ระบบระบุตำแหน่งตนเองโดยใช้ความแรงของคลื่น
ด้วยเทคนิควิธีเชิงปัญญาประดิษฐ์

นายวิชัย ศรีสุรักษ์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า
มหาวิทยาลัยเทคโนโลยีสุรนารี
ปีการศึกษา 2550

A FIELD STRENGTH BASED LOCAL POSITIONING SYSTEM USING AI TECHNIQUES

Wichai Srisuruk

A Thesis Submitted in Partial Fulfillment of the Requirements for the

Degree of Master of Engineering in Electrical Engineering

Suranaree University of Technology

Academic Year 2007

ระบบระบุตำแหน่งตนเองโดยใช้ความแรงของคลื่นด้วยเทคนิควิธีเชิงปัญญาประดิษฐ์

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

คณะกรรมการสอบวิทยานิพนธ์

_____

(อ. ดร.นิมิต ชมนาวัง)
ประธานกรรมการ

_____

(ผศ. ดร.อาทิตย์ ศรีแก้ว)
กรรมการ (อาจารย์ที่ปรึกษาวิทยานิพนธ์)

_____

(ผศ. ดร.รังสรรค์ ทองทา)
กรรมการ

_____

(รศ. ดร.เสาวณีย์ รัตนพานี)
รองอธิการบดีฝ่ายวิชาการ

_____

(รศ. น.อ. ดร.วรพจน์ ขำพิศ)
คณบดีสำนักวิชาวิศวกรรมศาสตร์

วิชัย ศรีสุรักษ์ : ระบบระบุตำแหน่งตนเองโดยใช้ความแรงของคลื่นด้วยเทคนิควิธีเชิงปัญญาประดิษฐ์ (A FIELD STRENGTH BASED LOCAL POSITIONING SYSTEM USING AI TECHNIQUES) อาจารย์ที่ปรึกษา : ผศ. ดร.อาทิตย์ ศรีแก้ว, 117 หน้า

ระบบนำทางหุ่นยนต์เคลื่อนที่ในอาคารเดิมอาศัยการนำทางโดยติดตามเครื่องหมายที่กำหนดไว้ ซึ่งการใช้งานต้องมีการกำหนดเส้นทางสำหรับการเคลื่อนที่ที่แน่นอนไว้ก่อนหุ่นยนต์จะไม่สามารถเคลื่อนที่ออกนอกเส้นทางที่กำหนดได้ ภายหลังมีการพัฒนาระบบนำทางโดยวิธีการระบุตำแหน่งตนเองบนแผนที่วิธีการคือการวัดระยะของจุดทดสอบเทียบกับตำแหน่งอ้างอิงที่กำหนดไว้ก่อน แล้วทำการประมวลผล วิทยานิพนธ์นี้ใช้การวัดระยะทางโดยการวัดความแรงของสัญญาณจากจุดกำเนิดสัญญาณของระบบโครงข่ายท้องถิ่นไร้สายนำมาประมวลผลโดยใช้โครงข่ายประสาทเทียมในการระบุพิกัดตำแหน่งตนเอง

สำหรับระบบระบุตำแหน่งตนเองนี้ ประกอบด้วยโครงข่ายประสาทเทียม 2 ชุด คือ Feed-Forward Multilayer Perceptrons (FF-MLPs) สำหรับเป็นโครงสร้างรูปแบบของพื้นที่ทดสอบและ Radian Basis Function (RBF) สำหรับปรับโครงสร้างของ FF-MLPs ให้เข้ากับสภาพแวดล้อมที่มีการเปลี่ยนแปลงขณะทำการทดสอบเพื่อระบุตำแหน่งตนเอง

โดยทั่วไปการกำหนดโครงสร้างของ FF-MLPs ไม่มีหลักการที่แน่นอนตายตัวและไม่มีทฤษฎีรองรับ แต่สำหรับงานวิจัยนี้โครงสร้างของ FF-MLPs สามารถหาได้แบบอัตโนมัติจากการทำงานของจีนเนติกอัลกอริทึม (Genetic Algorithm หรือ GA) การทำงานของ GA จะเริ่มจากกำหนดค่าความคลาดเคลื่อนมากที่สุดที่ยอมรับได้และโครงสร้างเริ่มต้นของ FF-MLPs ที่ขนาดใหญ่สุดก่อนเริ่มการค้นหา การทำงานของ GA จะมีวัตถุประสงค์หลักคือการมุ่งลดจำนวนชั้นซ่อนเร้นและจำนวนโนดในแต่ละชั้นซ่อนเร้นภายใต้ขอบเขตของค่าความคลาดเคลื่อนที่กำหนดไว้

ผลการทำงานของ GA ได้โครงสร้าง FF-MLPs ที่มีจำนวนชั้นซ่อนเร้นและจำนวนโนดในชั้นซ่อนเร้นลดลงและเหมาะสมกับการใช้งานในวิทยานิพนธ์นี้ ส่วนการระบุตำแหน่งตนเองนั้นสามารถระบุได้ถูกต้องในระยะ 1 เมตร ได้ 91.39 เปอร์เซ็นต์บนพื้นที่ทดสอบขนาด 20 เมตร x 25 เมตร

สาขาวิชาวิศวกรรมไฟฟ้า ลายมือชื่อนักศึกษา_____

ปีการศึกษา 2550 ลายมือชื่ออาจารย์ที่ปรึกษา_____

WICHAI SRISURUK : A FIELD STRENGTH BASED LOCAL POSITIONING SYSTEM USING AI TECHNIQUES. THESIS ADVISOR : ASST. PROF. ARTIT SRIKAEW, Ph.D. 117 PP.

LOCAL POSITIONING/ARTICIAL NEURAL NETWORK/WIRELESS LAN

Indoor robotic navigation systems conventionally depend on tagging navigator, for which the operation requires the predefined motion path that prohibits robot's excursion. Recent developments later based primarily on local positioning indicator referred by a map. The idea is to measure and process the relative distance between a testing point and the reference location. This thesis bases the measurement on the radio signal strength from Wireless Local Area Network (WLAN) sources. The processing unit then adopts the Artificial Neural Network (ANN) in self localizing. The unit consists of 2 sets of ANN, i.e., Feed-Forward Multilayer Perceptrons (FF-MLPs) and Radian Basis Function (RBF) as, respectively, a structure for the testing domain and an environmentally dependent FF-MLPs adjusting structure during self-positioning.

There is generally no exact principal in defining the FF-MLPs structure neither the supporting theory. In this research, however, FF-MLPs structure can be derived automatically using a Genetic Algorithm (GA). The algorithm starts off by defining acceptable imprecise tolerance with the largest possible FF-MLPs structure. The main purpose of the GA is then to reduce the hidden layers and the node numbers therein within the predefined bound. The outcome of the GA process yields the FF-MLPs

structure with the optimal numbers of hidden layers and the respective nodes, suitable for the application in this dissertation. As for the local self-positioning, 1-meter accuracy can be achieved 91.39% coverage of the testing domain of size 20 meter x 25 meter.

School of Electrical Engineering Student's Signature _____

Academic Year 2007 Advisor's Signature_____

กิตติกรรมประกาศ

วิทยานิพนธ์นี้ดำเนินการสำเร็จลุล่วงด้วยดี ผู้วิจัยขอขอบพระคุณบุคคลและกลุ่มบุคคลต่าง ๆ ที่ได้กรุณาให้คำปรึกษา แนะนำ รวมทั้งได้ให้ความช่วยเหลืออย่างดียิ่ง ทั้งด้านวิชาการและด้านการดำเนินงานวิจัย กล่าวคือ

- ผู้ช่วยศาสตราจารย์ ดร.อาทิตย์ ศรีแก้ว อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ได้ให้คำปรึกษา คำแนะนำ และแนะแนวทางอันเป็นประโยชน์ยิ่งต่องานวิจัย รวมถึงได้ช่วยตรวจทาน และแก้ไขรายงานวิทยานิพนธ์เล่มนี้จนทำให้มีความสมบูรณ์ยิ่งขึ้น รวมทั้งเป็นกำลังใจ และเป็นแบบอย่างที่ดีในการดำเนินชีวิตหลาย ๆ ด้านให้กับผู้วิจัยเสมอมา

- อาจารย์ประจำสาขาวิชาวิศวกรรมไฟฟ้า สาขาวิชาวิศวกรรมคอมพิวเตอร์ และสาขาวิชาวิศวกรรมโทรคมนาคม มหาวิทยาลัยเทคโนโลยีสุรนารีทุกท่าน ที่ได้กรุณาให้คำปรึกษา แนะนำ และความรู้ทางด้านวิชาการอย่างดียิ่งมาโดยตลอด

- ผู้ช่วยศาสตราจารย์ สุยุชน์ สัตยประกอบ ที่ให้โอกาสในการศึกษาต่อโดยได้รับทุนพัฒนาอาจารย์ (UDC) จากทบวงมหาวิทยาลัย

- ขอบคุณ พี่ ๆ เพื่อน ๆ น้อง ๆ บัณฑิตศึกษาทุกท่าน รวมถึงมิตรสหายทั้งในอดีตและปัจจุบันโดยเฉพาะอย่างยิ่ง พี่จุก ธวัชชัย กุลวรวานิชพงษ์ ที่คอยให้กำลังใจและถามไถ่ในการทำวิจัยมาโดยตลอด

- คุณประพล จาระตะคุ และคุณวันวิสาข์ ไทยวิโรจน์ พี่ชายและน้องสาวที่แสนดี ที่ช่วยดูแลจัดรูปแบบการพิมพ์ให้ออกมาสวยงาม

- คุณณัฐฏา ศรีสุรักษ์ และเด็กหญิงวิชิตา ศรีสุรักษ์ ภรรยาและลูกสาวที่คอยให้กำลังใจและช่วยเหลือโดยตลอดจนสำเร็จการศึกษาสมตามเจตนารมณ์

สุดท้ายนี้ ผู้วิจัยขอขอบคุณอาจารย์ผู้สอนทุกท่านที่ประสิทธิ์ประสาทความรู้ทางด้านต่าง ๆ ทั้งในอดีตและปัจจุบัน และขอกราบขอบพระคุณ บิดา มารดา รวมถึงญาติพี่น้องของผู้วิจัยทุกท่านที่ได้ให้ความรัก ความอบอุ่น ความห่วงใย การอบรมเลี้ยงดู และให้การสนับสนุนทางด้านการศึกษาอย่างดียิ่งมาโดยตลอด รวมทั้งเป็นกำลังใจที่ยิ่งใหญ่ในยามที่ผู้วิจัยท้อและทุกข์ใจ ช่วยให้มีพลังเข้มแข็งพร้อมเผชิญกับปัญหาอุปสรรคต่าง ๆ จนทำให้ผู้วิจัยประสบความสำเร็จในชีวิตตลอดมา

วิชัย ศรีสุรักษ์

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

ตารางที่ **หน้า**

สารบัญรูป

รูปที่ **หน้า**

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ระบบนำทางหุ่นยนต์เคลื่อนที่ในอาคารเดิมอาศัยการนำทางโดยติดตามเครื่องหมายที่กำหนดไว้ ซึ่งการใช้งานต้องมีการกำหนดเส้นทางสำหรับการเคลื่อนที่ที่แน่นอน หุ่นยนต์จะไม่สามารถเคลื่อนที่ออกจากเส้นทางที่กำหนดได้ภายหลังมีการพัฒนาระบบนำทางโดยวิธีการระบุตำแหน่งตนเองบนแผนที่ งานวิจัยนี้อาศัยระบบโครงข่ายท้องถิ่นไร้สาย (Wireless Local Area Network หรือ WLAN) โดยอ่านสัญญาณจากจุดเข้าถึง (access point) เพื่อวัดระยะจากจุดเข้าถึงที่ทราบตำแหน่งแน่นอนอย่างน้อย 3 จุดนำระยะทางมาประมวลผลเพื่อระบุตำแหน่งตนเองบนแผนที่

หลักสำคัญของการระบุตำแหน่งคือการวัดระยะของจุดทดสอบเทียบกับตำแหน่งอ้างอิงที่กำหนดไว้ การวัดระยะทางแบ่งได้ 3 วิธี คือ

1.1.1 การวัดเวลาระหว่างจุดสองจุด (Time Base Method) วิธีการนี้จำเป็นต้องการเก็บข้อมูลและการประมวลผลข้อมูลที่รวดเร็วมาก เนื่องจากระบบ WLAN ใช้คลื่นวิทยุและความเร็วของคลื่นวิทยุมีค่าประมาณความเร็วถ้าต้องการความถูกต้องของการระบุตำแหน่งที่ 1 เมตรนั่นคือต้องวัดความแตกต่างของเวลาให้ได้ 1 ใน $3x10^8$ เมตรต่อวินาที หรือ 3.333 นาโนวินาที

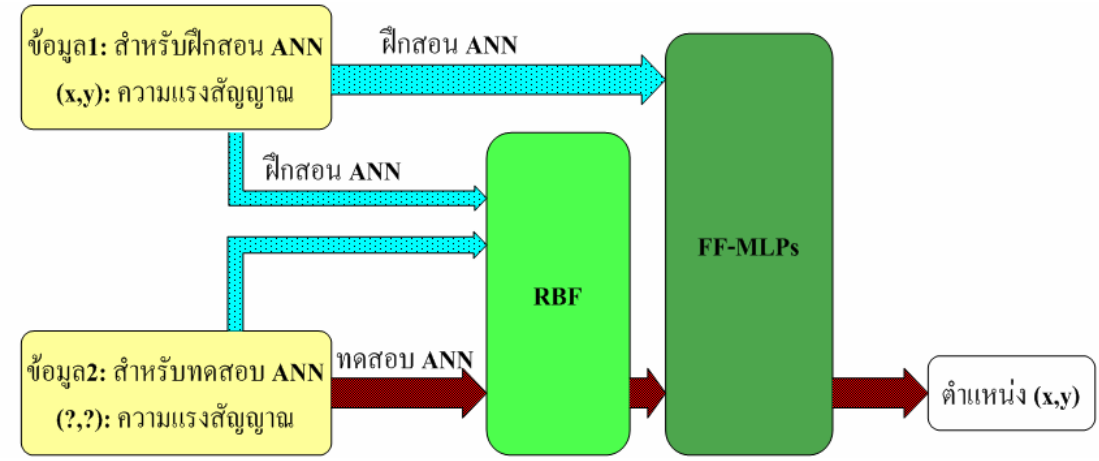
1.1.2 การวัดทิศทางจากจุดอ้างอิง (Direction Based Method) วิธีการนี้ต้องใช้สายอากาศแบบทิศทางที่มีค่าความกว้างลำครึ่งกำลัง (Half Power Beam Width : HPBW) ที่ต่ำมาก เพื่อระบุทิศทางที่แน่นอน สายอากาศที่มีค่าความกว้างลำครึ่งกำลังต่ำนี้ทำได้ยากและมีราคาค่อนข้างแพง อีกทั้งการวัดสัญญาณภายในอาคารจะได้สัญญาณที่มาจากการสะท้อนเข้ามาด้วยทำให้ยากในการระบุตำแหน่ง

1.1.3 การวัดความแรงของสัญญาณจากจุดกำเนิดสัญญาณ (Signal Strength) วิธีการนี้ต้องระวังเรื่องการแทรกสอดของคลื่นวิทยุเนื่องการระบบทำงานภายในอาคาร

ทั้งสามวิธีนี้เป็นเทคนิคในการวัดระยะทางเท่านั้นแต่การระบุตำแหน่งยังต้องอาศัยการประมวลผลสัญญาณเข้ามาร่วมด้วย โดยการประมวลผลสัญญาณเลือกใช้โครงข่ายประสาทเทียม (Artificial Neural Network หรือ ANN) เนื่องจากความแรงของสัญญาณที่วัดได้ภายในอาคารมีความซับซ้อนยากในการเขียนให้อยู่ในรูปความสัมพันธ์ทางคณิตศาสตร์ และโครงข่ายประสาทเทียมสามารถปรับโครงสร้างให้เหมาะสมกับข้อมูลที่มีได้

งานวิจัยวิทยานิพนธ์นี้เลือกใช้วิธีการวัดระยะทางด้วยความแรงของคลื่นจากจุดกำเนิด เพื่อทดสอบอัลกอริทึมในการระบุตำแหน่งตนเองจึงกำหนดให้สิ่งแวดล้อมบริเวณทดสอบไม่มีการเปลี่ยนแปลงและการเก็บข้อมูลจะเคลื่อนที่เล็กน้อยจากจุดทดสอบประมาณความยาวคลื่นของโครงข่ายแลนไร้สายหลาย ๆ จุดแล้วนำมาเฉลี่ยเพื่อลดผลของการแทรกสอดของคลื่นบริเวณจุดทดสอบ

สำหรับการวิจัยนี้ได้โครงสร้างของโครงข่ายประสาทเทียมดังรูปที่ 1.1 ประกอบด้วยโครงข่ายประสาทเทียม 2 ชุด คือ Feed-Forward Multilayer Perceptrons (FF-MLPs) สำหรับเป็นรูปแบบของพื้นที่ทดสอบ และ Radial Basis Function Network (RBF) สำหรับปรับโครงสร้างของ FF-MLPs ให้เข้ากับสภาพแวดล้อมที่มีการเปลี่ยนแปลงขณะทำการทดสอบเพื่อระบุตำแหน่งตนเอง



รูปที่ 1.1 โครงสร้างอย่างง่ายของโครงข่ายประสาทเทียมสำหรับระบบระบุตำแหน่งตนเอง

โดยทั่วไปการกำหนดโครงสร้างของ FF-MLPs ไม่มีหลักการที่แน่นอนตายตัวและไม่มีทฤษฎีรองรับ แต่สำหรับงานวิจัยนี้โครงสร้างของ FF-MLPs สามารถหาได้แบบอัตโนมัติจากการทำงานของจีนเนติกอัลกอริทึม (Genetic Algorithm หรือ GA) การทำงานของจีนเนติกเริ่มจากกำหนดค่าความคลาดเคลื่อนมากที่สุดที่ยอมรับได้และโครงสร้างเริ่มต้นของ FF-MLPs ที่ขนาดใหญ่สุดแล้วให้จีนเนติก ทำการค้นหาโดยมีวัตถุประสงค์หลัก คือ การมุ่งลดจำนวนชั้นซ่อนเร้นและจำนวนโนดในชั้นซ่อนเร้นลงภายใต้ขอบเขตของค่าความคลาดเคลื่อนที่กำหนดไว้

งานวิจัยวิทยานิพนธ์นี้ใช้พื้นที่ห้องปฏิบัติการไมโครโพรเซสเซอร์และห้องปฏิบัติการวงจรและอุปกรณ์ อาคารศูนย์เครื่องมือวิทยาศาสตร์และเทคโนโลยี 3 บนพื้นที่ขนาด 20 x 25 ตารางเมตรในการทดสอบการทำงานของระบบ

1.2 วัตถุประสงค์การวิจัย

1.2.1 เพื่อพัฒนาอัลกอริทึมในการประมวลผลสัญญาณความแรงของคลื่น เพื่อใช้ในการระบุตำแหน่งตนเอง

1.2.2 เพื่อพัฒนาระบบนำทางแบบอัตโนมัติสำหรับหุ่นยนต์ภายในอาคาร

1.3 ข้อตกลงเบื้องต้น

1.3.1 สิ่งแวดล้อมขณะทำการวัดและทดสอบไม่มีการเคลื่อนที่

1.3.2 ไม่พิจารณาผลของอุณหภูมิและความชื้นต่อการทำงานของระบบ

1.3.3 กำหนดให้การ์ดเชื่อมต่อแบบไร้สาย (wireless LAN card) และจุดเข้าถึงยี่ห้อเดียวกันและรุ่นเดียวกันมีคุณสมบัติเดียวกันทั้งหมด

1.4 ขอบเขตงานวิจัย

1.4.1 เก็บข้อมูลความแรงของคลื่นวิทยุ และตำแหน่งของจุดทดสอบเพื่อหาแบบจำลอง

1.4.2 สร้างอัลกอริทึมสำหรับการระบุตำแหน่งจากความแรงของคลื่นวิทยุแบบเวลาจริง

1.4.3 สามารถระบุตำแหน่งในพื้นที่ 20 x 20 ตารางเมตร ความคลาดเคลื่อนไม่เกิน 1 ตารางเมตร และมีความถูกในการระบุตำแหน่ง 80 เปอร์เซ็นต์

1.5 วิธีการดำเนินงานวิจัย

1.5.1 แนวทางการดำเนินงาน

- สำรวจปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้องกับวิทยานิพนธ์
- สำรวจหรือออกแบบโปรแกรมสำหรับเก็บข้อมูลความแรงสัญญาณจาก WLAN
- ออกแบบโปรแกรมสำหรับการระบุตำแหน่งตนเอง
- ทดสอบการทำงานของระบบระบุตำแหน่งตนเอง

1.5.2 ระเบียบวิธีวิจัยเป็นงานวิจัยประยุกต์ ซึ่งดำเนินการตามกรอบงานดังต่อไปนี้

- สำรวจปริทัศน์วรรณกรรม และงานวิจัยที่เกี่ยวข้อง
- ค้นหาหรือออกแบบโปรแกรมเพื่อดึงข้อมูลความแรงสัญญาณจากการ์ดเชื่อมต่อแบบไร้สาย
- ประมวลผลด้วยโปรแกรม MATLAB® สำหรับระบบระบุตำแหน่งตนเอง
- ทดสอบการทำงานของระบบระบุตำแหน่งตนเอง

1.5.3 สถานที่ทำการวิจัย ห้องปฏิบัติไมโครโพรเซสเซอร์ ศูนย์เครื่องมือวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยเทคโนโลยีสุรนารี 111 ถ.มหาวิทยาลัย ต.สุรนารี อ.เมือง นครราชสีมา จ.นครราชสีมา 30000 โทร.0-4422-3383 โทรสาร.0-4422-3394

1.5.4 เครื่องมือที่ใช้ในการวิจัย

- คอมพิวเตอร์ส่วนบุคคล
- โปรแกรมเฉพาะทางวิศวกรรม MATLAB®
- อุปกรณ์จุดเข้าถึงสำหรับโครงข่ายไร้สาย
- การ์ดเชื่อมต่อแบบไร้สายสำหรับคอมพิวเตอร์ส่วนบุคคล

1.5.5 การเก็บรวบรวมข้อมูล

- เก็บรวบรวมข้อมูลจาการสำรวจปริทัศน์วรรณกรรมที่เกี่ยวข้อง
- ใช้คอมพิวเตอร์ร่วมกับการ์ดเชื่อมต่อแบบไร้สายในการอ่านความแรงของสัญญาณจากจุดเข้าถึง และเก็บรวบรวมข้อมูลในรูปของไฟล์ข้อมูล
- นำข้อมูลดังกล่าวไปวิเคราะห์เพื่อระบุตำแหน่งตนเอง

1.5.6 การวิเคราะห์ข้อมูลความแรงสัญญาณที่วัดได้ ความรู้เกี่ยวกับการทำงานของโครงข่ายประสาทเทียมและความรู้เกี่ยวกับการทำงานของจีนเนติกอัลกอริทึมจะถูกนำไปวิเคราะห์ด้วยเทคนิควิธีเฉพาะทางวิศวกรรม

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1.6.1 มีอัลกอริทึมในการประมวลผลสัญญาณความแรงของคลื่นเพื่อใช้ในการระบุตำแหน่ง

1.6.2 มีแนวคิดสำหรับระบบนำทางแบบอัตโนมัติสำหรับหุ่นยนต์ภายในอาคาร

1.7 ส่วนประกอบของวิทยานิพนธ์

วิทยานิพนธ์นี้ได้แบ่งออกเป็น 6 บท ดังนี้คือ

บทที่ 1 เป็นบทนำกล่าวถึง ความเป็นมาและความสำคัญของปัญหา วัตถุประสงค์ของการวิจัย ข้อตกลงเบื้องต้น ขอบเขตของการวิจัย วิธีดำเนินการวิจัย ประโยชน์ที่คาดว่าจะได้รับ และรายละเอียดในวิทยานิพนธ์

บทที่ 2 กล่าวถึงทฤษฎีและหลักการที่ถูกใช้ในการประมาณค่าตำแหน่ง การใช้โครงข่ายประสาทเทียมในการประมาณค่าตำแหน่ง และแนวคิดการปรับปรุงโครงสร้างของโครงข่ายประสาทเทียมด้วยจีนเนติกอัลกอริทึมที่มีส่วนเกี่ยวกับงานวิจัยวิทยานิพนธ์นี้

บทที่ 3 การออกแบบระบบระบุตำแหน่งตนเอง กล่าวถึงพื้นฐานของโครงข่ายประสาทเทียมพร้อมทั้งอธิบายการทำงานของโครงข่ายประสาทเทียมชนิด Feed-Forward Multi-Layer Perceptrons (FF-MLPs) และ Radial Basis Function (RBF) การทำงานของโครงข่ายท้องถิ่นแบบไร้สาย และการใช้โครงข่ายประสาทเทียมในการระบุตำแหน่งตนเอง

บทที่ 4 การทดสอบระบบระบุตำแหน่งตนเอง กล่าวถึงการกำหนดพื้นที่สำหรับทดสอบ การเก็บข้อมูลความแรงของสัญญาณจากจุดเข้าถึง การหาแบบจำลองของพื้นที่ทดสอบด้วย FF-MLPs การทดสอบเพื่อระบุตำแหน่งโดยใช้ FF-MLPs ทำงานร่วมกับ RBF

บทที่ 5 การปรับปรุงโครงสร้างของ FF-MLPs โดยใช้จีนเนติกในค้นหาโครงสร้างของ FF-MLPs ที่เหมาะสมแบบอัตโนมัติ

บทที่ 6 สรุปงานวิจัยที่ทำมาทั้งหมด และมีข้อเสนอแนะเกี่ยวกับงานวิจัยที่สามารถทำได้ในอนาคต

บทที่ 2
ปริทัศน์วรรณกรรมและทฤษฎีที่เกี่ยวข้อง

2.1 บทนำ

การระบุตำแหน่งตนเอง คือการหาตำแหน่งตนเองเทียบกับตำแหน่งอ้างอิง การหาตำแหน่งจะสามารถนำไปใช้ในระบบนำทางหุ่นยนต์เคลื่อนที่ในอาคารแบบอัตโนมัติได้ หลักสำคัญของการระบุตำแหน่ง คือการวัดระยะของจุดทดสอบเทียบกับตำแหน่งอ้างอิง

เทคนิคการวัดระยะทางสามารถแบ่งได้ 3 วิธี คือ การวัดเวลาระหว่างจุดกำเนิดสัญญาณและจุดทดสอบ การวัดความแรงของสัญญาณจากจุดกำเนิดสัญญาณ และการวัดทิศทางจากจุดกำเนิดสัญญาณอ้างอิง ทั้งสามวิธีคือเทคนิคในการวัดเพื่อหาระยะทางจากจุดอ้างอิงแต่การระบุตำแหน่งต้องอาศัยการประมวลผลสัญญาณเข้ามาร่วมด้วยรายละเอียดของทั้ง 3 วิธี มีดังนี้

การวัดเวลาการมาถึงของสัญญาณ (Time Of Arrival Method: TOA Method): การวัดเวลาเมื่อนำมาใช้ภายในอาคารที่มีระยะทางที่สั้น เวลาที่ใช้ในการรับส่งสัญญาณจะน้อยมากทำให้ยากต่อการทำงาน ยกตัวอย่างเช่น ที่ความละเอียดของระยะทาง 1 เมตรใช้เวลาเท่ากับ $1/3x10^{8}$ เมตรต่อวินาทีหรือ $3.33x10^{-9}$ วินาที นั่นคือหากต้องการความถูกต้องที่ 1 เมตรระบบต้องสามารถแยกสัญญาณที่ต่างกันในเวลา $3.33x10^{-9}$ วินาทีได้

การวัดมุมการมาถึงของสัญญาณ (Angle Of Arrival Method: AOA Method): เทคนิคการวัดทิศทางจำเป็นต้องใช้สายอากาศแบบทิศทางซึ่งทำได้ยากและมีราคาค่อนข้างแพง อีกทั้งยังต้องเจอปัญหาความผิดพลาดจากปรากฏการณ์การกระเจิง (scattering) ของคลื่นวิทยุ

การวัดค่าความแรงของสัญญาณ (Signal Strength Method: SS Method): เป็นการวัดค่าความแรงของสัญญาณซึ่งถูกลดทอนลงเนื่องจากการสูญเสียในวิถี (path loss attenuation) การวัดค่าความแรงของสัญญาณจะได้รับผลกระทบจากปรากฏการณ์เฟดดิงพหุวิถี (multipath fading) และการถูกบัง (shadowing) ซึ่งจะมีผลกระทบต่อค่าความแรงของสัญญาณ ทำให้การวัดมีความคลาดเคลื่อน

งานวิจัยวิทยานิพนธ์นี้เลือกใช้เทคนิคการวัดความแรงของสัญญาณในการวัดระยะทางโดยใช้การ์ดเชื่อมต่อไร้สายในการรับสัญญาณ จากจุดเข้าถึงที่มีพิกัดตำแหน่งแน่นอน สำหรับการทดสอบการระบุตำแหน่งนี้เพื่อเลี่ยงผลของปรากฏการณ์เฟดดิงพหุวิถีและการถูกบังจะกำหนดสภาวะแวดล้อมไม่ให้มีการเปลี่ยนแปลงตลอดช่วงการเก็บข้อมูลสำหรับฝึกสอนและข้อมูลสำหรับการทดสอบ

เนื้อหาของบทนี้นอกจากการเลือกวิธีในการระบุตำแหน่ง แล้วยังอธิบายถึงที่มาของการใช้โครงข่ายประสาทเทียม และแนวคิดการใช้จีนเนติกในการช่วยหาโครงสร้างของโครงข่ายประสาทเทียมแบบหลายชั้นด้วย

2.2 แนวคิดการประมาณค่าเพื่อระบุตำแหน่ง

หัวใจหลักของการระบุตำแหน่งคือการวัดระยะทาง ในวิทยานิพนธ์นี้ใช้การวัดความแรงของสัญญาณจากจุดเข้าถึงที่มีตำแหน่งอ้างอิงต่าง ๆ กันอย่างน้อย 3 ตัว เพื่อคำนวณหาตำแหน่งของตนเอง จุดทดสอบจะเคลื่อนที่อยู่ในบริเวณที่ครอบคลุมการทำงานของจุดเข้าถึงทั้งสามตัว ข้อมูลที่ต้องการคือความแรงของสัญญาณจากการ์ดเชื่อมต่อไร้สายที่รับได้จากจุดอ้างอิงทั้งสามจุด ความแรงของสัญญาณนี้จะลดลงเมื่อระยะทางระหว่างเครื่องรับ-เครื่องส่ง เพิ่มขึ้น ดังความสัมพันธ์ที่ 2.1

$$P_r(d) = \frac{P_t G_t G_r l^2}{(4\pi)^2 d^2 L} \tag{2.1}$$

$$P_r(d) + loss(d) \propto \frac{1}{d^2} \tag{2.2}$$

เมื่อ

P_r, P_t	คือ กำลังของเครื่องรับและเครื่องส่ง
G_r, G_t	คือ กำลังขยาย (gain) ของสายอากาศรับและสายอากาศส่ง
l	คือ ความยาวคลื่น
L	คือ ตัวประกอบการสูญเสีย (loss factor)
d	คือ ระยะทางระหว่างเครื่องรับและเครื่องส่ง

หรืออาจกล่าวได้ว่าสัญญาณที่รับได้เป็นสัดส่วนผกผันกับระยะทางกำลังสอง อย่างไรก็ดียังมีส่วนของสัญญาณที่สูญเสียที่ต้องพิจารณาด้วยเพราะเป็นฟังก์ชันของระยะทางเหมือนกัน การประมาณค่าเพื่อระบุตำแหน่งนี้ แบ่งได้ 2 วิธีหลัก [Battiti and Villiani : 2002] คือ การประมาณค่าโดยใช้ระยะระหว่างจุดสามจุด (trilateration approach) และ การประมาณค่าโดยใช้วิธีฟิงเกอร์ปริ้นท์ (fingerprint approach)

2.2.1 การประมาณตำแหน่งค่าจากระยะระหว่างจุดสามจุด

วิธีการนี้เป็นวิธีที่ง่าย โดยหลักการคือการมีตำแหน่งอ้างอิง 3 จุด (หรือมากกว่า) ที่ทราบตำแหน่งแน่นอน เมื่อทราบระยะระหว่างจุดทดสอบถึงตำแหน่งอ้างอิงก็วาดวงกลมสามวงจะได้จุดตัดและจุดตัดนี้คือตำแหน่งของจุดทดสอบ ขั้นตอนการทำงานแบ่งออกเป็น 2 ขั้น คือ การวัดระยะจากขนาดของความแรงสัญญาณคลื่น และการคำนวณเพื่อหาตำแหน่งของจุดทดสอบ ขั้นตอนสำคัญคือการวัดระยะจากความแรงของสัญญาณ ขั้นตอนการทำงานสามารถอธิบายได้ดังนี้

ขั้นตอนการวัดระยะจากความแรงสัญญาณโดยเริ่มจากการวัดความแรงสัญญาณที่จุดต่าง ๆ สร้างสัมพันธ์ระหว่างความแรงสัญญาณกับระยะทาง คำนวณระยะจากความสัมพันธ์ที่มี จากข้อมูลความแรงสัญญาณ วิธีการนี้มีความซับซ้อนมากเนื่องจากสภาพแวดล้อมรอบตัวจะส่งผลต่อขนาดความแรงของสัญญาณ ดังนั้นระยะที่หาได้จะมีความคลาดเคลื่อนมาก

ขั้นตอนการประมาณตำแหน่ง จากรูปที่ 2.1 เมื่อทราบระยะทางจากจุดอ้างอิงสามจุดด้วยวิธีการทางคณิตศาสตร์สามารถเขียนในรูปสมการวงกลมได้ดังความสัมพันธ์ที่ 2.3 เลือกสมการวงกลมของ r_1 และ r_2 สามารถหาจุด a, b ที่ต้องการได้

สมการวงกลม

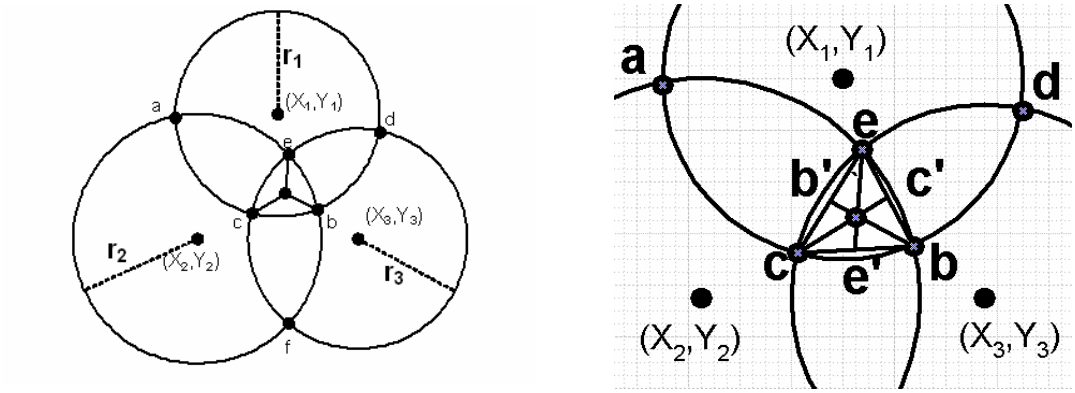
$$x^2 + y^2 + A_1 x + B_1 y + C_1 = 0 \tag{2.3}$$

กำหนดให้ $x = x_a$ ในสมการวงกลมรัศมี r_1 และ วงกลมรัศมี r_2 ตามลำดับจะได้

$$y^2 + B_1 y + D_1 = 0 \tag{2.4}$$

$$y^2 + B_2 y + D_2 = 0 \tag{2.5}$$

นำความสัมพันธ์ที่ 2.4 ลบด้วย 2.5 และแก้สมการได้ $y = y_a$ แทนค่าในความสัมพันธ์ที่ 2.3 จะได้ตำแหน่งของจุด a (x_a, y_a) ด้วยวิธีเดียวกันนี้สามารถหาจุด b, c, d, e และ f ได้ เลือกจุด c, b, e สร้างสามเหลี่ยม ceb เพื่อหาจุดศูนย์กลางมวล (center of mass) ของสามเหลี่ยม cbe มีขั้นตอนดังนี้ หาจุด c', b' และ e' ซึ่งเป็นจุดกลางของเส้นตรง eb, ec และcb ตามลำดับ เลือกเส้นตรง cc' หรือ bb' หรือ ee' หนึ่งคู่เพื่อหาจุดตัดสมการเส้นตรง คำตอบที่ได้เป็นจุดทดสอบ (x, y) ที่ต้องการ

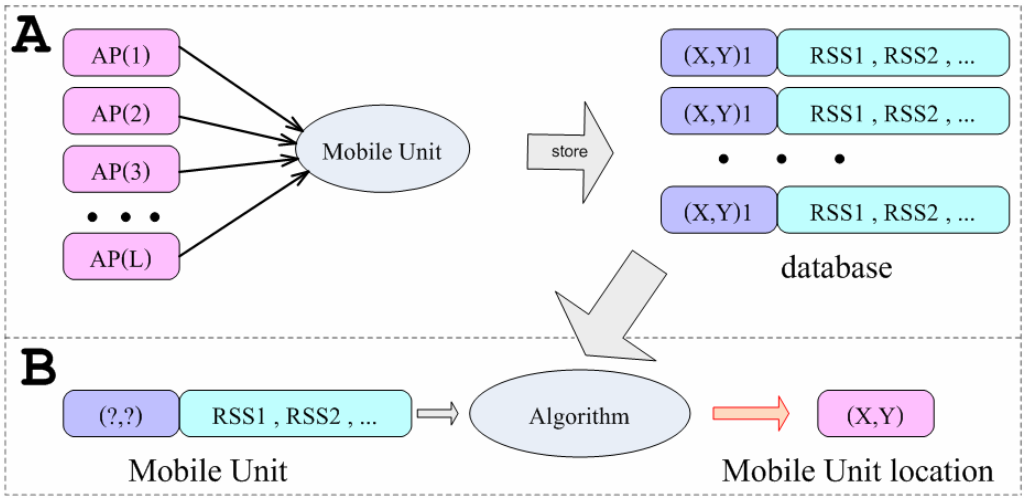


รูปที่ 2.1 การระบุตำแหน่งโดยวิธีหาผลเฉลยของคำตอบของวงกลม 3 รูปตัดกัน

ผลการระบุตำแหน่งด้วยวิธีการนี้ยังไม่ดีพอเนื่องจากการวัดความแรงสัญญาณภายในอาคารมีความซับซ้อนมาก สิ่งที่มีผลต่อความแรงสัญญาณมาจากหลายสาเหตุ เช่น ผลของระยะทางซึ่งเป็นสิ่งที่เราสนใจ ผลของการลดทอนสัญญาณจากกำแพงหรือสิ่งกีดขวาง ผลจากการสะท้อนของสัญญาณจากกำแพงหรือสิ่งกีดขวาง และการแทรกสอดจากสัญญาณโดยรอบ ดังนั้นการหาความสัมพันธ์ระหว่างความแรงสัญญาณกับระยะทางที่ถูกต้องจึงทำได้ยากมาก

2.2.2 การประมาณตำแหน่งด้วยวิธีฟิงเกอร์ปริ้นท์ติ้ง

การประมาณค่าตำแหน่งด้วยวิธีฟิงเกอร์ปริ้นท์ติ้งนี้ประกอบด้วย 2 ขั้นตอนคือ การฝึกสอนและการทดสอบเพื่อระบุพิกัดตำแหน่ง ขั้นตอนการฝึกสอนเพื่อหารูปแบบการกระจายของข้อมูลในพื้นที่ทดสอบ เริ่มจากการวัดความแรงสัญญาณจากจุดเข้าถึงทุก ๆ จุดที่มี เก็บข้อมูลความแรงสัญญาณที่จุดทดสอบต่าง ๆ จนครอบคลุมของพื้นที่ทดสอบ ขั้นตอนการทดสอบ คือ การอ่านค่าความแรงสัญญาณจากจุดเข้าถึงนำข้อมูลมาประมวลผลจากฐานข้อมูลที่มีเพื่อหาตำแหน่ง อัลกอริทึมที่ใช้เป็น appropriate search หรือ matching algorithm กระบวนการทั้งหมดแสดงดังรูปที่ 2.2



รูปที่ 2.2 การระบุตำแหน่งโดยวิธีฟิงเกอร์ปริ้นท์ติ้ง (A) ขั้นการฝึกสอน (B) ขั้นการทดสอบ

อัลกอริทึมที่ใช้ในการประมาณค่าตำแหน่งมีหลายวิธี วิธีพื้นฐานอย่างหนึ่งคือ Nearest Neighbors: NN วิธีการนี้ใช้การวัดระยะระหว่างความแรงของสัญญาณที่มีในฐานข้อมูล $[S_1,S_2,\ldots,S_n]$ กับความแรงของสัญญาณที่จุดทดสอบ $[s_1,s_2,\ldots,s_n]$ ดังความสัมพันธ์ที่ 2.6 คำตอบที่ได้คือจุดที่ให้ระยะน้อยสุดเมื่อเทียบกับจุดอื่น

$$L_q = \left(\sum_{i=1}^{n} |s_i - S_i|^q \right)^{\frac{1}{q}} \tag{2.6}$$

อัลกอริทึมอื่นที่ใช้ เช่น กระบวนการทางสถิติ (probabilistic method of fingerprint) หรือการใช้โครงข่ายประสาทเทียม เป็นต้น

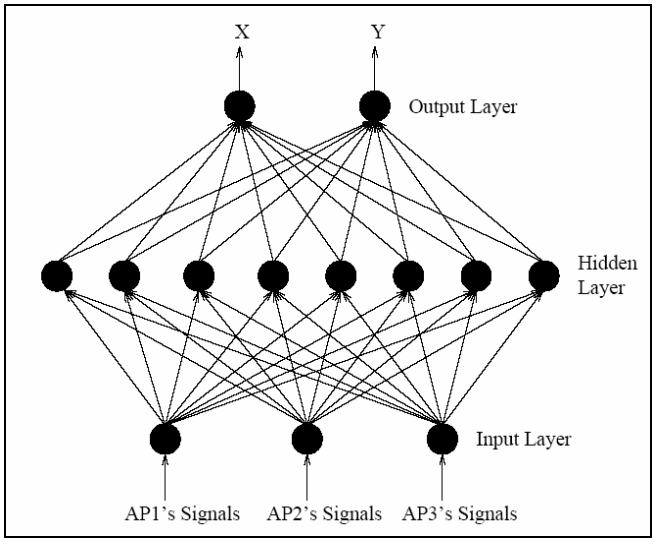
สำหรับงานวิจัยวิทยานิพนธ์นี้ใช้การประมาณค่าด้วยวิธีฟิงเกอร์ปริ้นท์ติ้งและใช้โครงข่ายประสาทเทียมเป็นอัลกอริทึมในการทำงาน รายละเอียดในการระบุตำแหน่งจะกล่าวเพิ่มเติมในบทที่ 3 การออกแบบระบบระบุตำแหน่งตนเองต่อไป

2.3 การใช้โครงข่ายประสาทเทียมเพื่อระบุตำแหน่ง

ปี ค.ศ. 2002 R.Battiti ได้นำเสนอการระบุตำแหน่งตนเองโดยใช้โครงข่ายแลนไร้สายร่วมกับโครงข่ายประสาทเทียม จากข้อเท็จจริงที่ว่าความแรงของสัญญาณวิทยุมาจากแหล่งกำเนิดสัญญาณหลายตัวมีความสัมพันธ์กับตำแหน่งของจุดรับสัญญาณ แต่การหาตำแหน่งจากความแรง

ของสัญญาณที่รับได้จากจุดกำเนิดต่าง ๆ ทำได้ยากและมีหลายคำตอบ ดังนั้นจึงมีการใช้แบบจำลองที่มีความยืดหยุ่นของโครงข่ายประสาทเทียม

ข้อดีของการใช้กระบวนการนี้ คือใช้โครงข่ายแลนไร้สายที่มีอยู่เดิม และด้วยลักษณะของแบบจำลองโครงข่ายประสาทเทียมที่มีความยืดหยุ่นปรับขนาดได้และมีการเรียนรู้ทำให้ผลลัพธ์ของการหาตำแหน่งถูกต้องมากขึ้น การระบุตำแหน่งด้วยวิธีนี้ไม่จำเป็นต้องทราบตำแหน่งที่แน่นอนของจุดอ้างอิงและไม่คำนึงถึงโครงสร้างของอาคารตราบใดที่ขั้นตอนการฝึกสอนและการทดสอบอยู่ภายใต้สภาวะแวดล้อมเดียวกัน



รูปที่ 2.3 ตัวอย่างโครงสร้างของโครงข่ายประสาทเทียม

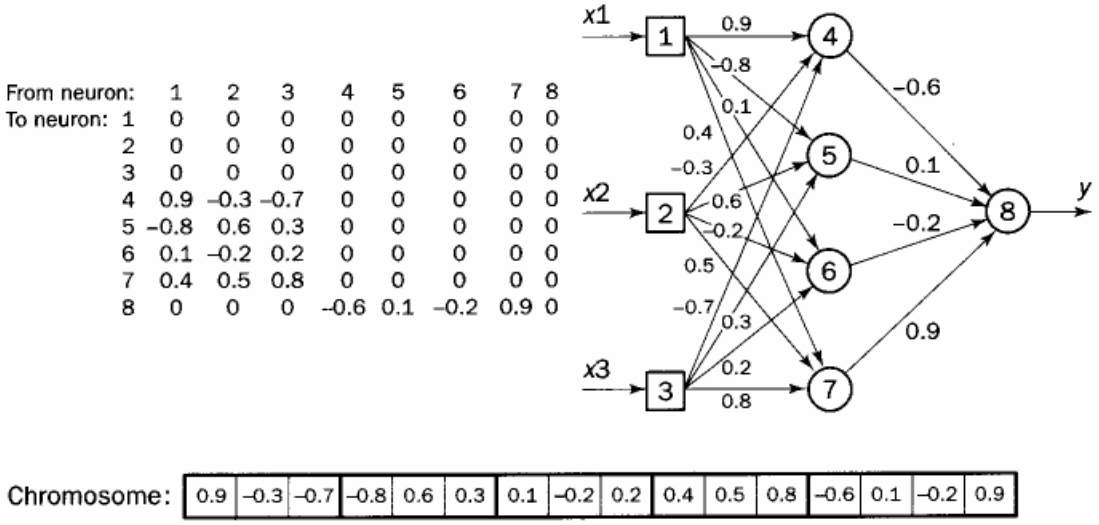
การทดสอบโดยการวัดความแรงสัญญาณที่จุดต่าง ๆ จากจุดเข้าถึงหลายตัวเลือกโครงสร้างแบบ Feed Forward Multi-Layer Perceptrons: FF-MLPs ที่มีชั้นซ่อนเร้น 1 ชั้น จำนวนโนด 16 โนด กำหนดฟังก์ชันการถ่ายโอนแบบซิกมอยดอล (sigmoidal function) ทดสอบบนพื้นที่ขนาด 25.5 เมตร x 24.5 เมตร กำหนดจุดทดสอบจำนวน 196 จุด ผลการทดสอบได้ความละเอียดเฉลี่ย 1.82 เมตร อย่างไรก็ตามสามารถเพิ่มความถูกต้องของการระบุตำแหน่งให้สูงขึ้นได้หากมีการเพิ่มจำนวนของ AP ที่ใช้อ้างอิง [Youssef and Ashok: 2004, P.Myllymaki and T. Roos: 2001]

สำหรับงานวิจัยวิทยานิพนธ์นี้ใช้โครงสร้างแบบ FF-MLPs ทำงานร่วมกับโครงข่ายประสาทเทียมชนิด Radius Basis Function Network: RBF เพื่อเพิ่มความถูกต้องในการระบุตำแหน่ง รายละเอียดจะกล่าวเพิ่มเติมในบทที่ 3 การออกแบบระบบระบุตำแหน่งตนเอง

2.4 การใช้จีนเนติกกับโครงข่ายประสาทเทียม

โครงข่ายประสาทเทียมถูกนำไปใช้ในการแก้ปัญหาต่าง ๆ อย่างแพร่หลาย ปัญหาในการใช้งานโครงข่ายประสาทเทียมอย่างหนึ่งเกิดจากขั้นตอนการฝึกสอนโครงข่ายประสาทเทียม โดยส่วนใหญ่โครงข่ายประสาทเทียมที่ใช้จะเป็นการฝึกสอนแบบแพร่กลับ การฝึกสอนโดยตรงอาจไม่ลู่เข้าหาคำตอบ สาเหตุอาจมาจากการกำหนดฟังก์ชันถ่ายโอนไม่เหมาะสม การกำหนดช่วงของอินพุตไม่เหมาะสมหรือแม้กระทั่งการกำหนดค่าเริ่มต้นสำหรับการฝึกสอนโครงข่าย อย่างไรก็ตามการเริ่มต้นใช้งานโครงข่ายประสาทเทียมยังต้องมีการกำหนดโครงสร้างของโครงข่ายประสาทเทียมก่อนซึ่งต้องอาศัยประสบการณ์ในการกำหนดโครงสร้าง

