวอย่าง ได้แก่

- ถ้า (if) ทิศทางเป็นซ้ายมาก (HL) และ (AND) ความเร็วของหุ่นยนต์เป็นปานกลาง (N) แล้ว (then) ความเร็วมอเตอร์ซ้ายจะเป็นปานกลาง (N) และ (AND) ความเร็วมอเตอร์ขวาจะเป็นช้ามาก (VL)
- ถ้า (if) ทิศทางเป็นตรงกลาง (C) และ (AND) ความเร็วของหุ่นยนต์เป็นช้า (L) แล้ว (then) ความเร็วมอเตอร์ซ้ายจะเป็นช้า (L) และ (AND) ความเร็วมอเตอร์ขวาจะเป็นช้า (L)

โดยกฎเงื่อนไขทุกข้อสามารถเขียนสรุปเป็นตารางความสัมพันธ์ที่ 1 และ 2 ซึ่งแสดงให้เห็นถึงอินพุทสองตัวที่เข้ามาว่า ถ้า อินพุทแต่ละตัว (ซึ่งในที่นี้ คือ ทิศทางและความเร็วของหุ่นยนต์) มีค่าสมาชิกเป็นของฟัซซี่เซ็ทใด แล้ว จะได้เอาท์พุทของมอเตอร์ซ้ายและขวา (คือ ความเร็วของมอเตอร์) ออกมามีค่าเป็นสมาชิกของฟัซซี่เซ็ทใด

ตารางที่ 3.4 กฎเงื่อนไขพื้นฐานของล้อขวา

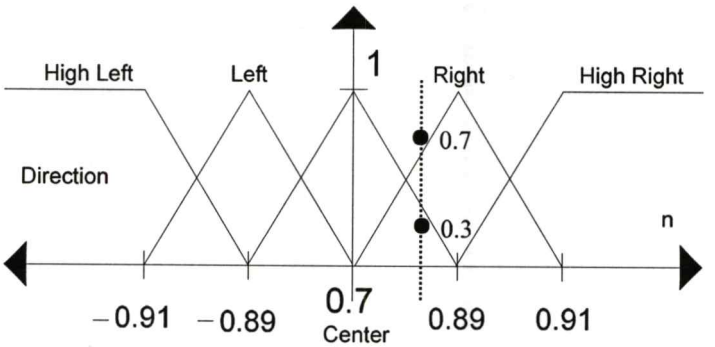
ความเร็วของล้อขวา		ทิศทาง				
		HL	L	C	R	HR
ความเร็วของหุ่นยนต์	VS	VS	VS	VS	VS	VS
	S	S	S	S	VS	VS
	N	N	N	N	S	VS
	F	F	F	F	N	S
	VF	VF	VF	VF	F	N

ตารางที่ 3.5 กฎเงื่อนไขพื้นฐานของล้อซ้าย

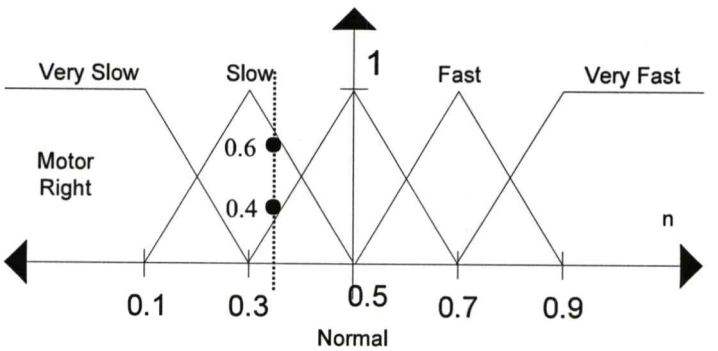
ความเร็วของล้อซ้าย		ทิศทาง				
		HL	L	C	R	HR
ความเร็วของหุ่นยนต์	VS	VS	VS	VS	VS	VS
	S	VS	VS	S	S	S
	N	VS	S	N	N	N
	F	S	N	F	F	F
	VF	N	F	VF	VF	VF

3.2.2.1 พิจารณาเมื่อมีอินพุทเข้ามายังตัวควบคุม (Fuzzicaification)

อินพุทซึ่งเป็นค่าต่างๆ นั้น จะถูกทำให้เป็นค่าทางฟัซซี่ โดยพิจารณาที่ค่าความเป็นสมาชิกในฟัซซี่เซ็ทของอินพุทนั้น (นำอินพุทที่เป็นค่าตัวเลขมาทำการ Fuzzicification)
ตัวอย่างเช่น ถ้าค่ามุมและความเร็วเชิงมุมอยู่ที่ตำแหน่งตามรูปที่ 3.12 และ 3.13 ตามลำดับ



รูปที่ 3.12 ตำแหน่งของทิศทางหุ่นยนต์



รูปที่ 3.13 ตำแหน่งของความเร็วของหุ่นยนต์

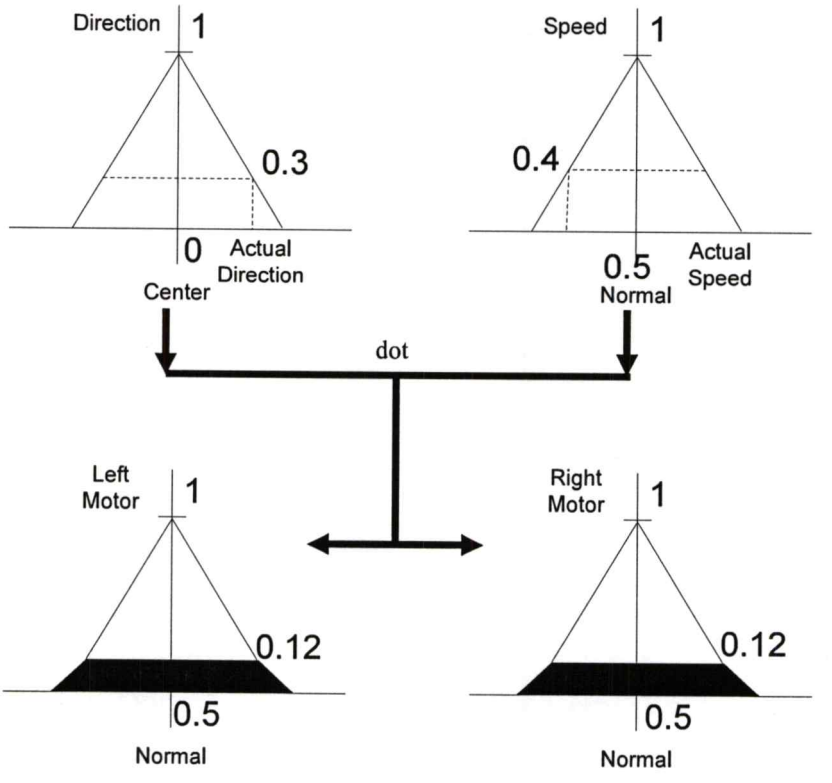
3.2.3 กลไกการวิเคราะห์ (INFERENCE MECHANISM) เพื่อหาค่าเอาท์พุท

จากค่าอินพุทตัวอย่างข้างต้น จะเห็นได้ว่าอินพุทแต่ละตัวนั้นจะตกไปอยู่ในช่วงของฟัซซี่เซ็ท 2 ช่วงดังนั้นอินพุทที่มี 2 ตัวจึงต้องนำกฎพื้นฐาน (Rule-Base) มาพิจารณาทั้งหมด 4 กรณีด้วยกัน

- พิจารณาเลือกใช้กฎ:- ถ้า (if) ทิศทางเป็นตรงกลาง (Center) และ (AND) ความเร็วของหุ่นยนต์เป็นปานกลาง (N) แล้ว (then) ความเร็วมอเตอร์ซ้ายจะเป็นปานกลาง (N) และ (AND) ความเร็วมอเตอร์ขวาจะเป็นปานกลาง (N)

ในตัวอย่างนี้ จะทำการพิจารณาเฉพาะเซ็ทของช่วงที่เป็นตรงกลางก่อน จากรูปที่ 3.12 จะเห็นได้ว่าระดับความเป็นสมาชิกของเซ็ทที่มีค่าเป็นตรงกลางของทิศทางของหุ่นยนต์ จากตัวอย่างนี้จะอยู่ที่ระดับ 0.3 และจากรูปที่ 3.13 จะเห็นได้ว่าระดับความเป็นสมาชิกของเซ็ทที่มีค่าปานกลางของความเร็วของหุ่นยนต์จะมีค่าเป็น 0.4 ตามกฎที่เลือกมานั้น กลไกการวิเคราะห์ในที่นี้จะทำการนำค่าระดับความเป็นสมาชิกทั้ง 2 นี้มาทำการคูณกัน (dot) โดยพิจารณาจากการหาค่าต่ำสุดของ 0.3 และ 0.4 คือ 0.3*0.4 = 0.12

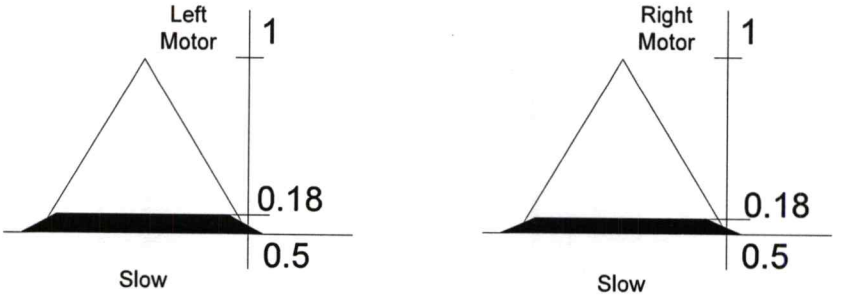
แล้วนำค่าที่ระดับนี้มาเป็นค่าของฟัซซี่เซ็ทของความเร็วของมอเตอร์ทั้งซ้ายและขวาในช่วงเซ็ทค่าที่เป็นปานกลางตามกฎพื้นฐาน (Rule-Base) ข้างต้น ดังรูปที่ 3.14 ข้างล่างนี้



