การวัดความคลาดเคลื่อนของพื้นที่ป่าสำหรับภาพถ่ายดาวเทียม

Forest Land Discrepancy Measurement for Satellite Images

นายอธิศักดิ์ กล่ำเหม็ง รหัส 51365054

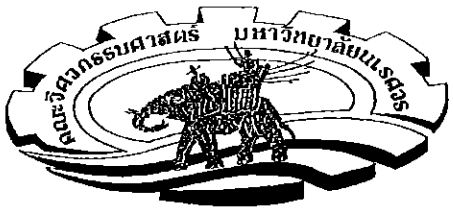
นายจิตรกร เกตุยา รหัส 51371215

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2556



ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อโครงงาน การวัดความคลาดเคลื่อนพื้นที่ป่าสำหรับภาพถ่ายดาวเทียม

ผู้ดำเนินโครงงาน	นายอธิศักดิ์	กล่ำเหม็ง	รหัส 51365054
	นายจิตรกร	เกตุยา	รหัส 51371215
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.พนมขวัญ ริยะมงคล		
อาจารย์ที่ปรึกษาร่วม	อาจารย์รัฐภูมิ วรานุสาสน์		
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2556		

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์

.....ที่ปรึกษาโครงงาน
(ผู้ช่วยศาสตราจารย์ ดร.พนมขวัญ ริยะมงคล)

.....ที่ปรึกษาร่วมโครงงาน
(อาจารย์รัฐภูมิ วรานุสาสน์)

.....กรรมการ
(ดร.สุรเดช จิตประไพกุลศาล)

.....กรรมการ
(อาจารย์สิรภพ คชรัตน์)

ชื่อหัวข้อโครงงาน	การวัดความคลาดเคลื่อนของพื้นที่ป่าสำหรับภาพถ่ายดาวเทียม
ผู้ดำเนินโครงงาน	นายอธิศักดิ์ กล่ำเหม็ง รหัส 51365054
	นายจิตรกร เกตุยา รหัส 51371215
ที่ปรึกษาโครงงาน	ผู้ช่วยศาสตราจารย์ ดร.พนมขวัญ ริยะมงคล
ที่ปรึกษาร่วมโครงงาน	อาจารย์รัฐภูมิ วรานุสาสน์
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2556

บทคัดย่อ

โครงงานนี้เป็นการวัดความคลาดเคลื่อนของข้อมูลพื้นที่ป่าจากภาพถ่ายดาวเทียมเปรียบเทียบกับพื้นที่ป่าจากการสำรวจพื้นที่จริง ซึ่งผู้วิจัยได้การออกแบบโปรแกรมที่สามารถกำหนดพื้นที่ป่าจากภาพถ่ายดาวเทียมเปรียบเทียบกับพื้นที่ป่าจากการสำรวจพื้นที่จริง เพื่อหาค่าความคลาดเคลื่อนระหว่างข้อมูลพื้นที่ป่าจากภาพถ่ายดาวเทียมกับข้อมูลพื้นที่ป่าจากการสำรวจพื้นที่จริง โดยใช้หลักการของเฮาซดอรฟดิสแทนซ์ในการหาความคลาดเคลื่อนในแนวแกน x และแกน y และฟังก์ชันตรีโกณมิติในการหาขนาดและทิศทาง โปรแกรมสามารถบันทึกข้อมูลในรูปแบบของพิกัด และค่าความคลาดเคลื่อนของพื้นที่ป่าเพื่อนำไปใช้งานต่อไปได้

Project Title	Forest Land Discrepancy Measurement for Satellite Image		
Name	Mr.Atisak	Klammeng	ID. 51365054
	Mr.Jittakorn	Ketya	ID. 51371215
Project Advisor	Assistant Professor Dr. Panomkhawn Riyamongkol		
Project Assistant Advisor	Rattapoom Waranusast		
Major	Computer Engineering.		
Department	Electrical and Computer Engineering.		
Academic Year	2556		

ABSTRACT

This project is measurement deviation of the data from the satellite images compared with the forests that is the exploration of a real space. The researchers have designed a program that can be compared with satellite images of forests and forest exploration of real space. For measurement deviation of the data from the satellite images compared with the forests that is the exploration of a real space use Hausdorff Distance algorithm finding errors in x axes and y axes and trigonometric functions finding scale and direction. This program can save the data in the form of coordinates. And the deviation of the forest area to be used further.