จีนเนติกอัลกอริทึมเป็นเครื่องมือหนึ่งในกระบวนการการหาค่าเหมาะที่สุด (optimization) การใช้งานจีนเนติกจะช่วยในการหาค่าน้ำหนักของแต่ละโนดและโครงสร้างของโครงข่ายประสาทเทียม

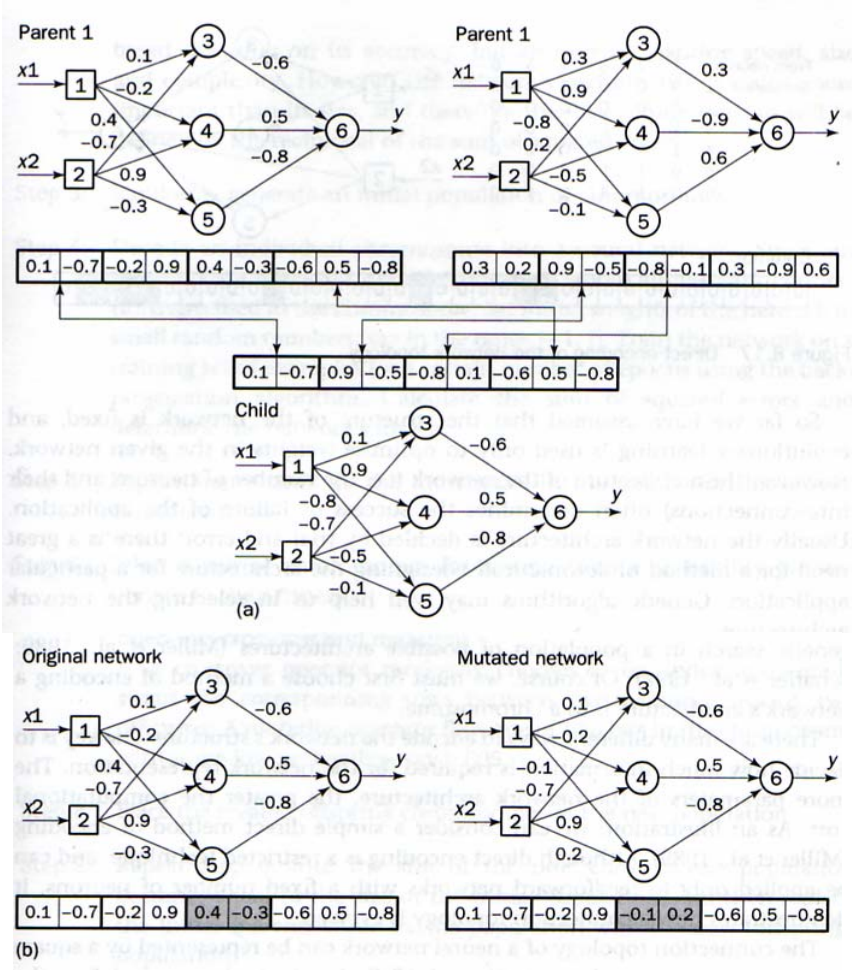


รูปที่ 2.4 การสร้างโครโมโซมจากค่าน้ำหนักของโครงข่ายประสาทเทียม [Negnevitsky: 2002]

แนวคิดเรื่องการฝึกสอนเริ่มจากการหาค่าน้ำหนักของแต่ละโนดในโครงข่ายประสาทเทียม (Montana and Davis: 1989; Whitley and Hanson: 1989) การทำงานดังรูปที่ 2.4 โดยการกำหนดให้โครโมโซมเป็นค่าน้ำหนักของโครงข่าย จากรูปประกอบด้วยค่าน้ำหนัก 16 ค่า มีค่าในช่วง -1 ถึง 1 ถ้าค่านี้เป็นศูนย์หมายถึงไม่มีการเชื่อมต่อระหว่างโนดเกิดขึ้น เมื่อกำหนดโครโมโซมได้แล้วก็เข้าสู่กระบวนการของจีนเนติกในการหาค่าองค์ประกอบของโครโมโซม จากนั้นนำค่าของโครโมโซม

ประกอบขึ้นเป็นโครงข่ายประสาทเทียมเพื่อคำนวณหาค่าผิดพลาดจากเป้าหมายที่ตั้งไว้ที่เกิดจากโครโมโซมทดสอบ

ในขั้นตอนการทำงานของจีนเนติกจะมีการทำครอสโอเวอร์และการทำมิวเตชัน รูปที่ 2.5 เป็นการทำครอสโอเวอร์ระหว่างโครโมโซม และเป็นการทำมิวเตชันโดยการแทนที่ค่าของโครโมโซมบางตัวในในโครโมโซมด้วยค่าระหว่าง -1 ถึง 1



รูปที่ 2.5 กระบวนการทางจีนเนติก (a) ครอสโอเวอร์ และ (b) มิวเตชัน [Negnevitsky: 2002]

กระบวนการทำงานของจีนเนติก เริ่มจากกำหนดชุดข้อมูลสำหรับฝึกสอนโครงข่ายประสาทเทียม และกำหนดโครงข่ายเริ่มต้นที่เป็นโครงข่ายประสาทเทียมใหญ่ที่สุดที่เป็นไปได้ จากนั้นทำงานตามลำดับดังนี้

ขั้นตอนที่ 1: กำหนดจำนวนประชากร, ค่าเปอร์เซ็นต์การทำครอสโอเวอร์, ค่าเปอร์เซ็นต์การทำมิวเตชั่นและจำนวนรอบการทำงานของจีนเนติก

ขั้นตอนที่ 2: กำหนดฟังก์ชันวัตถุประสงค์

ขั้นตอนที่ 3: สร้างประชากรเริ่มต้นโดยการสุ่ม

ขั้นตอนที่ 4: ถอดรหัสโครโมโซมให้เป็นโครงสร้างของโครงข่ายประสาททำการคำนวณหาค่าความผิดพลาดโดยใช้ข้อมูลทดสอบ

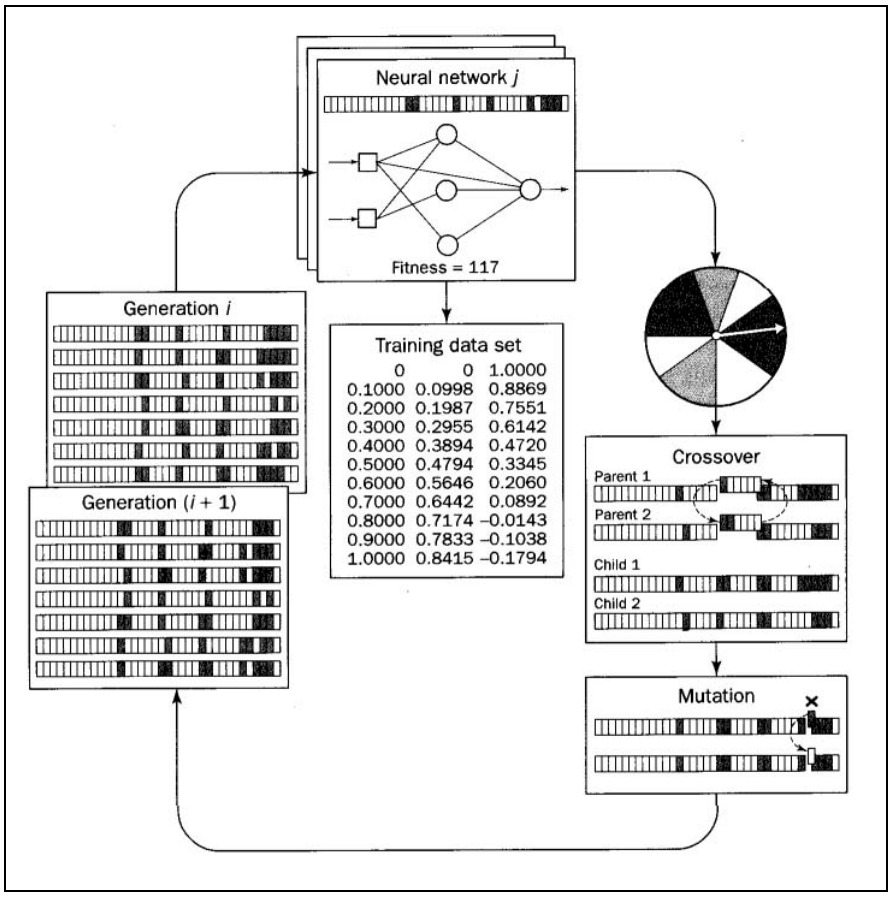
ขั้นตอนที่ 5: ทำซ้ำขั้นตอนที่ 4 สำหรับประชากรทุกตัว

ขั้นตอนที่ 6: คัดเลือกประชากรที่ให้ค่าความผิดพลาดน้อยที่สุด

ขั้นตอนที่ 7: สร้างประชากรรุ่นลูกหลานทำกระบวนการครอสโอเวอร์และทำกระบวนการทำมิวเตชันกับประชากรรุ่นลูกหลาน

ขั้นตอนที่ 8: แทนที่ประชากรเดิมด้วยประชากรรุ่นลูกหลาน

ขั้นตอนที่ 9: ทำซ้ำตั้งแต่ขั้นตอนที่ 4 จนครบจำนวนรอบการทำงานของจีนเนติกที่ตั้งไว้



รูปที่ 2.6 การทำงานของจีนเนติกในการหาโครงสร้างของโครงข่ายประสาทเทียม

ขั้นตอนการทำงานของจีนเนติกแสดงดัง รูปที่ 2.6 การทำงานของจีนเนติกในการหาโครงสร้างของโครงข่ายประสาทเทียม อย่างไรก็ดีคำตอบที่ได้อาจไม่ใช่โครงสร้างโครงข่ายประสาทเทียมที่ดีที่สุด (global solution) เนื่องจากคำตอบของการหาโครงสร้างของโครงข่ายประสาทเทียมมีได้หลายลักษณะและมีความสัมพันธ์ระหว่างกันซับซ้อน แต่การหาโครงสร้างด้วยจีนเนติกนี้ยืนยันได้ว่าโครงสร้างที่ได้ปีนคำตอบที่ดีที่สุดภายใต้เงื่อนไขที่กำหนด เช่น จำนวนรอบการทำงาน ข้อมูลที่ใช้สำหรับฝึกสอนโครงข่าย โครงสร้างเริ่มต้นสำหรับการค้นหา เป็นต้น

การใช้จีนเนติกสำหรับวิทยานิพนธ์ มีทั้งการนำแนวคิดบางส่วนมาใช้และเพิ่มแนวคิดใหม่เข้าไปด้วย แนวคิดที่เลือกมาใช้คือ การกำหนดค่าในโครโมโซมด้วยค่าน้ำหนักของแต่ละโนด การกำหนดว่าถ้าค่าในโครโมโซมเป็นศูนย์ให้หมายถึงไม่มีการเชื่อมต่อระหว่างโนด แนวคิดที่เพิ่มเติมเข้ามาคือ การกำหนดค่าความผิดพลาดมากสุดที่ยอมรับได้เพื่อบังคับให้การค้นหาของจีนเนติกให้ง่ายขึ้น และการบังคับให้โครโมโซมเป็นศูนย์ในรอบการค้นหาถัดไปเพื่อบังคับโครงสร้างของโครงข่ายประสาทเทียมให้มีโครงสร้างเล็กลง รายละเอียดการทำงานของจีนเนติกจะอธิบายอีกครั้งในบทที่ 5 การใช้จีนเนติกในการช่วยหาโครงสร้างของ FF-MLPs ที่เหมาะสม

2.5 สรุป

การระบุตำแหน่งคือการบอกระยะของจุดทดสอบเทียบกับจุดอ้างอิง การวัดระยะสามารถทำได้ 3 วิธี คือการวัดเวลาการวัดเวลาการมาถึงของสัญญาณ การวัดมุมการมาถึงของสัญญาณ และการวัดค่าความแรงของสัญญาณ ในงานวิจัยวิทยานิพนธ์นี้เลือกใช้การวัดค่าความแรงของสัญญาณสำหรับการประมาณค่าพิกัดตำแหน่งเลือกใช้วิธีฟิงเกอร์ปริ้นท์ติ้งและเลือกอัลกอริทึมการประมาณค่าพิกัดตำแหน่งด้วยโครงข่ายประสาทเทียม

สำหรับงานวิจัยวิทยานิพนธ์นี้ใช้โครงข่ายประสาทเทียมแบบ FF-MLPs ทำงานร่วมกับ RBF เพื่อเพิ่มความถูกต้องในการระบุตำแหน่งรายละเอียดจะกล่าวเพิ่มเติมในบทที่ 3 การออกแบบระบบระบุตำแหน่งตนเอง นอกจากนี้ยังมีการปรับปรุงโครงสร้างของ FF-MLPs ด้วยจีนเนติกอัลกอริทึ่มเพื่อหาโครงสร้างดีที่สุดสำหรับการทดสอบรายละเอียดส่วนนี้จะกล่าวเพิ่มเติมในบทที่ 5 การใช้จีนเนติกในการช่วยหาโครงสร้างของ FF-MLPs ที่เหมาะสม

บทที่ 3
การออกแบบระบบระบุตำแหน่งตนเอง

3.1 บทนำ

การระบุตำแหน่งตนเองในวิทยานิพนธ์นี้ นำเสนอการคำนวณเพื่อประมาณค่าพิกัดตำแหน่งที่ต้องการทราบโดยใช้โครงข่ายประสาทเทียม โครงข่ายประสาทเทียมมีอยู่หลายประเภทด้วยกันแต่ละประเภทมีโครงสร้างและวิธีการทำงานที่แตกต่างกันไป โครงข่ายประสาทเทียมมีข้อดีที่เป็นจุดสำคัญ คือ มีความสามารถในการแก้ปัญหาต่าง ๆ ที่ซับซ้อนได้ มีการปรับตัวเองได้ อีกทั้งยังมีความทนทานต่อความบกพร่องได้ดีอีกด้วย

เนื้อหาของบทนี้ส่วนหนึ่งอธิบายถึงมาตรฐาน IEEE 802.11b และมาตรฐาน IEEE 802.11g ในส่วนของการแบ่งช่องความถี่สำหรับใช้งานโครงข่ายท้องถิ่นแบบไร้สายเพื่อนำไปสู่การกำหนดช่องความถี่สำหรับใช้ทดสอบระบบ และการออกแบบระบบระบุตำแหน่งตนเองให้สามารถปรับตัวเองได้เมื่อมีการเปลี่ยนแปลงของสภาวะแวดล้อมเล็กน้อย เช่น การย้ายตำแหน่งโต๊ะ เก้าอี้

3.2 โครงข่ายประสาทเทียม

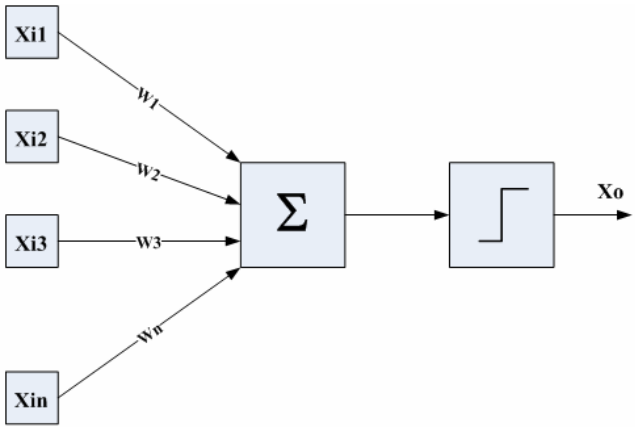
โครงข่ายประสาทเทียม คือ โมเดลทางคณิตศาสตร์ สำหรับประมวลผลสารสนเทศ เพื่อจำลองการทำงานของเครือข่ายประสาทในสมองมนุษย์ ให้มีความสามารถในการเรียนรู้การจดจำรูปแบบ (pattern recognition) และการอุปมานความรู้ (knowledge deduction) เช่นเดียวกับความสามารถที่มีในสมองมนุษย์

3.2.1 พื้นฐานของโครงข่ายประสาทเทียม

โครงข่ายประสาทเทียม เป็นการรวมกลุ่มแบบขนานของหน่วยประมวลผลย่อย ๆ ประกอบด้วยอินพุตและเอาต์พุตแต่ละตัวมีค่าถ่วงน้ำหนัก (weight) เป็นตัวกำหนดความสำคัญของอินพุตโดยนิวรอลแต่ละหน่วยจะมีค่าระดับ (threshold) เป็นตัวกำหนดว่าน้ำหนักรวมของอินพุตต้องมากเท่าใดจึงจะสามารถส่งเอาต์พุตไปยังนิวรอลตัวอื่นได้ เมื่อนำนิวรอลแต่ละหน่วยมาต่อกันให้ทำงานร่วมกันก็จะเหมือนปฏิกิริยาที่เกิดขึ้นในสมอง

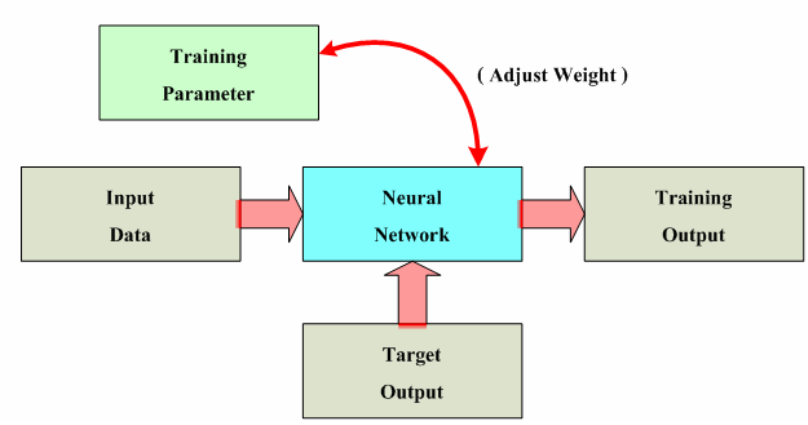
การทำงานโครงข่ายประสาทเทียม คือเมื่อมีอินพุตเข้ามาก็นำอินพุตมาคูณกับค่าถ่วงน้ำหนักของแต่ละโนด ผลที่ได้จากทุก ๆ โนดของนิวรอลจะเอามารวมกันแล้วก็เอามาเทียบกับค่าระดับที่กำหนดไว้ ถ้าผลรวมมีค่ามากกว่าค่าระดับแล้วนิวรอลก็จะส่งเอาต์พุตออกไป ค่าเอาต์พุตนี้

จะถูกส่งไปเป็นอินพุตของนิวรอลอื่น ๆ ที่เชื่อมกันในโครงข่าย แต่ถ้าค่าน้อยกว่าค่าระดับก็จะไม่เกิดการส่งต่อไปยังนิวรอลถัดไป



รูปที่ 3.1 โครงสร้างของโครงข่ายประสาทเทียม

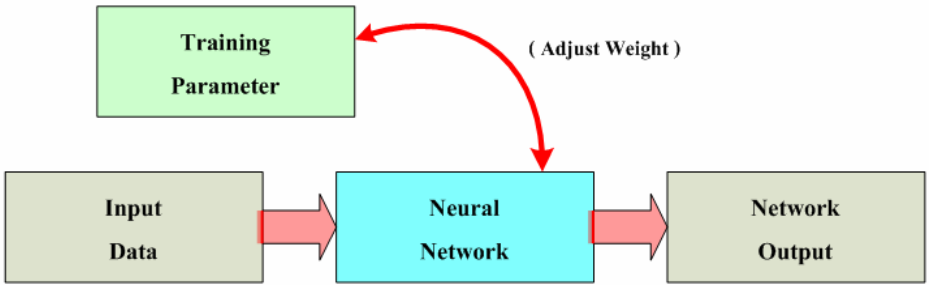
สิ่งสำคัญคือเราต้องทราบค่าถ่วงน้ำหนักและค่าระดับสำหรับระบบที่เราต้องการทดสอบเพื่อให้คอมพิวเตอร์ทำงานได้ ซึ่งค่าทั้งสองนี้เป็นค่าที่ไม่แน่นอนแต่สามารถกำหนดให้คอมพิวเตอร์ปรับค่าเหล่านั้นได้โดยการเรียนรู้ การเรียนรู้ของโครงข่ายประสาทเทียมแบ่งออกเป็น 2 แบบ คือ



รูปที่ 3.2 ขั้นตอนการเรียนรู้แบบมีการสอน

1. การเรียนรู้แบบมีการสอน (Supervise Learning) เป็นการเรียนแบบที่มีการตรวจคำตอบเพื่อให้โครงข่ายประสาทเทียมมีการปรับตัว ชุดข้อมูลที่ใช้สอน จะมีคำตอบไว้คอยตรวจดูว่าวงจรโครงข่ายประสาทเทียมให้คำตอบที่ถูกหรือไม่ ถ้าคำตอบที่ได้ไม่ถูกต้องโครงข่ายประสาทเทียมก็จะปรับตัวเองเพื่อให้ได้คำตอบที่ดีขึ้น (เปรียบเทียบกับคน เหมือนกับการสอนนักเรียนโดยมีครูผู้สอนคอยแนะนำ)

2. การเรียนรู้แบบไม่มีการสอน (Unsupervised Learning หรือ Self-Organizing) เป็นการเรียนแบบไม่มีผู้แนะนำ ไม่มีการตรวจคำตอบว่าถูกหรือผิด โครงข่ายประสาทเทียมจะจัดเรียงโครงสร้างด้วยตัวเองตามลักษณะของข้อมูล ผลลัพธ์ที่ได้โครงข่ายประสาทเทียมจะสามารถจัดหมวดหมู่ของข้อมูลได้ (เปรียบเทียบกับคน เช่น การที่เราสามารถแยกแยะพันธุ์พืช พันธุ์สัตว์ตามลักษณะรูปร่างของมันได้เองโดยไม่มีใครสอน)



รูปที่ 3.3 ขั้นตอนการเรียนรู้แบบไม่มีการสอน

การประยุกต์ใช้งานโครงข่ายประสาทเทียมได้มีมาอย่างแพร่หลาย ไม่ว่าจะเป็นทางวิศวกรรม ฟิสิกส์ จิตวิทยา การแพทย์ คณิตศาสตร์ วิทยาการคอมพิวเตอร์ เคมี หรือเศรษฐศาสตร์ เพราะว่าการใช้โครงข่ายประสาทเทียมสามารถแก้ปัญหาที่ซับซ้อนได้อย่างดี นอกจากนี้แล้วตัวโครงข่ายประสาทเทียมยังมีความทนทานต่อความบกพร่อง (fault tolerance) คือเมื่อเกิดความเสียหายที่นิวรอลในโครงข่ายระบบจะยังคงทำงานต่อไปได้อีก ลักษณะความทนทาน (robust) นี้เป็นลักษณะพื้นฐานของโครงข่ายประสาทเทียมอยู่แล้ว ทั้งนี้เนื่องจากข้อมูลภายในโครงข่ายจะถูกกระจายไปยังนิวรอลต่าง ๆ ทั่วทั้งโครงข่ายนั่นเอง อีกประการหนึ่งที่เป็นข้อดีของโครงข่ายประสาทเทียมก็คือ สามารถปรับตัวได้ (adaptive) กล่าวคือ โครงข่ายมีความสามารถโต้ตอบ (interact) และตอบสนอง (response) ต่อสภาวะแวดล้อมได้ เมื่อสภาวะแวดล้อมเปลี่ยนไปตัวโครงข่ายจะตอบสนองต่อการเปลี่ยนแปลงนั้น ๆ และทำการปรับตัวเองเพื่อให้ทำงานเข้ากับสภาวะแวดล้อมใหม่ได้ ดังนั้นโครงข่ายบางโครงข่ายสามารถออกแบบให้มีการปรับตัวในเวลาจริง (real–

time adaptation) ได้อีกด้วย การแบ่งประเภทของโครงข่ายประสาทเทียมนั้นสามารถพิจารณาได้หลายวิธี เช่น วิธีการฝึกสอน วิธีการเรียนรู้ การประยุกต์ใช้งาน ชนิดของอินพุตหรือสถาปัตยกรรมของเครือข่าย เป็นต้น

การจำแนกชนิดของโครงข่ายประสาทเทียมไม่มีการกำหนดแน่นอนไว้ตายตัว แต่ถ้าเป็นที่นิยมและแบ่งแยกได้ง่ายคือการแบ่งแยกตามวิธีการเรียนรู้ดังที่กล่าวไปแล้วข้างต้นและสามารถแสดงรายละเอียดเพื่อเปรียบเทียบโครงข่ายทั้งสองแบบแสดงในตารางที่ 3.1

ตารางที่ 3.1 ความแตกต่างระหว่างการเรียนรู้แบบมีผู้ฝึกสอนและการเรียนรู้แบบไม่มีผู้ฝึกสอน

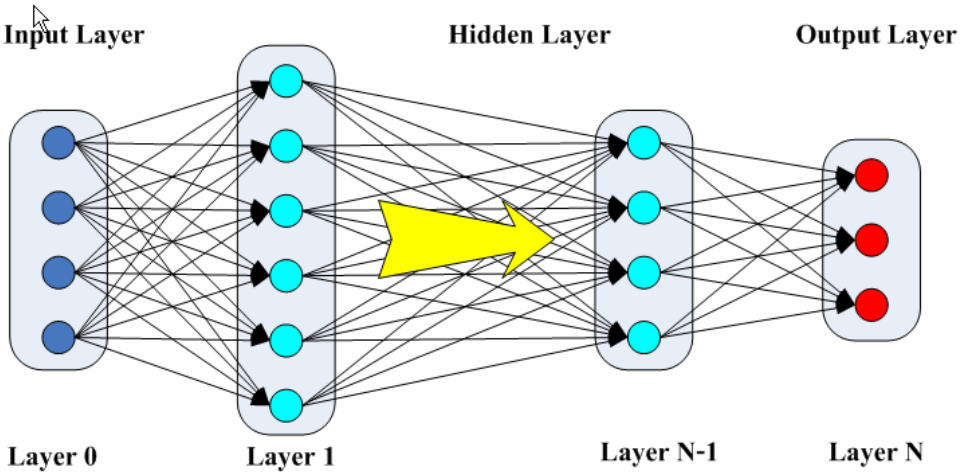
	การเรียนรู้แบบมีผู้ฝึกสอน	การเรียนรู้แบบไม่มีผู้ฝึกสอน
คุณลักษณะ	- เรียนรู้ที่จะสร้างผลลัพธ์ที่ต้องการให้ได้ตามตัวอย่างที่ได้รับ	- จัดข้อมูลอินพุตของระบบด้วยตัวเอง - ค้นหาคุณสมบัติของตัวเองจากอินพุต
ตัวอย่างกฎการเรียนรู้	- วิธีปรับแก้ค่าความผิดพลาด (error correction) โดยลดความผิดพลาดของเอาต์พุตให้น้อยที่สุดโดยเทียบกับจุดประสาท - วิธีเทียบความคล้ายโดยปรับจุดประสาทตามระดับความคล้าย	- วิธีสหสัมพันธ์ (correlation) โดยใช้กฎการเรียนรู้ของ Hebb - วิธีแข่งขัน (competitive) โดยนิวรอลที่เป็นเอาต์พุตแข่งขันกันเองจนกระทั่งได้ผู้ชนะ
ตัวอย่างโครงข่าย	- Perceptron - ADALINE(ADAptive LInear NEuron) - Back-Propagation Feed-Forward Network - Fuzzy - ARTMAP - LAPART	- PCA - ART (adaptive resonance theory) - แบบแผนผังคุณลักษณะ (feature map)
การประยุกต์ใช้งาน	- การรู้จำ (recognition) - การประมาณค่า (approximation)	- การจัดกลุ่มข้อมูล (recognition)

3.2.2 Feed-Forward Multi-Layer Perceptrons (FF-MLPs)

Feed-Forward Network คือ รูปแบบของโครงข่ายประสาทเทียมที่การไหลของข้อมูลขณะที่ประมวลผลในวงจรจะถูกส่งไปในทิศทางเดียวจากโนดอินพุตส่งต่อมาเรื่อย ๆ จนถึงโนดเอาต์พุตโดยไม่มีการย้อนกลับของข้อมูล หรือแม้แต่การไหลวนระหว่างโนดในชั้นเดียวกันก็ไม่มีการเชื่อมต่อกัน

Feed-Forward Multi-Layer Perceptrons หรือ FF-MLPs เป็นโครงสร้างอีกแบบหนึ่งของโครงข่ายประสาทเทียมโครงสร้างสามารถดังแสดงในรูปที่ 3.4 ประกอบด้วย

1. ชั้นอินพุต (Input Layer) จำนวน 1 ชั้นมีจำนวนโนดเท่ากับจำนวนอินพุตของระบบที่ต้องการให้โครงข่ายประสาทเทียมช่วยในการคำนวณหาคำตอบ



รูปที่ 3.4 โครงสร้างแบบ Feed-Forward Multi-Layer Perceptrons (FF-MLPs)

2. ชั้นซ่อนเร้น (Hidden Layer) อย่างน้อยหนึ่งชั้นและแต่ละชั้นอาจจะมีจำนวนโนดหลายโนด การกำหนดจำนวนชั้นและจำนวนโนดไม่มีข้อกำหนดแน่นอนตายตัวและไม่มีทฤษฎีรองรับแต่จะขึ้นกับความเหมาะสมของแต่ละงานที่นำไปใช้

3. ชั้นเอาต์พุต (Output Layer) จำนวน 1 ชั้นมีจำนวนโนดเท่ากับจำนวนเอาต์พุตของระบบที่ต้องการให้โครงข่ายประสาทเทียมช่วยในการคำนวณ

การคำนวณของ FF-MLPs ในแต่ละชั้นของชั้นซ่อนเร้นจะมีฟังก์ชันสำหรับคำนวณเรียกว่าฟังก์ชันการถ่ายโอน (transfer function) เมื่อได้รับค่าอินพุตจากเอาต์พุตของชั้นก่อนหน้านี้สามารถคำนวณค่าเอาต์พุตของชั้นซ่อนเร้นนี้ได้จาก

$$y_i^{(h)} = g\left(w_{i,1}^{(h)} y_1^{(h-1)} + w_{i,2}^{(h)} y_2^{(h-1)} + w_{i,3}^{(h)} y_3^{(h-1)} + \ldots + w_{i,m}^{(h)} y_m^{(h-1)} + \theta_i^{(h)}\right) \quad (3.1)$$

$$y_i^{(h)} = g\left(\sum_{j=1}^{m} w_{i,j}^{(h)} y_j^{(h-1)} + \theta_i^{(h)}\right) \quad (3.2)$$

โดยที่ $y_i^{(h)}$ คือ เอาต์พุตของโนดที่ i ของชั้น h

$w_{i,j}^{(h)}$ คือ ค่าถ่วงน้ำหนัก (weight) ที่เชื่อมต่อระหว่างโนดที่ j ของชั้น h-1 กับโนดที่ i ของชั้น h

θ_i^h คือ ค่าไบแอส (bias) ของโนดที่ i ของชั้น h

g คือ ฟังก์ชันการถ่ายโอนของโนดที่ i

การคำนวณเอาต์พุตจะเห็นได้ว่าอินพุตของชั้นปัจจุบันจะมาจากเอาต์พุตของชั้นก่อนหน้านี้ซึ่งการคำนวณลักษณะนี้เรียกว่า Feed forward network และจะทำการคำนวณเช่นนี้ไล่ไปทีละชั้นจนถึงชั้นเอาต์พุต

การฝึกสอนโครงข่ายประสาทเทียมแบบ FF-MLPs จะมีการคำนวณหาค่าความผิดพลาด (error : δ) เพื่อทำการตรวจสอบการเรียนรู้และนำค่าความผิดพลาดนี้เป็นแนวทางในการปรับค่าถ่วงน้ำหนักให้แก่โครงข่ายประสาทเทียมเพื่อให้ได้ค่าเอาต์พุตของวงจรที่ถูกต้องโดยการคำนวณค่าความผิดพลาดจะเริ่มคำนวณจากชั้นสุดท้ายของโครงข่ายประสาทเทียมก่อนจึงจะสามารถคำนวณค่าความผิดพลาดในชั้นถัดเข้ามา เรียกวิธีการคำนวณค่าความผิดพลาดเพื่อปรับค่าถ่วงน้ำหนักนี้ว่ากระบวนการเรียนรู้แบบแพร่กลับ (back-propagation algorithm)

กระบวนการเรียนรู้แบบแพร่กลับอธิบายได้ดังนี้ เริ่มจากคำนวณค่าความผิดพลาดของโนด ที่ i ในชั้นเอาต์พุต (layer N) ของโครงข่ายประสาทเทียมจาก

$$\delta_i^{(N)} = O_i - \hat{O}_i \quad (3.3)$$

เมื่อ $x = [x_1, x_2, \ldots, x_n]$ คือ อินพุตที่ป้อนให้กับระบบ

$o = [o_1, o_2, \ldots, o_n]$ คือ เอาต์พุตที่ต้องการเมื่อให้ระบบทำงาน

$\hat{o} = [\hat{o}_1, \hat{o}_2, \ldots, \hat{o}_n]$ คือ เอาต์พุตที่ได้จากการทำงานของระบบ

การหาค่าถ่วงน้ำหนัก $W_{\Phi}^{(N)}$ ที่ชั้นเอาต์พุต (layer N) คำนวณได้จาก

$$\Delta W_{ij}^{(N-1)} = \alpha \cdot \delta_i^{(N)} \cdot g'(h_i^{(N)}) \cdot y_j^{(N-1)} \tag{3.4}$$

$$h_i = \sum_k W_{ik} y_k^{(N-1)} \tag{3.5}$$

เมื่อ α คือ อัตราการเรียนรู้ (learning rate)

$\delta_i^{(N)}$ คือ ค่าความผิดพลาดที่โนด i ในชั้นเอาต์พุต (layer N)

$g'(h_i^{(N)})$ คือ อนุพันธ์ของฟังก์ชันถ่านโอนของโนดที่ i ในชั้นเอาต์พุต

$h_i^{(N)}$ คือ อินพุตของโนดที่ i ในชั้นเอาต์พุต

$y_j^{(N-1)}$ คือ เอาต์พุตจากโนดที่ j ในชั้นซ่อนเร้นที่ N-1

ค่าการปรับค่าถ่วงน้ำหนัก ($W_{ij}^{(k)}$) ในชั้นซ่อนเร้นใด ๆ หรือ layer k คำนวณได้จาก

$$\Delta W_{ij}^{(k)} = \alpha \cdot \delta_i^{(k)} \cdot y_j^{(k-1)} \tag{3.6}$$

$$\delta_i^{(k)} = g'(h_i^{(k)}) \sum_i W_{ji}^{(k-1)} \delta_i^{(k+1)} \tag{3.7}$$

เมื่อ $\delta_i^{(k)}$ คือ ค่าความผิดพลาดของโนดที่ i ในชั้นซ่อนเร้นใด ๆ หรือ layer k

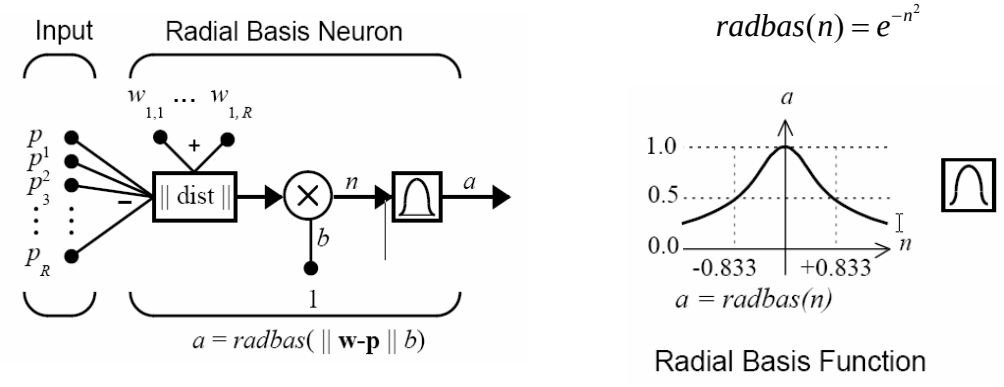
การปรับค่าถ่วงน้ำหนักจะทำในทุก ๆ รอบที่มีการป้อนอินพุตเข้าในโครงข่ายประสาทเทียม การฝึกนี้จะทำไปจนกว่าค่าความผิดพลาดของโครงข่ายข่ายประสาทเทียมมีค่าต่ำกว่าค่าที่ยอมรับได้หรือจนกว่าจำนวนรอบของการฝึกสอนครบตามจำนวนที่ตั้งไว้

3.2.3 Radius Basis Function (RBF) networks

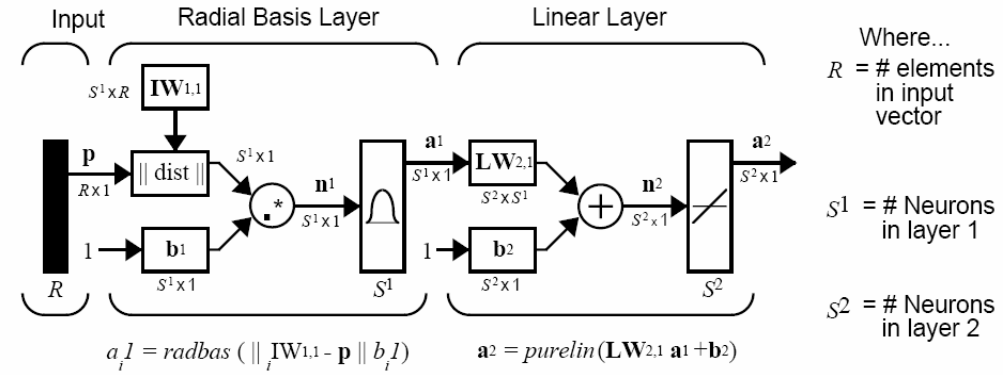
โครงสร้างของโครงข่ายประสาทเทียมแบบ RBF หรือโครงข่ายฟังก์ชันแบบฐานรัศมีแสดงดังรูปที่ 3.6 การใช้งาน RBF ส่วนใหญ่มักจะเป็นการหาความสัมพันธ์ระหว่างอินพุตไปยังเอาต์พุต (mapping) เริ่มจากโครงสร้างย่อยแบบ RBF นี้แตกต่างจาก FF-MLPs โดยการคำนวณเริ่ม

จากคำนวณระยะระหว่างค่าถ่วงน้ำหนักกับอินพุตที่ป้อนให้กับโครงข่ายแล้วคูณกับค่าไบแอสของโครงข่ายก่อนจะถูกส่งผ่านฟังก์ชันถ่ายโอน รูปที่ 3.5 จะแสดงส่วนย่อยของโครงสร้าง RBF และแสดงกราฟของฟังก์ชันถ่ายโอนแบบฐานรัศมี (radial basis)

ผลลัพธ์ที่ได้จากฟังก์ชันนี้จะมีค่ามากสุดคือ 1 หรือระยะทางของอินพุทกับค่าถ่วงน้ำหนักเท่ากับ 0 และค่าของฟังก์ชันนี้จะลดลงจนเข้าใกล้ 0 เมื่อระยะระหว่างอินพุตกับค่าถ่วงน้ำหนักมากขึ้น ส่วนค่าไบแอสจะเป็นค่าสำหรับปรับความไว (sensitivity) ของโครงข่าย



รูปที่ 3.5 ส่วนย่อยของโครงสร้าง RBF และฟังก์ชันถ่ายโอนแบบฐานรัศมี



รูปที่ 3.6 โครงสร้างของโครงข่ายประสาทเทียมแบบ RBF

โครงสร้างของ RBF ประกอบด้วยโครงข่าย 2 ชั้น ได้แก่ ชั้นซ่อนเร้นที่ใช้ฟังก์ชันถ่ายโอนแบบฐานรัศมีมีจำนวน s^1 โนด และชั้นเอาต์พุตที่ใช้ฟังก์ชันถ่านโอนแบบเชิงเส้นจำนวน S^2 โนด ดังรูปที่ 3.6 เพื่อให้เข้าใจการคำนวณของโครงข่ายประสาทเทียมแบบ RBF นี้สามารถอธิบายได้ดังนี้

$$a_i^1 = radbas(\| {}_i\mathrm{IW}^{1,1} - \mathbf{p} \| b_i^1) \tag{3.8}$$

$$\mathrm{a}^2 = purelin(\mathrm{LW}^{2,1}\ \mathrm{a}^1 + \mathrm{b}^2) \tag{3.9}$$

เมื่อ radbas(n) คือ ฟังก์ชันถ่ายโอนแบบฟังก์ชันฐานรัศมี มีค่าเท่ากับ e^{-n^2}

purelin(n) คือ ฟังก์ชันถ่ายโอนแบบเส้นตรง

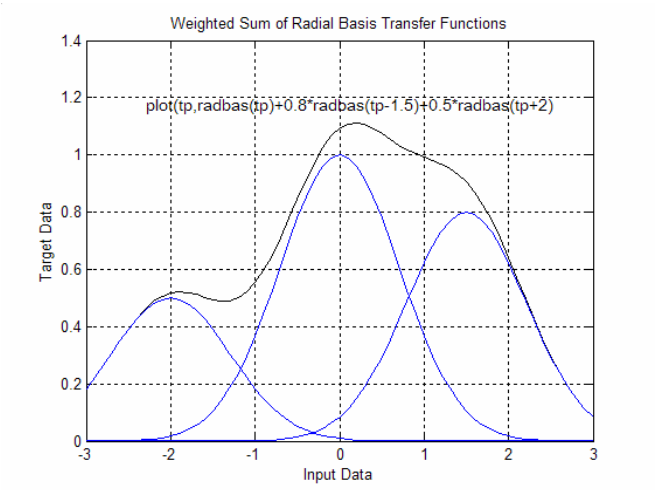
a_i^1 คือ เอาต์พุตของชั้นซ่อนเร้นโนดที่ i

$\| dist \|$ คือ เครื่องหมายระยะทางระหว่างจุด

${}_i\mathrm{IW}^{1,1}$ คือ เวกเตอร์ค่าถ่วงน้ำหนักชั้นซ่อนเร้นโนดที่ i

$\mathbf{p}$ คือ เวกเตอร์ค่าอินพุตขนาด R

b_i^1 คือ ค่าความไวของนิวรอลโนดที่ i

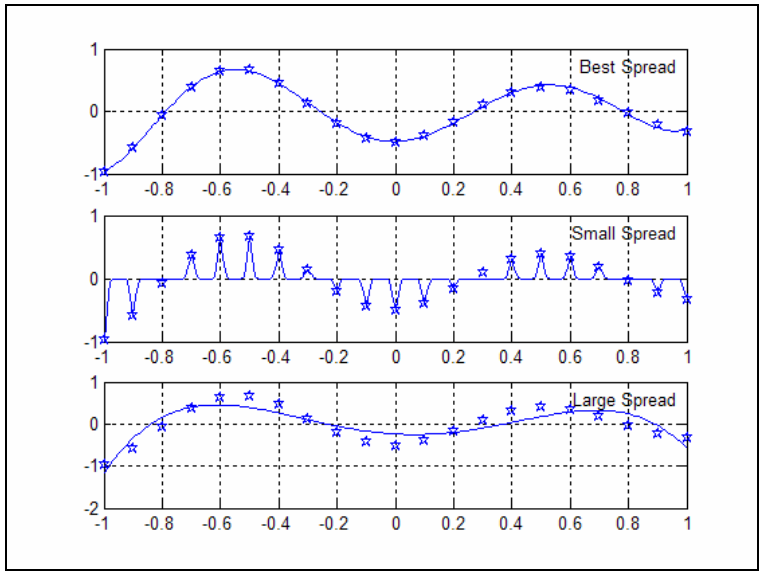


รูปที่ 3.7 การทำงานของโครงข่ายประสาทเทียม RBF สำหรับการประมาณค่าฟังก์ชัน

การทำงานของโครงข่ายประสาทเทียม RBF สำหรับการประมาณค่าฟังก์ชัน ยกตัวอย่างฟังก์ชันในรูปที่ 3.7 เส้นบนสุดเป็นฟังก์ชันที่ต้องการประมาณค่าซึ่งเกิดการใช้ฟังก์ชันฐานรัศมีหลายตัวทำการบวกกันโดยค่าตัวแปรของฟังก์ชันฐานรัศมีที่ต้องนำมาพิจารณาคือขนาดและตำแหน่งของฟังก์ชันฐานรัศมีจากรูปฟังก์ชันนี้ประกอบด้วยฟังก์ชันฐานรัศมี 3 ตัวรวมกัน ได้แก่ ฟังก์ชันฐานรัศมีตัวที่หนึ่งขนาดสูงสุดเท่ากับ 0.5 ที่ตำแหน่งศูนย์กลางเท่ากับ -2 ฟังก์ชันฐานรัศมี

ตัวที่สองขนาดสูงสุดเท่ากับ 1 ที่ตำแหน่งศูนย์กลางเท่ากับ 0 และฟังก์ชันฐานรัศมีตัวสุดท้ายขนาดสูงสุดเท่ากับ 0.8 ที่ตำแหน่งศูนย์กลางเท่ากับ 1

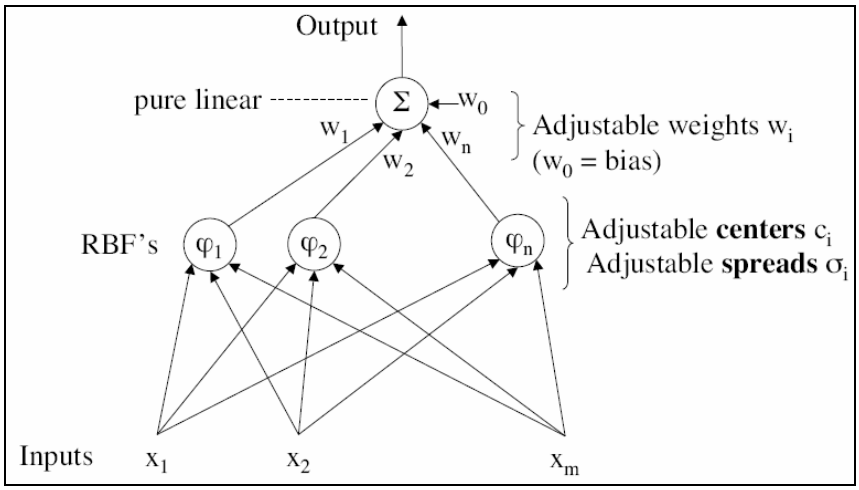
การทำงานข้างต้นยังไม่ได้ปรับขนาดการกระจาย (spread) ของฟังก์ชันฐานรัศมีการกำหนดค่าการกระจายจะมีผลต่อการประมาณค่าฟังก์ชันด้วย ยกตัวอย่างการประมาณค่าฟังก์ชันในรูปที่ 3.8 จุดรูปดาวคือค่าของฟังก์ชันที่ต้องการประมาณ เส้นคือผลการประมาณค่าด้วย RBF กรณีที่กำหนดค่าการกระจายเหมาะสมจะทำให้ฟังก์ชันที่ได้จาก RBF สามารถคำนวณค่าระหว่างจุดของฟังก์ชันที่ต้องการประมาณได้ถูกต้องดังรูป 3.8 บน ส่วนรูปกลางและรูปล่างคือการกำหนดการกระจายที่น้อยเกินไปและมากเกินไปตามลำดับ กรณีเลือกค่าการกระจายที่มากจะทำให้การประมาณค่าฟังก์ชันใช้จำนวนโนดน้อย



รูปที่ 3.8 ผลของการกระจายของฟังก์ชันฐานรัศมีกับการประมาณค่าฟังก์ชัน

ขั้นตอนการฝึกสอนโครงข่ายประสาทเทียมแบบ RBF สามารถอธิบายได้คือ เนื่องจากโครงสร้างของ RBF ประกอบด้วยชั้นซ่อนเร้นที่เป็นฟังก์ชันฐานรัศมีและชั้นเอาต์พุตที่เป็นฟังก์ชันเชิงเส้นดังรูปที่ 3.9 การฝึกสอนโครงข่ายประสาทเทียมแบบ RBF คือ การหาพารามิเตอร์ 3 ตัวได้แก่

1. ขนาดสูงสุดของฟังก์ชันฐานรัศมี สำหรับโครงข่าย RBF ทำได้โดยการปรับค่าถ่วงน้ำหนัก (W_i, W_0) ของชั้นเอาต์พุตแต่ละตัว
2. การปรับตำแหน่งศูนย์กลาง (c_i) ของฟังก์ชันฐานรัศมี
3. การปรับขนาดการกระจาย (σ_i) ของฟังก์ชันฐานรัศมี