รูปที่ 3.14 หาค่าระดับความเป็นสมาชิกของฟัซซี่เซ็ทของมอเตอร์ทั้ง 2

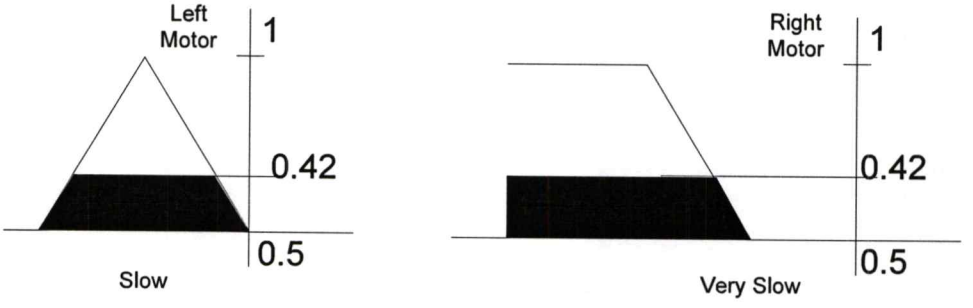
จากรูปที่ 3.12 และ 3.13 จะเห็นได้ว่าทิศทางและความเร็วของหุ่นยนต์ที่จุดนี้จะมีระดับความเป็นสมาชิกอยู่ในเซ็ทช่วงอื่นด้วย ซึ่งจะต้องทำการพิจารณาให้ครบทุกกรณีของกฎพื้นฐาน (Rule-Base) ที่ค่าอินพุทของกระบวนการสองตัวนี้เข้าเงื่อนไข โดยทำเช่นเดียวกับกรณีแรกดังต่อไปนี้

- พิจารณาเลือกใช้กฎ:- ถ้า (if) ทิศทางเป็นตรงกลาง (Center) และ (AND) ความเร็วของหุ่นยนต์เป็นช้า (S) แล้ว (then) ความเร็วมอเตอร์ซ้ายจะเป็นช้า (S) และ (AND) ความเร็วมอเตอร์ขวาจะเป็นช้า (S)



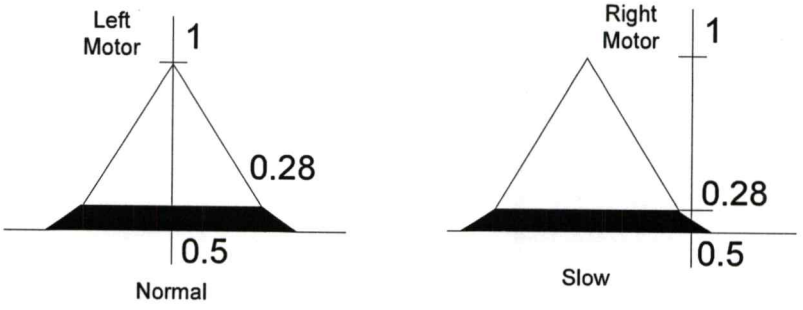
รูปที่ 3.15 ระดับความความเป็นสมาชิกของฟัซซี่เซ็ทของความเร็วมอเตอร์ทั้ง 2 ข้างในช่วงที่เป็นค่าช้า (Slow)

- พิจารณาเลือกใช้กฎ:- ถ้า (if) ทิศทางเป็นขวา (R) และ (AND) ความเร็วของหุ่นยนต์เป็นช้า (S) แล้ว (then) ความเร็วมอเตอร์ซ้ายจะเป็นช้า (S) และ (AND) ความเร็วมอเตอร์ขวาจะเป็นช้ามาก (VS)



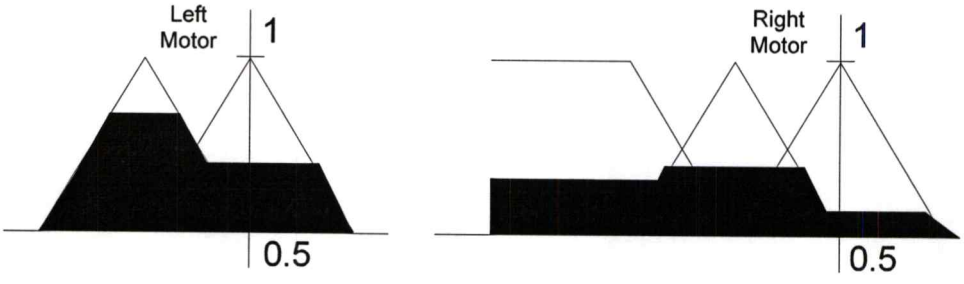
รูปที่ 3.16 ระดับความความเป็นสมาชิกของฟัซซี่เซ็ทของความเร็วมอเตอร์ทั้ง 2 ข้างในช่วงที่เป็นค่าช้ามาก (Very Slow) และช้า (Slow)

- พิจารณาเลือกใช้กฎ:- ถ้า (if) ทิศทางเป็นขวา (R) และ (AND) ความเร็วของหุ่นยนต์เป็นปานกลาง (N) แล้ว (then) ความเร็วมอเตอร์ซ้ายจะเป็นปานกลาง (N) และ (AND) ความเร็วมอเตอร์ขวาจะเป็นช้า (S)



รูปที่ 3.17 ระดับความความเป็นสมาชิกของฟัซซี่เซ็ทของความเร็วมอเตอร์ทั้ง 2 ข้างในช่วงที่เป็นค่าช้า (Slow) และปานกลาง (Normal)

จากนั้นนำฟัซซี่เซ็ทผลลัพธ์จากทุกกรณีมารวมกันโดยใช้การ ก็จะได้ฟัซซี่เซ็ทผลลัพธ์รวมทุกกรณีที่เป็นความเร็วของมอเตอร์ซ้ายและขวา ตามรูปที่ 3.18



รูปที่ 3.18 ผลลัพธ์ฟัซซี่เซ็ทความเร็วของมอเตอร์ซ้ายและขวาจากทุกกรณีรวมกันโดยการ OR

วิธีการตามขั้นตอนที่ได้แสดงมาข้างต้น เป็นขั้นตอนในการวิเคราะห์หาค่าระดับความเป็นสมาชิกที่เป็นไปได้ทั้งหมดของฟัซซี่เซ็ทของผลลัพธ์รวม (Inference Step) โดยวิธีการนี้มีลักษณะเป็นรูปแบบ Dot of Product ที่ใช้การ Operation หาค่าโดยทำการคูณกัน วิธีการตามที่ได้กล่าวมาข้างต้นเป็นวิธีการหนึ่งที่กันโดยทั่วไปของขั้นตอน Inference Step

3.2.4 หาค่าเอาท์พุท (นำค่าเอาท์พุทที่เป็นค่าทางฟัซซี่มาทำการ DEFUZZIFICATION)

ตามรูปที่ 3.19 เป็นการหาค่าความเร็วของมอเตอร์ซ้ายและขวาในฟัซซี่เซ็ทผลลัพธ์ของความเร็วของหุ่นยนต์ในทุกกรณีรวมกัน โดยหาค่าที่จุดศูนย์กลางเฉลี่ย (Center-Average) ตามขั้นตอนดังนี้

- Right Motor

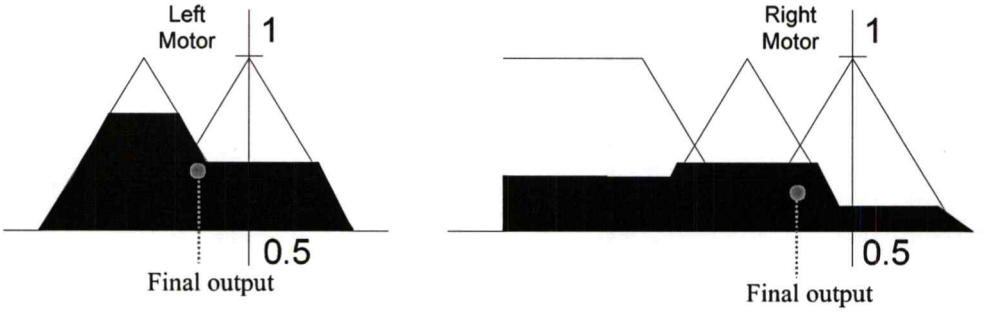
$$\text{Output} = \frac{(VS)(0.42) + (S)(0.4) + (N)(0.12)}{0.42 + 0.4 + 0.12}$$

$$= \frac{(0.1)(0.42) + (0.3)(0.4) + (0.5)(0.12)}{0.42 + 0.4 + 0.12} = 0.236$$

- Left Motor

$$\text{Output} = \frac{(S)(0.6) + (N)(0.4)}{0.6 + 0.4}$$

$$= \frac{(0.3)(0.6) + (0.5)(0.4)}{0.6 + 0.4} = 0.38$$



รูปที่ 3.19 ค่าผลลัพธ์ของความเร็วมอเตอร์ด้านซ้ายและขวา

3.3 สภาพแวดล้อม ส่วนประกอบและหลักการทำงานของหุ่นยนต์ในการเดินตามเส้นโค้ง

ในส่วนการออกแบบเพื่อให้หุ่นยนต์สามารถเดินตามเส้นโค้งได้นั้น จะมีการนำอุปกรณ์ต่างๆ เข้ามาใช้งานร่วมกันตามแต่ละประเภทของอุปกรณ์ ซึ่งจะต้องทำความเข้าใจว่าแต่ละอุปกรณ์ที่นำมาใช้งานนั้นมีกลไกการทำงานเป็นอย่างไร เพื่อที่จะสามารถทำการออกแบบการทำงานของระบบโดยรวม เพื่อให้หุ่นยนต์สามารถทำงานได้กับสภาพแวดล้อมที่กำหนดไว้

3.3.1 สภาพแวดล้อมและอุปกรณ์ที่นำมาใช้งาน

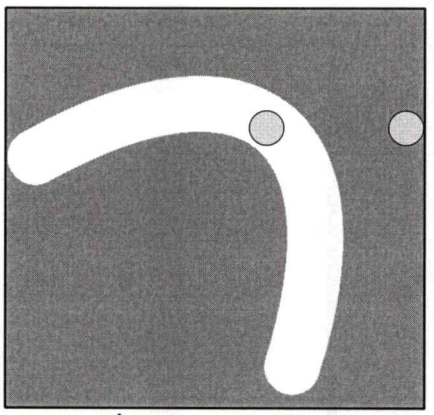
ในสภาพแวดล้อมที่จะให้หุ่นยนต์ทำงานนั้นจะเป็นพื้นที่ราบ โดยมีเส้นสีดำและพื้นสีขาว เพื่อให้หุ่นยนต์เดินตามเส้นนี้ไปยังจุดต่างๆ ได้ ซึ่งในการเดินตามเส้นนี้ต้องใช้อุปกรณ์การตรวจจับเส้นสีขาว คือ โฟโตเซ็นเซอร์ (Foto Sensor) เพื่อเป็นหนึ่งในอินพุทของระบบและอินพุทอีกหนึ่งชุด คือ มอเตอร์ที่ใช้ในการเคลื่อนที่และกำหนดทิศทางของหุ่นยนต์