กิตติกรรมประกาศ

ขอขอบพระคุณผู้ช่วยศาสตราจารย์ ดร.พนมขวัญ ริยะมงคล อาจารย์ที่ปรึกษาโครงงาน และอาจารย์รัฐภูมิ วรานุสาสน์ อาจารย์ที่ปรึกษาร่วมโครงงาน ซึ่งให้ความรู้ดูแลเอาใจใส่ตลอดจนให้คำปรึกษาและช่วยเหลือในการดำเนินงานเป็นอย่างดี

ขอขอบพระคุณอาจารย์อาจารย์จิราพร พุกสุก ดร.สุรเดช จิตประไพกุลศาล และอาจารย์สิรภพ คชรัตน์ คณะกรรมการสอบโครงงานที่ให้คำแนะนำในการปรับปรุงโครงงานให้มีความถูกต้อง

ขอขอบพระคุณคณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร ที่ได้ให้เงินทุนสำหรับสนับสนุนบางส่วนในการทำโครงงาน

สุดท้ายนี้ขอกราบขอบพระคุณ คุณพ่อและคุณแม่ และเพื่อนๆที่เป็นที่ปรึกษาและเป็นกำลังใจในการดำเนินโครงการนี้ตลอดมาจนได้รับความสำเร็จ

ผู้ดำเนินโครงการ
นายอธิศักดิ์ กล่ำเหม็ง
นายจิตรกร เกตุยา

มกราคม 2557

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

สารบัญรูป(ต่อ)

สารบัญรูป(ต่อ)

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

แผนที่เป็นตัวแทนของข้อมูลเชิงพื้นที่ที่ถูกเก็บไว้ในระบบระบบสารสนเทศภูมิศาสตร์โดยกระบวนการของระบบสารสนเทศภูมิศาสตร์ที่มีการใช้งานแผนที่เป็นข้อมูลอ้างถึงตำแหน่ง อ้างอิงมาตราส่วน ซึ่งแสดงในรูปของแผนที่ ความยุ่งยากและซับซ้อนทางด้านแผนที่มีผลต่อความคลาดเคลื่อนของข้อมูลที่เกิดขึ้นเสมอ ความคลาดเคลื่อนของข้อมูลสามารถเกิดขึ้นได้ตั้งแต่กระบวนการได้มาของข้อมูลแผนที่ การนำแผนที่มาประมวลผลวิเคราะห์ และการนำเสนอข้อมูลเชิงพื้นที่ย่อมมีโอกาสที่จะเกิดความคลาดเคลื่อนเสมอ [2]

ในปัจจุบันระบบการถ่ายภาพจากดาวเทียมถูกประยุกต์ใช้ในงานด้านภูมิศาสตร์อย่างมากโดยการนำข้อมูลทางด้านภูมิศาสตร์มาช่วยในการวิเคราะห์ การแบ่งขอบเขตการใช้งานของพื้นที่ การอนุรักษ์พื้นที่ป่า หรือเก็บข้อมูลทางภูมิศาสตร์ไว้เพื่อเปรียบเทียบกับข้อมูลโดยการสำรวจพื้นที่จริงโดยใช้ระบบจีพีเอส (GPS) ในการเก็บข้อมูลตามแนวป่า แนวเขา เพื่อกำหนดขอบเขตของพื้นที่ป่า แต่ด้วยปัญหาการเดินทางที่ยากต่อการเข้าถึงพื้นที่ หรือความผิดพลาดของอุปกรณ์ในการบันทึกข้อมูล ทำให้ข้อมูลนั้นไม่ถูกต้องตามขอบเขตของพื้นที่ป่าตามที่ระบุในพิกัด เมื่อตรวจสอบจากภาพถ่ายดาวเทียมในบางพื้นที่ป่ามีการคลาดเคลื่อนไปจากแผนที่จริง หรือขอบเขตไม่ชัดเจนทำให้ไม่สามารถระบุพื้นที่จริงของป่าได้อย่างชัดเจน [3]