รูปที่ 3.9 การทำงานของโครงข่ายประสาทเทียม RBF และตัวแปรที่เกี่ยวข้องกับการฝึกสอน

การปรับพารามิเตอร์ของโครงข่ายทั้งสามตัวสามารถทำได้แต่ใช้เวลามากในการฝึกสอนดังนั้นจึงกำหนดให้ขนาดของการกระจายข้อมูลเท่ากันหมดจึงเหลือเฉพาะขนาดและตำแหน่งของฟังก์ชันฐานรัศมีจึงทำให้การคำนวณง่ายขึ้น ขั้นตอนการฝึกสอนโครงข่ายประสาทเทียมแบบ RBF สามารถอธิบายเป็นขั้นตอนต่าง ๆ ได้ดังนี้

1. กำหนดขนาดการกระจาย (σ_i) ของฟังก์ชันฐานรัศมีให้เท่ากับค่าคงที่
2. กำหนดตำแหน่งศูนย์กลาง (c_i) ของฟังก์ชันฐานรัศมีเท่ากับตำแหน่งของจุดที่ต้องการประมาณค่าฟังก์ชัน
3. คำนวณเพื่อหาค่าน้ำหนัก (W_i, W_0) ของชั้นเอาต์พุต
4. คำนวณ φ_{ij} จาก

$$\varphi_{ji} = \varphi(\|x_i - x_j\|) \tag{3.10}$$

เมื่อ φ คือ ฟังก์ชันถ่ายโอนแบบฟังก์ชันฐานรัศมี

x_i, x_j คือ จุดต่าง ๆ ที่ใช้ฝึกสอนโครงข่าย

5. สร้างเมตริกซ์ Φ จากค่า φ_{ij} เรียกเมตริกซ์นี้ว่า interpolation matrix
6. เมตริกซ์ Φ มีคุณสมบัติดังนี้

$$\Phi W = d \tag{3.11}$$

เมื่อ

W คือ ค่าน้ำหนักที่ต้องการหาค่า

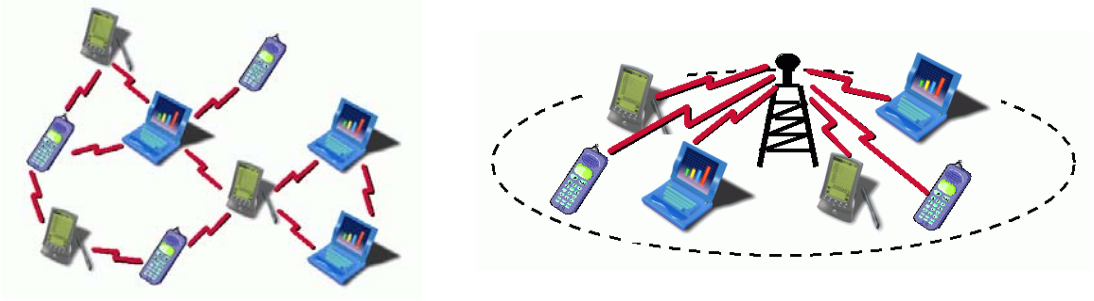
d คือ ค่าเอาต์พุตที่จุดต่าง ๆ สำหรับใช้ฝึกสอนโครงข่าย

7. คำนวณ W จาก

$$W = \Phi^{-1}d \tag{3.12}$$

3.3 โครงข่ายท้องถิ่นแบบไร้สาย

โครงข่ายท้องถิ่นแบบไร้สาย (Wireless Local Area Network : WLAN) หมายถึง โครงข่ายที่คอมพิวเตอร์ทุกเครื่องในบริเวณนั้นใช้อุปกรณ์โมเด็มไร้สายและสายอากาศในการสื่อสารกับคอมพิวเตอร์เครื่องอื่น โครงข่ายท้องถิ่นแบบไร้สายมีลักษณะการเชื่อมต่อโครงข่าย (network topology) อยู่ 2 รูปแบบ คือ แบบไม่มีโครงข่ายที่แน่นอน (ad-hoc network) หรืออาจเรียกว่าแบบจุดต่อจุด (pear-to-pear) และแบบที่มีโครงข่ายแน่นอน (infrastructure network) ลักษณะโดยทั่วไปของโครงข่ายทั้งสองประเภทได้แสดงไว้ในรูปที่ 3.10

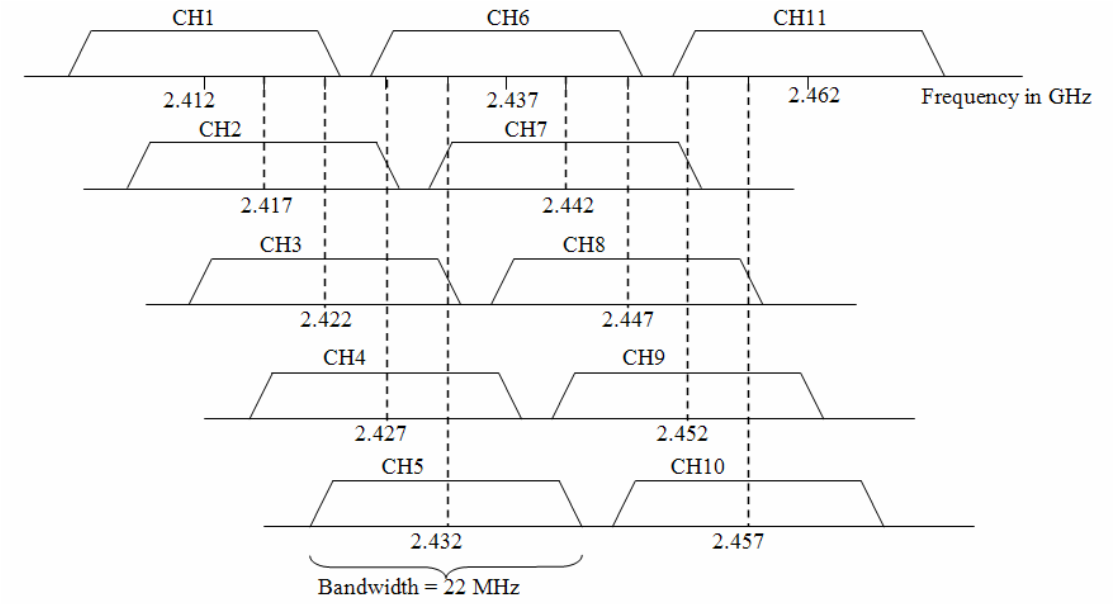


รูปที่ 3.10 โครงข่ายท้องถิ่นไร้สายแบบที่ไม่มีโครงสร้างแน่นอนและแบบที่มีโครงสร้างแน่นอน

มาตรฐาน IEEE 802.11b และ 802.11g เป็นมาตรฐานสำหรับโครงข่ายท้องถิ่นแบบไร้สายที่ได้รับความนิยมอย่างแพร่หลาย เนื่องจากความล้ำสมัยของมาตรฐานและราคาถูก ในข้อกำหนดสำหรับการทำงานของชั้นการสื่อสารกายภาพ (physical layer) มาตรฐาน IEEE 802.11b (IEEE802.11b, 1999, 2000) และมาตรฐาน IEEE 802.11g (IEEEStd802.11g, 2003) ได้กำหนดให้อุปกรณ์ทำการรับส่งข้อมูลด้วยความเร็ว 11 Mbps และ 54 Mbps ตามลำดับ สำหรับประเทศไทยนั้นอนุญาตให้ใช้งานทั้งสองมาตราฐานดังกล่าวส่งสัญญาณที่ย่านความถี่ 2.4 GHz การจัดสรร

ช่องสัญญาณที่ย่านความถี่นี้ถูกแบ่งออกเป็น 11 ช่องความถี่ (frequency channel) ตามรูปที่ 3.11 แต่ละช่องมีแบนด์วิดท์ (band width) 22 MHz ความถี่กลาง (center frequency) ของแต่ละช่องความถี่ห่างกัน 5 MHz จะสังเกตได้ว่า แบนด์วิดท์มีค่ามากกว่าความห่างระหว่างความถี่กลางของแต่ละช่องความถี่ที่อยู่ติดกัน (adjacent channel) ทำให้เกิดการซ้อนทับความถี่ของช่องความถี่ขึ้น แต่การซ้อนทับนี้จะไม่เกิดขึ้นเมื่อช่องความถี่ห่างกันมากกว่า 5 ช่อง นั่นคือจากทั้งหมด 11 ช่องจะมีเพียงแค่ 3 ช่องเท่านั้นที่ไม่มีการซ้อนทับความถี่ของช่องความถี่ ซึ่งได้แก่ ช่องที่ 1, 6, และ 11

สำหรับการทดสอบระบบได้มีการกำหนดให้มีจุดอ้างอิง 4 จุดครอบคลุมบริเวณทดสอบจากการแบ่งช่องความถี่ที่มี 11 ช่อง สามารถเลือกช่องความถี่สำหรับจุดเข้าถึงทั้ง 4 ตัวที่ความถี่ต่าง ๆ ดังนี้ CH1, CH4, CH8 และ CH11 หรือ 2.412MHz, 2.427MHz, 2.447MHz และ 2.462MHz ตามลำดับโดยกำหนดให้ระยะห่างกันมากสุดที่เป็นไปได้เพื่อลดการเกิดการแทรกสอดของคลื่น



รูปที่ 3.11 การแบ่งช่องความถี่สำหรับ IEEE 802.11b และ 802.11g

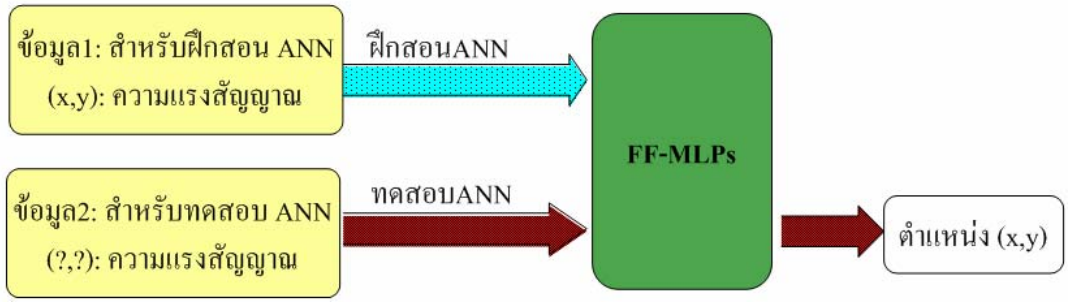
3.4 การใช้โครงข่ายประสาทเทียมในการระบุตำแหน่งตนเอง

ในวิทยานิพนธ์นี้มุ่งศึกษาเพื่อหาโครงสร้างของโครงข่ายประสาทเทียมที่เหมาะสมสำหรับงานการประมาณค่าเพื่อระบุตำแหน่งโดยโครงสร้างที่นำเสนอแบ่งเป็น 2 โครงสร้าง ได้แก่

1. โครงสร้างแบบ FF-MLPs

โครงสร้างแบบ FF-MLPs นี้เป็นโครงสร้างที่ง่ายและมีการใ ช้งานแพร่หลายโดยทั่วไป การใช้งานโครงสร้างนี้จะมีข้อมูลชุดที่ 1 สำหรับฝึกสอนโครงข่ายประสาทเทียม และข้อมูลชุดที่ 2 สำหรับทดสอบโครงข่ายประสาทเทียม ข้อมูลอินพุตคือความแรงของสัญญาณจากจุดเข้าถึงข้อมูล เอาต์พุตที่ต้องการคือตำแหน่ง (x, y)

โครงสร้างแบบ FF-MLPs แสดงดังรูปที่ 3.12 สามารถเพิ่มความถูกต้องในการระบุตำแหน่งด้วยการปรับโครงสร้างภายในของ FF-MLPs ให้เหมาะสม



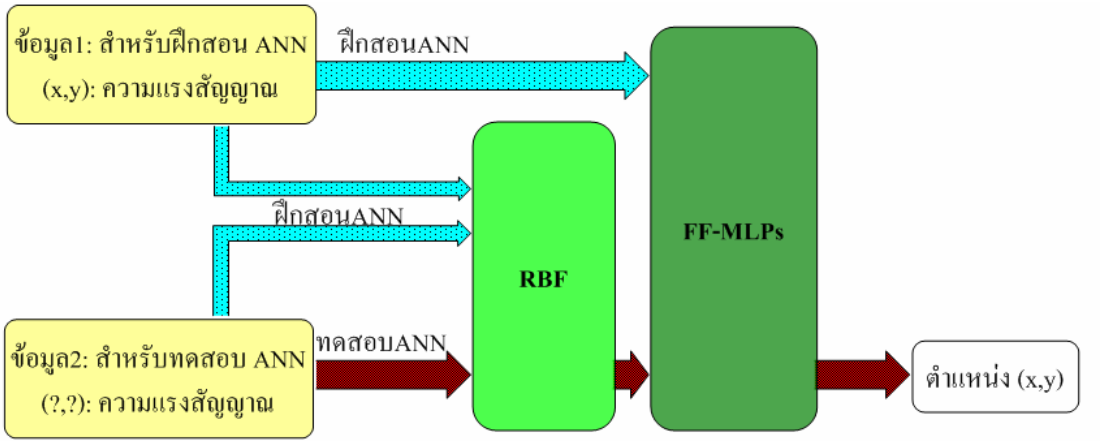
รูปที่ 3.12 แบบจำลองระบบระบุตำแหน่งโครงสร้างที่ 1 แบบ FF-MLPs

2. โครงสร้างแบบ FF-MLPs + RBF

โครงสร้างแบบ FF-MLPs + RBF นี้เป็นโครงสร้างที่นำเสนอขึ้นใหม่เพื่อให้การระบุตำแหน่งถูกต้องมากขึ้นแนวคิดนี้คือสร้างแบบจำลองของห้องทดสอบด้วยข้อมูลชุดที่ 1 และได้แบบจำลองห้องทดสอบคือ FF-MLPs ขั้นตอนการทดสอบจะมีการนำ RBF มาใช้เพื่อสร้างความสัมพันธ์ระหว่างข้อมูลฝึกสอนและข้อมูลทดสอบ บนพื้นที่ทดสอบบางจุดหรือกล่าวได้ว่ามีโครงสร้าง RBF เพื่อปรับปรุงสัญญาณทดสอบให้เหมาะสมตามสภาวะแวดล้อมที่เปลี่ยนแปลงการทดสอบด้วยการใช้ข้อมูลทดสอบป้อนให้ RBF ก่อนป้อนให้ FF-MLPs ผลลัพท์ที่ได้ คือตำแหน่ง (x, y) ยกตัวอย่างเหตุผลที่มีการใช้ RBF ปรับปรุงสัญญาณก่อนทดสอบ เช่น กรณีขณะทำการวัดสัญญาณทดสอบมีการเพิ่มกำลังส่งของจุดเข้าถึงขึ้นจากเดิม ดังนั้นจึงต้องมีส่วนของการปรับปรุงเพื่อชดเชยความแรงสัญญาณส่วนที่เพิ่มมาให้คล้ายกับข้อมูลที่ใช้ฝึกสอนโครงข่ายประสาทเทียม

โครงสร้างแบบ FF-MLPs แสดงดังรูปที่ 3.13 สามารถเพิ่มความถูกต้องในการระบุตำแหน่งด้วยการปรับโครงสร้างภายในของ FF-MLPs ให้เหมาะสมและบอกจุดอ้างอิงใน RBF ให้มากจุดขึ้น โดยรวมระบบประกอบด้วยอินพุต คือค่าความแรงสัญญาณที่อ่านได้จากจุดเข้าถึงของ

ระบบโครงข่ายท้องถิ่นไร้สายเก็บข้อมูลจากจุดเข้าถึงจำนวน 4 จุดที่วางอยู่ตำแหน่งต่าง ๆ ของห้องสำหรับฝึกสอนและทดสอบระบบ เอาต์พุตที่ต้องการจากระบบคือตำแหน่ง (x, y)



รูปที่ 3.13 แบบจำลองระบบระบุตำแหน่งโครงสร้างที่ 2 แบบ FF-MLPs + RBF

3.5 ขั้นตอนการทดสอบระบบระบุตำแหน่งในวิทยานิพนธ์นี้

ในวิทยานิพนธ์นี้จะพิจารณาแบบจำลองระบบโดยใช้ห้องปฏิบัติการไมโครโพรเซสเซอร์และห้องปฏิบัติการวงจรและอุปกรณ์ อาคารเครื่องมือ 3 (F3) มหาวิทยาลัยเทคโนโลยีสุรนารี กำหนดพื้นที่ทดสอบขนาด 20 เมตร x 25 เมตร และจุดทดสอบที่ระยะทุก ๆ 1 เมตรจะได้จุดสำหรับทดสอบจำนวน 21 x 26 = 546 จุด

ข้อสมมุติฐานสำหรับการทดสอบมีดังนี้

1. สิ่งแวดล้อมของระบบโดยรวมต้องอยู่ในตำแหน่งที่นิ่งหรือมีการเคลื่อนที่น้อยที่สุด
2. ไม่พิจารณาผลของอุณหภูมิและความชื้นต่อการทำงานของระบบ
3. กำหนดให้จุดเข้าถึงที่ใช้ทดสอบคุณสมบัติเดียวกันทั้งหมดเนื่องจากการเก็บข้อมูลเพื่อทำการฝึกสอนโครงข่ายและการเก็บข้อมูลเพื่อทดสอบจะคงตำแหน่งเดิมของจุดเข้าถึงไว้ไม่มีการเปลี่ยนแปลงตำแหน่ง
4. ไม่มีการควบคุมกำลัง (power control) ของจุดเข้าถึงในขณะใช้งานในระบบการประมาณค่าเพื่อระบุตำแหน่ง

ขั้นตอนการทดสอบโดยรวมมีดังนี้

1. กำหนดช่องความถี่สำหรับจุดเข้าถึงทั้ง 4 ตัวที่ความถี่ต่าง ๆ ดังนี้ CH1, CH4, CH8 และ CH11 หรือความถี่ 2.412MHz, 2.427MHz, 2.447MHz และ 2.462MHz ตามลำดับโดยกำหนดให้ระยะห่างกันมากสุดที่เป็นไปได้เพื่อลดการเกิดการแทรกสอดของคลื่น
2. สร้างโปรแกรมเพื่อเก็บข้อมูลความแรงสัญญาณจากจุดเข้าถึง โดยเปรียบเทียบความไวของการอ่านค่าความแรงสัญญาณกับโปรแกรมสำเร็จรูป “NetStumbler v0.4.0”
3. เก็บข้อมูลความแรงสัญญาณสำหรับทำการฝึกโครงข่ายประสาทเทียมและเก็บข้อมูลสำหรับทดสอบการทำงานของโครงข่ายประสาทเทียม
4. ทำการออกแบบโครงข่ายประสาทเทียมแบบ FF-MLPs และโครงข่ายประสาทเทียมแบบ FF-MLPs + RBF โครงข่ายทั้งสองให้มีอินพุตเป็นค่าความแรงของสัญญาณที่รับได้ จากจุดเข้าถึงที่ 1,2,3 และ 4 ตามลำดับ (SS_1, SS_2, SS_3 และ SS_4) และเอาต์พุตที่ให้ค่าผลลัพธ์เป็นค่าตำแหน่ง (x, y)
5. ทำการฝึกสอนโครงข่ายประสาทเทียมที่ออกแบบขึ้นด้วยข้อมูลชุดที่ 1
6. ทำการทดสอบโครงข่ายประสาทเทียมโดยใช้ชุดข้อมูลชุดที่ 2
7. วิเคราะห์ผลลัพธ์โดยเขียนเป็นกราฟ หรือตารางของค่าความผิดพลาดของผลลัพธ์ที่โครงข่ายประสาทเทียมประมาณค่าเปรียบเทียบกับค่าเป้าหมายที่ป้อนเข้าสู่ระบบ
8. ใช้จีนเนติกอัลกอริทึมเพื่อปรับปรุงโครงสร้าง FF-MLPs ให้ดีที่สุดสำหรับการทดสอบ

3.6 สรุป

ในบทนี้ได้กล่าวถึงพื้นฐานของโครงข่ายประสาทเทียมที่ใช้เป็นแบบจำลองการทำงานของระบบระบุตำแหน่งตนเอง ซึ่งโครงข่ายประสาทเทียมที่ใช้เป็นแบบที่ต้องมีการฝึกสอน เนื่องจากเป็นงานที่เกี่ยวข้องกับการประมาณค่า โครงข่ายประสาทเทียมที่เลือกใช้คือ FF-MLPs ที่มีความสามารถในการสร้างอาณาเขตการตัดสินข้อมูลที่ซับซ้อน (complex decision region) มากขึ้นได้และใช้ RBF ที่มีความสามารถในการหาความสัมพันธ์ (mapping) ของข้อมูลระหว่างอินพุตไปยังเอาต์พุต และมีโครงสร้างที่ซับซ้อนน้อยกว่า FF-MLPs ข้อดีในการใช้โครงข่ายประสาทเทียมในการประมาณค่าเพื่อระบุตำแหน่งตนเองที่สำคัญคือ ความสามารถในการประมาณค่าชนิดของข้อมูลที่มีความไม่เป็นเชิงเส้นได้ดีกว่าการใช้คณิตศาสตร์โดยตรง

ระบบโครงข่ายประสาทเทียมที่ทดสอบ กำหนดให้มีข้อมูลอินพุตคือความแรงของสัญญาณจากจุดเข้าถึง 4 ตำแหน่งที่วางครอบคลุมพื้นที่ทดสอบ กำหนดช่องความถี่สำหรับการสื่อสารของจุดเข้าถึงให้ห่างกันมากสุดเพื่อลดการรบกวนกันระหว่างช่องสัญญาณ ข้อมูลเอาต์พุตที่ต้องการคือค่า

พิกัดตำแหน่ง กำหนดโครงสร้างที่ใช้สำหรับการทดสอบมี 2 รูปแบบคือ โครงสร้างแบบ FF-MLPs และโครงสร้างแบบ FF-MLPs + RBF ขั้นตอนการทดสอบจะปรับองค์ประกอบของโครงสร้างทั้ง 2 รูปแบบเพื่อให้ผลการระบุพิกัดตำแหน่งได้ถูกต้องแม่นยำมากขึ้น

บทที่ 4
การทดสอบระบบระบุตำแหน่งตนเอง

4.1 บทนำ

การทดสอบระบบระบุตำแหน่งนี้ เป้าหมายสำคัญ คือเมื่อทดสอบการระบุตำแหน่งตนเองแล้วภายในพื้นที่ทดสอบขนาด 20 x 20 ตารางเมตรผลการระบุตำแหน่งต้องมีความถูกต้องในระยะ 1 เมตร จำนวนอย่างน้อย 80 เปอร์เซ็นต์จากจุดทดสอบทั้งหมด

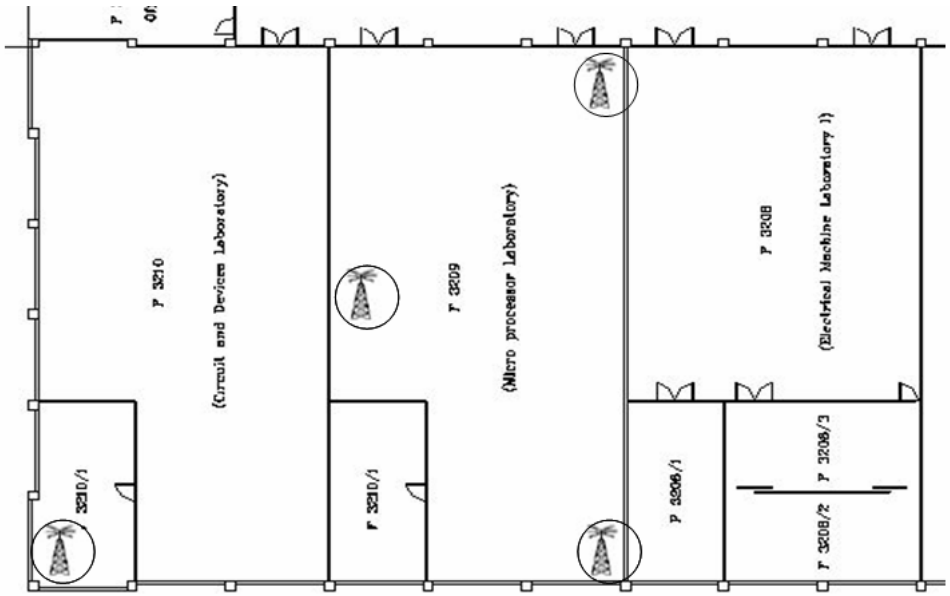
การทดสอบเริ่มจากการตั้งจุดเข้าถึง 4 จุดครอบคลุมพื้นที่ทดสอบขนาด 20 x 25 ตารางเมตร กำหนดจุดทดสอบที่ระยะทุก ๆ 1 เมตรจะได้จุดทดสอบทั้งหมด 21 x 26 = 546 จุด เก็บข้อมูลที่จุดทดสอบต่าง ๆ จำนวน 100 ครั้ง จำนวน 2 ชุดเพื่อสำหรับฝึกสอนและเพื่อสำหรับทดสอบโครงข่ายประสาทเทียม

ฝึกสอนและทดสอบโครงข่ายประสาทเทียมแบบที่ 1: FF-MLPs และ ฝึกสอนและทดสอบโครงข่ายประสาทเทียมแบบที่ 2: FF-MLPs + RBF โดยมีวัตถุประสงค์เพื่อหาองค์ประกอบของโครงสร้างที่เหมาะสมที่ให้การระบุตำแหน่งถูกต้องตามเป้าหมายที่ตั้งไว้และให้โครงข่ายประสาทเทียมมีขนาดเล็กสุด องค์ประกอบโครงสร้างของโครงข่ายประสาท เช่น จำนวนชั้นซ่อนเร้น จำนวนโนดในแต่ละชั้นเป็นต้น รายละเอียดของขั้นตอนการทดสอบ และผลการทดสอบจะนำเสนอต่อไป

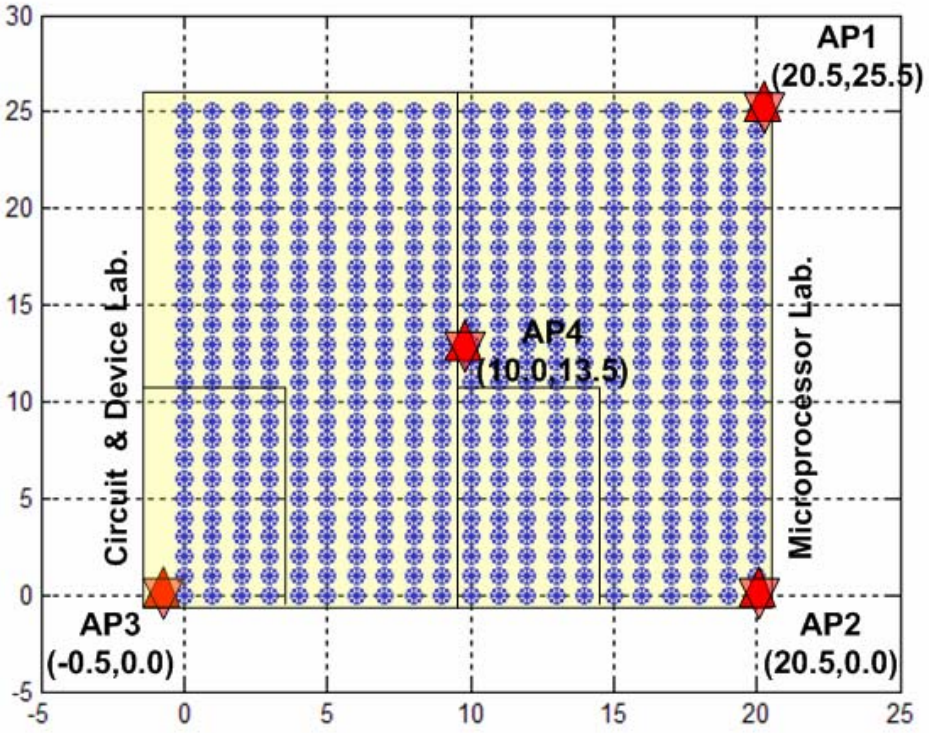
4.2 การกำหนดพื้นที่สำหรับทดสอบ

ในวิทยานิพนธ์นี้จะพิจารณาแบบจำลองระบบโดยใช้ห้องปฏิบัติการไมโครโพรเซสเซอร์และห้องปฏิบัติการวงจรและอุปกรณ์ อาคารเครื่องมือ 3 (F3) มหาวิทยาลัยเทคโนโลยีสุรนารี ดังรูปที่ 4.1 แผนผังของพื้นที่ที่ใช้สำหรับทดสอบ กำหนดพื้นที่ทดสอบขนาด 20 เมตร x 25 เมตร และกำหนดจุดทดสอบที่ระยะทุก ๆ 1 เมตรจะได้จุดทดสอบจำนวน 21 x 26 = 546 จุด ดังรูปที่ 4.2 แสดงจุดต่าง ๆ ที่ทำการทดสอบเพื่อระบุตำแหน่ง

กำหนดให้วางจุดเข้าถึง 4 จุดที่ตำแหน่งวางที่ตำแหน่ง(x, y) ต่าง ๆ ดังต่อไปนี้ AP1 (20.5, 25.5), AP2 (20.5, 0.0), AP3 (-0.5, 0.0) และ AP4 (10.0, 13.5)



รูปที่ 4.1 แผนผังของพื้นที่ที่ใช้สำหรับทดสอบ



รูปที่ 4.2 จุดต่าง ๆ ที่ทำการทดสอบเพื่อระบุตำแหน่ง

4.3 การเก็บข้อมูลความแรงของสัญญาณจากจุดเข้าถึง

เนื่องจากข้อมูลที่ต้องการเก็บมีจำนวนหลายตำแหน่งและแต่ละตำแหน่งก็เก็บตัวอย่างหลายครั้ง ดังนั้นจึงต้องมีโปรแกรมสำหรับเก็บข้อมูลโดยโปรแกรมสำหรับเก็บข้อมูลควรมีความสามารถดังต่อไปนี้

- บันทึกข้อมูลในรูปแบบของไฟล์ข้อมูลเพื่อง่ายต่อการนำข้อมูลไปใช้งาน
- ข้อมูลที่บันทึกต้องมีค่าความแรงของสัญญาณของจุดเข้าถึงทุกตัวที่ปรากฏในบริเวณที่เก็บข้อมูล
- ต้องสามารถใส่ค่าตำแหน่ง (x, y) เพื่ออ้างอิงตำแหน่งในไฟล์ข้อมูลเพื่อความง่ายในการจัดการข้อมูล

ในวิทยานิพนธ์นี้ใช้วิธีการเก็บข้อมูลด้วยโปรแกรม NetStumbler V0.4.0 ซึ่งเป็นโปรแกรมสแกนหาจุดเข้าถึงของเครือข่ายแลนไร้สาย และโปรแกรม Site Survey ซึ่งเป็นโปรแกรมที่พัฒนาขึ้นมาใหม่

4.3.1 โปรแกรม NetStumbler

โปรแกรม NetStumbler เป็นโปรแกรมที่ทำงานบนระบบปฏิบัติการวินโดว์ใช้สำหรับตรวจการทำงานของระบบเครือข่ายแลนไร้สายที่ใช้มาตรฐาน 802.11b, 802.11a และ 802.11g โปรแกรมนี้มีความสามารถต่าง ๆ ดังนี้

- ตรวจสอบการทำงานของระบบโครงข่ายแลนไร้สายที่ใช้งาน
- ตรวจหาตำแหน่งของจุดอับสัญญาณของระบบโครงข่ายแลนไร้สายที่ติดตั้งขึ้น
- ตรวจจับการทำงานของอุปกรณ์โครงข่ายอื่นที่เข้ามารบกวนการทำงานของโครงข่ายที่ ติดตั้งใช้งาน
- ตรวจจับจุดเข้าถึงที่ไม่ได้รับอนุญาตให้ติดตั้งในพื้นที่
- ช่วยในการหาทิศทางในการติดตั้งสายอากาศแบบทิศทางสำหรับเชื่อมต่อโครงข่ายแลนไร้สาย
- ระบุตำแหน่งที่ทำการวัดการวัดโดยใช้เครื่องรับสัญญาณดาวเทียม GPS (Global Positioning System) ต่อพ่วงเข้ากับคอมพิวเตอร์ที่ทำการวัด

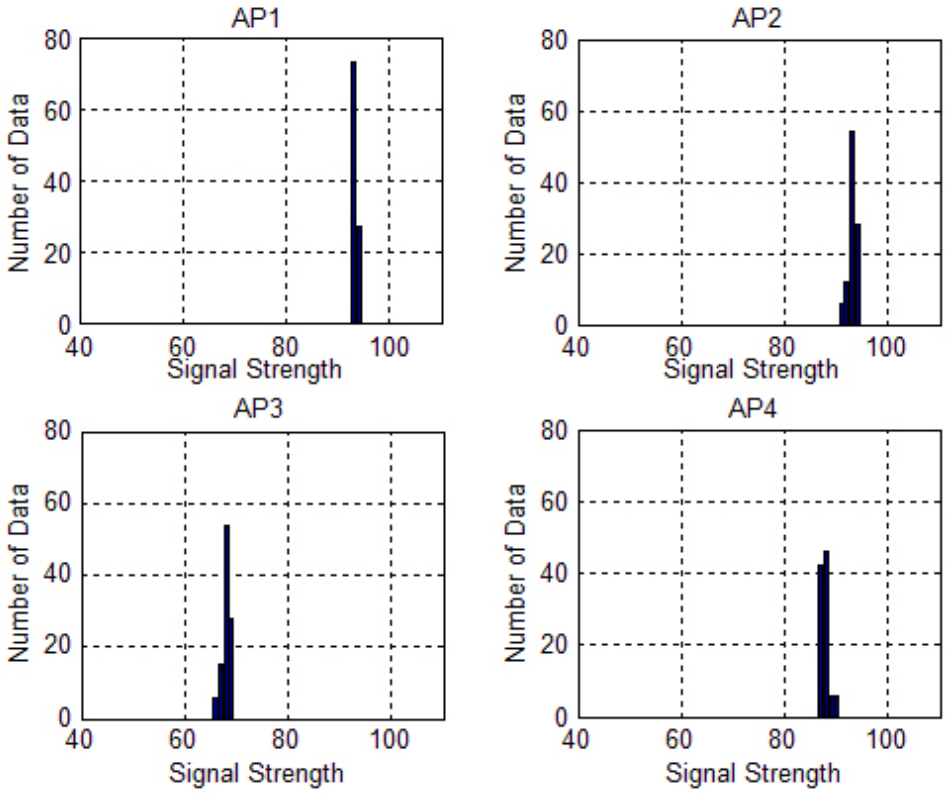
ข้อมูลที่เก็บได้ด้วยโปรแกรมนี้ได้ประกอบด้วยตำแหน่งซึ่งบอกเป็นเส้นรุ้ง (latitude) เส้นแวง (longitude) ที่อ่านจากเครื่องรับสัญญาณดาวเทียมจีพีเอส (GPS) ชื่อของจุดเข้าถึงที่สามารถวัดค่าสัญญาณได้ บอกเป็นคุณภาพของสัญญาณในรูปของอัตราส่วนของสัญญาณต่อสัญญาณรบกวน (Signal to Noise Ratio: SNR) ความแรงของสัญญาญที่รับได้และขนาดของสัญญาณรบกวน

ผลการเก็บข้อมูลโดยใช้โปรแกรมนี้ยังมีข้อบกพร่องอยู่สองประการ ประการแรกคือขาดข้อมูลการระบุตำแหน่ง (x, y) เพื่อจัดเก็บลงบนไฟล์ข้อมูลได้และประการที่สองจากการทดสอบโดยย้ายตำแหน่งการรับสัญญาณไปยังจุดใหม่ เวลาที่ใช้ในการเปลี่ยนแปลงค่าค่าสัญญาณมากกว่าหนึ่งนาทีซึ่งถือว่าช้าไม่เหมาะสำหรับเก็บข้อมูลหลายจุด

4.3.2 โปรแกรม Site Survey

จากการทำงานของโปรแกรม D-Link AirPlus ซึ่งเป็นโปรแกรมขับ (driver program) ที่ให้มาพร้อมกับการ์ดเชื่อมต่อไร้สาย เมื่อโปรแกรม D-Link AirPlus เริ่มทำงานจะทำการสแกนหาจุดเข้าถึงที่สามารถรับสัญญาณได้แล้วแสดงค่าความแรงสัญญาณของจุดเข้าถึงต่าง ๆ

โปรแกรม Site Survey เป็นโปรแกรมที่พัฒนาขึ้นด้วย Virtual Basic 6.0 วิธีการทำงานของโปรแกรมนี้ คือการอ่านค่าความแรงสัญญาณจากหน้าต่างของโปรแกรม D-Link AirPlus แล้วทำการบันทึกเป็นไฟล์ข้อมูล ข้อมูลที่บันทึกลงบนไฟล์ข้อมูลประกอบด้วยตำแหน่ง (x, y) จากการป้อนค่าโดยผู้ใช้งาน ชื่อของจุดเข้าถึง และความแรงสัญญาณของจุดเข้าถึงทั้งหมดที่โปรแกรม D-Link AirPlus สามารถจับสัญญาณได้



รูปที่ 4.3 การกระจายของสัญญาณที่วัดได้ที่ตำแหน่ง (20, 5)

ผลการทดสอบโดยกำหนดจุดที่ตำแหน่ง (20, 5) ทำการวัดสัญญาณ 100 ครั้ง แล้วแสดงผลด้วยกราฟแท่งดังรูป 4.3 เพื่อดูการกระจายของสัญญาณที่วัดได้ แกนตั้งคือจำนวนข้อมูล แกนนอนคือค่าเปอร์เซ็นต์ความแรงของสัญญาณที่วัดได้ ตารางที่ 4.1 แสดงค่าการวัดความแรงสัญญาณที่ตำแหน่ง (20, 5)

ในวิทยานิพนธ์นี้จะทำการวัดจุดทดสอบละ 100 ครั้งแล้วนำค่าเฉลี่ยเป็นค่าความแรงของสัญญาณสำหรับการฝึกสอนและทดสอบโครงข่ายประสาทเทียมต่อไป

ตารางที่ 4.1 ข้อมูลค่าการวัดความแรงสัญญาณที่ตำแหน่ง (20, 5)

จุดเข้าถึง	จำนวนครั้ง	ความแรงของสัญญาณที่วัดได้(เปอร์เซ็นต์)			
		ค่าต่ำสุด	ค่าเฉลี่ย	ค่าสูงสุด	ค่าเบี่ยงเบน
AP1	100	92	94.04	95	0.803
AP2	100	89	93.23	94	0.821
AP3	100	73	71.13	76	0.818
AP4	100	84	84.27	85	0.446

4.4 การหาโครงสร้างของโครงข่ายประสาทเทียมแบบ FF-MLPs

เนื้อหาส่วนนี้จะกล่าวถึงการออกแบบโครงข่ายประสาทเทียมโดยวิธีการค้นหาแบบวนรอบจากโครงสร้างที่เล็กสุดก่อน โครงสร้างที่เล็กสุดประกอบด้วยชั้นซ่อนเร้น 1 ชั้นและมีจำนวนโนด 1 ตัวแล้วเพิ่มขนาดของโครงสร้างขึ้น เนื้อหาแบ่งออกเป็น 2 ส่วน คือ ขั้นตอนการทดสอบเพื่อหาโครงข่ายประสาทเทียมที่เหมาะสมสำหรับเป็นแบบจำลองพื้นที่ และผลการทดสอบ

4.4.1 การทดสอบเพื่อหาโครงสร้างของ FF-MLPs

ขั้นตอนการทดสอบเพื่อหาโครงสร้างของ FF-MLPs ที่เหมาะสมในการเป็นแบบจำลองของพื้นที่ มีขั้นตอนการทำงานดังนี้

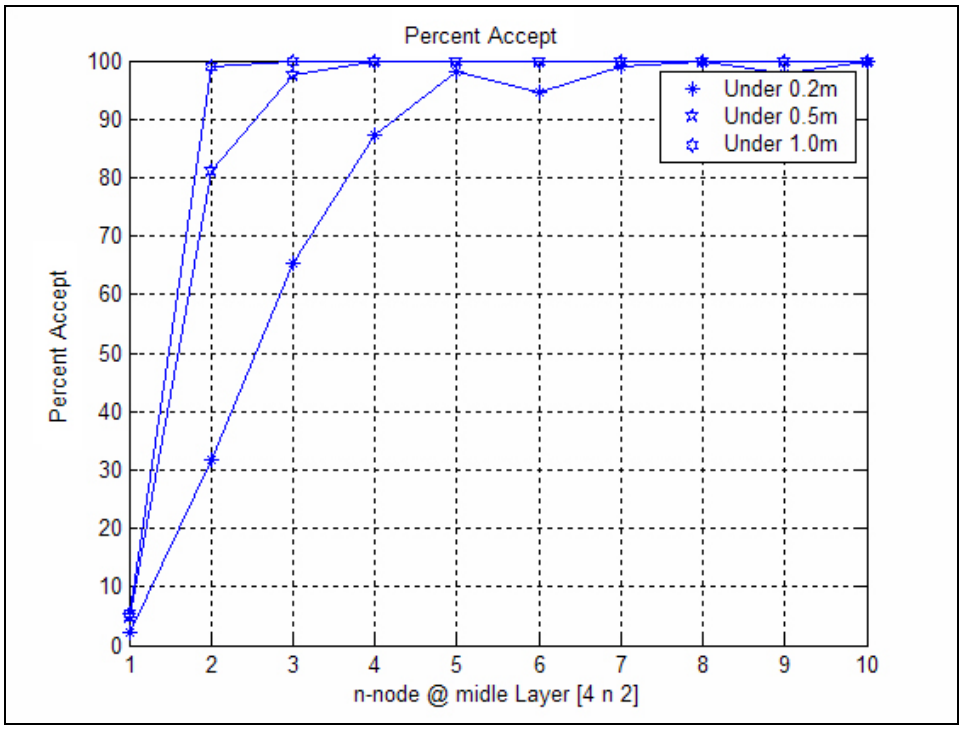
ตอนที่ 1 : ทดสอบโครงสร้างโครงข่ายประสาทเทียมแบบมีชั้นซ่อนเร้น 1 ชั้น

1. ทดสอบโครงสร้าง FF-MLPs แบบ [4 n 2] {tansig, tansig, purelin} ซึ่งเป็นโครงสร้างที่มีชั้นซ่อนเร้น 1 ชั้น กำหนดฟังก์ชันถ่ายโอนของชั้นอินพุต ชั้นซ่อนเร้น และชั้นเอาต์พุต คือ tansig , tansig และ purelin ตามลำดับ
2. กำหนดตัวแปร n คือจำนวนโนดในชั้นซ่อนเร้น มีค่าตั้งแต่ 1 ถึง 10

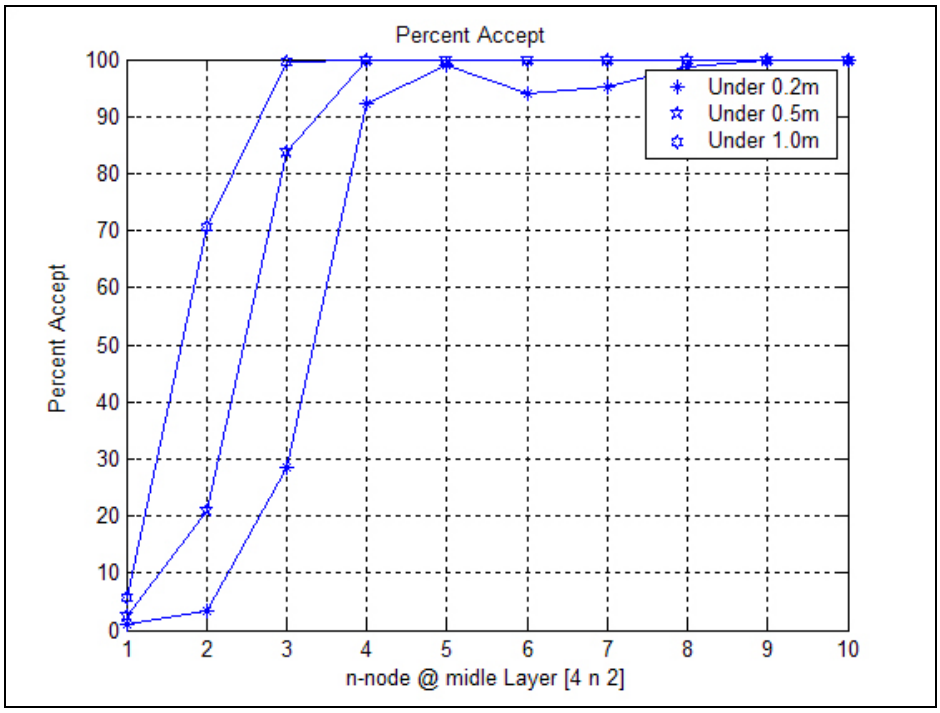
3. สร้างโครงข่ายประสาทเทียมตามข้อ 1 ฝึกสอนและทดสอบด้วยข้อมูลชุดที่ 1 โดยมีจำนวนรอบการฝึกสอน 1,000 รอบ
4. คำนวนหาจำนวนจุดที่ให้ผลการทดสอบคลาดเคลื่อนจากตำแหน่งจริงที่ระยะต่ำกว่า 0.2 เมตร, 0.5 เมตร และ 1.0 เมตร คำนวณเป็นค่าเปอร์เซ็นต์จุดที่ยอมรับ
5. ทำซ้ำขั้นตอนที่ 3,4 จำนวน 10 ครั้ง เลือกรอบที่ให้ค่าเปอร์เซ็นต์จุดที่ยอมรับดีที่สุดเป็นตัวแทนของโครงสร้างในข้อ 3
6. ทำซ้ำขั้นตอนที่ 3, 4 และ 5 จนครบโครงสร้างที่มีโนด 10 โนด
7. แสดงผลการทดสอบในรูปของกราฟ โดยให้แกนนอนเป็นจำนวนโนดในชั้นซ่อนเร้น ตั้งแต่ 1 ถึง 10 แกนตั้งเป็นเปอร์เซ็นต์จุดทดสอบตามค่าคลาดเคลื่อนที่ระยะต่ำกว่า 0.2 เมตร, 0.5 เมตร และ 1.0 เมตร
8. ทำซ้ำขั้นตอนที่ 2 ถึง 7 อีกครั้ง และแสดงผลดังรูปที่ 4.4 และรูปที่ 4.5

ตอนที่ 2 : ทดสอบโครงสร้างโครงข่ายประสาทเทียมแบบมีชั้นซ่อนเร้น 2 ชั้น

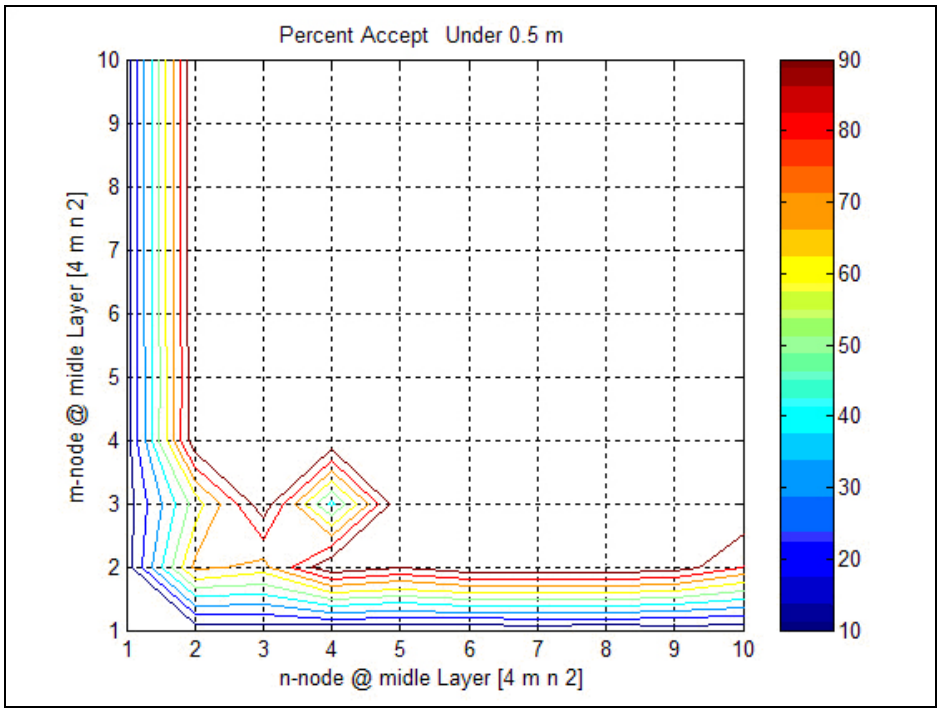
1. ทดสอบโครงสร้าง FF-MLPs แบบ [4 m n 2] { tansig, tansig, tansig, purelin } ซึ่งเป็นโครงสร้างที่มีชั้นซ่อนเร้น 2 ชั้น กำหนดฟังก์ชันถ่ายโอนของชั้นอินพุต ชั้นซ่อนเร้น1 ชั้นซ่อนเร้น2 และชั้นเอาต์พุต คือ tansig, tansig, tansig และ purelin ตามลำดับ
2. กำหนดตัวแปร m คือจำนวนโนดในชั้นซ่อนเร้นที่ 1 มีค่าตั้งแต่ 1 ถึง 10
3. กำหนดตัวแปร n คือจำนวนโนดในชั้นซ่อนเร้นที่ 2 มีค่าตั้งแต่ 1 ถึง 10
4. สร้างโครงข่ายประสาทเทียมตามข้อ 1 แล้วใช้ข้อมูลสำหรับฝึกสอนแต่ละโครงสร้าง 10 ครั้งแต่ละครั้งมีรอบการฝึกสอน 1,000 รอบ เลือกชุดโครงข่ายประสาทเทียมที่ให้ค่าความคลาดเคลื่อนต่ำสุด
5. แสดงผลการทดสอบในรูปของกราฟ โดยให้แกนนอนเป็นจำนวนโนดในชั้นซ่อนเร้น1 ตั้งแต่ 1 ถึง 10 แกนตั้งจำนวนโนดในชั้นซ่อนเร้น2 ตั้งแต่ 1 ถึง 10 สีของรูปกราฟแสดงเปอร์เซ็นต์จุดทดสอบตามค่าคลาดเคลื่อนที่ระยะต่ำกว่า 0.5 เมตร และ 1.0 เมตร ดังรูปที่ 4.6 และ 4.7 ตามลำดับ