3.3.1.1 อินพุทของระบบ

อินพุทของระบบตัวที่หนึ่งจะใช้โฟโตเซ็นเซอร์ (Foto Sensor) จำนวน 2 ตัวติดตั้งไว้ที่ด้านหน้าของหุ่นยนต์ เพื่อทำหน้าที่ตรวจจับว่าเส้นโค้งกับตัวหุ่นยนต์นั้นมีทิศทางเป็นอย่างไร โดย

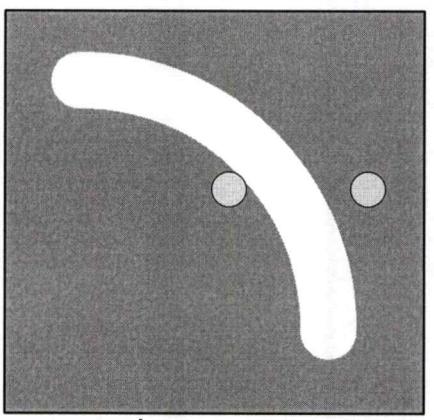
ใช้หลักการความแตกต่างของสีทั้ง 2 โดยเมื่อโฟโตเซ็นเซอร์ (Foto Sensor) ตรวจพบสีเทาจะให้เอาท์พุทออกมาเป็นค่า 0.628 , เมื่อตรวจพบขอบสีขาวจะให้เอาท์พุทมาเป็นค่า 0.88 และเมื่อตรวจพบสีขาวจะให้เอาท์พุทมาเป็นค่า 0.919 โดยทำการอ่านค่าจากโฟโตเซ็นเซอร์ (Foto Sensor) ทั้ง 2 ข้างก็จะสามารถรู้ได้ว่าเส้นทางข้างหน้าเป็นอย่างไร โดยสามารถระบุได้ว่ารูปแบบเส้นทางข้างหน้าได้ 5 รูปแบบดังนี้

- เซ็นเซอร์ซ้ายอ่านค่าได้ 0.919 เซ็นเซอร์ขวาอ่านค่าได้ 0.628 แสดงว่าเส้นโค้งมีลักษณะโค้งไปทางซ้ายมาก



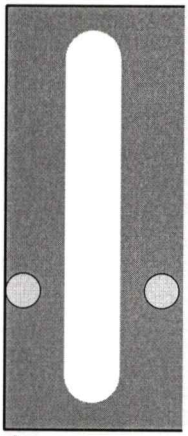
รูปที่ 3.20 เส้นโค้งซ้ายมาก

- เซ็นเซอร์ซ้ายอ่านค่าได้ 0.88 เซ็นเซอร์ขวาอ่านค่าได้ 0.628 แสดงว่าเส้นโค้งมีลักษณะโค้งไปทางซ้าย



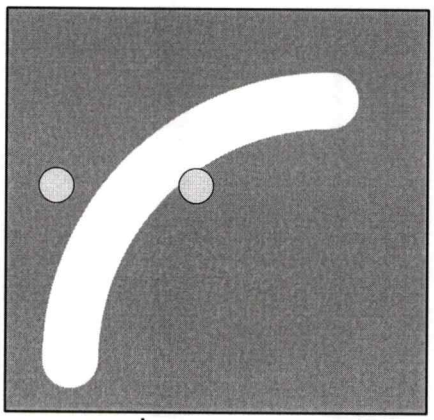
รูปที่ 3.21 เส้นโค้งซ้าย

- เซ็นเซอร์ซ้ายอ่านค่าได้ 0.628 เซ็นเซอร์ขวาอ่านค่าได้ 0.628 แสดงว่าเป็นเส้นตรง



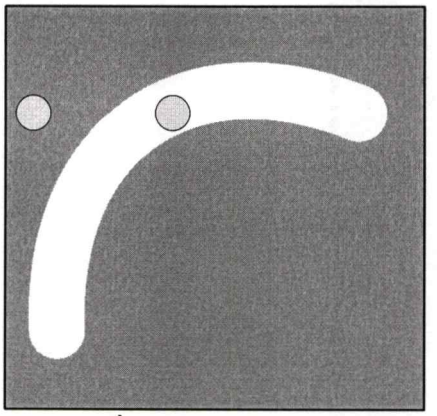
รูปที่ 3.22 เส้นตรง

- เซ็นเซอร์ซ้ายอ่านค่าได้ 0.628 เซ็นเซอร์ขวาอ่านค่าได้ 0.88 แสดงว่าเส้นโค้งมีลักษณะโค้งไปทางขวา



รูปที่ 3.23 เส้นโค้งขวา

- เซ็นเซอร์ซ้ายอ่านค่าได้ 0.628 เซ็นเซอร์ขวาอ่านค่าได้ 0.919 แสดงว่าเส้นโค้งมีลักษณะโค้งไปทางซ้ายมาก



รูปที่ 3.24 เส้นโค้งขวามาก

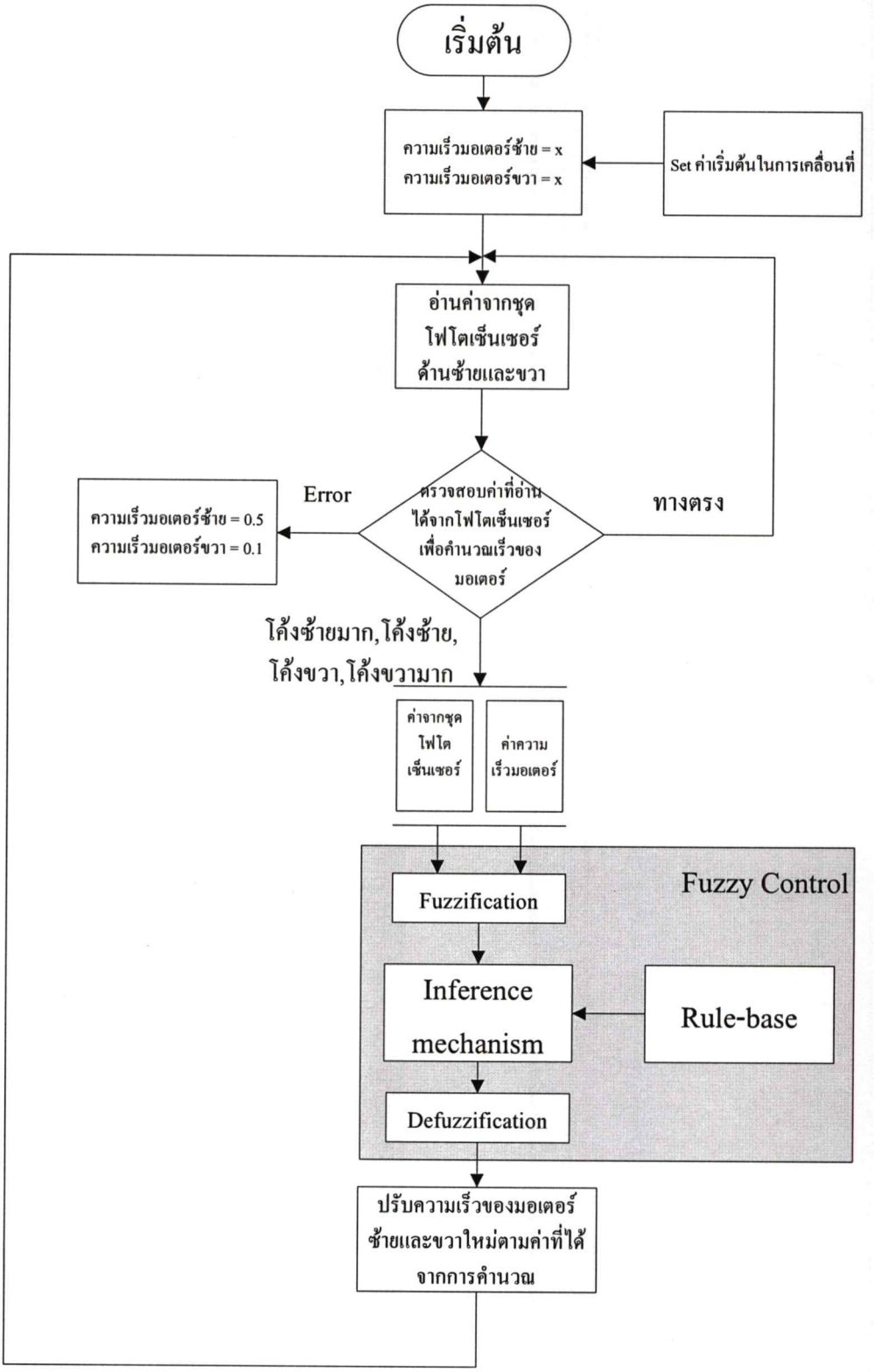
อินพุทของระบบตัวที่สองจะใช้มอเตอร์ที่ใช้สำหรับเคลื่อนที่หุ่นยนต์ไปตามเส้นโค้งโดยนำความเร็วของมอเตอร์มาเป็นอินพุทของระบบอีกตัวหนึ่ง โดยมอเตอร์แต่ละตัวจะสามารถปรับความเร็วได้ตั้งแต่ 0 – 1 เมตรต่อวินาที (m/s) ซึ่งมีระดับการเพิ่มหรือลดลงของความเร็วทีละ 0.01 ซึ่งจะนำความเร็วของหุ่นยนต์ช่วงก่อนเข้าเส้นโค้งมาทำการคำนวณร่วมกับค่าที่อ่านได้จากชุดโฟโตเซ็นเซอร์ (Foto Sensor)

3.3.1.2 เอาท์พุทของระบบ

เอาท์พุทของระบบ คือ ความเร็วของมอเตอร์ในแต่ละตัว ซึ่งจะได้ค่าออกจากการคำนวณ โดยค่าที่คำนวณได้ก็จะบอกว่ามอเตอร์ซ้ายใช้ความเร็วเท่าไหร่และขวาความเร็วเท่าไหร่ ในการเคลื่อนที่ไปตามแต่ละโค้งนั้น เมื่อเปลี่ยนเส้นโค้งใหม่ระบบก็จะทำการคำนวณค่าใหม่ เพื่อได้ค่าเอาท์พุทที่เหมาะสมของแต่ละสถานการณ์นั้น