จากปัญหาที่เกิดขึ้นทำให้เกิดแนวคิดในการสร้างจริง ผู้วิจัยได้การออกแบบโปรแกรมที่สามารถกำหนดพื้นที่ป่าจากภาพถ่ายดาวเทียมเปรียบเทียบกับพื้นที่ป่าจากการสำรวจพื้นที่จริง เพื่อหาค่าความคลาดเคลื่อนระหว่างข้อมูลพื้นที่ป่าจากภาพถ่ายดาวเทียมกับข้อมูลพื้นที่ป่าจากการสำรวจพื้นที่จริงในแกน x แกน y ระยะกระจัด และทิศทาง โดยสามารถบันทึกข้อมูลทั้งหมดเพื่อนำไปใช้งานต่อไปได้

1.2 วัตถุประสงค์

สร้างโปรแกรมที่สามารถกำหนดพื้นที่ป่าจากภาพถ่ายดาวเทียมและ คำนวณค่าความคลาดเคลื่อนพื้นที่ป่าจากภาพถ่ายดาวเทียมกับพื้นที่ป่าจากการสำรวจพื้นที่จริง

1.3 ขอบข่ายของโครงงาน

1. สามารถกำหนดพื้นที่จากภาพถ่ายดาวเทียมได้

2. สามารถบอกความคลาดเคลื่อนของข้อมูลพื้นที่ป่าจากภาพถ่ายดาวเทียมกับข้อมูลพื้นที่ป่าจากการสำรวจพื้นที่จริงในแกน x แกน y ระยะกระจัด และทิศทางได้

3. ซ้อนทับภาพถ่ายจากดาวเทียมกับข้อมูลที่ระบุพิกัดได้

4. สามารถย่อ ขยาย ภาพถ่ายจากดาวเทียมได้

5. บันทึกข้อมูลพิกัดของพื้นที่ป่า และความคลาดเคลื่อนของข้อมูลพื้นที่ป่าได้

1.4 ขั้นตอนการดำเนินงาน

1. ศึกษาระบบสารสนเทศภูมิศาสตร์ อัลกอริทึมที่ใช้ และการเขียนโปรแกรม
2. เขียนโปรแกรมกำหนดพื้นที่ป่าจากภาพถ่ายดาวเทียม
3. ออกแบบโปรแกรมเพื่อคำนวณความคลาดเคลื่อน
4. จัดทำสรุปและข้อผิดพลาด และแก้ไขในส่วนของข้อผิดพลาดที่เกิดขึ้น

1.5 แผนการดำเนินงาน

ตารางที่ 1.1 แสดงแผนการดำเนินงาน

กิจกรรม	เวลา									
	พ.ศ.2556									
	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.
ศึกษาระบบสารสนเทศภูมิศาสตร์ และการเขียนโปรแกรม	←	—	—	→						

ตารางที่ 1.1 (ต่อ) แสดงแผนการดำเนินงาน

กิจกรรม	เวลา									
	พ.ศ.2556									
	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.
เขียนโปรแกรมกำหนดพื้นที่ป่าจากภาพถ่ายดาวเทียม					←	—	→			
เขียนโปรแกรมเพื่อคำนวณความคลาดเคลื่อน						←	—	→		
จัดทำสรุปและข้อผิดพลาดและแก้ไขในส่วนของข้อผิดพลาดที่เกิดขึ้น							←	—	→	

1.6 ผลที่คาดว่าจะได้รับ

1. สามารถนำข้อมูลความคลาดเคลื่อน ระหว่างพื้นที่ป่าจากภาพถ่ายดาวเทียมกับพื้นที่ป่าจากการสำรวจพื้นที่จริง ไปใช้งานต่อไปได้