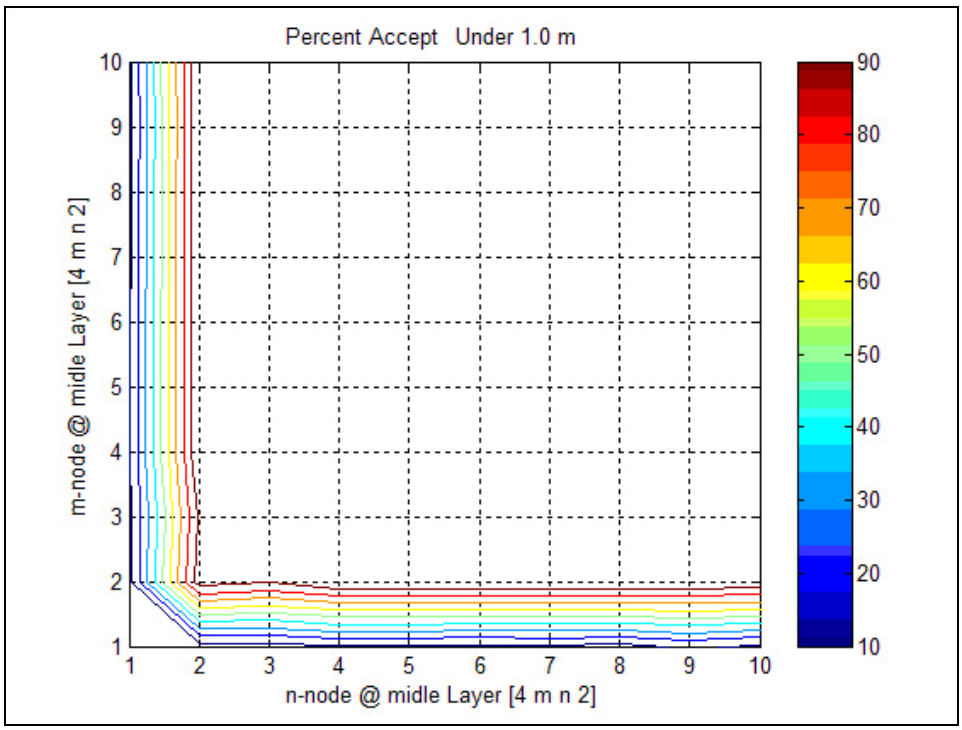
รูปที่ 4.4 เปอร์เซ็นต์ของจุดทดสอบที่ค่าคลาดเคลื่อนต่ำกว่า 0.2, 0.5 และ1.0 เมตรครั้งที่ 1



รูปที่ 4.5 เปอร์เซ็นต์ของจุดทดสอบที่ค่าคลาดเคลื่อนต่ำกว่า 0.2, 0.5 และ1.0 เมตรครั้งที่ 2



รูปที่ 4.6 เปอร์เซ็นต์ของจุดทดสอบที่ค่าคลาดเคลื่อนต่ำกว่า 0.5 เมตร

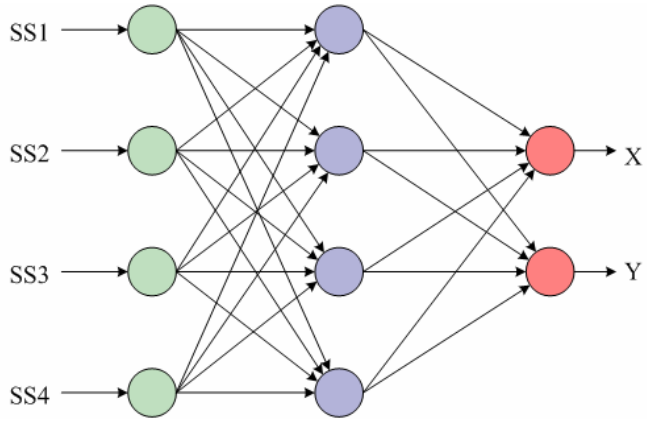


รูปที่ 4.7 เปอร์เซ็นต์ของจุดทดสอบที่ค่าคลาดเคลื่อนต่ำกว่า 1.0 เมตร

4.4.2 ผลการทดสอบเพื่อหาโครงสร้างของ FF-MLPs

จากการทดสอบสรุปได้ว่าขนาดของโครงสร้างแบบ 4-4-2 เป็นโครงสร้างเล็กสุดที่ให้เปอร์เซ็นต์ความคลาดเคลื่อนที่ระยะต่ำกว่า 0.5 เมตรและ 1.0 เมตรที่ 100 เปอร์เซ็นต์และโครงสร้างชนิดที่มีชั้นซ่อนเร้น 2 ชั้นโครงสร้างที่ให้ความคลาดเคลื่อนที่ระยะต่ำกว่า 1.0 เมตรที่ 100 เปอร์เซ็นต์จำเป็นต้องมีชั้นซ่อนเร้นตั้งแต่ 2 โนดขึ้นหรือมีโครงสร้างอย่างต่ำแบบ 4-2-2-2

สำหรับวิทยานิพนธ์นี้เลือกใช้โครงสร้างแบบ 4-4-2 ซึ่งประกอบด้วยชั้นอินพุต 4 โนดจากจุดเข้าถึง 4 ตำแหน่ง ชั้นซ่อนเร้น 1 ชั้นจำนวนโนด 4 โนด และชั้นเอาต์พุต 2 โนดคือตำแหน่ง (x, y) ฟังก์ชั่นการถ่ายโอนของแต่ละชั้นประกอบด้วย tansig, tansig และ purelin ตามลำดับ โครงสร้างโครงข่ายประสาทเทียมแบบ FF-MLPs ที่เลือกสามารถแสดงดังรูปที่ 4.8



รูปที่ 4.8 โครงข่ายประสาทเทียมสำหรับเป็นแบบจำลองพื้นที่ทดสอบ

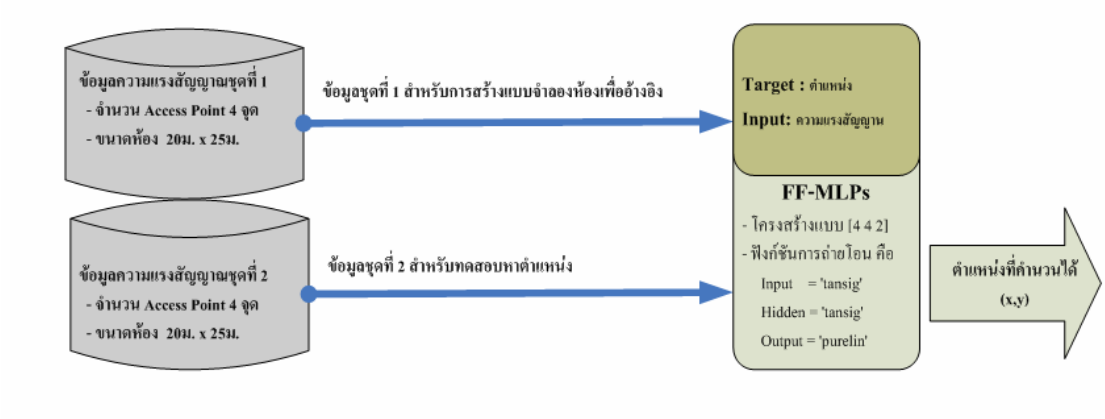
4.5 ทดสอบระบบระบุตำแหน่งตนเอง

การทดสอบระบบระบุตำแหน่งตนเอง การทดสอบนี้ใช้ข้อมูลชุดที่ 2 หรือข้อมูลสำหรับทดสอบป้อนให้กับโครงข่ายประสาทเทียมเพื่อหาตำแหน่งตนเอง การทดสอบแบ่งออกเป็น 2 ขั้นตอน คือ การทดสอบโดยใช้โครงสร้าง FF-MLPs อย่างเดียว และการทดสอบโดยใช้โครงสร้าง FF-MLPs + RBF

4.5.1 การทดสอบเพื่อระบุตำแหน่งโดยใช้โครงสร้าง FF-MLPs

จากขั้นตอนการหาแบบจำลองพื้นที่ทดสอบโดยใช้โครงข่ายประสาทเทียมแบบ FF-MLPsได้โครงสร้างแบบ 4-4-2 กระบวนการที่ทดสอบคือใช้ข้อมูลชุดที่ 1 ฝึกสอนและทดสอบโครงข่ายประสาทเทียม แต่การทดสอบเพื่อระบุตำแหน่งโดย FF-MLPs นี้จะใช้ข้อมูลชุดที่ 2

ทดสอบเพื่อยืนยันโครงสร้างของโครงข่ายประสาทเทียม กระบวนการทำงานสามารถแสดงดัง รูปที่ 4.9 ระบบระบุตำแหน่งโดยโครงข่ายประสาทเทียมแบบ FF-MLPs

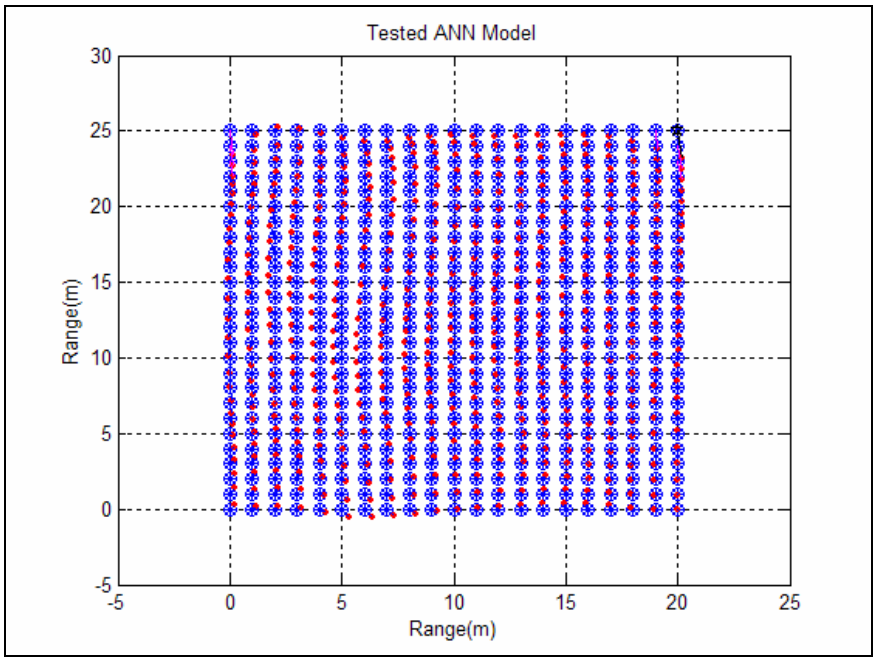


รูปที่ 4.9 ระบบระบุตำแหน่งโดยโครงข่ายประสาทเทียมแบบ FF-MLPs

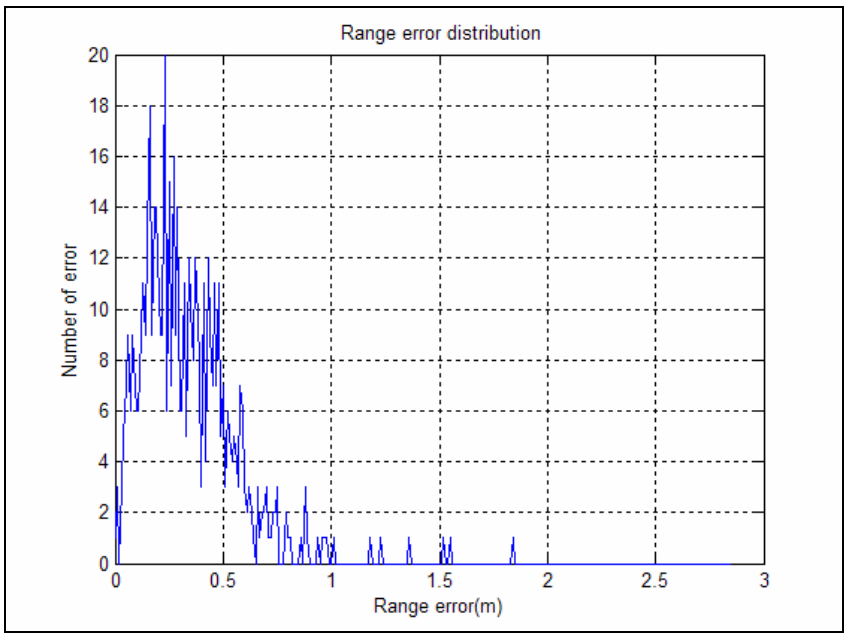
ผลการทดสอบโดยแสดงดังรูปที่ 4.10 ผลการทดสอบด้วยข้อมูลชุดที่ 1 หรือข้อมูลสำหรับฝึกสอนโครงข่ายประสาทเทียมเครื่องหมาย ⊗ แทนตำแหน่งจริงและ ·(จุด) แทนตำแหน่งที่ได้จากการทดสอบโครงข่ายประสาทเทียม จุดทดสอบที่ค่าคลาดเคลื่อนจากตำแหน่งจริงเกินกว่า 1 เมตร มีจำนวน 539 จาก 546 จุดหรือ 98.72 เปอร์เซ็นต์ เมื่อกำหนดระยะคลาดเคลื่อนให้มีความละเอียดทุก ๆ 1 เซนติเมตรนับจำนวนครั้งของระยะคลาดเคลื่อนต่าง ๆ แสดงในรูปที่ 4.11 การกระจายของระยะคลาดเคลื่อนจากการทดสอบด้วยข้อมูลชุดที่ 1 ให้แกนนอนเป็นระยะคลาดเคลื่อนหน่วยเป็นเมตรและแกนตั้งเป็นจำนวนครั้งของระยะคลาดเคลื่อนจากรูปมีจำนวนจุดที่ระยะคลาดเคลื่อนมากกว่า 1 เมตร 7 จุดหรือ 1.28 เปอร์เซ็นต์

รูปที่ 4.12 ผลการทดสอบด้วยข้อมูลชุดที่ 2 หรือข้อมูลสำหรับทดสอบโครงข่ายประสาทเทียมเส้นที่ขีดขึ้นจากเครื่องหมาย ⊗ ไปยังเครื่องหมาย ·(จุด) คือระยะจากจุดจริงถึงจุดทดสอบ ผลที่ได้มีค่าคลาดเคลื่อนจากตำแหน่งจริงเกินกว่า 1 เมตร จำนวน 55 จาก 546 จุด หรือ 10.07 เปอร์เซ็นต์ จากรูปที่ 4.13 การกระจายของระยะคลาดเคลื่อนจากการทดสอบด้วยข้อมูลชุดที่ 2 เห็นได้ว่ามีระยะคลาดเคลื่อนสูงสุดถึง 16 เมตร

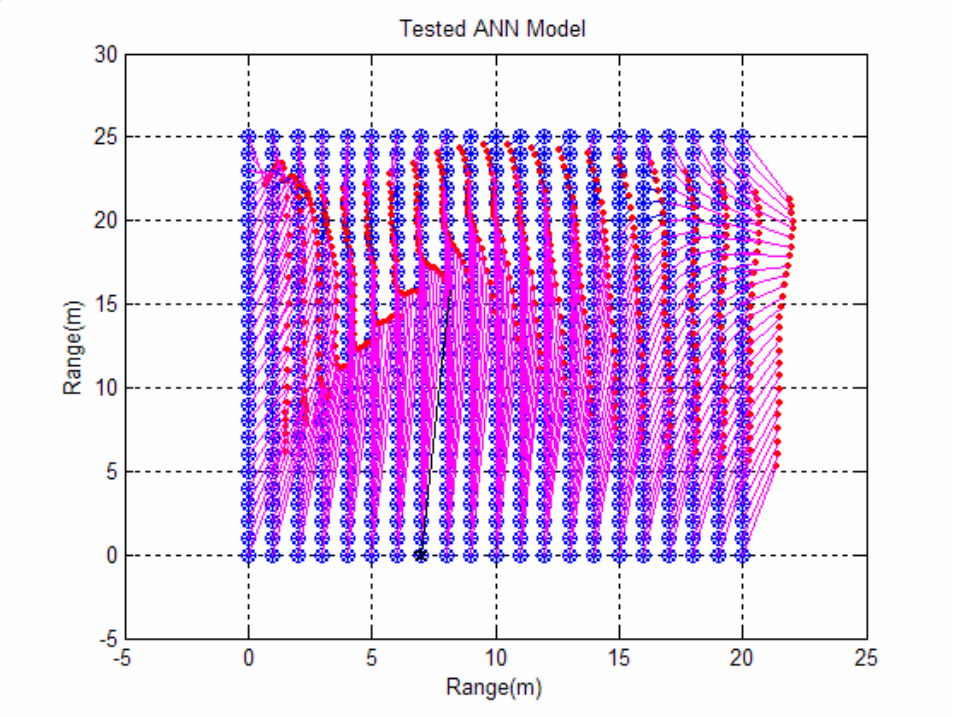
นั่นคือการใช้โครงข่ายประสาทเทียมแบบ FF-MLPs เพียงอย่างเดียวไม่สามารถใช้ระบุตำแหน่งตนเองได้ถูกต้องในระยะ 1 เมตรอย่างน้อย 80 เปอร์เซ็นต์ได้ตามวัตถุประสงค์ที่ตั้งไว้ ถึงแม้ว่าการสร้างแบบจำลองพื้นที่ทดสอบตามรูปที่ 4.10 จะให้ความถูกต้องสูงถึง 98.72 เปอร์เซ็นต์ก็ตาม



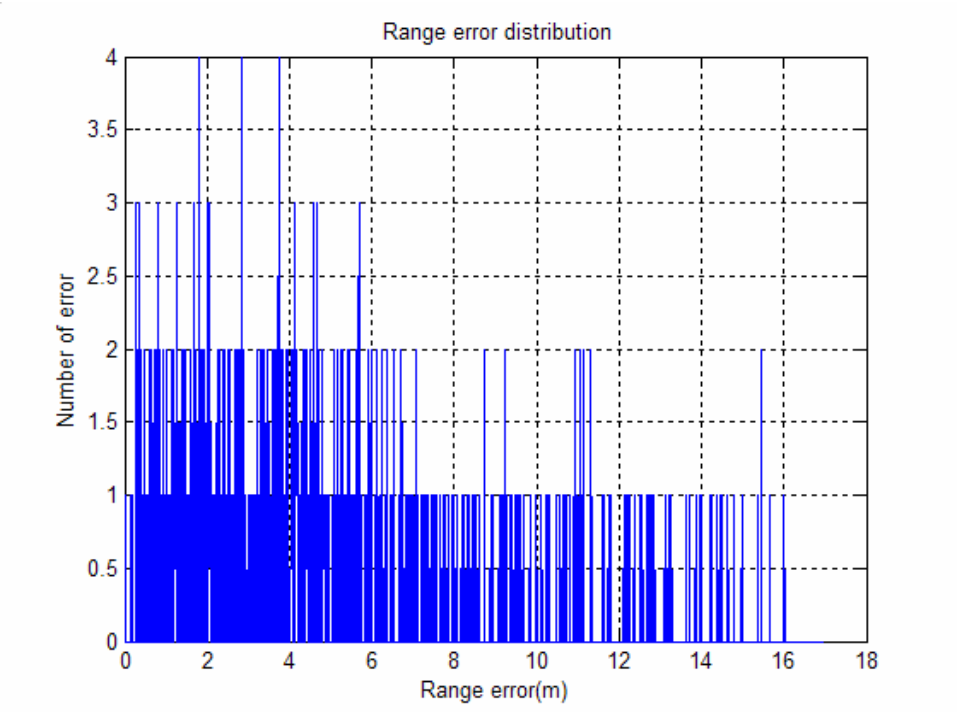
รูปที่ 4.10 ผลการทดสอบด้วยข้อมูลชุดที่ 1 หรือข้อมูลสำหรับฝึกสอนโครงข่ายประสาทเทียม เครื่องหมาย ⊗ แทนตำแหน่งจริงและ •(จุด) แทนตำแหน่งที่ได้โครงข่าย



รูปที่ 4.11 การกระจายของระยะคลาดเคลื่อนจากการทดสอบด้วยข้อมูลชุดที่ 1



รูปที่ 4.12 ผลการทดสอบด้วยข้อมูลชุดที่ 2 หรือข้อมูลสำหรับทดสอบโครงข่ายประสาทเทียม

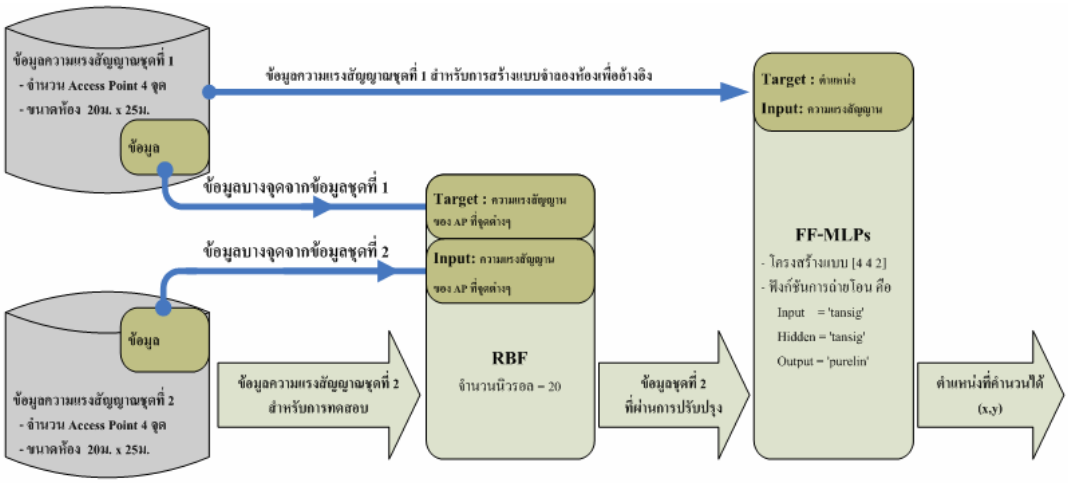


รูปที่ 4.13 การกระจายของระยะคลาดเคลื่อนจากการทดสอบด้วยข้อมูลชุดที่ 2

4.5.2 การทดสอบเพื่อระบุตำแหน่งโดย FF-MLPs ร่วมกับ RBF

การทดสอบโครงข่ายแบบประสาทเทียมแบบ FF-MLPs เพียงอย่างเดียวผลลัพธ์ที่ได้ไม่ดีนักจึงปรับแนวคิดใหม่ กล่าวคือวิธีการทดสอบโดยตรงด้วย FF-MLPs เปรียบเสมือนให้คนปิดตาค้นหาสิ่งของที่วางไว้ในห้องโดยบอกลักษณะแผนผังของห้องและตำแหน่งสิ่งของที่ต้องการให้ค้นหาวิธีการนี้ถือว่ายากมากสำหรับคนที่ไม่เคยชินกับลักษณะของห้อง แต่จะง่ายมากขึ้นหากเราบอกแผนผังห้องและบอกตำแหน่งอ้างอิงบางจุดให้คนปิดตารู้แล้วค่อยให้คนปิดตานี้ค้นหาสิ่งของ

ด้วยแนวคิดแบบนี้จึงเสนอโครงข่ายประสาทเทียมแบบที่ให้ FF-MLPs ให้ทำงานร่วมกับ RBF โดยหน้าที่ของโครงข่าย RBF คือการหาฟังก์ชันเพื่อส่งผ่านข้อมูลจริงที่กำลังทดสอบให้มีลักษณะเดียวกับข้อมูลที่สร้างแบบจำลองพื้นที่ทดสอบกระบวนการทำงานนี้สามารถแสดงดังรูปที่ 4.14 ระบบระบุตำแหน่งโดยโครงข่ายประสาทเทียมแบบ FF-MLPs ร่วมกับ RBF



รูปที่ 4.14 ระบบระบุตำแหน่งโดยโครงข่ายประสาทเทียมแบบ FF-MLPs ร่วมกับ RBF

ขั้นตอนการทดสอบมีดังนี้

1. นำข้อมูลบางจุดที่ระยะทุก ๆ 6 เมตรจากข้อมูลชุดที่ 1 เป็นค่าเป้าหมายและใช้ข้อมูลบางจุดที่ระยะทุก ๆ 6 เมตรเช่นกันจากข้อมูลชุดที่ 2 เป็นค่าอินพุต
2. ทำการสร้างและฝึกสอน RBF กำหนดจำนวนโนดของ RBF เท่ากับ 20 โนด
3. ทดสอบโดยการนำข้อมูลชุดที่ 2 ทั้งหมดป้อนให้กับโครงข่าย RBF ได้ผลลัพธ์เป็นข้อมูลชุดที่ 2 ที่ผ่านการปรับปรุงให้เหมาะสมเพื่อทดสอบโดย FF-MLPs ต่อไป
4. ทดสอบด้วย FF-MLPs ด้วยข้อมูลจากข้อ 3

5. ทำซ้ำขั้นตอนที่ 3 ถึง 4 เพื่อยืนยันการทำงานของโครงสร้างผสมระหว่าง FF-MLPs และ RBF ด้วยข้อมูลชุดที่ 1 (ระยะอ้างอิง RBF ทุก ๆ 4 เมตร)
6. ทำซ้ำขั้นตอนที่ 1 ถึง 4 แต่กำหนดจุดอ้างอิงที่ระยะทุก ๆ 4 เมตรเพื่อทดสอบการทำงานของโครงสร้างผสมระหว่าง FF-MLPs และ RBF ด้วยข้อมูลชุดที่ 2
7. ทำซ้ำขั้นตอนที่ 3 ถึง 4 เพื่อยืนยันการทำงานของโครงสร้างผสมระหว่าง FF-MLPs และ RBF ด้วยข้อมูลชุดที่ 1 (ระยะอ้างอิง RBF ทุก ๆ 6 เมตร)

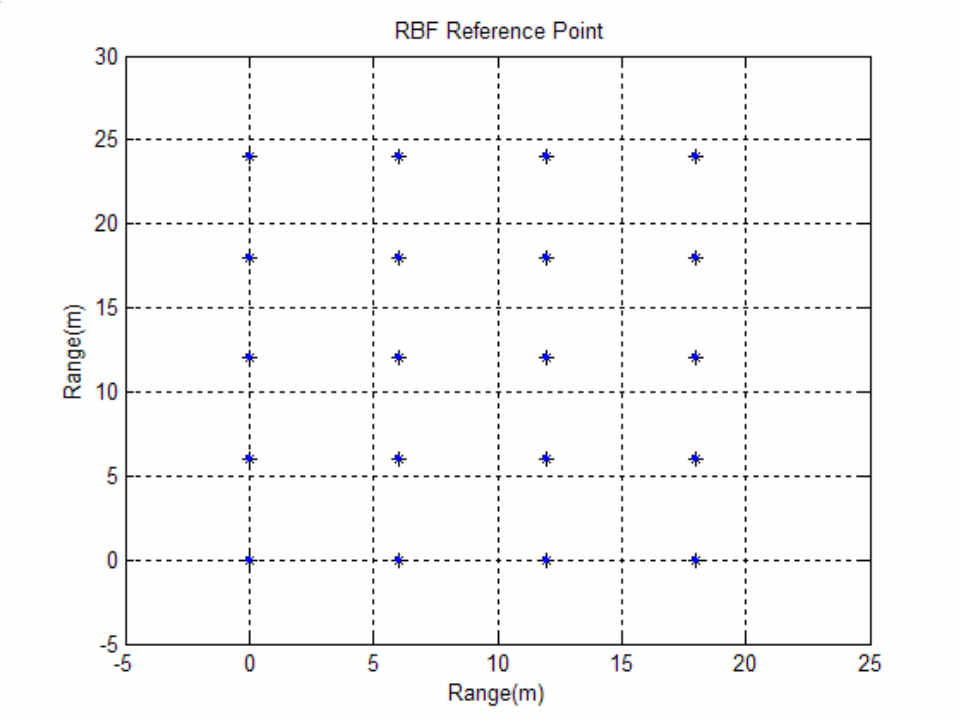
ผลการทดสอบระบบระบุตำแหน่งตนเองทั้งหมดได้แสดงไว้ตามตารางที่ 4.2 เห็นได้ว่าจากพื้นที่ทดสอบขนาด 20 เมตร x 25 เมตร กำหนดจุดทดสอบจำนวน 21 x 26 = 546 จุด เมื่อนำโครงข่ายประสาทเทียมแบบ FF-MLPs ทำงานร่วมกับ RBF ที่จุดอ้างอิงทุก ๆ 6 เมตรจะสามารถระบุตำแหน่งได้ถูกต้องในระยะไม่เกิน 1.0 เมตรจำนวน 388 จุดจากจุดทดสอบ 546 จุดหรือ 71.06 เปอร์เซ็นต์และที่จุดอ้างอิงทุก ๆ 4 เมตรจะสามารถระบุตำแหน่งได้ถูกต้องในระยะไม่เกิน 1 เมตร จำนวน 449 จุดจากจุดทดสอบ 546 จุด หรือ 82.23 เปอร์เซ็นต์

เพื่อยืนยันการทำงานของโครงข่ายประสาทเทียมแบบ FF-MLPs ทำงานร่วมกับ RBF จึงใช้ข้อมูลชุดที่ 1 ทดสอบการทำงานโดยผลการทดสอบเมื่อให้ระยะอ้างอิงทุก ๆ 6 เมตรสามารถระบุตำแหน่งได้ถูกต้องในระยะไม่เกิน 1 เมตร 82.23 เปอร์เซ็นต์และระยะอ้างอิงทุก ๆ 4 เมตร สามารถระบุตำแหน่งได้ถูกต้อง 82.23 เปอร์เซ็นต์ ลดลงจากโครงสร้างแบบ FF-MLPs เพียงอย่างเดียวเล็กน้อยจากผลการทำงานของ RBF

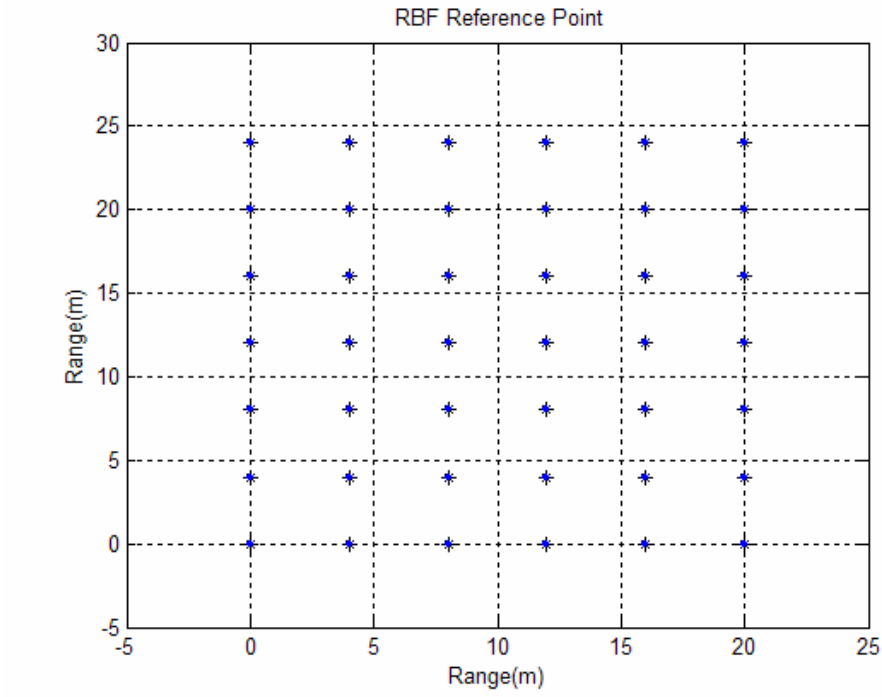
ผลการทดสอบที่เงื่อนไขการทดสอบต่าง ๆ แสดงในรูปกราฟการเบี่ยงเบนข้อมูลและกราฟกระจายข้อมูลของระยะคลาดเคลื่อน

ตารางที่ 4.2 เปอร์เซ็นต์ความถูกต้องของการทดสอบการระบุตำแหน่งตนเอง

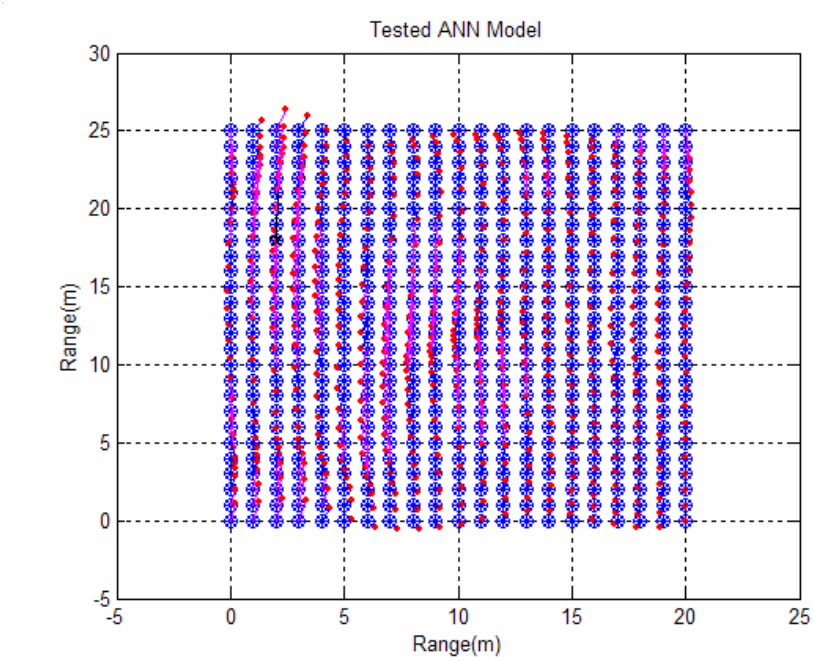
เงื่อนไขการทดสอบ	ขอบเขตระยะถูกต้องสูงสุด			ระยะผิดพลาดเฉลี่ย	ระยะผิดพลาดมากสุด
	1.0 เมตร	0.5 เมตร	0.2 เมตร		
ทดสอบด้วย FF-MLPs ด้วยข้อมูลชุดที่ 1 ดังรูปที่ 4.10	98.72%	80.77%	28.39%	0.34 เมตร	1.85 เมตร ที่จุด (20, 25)
ทดสอบด้วย FF-MLPs ด้วยข้อมูลชุดที่ 2 ดังรูปที่ 4.12	10.07%	4.21%	0.55%	4.87 เมตร	16.00 เมตร ที่จุด (7, 0)
ทดสอบด้วย FF-MLPs + RBF ด้วยข้อมูลชุดที่ 2 ที่ขนาดจุดอ้างอิงทุก ๆ 6 เมตรดังรูปที่ 4.17	71.06%	48.53%	15.01%	0.73 เมตร	2.10 เมตร ที่จุด (6, 0)
ทดสอบด้วย FF-MLPs + RBF ด้วยข้อมูลชุดที่ 1 ที่ขนาดจุดอ้างอิงทุก ๆ 6 เมตรดังรูปที่ 4.19	97.44%	69.96%	21.61%	0.41 เมตร	2.09 เมตร ที่จุด (20, 25)
ทดสอบด้วย FF-MLPs + RBF ด้วยข้อมูลชุดที่ 2 ที่ขนาดจุดอ้างอิงทุก ๆ 4 เมตรดังรูปที่ 4.21	82.23%	52.20%	18.50%	0.58 เมตร	1.94 เมตร ที่จุด (5, 0)
ทดสอบด้วย FF-MLPs + RBF ด้วยข้อมูลชุดที่ 1 ที่ขนาดจุดอ้างอิงทุก ๆ 4 เมตรดังรูปที่ 4.23	95.24%	62.27%	13.55%	0.47 เมตร	2.01 เมตร ที่จุด (20, 25)



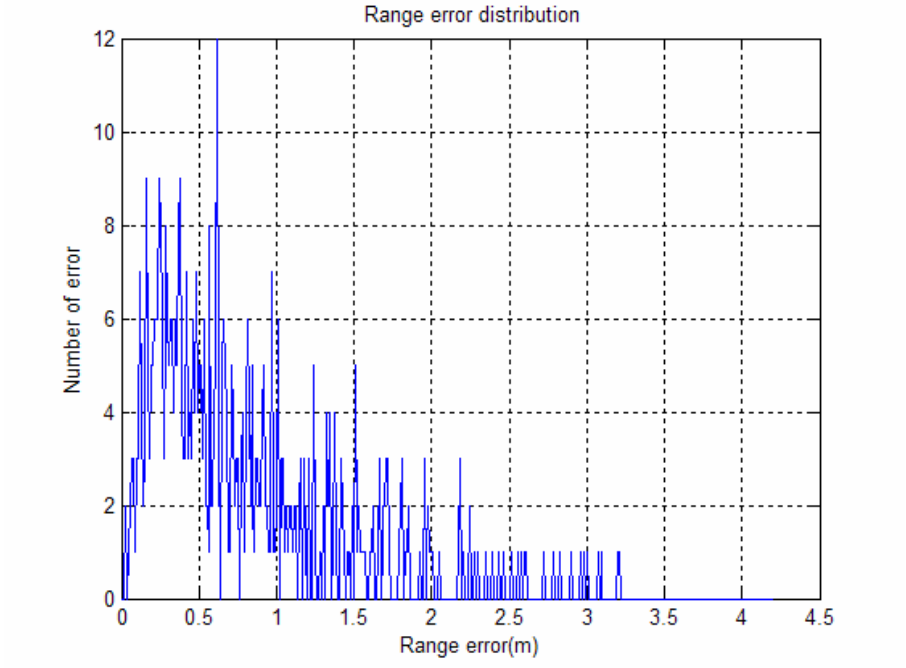
รูปที่ 4.15 ตำแหน่งจุดอ้างอิงทุก ๆ 6 เมตรสำหรับการสร้างและฝึกสอน RBF



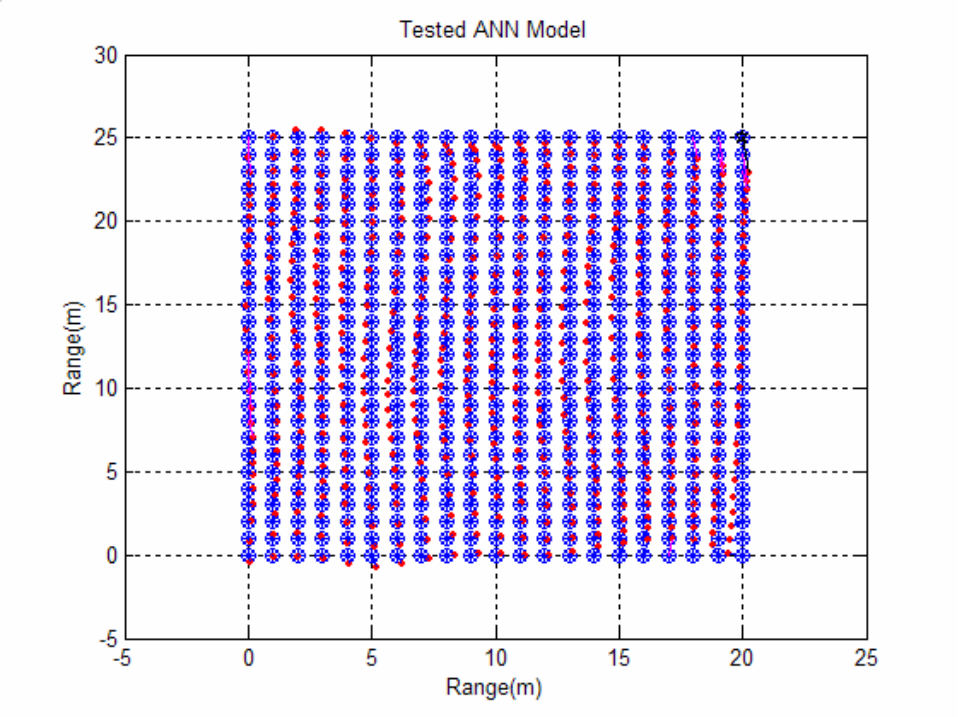
รูปที่ 4.16 ตำแหน่งจุดอ้างอิงทุก ๆ 4 เมตรสำหรับการสร้างและฝึกสอน RBF



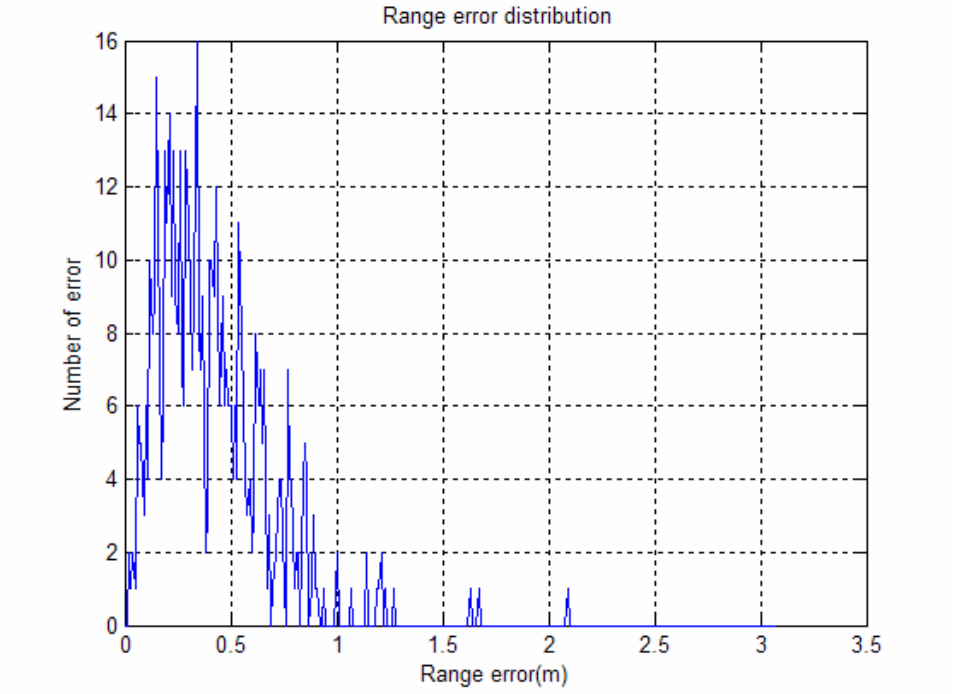
รูปที่ 4.17 ผลการทดสอบ FF-MLPs + RBF ที่จุดอ้างอิงทุก ๆ 6 เมตรด้วยข้อมูลชุดที่ 2



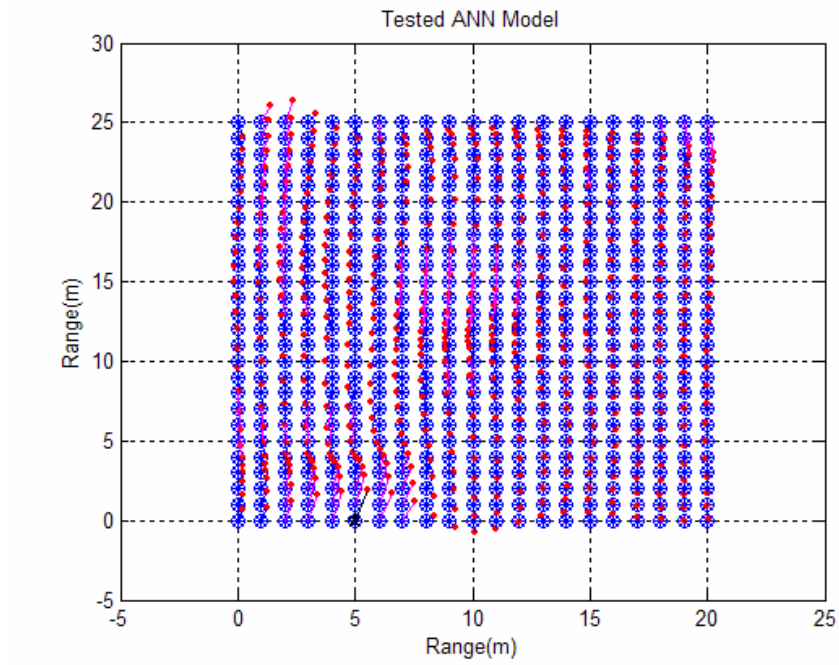
รูปที่ 4.18 การกระจายของระยะคลาดเคลื่อนที่ได้จากการทดสอบโครงสร้าง FF-MLPs + RBF กำหนดจุดอ้างอิงทุก ๆ 6 เมตรและทดสอบด้วยข้อมูลชุดที่ 2



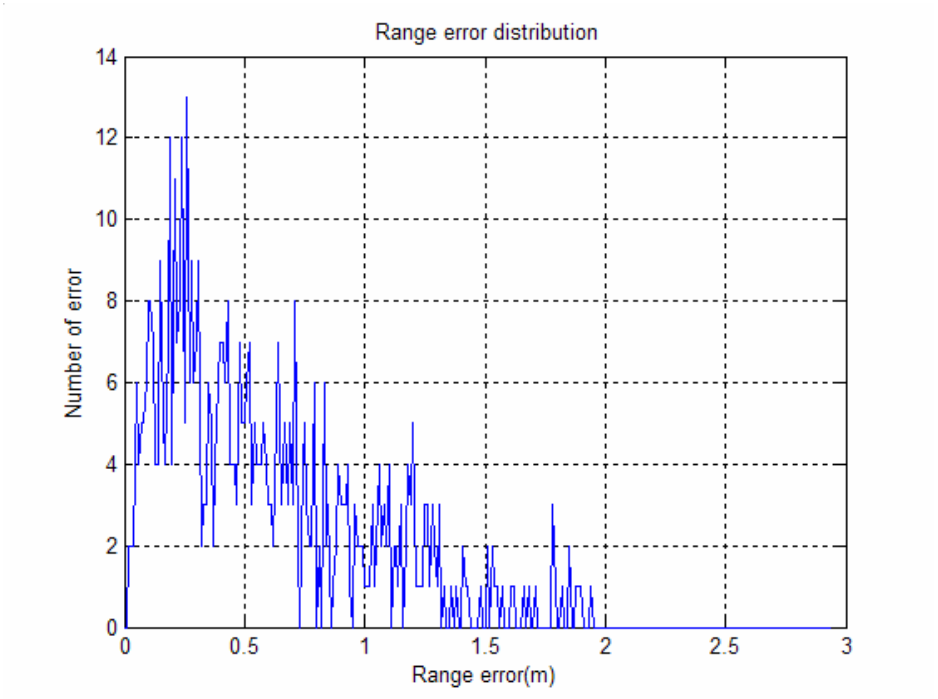
รูปที่ 4.19 ผลการทดสอบ FF-MLPs + RBF ที่จุดอ้างอิงทุก ๆ 6 เมตรด้วยข้อมูลชุดที่ 1



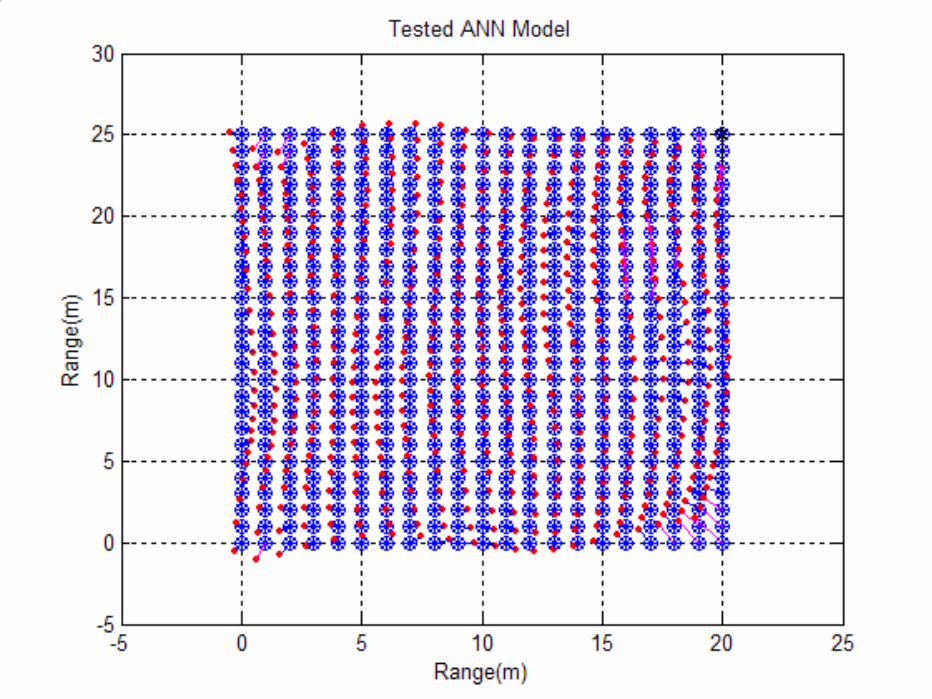
รูปที่ 4.20 การกระจายของระยะคลาดเคลื่อนที่ได้จากการทดสอบโครงสร้าง FF-MLPs + RBF กำหนดจุดอ้างอิงทุก ๆ 6 เมตรและทดสอบด้วยข้อมูลชุดที่ 1