3.3.2 หลักการทำงานของหุ่นยนต์เดินตามเส้นโค้ง

ในหัวข้อนี้จะเป็นการแสดงแผนภาพการทำงานของหุ่นยนต์ในการเดินตามเส้นโค้ง ซึ่งได้นำฟัซซี่คอนโทลมาใช้ในการควบคุมการเคลื่อนที่ของหุ่นยนต์



รูปที่ 3.25 โฟลวชาร์ตแสดงขั้นตอนการทำงานของโปรแกรม

บทที่ 4
การสร้างโปรแกรมและจำลองการทำงานของโปรแกรมสำหรับควบคุมหุ่นยนต์เดินตามเส้นโค้ง

ในบทนี้จะกล่าวถึงการสร้างโปรแกรมและจำลองการทำงานของโปรแกรมการควบคุมหุ่นยนต์เดินตามเส้นโค้ง โดยใช้ฟัซซี่ลอจิกคอนโทล แผนที่ที่กำหนดให้แล้วให้หุ่นยนต์ทำการเดินตามเส้นโค้งบนแผนที่ จากนั้นจะทำการจับเวลาว่าใช้ระยะเวลาเท่าไหร่ในการเดินจากต้นทางไปยังปลายทาง

4.1 เครื่องมือและภาษาที่ใช้ในการพัฒนาโปรแกรม

การพัฒนาโปรแกรมการควบคุมหุ่นยนต์ให้เดินตามเส้นโค้ง โดยใช้ฟัซซี่ลอจิกได้เครื่องมือและภาษาในการพัฒนาดังนี้

4.1.1 ฮาร์ดแวร์

เครื่องคอมพิวเตอร์ที่ใช้ในการพัฒนาโปรแกรมและทดสอบระบบ โดยมีคุณสมบัติดังนี้

- CPU : Intel ®Core 2 Duo E7400 2.8GHz
- Memory : 4GB
- Hard Disk : 640GB
- Graphic Card : Nvidia GeForce GTS250 Memory 512MB

4.1.2 ซอฟต์แวร์

ซอฟต์แวร์ที่ใช้ในการพัฒนาระบบ มีดังนี้

- Microsoft Robotics Developer Studio 2008
- ProMRDS_Standard_20090114.exe
- Microsoft Windows XP Professional SP3

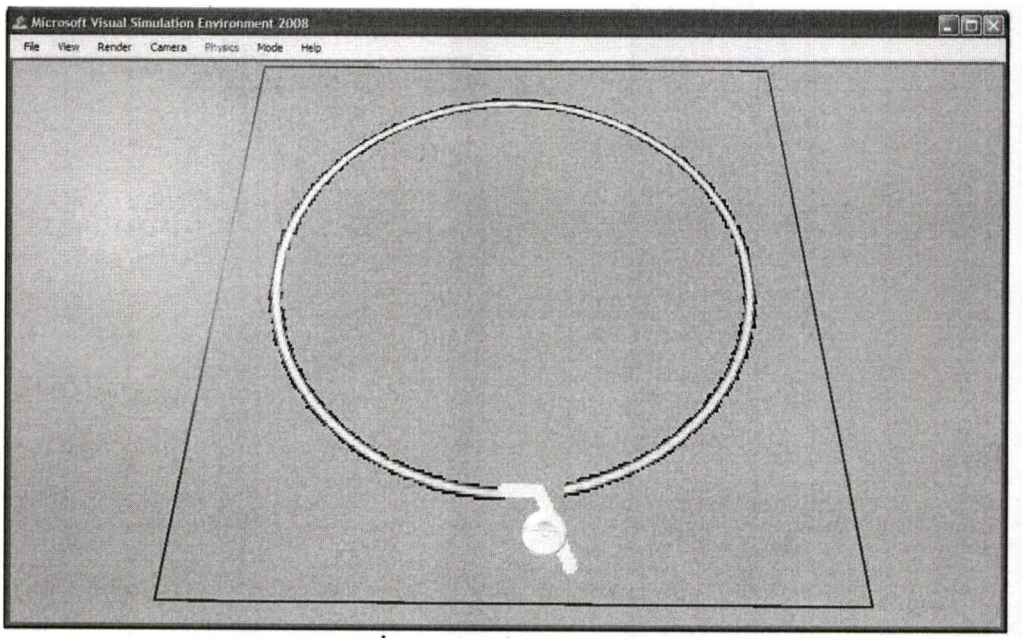
4.1.3 เครื่องมือ

เครื่องมือที่ใช้ในการพัฒนาโปรแกรม

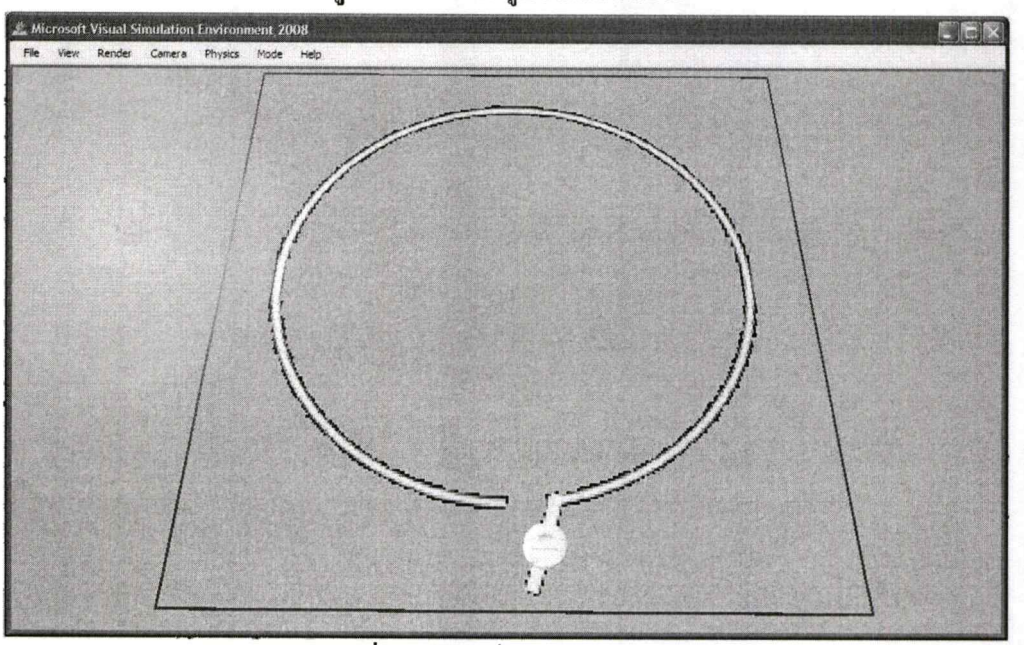
- Visual Programming Language 2008

4.2 แผนที่สำหรับทำการทดสอบการทำงาน

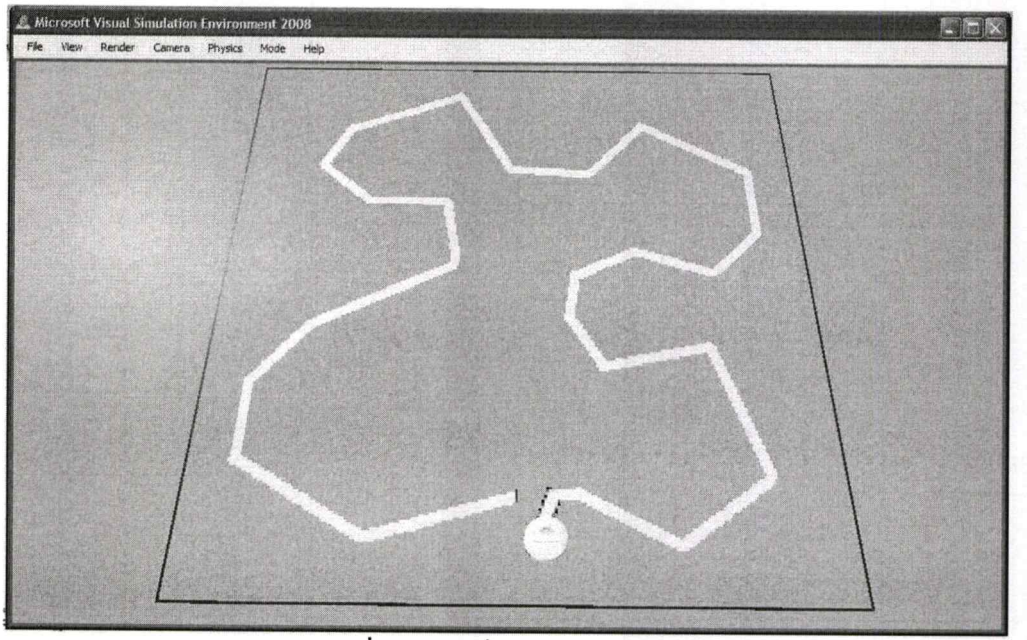
แผนที่สำหรับทำการทดสอบโปรแกรมการควบคุมหุ่นยนต์ให้เดินตามเส้นโค้ง โดยใช้ฟัซซี่ลอจิกคอนโทลที่ได้ทำการพัฒนาขึ้นมา ซึ่งจะใช้แผนที่ในการทำสอบด้วยกัน 3 แผนที่ คือ 1.แผนที่รูปวงกลมวนซ้าย, 2.แผนที่รูปวงกลมวนขวา และ 3.แผนที่ไม่มีรูปแบบเฉพาะ