2. ช่วยลดการทำงานผู้สำรวจพื้นที่จริง

1.7 งบประมาณที่ใช้

ค่ากระดาษ	200	บาท
ค่าจัดทำรูปเล่ม	800	บาท
ค่าหนังสือและเอกสารอื่น	1000	บาท
รวม	2000	บาท
	(สองพันบาทถ้วน)	

บทที่ 2
หลักการและทฤษฎีที่เกี่ยวข้อง

ในการจัดทำโครงงานนี้จำเป็นอย่างยิ่งที่ต้องอาศัยความรู้ในเรื่องทฤษฎีต่างๆที่เกี่ยวข้องกับข้อมูลพื้นฐานทางภูมิศาสตร์ หลักการคำนวณระยะทาง และการเขียนโปรแกรมเพื่อใช้ในการพัฒนาโครงงานให้มีประสิทธิภาพในการทำงาน และมีการทำงานที่ถูกต้องเพื่อให้เหมาะสมกับผู้ใช้งานซึ่งทฤษฎีที่เกี่ยวข้องนั้นสามารถอธิบายได้ดังนี้

1. ระบบสารสนเทศภูมิศาสตร์
2. ระบบจีพีเอส (GPS)
3. ระบบอาร์เอส (RS)
4. ระบบจีไอเอส (GIS)
5. ข้อมูล
6. หลักการคำนวณ

2.1 ระบบสารสนเทศภูมิศาสตร์ [1]

จากระบบนิเวศสู่มาตราส่วนในแผนที่การศึกษาสิ่งแวดล้อมส่วนใหญ่ขึ้นอยู่กับข้อมูลซึ่งถูกรวบรวมในระดับมาตราส่วนเชิงพื้นที่ (Spatial Scales) ที่ละเอียด เช่น ระดับแปลง นอกจากนี้มาตราส่วนเชิงเวลา (Temporal Scales) ของการศึกษาเหล่านี้ค่อนข้างสั้น เช่น ระดับวัน ระดับอาทิตย์ ระดับเดือน การศึกษามีจำนวนน้อยมาก อย่างไรก็ตามการศึกษาระดับมาตราส่วนในรายละเอียดระดับแปลง และมาตราส่วนเวลาที่ค่อนข้างสั้นเหล่านี้เริ่มล้าสมัย เนื่องจากปัจจุบันนักวิทยาศาสตร์สิ่งแวดล้อมกำลังค้นพบการประมาณการที่เพิ่มขึ้นจากปกติจากการศึกษาระดับแปลง (Plot level) ไปยังขนาดพื้นที่ที่กว้างใหญ่ขึ้น เช่น ระดับภูมิภาค และจากการศึกษาระยะสั้น (Short-Term) ไปเป็นขนาดเชิงเวลาที่ยาวนานขึ้น เพื่อค้นหาคำตอบที่เกี่ยวข้องกับกระบวนการที่ใช้เวลาระยะยาว (Long-term Processes) เช่น การเพิ่มขึ้น ของอุณหภูมิโลก และการเพิ่มขึ้นของระดับน้ำทะเล เป็นต้น การวิจัยหรือการเสาะแสวงหาความรู้ในยุคปัจจุบันจึงให้โอกาสกับนักวิทยาศาสตร์ในการออกแบบและดำเนินการค้นคว้าวิจัยที่เน้นคำถามระดับมหภาคหรือคำถามที่สามารถตอบผลในระยะยาวได้คำถามที่ซับซ้อนของความต้องการระดับนานาชาติเป็นสิ่งที่นักวิทยาศาสตร์จำเป็นต้องอาศัยข้อดีของเทคโนโลยีสมัยใหม่ๆ

จากการพัฒนาเทคโนโลยีสื่อสารระยะไกลและคอมพิวเตอร์และการนำเทคโนโลยีนี้มาใช้กับข้อมูลเชิงพื้นที่ทำให้เกิดเทคโนโลยีแขนงหนึ่งที่เรียกว่าเทคโนโลยีภูมิสารสนเทศ (Geo-Informatics