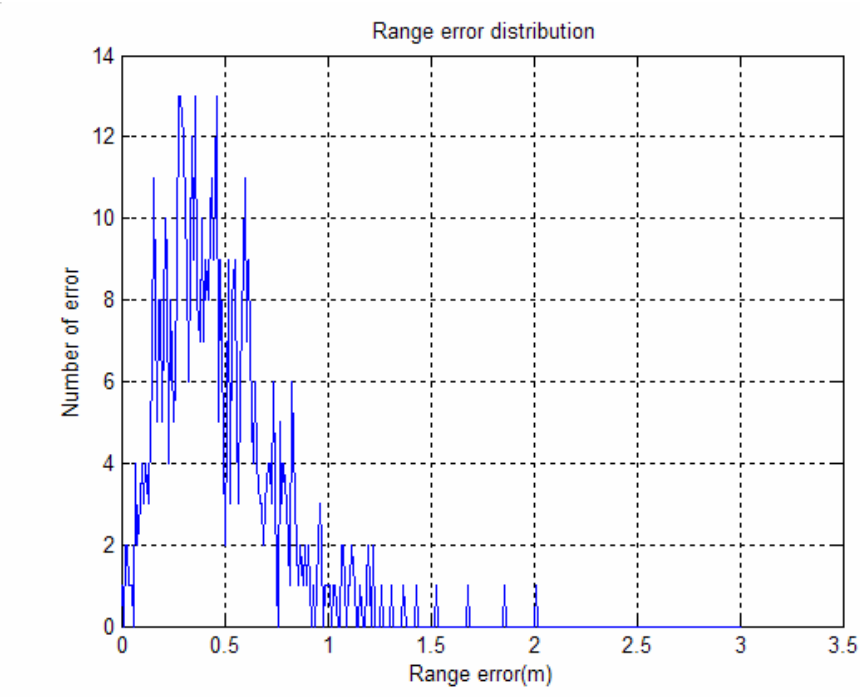
รูปที่ 4.21 ผลการทดสอบ FF-MLPs + RBF ที่จุดอ้างอิงทุก ๆ 4 เมตรด้วยข้อมูลชุดที่ 2



รูปที่ 4.22 การกระจายของระยะคลาดเคลื่อนที่ได้จากการทดสอบโครงสร้าง FF-MLPs + RBF กำหนดจุดอ้างอิงทุก ๆ 4 เมตรและทดสอบด้วยข้อมูลชุดที่ 2



รูปที่ 4.23 ผลการทดสอบ FF-MLPs + RBF ที่จุดอ้างอิงทุก ๆ 4 เมตรด้วยข้อมูลชุดที่ 1



รูปที่ 4.24 การกระจายของระยะคลาดเคลื่อนที่ได้จากการทดสอบโครงสร้าง FF-MLPs + RBF กำหนดจุดอ้างอิงทุก ๆ 4 เมตรและทดสอบด้วยข้อมูลชุดที่ 1

4.6 สรุป

การทดสอบระบบระบุตำแหน่งนี้ทำการทดสอบในพื้นที่ 20 x 25 ตารางเมตรภายในห้องปฏิบัติการไมโครโพรเซสเซอร์และห้องปฏิบัติการวงจรและอุปกรณ์ อาคารศูนย์เครื่องมือวิทยาศาสตร์และเทคโนโลยี 3 มหาวิทยาลัยเทคโนโลยีสุรนารี กำหนดจุดอ้างอิงและทดสอบจำนวน 546 จุด เก็บข้อมูลด้วยโปรแกรม “Site Survey” ที่พัฒนาขึ้นโดยโปรแกรมวิชวลเบสิค 6.0 (VB6.0) เก็บข้อมูลจากจุดเข้าถึง 4 จุดจำนวน 2 ชุด โดยที่ชุดที่ 1 สำหรับสร้างโครงข่ายประสาทเทียมแบบ FF-MLPs เพื่อเป็นแบบจำลองของพื้นที่ทดสอบ และข้อมูลชุดที่ 2 สำหรับทดสอบการทำงาน

ระบบระบุตำแหน่งตนเองประกอบด้วยโครงข่ายประสาทเทียมแบบ FF-MLPs และ RBF ทำงานร่วมกันโดย RBF ทำหน้าที่ปรับปรุงข้อมูลสำหรับทดสอบให้เหมาะสม สร้างขึ้นจากข้อมูลบางส่วนของข้อมูลชุดที่ 1 และ 2 การทดสอบเมื่อได้โครงข่ายประสาทเทียมแบบ FF-MLPs แล้วจะใช้ข้อมูลชุดที่ 2 ป้อนให้ RBF นำผลที่ได้ไปทดสอบด้วย FF-MLPs เพื่อหาตำแหน่ง

ผลการทดสอบสามารถสรุปได้ ดังนี้

1. โครงสร้างของระบบโดยรวมแสดงดังรูปที่ 4.14 ประกอบด้วย FF-MLPs และ RBF
2. โครงสร้างของ FF-MLPs สำหรับเป็นแบบจำลองของพื้นที่ทดสอบ ประกอบด้วยชั้นอินพุต 4 โนด ชั้นซ่อนเร้น 1 ชั้น 4 โนดและชั้นเอาต์พุต 2 โนดและมีฟังก์ชันการถ่ายโอนเป็น tansig ,tansig และ purelin ตามลำดับ
3. โครงสร้างแบบ RBF กำหนดให้มีโนดในชั้นซ่อนเร้นจำนวน 20 โนด
4. สามารถระบุตำแหน่งได้ถูกต้องที่ระยะไม่เกิน 1 เมตร 74.81 เปอร์เซ็นต์เมื่อใช้จุดอ้างอิงทุก ๆ 6 เมตรในการสร้าง RBF
5. ยืนยันการทำงานของโครงสร้าง FF-MLPs + RBF ด้วยข้อมูลชุดที่ 1 จากโครงสร้าง FF-MLPs เพียงอย่างเดียวถูกต้อง 98.72 เปอร์เซ็นต์ ลดลงเหลือ 97.44 เปอร์เซ็นต์
6. สามารถระบุตำแหน่งได้ถูกต้องที่ระยะไม่เกิน 1 เมตร 82.23 เปอร์เซ็นต์เมื่อใช้จุดอ้างอิงทุก ๆ 4 เมตรในการสร้าง RBF
7. ยืนยันการทำงานของโครงสร้าง FF-MLPs + RBF ด้วยข้อมูลชุดที่ 1 จากโครงสร้าง FF-MLPs เพียงอย่างเดียวถูกต้อง 98.72 เปอร์เซ็นต์ ลดลงเหลือ 95.24 เปอร์เซ็นต์

บทที่ 5
การใช้จีนเนติกในการช่วยหาโครงสร้างของ FF-MLPs ที่เหมาะสม

5.1 บทนำ

จากการหาโครงสร้างของ FF-MLPs โดยวิธีการลองผิดลองถูก (trial and error) เพื่อทำการหาโครงสร้างและองค์ประกอบของโครงข่ายประสาทเทียม ทำให้ได้โครงสร้าง FF-MLPs ของพื้นที่ทดสอบในบทที่ 4 คือ [4, 4, 2] ซึ่งประกอบด้วยชั้นอินพุตจำนวนโนด 4 โนดจากความแรงของสัญญาณจากจุดเข้าถึง 4 จุด ชั้นซ่อนเร้นหนึ่งชั้นจำนวนโนด 4 โนด และชั้นเอาต์พุตจำนวนโนด 2 โนด คือตำแหน่ง (x, y)

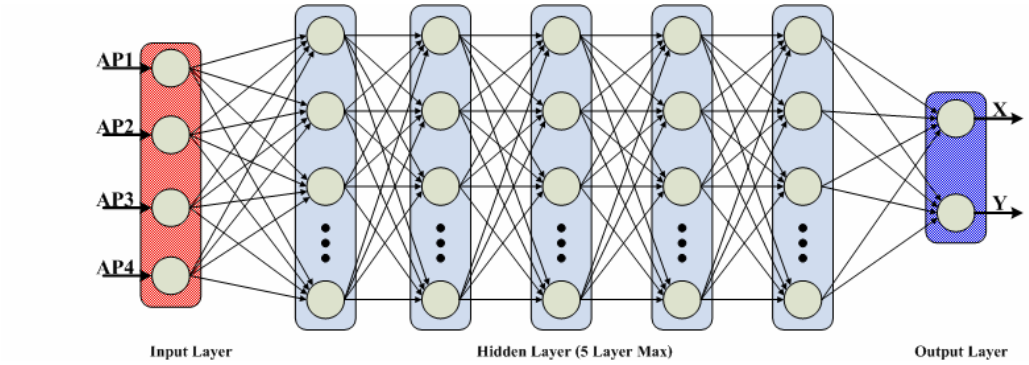
เพื่อเป็นการปรับปรุงการทำงานของ FF-MLPs ให้ดีขึ้นและยืนยันโครงสร้างที่ใช้ทดสอบว่าเหมาะสมดีแล้วจึงใช้จีนเนติกอัลกอริทึมเพื่อหาโครงสร้างของ FF-MLPs ที่ดีที่สุดสำหรับการทดสอบนี้ ขั้นตอนการทำงานจะเริ่มจากกำหนดโครงสร้างของ FF-MLPs ให้มีขนาดใหญ่และให้จีนเนติกทำการค้นหาโครงสร้างใหม่ที่เหมาะสมโดยมีกำหนดสำคัญ 2 ข้อ คือ โครงสร้างใหม่ที่ได้ต้องสามารถระบุตำแหน่งได้ถูกต้องภายใต้ขอบเขตความคลาดเคลื่อนที่กำหนดและโครงสร้างใหม่ต้องมีขนาดโครงข่ายประสาทเทียมที่เล็กลงทั้งจำนวนชั้นซ่อนเร้นและจำนวนโนด

5.2 วิธีการค้นหาโครงสร้างของ FF-MLPs ด้วยจีนเนติกอัลกอริทึม

การหาโครงสร้างของ FF-MLPs ด้วยจีนเนติกอัลกอริทึมในงานวิจัยวิทยานิพนธ์นี้ กำหนดตัวแปรที่ต้องการให้จีนเนติกค้นหาคือจำนวนชั้นซ่อนเร้นและจำนวนโนดของแต่ละชั้นของ FF-MLPs ในส่วนของกระบวนการฟังก์ชันวัตถุประสงค์จะทำการสร้างโครงข่ายประสาทเทียมขึ้นมาและทำการฝึกสอนด้วยการฝึกสอนแบบแพร่กลับ (back-propagation) ทดสอบโครงข่ายประสาทเทียมที่ได้คำนวนหาจำนวนจุด OffError นอกจากนี้ผลลัพธ์จากการทำงานของฟังก์ชันวัตถุประสงค์จะได้โครงข่ายประสาทเทียมที่ผ่านการฝึกสอนแล้วด้วย

5.2.1 การกำหนดพารามิเตอร์ของ FF-MLPs ที่ต้องการให้จีนเนติกอัลกอริทึมค้นหา

สำหรับการวิจัยวิทยานิพนธ์นี้กำหนดโครงสร้างเริ่มต้นของ FF-MLPs ที่ต้องการค้นหาให้มีรายละเอียดดังต่อนี้ ให้มีชั้นอินพุต 4 โนดจากจุดเข้าถึง 4 จุด, ชั้นเอาต์พุต 2 โนด คือตำแหน่ง (x, y) และชั้นซ่อนเร้นสูงสุด 5 ชั้น ดังรูปที่ 5.1 กำหนดให้ชั้นซ่อนเร้นแต่ละชั้นมีจำนวนโนดสูงสุด 7 โนด และต่ำสุด 0 โนด ความหมายของการมีโนดเท่ากับ 0 คือไม่มีชั้นซ่อนเร้นนั้นปรากฏอยู่



รูปที่ 5.1 โครงสร้างเริ่มต้นของ FF-MLPs ที่ต้องการค้นหา

กำหนดตัวแปรต่าง ๆ ที่ใช้ในการทำงานของจีนเนติก ดังนี้

OffError คือ จำนวนจุดทดสอบที่มีระยะจากการทดสอบคลาดเคลื่อนจากตำแหน่งจริงมากกว่าระยะที่กำหนดไว้ เช่น OffError0.2 หมายถึง จำนวนจุดทดสอบที่ให้ค่าคลาดเคลื่อนจากจุดอ้างอิงเกิน 0.2 เมตร

AcceptError คือ ค่าเปอร์เซ็นของจุด OffError มากสุดที่ยอมรับได้เมื่อทำการทดสอบโครงข่ายประสาทเทียม

การกำหนดค่า AcceptError เพื่อเป็นการรับประกันค่าความผิดพลาดในการระบุตำแหน่งสูงสุดที่โครงข่ายประสาทเทียมทำงานได้ ค่านี้จะนำไปสู่การการลดจำนวนชั้นซ่อนเร้นลง กระบวนการทำงานคือ จำนวนชั้นซ่อนเร้นจะลดลงก็ต่อเมื่อผลการค้นหาของจีนเนติกอัลกอริทึม ให้จำนวนโนดในบางชั้นเป็น 0 และค่าความคลาดเคลื่อนที่คำนวณได้น้อยว่า AcceptError รอบการค้นหาถัดไปของจีนเนติกอัลกอริทึมจะบังคับให้ชั้นซ่อนเร้นนั้นมีจำนวนโนดเท่ากับ 0 หรือไม่มีชั้นซ่อนเร้นปรากฏอยู่

5.2.2 การหาโครงสร้างของ FF-MLPs ด้วยจีนเนติกอัลกอริทึม

การค้นหาโครงสร้างด้วยจีนเนติกนี้ใช้โปรแกรม MATLAB® ในการทำงาน เริ่มจากโครงสร้างเริ่มต้นของ FF-MLPs ที่ต้องการค้นหาในรูปที่ 5.1 จำนวนชั้นซ่อนเร้นสูงสุดเท่ากับ 5 ดังนั้นเราจึงกำหนดจำนวนพารามิเตอร์ของระบบที่ต้องการค้นหาเท่ากับ 5 ด้วย จากนั้นกำหนดให้ค่าของตัวแปรพารามิเตอร์ คือ จำนวนโนดที่มีในแต่ละชั้นซ่อนเร้น จะได้ว่าค่าสูงสุดของตัวแปรเท่ากับ 10, ค่าต่ำสุดของตัวแปรเท่ากับ 0 และเห็นได้ว่าค่าความละเอียดของตัวแปรจะมีค่าเป็น 1 เนื่องจากจำนวนโนดต้องเป็นจำนวนเต็ม ดังนั้นเมื่อแทนพารามิเตอร์นี้ด้วยระบบเลขฐานสอง สามารถคำนวณ

จำนวนบิตได้ดังนี้

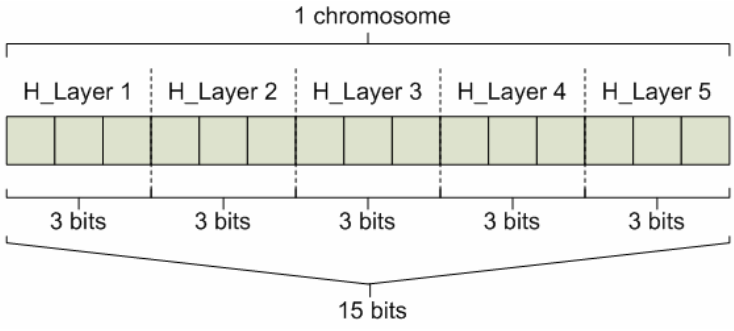
$$2^{nBit} \geq \frac{(Max\_Var - Min\_Var)}{Resolution\ Step} + 1 \tag{5.1}$$

$$2^{nBit} \geq \frac{(7 - 0)}{1} + 1 \tag{5.2}$$

$$nBit \geq \frac{\log(8)}{\log(2)} = 3 \tag{5.3}$$

Max_Var	คือ	ค่าสูงสุดของตัวแปรเท่ากับ 10
Min_Var	คือ	ค่าต่ำสุดของตัวแปรเท่ากับ 0
Resolution Step	คือ	ค่าความละเอียดของตัวแปรจะมีค่าเป็น 1
nBit	คือ	จำนวนบิตของตัวแปรเมื่อแทนเป็นเลขฐานสอง

นั่นคือกำหนดความละเอียดของค่าพารามิเตอร์เมื่อแทนเป็นระบบเลขฐานสองเท่ากับ 3 บิตจากการประกอบเข้ากันเป็นโครโมโซม 1 ชุด จะแทนด้วยระบบเลขฐานสองเท่ากับ 15 บิตมีองค์ประกอบดังรูปที่ 5.2 โดยแบ่งเป็นกลุ่ม ๆ ละ 3 บิต แต่ละกลุ่มแทนจำนวนโนดในชั้นซ่อนเร้นที่ 1, 2, 3, 4 และ 5 ตามลำดับ



รูปที่ 5.2 รายละเอียดค่าความยาวของโครโมโซม 1 ตัว

การประเมินผลคำตอบด้วยฟังก์ชันวัตถุประสงค์ และคำนวณหาค่าความเหมาะสมส่งกลับไปเพื่อใช้ในการคัดเลือกโครโมโซมที่ดีสำหรับการสืบสายพันธุ์ กำหนดให้ค่า OffError0.2 หรือในการคำนวณจะนับจำนวนจุดทดสอบที่ให้ค่าคลาดเคลื่อนจากจุดอ้างอิงเกิน 0.2 เมตร ให้ชื่อฟังก์ชันในโปรแกรม MATLAB® คือ GA_Objfunction ซึ่งมีขั้นตอนการทำงานดังต่อไปนี้

1. สร้างโครงข่ายประสาทเทียมจากโครงสร้างโครโมโซมที่ต้องการทดสอบ
2. ฝึกสอนโครงข่ายประสาทเทียม
3. ทดสอบโครงข่ายประสาทเทียมเพื่อคำนวณหา OffError0.2 หรือจำนวนจุดที่มีระยะคลาดเคลื่อนเกิน 0.2 เมตร
4. คำนวณค่าฟังก์ชันวัตถุประสงค์ (ObjVSel) แบ่งเป็น 2 กรณี

กรณีที่ 1 OffError น้อยกว่า AcceptError

$$\text{ObjVSel} = \text{AcceptError} \times k_1 \tag{5-4}$$

$$k_1 = \text{SumNode} \times 10^{\text{SumLayer}} \tag{5-5}$$

เมื่อ

SumNode คือ ผลรวมจำนวนโนดในชั้นซ่อนเร้นทั้งหมด

SumLayer คือ ผลรวมจำนวนชั้นซ่อนเร้นที่มีโนดไม่เป็นศูนย์

กรณีที่ 2 OffError มากกว่า AcceptError

$$\text{ObjVSel} = \text{OffError} \times k_2 \tag{5-6}$$

$$k_2 = \sum\left([4\ 7\ 7\ 7\ 7\ 7\ 2]\right)\times 10^{\sum([1\ 1\ 1\ 1\ 1\ 1\ 1])} = 41\times 10^{7}$$

5. ทำซ้ำขั้นตอนที่ 2,3 และ 4 ตามจำนวนรอบการฝึกสอนที่ตั้งไว้เพื่อเป็นการตรวจสอบซ้ำสำหรับการฝึกสอนโครงข่ายประสาทเทียม

6. เลือกรอบการทำงานที่ให้ค่า OffError0.2 ดีที่สุดเป็นค่าฟังก์ชันวัตถุประสงค์สำหรับโครโมโซมนี้
7. ทำซ้ำขั้นตอนที่ 1, 2, 3, 4, 5 และ 6 สำหรับโครโมโซมทุกตัว

จากโปรแกรมฟังก์ชันวัตถุประสงค์ข้างต้น มีเพื่อให้ความสำคัญกับการกำจัดประชากรที่ให้โครงสร้างของ FF-MLPs ที่มีจำนวนชั้นซ่อนเร้นมาก จึงมีการกำหนดค่าค่าฟังก์ชันวัตถุประสงค์ ขึ้นมา 2 กรณี

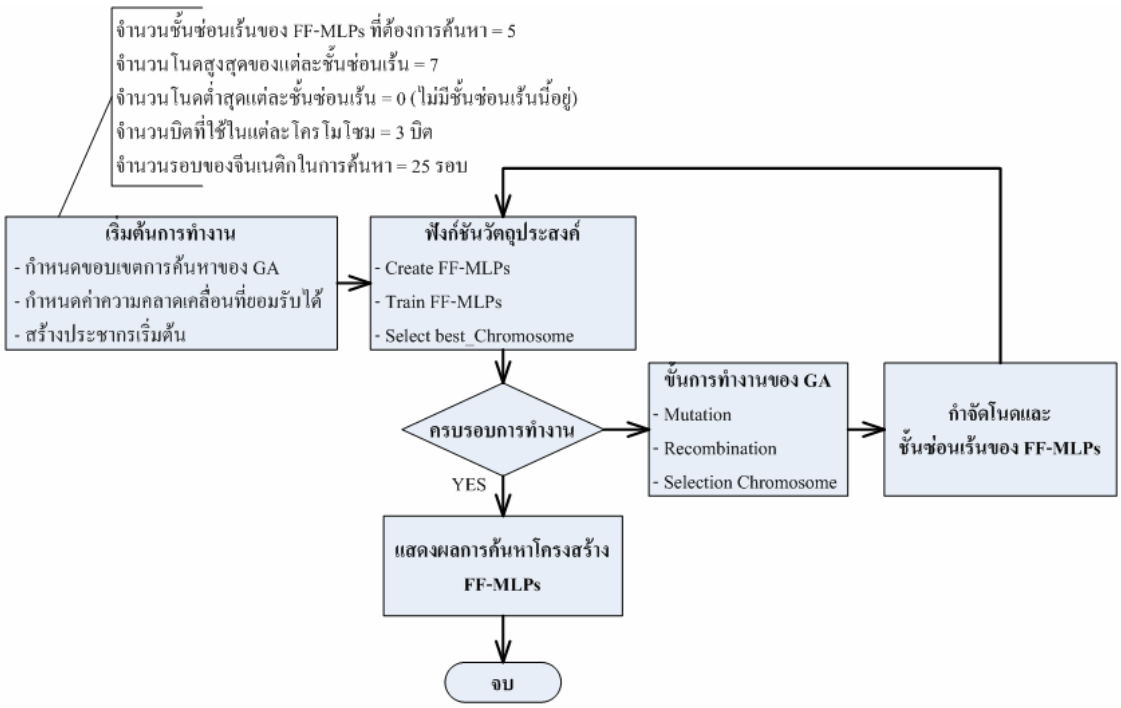
กรณีที่ 1 สำหรับกรณีที่ค่า OffError น้อยกว่าค่า AcceptError กรณีนี้คือ โครโมโซมชุดนี้สามารถสร้างโครงข่ายประสาทเทียมที่ให้ค่าการทดสอบดีกว่าระดับที่ตั้งไว้เหมาะสำหรับใช้สืบทอดพันธุกรรมต่อไป ค่าฟังก์ชันจึงให้ความสำคัญกับชั้นซ่อนเร้นที่เป็น 0 หรือลดจำนวนชั้นซ่อนเร้นลงเป็นลำดับแรกและจำนวนโนดรวมเป็นลำดับถัดลงมา

ยกตัวอย่าง

- โครโมโซมตัวที่ 1 มีโครงสร้าง [4 2 1 1 1 0 2] จะให้ค่า k_1 เท่ากับ 11×10^6
- โครโมโซมตัวที่ 2 มีโครงสร้าง [4 2 1 2 0 0 2] จะให้ค่า k_1 เท่ากับ 11×10^5

จากค่า ObjVSel ทั้งสองเมื่อทำการคัดเลือกสายพันธุ์โดยวิธีการจัดอันดับโครโมโซมตัวที่ 2 จะมีโอกาสในการให้กำเนิดประชากรรุ่นลูกจะสูงกว่า

กรณีที่ 2 สำหรับกรณีที่ค่า OffError มากกว่าค่า AcceptError กรณีนี้คือ โครโมโซมชุดนี้ยังไม่สามารถสร้างโครงข่ายประสาทเทียมที่ให้ค่าการทดสอบดีกว่าระดับที่ตั้งไว้ ค่าฟังก์ชันจึงให้ความสำคัญกับค่า OffError ที่คำนวณได้แต่ต้องไม่ดีกว่าการคำนวณด้วยในกรณีที่ 1 ดังนั้นค่าของฟังก์ชันจึงเท่ากับค่า OffError คูณด้วยค่าคงที่



รูปที่ 5.3 ขั้นตอนการทำงานของโปรแกรมในการค้นหาโครงสร้างของ FF-MLPs

ค่าคงที่นี้จะอ้างอิงจากกรณีที่ 1 โดยโครโมโซมใหญ่สุดที่เป็นไปได้คือ [4 7 7 7 7 7 2] จึงคำนวณได้จากผลรวมของโครโมโซมใหญ่สุดที่เป็นไปได้คือ 41 ผลรวมจำนวนของชั้นซ่อนเร้นที่ไม่เป็นศูนย์มีจำนวน 7 ชั้น ดังนั้นค่าคงที่สำหรับกรณีนี้ คือ $41x10^7$

การทำงานของจีนเนติกเพื่อค้นหาโครงสร้างของโครงข่ายประสาทเทียมแสดงไว้ในรูปที่ 5.3 ขั้นตอนการทำงานของจีนเนติก มีดังนี้

1. กำหนดค่าตัวแปรเริ่มต้นสำหรับจีนเนติก ได้แก่
   - จำนวนชั้นซ่อนเร้นสูงสุดที่ต้องการเริ่มค้นหา
   - จำนวนโนดสูงสุดในแต่ละชั้นซ่อนเร้น
   - จำนวนโนดต่ำสุดในแต่ละชั้นซ่อนเร้น
   - จำนวนบิตเพื่อบอกความละเอียดของตัวแปร
   - จำนวนรอบของการทำงานของจีนเนติก
2. กำหนดตัวแปรเริ่มต้นในการใช้จีนเนติกหาโครงสร้างของโครงข่ายประสาทเทียม เช่น
   - ค่า AcceptError

- จำนวนรอบการฝึกสอนโครงข่ายประสาทเทียมของประชากรแต่ละตัว

3. ประเมินประชากรเพื่อจัดลำดับด้วยฟังก์ชันวัตถุประสงค์
4. ดำเนินการตามกระบวนการของจีนเนติก เช่น
   - การทำมิวเตชัน (mutation)
   - การทำครอสโอเวอร์ (crossover)
   - การแทนที่ (replacement) ประชากรรุ่นก่อนด้วยประชากรรุ่นลูกหลาน
5. กำจัดโนดและชั้นซ่อนเร้นของโครงข่ายประสาทเทียม
6. ทำซ้ำขั้นตอนที่ 3, 4 และ 5 จนกว่าจะครบรอบการทำงานของจีนเนติก

การกำจัดชั้นซ่อนเร้นบางชั้นออก คือการบังคับให้บิตที่แทนจำนวนโนดของชั้นซ่อนเร้นนั้นเป็นศูนย์มีหลักสำคัญคือการสร้างอาร์เรย์เพื่อมาคูณกับโครโมโซม การทำงานของจีนเนติกจะกำหนด Masking_Array ขึ้นมาเพื่อนำมาคูณกับโครโมโซมจะทำให้ชั้นซ่อนเร้นบางชั้นหายไป ยกตัวอย่างค่า Masking_Array ดังความสัมพันธ์ 5.7 และความสัมพันธ์ 5.8

Masking_Array = [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]; (5.7)

Masking_Array = [1 1 1 0 0 0 1 1 1 1 1 1 1 1 1]; (5.8)

ความสัมพันธ์ 5.7 เป็นค่าเริ่มต้นเมื่อจีนเนติกเริ่มทำงานจะกำหนดให้ทุกบิตมีค่าเป็น 1 ความยาวของ Masking_Array จะมีจำนวนบิตเท่ากับโครโมโซมหนึ่งตัว หากผลการประเมินด้วยฟังก์ชันวัตถุประสงค์แล้วให้ค่า OffError น้อยกว่า AcceptError และจำนวนโนดในชั้นซ่อนเร้นที่สองเป็น 0 จะทำการกำหนด Masking_Array ใหม่ดังความสัมพันธ์ 5.8

5.3 ผลการค้นหาโครงสร้างของ FF-MLPs และอภิปราย

การทดสอบแบ่งออกเป็น 2 ส่วนคือ ส่วนที่ 1 เป็นการหาโครงสร้างของ FF-MLPs ที่เหมาะสมสำหรับการจำลองพื้นที่ห้องทดสอบ และส่วนที่ 2 คือการนำ FF-MLPs ที่ได้ไปใช้แทนโครงสร้างเดิมเพื่อเปรียบเทียบประสิทธิภาพในการระบุตำแหน่งตัวเอง ดังรายละเอียดต่อไปนี้

ส่วนที่ 1 การหาโครงสร้างของ FF-MLPs ที่เหมาะสม

การหาโครงสร้าง FF-MLPs ที่เหมาะสมนี้โดยให้คอมพิวเตอร์ส่วนบุคคลทำการทดสอบจำนวน 120 ตัวอย่าง กำหนดให้ปรับเปลี่ยนพารามิเตอร์สองตัว พารามิเตอร์ตัวแรกคือเปอร์เซ็นต์การ

ทำครอสโอเวอร์ที่ 0.7 และ 0.9 พารามิเตอร์ตัวที่สองคือค่า AcceptError ที่ 8, 10 และ 12 แต่ละชุดให้ทำการทดสอบจำนวน 20 ตัวอย่าง ส่วนพารามิเตอร์ที่กำหนดให้เหมือนกัน คือ จำนวนรอบการทำงานของจีนเนติกเท่ากับ 25 จำนวนประชากรแต่ละรุ่นเท่ากับ 25 ความละเอียดของตัวแปรเท่ากับ 3 และจำนวนรอบการฝึกสอนโครงข่ายประสาทเทียมของประชากรแต่ละตัวเท่ากับ 4

นำผลการค้นหาทั้ง 120 ตัวอย่างมาเรียงลำดับตาม FF_Ranking และ %OffError ตารางที่ 5.1 นำผลการเรียงลำดับมาแสดงเฉพาะ 10 ลำดับแรก ผลการทำงานได้โครงสร้างดีที่สุดคือตัวอย่างทดสอบหมายเลข 2-52 เป็นโครงข่ายประสาทเทียมที่มีชั้นซ่อนเร้น 1 ชั้นจำนวนโนด 4 โนด แบบจำลองโครงข่ายประสาทเทียมที่ได้เมื่อทดสอบแล้วให้ค่า OffError 0.2 เมตรที่ 5.6777 เปอร์เซ็นต์หรือได้จำนวนจุดทดสอบที่มีระยะคลาดเคลื่อนมากกว่า 0.2 เมตร จำนวน 5.6777 เปอร์เซ็นต์จากจำนวนจุดทดสอบ 546 จุด ผลการทำงานของตัวอย่างทั้งหมดสามารถดูได้จาก ภาคผนวก ข. ตารางที่ ข.1

หากพิจารณาเฉพาะผลของการทำงานที่ให้จำนวนชั้นซ่อนเร้นเท่ากับ 1 ตามตารางที่ 5.2 เพื่อยืนยันการทำงานของจีนเนติกอัลกอริทึมว่าสามารถลดโครงสร้างของโครงข่ายประสาทเทียม ได้ ตัวเลขในตารางจะแสดงจำนวนของ FF-MLPs ที่มีชั้นซ่อนเร้นหนึ่งชั้นจากจำนวนตัวอย่างที่ทำการทดสอบ 20 ตัวอย่าง ซึ่งบอกได้ว่าจีนเนติกอัลกอริทึมที่พัฒนาขึ้นสามารถลดขนาดโครงสร้างของ FF-MLPs ลง ส่วนรายละเอียดการทดสอบแต่ละตัวอย่างสามารถดูได้จาก ภาคผนวก ข. ตารางที่ ข.2 และตารางที่ ข.3

ตารางที่ 5.1 ผลการทดสอบเมื่อเรียงลำดับตาม FF-Ranking และ %OffError เฉพาะ 10 ลำดับแรก

Max_Gen=25, nPopulation=25, n_Bit=3, FF-MLPs Train_Loop=4						
Item	PC	AcceptErro r	P_Crossove r	%OffError	FF_ANN	FF_Ranking
1	2 52	12	0.7	5.6777	4 0 4 0 0 0 2	100
2	2 27	8	0.7	7.6923	4 0 0 0 4 0 2	100
3	3 49	12	0.9	8.6081	4 0 4 0 0 0 2	100
4	1 44	12	0.9	11.3553	4 4 0 0 0 0 2	100
5	1 45	12	0.9	11.3553	4 4 0 0 0 0 2	100
6	1 46	12	0.9	11.3553	4 4 0 0 0 0 2	100
7	2 24	10	0.7	0.1832	4 5 0 0 0 0 2	110
8	3 27	8	0.9	0.1832	4 0 0 0 0 5 2	110
9	2 22	10	0.7	0.3663	4 0 0 0 0 5 2	110
10	2 46	12	0.7	0.3663	4 5 0 0 0 0 2	110

OffError	จำนวนจุดทดสอบที่ให้ระยะคลาดเคลื่อนมากกว่า 0.2 เมตร
PC	ชื่อของตัวอย่างที่ทดสอบ
AcceptError	เปอร์เซ็นต์ของจำนวนจุด OffError มากสุดที่ยอมรับได้
P_Crossover	ค่าการทำครอสโอเวอร์ของจีนเนติกอัลกอริทึม
%OffError	เปอร์เซ็นต์ของจำนวนจุด OffError จากจุดที่ทดสอบทั้งหมด 546 จุด
FF-Ann	โครงสร้างของ FF-MLPs ที่ได้จากการทำงานของจีนเนติกอัลกอริทึม
FF_Ranking	การคำนวนลำดับของโครงสร้าง FF-MLPs โดย คำนวนจาก FF_RankScale = 10 ^ จำนวนชั้นซ่อนเร้นที่ไม่เป็นศูนย์ FF_Ranking = ผลรวมของโนดในแต่ละชั้น x FF_RankScale

ตารางที่ 5.2 ผลการค้นหาโครงสร้างของ FF-MLPs ที่ให้จำนวนชั้นซ่อนเร้นเท่ากับ 1

	ค่าการทำครอสโอเวอร์= 0.9	ค่าการทำครอสโอเวอร์= 0.7
AcceptError = 12	20/20	17/20
AcceptError = 10	17/20	16/20
AcceptError = 8	16/20	18/20

ส่วนที่ 2 การทดสอบโครงสร้างของ FF-MLPs ที่ได้จากจีนเนติก

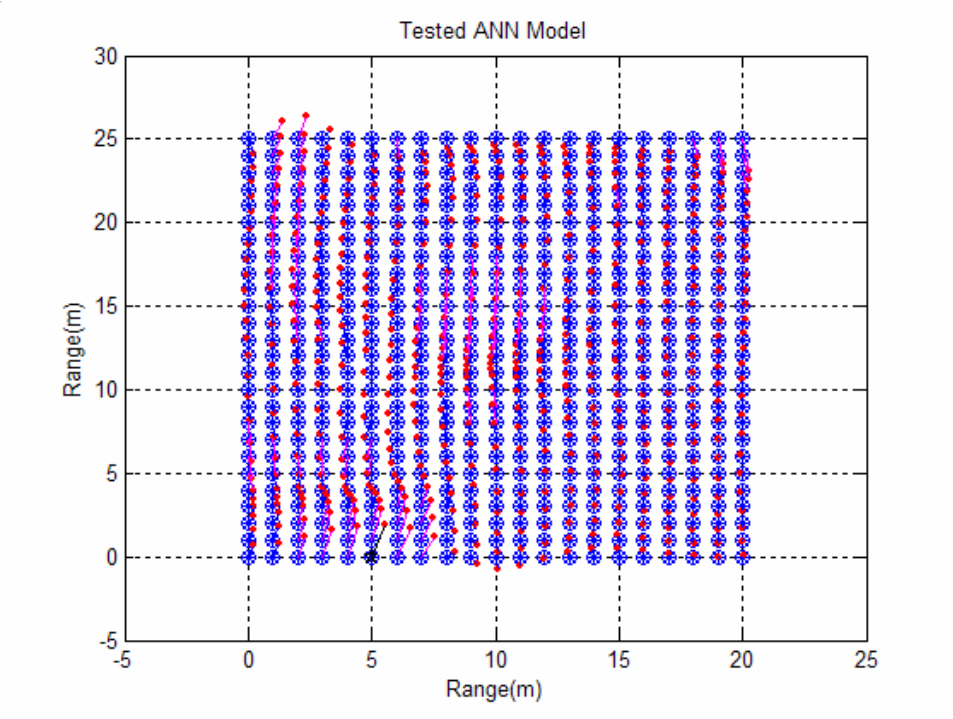
การทดสอบโดยการแทนที่แบบจำลองโครงข่ายประสาทเทียมที่ได้จากการทำงานของโปรแกรมจีนเนติกเข้าไปในระบบระบุตำแหน่งเพื่อดูประสิทธิภาพของระบบโดยนับจำนวนจุดที่มีระยะคลาดเคลื่อนจากจากจุดทดสอบเกิน 1 เมตร

จากรูปที่ 5.4 เป็นผลการทดสอบด้วย FF-MLPs โครงสร้างเดิมจากการทดสอบในบทที่ 4 และรูปที่ 5.6 แสดงผลการทดสอบด้วย FF-MLPs โครงสร้างใหม่ที่ได้จากการทำงานของจีนเนติก ซึ่งให้ผลการทำงานที่ดีขึ้นจากเดิม

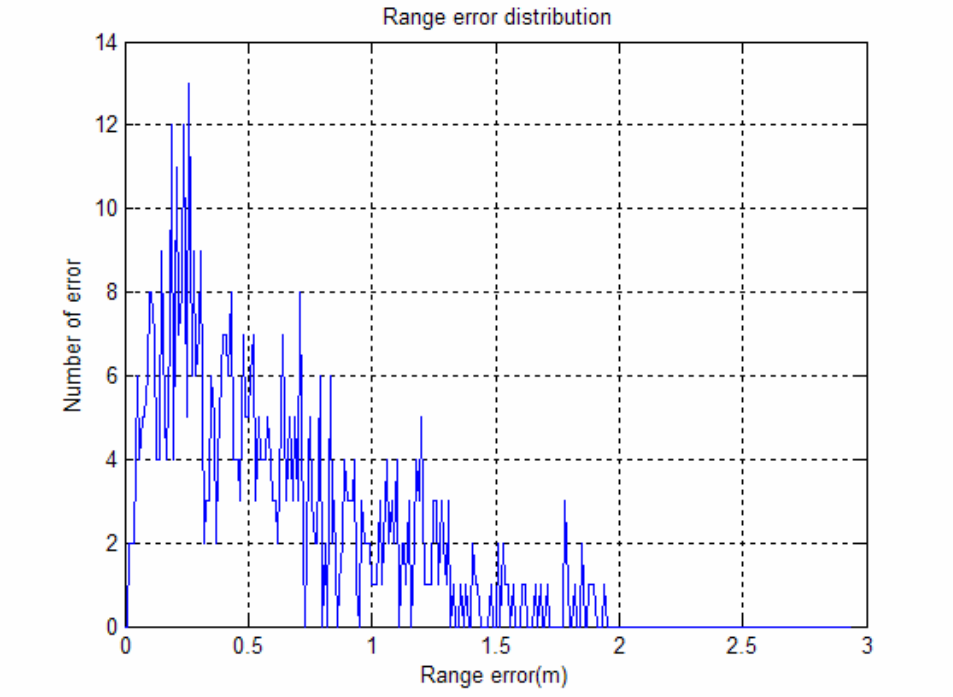
ผลการทดสอบที่แสดงในตารางที่ 5.3 จะเห็นว่าโครงสร้าง FF-MLPs ที่ได้จากจีนเนติกให้ผลการทดสอบดีกว่า คือ จำนวนจุดที่ระยะเบี่ยงเบนจากจุดทดสอบน้อยกว่า 1.0 เมตรเท่ากับ 499 จุดจากจุดทดสอบ 546 จุดหรือ 91.39 เปอร์เซ็นต์มีระยะเบี่ยงเบนจากจุดทดสอบเฉลี่ย 0.4566 เมตร และมีระยะเบี่ยงเบนจากจุดทดสอบสูงสุด 1.9440 เมตร

ตารางที่ 5.3 ผลการทดสอบเพื่อเปรียบเทียบประสิทธิภาพของ FF-MLPs

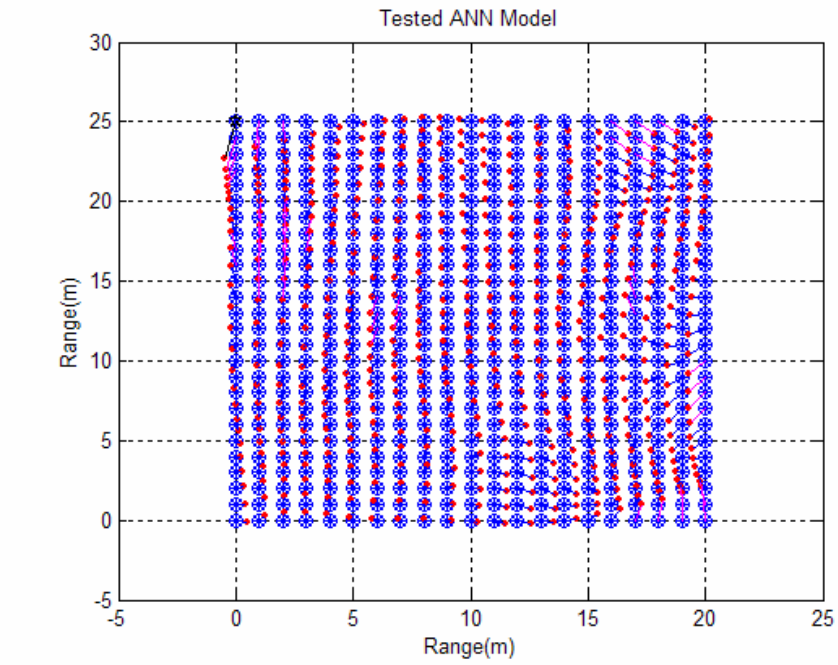
โครงสร้าง FF-MLPs ที่ใช้ทดสอบ	โครงสร้างเดิม จากบทที่ 4	โครงสร้างใหม่ จากโปรแกรมจีนเนติก
จำนวนจุดที่ระยะเบี่ยงเบนน้อยกว่า 1.0 เมตร	449/546 (82.23%)	499/546 (91.39%)
จำนวนจุดที่ระยะเบี่ยงเบนน้อยกว่า 0.5 เมตร	285/546 (52.20%)	353/546 (64.65%)
จำนวนจุดที่ระยะเบี่ยงเบนน้อยกว่า 0.2 เมตร	101/546 (18.50%)	157/546 (28.75%)
ระยะเบี่ยงเบนจากจุดทดสอบเฉลี่ย(เมตร)	0.58	0.54
ระยะเบี่ยงเบนจากจุดทดสอบสูงสุด(เมตร)	1.94	2.38
ตำแหน่งที่เกิดการเบี่ยงเบนสูงสุด	(5, 0)	(0, 25)



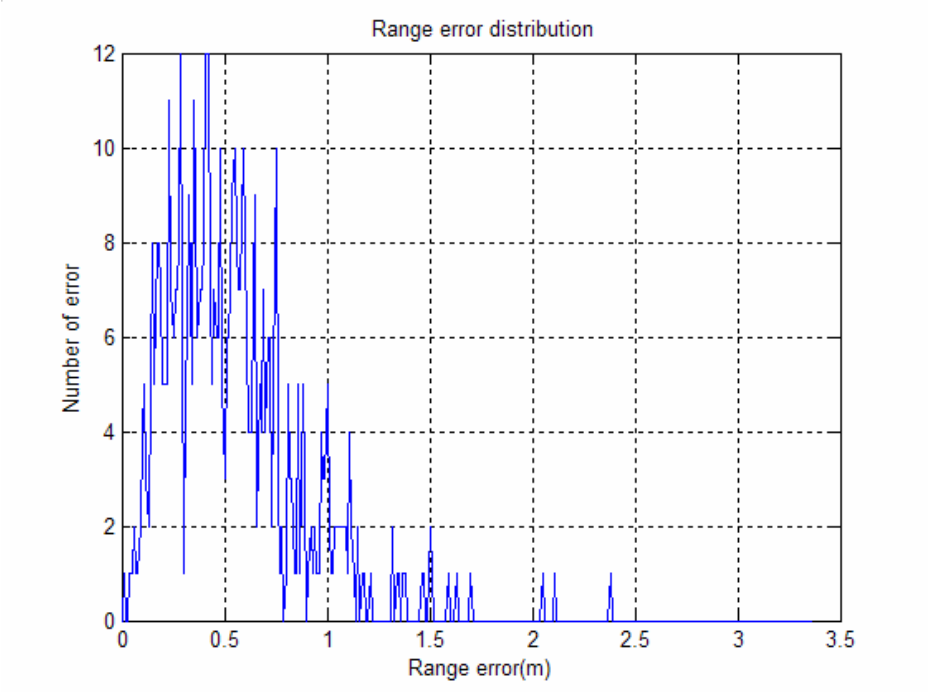
รูปที่ 5.4 ผลการทดสอบด้วย FF-MLPs โครงสร้างเดิมจากการทดสอบในบทที่ 4



รูปที่ 5.5 การกระจายของระยะคลาดเคลื่อนจากการทดสอบโครงสร้างเดิมในบทที่ 4



รูปที่ 5.6 ผลการทดสอบด้วย FF-MLPs โครงสร้างใหม่ที่ได้จากการทำงานของจีนเนติกอัลกอริทึม



รูปที่ 5.7 การกระจายของระยะคลาดเคลื่อนจากการทดสอบโครงสร้าง
ได้จากการทำงานของจีนเนติกอัลกอริทึม

5.4 สรุป

จากที่กล่าวมาทั้งหมดข้างต้น ได้นำเสนอถึงวิธีการและหลักการของจีนเนติกอัลกอริทึมสำหรับการค้นหาโครงสร้าง FF-MLPs การหาเริ่มต้นที่โครงสร้างที่มีใหญ่แบบ [4 7 7 7 7 7 2] ประกอบด้วยชั้นซ่อนเร้น 5 ชั้นๆละ 7 โนด, ชั้นอินพุต 4 โนดจากจุดเข้าถึง 4 จุด และชั้นเอาต์พุต 2 โนด คือตำแหน่ง (x, y) ผลการทำงานสามารถลดขนาดโครงสร้างลงได้เล็กสุดสำหรับชุดข้อมูลนี้เป็นแบบ [4, 4, 2] หรือเหลือชั้นซ่อนเร้น 1 ชั้นจำนวน 4 โนด ภายใต้เงื่อนไขค่าความคลาดเคลื่อนที่กำหนดไว้

การหาโครงสร้างของ FF-MLPs ด้วยจีนเนติกอัลกอริทึมให้ผลเป็นที่น่าพอใจเป็นอย่างยิ่ง โดยเฉพาะการนำเสนอแนวคิดในการใช้จีนเนติกอัลกอริทึมในการหาโครงสร้างของ FF-MLPs ซึ่งเป็นการยืนยันให้เห็นถึงจุดแข็งของการใช้วิธีการทางปัญญาประดิษฐ์ที่เรียกว่าจีนเนติกอัลกอริทึมกับงานวิจัยทางด้านวิศวกรรมรูปแบบหนึ่ง

บทที่ 6
บทสรุปและข้อเสนอแนะ

6.1 สรุป

งานวิจัยวิทยานิพนธ์นี้ดำเนินการศึกษาและพัฒนาระบบการระบุตำแหน่งภายในอาคารด้วยเทคนิควิธีเชิงปัญญาประดิษฐ์ โดยเลือกใช้โครงข่ายประสาทเทียมแบบ FF-MLPs ให้ทำงานร่วมกับ RBF เพื่อปรับปรุงให้มีประสิทธิภาพสูงสุด สำหรับงานวิจัยนี้ได้ใช้การค้นหาแบบจีนเนติกในการหาโครงสร้างของ FF-MLPs การดำเนินงานวิจัยวิทยานิพนธ์ดังกล่าวสำเร็จลุล่วงตามวัตถุประสงค์ โดยสามารถสรุปผลการศึกษาวิจัยและพัฒนาทางวิศวกรรมเป็นข้อสรุปได้ดังต่อไปนี้

ทำการทดสอบเพื่อระบุตำแหน่งภายใน ห้องปฏิบัติการวงจรและอุปกรณ์และห้องปฏิบัติการไมโครโพรเซสเซอร์ อาคารเครื่องมือ 3 ศูนย์เครื่องมือวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยเทคโนโลยีสุรนารี การทำงานประกอบด้วยขั้นตอนการวางจุดเข้าถึง 4 จุดที่ครอบคลุมบริเวณพื้นที่ทดสอบ เก็บข้อมูลความแรงของสัญญาณเทียบกับตำแหน่ง (x, y) ขนาดพื้นที่ทดสอบ 20 เมตร x 25 เมตร เมื่อกำหนดจุดทดสอบที่ระยะทุก ๆ 1 เมตรสามารถกำหนดจุดทดสอบทั้งหมด 546 จุด เก็บข้อมูลจำนวน 100 ค่าต่อจุดและใช้ค่าเฉลี่ยของความแรงสัญญาปืนตัวแทนข้อมูล เก็บข้อมูลจำนวน 2 ชุดสำหรับเป็นชุดข้อมูลฝึกสอนและสำหรับเป็นชุดข้อมูลทดสอบ

การทดสอบโดยใช้โครงข่ายประสาทเทียมแบบ FF-MLPs ทำงานร่วมกับ RBF หน้าที่ของ FF-MLPs คือเป็นแบบจำลองโครงสร้างพื้นที่ทดสอบสามารถหาโครงสร้างได้จากจีนเนติกอัลกอริทึมได้โครงสร้างดีที่สุดสำหรับการทดสอบนี้ประกอบด้วย ชั้นอินพุต 4 โนดจากจุดเข้าถึง 4 จุด ชั้นซ่อนเร้น 1 ชั้นจำนวน 4 โนด และชั้นเอาต์พุต 2 โนดคือตำแหน่ง (x, y) หน้าที่ของ RBF คือการปรับปรุงข้อมูลสำหรับทดสอบก่อนทำการทดสอบระบุตำแหน่ง

ผลการทดสอบระบบระบุตำแหน่งภายในอาคารด้วยเทคนิควิธีเชิงปัญญาประดิษฐ์ด้วยการอาศัยความแรงสัญญาณจากระบบโครงข่ายแลนไร้สายในการวัดระยะทาง สามารถระบุพิกัดตำแหน่งได้ถูกต้อง 64.65 เปอร์เซ็นต์ในระยะคลาดเคลื่อนน้อยกว่า 0.5 เมตร และสามารถระบุพิกัดตำแหน่งได้ถูกต้อง 91.39 เปอร์เซ็นต์ในระยะคลาดเคลื่อนน้อยกว่า 1 เมตร จากข้อมูลทั้งหมดพบว่ามีความคลาดเคลื่อนเฉลี่ยเท่ากับ 0.4566 เมตร ความคลาดเคลื่อนต่ำสุดเท่ากับ 0.0012 เมตรและความคลาดเคลื่อนสูงสุดเท่ากับ 1.9440 เมตร

6.2 ข้อเสนอแนะ

ปัญหาที่เกิดขึ้นในการทำงานวิจัยในวิทยานิพนธ์เล่มนี้ ได้แก่

1. เอกสารและข้อมูลที่เกี่ยวข้องกับวิธีการคำนวณรวมถึงหน่วยงานที่ศึกษาวิจัยเรื่องการประมาณค่าตำแหน่ง (position location) มีอยู่น้อยมากในประเทศไทย จึงควรมีการจัดหาเอกสารหรือการเข้าร่วมประชุมสัมมนาขององค์กรต่าง ๆ ที่วิจัยในเรื่องที่เกี่ยวข้อง
2. บุคลากรที่สนใจในการทำงานต่อเนื่องมีอยู่น้อยมากทำให้งานวิจัยเกิดการขาดช่วงและไม่ต่อเนื่อง
3. จากข้อกำหนดเบื้องต้นที่ว่าสภาพแวดล้อมไม่มีการเปลี่ยนแปลง ในความเป็นจริงอาคารที่ทำการทดสอบเป็นอาคารเรียนและมีการปรับเปลี่ยนโครงสร้างบ้างดังนั้นการเก็บข้อมูลชุดที่ 1 สำหรับฝึกสอน เก็บข้อมูลชุดที่ 2 สำหรับทดสอบเรียบร้อยแล้วเมื่อต้องการทดสอบในสภาพแวดล้อมจริงจะทำให้เกิดข้อผิดพลาดสูงขึ้น

งานที่ควรได้รับการศึกษาหรือพัฒนาต่อไปในอนาคต คือ

1. ข้อมูลที่ใช้ฝึกสอนและทดสอบโครงข่ายประสาทเทียมเดิมเป็นความแรงของสัญญาณเพียงอย่างเดียว ควรเพิ่มลำดับของการเก็บข้อมูลเข้าไปด้วยเพื่อเพิ่มความถูกต้องของการระบุตำแหน่ง โดยลำดับของการเก็บข้อมูลคือเส้นทางการเดินขณะทำการเก็บข้อมูลฝึกสอนและทดสอบในความเป็นจริงแล้วการนำทางหุ่นยนต์กรณีเดินภายในอาคารเส้นทางการเดินจะมีการกำหนดไว้ชัดเจนอยู่แล้วดังนั้นการเก็บข้อมูลโดยเน้นบริเวณทางเดินและลักษณะของสัญญาณจากจุดก่อนนี้ และจุดปัจจุบันอาจจะสามารถเพิ่มความถูกต้องของการระบุพิกัดตำแหน่งได้
2. หากพัฒนาเป็นระบบนำวิธีหุ่นยนต์อัตโนมัติขึ้น ในขั้นตอนการฝึกสอน RBF ตำแหน่งจริงสำหรับอ้างอิงอาจมาจากเซ็นเซอร์บริเวณทางเดินหรือการเป็นการนำทางหุ่นยนต์หนึ่งรอบก่อนใช้งานจริง
3. ควรมีการปรับปรุงอัลกอริทึมในกรณีสภาพแวดล้อมมีการเปลี่ยนแปลงไป
4. ควรศึกษากรณีพื้นที่ทดสอบรูปทรงไม่เป็นสี่เหลี่ยมหรือกรณีการวางตำแหน่งของจุดเข้าถึงรูปทรงไม่เป็นสี่เหลี่ยม เพื่อหาความสัมพันธ์ของพื้นที่ทดสอบกับลักษณะโครงสร้างของโครงข่ายประสาทเทียม

รายการอ้างอิง

Bahl, P., & Padmanabhan, V.N., RADAR: An in-building RF-based user location and tracking system, **IEEE Infocom 2000**, Tel Aviv, Israel, 26-30 March 2000, vol. 2, pp. 775-784.