รูปที่ 4.1 แผนที่รูปวงกลมวนซ้าย



รูปที่ 4.2 แผนที่รูปวงกลมวนขวา

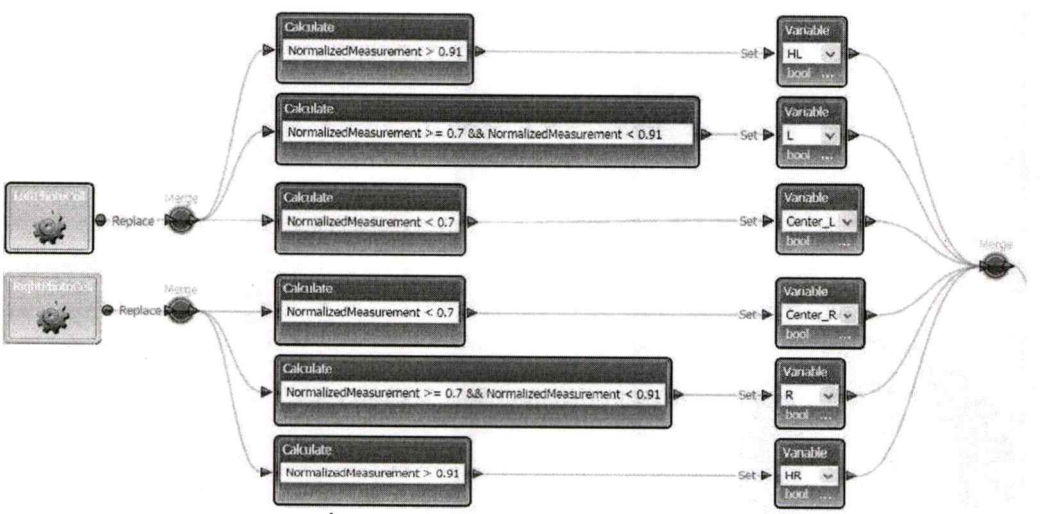


รูปที่ 4.3 แผนที่ไม่มีรูปแบบเฉพาะ

4.3 โปรแกรมที่ได้ทำการพัฒนา

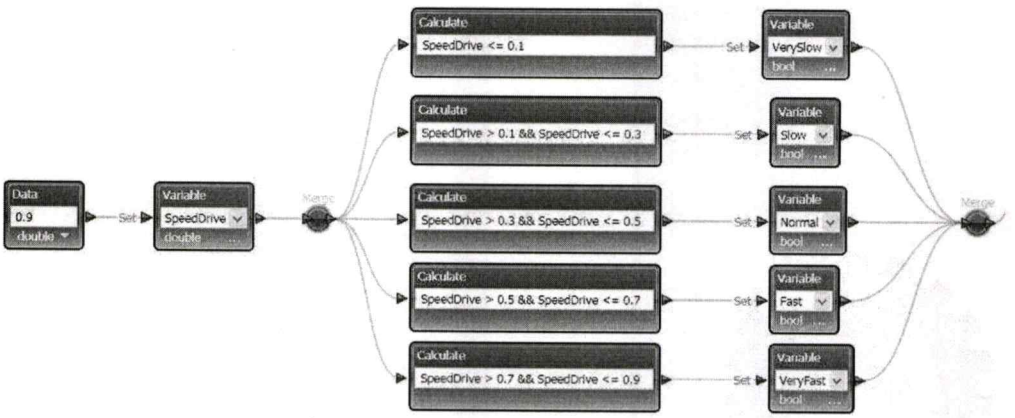
ในส่วนของโปรแกรมที่ได้ทำการพัฒนาขึ้นมาใหม่ โดยใช้หลักการของฟัซซี่ลอจิกคอนโทลนั้น ในส่วนของโปรแกรมสามารถอธิบายส่วนต่างๆ ได้ดังนี้

- โฟโต้เซ็นเซอร์ (Foto Sensor) ทำหน้าที่ตรวจสอบลักษณะเส้นทางข้างหน้าว่าเป็นทางแบบใด ซึ่งจะมีรูปแบบโปรแกรมเป็นดังรูปที่ 4.4



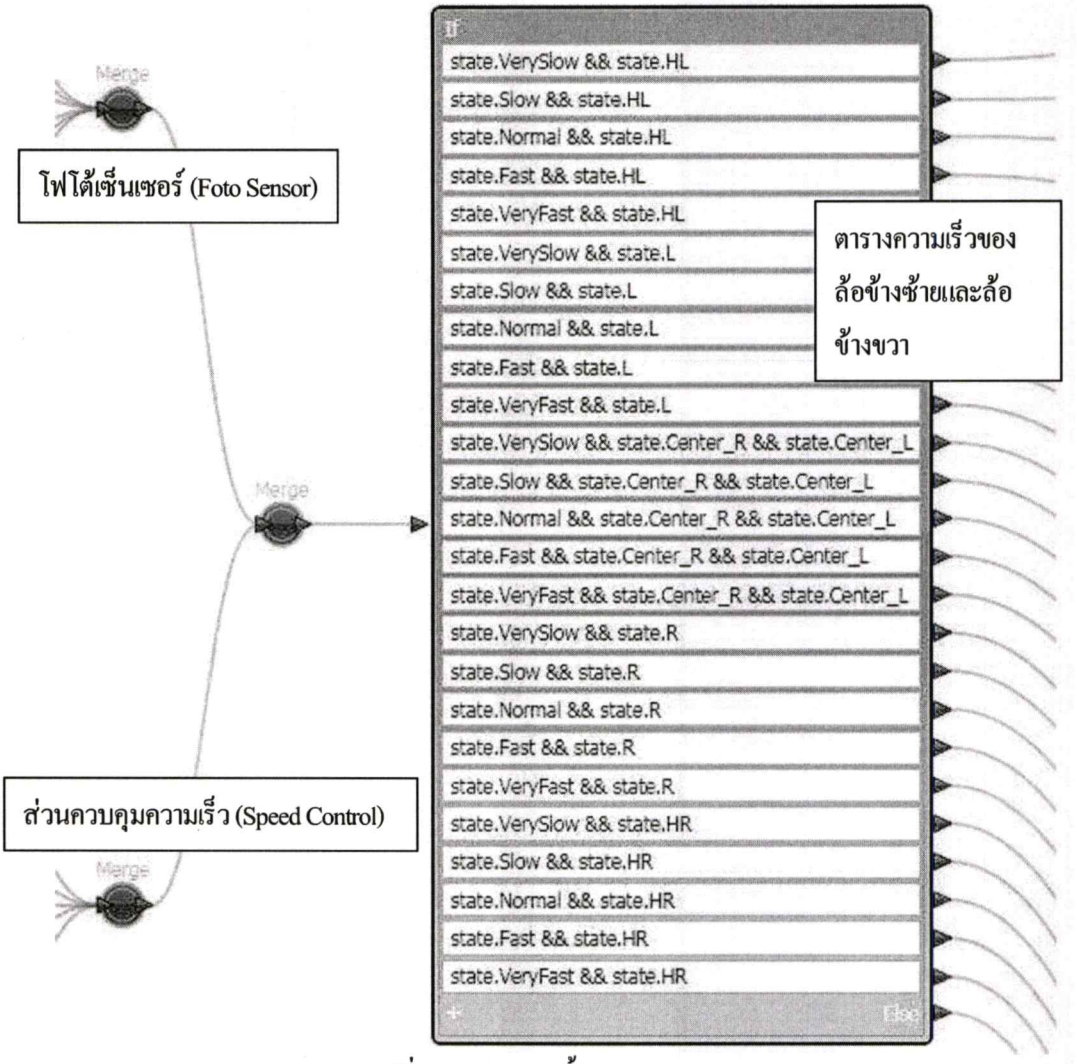
รูปที่ 4.4 ส่วนโฟโต้เซ็นเซอร์ (Foto Sensor)

- ส่วนควบคุมความเร็ว (Speed Control) ทำหน้าที่กำหนดความเร็วเบื้องต้นและควบคุมความเร็วของหุ่นยนต์ ซึ่งจะมีรูปแบบโปรแกรมเป็นดังรูปที่ 4.5



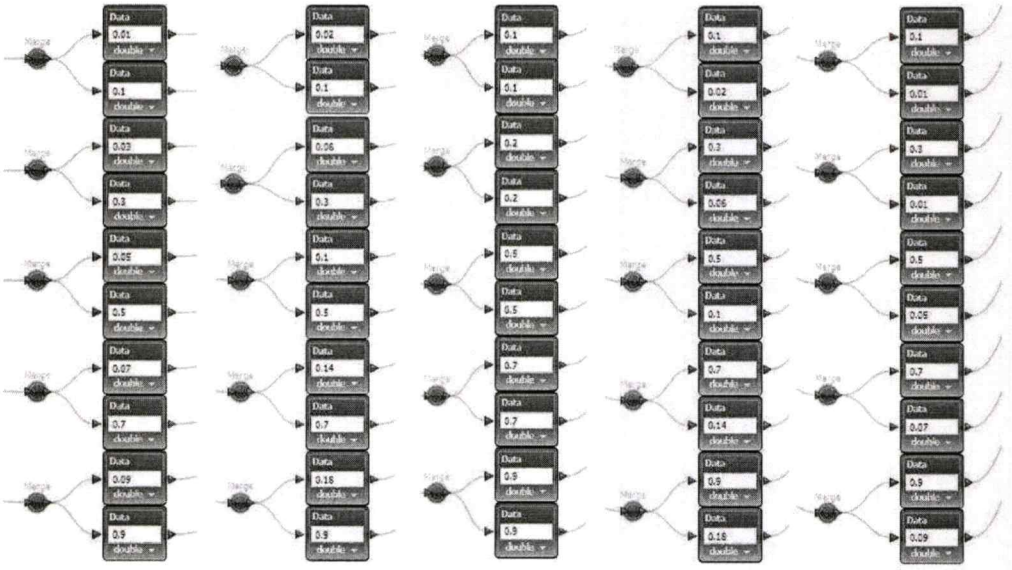
รูปที่ 4.5 ส่วนควบคุมความเร็ว (Speed Control)

- ส่วนกฎพื้นฐาน (Rule-Base) ทำหน้าที่เลือกความเร็วล้อข้างซ้ายและล้อข้างขวา โดยพิจารณาจากกฎพื้นฐาน (Rule-Base) ซึ่งจะมีรูปแบบโปรแกรมเป็นดังรูปที่ 4.6



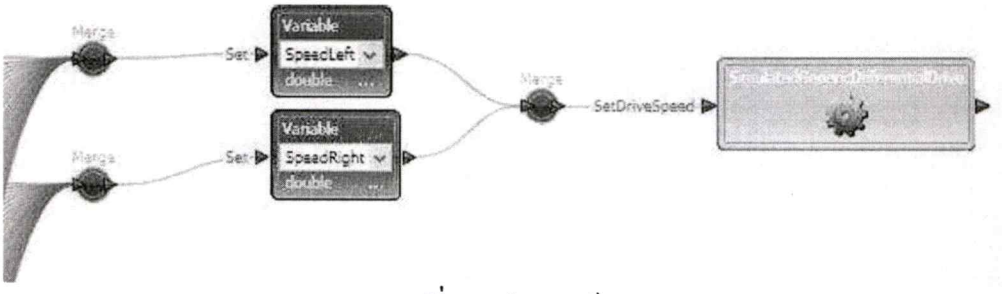
รูปที่ 4.6 ส่วนกฎพื้นฐาน (Rule-Base)