Technology) โดยเป็นเทคโนโลยีเกี่ยวข้องกับการได้มาซึ่งข้อมูลรวมทั้งการวิเคราะห์และประมวลผลข้อมูลเพื่อให้ได้ซึ่งสารสนเทศต่างๆที่เกี่ยวกับภูมิลักษณ์ทางภูมิศาสตร์ (Geographical Feature) เทคโนโลยีภูมิสารสนเทศประกอบด้วยเทคโนโลยี 3S คือ ระบบจีพีเอส GPS (Global Positioning System) , ระบบอาร์เอส RS (Remote Sensing) , ระบบจีไอเอส GIS (Geographic Information System)

2.2 ระบบจีพีเอส GPS [1]

ระบบจีพีเอส GPS (Global Positioning System) หรือระบบกำหนดตำแหน่งบนโลก หมายถึง ระบบเป็นการสำรวจและบันทึกข้อมูลโดยระบบนำทาง (Navigation) ด้วยดาวเทียมจำนวน 24 ดวง โคจรในระดับสูงที่พ้นจากคลื่นวิทยุรบกวนมีวัตถุประสงค์เพื่อการทหารสำหรับกระทรวงกลาโหมของประเทศสหรัฐอเมริกา ต่อมาจึงขยายการใช้เพื่อกิจการพลเรือนการโคจรของดาวเทียมจำเป็นต้องมีแนววงโคจรที่แม่นยำสูงเพื่อสามารถส่งสัญญาณกลับมายังโลกเพื่อบอกตำแหน่งบนโลกอย่างแม่นยำและรวดเร็วตลอด 24 ชั่วโมงความแม่นยำทางตำแหน่งของจีพีเอส (GPS) ในปัจจุบันอยู่ในช่วงตั้งแต่ 0.5 มิลลิมตร ถึง 100 เมตร หลักการทำงานของระบบกำหนดตำแหน่งบนพื้นโลกคืออาศัยการสื่อสารระหว่างเครื่องกำหนดพิกัดตำแหน่ง (อยู่ระดับพื้นดิน) กับดาวเทียม (โคจรอยู่ในอวกาศ) เมื่อเสาอากาศ (Antenna) ของเครื่องสามารถค้นหาตำแหน่งดาวเทียมได้อย่างน้อย 3 ดวง ก็สามารถแสดงผลตำแหน่งภาคพื้นดินแนวราบ 2 มิติแสดงที่อยู่ของเครื่องได้ โดยตำแหน่งนี้สามารถเปลี่ยนแปลงไปตามการเคลื่อนที่

2.3 ระบบอาร์เอส RS [1]

ระบบอาร์เอส RS (Remote Sensing) หรือ ระบบการสำรวจระยะไกลหมายถึง ระบบการสำรวจที่มีบันทึกข้อมูลผ่านระบบดาวเทียม ซึ่งระบบการสำรวจระยะไกลที่มีอยู่ปัจจุบันนั้น ก็มีข้อจำกัดในการวัดช่วงคลื่นแตกต่างกันออกไปโดยสามารถแบ่งออกได้ 2 ระบบคือ

2.3.1 ระบบถ่ายภาพ (Photographic system)

ระบบนี้จะประกอบไปด้วย กล้อง เลนส์ ชัตเตอร์ และฟิล์มที่ไวต่อแสง ภาพจะถูกบันทึกเมื่อคลื่นรังสีแม่เหล็กมักจะเป็นคลื่นแสงที่สามารถมองเห็นได้ผ่านเข้าไปในเลนส์ไปสัมผัสกับพื้นผิวของฟิล์มทำให้เกิดภาพขึ้น ระบบการบันทึกภาพแบบนี้จะแตกต่างกับระบบการบันทึกภาพชนิดอื่นๆ ซึ่งใช้ส่วนอื่นของคลื่นแม่เหล็กไฟฟ้า (ที่ตามนุษย์ไม่สามารถมองเห็นได้) ระบบภาพถ่ายใช้มากในยุคเริ่มต้นของการพัฒนาการด้านรีโมทเซนซิ่งภาพที่ได้จากดาวเทียมเหล่านี้จะถูกบันทึกด้วยระบบถ่ายภาพทั้งหมด