Battiti, R., Nhat, T.L., & Villani, A., "Location-aware computing: a neural network model for determining location in wireless LANs, **Technical Report DIT-02-0083**, February 2002

Fox, D., Markov Localization: A probabilistic Frame work for Mobile Robot Localization and Navigation, PhD thesis,Institute of Computer Science III, 1998.

Hightower, J., & Borriello, G., Location systems for ubiquitous computing, **IEEE Computer**, August 2001, 34, (8), pp. 57-66

Ladd, A.M., Bekris, K.E., Rudys, A., Marceau, G., Kavraki L.E., & Dan, S., Robotics-based location sensing using wireless Ethernet, **Eighth ACM Int. Conf. on Mobile Computing & Networking (MOBICOM)**, Atlanta, Georgia, US, 23-28 September 2002, pp.227-238

Li, B., Dempster, A.G., Rizos, C., & Barnes, J., Hybrid method for localization using WLAN, **Spatial Sciences Conference**, Melbourne, Australia, 12-16 September 2005, 341-350, CD-ROM procs.

Moustafa Youssef and Ashok Agrawala, On the Optimality of WLAN Location Determination Systems, **Communication Networks and Distributed Systems Modeling and Simulation Conference**, January 18-24 2004, San Diego, California.

Negnevitsky, Michael., **Artificial intelligence : a guide to intelligent systems**, Harlow, England; New York: Addison-Wesley, 2002.

P. Myllymaki, T. Roos, H. Tirri, P. Misikangas, J. Sievanen, A Probabilistic Approach to WLAN User Location Estimation, **The Third IEEE Workshop on Wireless LANs**, 2001.

Pandya, D., Jain, R., & Lupu, E., Indoor location estimation using multiple wireless technologies, **14th IEEE Int. Symp. on Personal, Indoor, & Mobile Radio Communications (PIMRC)**, Beijing, China, 7-10 September 2003, vol. 3, pp. 2208-2212.

Roos, T., Myllymaki, P., Tirri, H., Misikangas, P., & Sievanan, J., A probabilistic approach to WLAN user location estimation, **International Journal of Wireless Information Networks**, 9(3), July 2002.

Saha, S., Chaudhuri, K., Sanghi, D., & Bhagwat, P., Location determination of a mobile device

using IEEE 802.11b access point signals, **IEEE Wireless Communications &Networking Conference (WCNC)**, New Orleans, Louisiana, US, 16-20 March 2003, vol. 3, pp.1987-1992.

Simmons, R., & Koenig, S., Probabilistic robot navigation in partially observable environments, **The International Joint Conference on Artificial Intelligence (IJCAI'95)**, Montreal, Canada, 20-25 August 1995, pp. 1080-1087.

Simon Haykin., Neural networks: a comprehensive foundation, **New York: Maxwell Macmillan International**, c1994. Satish Kumar. Neural networks: a classroom approach ", Boston: McGraw Hill, c2005.

Smailagic, A., and Kogan, D., Location Sensing and Privacy in a Context-aware Computing Environment, **IEEE Wireless Communications**, Vol. 9 No. 5, October 2002

Wang, Y., Jia, X., Lee, H.K., and Li, G.Y., An indoor wireless positioning system based on WLAN infrastructure, **6th Int. Symp. on Satellite Navigation echnology Including Mobile Positioning & Location Services**, Melbourne, Australia, 22-25 July 2003, CD-ROM proc., paper 54.

Youssef, M., Agrawala A., & Shankar, A.U., WLAN location determination via clustering and probability distributions, **IEEE Int. Conf. on Pervasive Computing & Communications (PerCom) 2003**,Texas, US, 23-26 March 2003, pp.143-150.

ภาคผนวก ก.

การใช้จีนเนติกอัลกอริทึมเพื่อหาโครงสร้างของ FF-MLPs

การใช้จีนเนติกอัลกอริทึมเพื่อหาโครงสร้างของ FF-MLPs

1. บทนำ

โดยทั่วไปโครงสร้างของ FF-MLPs (Feed-Forward Multi-Layer Perceptrons) หาได้ด้วยวิธีการลองผิดลองถูก (trial and error) ไม่มีข้อกำหนดแน่นอนตายตัวในการเลือกโครงสร้าง การใช้งาน FF-MLPs โดยทั่วไปจะกำหนดโครงสร้างขึ้นมาทำการฝึกสอนและทดสอบหากข้อผิดพลาดที่ได้จากโครงข่ายประสาทเทียมยอมรับได้ก็ใช้โครงสร้างนั้นได้เลย แต่ถ้าข้อผิดพลาดยังสูงก็ปรับโครงสร้าง FF-MLPs ไปเรื่อย ๆ จนกว่าข้อผิดพลาดลดลง

บทความนี้นำเสนอการใช้วิธีทางปัญญาประดิษฐ์ที่เรียกว่าจีนเนติกอัลกอริทึม (genetic algorithm) ในการช่วยหาโครงสร้างของ FF-MLPs การทดสอบนี้ยกตัวอย่างการหาแบบจำลองของพื้นที่ด้วยความแรงสัญญาณวิทยุจากจุดเข้าถึง (access point) ของโครงข่ายแลนไร้สาย (wireless LAN) ข้อมูลที่ใช้ฝึกสอนและทดสอบโครงข่าย อินพุตมาจากจุดเข้าถึง 4 จุดและเอาต์พุตคือตำแหน่ง (x, y) ดังนั้นโครงสร้างของ FF-MLPs จะประกอบด้วยชั้นอินพุตจำนวนโนด 4 โนด ชั้นเอาต์พุตจำนวนโนด 2 โนดคือตำแหน่ง (x, y) ส่วนจำนวนชั้นซ่อนเร้นชั้นและจำนวนโนดในแต่ละชั้นคือข้อมูลที่ต้องการให้จีนเนติกค้นหา

2. หลักการของจีนเนติกอัลกอริทึม

หลักการของจีนเนติกอัลกอริทึมได้แบ่งการอธิบายออกเป็น 3 หัวข้อที่สำคัญ ๆ โดยจะนำเสนอดังต่อไปนี้

2.1 กระบวนการของจีนเนติกอัลกอริทึม

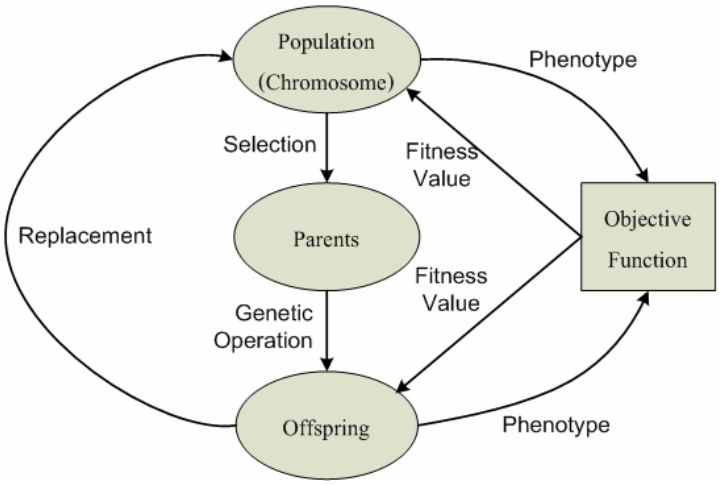
กระบวนการที่สำคัญของจีนเนติกอัลกอริทึม ประกอบไปด้วย 3 กระบวนการ คือ

การคัดเลือกสายพันธุ์ (Selection) คือ ขั้นตอนสำหรับการคัดเลือกประชากรที่ดีจากเงื่อนไขที่กำหนดในระบบเพื่อนำไปเป็นต้นกำเนิดสายพันธุ์ในการให้กำเนิดลูกหลานในรุ่นถัดไป

*ปฏิบัติการทางสาย*พันธุ์ (Genetic operation) คือ กรรมวิธีการเปลี่ยนแปลงโครโมโซมด้วยวิธีการทางสายพันธุ์ เป็นขั้นตอนการสร้างลูกหลานจากต้นกำเนิดสายพันธุ์ ซึ่งจะกล่าวรายละเอียดในหัวข้อที่ 2.3

การแทนที่ (Replacement) คือ ขั้นตอนการนำลูกหลานที่ได้จากต้นกำเนิดสายพันธุ์ไปแทนที่ประชากรเก่าในรุ่นก่อน

หลักการทำงานแสดงเป็นภาพรวมวัฏจักรการทำงานของจีนเนติกอัลกอริทึมดังรูปที่ ก.1



รูปที่ ก.1 วัฏจักรการทำงานของจีนเนติกอัลกอริทึม

จากรูปที่ ก.1 รายละเอียดต่าง ๆ ขององค์ประกอบวัฏจักรการทำงานของจีนเนติกอัลกอริทึมอธิบายได้ดังต่อไปนี้

ประชากร (Population) ประกอบด้วยกลุ่มของโครโมโซม (chromosome) ซึ่งเป็นตัวแทนของคำตอบในระบบที่ต้องการค้นหา

ต้นกำเนิดสายพันธุ์ (Parents) เป็นกลุ่มประชากรที่ถูกคัดเลือกสำหรับให้กำเนิดสายพันธุ์ใหม่ในรุ่นถัดไป ดังนั้นประชากรกลุ่มนี้จึงเปรียบเสมือนพ่อแม่ในขณะที่สายพันธุ์ใหม่จะเปรียบเสมือนลูกหลานนั่นเอง

สายพันธุ์ใหม่ (Offspring) หรือที่เรียกว่าลูกหลานเป็นประชากรกลุ่มใหม่ที่ได้รับการถ่ายทอดสายพันธุ์มาจากพ่อแม่โดยวิธีปฏิบัติการทางสายพันธุ์โดยคาดหวังที่จะได้รับสายพันธุ์ที่ดีที่สุดเพื่อถ่ายทอดต่อ ๆ กันในประชากรรุ่นถัดไป (next generation)

ฟังก์ชันวัตถุประสงค์ (objective function) เป็นกระบวนการสำหรับการประเมินผลคำตอบของระบบว่าดีหรือไม่ดีแค่ไหน ซึ่งจากแผนรูปที่ ก.1 โครโมโซมที่นำไปประเมินค่าด้วยฟังก์ชันวัตถุประสงค์จะต้องอยู่ในรูปแบบที่ระบบเข้าใจซึ่งเรียกว่ารูปแบบฟีโนไทป์ (phenotype) หลังจากการประเมินค่าผลคำตอบของระบบแล้ว (การประเมินค่าคำตอบของระบบขึ้นอยู่กับวัตถุประสงค์ของงานแต่ละงาน) จะส่งค่าความเหมาะสม (fitness value) เพื่อนำไปใช้เป็นเครื่องมือสำหรับตัดสินคัดเลือกโครโมโซมที่ดีเพื่อใช้สำหรับการสืบสายพันธุ์เป็นลูกหลานต่อไปโดยค่าความเหมาะสมดังกล่าวที่ให้กับโมโครโซมแต่ละตัวจะมีการเปรียบเทียบกันเองในกลุ่มประชากร

2.2 ขั้นตอนการทำงานของจีนเนติกอัลกอริทึม

การทำงานของจีนเนติกอัลกอริทึม แบ่งออกเป็นลำดับขั้นตอนต่าง ๆ 8 ขั้นตอนดังนี้

ขั้นตอนที่ 1 สร้างประชากรเริ่มต้น โดยปกติจะใช้การสุ่ม (random)

ขั้นตอนที่ 2 ประเมินค่าโครโมโซมของกลุ่มประชากรทั้งหมด ด้วยฟังก์ชันวัตถุประสงค์และเนื่องจากระบบไม่สามารถเข้าใจค่าของโครโมโซม ดังนั้นโครโมโซมจะต้องถูกถอดรหัสให้เป็นรูปแบบฟีโนไทป์ ก่อนที่จะถูกการประเมินด้วยฟังก์ชันวัตถุประสงค์

ขั้นตอนที่ 3 ประเมินผลคำตอบของระบบด้วยฟังก์ชันวัตถุประสงค์ และคำนวณหาค่าความเหมาะสมส่งกลับไปเพื่อใช้ในการคัดเลือกโครโมโซมที่ดีสำหรับการสืบสายพันธุ์

ขั้นตอนที่ 4 ใช้ค่าความเหมาะสมที่ได้จากการคำนวณในขั้นตอนที่ 3 เพื่อดำเนินการคัดเลือกโครโมโซมบางกลุ่มมาเป็นต้นกำเนิดสายพันธุ์

ขั้นตอนที่ 5 นำต้นกำเนิดสายพันธุ์มาทำการสร้างลูกหลาน ด้วยปฏิบัติการทางสายพันธุ์ ซึ่งโดยทั่วไปจะมีอยู่ 2 วิธี คือ การทำครอสโอเวอร์ (crossover) และการทำมิวเทชัน (mutation)

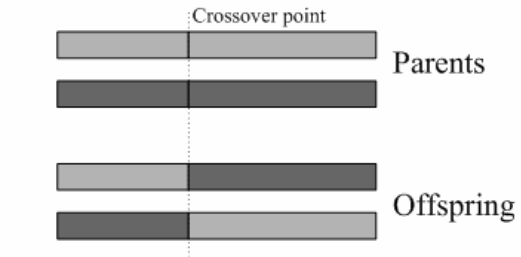
ขั้นตอนที่ 6 คำนวณค่าความเหมาะสมของโครโมโซมลูกหลาน ซึ่งใช้วิธีการเดียวกับขั้นตอนที่ 3

ขั้นตอนที่ 7 โครโมโซมในประชากรเดิมจะถูกแทนที่ด้วยลูกหลานที่ได้จากขั้นตอนที่ 5 ซึ่งประชากรเพียงบางส่วนเท่านั้นที่จะถูกแทนที่ด้วยกลวิธีเฉพาะสำหรับขั้นตอนของการแทนที่โดยใช่ค่าความเหมาะสมในการตัดสิน

ขั้นตอนที่ 8 เริ่มต้นทำซ้ำจากขั้นตอนที่ 2 ไปเรื่อย ๆ จนกระทั่งได้คำตอบที่ต้องการ

2.3 ปฏิบัติการทางสายพันธุ์ (genetic operation) ของจีนเนติกอัลกอริทึม

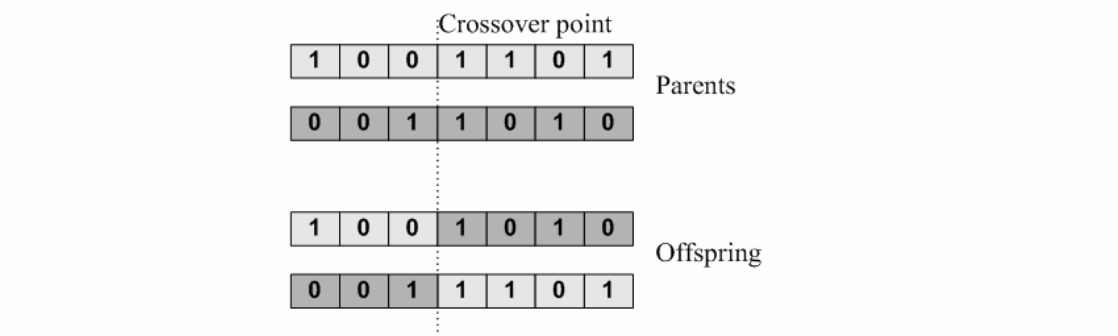
ปฏิบัติการทางสายพันธุ์เป็นการนำโครโมโซมต้นกำเนิดสายพันธุ์มาทำการเปลี่ยนแปลงเพื่อให้เกิดโครโมโซมใหม่ขึ้นกลายเป็นโครโมโซมลูกหลาน ขั้นตอนนี้เป็นขั้นตอนที่สำคัญอีกขั้นตอนหนึ่งในวัฏจักรการทำงานของจีนเนติกอัลกอริทึมดังรูปที่ ก.1 ซึ่งโดยทั่วไปปฏิบัติการทางสายพันธุ์จะมีอยู่ 2 วิธีหลัก ๆ ดังต่อไปนี้



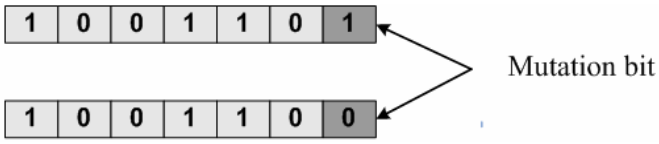
รูปที่ ก.2 การทำครอสโอเวอร์แบบจุดเดียว

การทำครอสโอเวอร์ (crossover) เป็นกรรมวิธีสำหรับการรวมตัวใหม่ของโครโมโซม (recombination operator) โดยทำการรวมส่วนย่อยระหว่างโครโมโซมต้นกำเนิดสายพันธุ์ตั้งแต่สองโครโมโซมขึ้นไปเพื่อให้กลายเป็นโครโมโซมลูกหลาน การทำครอสโอเวอร์ของจีนเนติกอัลกอริทึมมีอยู่หลายแบบด้วยกัน เช่น การทำครอสโอเวอร์แบบจุดเดียว (single - point crossover) การทำครอสโอเวอร์แบบสองจุด (double - point crossover) การทำครอสโอเวอร์แบบหลายจุด (multiple - point crossover) และการทำครอสโอเวอร์แบบสลับที่ (shuffle crossover) แต่ในงานวิจัยนี้ได้เลือกใช้วิธีการทำครอสโอเวอร์แบบจุดเดียวซึ่งการทำครอสโอเวอร์แบบจุดเดียวแสดงได้ดังรูปที่ ก.2

ตามหลักการของจีนเนติกอัลกอริทึมส่วนย่อยของโครโมโซมจะเรียกว่า ยีน ซึ่งในทางปฏิบัติยีนของโครโมโซมก็คือบิตในระบบตัวเลขของคอมพิวเตอร์ ซึ่งแสดงดังรูปที่ ก.3 ดังนี้



รูปที่ ก.3 การทำครอสโอเวอร์แบบจุดเดียวในรูปแบบของบิต

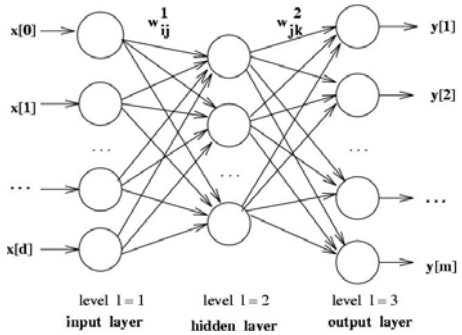


รูปที่ ก.4 การทำมิวเทชัน

การทำมิวเทชัน (mutation) เป็นวิธีการแปรผันยีนหรือส่วนย่อยของโครโมโซม ดังแสดงในรูปที่ ก.4 การทำมิวเทชันเป็นกระบวนการที่สำคัญอย่างหนึ่งที่ขาดไม่ได้ เพราะถ้าหากไม่จัดให้มีมิวเทชันอาจทำให้พลาดโอกาสที่จะค้นพบคำตอบที่ดีที่สุดในวงกว้าง (global solution) ของระบบ ซึ่งตามกระบวนการของจีนเนติกอัลกอริทึมไม่ควรหลีกเลี่ยนกระบวนการนี้ ควรจะมีทั้งกระบวนการทำ ครอสโอเวอร์และการทำมิวเทชันควบคู่กันไปทั้งสองกระบวนการ

3. โครงสร้างของ FF-MLPs: Feed-forward Multi-Layer Perceptrons

องค์ประกอบหลักของเครือข่ายแบบ FF-MLPs ได้แก่ ชั้นอินพุต (Input layer) ชั้นซ่อนเร้น (hidden layer) และชั้นเอาต์พุต (output layer) โดยจะมีการเชื่อมต่อระหว่างชั้นต่าง ๆ โนดในชั้นอินพุตจะส่งสัญญาณไปยังทุก ๆ โนดในชั้นซ่อนเร้นชั้นแรก และทุกโนดในชั้นซ่อนเร้นชั้นแรกจะส่งสัญญาณไปยังทุก ๆ โนดในชั้นถัดไป จนกระทั่งถึงชั้นซ่อนเร้นชั้นสุดท้ายซึ่งจะส่งสัญญาณไปยังทุก ๆ โนดในชั้นเอาต์พุต ดังแสดงในรูปที่ ก.5



รูปที่ ก.5 แสดงโครงสร้างของ FF-MLPs

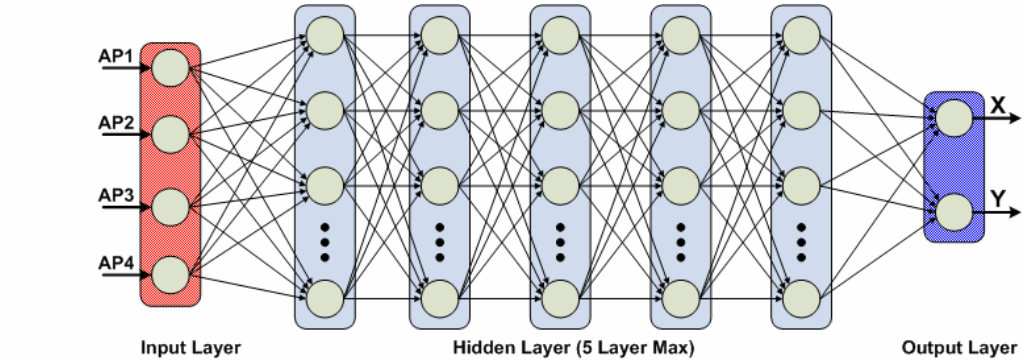
การใช้งาน FF-MLPs จะเริ่มจากการกำหนดโครงสร้างของ FF-MLPs ที่เหมาะสมแล้วทำการฝึกสอนด้วยข้อมูลอ้างอิงก่อนจะถูกนำไปใช้งาน การกำหนดโครงสร้างของ FF-MLPs คือการหาจำนวนชั้นซ่อนเร้นและการหาจำนวนโนดในแต่ละชั้นให้เหมาะสมกับการใช้งาน

4. วิธีการค้นหาโครงสร้างของ FF-MLPs ด้วยจีนเนติกอัลกอริทึม

การหาโครงสร้างของ FF-MLPs ด้วยจีนเนติกอัลกอริทึมในงานวิจัยวิทยานิพนธ์นี้ดำเนินการค้นหาจำนวนชั้นซ่อนเร้น และจำนวนโนดของแต่ละชั้นของ FF-MLPs จากนั้นฝึกสอนด้วยกระบวนการฝึกสอนแบบแพร่กลับ (back-propagation) ผลการทำงานของโปรแกรมจีนเนติกอัลกอริทึมนี้จะได้แบบจำลองนิวรอลที่ผ่านการฝึกสอนแล้วเป็นผลลัพธ์ด้วย

4.1 การกำหนดพารามิเตอร์ของ FF-MLPs ที่ต้องการให้จีนเนติกอัลกอริทึมค้นหา

สำหรับการวิจัยวิทยานิพนธ์นี้กำหนดโครงสร้างเริ่มต้นของ FF-MLPs ที่ต้องการค้นหามีรายละเอียดดังต่อนี้ ให้มีชั้นอินพุต 4 โนดจากจุดเข้าถึง 4 จุด, ชั้นเอาต์พุต 2 โนดคือตำแหน่ง (x, y) และชั้นซ่อนเร้นสูงสุด 5 ชั้น ดังรูปที่ ก.6 กำหนดให้ชั้นซ่อนเร้นแต่ละชั้นมีโนดสูงสุด 7 โนดและต่ำสุด 0 โนด ความหมายของการมีโนดเท่ากับ 0 คือไม่มีชั้นซ่อนเร้นนั้นปรากฏอยู่



รูปที่ ก.6 แสดงโครงสร้างเริ่มต้นของ FF-MLPs ที่ต้องการค้นหา

เพื่อที่จะลดจำนวนชั้นซ่อนเร้นของ FF-MLPs ลงจึงมีการจึงกำหนดค่า “AcceptError” หรือค่าคลาดเคลื่อนมากสุดที่ยอมรับได้ หลักสำคัญในการลดจำนวนชั้นซ่อนเร้น คือ ชั้นซ่อนเร้นจะลดลงก็ต่อเมื่อผลการค้นหาของจีนเนติกอัลกอริทึม ให้จำนวนโนดในบางชั้นเป็น 0 และค่าความคลาดเคลื่อนน้อยว่า AcceptError รอบการค้นหาถัดไปของจีนเนติกอัลกอริทึม จะบังคับให้ชั้นซ่อนเร้นนั้นมีจำนวนโนดเท่ากับ 0 หรือไม่มีชั้นซ่อนเร้นปรากฏอยู่

4.2 การหาโครงสร้างของ FF-MLPs ด้วยจีนเนติกอัลกอริทึม

การค้นหาโครงสร้างนี้ใช้โปรแกรม MATLAB® Toolbox โดยมีขั้นตอนการเขียนโปรแกรม 8 ขั้นตอน ดังที่แสดงในหัวข้อที่ 2.2 และก่อนจะเริ่มเขียนโปรแกรมในขั้นตอนที่ 1 ควรจะทำการกำหนดค่าเริ่มต้นให้กับจีนเนติกอัลกอริทึมก่อน ซึ่งค่าเริ่มต้นของจีนเนติกอัลกอริทึม ได้แก่ จำนวนประชากรเริ่มต้น จำนวนรอบการทำงานของจีนเนติกอัลกอริทึม ร้อยละของการคัดเลือกสายพันธ์ว่าจะเลือกจำนวนเท่าใดจากประชากรเริ่มต้นที่กำหนด จำนวนพารามิเตอร์ของระบบที่ต้องการค้นหา และความละเอียดของค่าพารามิเตอร์เมื่อแทนเป็นระบบเลขฐานสอง ซึ่งในโปรแกรม MATLAB® ได้ใช้ตัวแปรเพื่อบ่งบอกปริมาณค่าเริ่มต้นของจีนเนติกอัลกอริทึมดังกล่าว ดังต่อไปนี้

nPopulation	คือ จำนวนประชากรเริ่มต้น กำหนดให้เท่ากับ 25
Max_Gen	คือ จำนวนรอบการทำงานของจีนเนติกอัลกอริทึม กำหนดให้เท่ากับ 25
GGAP	คือ ร้อยละของการคัดเลือกสายพันธ์ กำหนดให้เท่ากับ 0.6
n_Var	คือ จำนวนพารามิเตอร์ของระบบที่ต้องการค้นหา กำหนดให้เท่ากับ 5
n_Bit	คือ ความละเอียดของค่าพารามิเตอร์เมื่อแทนเป็นระบบเลขฐานสอง

ขั้นตอนที่ 1 สร้างประชากรใหม่ให้กับจีนเนติกอัลกอริทึม โดยช้ำสั่งของโปรแกรม MATLAB® ดังนี้

```
Chrom = crtbp(nPopulation , n_Var * n_Bit);
```

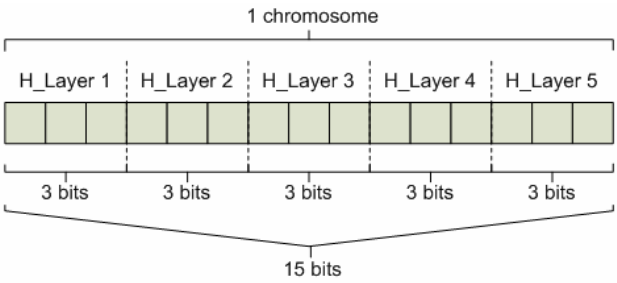
Chrom คือ กลุ่มของโครโมโซมเริ่มต้น ส่วยย่อยของโครโมโซม (ยีน) ที่ได้จากคำสั่งนี้จะมีลักษณะเป็นบิตเพราะฉนั้นโครโมโซมที่ได้จะมีลักษณะเป็นไบนารี (binary) ซึ่งจำนวนโครโมโซมจะมีเท่ากับ nPopulation ที่กำหนด

จากโครงสร้างเริ่มต้นของ FF-MLPs ที่ต้องการค้นหาในรูปที่ ก.6 จำนวนชั้นซ่อนเร้นสูงสุดเท่ากับ 5 ดังนั้นเราจึงกำหนด n_Var เท่ากับ 5 จากนั้นกำหนดให้ค่าของตัวแปรพารามิเตอร์คือจำนวนโนดที่มีในแต่ละชั้นซ่อนเร้น จะได้ว่า Max_Var เท่ากับ 10, Min_Var เท่ากับ 0 และเห็นได้ว่าค่าความละเอียดของตัวแปร (resolution step) จะมีค่าเป็น 1 เนื่องจากจำนวนโนดจะต้องเป็นจำนวนเต็ม ดังนั้นเมื่อแทนพารามิเตอร์นี้ด้วยระบบเลขฐานสองสามารถคำนวณจำนวนบิตได้ดังนี้

$$2^{nBit} \geq \frac{\left(\text{Max\_Var - Min\_Var}\right)}{\text{Resolution Step}} + 1 \qquad \text{(ก.1)}$$

$$2^{nBit} \geq \frac{\left(7 - 0\right)}{1} + 1 \qquad \text{(ก.2)}$$

$$nBit \geq \frac{\log(8)}{\log(2)} = 3 \qquad \text{(ก.3)}$$



รูปที่ ก.7 รายละเอียดค่าความยาวของโครโมโซม 1 ตัว

นั่นคือกำหนดความค่าของพารามิเตอร์นี้แทนเป็นระบบเลขฐานสองเท่ากับ 3 บิต และเมื่อประกอบเข้ากันเป็นโครโมโซม 1 ตัวจะแทนด้วยระบบเลขฐานสองเท่ากับ 15 บิต ดังรูปที่ ก.7

ขั้นตอนที่ 2 ประเมินค่าโครโมโซมของกลุ่มประชากรทั้งหมด ด้วยฟังก์ชันวัตถุประสงค์และเนื่องจากระบบไม่สามารถเข้าใจค่าของโครโมโซม ดังนั้นโครโมโซมจะต้องถูกถอดรหัสให้เป็นรูปแบบฟีโนไทป์ ก่อนที่จะถูกการประเมินด้วยฟังก์ชันวัตถุประสงค์ ซึ่งในงานวิจัยวิทยานิพนธ์นี้รูปแบบฟีโนไทป์ คือ เลขฐานสิบ ซึ่งถอดรหัสดังกล่าวจะใช้คำสั่งของโปรแกรม MATLAB® ดังนี้

```
Phen = bs2rv(Chrom,FieldD)
```

Phen คือ รูปแบบฟีโนไทป์ที่จะถูกประเมินด้วยฟังก์ชันวัตถุประสงค์

FieldD คือ รูปแบบของการกำหนดค่าในการถอดรหัสโครโมโซมที่เป็นไบนารีเป็นรูปแบบฟีโนไทป์ที่เป็นเลขฐานสิบ

การกำหนดรูปแบบของ FieldD จะมีโครงสร้างการกำหนดรูปแบบซึ่งอธิบายได้ดังนี้

$$\text{FieldD} = \begin{bmatrix} \text{length} \\ \text{lower limit} \\ \text{upper limit} \\ \text{code} \\ \text{scale} \\ \text{lower bound} \\ \text{upper bound} \end{bmatrix} \qquad \text{(ก.4)}$$

ความสัมพันธ์ ก.4 แสดงการกำหนดโครงสร้างของ FieldD แบบเมตริกซ์ แต่ละอิลิเมนต์มีความหมายและมีกรอบการใช้ดังต่อไปนี้

- length เป็นโครงสร้างสำหรับกำหนดความยาวในแต่ละโครโมโซม ซึ่งกำหนดจากจำนวนพารามิเตอร์ที่ต้องการค้นหาและความละเอียดของพารามิเตอร์แต่ละตัว การกำหนดโครงสร้างของ length ในงานวิจัยวิทยานิพนธ์เป็นดังนี้

```
length = [3   3   3   3   3];
```
(ก.5)

จากการกำหนดโครงสร้างของ length ข้างต้น หมายถึงมีค่าพารามิเตอร์ที่ต้องการค้นหาทั้ง 5 ตัว ซึ่งแต่ละตัวมีค่าความละเอียดเท่ากับ 3 บิต ค่าที่ปรากฏในแต่ละคอลัมน์คือจำนวนโนดของ FF-MLPs ในแต่ละชั้น

- lower limit คือ ขอบเขตของค่าต่ำสุดที่เป็นไปได้ในแต่ละพารามิเตอร์ที่ต้องการค้นหา
- upper limit คือ ขอบเขตของค่าสูงสุดที่เป็นไปได้ในแต่ละพารามิเตอร์ที่ต้องการค้นหา

ซึ่งในงานวิจัยนี้ได้กำหนดโครงสร้างของ lower limit และ upper limit ดังนี้

lower limit = [0 0 0 0 0]; (ก.6)

upper limit = [7 7 7 7 7]; (ก.7)

จากการกำหนดโครงสร้างของ lower limit ข้างต้น หมายถึงขอบเขตของค่าต่ำสุดที่เป็นไปได้ของจำนวนโนดในและชั้นซ่อนเร้น ค่า 0 หมายถึงไม่มีชั้นซ่อนเร้นนี้ปรากฏอยู่ และโครงสร้างของ upper limit ข้างต้น หมายถึงขอบเขตของค่าสูงสุดที่เป็นไปได้ของจำนวนโนดในและชั้นซ่อนเร้นจำนวน 7 โนด

- code เป็นโครงสร้างสำหรับการเลือกรหัสที่ใช้ โดยในโปรแกรม MATLAB® ให้เลือกการให้รหัสอยู่ 2 แบบด้วยกันคือ ถ้า code มีค่าเท่ากับศูนย์ หมายถึง การเข้ารหัสแบบบไบนารีมาตรฐาน และค่า code มีค่าเท่ากับหนึ่ง หมายถึง การเข้ารหัสเกรย์ ซึ่งในงานวิจัยนี้ได้เลือกการเข้ารหัสแบบไบนารีมาตรฐาน

- scale เป็นโครงสร้างสำหรับเลือกเทคนิคการสเกลค่า ในช่วงระหว่างขอบเขตของค่าต่ำสุดที่เป็นไปได้และค่าสูงสุดที่เป็นไปได้ซึ่งในโปรแกรม MATLAB® ให้เลือกเทคนิคการสเกลอยู่ 2 แบบ คือ การสเกลเชิงเลขคณิต (arithmetic scaling) โดยการกำหนดค่า scale เป็นศูนย์และการสเกลเชิงลอการิทึม (logarithmic scaling) จะกำหนดค่า scale เป็นหนึ่งซึ่งงานวิจัยวิทยานิพนธ์นี้ได้เลือกเทคนิคการสเกลเชิงเลขคณิต

- lower bound เป็นการกำหนดรูปแบบว่าจะนำค่าขอบเขตต่ำสุด (lower limit) ไปร่วมพิจารณาด้วยหรือไม่ เช่น ในงานวิจัยวิทยานิพนธ์นี้ค่าขอบเขตต่ำสุดของสุดของพารามิเตอร์ทั้งห้าค่าคือ 0 ซึ่งถ้าต้องการพิจารณาคำตอบที่ค่าดังกล่าวด้วยจะกำหนดค่านี้เป็นหนึ่งแต่ถ้าไม่ต้องการพิจารณาคำตอบที่ขอบเขตต่ำสุดจะกำหนดค่านี้เป็นศูนย์ซึ่งในงานวิจัยวิทยานิพนธ์นี้ได้เลือกที่จะพิจารณาคำตอบที่ค่าขอบเขตต่ำสุดด้วยจึงกำหนดค่า lower bound นี้ให้มีค่าเป็นหนึ่ง

- upper bound ความหมายเช่นเดียวกับ lower bound แต่สำหรับ upper bound จะพิจารณาที่ค่าขอบเขตสูงสุดที่เป็นไปได้(upper limit)แทน

จากการอธิบายโครงสร้างของ FieldD ที่กล่างทั้งหมดข้างต้นงานวิจัยวิทยานิพนธ์นี้ได้กำหนดโครงสร้างของ FieldD ตามความเหมาะสมของงานดังต่อไปนี้

$$\text{FieldD} = \begin{bmatrix} \text{length} \\ \text{lower limit} \\ \text{upper limit} \\ \text{code} \\ \text{scale} \\ \text{lower bound} \\ \text{upper bound} \end{bmatrix} = \begin{bmatrix} 3 & 3 & 3 & 3 & 3 \\ 0 & 0 & 0 & 0 & 0 \\ 7 & 7 & 7 & 7 & 7 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad \text{(ก.8)}$$

ขั้นตอนที่ 3 ประเมินผลคำตอบของระบบด้วยฟังก์ชันวัตถุประสงค์และคำนวณหาค่าความเหมาะสมส่งกลับไปเพื่อใช้ในการคัดเลือกโครโมโซมที่ดีสำหรับการสืบสายพันธุ์ สำหรับงานวิจัยวิทยานิพนธ์นี้กำหนกค่า OffError หรือจำนวนจุดทดสอบที่ให้ค่าคลาดเคลื่อนมากกว่ากว่าระยะที่กำหนด ฟังก์ชันวัตถุประสงค์จะทำการสร้าง ฝึกสอน และทดสอบ FF-MLPs เพื่อหาค่า OffError สำหรับการทดสอบกำหนดค่า OffError0.2 หรือให้มีจำนวนจุดทดสอบที่ให้ค่าคลาดเคลื่อนจากจุดอ้างอิงเกิน 0.2 เมตร ให้ชื่อฟังก์ชันของโปรแกรม MATLAB® คือ GA_Objfunction ซึ่งจะนำเสนอในรูปของชุดคำสั้งเทียบ (pseudo code) ดังรูปที่ ก.8

จากโปรแกรมฟังก์ชันวัตถุประสงค์ข้างต้น การคำนวณค่า ObjVSel หรือค่าความคาดเคลื่อนที่เกิดจากการเปรียบเทียบผลที่ได้จากการประมวลผลโปรแกรมของฟังก์ชันวัตถุประสงค์จากข้อมูลจริงที่ได้จากการทดสอบ เพื่อต้องการกำจัดประชากรที่ให้โครงสร้างของ FF-MLPs ที่มีจำนวนชั้นซ่อนเร้นมากให้น้อยลง จึงมีการกำหนดค่า ObjVSel ขึ้นมา 2 ชุด ชุดที่หนึ่งสำหรับกรณีที่ค่า OffError มากกว่า AcceptError จะคำนวณโดยให้ความสำคัญกับค่า OffError และชุดที่สองสำหรับกรณีที่มีค่า OffError น้อยกว่า AcceptError จะคำนวณได้โดยให้ความสำคัญกับจำนวนโนดและจำนวนชั้นซ่อนเร้นของ FF-MLPs ที่กำลังทดสอบ

การคำนวณค่า ObjVSel ส่วนแรกสำหรับโครโมโซมที่ให้ค่า OffError มากกว่า AcceptError ความหมายของโครโมโซมตัวนี้คือ เมื่อคำนวณค่า OffError แล้วยังไม่อยู่ในขอบเขตค่าความคลาดเคลื่อนที่ตั้งไว้ ดังนั้นจึงกำหนดให้ค่า ObjVSel ขึ้นกับค่า OffError ที่คำนวณได้ ดังนี้

$$ObjVSel = OffError \times k_1 \quad (ก.9)$$

$$k_1 = sum(mChrom) \times 10^{sum(round(mChrom./(mChrom+0.001)))} \quad (ก.10)$$

$$k_1 = sum([4\ 7\ 7\ 7\ 7\ 7\ 2]) \times 10^{sum([1\ 1\ 1\ 1\ 1\ 1\ 1])} = 41 \times 10^7 \quad (ก.11)$$

mChrom คือ โครโมโซมใหญ่สุดที่เป็นไปได้ ซึ่งเท่ากับ [4 7 7 7 7 7 2]

ค่า k1 เป็นค่าที่ได้จากผลรวมของโครโมโซมใหญ่สุดที่เป็นไปได้คือ 41 ในส่วนของกำลังของเลขสิบ คือการคำนวณจำนวนของชั้นซ่อนเร้นที่ไม่เป็นศูนย์

ส่วนที่สองสำหรับโครโมโซมที่ให้ค่า OffError น้อยกว่า AcceptError นั่นคือโครโมโซมตัวนี้เหมาะเป็นต้นกำเนิดของประชากรรุ่นถัดไปจึงให้ความสำคัญกับการลดจำนวนชั้นซ่อนเร้นลงการคำนวณจะให้ความสำคัญกับจำนวนโนดและจำนวนชั้นซ่อนเร้นของ FF-MLPs ค่า ObjVSel สามารถคำนวณได้ ดังนี้

$$ObjVSel = OffError \times k_2 \quad (ก.12)$$

$$k_2 = sum(tChrom) \times 10^{sum(round(tChrom./(tChrom+0.001)))} \quad (ก.13)$$

ยกตัวอย่าง

- โครโมโซมตัวที่ 1 มีโครงสร้าง [4 2 1 1 1 0 2] จะให้ค่า ObjVSel = 11 x 10^6
- โครโมโซมตัวที่ 2 มีโครงสร้าง [4 2 1 2 0 0 2] จะให้ค่า ObjVSel = 11 x 10^5

จากค่า ObjVSel ทั้งสองเมื่อทำการคัดเลือกสายพันธุ์โดยวิธีการจัดอันดับโครโมโซมตัวที่ 2 จะมีโอกาสในการให้ประชากรรุ่นลูกจะสูงกว่า

จากการประเมินด้วยฟังก์ชันวัตถุประสงค์จะให้ค่า best_Chromosome หรือโครงสร้าง FF-MLPs ที่ดีที่สุดจากกลุ่มประชากรที่มี best_net หรือ FF-MLPs ที่ทำการฝึกสอนเรียบร้อยแล้ว และ ObjVSel ค่านี้จะไปคำนวณค่าความเหมาะสมซึ่งการคำนวณค่าความเหมาะสมในงานวิจัยนี้ได้ใช้คำสั่งโปรแกรม MATLAB® โดยใช้วิธีการจัดอันดับ (ranking selection) สำหรับกำหนดค่าความเหมาะสม โดยใช้โปรแกรม MATLAB® ที่ใช้เป็นดังนี้

FintV = ranking(ObjVSel)