- ส่วนตารางค่าความเร็วของล้อข้างซ้ายและล้อข้างขวา ทำหน้าที่กำหนดค่าความเร็วล้อข้างซ้ายและล้อข้างขวา หลังจากที่ส่วนกฎพื้นฐาน (Rule-Base) ได้ทำการคำนวณค่าเอาท์พุทจากอินพุทที่เข้ามายังระบบ ซึ่งจะมีรูปแบบโปรแกรมเป็นดังรูปที่ 4.7



รูปที่ 4.7 ส่วนตารางค่าความเร็วของล้อข้างซ้ายและล้อข้างขวา

- ส่วนเอาท์พุท ทำหน้าที่ส่งค่าที่ได้จากการคำนวณไปยังล้อข้างซ้ายและล้อข้างขวา ซึ่งจะมีรูปแบบโปรแกรมเป็นดังรูปที่ 4.8



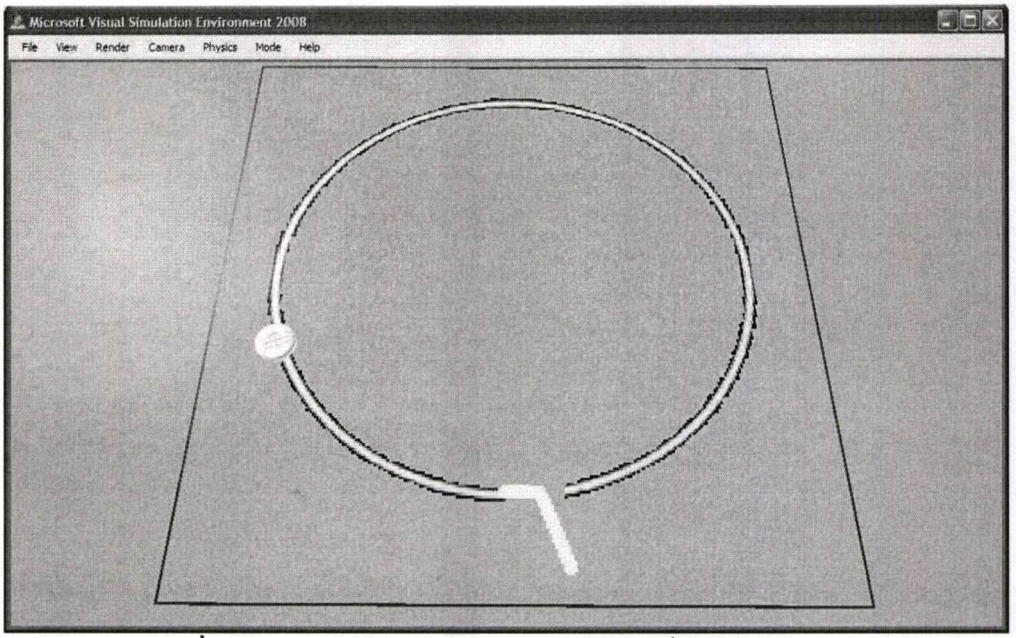
รูปที่ 4.8 ส่วนเอาท์พุท

4.4 ทดสอบสมรรถนะของโปรแกรม

ในหัวข้อจะทำการทำการทดสอบการทำงานของโปรแกรมที่ได้ทำการพัฒนามา โดยรูปแบบการทดสอบดังนี้

4.4.1 ทดสอบโดยใช้แผนที่รูปวงกลมวนซ้าย

ในการทดสอบหัวข้อนี้จะใช้แผนที่รูปวงกลมวนซ้ายดังรูปที่ 4.1 มาทำการทดสอบโดยทำการจับเวลาการเดินของหุ่นยนต์จากต้นทางไปยังปลายทาง ซึ่งจะทำการทดสอบแต่ละจำนวน 5 ครั้ง และเพิ่มการทดสอบโดยกำหนดค่าที่ส่วนควบคุมความเร็ว (Speed Control) จำนวน 2 ค่า คือ ช้ามากกับช้า ซึ่งได้ผลการทดสอบออกมาดังตารางที่ 4.1



รูปที่ 4.9 ขณะทำการทดสอบโปรแกรมด้วยแผนที่รูปวงกลมวนซ้าย

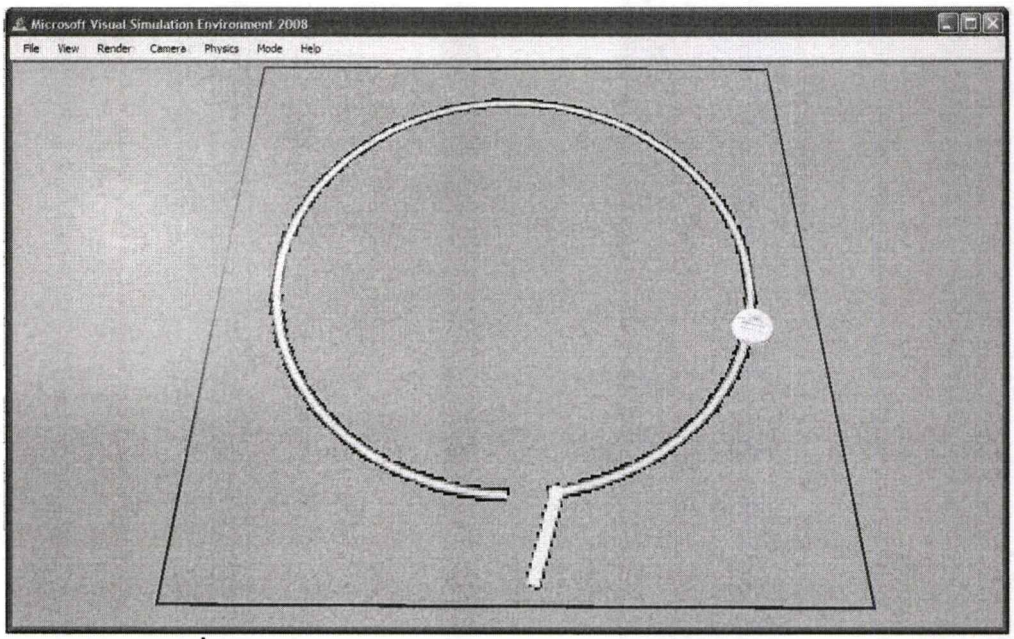
ตารางที่ 4.1 ผลการทดสอบโดยใช้แผนที่วงกลมวนซ้าย

ครั้งที่	Speed Control = ช้ามาก		Speed Control = ช้า	
	เวลา (นาที)	หมายเหตุ	เวลา (นาที)	หมายเหตุ
1	2:22:07	-	1:06:03	-
2	2:22:04	-	-	เดินออกนอกเส้นทาง
3	2:22:03	-	1:06:01	-
4	2:22:01	-	-	เดินออกนอกเส้นทาง
5	2:21:09	-	1:05:05	-
เฉลี่ย	2:21:53		1:05:43	-

จากผลการทดสอบแผนที่รูปวงกลมวนซ้าย พบว่า ถ้ากำหนด Speed Control = ช้ามาก หุ่นยนต์สามารถเดินจากต้นทางไปยังปลายทางโดยไม่ออกนอกเส้นทางทุกครั้งจากการทดสอบจำนวน 5 ครั้ง ซึ่งใช้เวลาเฉลี่ยในการเดิน คือ 2:12:53 นาที แต่ถ้ากำหนด Speed Control = ช้า หุ่นยนต์สามารถเดินจากต้นทางไปยังปลายได้เพียง 3 ครั้งจากการทดสอบทั้งหมด 5 ครั้ง ซึ่งใช้เวลาเฉลี่ยในการเดิน คือ 1:05:43 นาที

4.4.2 ทดสอบโดยใช้แผนที่รูปวงกลมวนขวา

ในการทดสอบหัวข้อนี้จะใช้แผนที่รูปวงกลมวนขวาดังรูปที่ 4.2 มาทำการทดสอบโดยทำการจับเวลาการเดินของหุ่นยนต์จากต้นทางไปยังปลายทาง ซึ่งจะทำการทดสอบแต่ละจำนวน 5 ครั้ง และเพิ่มการทดสอบโดยกำหนดค่าที่ส่วนควบคุมความเร็ว (Speed Control) จำนวน 2 ค่า คือ ช้ามากกับช้า ซึ่งได้ผลการทดสอบออกมาดังตารางที่ 4.2



รูปที่ 4.9 ขณะทำการทดสอบโปรแกรมด้วยแผนที่รูปวงกลมวนขวา

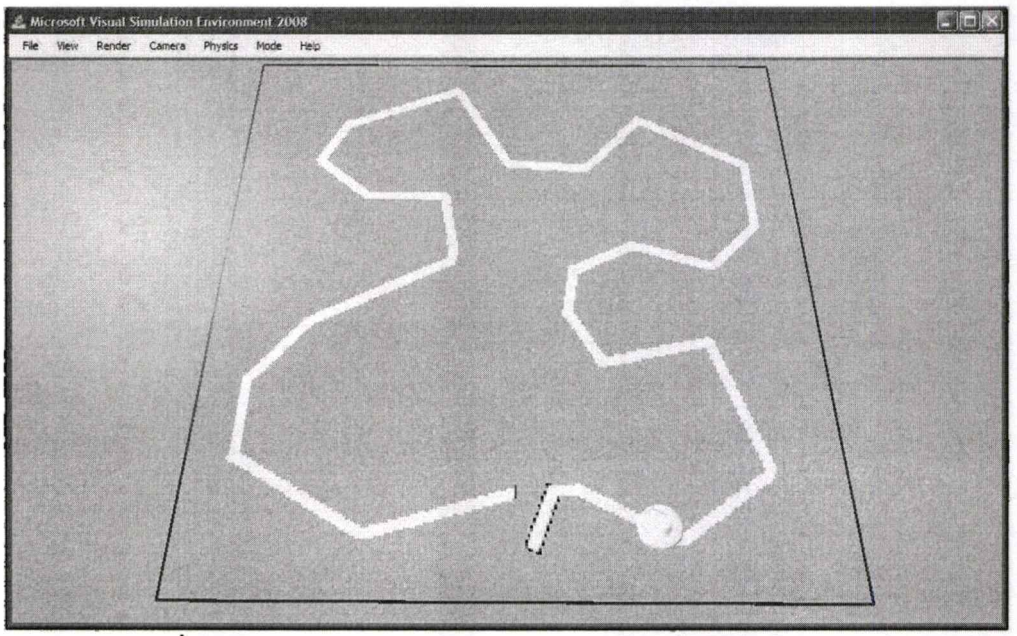
ตารางที่ 4.2 ผลการทดสอบโดยใช้แผนที่วงกลมวนขวา