เมื่อ FintV คือ ค่าความเหมาะสมที่ได้จากวิธีการจัดอันดับ

ObjVSel คือ เป็นค่าความคลาดเคลื่อนที่ได้จากการคำนวณในฟังก์ชันวัตถุประสงค์

```
1  Function[ ObjVSel,best_chrom,best_net]= GA_Objfunction(ObjV,AcceptError);
2  For  idex=1 To nPopulation
3  test_error = 99999999999999999.9;
4       For t_index=1 To test_Loop
5            Create(FF-MLPs);                  % สร้าง FF-MLPs
6            Tarin(FF-MLPs);                   % ฝึกสอน FF-MLPs
7            Calculate_OffError(FF-MLPs);      % คำนวณค่า OffError โดย FF-MLPs
8            tChrom = structure of FF-MLPs     % กำหนดโครงสร้างของFF-MLPs คือ tChrom
9            k1 = sum(tChrom)*10^(sum(round(tChrom./(tChrom+0.001))));
10           k2 = sum([4 7 7 7 7 7 2])*10^7);
11           If OffError < test_error
12                test_error = OffError;
13                test_net   = FF-MLPs net;
14                If OffError < AcceptError
15                ObjVSel(idex,1)= AcceptError * k1;
16                Else
17                ObjVSel(idex,1)= OffError * k2;
18                EndIf
19           EndIf
20      EndFor {t_idex}
21      Select_and_Save(best_chrom,best_net);
22 EndFor {idex}
```

รูปที่ ก.8 ชุดคำสั้งเทียบของฟังก์ชันวัตถุประสงค์

จากที่กล่าวมาข้างต้น นอกจากการหาค่าความเหมาะสมด้วยวิธีการจัดอันดับแล้วยังมีอีกหลายวิธีที่ใช้กันอย่างแพร่หลาย เช่น วิธีการแบ่งสัดส่วน (proportionate) วิธีการโบลต์ซมันน์ (Boltzmann) และวิธีการแข่งขัน (tournament) เป็นต้น

ขั้นตอนที่ 4 ใช้ค่าความเหมาะสมที่ได้จากขั้นตอนที่ 3 เพื่อคัดเลือกโครโมโซมบางกลุ่มมาเป็นต้นกำเนิดสายพันธุ์ ซึ่งการคัดเลือกดังกล่าวในโปรแกรม MATLAB® มีให้เลือก 2 วิธี คือ วิธีการชักตัวอย่างของวงล้อรูเล็ทและวิธีการชักตัวอย่างของกระบวนการเฟ้นสุ่มรอบจักรวาลโดยในงานวิจัยวิทยานิพนธ์นี้ได้เลือกใช้วิธีการชักตัวอย่างของกระบวนการเฟ้นสุ่มครอบจักรวาล คำสั่งในโปรแกรม สำหรับขั้นตอนนี้คือ

```
SelCh = select('sus',Chrom,FitnV,GGAP);
```

SelCh คือ ต้นกำเนิดสายพันธุ์ที่ได้จากการคัดเลือกเพื่อเตรียมที่จะสร้างลูกหลานด้วยปฏิบัติการทางสายพันธุ์ในขั้นตอนต่อไป

sus คือ เป็นการกำหนดการใช้วิธีการชักตัวอย่างของกระบวนการเฟ้นสุ่มครอบจักรวาลแต่เลือกใช้วิธีการชักตัวอย่างของวงล้อรูเล็ทจำได้ชื่อว่า rws แทน

Chrom คือ ประชากรเริ่มต้นที่ได้จากขั้นตอนที่ 1

FitnV คือ ค่าความเหมาะสมที่ได้จากขั้นตอนที่ 3

GGAP คือ ร้อยละของการคัดเลือกสายพันธุ์จากประชากรเริ่มต้น สำหรับงานวิจัยนี้ได้กำหนด มีค่าเท่ากับ 0.6

ขั้นตอนที่ 5 นำต้นกำเนิดสายพันธุ์มาสร้างลูกหลาน ด้วยปฏิบัติการทางสายพันธุ์ ซึ่งแบ่งเป็นสองขั้นตอนย่อย คือ การทำครอสโอเวอร์กับการทำมิวเทชัน โดยมีคำสั่ง MATLAB® สำหรับการทำครอสโอเวอร์และการทำมิวเทชัน เป็นดังนี้

```
SelCh = recombin('xovsp',SelCh,P_Crossover);
SelCh = mut(SelCh);
```

SelCh คือ โครโมโซมหลังจากการทำครอสโอเวอร์ข้างต้นกำเนิดสายพันธุ์

xovsp คือ การทำครอสโอเวอร์แบบจุดเดียว(single point crossover)

P_Crossover คือ ความน่าจะเป็นในการทำครอสโอเวอร์

สำหรับงานวิจัยวิทยานิพนธ์นี้กำหนดค่าความน่าจะเป็นในการทำครอสโอเวอร์เท่ากับ 0.9 และ 0.7 โดยค่าดังกล่าวได้ทำการทดสอบและนำเสนอผลการทดสอบไว้ในหัวข้อที่ 5

ขั้นตอนที่ 6 คำนวณค่าความเหมาะสมของโครโมโซมลูกหลาน ซึ่งใช้วิธีการเดียวกับขั้นตอนที่ 3 ประเมินผลคำตอบของระบบด้วยฟังก์ชันวัตถุประสงค์ แต่ก่อนจะประเมินคำตอบด้วยด้วยฟังก์ชันวัตถุประสงค์ต้องมีการกำจัดชั้นซ่อนเร้นที่ให้ค่าโนดเป็น 0 ก่อน

จากโครงสร้างของโปรแกรมรูปที่ ก.9 การค้นหาจำนวนโนดในแต่ละชั้นซ่อนเร้นของ FF-MLPs เมื่อแทนตัวแปรที่ต้องการให้จีนเนติกอัลกอริทึมค้นหาคือจำนวนโนดในแต่ละชั้นซ่อนเร้น หากต้องการกำจัดชั้นซ่อนเร้นบางชั้นออกจะทำให้จำนวนบิตของฟีโนไทป์ในจีนเนติกอัลกอริทึมจะมีการลดจำนวนบิตลง การเขียนโปรแกรมให้ง่ายจึงเลือกใช้วิธีบังคับให้ฟีโนไทป์ที่แทนชั้นซ่อนเร้นที่หายไปหรือให้เป็นศูนย์โดย กำหนด Masking_Array เพื่อกำจัดชั้นซ่อนเร้นให้หายไป ยกตัวอย่างค่า Masking_Array ดังความสัมพันธ์ ก.14 และความสัมพันธ์ ก.15

Masking_Array = [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]; (ก.14)

Masking_Array = [1 1 1 0 0 0 1 1 1 1 1 1 1 1 1]; (ก.15)

จากความสัมพันธ์ ก.14 เป็นค่าเริ่มต้นเมื่อโปรแกรมจีนเนติกเริ่มทำงานจะกำหนดให้ทุกบิตมีค่าเป็น 1 มีจำนวนบิตเท่ากับความยาวของโครโมโซมหนึ่งตัว หากผลการประเมินด้วยฟังก์ชันวัตถุประสงค์แล้วให้ค่า OffError น้อยกว่า AcceptError และจำนวนโนดในชั้นซ่อนเร้นที่สองเป็น 0 จะทำการกำหนด Masking_Array ใหม่ดังความสัมพันธ์ ก.15

จากแนวคิดนี้เมื่อโปรแกรมทำงานได้ประชากรกลุ่มแรก ผ่านกระบวนการต่าง ๆ ของจีนเนติกอัลกอริทึมแล้วจากนั้นจะทำการกำหนด Masking_Array แล้วคูณเข้ากับฟีโนไทป์ของประชากรทุกตัวก่อนที่จะประเมินด้วยฟังก์ชันวัตถุประสงค์ ดังรูปที่ ก.9 ขั้นตอนการทำงานของโปรแกรม MATLAB® ในการค้นหาโครงสร้างของ FF-MLPs

ขั้นตอนที่ 7 โครโมโซมในประชากรเดิมจะถูกแทนที่ด้วยลูกหลานที่ได้จากขั้นตอนที่ 5 ซึ่งประชากรเพียงบางส่วนเท่านั้นที่จะถูกแทนที่ด้วยกลวิธีเฉพาะสำหรับขั้นตอนของการแทนที่โดยใช่ค่าความเหมาะสมในการตัดสิน ซึ่งคำสั่งโปรแกรม MATLAB® ที่ใช้สำหรับขั้นตอนนี้ คือ

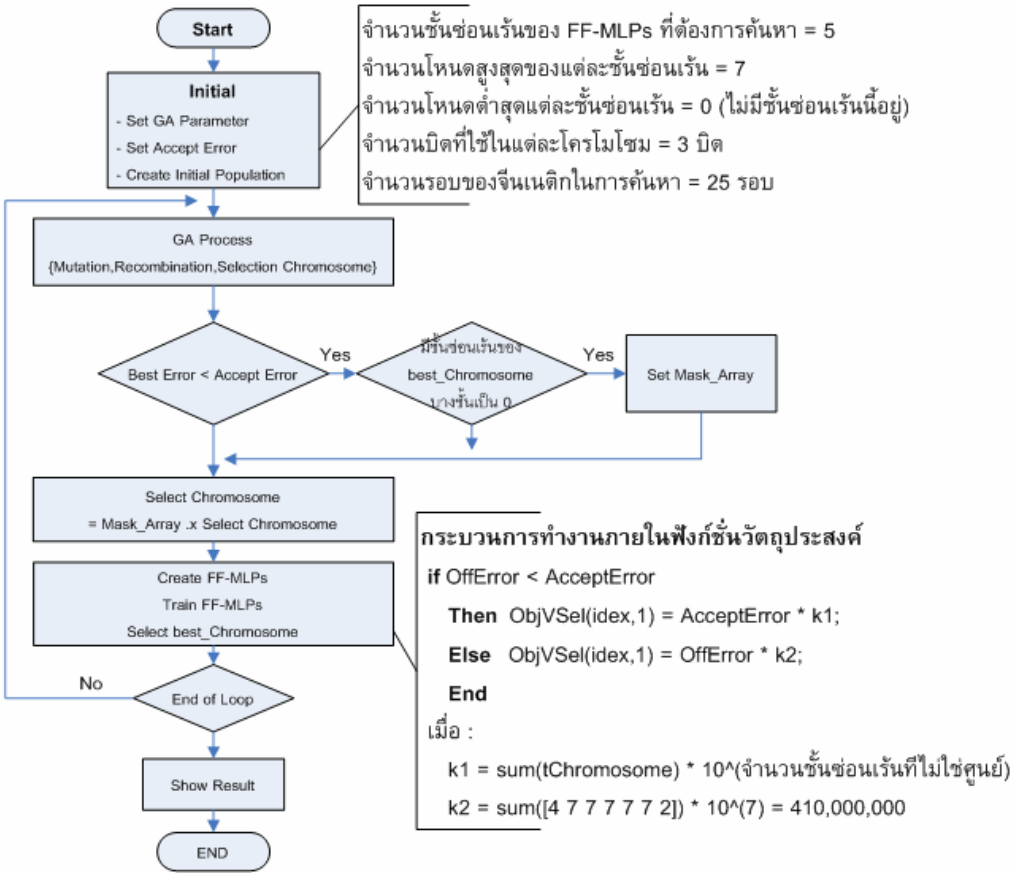
```
[Chrom ObjV] = reins(Chrom,SelCh,1,1,ObjV,ObjVSel);
```

ObjV คือ ค่าการประเมิน (objective value) ซึ่งเป็นผลที่ได้จากการคำนวณในฟังก์ชันวัตถุประสงค์ในขั้นตอนที่ 3

ObjVSel คือ ค่าการประเมิน (objective value) ซึ่งเป็นผลที่ได้จากการคำนวณในฟังก์ชันวัตถุประสงค์ในขั้นตอนที่ 6

ขั้นตอนที่ 8 เริ่มต้นทำซ้ำจากขั้นตอนที่ 2 ไปเรื่อย ๆ จนกระทั่งได้คำตอบที่ต้องการ

โปรแกรม MATLAB® ที่ใช้หาโครงสร้างของ FF-MLPs ด้วยจีนเนติกอัลกอริทึมในงานวิจัยวิทยานิพนธ์นี้ ที่ครอบคลุมการอธิบายดังกล่าวทั้งหมดข้างต้นอาจดูได้จาก ภาคผนวก ค. โปรแกรม MATLAB® สำหรับให้จีนเนติกอัลกอริทึมทำงานเพื่อหาโครงสร้างของ FF-MLPs



รูปที่ ก.9 ขั้นตอนการทำงานของโปรแกรมในการค้นหาโครงสร้างของ FF-MLPs

5. ผลการค้นหาโครงสร้างของ FF-MLPs และอภิปราย

การทดสอบแบ่งออกเป็น 2 ส่วนคือ ส่วนที่ 1 เป็นการหาโครงสร้างของ FF-MLPs ที่เหมาะสมสำหรับการจำลองแผนที่ของห้องที่ใช้ทดสอบการระบุตำแหน่งตัวเอง และส่วนที่ 2 คือการนำ FF-MLPs ที่ได้ไปใช้แทนโครงสร้างเดิมเพื่อเปรียบเทียบประสิทธิภาพในการระบุตำแหน่งตัวเอง ดังรายละเอียดต่อไปนี้

ส่วนที่ 1 การหาโครงสร้างของ FF-MLPs ที่เหมาะสม

การหาโครงสร้าง FF-MLPs ที่เหมาะสมนี้โดยให้คอมพิวเตอร์ส่วนบุคคลทำการทดสอบจำนวน 120 ตัวอย่าง กำหนดให้ปรับเปลี่ยนพารามิเตอร์สองตัว พารามิเตอร์ตัวแรกคือเปอร์เซ็นต์การทำครอสโอเวอร์ที่ 0.7 และ 0.9 พารามิเตอร์ตัวที่สองคือค่า AcceptError ที่ 8, 10 และ 12 แต่ละชุดให้ทำการทดสอบจำนวน 20 ตัวอย่าง ส่วนพารามิเตอร์ที่กำหนดให้เหมือนกัน คือ Max_Gen เท่ากับ 25, nPopulation เท่ากับ 25, n_Bit เท่ากับ 3, GGAP เท่ากับ 0.6 และ FF-MLPs Train_Loop เท่ากับ 4

ผลการทำงานได้โครงสร้างที่มีชั้นซ่อนเร้นหนึ่งชั้นจำนวนโนด 4โนด แบบจำลองนิวรอลที่ได้เมื่อทดสอบแล้วให้ค่า OffError 0.2 เมตรที่ 7.6923% จากจำนวนจุดทดสอบ 546 จุด ใช้เวลาทำงาน 8 ชั่วโมง 30 นาที นำผลการค้นหาทั้ง 120 ตัวอย่างมาเรียงลำดับตาม FF_Ranking และ %OffError ดังตารางที่ ก.1 นำผลการเรียงลำดับมาแสดงเฉพาะ 10 ลำดับแรกทำให้สามารถเลือกโครงข่ายประสาทเทียมของตัวอย่างทดสอบหมายเลข 2-52 ซึ่งโครงสร้างที่มีชั้นซ่อนเร้น 1 ชั้น จำนวนโนด 4 โนด แบบจำลองที่ได้เมื่อทดสอบแล้วให้ค่า OffError 0.2 เมตรที่ 5.6777 เปอร์เซ็นต์

ตารางที่ ก.1 ผลการทดสอบเมื่อเรียงลำดับตาม FF-Ranking และ %OffError เฉพาะ 10 ลำดับแรก

Item	PC	AcceptError	%Crossover	%OffError	FF_ANN	FF_Ranking
1	2 52	12	0.7	5.6777	4 0 4 0 0 0 2	100
2	2 27	8	0.7	7.6923	4 0 0 0 4 0 2	100
3	3 49	12	0.9	8.6081	4 0 4 0 0 0 2	100
4	1 44	12	0.9	11.3553	4 4 0 0 0 0 2	100
5	1 45	12	0.9	11.3553	4 4 0 0 0 0 2	100
6	1 46	12	0.9	11.3553	4 4 0 0 0 0 2	100
7	2 24	10	0.7	0.1832	4 5 0 0 0 0 2	110
8	3 27	8	0.9	0.1832	4 0 0 0 0 5 2	110
9	2 22	10	0.7	0.3663	4 0 0 0 0 5 2	110
10	2 46	12	0.7	0.3663	4 5 0 0 0 0 2	110

OffError	จำนวนจุดทดสอบที่ให้ระยะคลาดเคลื่อนจากตำแหน่งจริงมากกว่า 0.2 เมตร
PC	ชื่อของตัวอย่างที่ทดสอบ
AcceptError	เปอร์เซ็นต์ของจำนวนจุด OffError มากสุดที่ยอมรับได้
P_Crossover	ค่าการทำครอสโอเวอร์ของจีนเนติกอัลกอริทึม
%OffError	เปอร์เซ็นต์ของจำนวนจุด OffError จากจุดที่ทดสอบทั้งหมด 546 จุด
FF-Ann	โครงสร้างของ FF-MLPs ที่ได้จากการทำงานของจีนเนติกอัลกอริทึม

FF_Ranking การคำนวนลำดับของโครงสร้าง FF-MLPs โดย คำนวนจาก

FF_RankScale = 10 ^ จำนวนชั้นซ่อนเร้นที่ไม่เป็นศูนย์

FF_Ranking = ผลรวมของโนดในแต่ละเลเยอร์ * FF_RankScale

หากพิจารณาเฉพาะผลของการทำงานที่ให้จำนวนชั้นซ่อนเร้นเท่ากับ 1 ตามตารางที่ ก.2 เพื่อยืนยันการทำงานของโปรแกรมจีนเนติกอัลกอริทึมว่าสามารถลดโครงสร้างของ FF-MLPs ได้ ตัวเลขในตารางจะแสดงจำนวนของ FF-MLPs ที่มีชั้นซ่อนเร้นหนึ่งชั้น/จำนวนตัวอย่างที่ทำการทดสอบ ซึ่งบอกได้ว่าโปรแกรมจีนเนติกที่พัฒนาขึ้นมีแนวโน้มในการลดโครงสร้าง FF-MLPs ลง

ตารางที่ ก.2 ผลการค้นหาโครงสร้างของ FF-MLPs ที่ให้จำนวนชั้นซ่อนเร้นเท่ากับ 1

	ค่าการทำครอสโอเวอร์ = 0.9	ค่าการทำครอสโอเวอร์ = 0.7
AcceptError = 12	20/20	17/20
AcceptError = 10	17/20	16/20
AcceptError = 8	16/20	18/20

ส่วนที่ 2 การทดสอบโครงสร้างของ FF-MLPs ที่ได้จากจีนเนติกอัลกอริทึม

การทดสอบโดยการแทนที่แบบจำลองนิวรอลที่ได้จากการทำงานของโปรแกรมจีนเนติกอัลกอริทึมเข้าไปในระบบระบุตำแหน่งเพื่อดูประสิทธิภาพของระบบโดยนับระยะเบี่ยงเบนจากจุดทดสอบผลการทดสอบ

จากรูปที่ ก.10 เป็นผลการทดสอบด้วย FF-MLPs โครงสร้างเดิมจากการทดสอบโดยการลองผิดลองถูก และรูปที่ ก.11 แสดงผลการทดสอบด้วย FF-MLPs โครงสร้างใหม่ที่ได้จากการทำงานของจีนเนติกอัลกอริทึมซึ่งให้ผลการทำงานที่ดีขึ้นจากเดิม

ตารางที่ ก.3 ผลการทดสอบเพื่อเปรียบเทียบประสิทธิภาพของ FF-MLPs

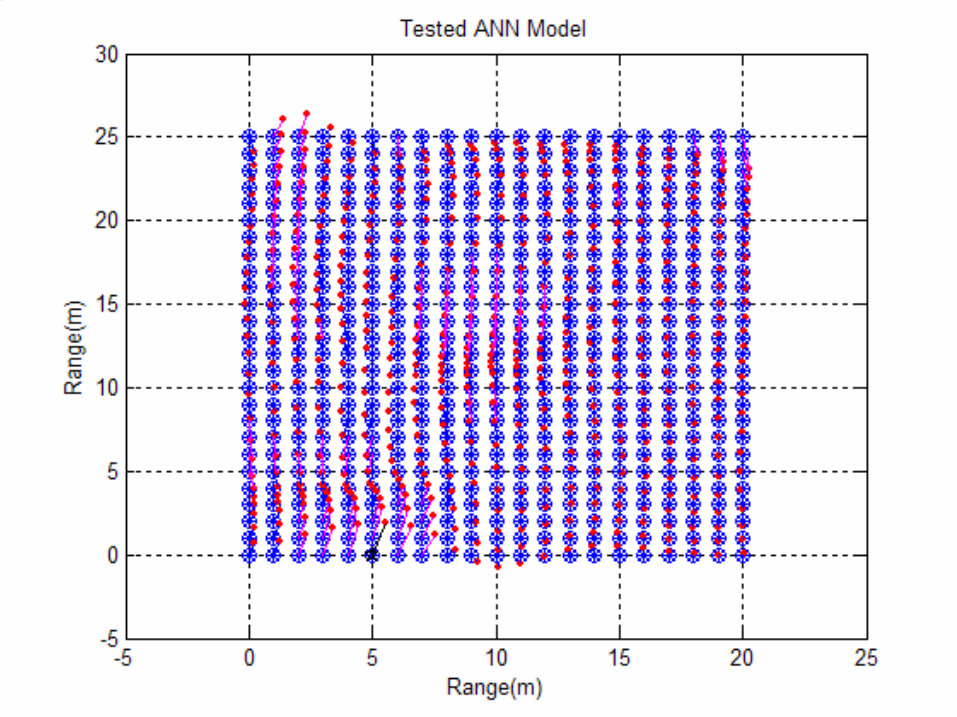
โครงสร้าง FF-MLPs ที่ใช้ทดสอบ	โครงสร้างเดิม จากการลองผิดลองถูก	โครงสร้างใหม่ จากโปรแกรมจีนเนติกอัลกอริทึม
จำนวนจุดที่ระยะเบี่ยงเบนน้อยกว่า 1.0 เมตร	449/546 (82.23%)	499/546 (91.39%)
จำนวนจุดที่ระยะเบี่ยงเบนน้อยกว่า 0.5 เมตร	285/546 (52.20%)	353/546 (64.65%)
จำนวนจุดที่ระยะเบี่ยงเบนน้อยกว่า 0.2 เมตร	101/546 (18.50%)	157/546 (28.75%)
ระยะเบี่ยงเบนจากจุดทดสอบเฉลี่ย(เมตร)	0.5808	0.4566
ระยะเบี่ยงเบนจากจุดทดสอบสูงสุด(เมตร)	1.9445	1.9440
ตำแหน่งที่เกิดการเบี่ยงเบนสูงสุด	(5, 0)	(6, 0)

ผลการทดสอบที่แสดงในตารางที่ ก.3 จะเห็นว่าโครงสร้าง FF-MLPs ที่ได้จากโปรแกรมจีนเนติกอัลกอริทึมให้ผลการทดสอบดีกว่า คือ จำนวนจุดที่ระยะเบี่ยงเบนจากจุดทดสอบน้อยกว่า 1.0 เมตรเท่ากับ 499 จุดจากจุดทดสอบ 546 จุดหรือ 91.39 เปอร์เซ็นต์ มีระยะเบี่ยงเบนจากจุดทดสอบเฉลี่ย 0.4566 เมตรและมีระยะเบี่ยงเบนจากจุดทดสอบสูงสุด 1.9440 เมตร

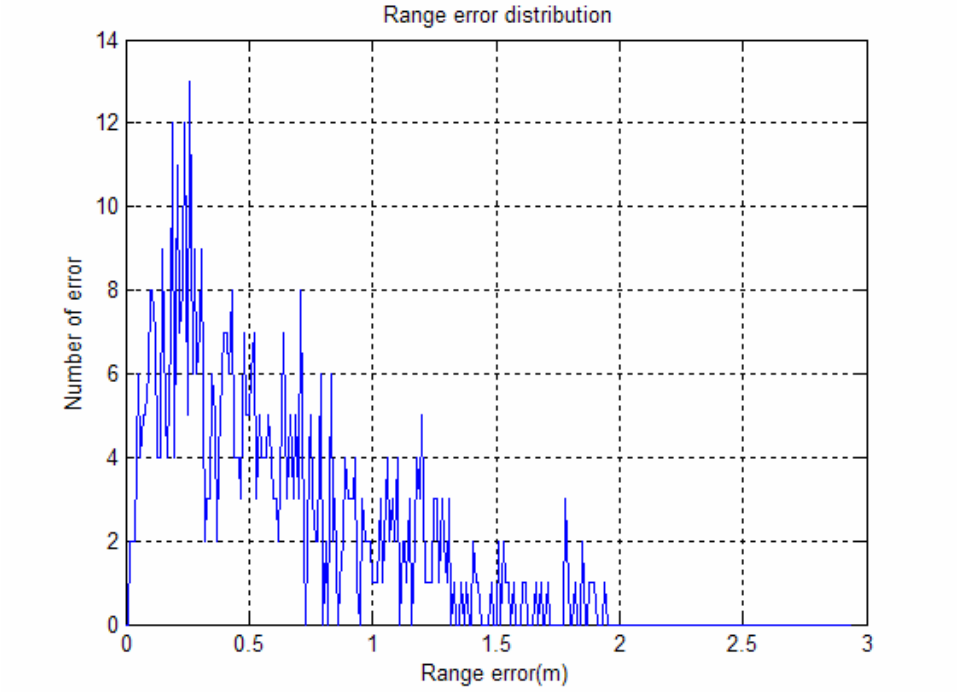
6. สรุป

จากที่กล่าวมาทั้งหมดข้างต้น ได้นำเสนอถึงวิธีการและหลักการของจีนเนติกอัลกอริทึมสำหรับการค้นหาโครงสร้าง FF-MLPs ซึ่งผลของการหาได้โครงสร้างที่เล็กสุดสำหรับชุดข้อมูลนี้แบบ [4, 4, 2] ประกอบด้วยชั้นอินพุต 4 โนดจากจุดเข้าถึง 4 จุด, ชั้นเอาต์พุต 2 โนดคือตำแหน่ง (x, y) และชั้นซ่อนเร้น 1 ชั้นจำนวน 4 โนด

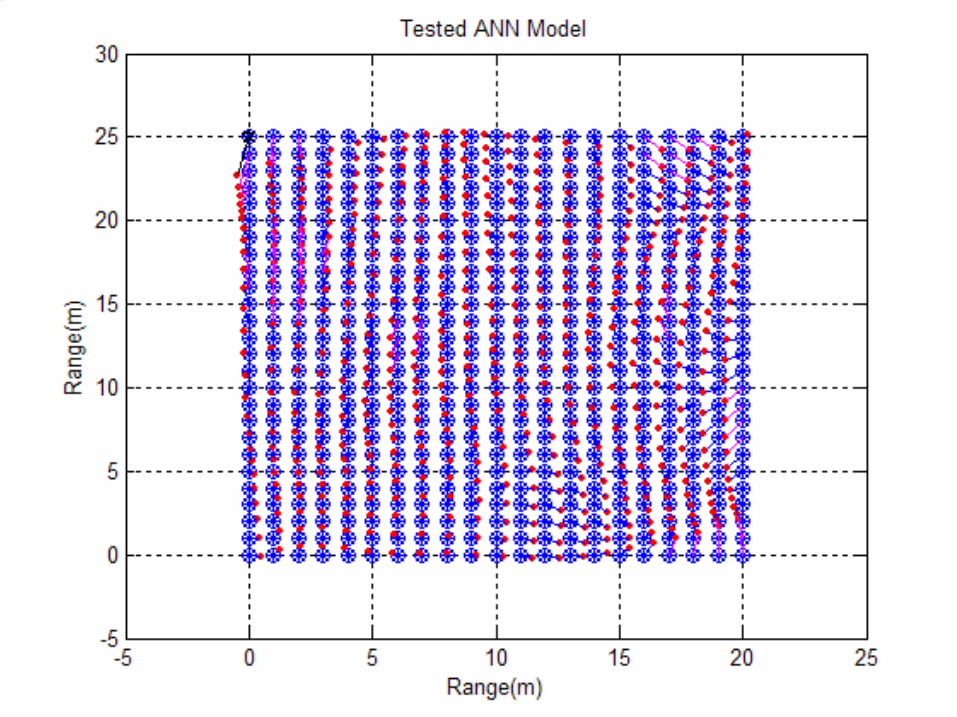
การหาโครงสร้างของ FF-MLPs ด้วยจีนเนติกอัลกอริทึมให้ผลเป็นที่น่าพอใจปีนอย่างยิ่ง โดยเฉพาะการนำเสนอแนวคิดในการใช้จีนเนติกอัลกอริทึมในการหาโครงสร้างของ FF-MLPs ซึ่งเป็นการยืนยันให้เห็นถึงจุดแข็งของการใช้วิธีการทางปัญญาประดิษฐ์ที่เรียกว่าจีนเนติกอัลกอริทึมกับงานวิจัยทางด้านวิศวกรรมรูปแบบหนึ่ง



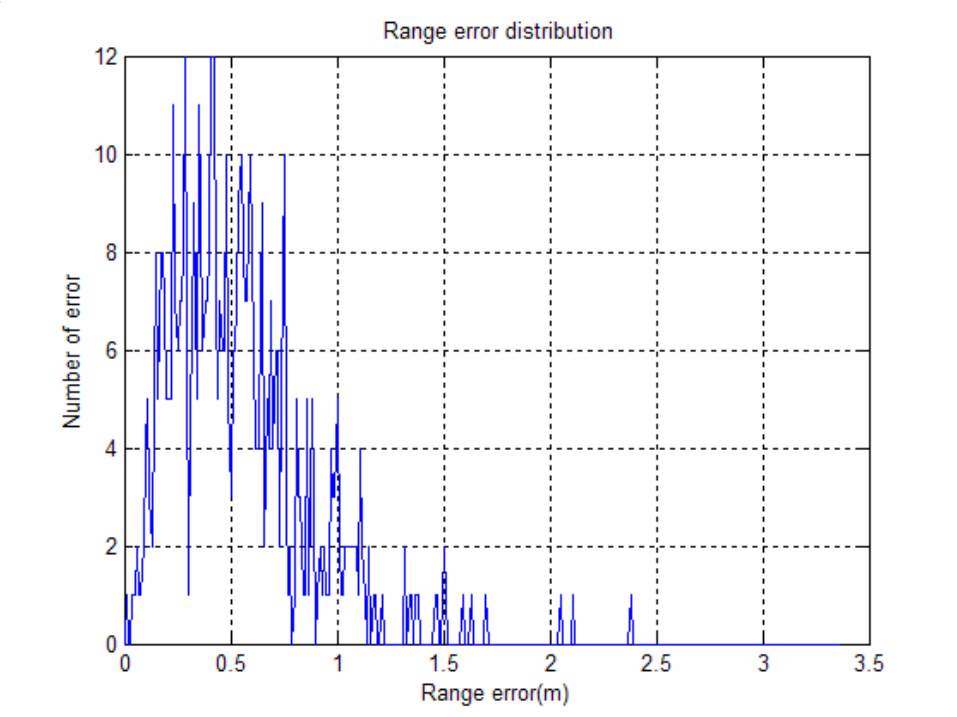
รูปที่ ก.4 ผลการทดสอบด้วย FF-MLPs โครงสร้างเดิมจากการจากการลองผิดลองถูก



รูปที่ ก.5 การกระจายของระยะคลาดเคลื่อนจากการลองผิดลองถูก



รูปที่ ก.6 ผลการทดสอบด้วย FF-MLPs โครงสร้างใหม่ที่ได้จากจีนเนติกอัลกอริทึม



รูปที่ ก.7 การกระจายของระยะคลาดเคลื่อนจากการทดสอบโครงสร้างได้จากจีนเนติกอัลกอริทึม

ภาคผนวก ข.

ผลการทดสอบการหาโครงสร้าง FF-MLPs ด้วยจีนเนติกอัลกอริทึม

ตารางที่ ข.1: ผลการทดสอบเมื่อเรียงลำดับตาม FF-Ranking และเปอร์เซ็นต์ของจุด OffError

Max_Gen=25, nPopulation=25, n_Bit=3, GGAP=0.6, FF-MLPs Train_Loop=4					
Item	AcceptError	P_Crossover	%OffError	FF_ANN	FF_Ranking
1	12	0.7	5.6777	4 0 4 0 0 0 2	100
2	8	0.7	7.6923	4 0 0 0 4 0 2	100
3	12	0.9	8.6081	4 0 4 0 0 0 2	100
4	12	0.9	11.3553	4 4 0 0 0 0 2	100
5	12	0.9	11.3553	4 4 0 0 0 0 2	100
6	12	0.9	11.3553	4 4 0 0 0 0 2	100
7	10	0.7	0.1832	4 5 0 0 0 0 2	110
8	8	0.9	0.1832	4 0 0 0 0 5 2	110
9	10	0.7	0.3663	4 0 0 0 0 5 2	110
10	12	0.7	0.3663	4 5 0 0 0 0 2	110
11	12	0.7	0.5495	4 5 0 0 0 0 2	110
12	8	0.9	0.5495	4 0 5 0 0 0 2	110
13	8	0.9	0.5495	4 5 0 0 0 0 2	110
14	12	0.9	0.7326	4 0 0 0 0 5 2	110
15	12	0.7	0.9158	4 0 0 0 0 5 2	110
16	12	0.7	0.9158	4 5 0 0 0 0 2	110
17	12	0.9	0.9158	4 0 0 0 5 0 2	110
18	10	0.7	1.0989	4 0 0 0 5 0 2	110
19	12	0.7	1.0989	4 0 5 0 0 0 2	110
20	12	0.7	1.0989	4 0 0 5 0 0 2	110
21	10	0.9	1.0989	4 0 5 0 0 0 2	110
22	10	0.7	1.2821	4 0 0 0 0 5 2	110
23	10	0.9	1.2821	4 5 0 0 0 0 2	110
24	12	0.9	1.2821	4 5 0 0 0 0 2	110
25	12	0.9	1.2821	4 0 0 0 5 0 2	110
26	12	0.7	1.4652	4 5 0 0 0 0 2	110
27	8	0.9	1.4652	4 5 0 0 0 0 2	110
28	12	0.7	1.8315	4 0 0 0 0 5 2	110
29	10	0.9	2.1978	4 0 5 0 0 0 2	110
30	12	0.9	2.1978	4 0 0 0 0 5 2	110
31	10	0.7	2.3810	4 0 0 5 0 0 2	110
32	10	0.9	2.3810	4 0 0 5 0 0 2	110
33	12	0.7	2.7473	4 0 0 5 0 0 2	110
34	10	0.9	2.9304	4 0 0 5 0 0 2	110
35	10	0.9	2.9304	4 0 0 0 0 5 2	110
36	12	0.7	3.4799	4 0 5 0 0 0 2	110
37	12	0.7	3.8462	4 0 0 0 5 0 2	110
38	10	0.9	5.1282	4 5 0 0 0 0 2	110
39	8	0.7	0.0000	4 0 0 0 6 0 2	120
40	8	0.7	0.0000	4 0 0 0 6 0 2	120
41	10	0.7	0.0000	4 0 0 6 0 0 2	120
42	10	0.7	0.0000	4 0 6 0 0 0 2	120
43	10	0.7	0.0000	4 0 0 0 6 0 2	120
44	12	0.7	0.0000	4 0 0 0 6 0 2	120
45	12	0.7	0.0000	4 0 0 0 6 0 2	120
46	8	0.9	0.0000	4 0 6 0 0 0 2	120
47	8	0.9	0.0000	4 0 0 0 0 6 2	120
48	10	0.9	0.0000	4 6 0 0 0 0 2	120
49	10	0.9	0.0000	4 6 0 0 0 0 2	120
50	10	0.9	0.0000	4 0 6 0 0 0 2	120
51	12	0.9	0.0000	4 0 6 0 0 0 2	120

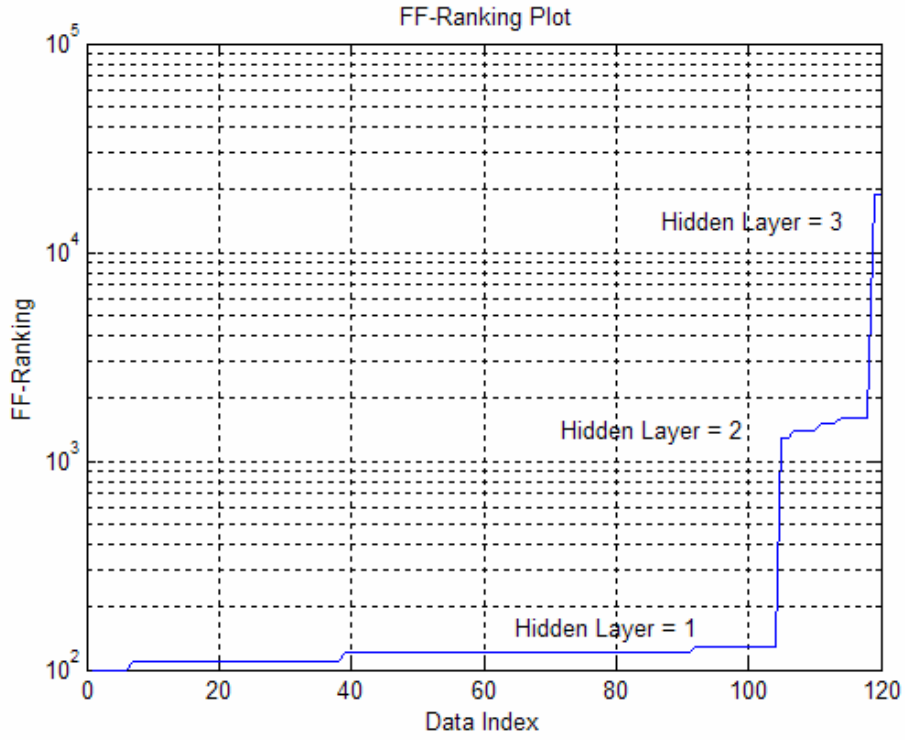
ตารางที่ ข.1: ผลการทดสอบเมื่อเรียงลำดับตาม FF-Ranking และเปอร์เซ็นต์ของจุด OffError (ต่อ)

Item	AcceptError	P_Crossover	%OffError	FF_ANN	FF_Ranking
52	12	0.9	0.0000	4 0 6 0 0 0 2	120
53	12	0.9	0.0000	4 0 0 6 0 0 2	120
54	8	0.7	0.1832	4 6 0 0 0 0 2	120
55	8	0.7	0.1832	4 6 0 0 0 0 2	120
56	10	0.7	0.1832	4 6 0 0 0 0 2	120
57	8	0.9	0.1832	4 6 0 0 0 0 2	120
58	10	0.9	0.1832	4 0 0 6 0 0 2	120
59	10	0.9	0.1832	4 0 0 0 6 0 2	120
60	12	0.9	0.1832	4 0 0 6 0 0 2	120
61	12	0.9	0.1832	4 0 0 0 6 0 2	120
62	8	0.7	0.3663	4 0 0 6 0 0 2	120
63	10	0.7	0.3663	4 0 0 0 6 0 2	120
64	10	0.7	0.3663	4 0 0 0 6 0 2	120
65	8	0.9	0.3663	4 0 6 0 0 0 2	120
66	8	0.9	0.3663	4 0 6 0 0 0 2	120
67	10	0.9	0.3663	4 0 0 6 0 0 2	120
68	12	0.9	0.3663	4 0 0 6 0 0 2	120
69	8	0.7	0.5495	4 6 0 0 0 0 2	120
70	12	0.7	0.5495	4 0 6 0 0 0 2	120
71	12	0.7	0.5495	4 0 0 0 6 0 2	120
72	10	0.9	0.5495	4 0 0 6 0 0 2	120
73	10	0.9	0.5495	4 0 0 0 0 6 2	120
74	10	0.9	0.5495	4 0 0 0 0 6 2	120
75	12	0.9	0.5495	4 6 0 0 0 0 2	120
76	12	0.9	0.5495	4 6 0 0 0 0 2	120
77	8	0.7	0.7326	4 0 6 0 0 0 2	120
78	10	0.7	0.9158	4 0 6 0 0 0 2	120
79	10	0.9	0.9158	4 0 6 0 0 0 2	120
80	8	0.7	1.0989	4 0 0 0 6 0 2	120
81	10	0.7	1.0989	4 0 0 0 6 0 2	120
82	8	0.9	1.0989	4 6 0 0 0 0 2	120
83	8	0.9	1.4652	4 0 0 0 0 6 2	120
84	8	0.9	1.4652	4 0 0 0 0 6 2	120
85	12	0.9	1.4652	4 0 0 6 0 0 2	120
86	8	0.9	1.6484	4 0 6 0 0 0 2	120
87	8	0.7	2.0147	4 6 0 0 0 0 2	120
88	8	0.7	2.0147	4 6 0 0 0 0 2	120
89	10	0.7	2.9304	4 6 0 0 0 0 2	120
90	8	0.7	4.3956	4 6 0 0 0 0 2	120
91	8	0.7	5.4945	4 0 0 0 0 6 2	120
92	8	0.7	0.0000	4 7 0 0 0 0 2	130
93	8	0.7	0.0000	4 0 0 7 0 0 2	130
94	8	0.7	0.0000	4 7 0 0 0 0 2	130
95	10	0.7	0.0000	4 0 7 0 0 0 2	130
96	10	0.7	0.0000	4 7 0 0 0 0 2	130
97	12	0.7	0.0000	4 0 0 0 7 0 2	130
98	8	0.9	0.0000	4 0 0 0 7 0 2	130
99	12	0.9	0.0000	4 7 0 0 0 0 2	130
100	12	0.9	0.3663	4 0 0 7 0 0 2	130
101	8	0.9	1.0989	4 0 0 7 0 0 2	130
102	8	0.7	1.6484	4 7 0 0 0 0 2	130
103	8	0.9	2.0147	4 0 7 0 0 0 2	130
104	8	0.7	5.1282	4 7 0 0 0 0 2	130
105	10	0.7	3.1136	4 0 0 0 5 2 2	1300
106	12	0.7	4.5788	4 0 4 3 0 0 2	1300

ตารางที่ ข.1: ผลการทดสอบเมื่อเรียงลำดับตาม FF-Ranking และเปอร์เซ็นต์ของจุด OffError (ต่อ)

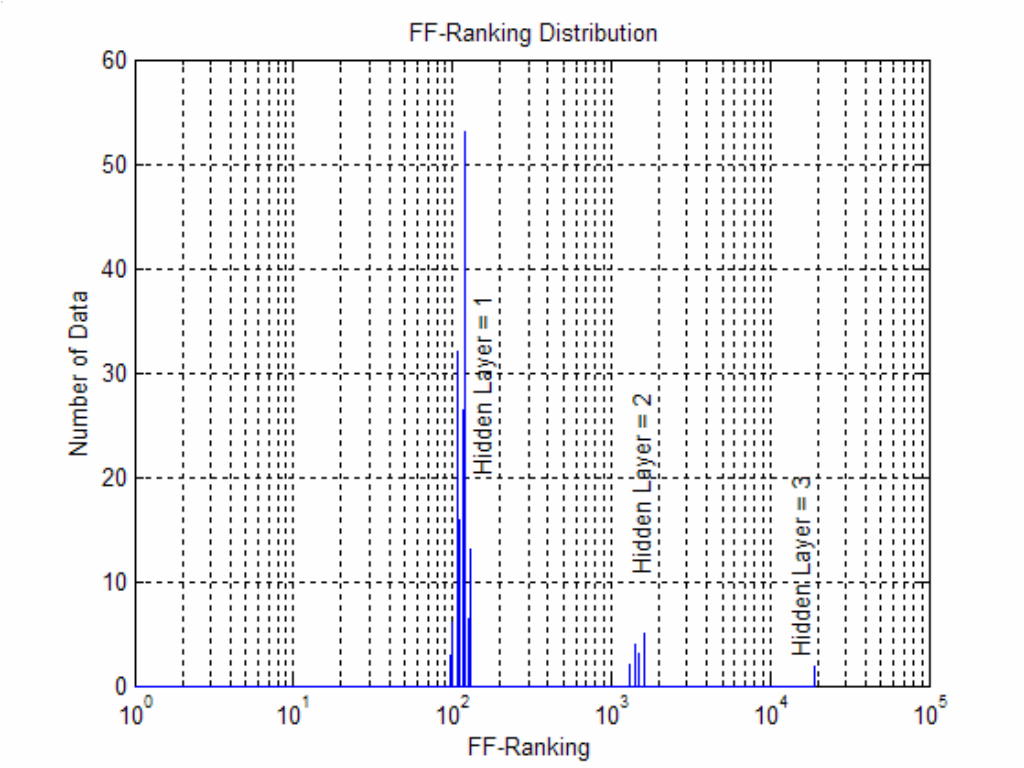
Item	AcceptError	P_Crossover	%OffError	FF_ANN	FF_Ranking
107	10	0.9	0.7326	4 0 0 6 0 2 2	1400
108	8	0.9	1.0989	4 6 2 0 0 0 2	1400
109	8	0.9	3.1136	4 5 0 0 0 3 2	1400
110	8	0.7	4.3956	4 5 0 0 0 3 2	1400
111	12	0.7	0.7326	4 0 5 0 4 0 2	1500
112	10	0.9	0.7326	4 5 4 0 0 0 2	1500
113	12	0.7	1.0989	4 7 2 0 0 0 2	1500
114	8	0.7	0.0000	4 0 0 3 0 7 2	1600
115	10	0.7	0.0000	4 0 5 0 5 0 2	1600
116	10	0.9	0.1832	4 5 0 0 5 0 2	1600
117	8	0.9	0.9158	4 0 7 3 0 0 2	1600
118	10	0.7	5.1282	4 0 4 0 0 6 2	1600
119	10	0.7	0.0000	4 2 6 5 0 0 2	19000
120	8	0.9	0.0000	4 4 0 0 6 3 2	19000

จากผลการทดสอบจำนวน 120 ตัวอย่างการค้นหาที่มีการปรับเปลี่ยนตัวแปรต่าง ๆ ของจีนเนติกอัลกอริทึมเพื่อหาโครงสร้าง FF-MLPs เล็กที่สุดในการเป็นแบบจำลองของห้องพบว่าโดยส่วนใหญ่ได้โครงสร้างที่มีชั้นซ่อนเร้น 1 ชั้น และมีโครงสร้างเล็กสุดที่ได้คือ 4-4-2 ให้จำนวนจุดคลาดเคลื่อนจากจุดอ้างอิงมากกว่า 0.2 เมตรจำนวน 5.6777 เปอร์เซ็นต์



รูปที่ ข.1 ค่า FF-Ranking จากผลการทำงานของจีนเนติกอัลกอริทึม 120 ตัวอย่าง

จากผลการทำการค้นหาโครงสร้างของจีนเนติกอัลกอริทึม 120 ตัวอย่างนำค่า FF-Ranking ของตัวอย่างทั้งหมดมาเรียงลำดับจากน้อยสุดไปหามากสุดนำค่า FF-Ranking แสดงในรูปกราฟรูปที่ ข.1 แกนตั้งคือค่า FF-Ranking แกนนอนเป็นค่าลำดับที่ของตัวอย่างทดสอบ จะเห็นว่าตัวอย่างส่วนใหญ่ให้โครงสร้าง FF-MLPs แบบมีชั้นซ่อนเร้น 1 ชั้น รูปที่ ข.2 แสดงการกระจายของ FF-Ranking โดยแกนนอนเป็นค่า FF-Ranking แกนตั้งเป็นจำนวนครั้งของ FF-Ranking ที่ให้ผลการคำนวณซ้ำกัน



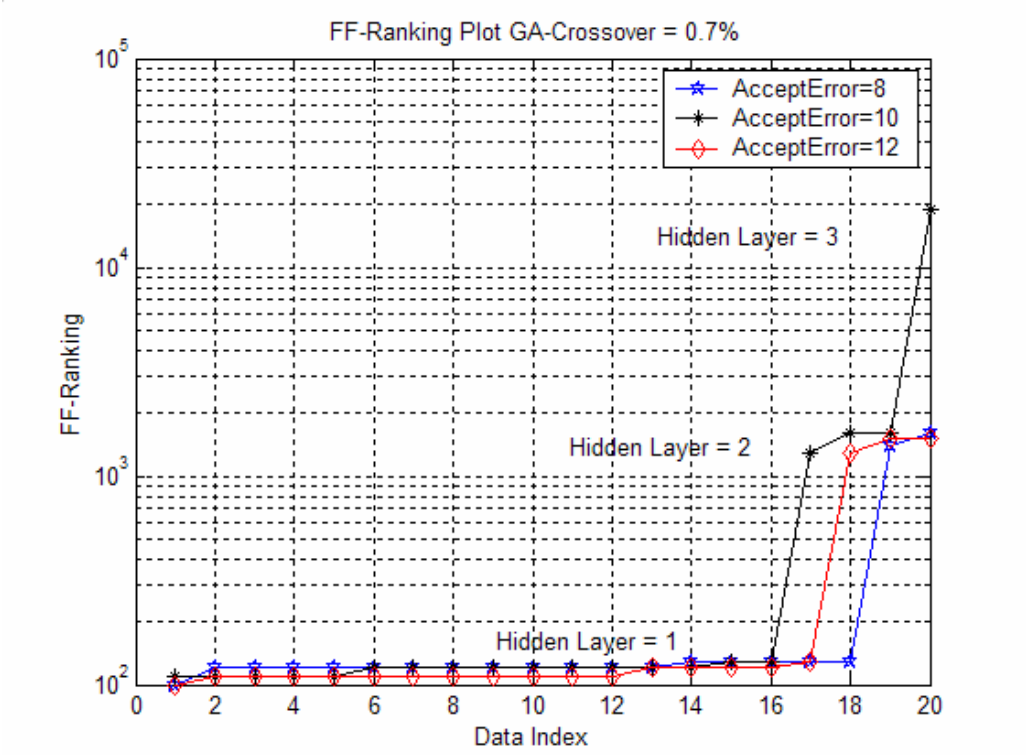
รูปที่ ข.2 การกระจายของ FF-Ranking ของผลการค้นหาด้วยจีนเนติกอัลกอริทึม