ครั้งที่	Speed Control = ช้ามาก		Speed Control = ช้า	
	เวลา (นาที)	หมายเหตุ	เวลา (นาที)	หมายเหตุ
1	2:19:00	-	-	เดินออกนอกเส้นทาง
2	2:21:03	-	-	เดินออกนอกเส้นทาง
3	2:19:03	-	-	เดินออกนอกเส้นทาง
4	2:21:04	-	-	เดินออกนอกเส้นทาง
5	2:21:00	-	1:04:07	-
เฉลี่ย	2:20:14		1:04:07	-

จากผลการทดสอบแผนที่รูปวงกลมวนขวา พบว่า ถ้ากำหนด Speed Control = ช้ามาก หุ่นยนต์สามารรถเดินจากต้นทางไปยังปลายทางโดยไม่ออกนอกเส้นทางทุกครั้งจากการทดสอบจำนวน 5 ครั้ง ซึ่งใช้เวลาเฉลี่ยในการเดิน คือ 2:20:514 นาที แต่ถ้ากำหนด Speed Control = ช้า หุ่นยนต์สามารถเดินจากต้นทางไปยังปลายได้เพียง 1 ครั้งจากการทดสอบทั้งหมด 5 ครั้ง ซึ่งใช้เวลาเฉลี่ยในการเดิน คือ 1:04:07 นาที

4.4.3 ทดสอบโดยใช้แผนที่ไม่มีรูปแบบเฉพาะ

ในการทดสอบหัวข้อนี้จะใช้แผนที่ไม่มีรูปแบบเฉพาะดังรูปที่ 4.3 มาทำการทดสอบโดยทำการจับเวลาการเดินของหุ่นยนต์จากต้นทางไปยังปลายทาง ซึ่งจะทำการทดสอบแต่ละจำนวน 5 ครั้ง และเพิ่มการทดสอบโดยกำหนดค่าที่ส่วนควบคุมความเร็ว (Speed Control) จำนวน 2 ค่า คือ ช้ามากกับช้า ซึ่งได้ผลการทดสอบออกมาดังตารางที่ 4.3



รูปที่ 4.10 ขณะทำการทดสอบโปรแกรมด้วยแผนที่ไม่มีรูปแบบเฉพาะ

ตารางที่ 4.3 ผลการทดสอบโดยใช้แผนที่ไม่มีรูปแบบเฉพาะ

ครั้งที่	Speed Control = ช้ามาก		Speed Control = ช้า	
	เวลา (นาที)	หมายเหตุ	เวลา (นาที)	หมายเหตุ
1	3:59:09	-	-	เดินออกนอกเส้นทาง
2	3:58:08	-	-	เดินออกนอกเส้นทาง
3	3:58:03	-	1:45:09	-
4	3:59:03	-	-	เดินออกนอกเส้นทาง
5	3:59:01	-	1:44:02	-
เฉลี่ย	3:58:41		1:44:35	-

จากผลการทดสอบแผนที่ไม่มีรูปแบบเฉพาะ พบว่า ถ้ากำหนด Speed Control = ช้ามาก หุ่นยนต์สามารถเดินจากต้นทางไปยังปลายทางโดยไม่ออกนอกเส้นทางทุกครั้งจากการทดสอบจำนวน 5 ครั้ง ซึ่งใช้เวลาเฉลี่ยในการเดิน คือ 3:58:41 นาที แต่ถ้ากำหนด Speed Control = ช้า หุ่นยนต์สามารถเดินจากต้นทางไปยังปลายได้เพียง 2 ครั้งจากการทดสอบทั้งหมด 5 ครั้ง ซึ่งใช้เวลาเฉลี่ยในการเดิน คือ 1:44:35 นาที

4.5 สรุปผลการทดสอบสมรรถนะของโปรแกรมทั้ง 2 โปรแกรม

จากที่ได้ทำการทดสอบสมรรถนะของโปรแกรม FollowLine ในบทที่ 3 และโปรแกรมที่ได้ทำการพัฒนาขึ้นมาใหม่ โดยใช้หลักการของฟัซซี่ลอจิกคอนโทลนั้น ได้ผลการทดสอบรวมดังตารางที่ 4.4, 4.5 และ 45

ตารางที่ 4.4 ผลการทดสอบสมรรถนะของโปรแกรมทั้ง 2 โดยใช้แผนที่วงกลมวนซ้าย

ครั้งที่	โปรแกรม FollowLine		โปรแกรมใหม่ Speed Control = ช้ามาก		โปรแกรมใหม่ Speed Control = ช้า	
	เวลา (นาที)	หมายเหตุ	เวลา (นาที)	หมายเหตุ	เวลา (นาที)	หมายเหตุ
1	2:35:06	หลุดจากเส้น 2 ครั้ง	2:22:07	-	1:06:03	
2	2:35:09	หลุดจากเส้น 2 ครั้ง	2:22:04	-	-	เดินออกนอกเส้นทาง
3	2:35:03	หลุดจากเส้น 2 ครั้ง	2:22:03	-	1:06:01	-
4	2:36:04	หลุดจากเส้น 2 ครั้ง	2:22:01	-	-	เดินออกนอกเส้นทาง
5	2:35:08	หลุดจากเส้น 2 ครั้ง	2:21:09	-	1:05:05	-
เฉลี่ย	2:35:18		2:21:53		1:05:43	-

ตารางที่ 4.5 ผลการทดสอบสมรรถนะของโปรแกรมทั้ง 2 โดยใช้แผนที่วงกลมวนขวา

ครั้งที่	โปรแกรม FollowLine		โปรแกรมใหม่ Speed Control = ช้ามาก		โปรแกรมใหม่ Speed Control = ช้า	
	เวลา (นาที)	หมายเหตุ	เวลา (นาที)	หมายเหตุ	เวลา (นาที)	หมายเหตุ
1	-	เดินออกนอกเส้นทาง	2:19:00	-	-	เดินออกนอกเส้นทาง
2	-	เดินออกนอกเส้นทาง	2:21:03	-	-	เดินออกนอกเส้นทาง
3	-	เดินออกนอกเส้นทาง	2:19:03	-	-	เดินออกนอกเส้นทาง
4	-	เดินออกนอกเส้นทาง	2:21:04	-	-	เดินออกนอกเส้นทาง
5	-	เดินออกนอกเส้นทาง	2:21:00	-	1:04:07	-
เฉลี่ย	-		2:20:14		1:04:07	-

ตารางที่ 4.6 ผลการทดสอบสมรรถนะของโปรแกรมทั้ง 2 โดยใช้แผนที่ไม่มีรูปแบบเฉพาะ

ครั้งที่	โปรแกรม FollowLine		โปรแกรมใหม่ Speed Control = ช้ามาก		โปรแกรมใหม่ Speed Control = ช้า	
	เวลา (นาที)	หมายเหตุ	เวลา (นาที)	หมายเหตุ	เวลา (นาที)	หมายเหตุ
1	3:19:01	หลุดจากเส้น 3 ครั้ง	3:59:09	-	-	เดินออกนอกเส้นทาง
2	3:05:02	หลุดจากเส้น 2 ครั้ง	3:58:08	-	-	เดินออกนอกเส้นทาง
3	3:21:04	หลุดจากเส้น 2 ครั้ง	3:58:03	-	1:45:09	-
4	3:05:08	หลุดจากเส้น 1 ครั้ง	3:59:03	-	-	เดินออกนอกเส้นทาง
5	3:08:03	หลุดจากเส้น 1 ครั้ง	3:59:01	-	1:44:02	-
เฉลี่ย	3:11:40		3:58:41		1:44:35	-

จากผลการทดสอบทั้งหมดสามารถสรุปได้ว่า โปรแกรม FollowLine กับโปรแกรมที่ได้ทำการพัฒนาขึ้นมาใหม่ โดยใช้ฟัซซี่ลอจิกคอนโทลนั้น พบว่า

- แผนที่วงกลมวนซ้าย โปรแกรมที่ได้ทำการพัฒนาใหม่นั้นที่ Speed Control = ช้ามาก ใช้เวลาในการเดินน้อยกว่าโปรแกรม FollowLine และไม่มีการเดินออกนอกเส้นทางโดยใช้เวลาเฉลี่ยในการเดิน คือ 2:21:53 นาที และที่ Speed Control = ช้า ใช้เวลาในการเดินน้อยลงกว่า Speed Control = ช้ามาก แต่มีการเดินออกนอกเส้นทางจำนวน 2 ครั้งจากการทดสอบ 5 ครั้ง
- แผนที่วงกลมรูปวนขวา โปรแกรมที่ได้ทำการพัฒนาขึ้นมาใหม่นั้นที่ Speed Control = ช้ามาก สามารถเดิมจากต้นทางไปยังปลายทางได้ด้วยเวลาเฉลี่ยในการเดิน คือ 2:20:14 นาทีและไม่มีการเดินออกนอกเส้นทาง ในขณะที่โปรแกรม FollowLine ไม่สามารถเดินจากต้นทางไปยังปลายทางได้สำเร็จสักครั้งในการทดสอบ และที่ Speed Control = ช้า ใช้เวลาในการเดินน้อยลงกว่า Speed Control = ช้ามาก แต่มีการเดินออกนอกเส้นทางจำนวน 4 ครั้งจากการทดสอบ 5 ครั้ง

- แผนที่ไม่มีรูปแบบเฉพาะ โปรแกรมที่ได้ทำการพัฒนาใหม่นั้นที่ Speed Control = ช้ามาก ใช้เวลาในการเดินมากกว่าโปรแกรม FollowLine และไม่มีการเดินออกนอกเส้นทางโดยใช้เวลาเฉลี่ยในการเดิน คือ 3:11:40 นาที และที่ Speed Control = ช้า ใช้เวลาในการเดินน้อยลงกว่าโปรแกรม FollowLine และ Speed Control = ช้ามาก แต่มีการเดินออกนอกเส้นทางจำนวน 2 ครั้ง จากการทดสอบ 5 ครั้ง