ตารางที่ ข.2: ผลการทดสอบเมื่อกำหนดค่าเปอร์เซ็นต์การทำครอสโอเวอร์ของจีนเนติกเท่ากับ 0.7 เรียงลำดับตามค่า AcceptError, FF_Ranking และค่าเปอร์เซ็นต์ OffError

Max_Gen=25, nPopulation=25, n_Bit=3, GGAP=0.6, FF-MLPs Train_Loop=4					
Item	AcceptError	P_Crossover	%OffError	FF_ANN	FF_Ranking
1	8	0.7	7.6923	4 0 0 0 4 0 2	100
2	8	0.7	0.0000	4 0 0 0 6 0 2	120
3	8	0.7	0.0000	4 0 0 0 6 0 2	120
4	8	0.7	0.1832	4 6 0 0 0 0 2	120
5	8	0.7	0.1832	4 6 0 0 0 0 2	120
6	8	0.7	0.3663	4 0 0 6 0 0 2	120
7	8	0.7	0.5495	4 6 0 0 0 0 2	120
8	8	0.7	0.7326	4 0 6 0 0 0 2	120
9	8	0.7	1.0989	4 0 0 0 6 0 2	120
10	8	0.7	2.0147	4 6 0 0 0 0 2	120
11	8	0.7	2.0147	4 6 0 0 0 0 2	120
12	8	0.7	4.3956	4 6 0 0 0 0 2	120
13	8	0.7	5.4945	4 0 0 0 0 6 2	120
14	8	0.7	0.0000	4 7 0 0 0 0 2	130
15	8	0.7	0.0000	4 0 0 7 0 0 2	130
16	8	0.7	0.0000	4 7 0 0 0 0 2	130
17	8	0.7	1.6484	4 7 0 0 0 0 2	130
18	8	0.7	5.1282	4 7 0 0 0 0 2	130
19	8	0.7	4.3956	4 5 0 0 0 3 2	1400
20	8	0.7	0.0000	4 0 0 3 0 7 2	1600
21	10	0.7	0.1832	4 5 0 0 0 0 2	110
22	10	0.7	0.3663	4 0 0 0 0 5 2	110
23	10	0.7	1.0989	4 0 0 0 5 0 2	110
24	10	0.7	1.2821	4 0 0 0 0 5 2	110
25	10	0.7	2.3810	4 0 0 5 0 0 2	110
26	10	0.7	0.0000	4 0 0 6 0 0 2	120
27	10	0.7	0.0000	4 0 6 0 0 0 2	120
28	10	0.7	0.0000	4 0 0 0 6 0 2	120
29	10	0.7	0.1832	4 6 0 0 0 0 2	120
30	10	0.7	0.3663	4 0 0 0 6 0 2	120
31	10	0.7	0.3663	4 0 0 0 6 0 2	120
32	10	0.7	0.9158	4 0 6 0 0 0 2	120
33	10	0.7	1.0989	4 0 0 0 6 0 2	120
34	10	0.7	2.9304	4 6 0 0 0 0 2	120
35	10	0.7	0.0000	4 0 7 0 0 0 2	130
36	10	0.7	0.0000	4 7 0 0 0 0 2	130
37	10	0.7	3.1136	4 0 0 0 5 2 2	1300
38	10	0.7	0.0000	4 0 5 0 5 0 2	1600
39	10	0.7	5.1282	4 0 4 0 0 6 2	1600
40	10	0.7	0.0000	4 2 6 5 0 0 2	19000
41	12	0.7	5.6777	4 0 4 0 0 0 2	100
42	12	0.7	0.3663	4 5 0 0 0 0 2	110
43	12	0.7	0.5495	4 5 0 0 0 0 2	110
44	12	0.7	0.9158	4 0 0 0 0 5 2	110
45	12	0.7	0.9158	4 5 0 0 0 0 2	110
46	12	0.7	1.0989	4 0 5 0 0 0 2	110
47	12	0.7	1.0989	4 0 0 5 0 0 2	110
48	12	0.7	1.4652	4 5 0 0 0 0 2	110
49	12	0.7	1.8315	4 0 0 0 0 5 2	110
50	12	0.7	2.7473	4 0 0 5 0 0 2	110
51	12	0.7	3.4799	4 0 5 0 0 0 2	110

ตารางที่ ข.2: ผลการทดสอบเมื่อกำหนดค่าเปอร์เซ็นต์การทำครอสโอเวอร์ของจีนเนติกเท่ากับ 0.7 เรียงลำดับตามค่า AcceptError, FF_Ranking และค่าเปอร์เซ็นต์ OffError (ต่อ)

Item	AcceptError	P_Crossover	%OffError	FF_ANN	FF_Ranking
52	12	0.7	3.8462	4 0 0 0 5 0 2	110
53	12	0.7	0.0000	4 0 0 0 6 0 2	120
54	12	0.7	0.0000	4 0 0 0 6 0 2	120
55	12	0.7	0.5495	4 0 6 0 0 0 2	120
56	12	0.7	0.5495	4 0 0 0 6 0 2	120
57	12	0.7	0.0000	4 0 0 0 7 0 2	130
58	12	0.7	4.5788	4 0 4 3 0 0 2	1300
59	12	0.7	0.7326	4 0 5 0 4 0 2	1500
60	12	0.7	1.0989	4 7 2 0 0 0 2	1500



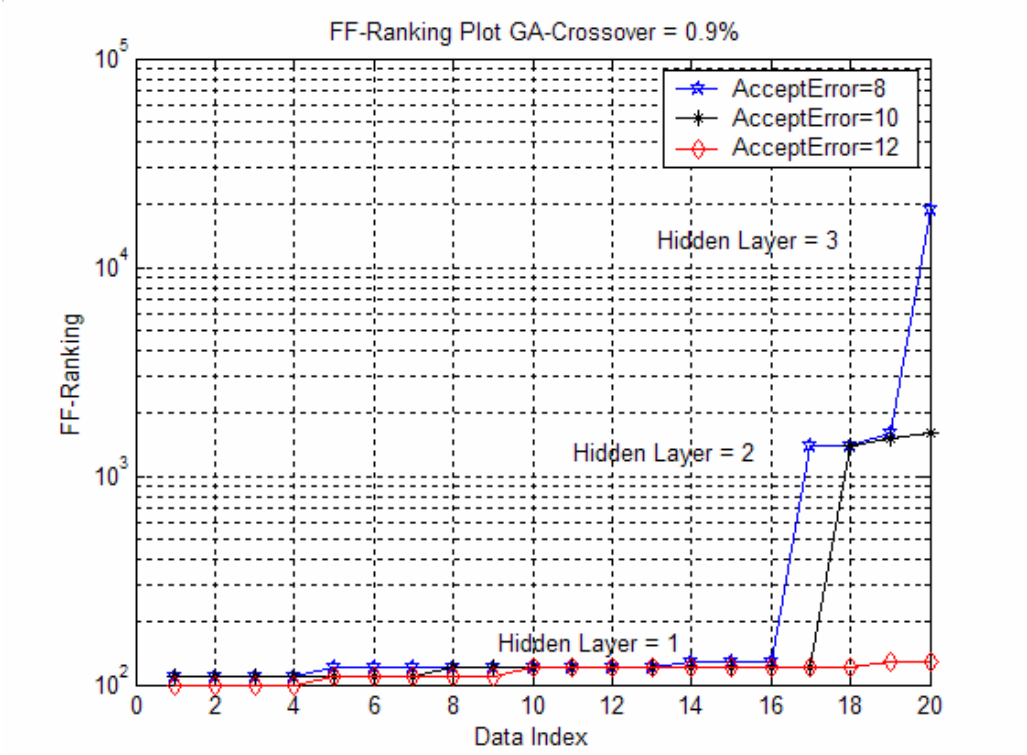
รูปที่ ข.3 ค่า FF-Ranking เมื่อกำหนดตัวแปร GA-Crossover 0.7 เปอร์เซ็นต์

ตารางที่ ข.3: ผลการทดสอบเมื่อกำหนดค่าเปอร์เซ็นต์การทำครอสโอเวอร์ของจีนเนติกเท่ากับ 0.9 เรียงลำดับตามค่า AcceptError, FF_Ranking และค่าเปอร์เซ็นต์ OffError

Max_Gen=25, nPopulation=25, n_Bit=3, GGAP=0.6, FF-MLPs Train_Loop=4					
Item	AcceptError	P_Crossover	%OffError	FF_ANN	FF_Ranking
1	8	0.9	0.1832	4 0 0 0 0 5 2	110
2	8	0.9	0.5495	4 0 5 0 0 0 2	110
3	8	0.9	0.5495	4 5 0 0 0 0 2	110
4	8	0.9	1.4652	4 5 0 0 0 0 2	110
5	8	0.9	0.0000	4 0 6 0 0 0 2	120
6	8	0.9	0.0000	4 0 0 0 0 6 2	120
7	8	0.9	0.1832	4 6 0 0 0 0 2	120
8	8	0.9	0.3663	4 0 6 0 0 0 2	120
9	8	0.9	0.3663	4 0 6 0 0 0 2	120
10	8	0.9	1.0989	4 6 0 0 0 0 2	120
11	8	0.9	1.4652	4 0 0 0 0 6 2	120
12	8	0.9	1.4652	4 0 0 0 0 6 2	120
13	8	0.9	1.6484	4 0 6 0 0 0 2	120
14	8	0.9	0.0000	4 0 0 0 7 0 2	130
15	8	0.9	1.0989	4 0 0 7 0 0 2	130
16	8	0.9	2.0147	4 0 7 0 0 0 2	130
17	8	0.9	1.0989	4 6 2 0 0 0 2	1400
18	8	0.9	3.1136	4 5 0 0 0 3 2	1400
19	8	0.9	0.9158	4 0 7 3 0 0 2	1600
20	8	0.9	0.0000	4 4 0 0 6 3 2	19000
21	10	0.9	1.0989	4 0 5 0 0 0 2	110
22	10	0.9	1.2821	4 5 0 0 0 0 2	110
23	10	0.9	2.1978	4 0 5 0 0 0 2	110
24	10	0.9	2.3810	4 0 0 5 0 0 2	110
25	10	0.9	2.9304	4 0 0 5 0 0 2	110
26	10	0.9	2.9304	4 0 0 0 0 5 2	110
27	10	0.9	5.1282	4 5 0 0 0 0 2	110
28	10	0.9	0.0000	4 6 0 0 0 0 2	120
29	10	0.9	0.0000	4 6 0 0 0 0 2	120
30	10	0.9	0.0000	4 0 6 0 0 0 2	120
31	10	0.9	0.1832	4 0 0 6 0 0 2	120
32	10	0.9	0.1832	4 0 0 0 6 0 2	120
33	10	0.9	0.3663	4 0 0 6 0 0 2	120
34	10	0.9	0.5495	4 0 0 6 0 0 2	120
35	10	0.9	0.5495	4 0 0 0 0 6 2	120
36	10	0.9	0.5495	4 0 0 0 0 6 2	120
37	10	0.9	0.9158	4 0 6 0 0 0 2	120
38	10	0.9	0.7326	4 0 0 6 0 2 2	1400
39	10	0.9	0.7326	4 5 4 0 0 0 2	1500
40	10	0.9	0.1832	4 5 0 0 5 0 2	1600
41	12	0.9	8.6081	4 0 4 0 0 0 2	100
42	12	0.9	11.3553	4 4 0 0 0 0 2	100
43	12	0.9	11.3553	4 4 0 0 0 0 2	100
44	12	0.9	11.3553	4 4 0 0 0 0 2	100
45	12	0.9	0.7326	4 0 0 0 0 5 2	110
46	12	0.9	0.9158	4 0 0 0 5 0 2	110
47	12	0.9	1.2821	4 5 0 0 0 0 2	110
48	12	0.9	1.2821	4 0 0 0 5 0 2	110
49	12	0.9	2.1978	4 0 0 0 0 5 2	110
50	12	0.9	0.0000	4 0 6 0 0 0 2	120
51	12	0.9	0.0000	4 0 6 0 0 0 2	120

ตารางที่ ข.3: ผลการทดสอบเมื่อกำหนดค่าเปอร์เซ็นต์การทำครอสโอเวอร์ของจีนเนติกเท่ากับ 0.9 เรียงลำดับตามค่า AcceptError, FF_Ranking และค่าเปอร์เซ็นต์ OffError (ต่อ)

Item	AcceptError	P_Crossover	%OffError	FF_ANN	FF_Ranking
52	12	0.9	0.0000	4 0 0 6 0 0 2	120
53	12	0.9	0.1832	4 0 0 6 0 0 2	120
54	12	0.9	0.1832	4 0 0 0 6 0 2	120
55	12	0.9	0.3663	4 0 0 6 0 0 2	120
56	12	0.9	0.5495	4 6 0 0 0 0 2	120
57	12	0.9	0.5495	4 6 0 0 0 0 2	120
58	12	0.9	1.4652	4 0 0 6 0 0 2	120
59	12	0.9	0.0000	4 7 0 0 0 0 2	130
60	12	0.9	0.3663	4 0 0 7 0 0 2	130



รูปที่ ข.4 ค่า FF-Ranking เมื่อกำหนดตัวแปร GA-Crossover 0.9 เปอร์เซ็นต์

ภาคผนวก ค.

โปรแกรม MATLAB® สำหรับให้จีนเนติกอัลกอริทึมทำงาน เพื่อหาโครงสร้างของ FF-MLPs

โปรแกรม MATLAB® สำหรับให้จีนเนติกอัลกอริทึมทำงานเพื่อหาโครงสร้างของ FF-MLPs

การทำงานของ MATLAB® ประกอบด้วยโปรแกรมหลักและโปรแกรมย่อยทั้งหมด 5 โปรแกรม แบ่งหน้าที่การทำงานต่าง ๆ ดังนี้

1. โปรแกรม GA_PeakAI เป็นโปรแกรมหลักสำหรับให้จีนเนติกอัลกอริทึมทำงาน ผลลัพธ์การทำงานจะเก็บในรูปของไฟล์ข้อมูลเพื่อให้งานในการนำไปใช้ต่อไป
2. โปรแกรม GA_Objfunction เป็นโปรแกรมย่อยสำหรับการคำนวนค่าของฟังก์ชันวัตถุประสงค์
3. โปรแกรม p2_plotTest เป็นโปรแกรมย่อยสำหรับคำนวณค่า OffError และแสดงกราฟระยะทางคลาดเคลื่อนจากผลการทำงานของ FF-MLPs
4. โปรแกรม GA_SortData เป็นโปรแกรมย่อยสำหรับจัดเรียงข้อมูลในอยู่ในรูปที่เหมาะในการนำไปใช้คำนวณของ FF-MLPs
5. โปรแกรม GA_ErrorTest เป็นโปรแกรมย่อยสำหรับคำนวณค่า OffError โดยไม่มีการแสดงรูปกราฟระยะคลาดเคลื่อน

โปรแกรมหลัก Function GA_PeakAI

```
%=====================================================
function GA_PeakAi
%=====================================================
clc,clear all, close all, fprintf('\n');
Run_mode = 0;   % 0=New_Run 1=Countinue
file_Name = 'GA_Data41';

%-----------------------------------------------------
%Step1: GA Initial
AcceptError = 20;      % Maximum Error Accept 5% (Correct  95%)
P_Crossover = 0.9;     % Percent of Crossover
Max_Gen     = 25;      % 25 max num of generation
nPopulation = 10;      % 25 num of chromosome(population)
test_Loop   = 4;       % 4 num of test loop @Population
train_Loop  = 400;     % 400 Success Minimum 300 num of ff-ANN train loop
n_Bit       = 3;       % nBitsion of variables
GGAP        = 0.8;     % Percent of Generation Gap
n_Var       = 5;       % num of variables {Max Hidden Layer}
Max_Var     = 7;       % Max of variables..[7]
Min_Var     = 0;       % Min of variables..[0]
Rec_Error   = 99999999999999999.99;
MaskField   = ones(1,n_Var*n_Bit);

file_PName  = sprintf('%s%s',file_Name,'_Par.mat');
file_Name   = sprintf('%s%s',file_Name,'.mat');
save(file_PName,'Max_Gen','nPopulation','test_Loop',…
       'AcceptError','n_Bit','GGAP','P_Crossover');

%-----------------------------------------------------
%Step2: System Initial
load('Data_T21.mat');     % nRow nCol 4Data[x,y,?,mean] 4AP
[nGrid_Row nGrid_Col nData_Type nAP] = size(Data);
[RefInput ,Target] = GA_SortData(Data);
```

```
%Step3: Build field descriptor
FieldD =[   rep([n_Bit],            [1,n_Var]);...
            rep([Min_Var;Max_Var],  [1,n_Var]);...
            rep([0;0;1;1],          [1,n_Var])];
%-----------------------------------------------------
if Run_mode==0
%Step4: Initialize population
    Chrom = crtbp(nPopulation,n_Var*n_Bit);
    gen=1;  %counter

%Step5: evaluate initial population
    tCount = 0;
    MaskField = [ones(1,n_Var*n_Bit)];
    fprintf('--####-- Evaluate initial population --####--\n');
    [ObjV,best_chrom,best_chrom_index,best_net,best_error,tCount]...
        =   GA_Objfunction(bs2rv(Chrom,FieldD),RefInput,Target,test_Loop,...
            train_Loop,gen,1,tCount,AcceptError,Max_Var);
    tCount = 0;
    fprintf('\n\n');
else
    load(file_Name)
end

%Step6: GA generalation loop
while (gen <= Max_Gen)
    %Step6.1: Assign fitness values to entire population
    FitnV = ranking(ObjV);

    %Step6.2: Select chromosomes for breeding
    SelCh = select('sus',Chrom,FitnV,GGAP);

    %Step6.3: Recombine chromosomes (crossover)
    SelCh = recombin('xovsp',SelCh,P_Crossover);

    %Step6.4: Apply mutation
    SelCh = mut(SelCh);

    %- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
    %---=== Set masking File ===----
    if best_error < AcceptError
        for loop_MaskField = 1:(size(best_chrom,2)-2)   % [ 1 xxx 1 ]
            if (best_chrom(loop_MaskField+1) == 0)      % [         ]
                for loop2_MaskField = 1:n_Bit
                    MaskField((loop_MaskField-1)*n_Bit+loop2_MaskField) = 0;
                end
            end
        end
    end
    %---=== Mul Mask File ===----
    for i_mask = 1:size(SelCh,1)
       SelCh(i_mask,:) =  SelCh(i_mask,:) .* MaskField;
    end
    %- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

    %Step6.5: Evaluate offspring, call objective function
    fprintf('--####-- Evaluate offspring Generation:%3d/%3d --####--\n',gen,Max_Gen);
    [ObjVSel,best_chrom,best_chrom_index,best_net,best_error,tCount]...
        =   GA_Objfunction(bs2rv(SelCh,FieldD),RefInput,Target,test_Loop,...
            train_Loop,gen,Max_Gen,tCount,AcceptError,Max_Var);

    %Step6.6: Reinsert offspring into population
    [Chrom ObjV] = reins(Chrom,SelCh,1,1,ObjV,ObjVSel);

    %Step6.7: Increment counter
    %---=== Save Best Error ===----
    refChrom = [1 ,Max_Var ,Max_Var ,Max_Var ,Max_Var ,Max_Var ,1];
    if gen == 1
        Rec_Error = best_error;
        Rec_chrom = best_chrom;
        Rec_net   = best_net;
    end
```

```
    sum_tstChrom = sum(best_chrom);
    sum_recChrom = sum(Rec_chrom);
    if (best_error < AcceptError)&(sum_tstChrom < sum_recChrom)
        Rec_Error = best_error;
        Rec_chrom = best_chrom;
        Rec_net = best_net;
    end

    if best_error < Rec_Error
        Rec_Error = best_error;
        Rec_chrom = best_chrom;
        Rec_net   = best_net;
    end

    %---=== Store tResult DAT ===----
    t_Result_Dat = [gen, Rec_Error ,Rec_chrom, best_error, best_chrom];
    if gen ==1
        Result_Dat = t_Result_Dat;
    else
        load(file_Name,'Result_Dat');
        Result_Dat = [Result_Dat; t_Result_Dat;];
    end
    %---=== Save All Data to File ===----
    gen = gen+1;
    fprintf('Select Chromosome =[%2d %2d %2d %2d %2d %2d %2d]\t',Rec_chrom);
    fprintf('Select Error = %8.4f\n',Rec_Error);
    save(file_Name,'Result_Dat','Rec_net','Rec_Error','Rec_chrom','best_error',…
        'best_chrom','best_net','gen','Chrom','tCount','ObjV','ObjVSel',…
        'MaskField','RefInput','Target');

    netOutput = sim(Rec_net,RefInput);
    [Norm_Average_m,Norm_Max_m,P10,P05,P02,n] = GA_plotTest(Target,netOutput);
    fprintf('In Pause key \n'); pause(1)


    fprintf('\n\n');
end

%-------------------------------------------------------
%Step7: Display Result
netOutput = sim(Rec_net,RefInput);
[Norm_Average_m,Norm_Max_m,P10,P05,P02,n] = GA_plotTest(Target,netOutput);
fprintf('Select Chromosome =[%2d %2d %2d %2d %2d %2d %2d]\t',Rec_chrom);
fprintf('Select Error = %8.4f\n',Rec_Error);
```

โปรแกรมย่อยที่ 1: Function GA_Objfunction

```
%=========================================================
function [ObjVSel,best_chrom,best_chrom_index,best_net,best_error,rtCount]...
    =   GA_Objfunction(ObjV,RefInput,Target,test_Loop,...
        train_Loop,gen,Max_Gen,tCount,AcceptError,Max_Var);
%=========================================================
    % ObjV = Matrix of chromosome 50x41 Data
    % ObjVSel = Matrix of Error @ chromosome 50  Data
    max_Chrom = size(ObjV,1);
    best_error = 99999999999999999.9;
    for idex=1:max_Chrom
        %-------------------------------------------------
        mChromosome = [Max_Var ,Max_Var ,Max_Var ,Max_Var ,Max_Var];
        tChromosome = ObjV(idex,:);
        n_Var = size(tChromosome,2);
        tChromosome = round(tChromosome);
        net_Traf = 0;
        for i_node = 1:size(tChromosome,2)
            if tChromosome(i_node) ~= 0
                if net_Traf==0
                    net_Node = tChromosome(i_node);
                else
                    net_Node = [net_Node tChromosome(i_node)];
                end
                net_Traf = net_Traf + 1;
            end
        end
        %-------------------------------------------------
        test_error = 999999999999999.9;
        for t_index=1:test_Loop
            tCount = tCount + 1;
            X_Input = [0,1; 0,1; 0,1; 0,1;];
            switch net_Traf
                case 0
                    net = newff(X_Input,[4 2],{'tansig' 'purelin'});
                case 1
                    net = newff(X_Input,[4 net_Node 2],{'tansig' 'tansig'…
                                'purelin'});
                case 2
                    net = newff(X_Input,[4 net_Node 2],{'tansig' 'tansig'…
                                'tansig','purelin'});
                case 3
                    net = newff(X_Input,[4 net_Node 2],{'tansig' 'tansig'…
                                'tansig','tansig' 'purelin'});
                case 4
                    net = newff(X_Input,[4 net_Node 2],{'tansig' 'tansig'…
                                'tansig','tansig' 'tansig' 'purelin'});
                case 5
                    net = newff(X_Input,[4 net_Node 2],{'tansig' 'tansig'…
                                'tansig','tansig' 'tansig' 'tansig' 'purelin'});
            end
            tic;        % Start Timer Count
            net.trainParam.epochs = train_Loop;
            net.trainParam.show = Inf;  % Not Show Training Value
            net = train(net,RefInput,Target);
            netOutput = sim(net,RefInput);
            [Norm_Average_m,Norm_Max_m,P10,P05,P02,n] = GA_ErrorTest(Target,neutput);
            netError  = 100 - P02;
            maxProcess = Max_Gen*max_Chrom*test_Loop;
            Process = tCount*100/maxProcess;
            t = toc;     % Stop Timer Count

            EOT = datestr(now + (100-Process)*t/100000);
            fprintf('#Test [4 %2d %2d %2d %2d %2d  2]',tChromosome);
            fprintf(' ,gen:%2d/%2d ,Chrom:%2d/%2d ,Loop:%2d ,OffError=…
                      %8.4f%%',gen,Max_Gen,idex,max_Chrom,t_index,netError);
            fprintf(' ,Process=%6.2f%%(%d/%d), EOT=…
                      %s\n',Process,tCount,maxProcess,EOT);
            rtCount = tCount;   % Return tCount
            k1 = sum(tChromosome)*10^(sum(round(tChromosome./(tChromosome+0.001))));
            k2 = sum(mChromosome)*10^(sum(round(mChromosome./(mChromosome+0.001))));
            if netError < test_error
                test_error = netError;
                test_net   = net;
                if netError < AcceptError
                    ObjVSel(idex,1) = AcceptError * k1;
                    ObjVSel(idex,1) = netError * k2;
```

```
            end
        end
    end
    %-------------------------------------------------
    if idex==1
        best_chrom_index = idex;
        best_error = test_error;
        best_chrom = tChromosome;
        best_net = test_net;
    end

    sum_tChrom = sum(tChromosome);
    sum_bChrom = sum(best_chrom);
    if (test_error < AcceptError)&(sum_tChrom < sum_bChrom)
        best_chrom_index = idex;
        best_error = test_error;
        best_chrom = tChromosome;
        best_net = test_net;
    end

    if test_error < best_error
        best_chrom_index = idex;
        best_error = test_error;
        best_chrom = tChromosome;
        best_net = test_net;
    end
    %-------------------------------------------------
end

best_chrom = [4, best_chrom, 2];
fprintf('Best Chromosome   =[%2d %2d %2d %2d %2d %2d %2d]\t',best_chrom);
fprintf('Best Error   = %8.4f\n',best_error);
```

โปรแกรมย่อยที่ 2: Function p2_plotTest

```
%=================================================
%       Plot result ann Input
%=================================================
function [Norm_Average_m,Norm_Max_m,P10,P05,P02,n] = p2_plotTest(ref_Data,net_Data);

figure(9);
hold off;   plot(ref_Data(1,:),ref_Data(2,:),'ob');
hold on;    plot(ref_Data(1,:),ref_Data(2,:),'*b');
grid on;    plot(net_Data(1,:),net_Data(2,:),'.r');

%-------------------------------------
L_Summ = 0;    Norm_Max_m = 0;
Under_02M=0; Under_05M=0; Under_10M=0;
%-------------------------------------
for k=1:size(ref_Data,2)
    i  = [ ref_Data(1,k) , net_Data(1,k)];  %(x1,x2)
    j  = [ ref_Data(2,k) , net_Data(2,k)];  %(y1,y2)
    ii =   ref_Data(1,k) - net_Data(1,k);   %(x1-x2)
    jj =   ref_Data(2,k) - net_Data(2,k);   %(y1-y2)

    xNorm  = sqrt(ii^2+jj^2);
    L_Summ = L_Summ + xNorm;
    %======== Over 2.0 m ========
    if xNorm < 1.0
        plot(i,j,'b');
        Under_10M = Under_10M + 1;
    else
        plot(i,j,'m');
    end
    %======== Over 0.5 m ========
    if xNorm < 0.5
        Under_05M = Under_05M + 1;
    end
    %======== Over 0.2 m ========
    if xNorm < 0.2
        Under_02M = Under_02M + 1;
    end
    %========= Max Norm =========
    if xNorm > Norm_Max_m
        Norm_Max_m = xNorm;
        Max = [ref_Data(1,k),ref_Data(2,k)];
        i_Max = i;
        j_Max = j;
    end
end

title(' Tested ANN Model ');
Xlabel('Range(m)');ylabel('Range(m)');
plot(Max(1),Max(2),'pk');   % Start at Max
plot(i_Max,j_Max,'k');      % Line at Max
n = size(ref_Data,2);       % Number of Data
Norm_Average_m  = L_Summ/n; % Average
P02 = Under_02M*100 / n;      % Percent of error Point
P05 = Under_05M*100 / n;      % Percent of error Point
P10 = Under_10M*100 / n;      % Percent of error Point

fprintf('\n=============== Result ================\n');
fprintf('Deviation under 1.0 m. = %3d/%3d(%6.2f%%)\n',Under_10M,n,P10);
fprintf('Deviation under 0.5 m. = %3d/%3d(%6.2f%%)\n',Under_05M,n,P05);
fprintf('Deviation under 0.2 m. = %3d/%3d(%6.2f%%)\n',Under_02M,n,P02);
fprintf('Norm Average = %5.4f, (%e) Unit\n',Norm_Average_m,Norm_Average_m);
fprintf('Norm Max = %5.4f, (%e)  Unit
@(%3d,%3d)\n',Norm_Max_m,Norm_Max_m,Max(1),Max(2));
fprintf('=======================================\n');
```

โปรแกรมย่อยที่ 3: Function GA_SortData

```
%=================================================
%      Sort Data
%=================================================
function [D_Input1 ,D_Output1] = GA_SortData(Data);
[nGrid_Row nGrid_Col nData_Type nAP] = size(Data);

k=1;
for i=1:4:nGrid_Row            % Index i = ON Y-axis 11 Data
    for j = 1:4:nGrid_Col      % Index j = ON X-axis 10 Data
        xData(1,k) = Data(i,j,4,1);     % Mean of AP1
        xData(2,k) = Data(i,j,4,2);     % Mean of AP3x
        xData(3,k) = Data(i,j,4,3);     % Mean of AP3
        xData(4,k) = Data(i,j,4,4);     % Mean of AP4
        xData(5,k) = Data(i,j,1,4);     % X-Position
        xData(6,k) = Data(i,j,2,4);     % Y-Position
        k = k+1;
    end
end
D_Input1  = [ xData(1,:); xData(2,:); xData(3,:); xData(4,:);];
D_Output1 = [ xData(5,:); xData(6,:);];
```

โปรแกรมย่อยที่ 4: Function GA_ErrorTest

```
%=================================================
%       Plot result ann Input
%=================================================
function [Norm_Average_m,Norm_Max_m,P10,P05,P02,n] = GA_ErrorTest(ref_Data,net_Data);
%--------------------------------------
L_Summ = 0;    Norm_Max_m = 0;
Under_02M=0; Under_05M=0; Under_10M=0;
%--------------------------------------
for k=1:size(ref_Data,2)
    i  = [ ref_Data(1,k) , net_Data(1,k)];  %(x1,x2)
    j  = [ ref_Data(2,k) , net_Data(2,k)];  %(y1,y2)
    ii =   ref_Data(1,k) - net_Data(1,k);   %(x1-x2)
    jj =   ref_Data(2,k) - net_Data(2,k);   %(y1-y2)

    xNorm  = sqrt(ii^2+jj^2);
    L_Summ = L_Summ + xNorm;
    %======== Over 2.0 m ========
    if xNorm < 1.0
        Under_10M = Under_10M + 1;
    end
    %======== Over 0.5 m ========
    if xNorm < 0.5
        Under_05M = Under_05M + 1;
    end
    %======== Over 0.2 m ========
    if xNorm < 0.2
        Under_02M = Under_02M + 1;
    end
    %========= Max Norm =========
    if xNorm > Norm_Max_m
        Norm_Max_m = xNorm;
        Max = [ref_Data(1,k),ref_Data(2,k)];
        i_Max = i;
        j_Max = j;
    end
end
n = size(ref_Data,2);       % Number of Data
Norm_Average_m  = L_Summ/n; % Average
P02 = Under_02M*100 / n;    % Percent of error Point
P05 = Under_05M*100 / n;    % Percent of error Point
P10 = Under_10M*100 / n;    % Percent of error Point
```

ภาคผนวก ง.

โปรแกรม MATLAB® สำหรับการทำงานของระบบระบุตำแหน่งตนเอง

โปรแกรม โปรแกรม MATLAB® สำหรับการทำงานของระบบระบุตำแหน่งตนเอง

การทำงานของ MATLAB® ประกอบด้วยโปรแกรมหลักและโปรแกรมย่อยทั้งหมด 5 โปรแกรมแบ่งหน้าที่การทำงานต่าง ๆ ดังนี้

1. โปรแกรม Create FF-MLPs เป็นโปรแกรมหลักสำหรับสร้าง ฝึกสอนและทดสอบโครงข่ายประสาทเทียม FF-MLPs
2. โปรแกรม Create RBF เป็นโปรแกรมหลักสำหรับสร้าง ฝึกสอนและทดสอบโครงข่ายประสาทเทียม RBF
3. โปรแกรม Test FF-MLPs + RBF เป็นโปรแกรมหลักสำหรับทดสอบโครงข่ายประสาทเทียมเพื่อระบุตำแหน่งด้วยการป้อนข้อมูลทดสอบผ่าน RBF และ FF-MLPs ตามลำดับ
4. โปรแกรม p2_plotTest เป็นโปรแกรมย่อยสำหรับคำนวณค่า OffError และแสดงกราฟระยะทางคลาดเคลื่อนจากผลการทำงานของ FF-MLPs
5. โปรแกรม GA_SortData เป็นโปรแกรมย่อยสำหรับจัดเรียงข้อมูลในอยู่ในรูปที่เหมาะในการนำไปใช้คำนวณของ FF-MLPs
6. โปรแกรม Pk_plotErrorBar เป็นโปรแกรมย่อยสำหรับนำค่าระยะคลาดเคลื่อนทั้งหมดแสดงในรูปกราฟของระยะคลาดเคลื่อนที่เกิดขึ้นกับจำนวนจุดของระยะคลาดเคลื่อนนั้น

โปรแกรมหลักที่ 1: Create FF-MLPs

```
%=================================================
%      Input Load
%=================================================
clear all,  clc, close all, fprintf('\n');
Train_f = 1;            % If 1=Tarin  0=Not Tarin
load('Data_T21.mat');   % nRow nCol 4Data[x,y,?,mean] 4AP
%=================================================
%       Prepare Data
%=================================================
[nGrid_Row nGrid_Col nData_Type nAP] = size(Data);
figure(1);
AP_Array = [1 2 3 4];
for APt=1:4
    iAP = AP_Array(APt);    subplot(2,2,iAP);
    hold off;   contourf(Data(:,:,4,iAP));  % Show mean
    switch iAP
        case 1, title('F3-AP1 BackRoom');
        case 2, title('F3-AP2 Front-L');
        case 3, title('F3-AP3LS Front-R');
        case 4, title('F3-AP4LS Center');
    end
    colorbar;
    grid on;
end

%=================================================
%      Sort Data
%=================================================
[nGrid_Row nGrid_Col nData_Type nAP] = size(Data);
[D_Input ,D_Output] = Pk_SortData(Data);


%=================================================
%      Created RBF and Train
%=================================================
if Train_f == 1
    %fprintf(' #Create_FF and Train\n');
    %X_Input = [0,1; 0,1; 0,1; 0,1;];
    load('DatFF_Net');
    %net = newff(X_Input,[4 4 2],{'tansig' 'tansig' 'purelin'});
    net.trainParam.epochs = 500;
    net.trainParam.show = 25;
    net.trainParam.goal = 0.1;
    net = train(net,D_Input,D_Output);
else
    fprintf(' #Test_ANN\n');
    load('DatFF_Net_best');
end
net_Data = sim(net,D_Input);
Target_Data = D_Output;
[Norm_Average_m,Norm_Max_m,P10,P05,P02,n] = Pk_plotTest(Target_Data,net_Data);

%======== Clear all Variable ========================
save DatFF_Net net;
Pk_plotErrorBar(Target_Data,net_Data,Norm_Max_m+1);
clear all;
load('DatFF_Net');
```

โปรแกรมหลักที่ 2: Create RBF

```
%=================================================
%      Input Load
%=================================================
clear all,  clc, close all, fprintf('\n');
load('Data_T21.mat');  % nRow nCol 4Data[x,y,?,mean] 4AP
Data_Ref = Data;
load('Data_T22.mat');  % nRow nCol 4Data[x,y,?,mean] 4AP
Data_Test = Data;
[nGrid_Row nGrid_Col nData_Type nAP] = size(Data_Ref);
Step_RBF_Train = 8;     % 1=1 Meter

%=================================================
%      Sort Data
%=================================================
fprintf(' #Sort Data_Input\n'); k=1;
for i=1:Step_RBF_Train*4:nGrid_Row            % Index i = ON Y-axis 11 Data
    for j = 1:Step_RBF_Train*4:nGrid_Col      % Index j = ON X-axis 10 Data
        xData(1,k) = Data_Test(i,j,4,1);    % Mean of AP1
        xData(2,k) = Data_Test(i,j,4,2);    % Mean of AP3x
        xData(3,k) = Data_Test(i,j,4,3);    % Mean of AP3
        xData(4,k) = Data_Test(i,j,4,4);    % Mean of AP4
        xData(5,k) = Data_Ref(i,j,4,1);     % Mean of AP1
        xData(6,k) = Data_Ref(i,j,4,2);     % Mean of AP3x
        xData(7,k) = Data_Ref(i,j,4,3);     % Mean of AP3
        xData(8,k) = Data_Ref(i,j,4,4);     % Mean of AP4

        xData(10,k) = Data_Ref(i,j,1,1);    % X-Position
        xData(11,k) = Data_Ref(i,j,2,1);    % Y-Position
        k = k+1;
    end
end

%=================================================
%      Created RBF and Train
%=================================================
D_Input  = [ xData(1,:); xData(2,:); xData(3,:); xData(4,:); ];
D_Output = [ xData(5,:); xData(6,:); xData(7,:); xData(8,:); ];

fprintf(' #Create_RBF and Train\n');
net = newrb(D_Input,D_Output,0,1,20);
clc, save DatRBF_Net net;
fprintf(' # Transfer ANN Complete \n');


%=================================================
close (figure(1),figure(2))
figure(1);
hold off;   plot(xData(10,:),xData(11,:),'*k');
hold on;    plot(xData(10,:),xData(11,:),'.b');
axis([-5 25 -5 30]);
grid on;
title(' RBF Reference Point');
Xlabel('Range(m)');   ylabel('Range(m)');
clear all
```

โปรแกรมหลักที่ 3: Test FF-MLPs + RBF

```
%=====================================================
%      Input Load
%=====================================================
clear all,  clc, close all, fprintf('\n');
Train_f = 0;                    % If 1=Tarin  0=Not Tarin
load('Data_T22.mat'); % nRow nCol 4Data[x,y,?,mean] 4AP
[nGrid_Row nGrid_Col nData_Type nAP] = size(Data);

%=====================================================
%      Prepare Data
%=====================================================
figure(1);
AP_Array = [1 2 3 4];
for APt=1:4
    iAP = AP_Array(APt);    subplot(2,2,iAP);
    hold off;   contourf(Data(:,:,4,iAP));  % Show mean
    %hold on;    plot(Data(:,:,2,iAP),Data(:,:,1,iAP),'.k');
    switch iAP
        case 1, title('F3-AP1 BackRoom');
        case 2, title('F3-AP2 Front-L');
        case 3, title('F3-AP3LS Front-R');
        case 4, title('F3-AP4LS Center');
    end
    colorbar;
    grid on;
end

%=====================================================
%      Sort Data
%=====================================================
load('DatRBF_Net');               % Load Transfer Data ANN_Net
fprintf(' #Sort Data_Input\n'); k=1;
for i=1:4:nGrid_Row                  % Index i = ON Y-axis 11 Data
    for j = 1:4:nGrid_Col              % Index j = ON X-axis 10 Data
        W = Data(i,j,4,1);     % Mean of AP1
        X = Data(i,j,4,2);     % Mean of AP3x
        Y = Data(i,j,4,3);     % Mean of AP3
        Z = Data(i,j,4,4);     % Mean of AP4
        transferData = sim(net,[W; X; Y; Z;]);
        xData(1,k) = transferData(1,1);
        xData(2,k) = transferData(2,1);
        xData(3,k) = transferData(3,1);
        xData(4,k) = transferData(4,1);
        xData(5,k) = Data(i,j,1,4);     % X-Position
        xData(6,k) = Data(i,j,2,4);     % Y-Position
        k = k+1;
    end
end

%=====================================================
%      Created RBF and Train
%=====================================================
D_Input  = [ xData(1,:); xData(2,:); xData(3,:); xData(4,:); ];
D_Output = [ xData(5,:); xData(6,:);];

fprintf(' #Test_ANN\n');
load('DatFF_Net98');
net_Data = sim(net,D_Input);
Target_Data = D_Output;
[Norm_Average_m,Norm_Max_m,P10,P05,P02,n] = Pk_plotTest(Target_Data,net_Data);
Pk_plotErrorBar(Target_Data,net_Data,Norm_Max_m+1);
```

โปรแกรมย่อยที่ 1: Function p2_plotTest

```
%=================================================
%       Plot result ann Input
%=================================================
function [Norm_Average_m,Norm_Max_m,P10,P05,P02,n] ...
    = p2_plotTest(ref_Data,net_Data);

figure(9);

hold off;   plot(ref_Data(1,:),ref_Data(2,:),'ob');
hold on;    plot(ref_Data(1,:),ref_Data(2,:),'*b');
grid on;    plot(net_Data(1,:),net_Data(2,:),'.r');

axis([-5 25 -5 30]);

%-------------------------------------
L_Summ = 0;        Norm_Max_m = 0;
Under_02M=0;    Under_05M=0;        Under_10M=0;
%-------------------------------------
for k=1:size(ref_Data,2)
    i  = [ ref_Data(1,k) , net_Data(1,k)];  %(x1,x2)
    j  = [ ref_Data(2,k) , net_Data(2,k)];  %(y1,y2)
    ii =   ref_Data(1,k) - net_Data(1,k);   %(x1-x2)
    jj =   ref_Data(2,k) - net_Data(2,k);   %(y1-y2)

    xNorm  = sqrt(ii^2+jj^2);
    L_Summ = L_Summ + xNorm;
    %======== Over 2.0 m ========
    if xNorm < 1.0
        plot(i,j,'b');
        Under_10M = Under_10M + 1;
    else
        plot(i,j,'m');
    end
    %======== Over 0.5 m ========
    if xNorm < 0.5
        Under_05M = Under_05M + 1;
    end
    %======== Over 0.2 m ========
    if xNorm < 0.2
        Under_02M = Under_02M + 1;
    end
    %========= Max Norm =========
    if xNorm > Norm_Max_m
        Norm_Max_m = xNorm;
        Max = [ref_Data(1,k),ref_Data(2,k)];
        i_Max = i;
        j_Max = j;
    end
end

title(' Tested ANN Model ');
Xlabel('Range(m)');    ylabel('Range(m)');
plot(Max(1),Max(2),'pk');   % Start at Max
plot(i_Max,j_Max,'k');      % Line at Max
n = size(ref_Data,2);       % Number of Data
Norm_Average_m  = L_Summ/n; % Average
P02 = Under_02M*100 / n;      % Percent of error Point
P05 = Under_05M*100 / n;      % Percent of error Point
P10 = Under_10M*100 / n;      % Percent of error Point

fprintf('\n=============== Result ================\n');
fprintf('Deviation under 1.0 m. = %3d/%3d(%6.2f%%)\n',Under_10M,n,P10);
fprintf('Deviation under 0.5 m. = %3d/%3d(%6.2f%%)\n',Under_05M,n,P05);
fprintf('Deviation under 0.2 m. = %3d/%3d(%6.2f%%)\n',Under_02M,n,P02);
fprintf('Norm Average = %5.4f, (%e) Unit\n',Norm_Average_m,Norm_Average_m);
fprintf('Norm Max = %5.4f, (%e)  Unit
@(%3d,%3d)\n',Norm_Max_m,Norm_Max_m,Max(1),Max(2));
fprintf('=======================================\n');
```

โปรแกรมย่อยที่ 2: Function GA_SortData

```
%=================================================
%      Sort Data
%=================================================
function [D_Input1 ,D_Output1] = GA_SortData(Data);
[nGrid_Row nGrid_Col nData_Type nAP] = size(Data);

k=1;
for i=1:4:nGrid_Row            % Index i = ON Y-axis 11 Data
    for j = 1:4:nGrid_Col      % Index j = ON X-axis 10 Data
        xData(1,k) = Data(i,j,4,1);     % Mean of AP1
        xData(2,k) = Data(i,j,4,2);     % Mean of AP3x
        xData(3,k) = Data(i,j,4,3);     % Mean of AP3
        xData(4,k) = Data(i,j,4,4);     % Mean of AP4
        xData(5,k) = Data(i,j,1,4);     % X-Position
        xData(6,k) = Data(i,j,2,4);     % Y-Position
        k = k+1;
    end
end
D_Input1  = [ xData(1,:); xData(2,:); xData(3,:); xData(4,:);];
D_Output1 = [ xData(5,:); xData(6,:);];
```

โปรแกรมย่อยที่ 3: Function Pk_plotErrorBar

```
%=================================================
%       Plot Error Distance Bar Graph
%=================================================
function Pk_plotErrorBar(ref_Data,net_Data,Max_Distance);

Resolution = 100;    % Data Resolution = 1/10 m
%Max_Distance = 20;  % Max Distance 5 m
figure(10);
%-------------------------------------
Result = 0:1/Resolution:Max_Distance;
Result = [Result; Result*0;];
size(Result)

for k=1:size(ref_Data,2)
    ii =   ref_Data(1,k) - net_Data(1,k);   %(x1-x2)
    jj =   ref_Data(2,k) - net_Data(2,k);   %(y1-y2)
    xNorm = sqrt(ii^2+jj^2);
    xNorm = round(xNorm*Resolution)+1;
    Result(2,xNorm) = Result(2,xNorm)+1;
end
%-------------------------------------
figure(10);
plot(Result(1,:),Result(2,:));
axis([0 20 0 20]);
title(' Range error distribution');
xlabel('Range error(m)');    ylabel('Number of error');
grid on;
```

ภาคผนวก จ.

บทความที่ได้รับการตีพิมพ์เผยแพร่

รายชื่อบทความที่ได้รับการตีพิมพ์เผยแพร่ในขณะศึกษา

วิชัย ศรีสุรักษ์ และ อาทิตย์ ศรีแก้ว ระบบระบุตำแหน่งตนเองโดยใช้ความแรงของคลื่นด้วยเทคนิควิธีเชิงปัญญาประดิษฐ์ **TRS Conference on Robotics and Industrial Technology 2007 (CRIT2007)** ครั้งที่ 4 ระหว่างวันที่ 14 – 15 มิถุนายน 2550

.

ประวัติผู้เขียน

นายวิชัย ศรีสุรักษ์ เกิดเมื่อวันที่ 21 มีนาคม พ.ศ. 2517 ที่ตำบลรังแร้ง อำเภออุทุมพรพิสัย จังหวัดศรีสะเกษ ปัจจุบันอาศัยอยู่ที่ บ้านเลขที่ 123 หมู่ที่ 12 ตำบลธงชัยเหนือ อำเภอปักธงชัย จังหวัดนครราชสีมา สำเร็จการศึกษาระดับมัธยมศึกษาตอนปลาย จากโรงเรียนสตรีศิริเกศ อำเภอเมือง จังหวัดศรีสะเกษ สำเร็จการศึกษาระดับอนุปริญญาวิทยาศาสตร์ (สาขาเคมีปฏิบัติ) จากมหาวิทยาลัยราชภัฏสุรินทร์ จังหวัดสุรินทร์ เมื่อ พ.ศ. 2536 และสำเร็จการศึกษาระดับปริญญาตรีวิศวกรรมศาสตรบัณฑิต (วิศวกรรมโทรคมนาคม) จากมหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา เมื่อ พ.ศ. 2541 จากนั้นศึกษาต่อในระดับปริญญาโท สาขาวิชาวิศวกรรมไฟฟ้า โดยได้รับทุนพัฒนาอาจารย์ (UDC) จากทบวงมหาวิทยาลัย ขณะศึกษาได้เป็นผู้ช่วยสอนปฏิบัติการของสาขาวิชาวิศวกรรมคอมพิวเตอร์ สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี จำนวน 2 รายวิชา ได้แก่ วิชาปฏิบัติการระบบไมโครโพรเซสเซอร์ (Microprocessor System Lab) และวิชาปฏิบัติการระบบดิจิตอล (Digital System Lab) ทั้งนี้มีความสนใจในด้านระบบไมโครโพรเซสเซอร์ วงจรดิจิตอลและการประยุกต์ใช้เป็นควบคุมระบบอัตโนมัติ มีผลงานวิจัยตีพิมพ์เผยแพร่ในขณะศึกษา ดังรายชื่อที่ปรากฏในภาคผนวก จ.