จากผลการทดสอบทั้งหมดสามารถสรุปได้ว่าโปรแกรมที่ได้ทำการพัฒนาขึ้นมาใหม่ โดยใช้ฟัซซี่ลอจิกคอนโทลนั้น มีประสิทธิภาพดีกว่าและใช้เวลาในการเดินน้อยกว่า ยกเว้นแผนที่ไม่มีรูปแบบเฉพาะ ที่ Speed Control = ช้ามาก กรณีเดียวที่ใช้เวลามากกว่า แต่ถ้าใช้ Speed Control = ช้า จะใช้เวลาในการเดินน้อยลงไปอีก แต่มีประสิทธิภาพที่แย่กว่า คือ มีการเดินออกนอกเส้นทางบ้างครั้ง ซึ่งที่เป็นเช่นนี้ เพราะว่าตัวโปรแกรมที่ได้ทำการพัฒนาขึ้นมาใหม่นั้นใช้กล่องคำสั่งที่มากกว่าโปรแกรม FollowLine โดยโปรแกรม FollowLine ใช้คำสั่งจำนวน 11 คำสั่ง แต่โปรแกรมที่พัฒนาใหม่นั้นใช้คำสั่งจำนวน 115 คำสั่ง ซึ่ง Visual Programming Language 2008 ถ้าใช้คำสั่งมากเท่าไหร่นั้น ยิ่งทำให้โปรแกรมทำงานได้ช้าลง เพื่อหลีกเลี่ยงปัญหานี้ โดยใช้ซอฟต์แวร์ Microsoft Visual Studio 2008 โดยใช้ภาษา C# ในการพัฒนาโปรแกรมแทน Visual Programming Language 2008 ได้

บทที่ 5
บทสรุปและข้อเสนอแนะ

5.1 สรุปผลการพัฒนาโปรแกรม

การพัฒนาโปรแกรมการควบคุมหุ่นยนต์เดินตามเส้นโค้ง โดยใช้ฟัซซี่ลอจิกคอนโทล ได้ดำเนินการศึกษาทฤษฎีฟัซซี่ลอจิกคอนโทล และเทคโนโลยีที่เกี่ยวข้องกับการพัฒนาโปรแกรม โดยนำความรู้ที่ได้ศึกษามาช่วยในการพัฒนาโปรแกรม

สำหรับการวิเคราะห์และออกแบบโปรแกรม ได้ทำการศึกษาและวิเคราะห์ปัญหาจากโปรแกรม FollowLine แล้วนำมาแก้ไขปรับปรุงโปรแกรมใหม่ เพื่อให้หุ่นยนต์สามารถเดินตามเส้นโค้งได้อย่างมีประสิทธิภาพ พบว่าโปรแกรมการควบคุมหุ่นยนต์เดินตามเส้นโค้ง โดยใช้ฟัซซี่ลอจิกคอนโทลนั้น มีประสิทธิภาพที่ดีขึ้น โดยสามารถเดินตามเส้นโค้งแบบที่กำหนดไว้ได้ทุกครั้งในการทดสอบ ซึ่งโปรแกรมการควบคุมหุ่นยนต์เดินตามเส้นโค้ง โดยใช้ฟัซซี่ลอจิกคอนโทลนี้จะใช้ได้ดีก็ต่อเมื่อทำการกำหนดค่า Speed Control = ช้ามาก ซึ่งเมื่อลองกำหนดค่า Speed Control = ช้า พบว่าบางครั้งหุ่นยนต์จะเดินออกนอกเส้นทางที่ได้กำหนดไว้ แต่ผลที่ได้กลับมาคือ ใช้เวลาในการเดินจากต้นทางไปยังปลายทางด้วยเวลาที่น้อยลง ซึ่งสาเหตุที่เป็นเช่นนี้เพราะ

1. ตัวโฟโตเซ็นเซอร์ (Foto Sensor) นั้นมีความไวที่ช้ากว่าการเคลื่อนที่ของหุ่นยนต์ให้ทำให้ส่งค่าไปยังโปรแกรมไม่ทัน ทำให้หุ่นยนต์เดินิกนาเส้นทางได้ ซึ่งอาจจะทำการแก้ไขด้วยหาเซ็ตค่าความไวของตัวโฟโตเซ็นเซอร์ (Foto Sensor) ให้มีค่าความไวที่มากขึ้น เพื่อจะได้ตอบสนองต่อความเร็วของหุ่นยนต์ได้อย่างทันท้วงที
2. กฎพื้นฐาน (Rule-Base) นั้นไม่เหมาะสมกับสภาพแวดล้อมในการทดสอบ ซึ่งกฎพื้นฐาน (Rule-Base) นี้สามารถทำการแก้ไขให้เหมาะสมกับสภาพแวดล้อมแต่ละประเภทได้ แต่ต้องใช้เวลาในการปรับแต่งค่าของกฎพื้นฐาน (Rule-Base) เป็นอย่างมาก เพื่อให้ได้กฎพื้นฐาน (Rule-Base) ที่ดีที่สุด จึงอาจจะสามารถทำให้หุ่นยนต์เดินได้เร็วกกว่านี้ได้

5.2 ประโยชน์ที่ได้รับจากการพัฒนาโปรแกรม

ประโยชน์ที่ได้รับจากการพัฒนาโปรแกรมในโครงงานนี้ สรุปได้ดังนี้

1. ได้นำความรู้ที่ศึกษามาประยุกต์ใช้ในการวิเคราะห์ ออกแบบและพัฒนาโปรแกรมด้วย Microsoft Robotics Developer Studio 2008 รวมถึงได้เรียนรู้ภาษาและเครื่องมือต่างๆ เพื่อใช้ในการพัฒนาหุ่นยนต์ได้ในรูปแบบอื่นๆ ได้

2. เป็นการนำทฤษฎีต่างๆ ที่มีใช้อยู่นั้นมาทำการจำลองการทำงานด้วยเครื่องคอมพิวเตอร์ ทำให้เป็นการเปิดโอกาสแก่ผู้ที่ไม่มีความชำนาญด้านเครื่องกล สามารถสร้าง ออกแบบโปรแกรม และจำลองโปรแกรมที่ได้ทำการออกแบบมานั้นด้วยเครื่องคอมพิวเตอร์

5.3 ปัญหาและอุปสรรคระหว่างการออกแบบและพัฒนาโปรแกรม

ปัญหาและอุปสรรคที่ได้รับจากการพัฒนาโปรแกรมในโครงงานนี้ สรุปได้ดังนี้

1. ต้องใช้เวลาการศึกษาทฤษฎีฟัซซี่ลอจิกคอนโทล เพื่อนำทฤษฎีนี้มาประยุกต์ใช้กับการพัฒนาโปรแกรมด้วย Visual Programming Language 2008
2. เครื่องมือ Visual Programming Language 2008 เป็นเครื่องมือที่มีการกินทรัพยากรของเครื่องเป็นอย่างมาก เพราะฉะนั้นควรจัดเตรียมเครื่องคอมพิวเตอร์ที่มีสมรรถนะสูงไว้สำหรับใช้งาน
3. ตัวแปรในเครื่องมือ Visual Programming Language 2008 ไม่สามารถเปลี่ยนค่าสถานะได้ ถ้าไม่มีการคำนวณหรือการแทนค่าเข้าไปยังตัวแปรนั้น ทำให้ไม่สามารถพัฒนาโปรแกรม โดยใช้ทฤษฎีฟัซซี่ลอจิกคอนโทลโดยตรงได้ จึงต้องนำทฤษฎีฟัซซี่ลอจิกคอนโทลมาทำการประยุกต์ให้เข้ากับข้อจำกัดนี้ ของเครื่องมือ Visual Programming Language 2008

5.4 ข้อเสนอแนะและแนวทางในการพัฒนาโปรแกรม

การควบคุมหุ่นยนต์ให้เดินตามเส้นโค้ง โดยใช้ฟัซซี่ลอจิกคอนโทลในอนาคต ควรจะพิจารณาให้ดำเนินการตามแนวทางต่อไปนี้

1. เครื่องมือ Visual Programming Language 2008 นั้น มีข้อจำกัดบางอยู่ ซึ่ง Microsoft Robotics Developer Studio 2008 นั้นไม่ได้จำกัดว่าต้องใช้เครื่องมือ Visual Programming Language 2008 ในการพัฒนาโปรแกรมเท่านั้น ซึ่งสามารถใช้ซอฟต์แวร์ Microsoft Visual Studio 2008 โดยใช้ภาษา C# ในการพัฒนาโปรแกรมได้เช่นกัน
2. สามารถนำทฤษฎีของระบบอัตโนมัติต่างๆ เช่น นิวรอนเน็ตเวิร์ก, จีเนติก อัลกอริทึ่ม หรือ ปัญญาประดิษฐ์ มาประยุกต์ใช้กับโปรแกรมในการควบคุมหุ่นยนต์ให้สามารถทำงานในรูปแบบต่างๆ ได้

บรรณานุกรม

นันทวัช อัชชารกูร. 2543. "ตัวควบคุมแบบฟัซซี่สำหรับอินเวอร์เท็ดเพนดูลัม." โครงงานพัฒนาระบบงาน หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ คณะเทคโนโลยีสารสนเทศ, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

Johns, K. and Taylor, T. 2008. **ProMRDS Home Page**. [Online] Available: http://www.promrds.com/default.htm

Microsoft. 2008 a. **Microsoft Robotics – Simulation Forum**. [Online] Available: http://social.msdn.microsoft.com/Forums/en-US/roboticssimulation/threads

Microsoft. 2008 b. **Microsoft Robotics – Visual Programming Language Forum**. [Online] Available: http://social.msdn.microsoft.com/Forums/en-US/roboticsvisualprogramminglanguage/threads

Passino, K. M. and Yurkovich, S. 1997. **Fuzzy Control**. California: Addison-Wesley

ประวัติผู้เขียน

ชื่อผู้เขียน	นาย ณัฐพล จิระบวรภิญโญ
สถานที่เกิด	จังหวัดกรุงเทพมหานคร

ประวัติการศึกษา

ระดับมัธยมศึกษาตอนปลาย	โรงเรียนเทพศิรินทร์ จังหวัดกรุงเทพมหานคร
ระดับอุดมศึกษา	วิศวกรรมศาสตรบัณฑิต ภาควิชาวิศวกรรมอิเล็กทรอนิกส์และโทรคมนาคม คณะวิศวกรรมศาสตร์ มหาวิทยาเทคโนโลยีพระจอมเกล้าธนบุรี

ประวัติการทำงาน

พ.ค. 2549 – ปัจจุบัน	บริษัท ศูนย์ประมวลผล จำกัด