2.3.2 ระบบเชิงเลข (Non-photographic system หรือ Digital)

ภาพข้อมูลที่ได้จากระบบนี้จะถูกบันทึกโดยใช้คลื่นแม่เหล็กไฟฟ้าที่เป็นต้นกำเนิดของรังสีต่างๆ คือ ที่ตามนุษย์สามารถมองเห็นรังสีอินฟราเรด รังสีอุลตราไวโอเล็ต และคลื่นวิทยุ ซึ่ง

สะท้อนหรือส่งมาจากเครื่องบันทึกข้อมูล หรือสายอากาศแล้วแปลงให้เป็นสัญญาณไฟฟ้าและระบบก็จะทำการบันทึกเอาไว้ภาพที่ได้นี้ถือว่าเป็น "Image" ระบบเลขนับได้ว่าเป็นระบบที่นิยมใช้มากในปัจจุบันนี้ เพราะมีความได้เปรียบมากกว่าระบบภาพถ่ายหลายประการ เช่น สามารถปรับแก้ความคลาดเคลื่อนทางตำแหน่งให้ถูกต้องได้ สามารถนำวิธีการทางสถิติและคณิตศาสตร์มาประยุกต์ใช้กับข้อมูลได้โดยระบบเชิงเลขนั้นประกอบด้วยข้อมูล 2 แบบคือ

2.3.2.1 ระบบ Multispectral Scanning System (MSS)

จะทำการบันทึกข้อมูลภาพโดยใช้ระบบ MSS โดยจะบันทึกภาพจำนวน 4 ภาพ ต่อ 1 Scene และแต่ละภาพจะครอบคลุมพื้นที่ 185 x 185 ตารางกิโลเมตร และมีความแยกต่างจากพื้นที่ 79 เมตร ซึ่งภาพทั้ง 4 ภาพนี้จะประกอบด้วย 4 ช่วงคลื่น คือ สีเขียว แดง และอินฟราเรดใกล้ 2 ช่วงคลื่นเป็นช่วงคลื่นดังแสดงในตารางที่ 2.1

ตารางที่ 2.1 แสดงคุณลักษณะระบบบันทึกภาพดาวเทียมในระบบ MSS

ชื่อคลื่น	ความยาวคลื่น	ระบบที่ดาวเทียมบันทึก	
		LANDSAT 1,2,3	LANDSAT 4,5
เขียว (Visible Green)	0.5-0.6 µm	Band 4	Band 1
แดง (Visible Red)	0.6-0.7 µm	Band 5	Band 2
อินฟราเรดใกล้ [Near Infrared (NI)]	0.7-0.8 µm	Band 6	Band 3
อินฟราเรดใกล้ [Near Infrared (NI)]	0.8-1.0 µm	Band 7	Band 4

เครื่องบันทึกข้อมูลระบบจะทำการบันทึกภาพโดยการสะท้อนของพลังงานความร้อนจากพื้นที่ ที่มีความกว้าง 79 เมตรภายใต้สนามมุมมอง (Instantaneous Field Of View - IFOV) ของเครื่องบันทึก แล้วทำการบันทึกข้อมูลบนพื้นผิวโลกสู่เครื่องรับสัญญาณ (Detector) ที่อยู่บนดาวเทียมในขณะที่ดาวเทียมเคลื่อนที่ไปข้างหน้านั้นจะทำการกวาดพื้นที่แต่ละครั้งจำนวน 6 เส้นต่อ 1 แบนด์ ดังนั้นจึงต้องใช้เครื่องรับสัญญาณถึง 24 ตัวสำหรับ 4 แบนด์ เครื่องรับสัญญาณแต่ละตัวก็จะเปลี่ยนค่าพลังงานความร้อนไปเป็นสัญญาณไฟฟ้าฟ้าที่ต่อเนื่อง แล้วเปลี่ยนไปเป็นค่าตัวเลขชนิด 6 bit (0 – 64 ระดับ) ซึ่งอาจจะบันทึกภาพลงบนเทปแม่เหล็กหรือส่งผ่านกลับมายังโลก ข้อมูลจะถูกเปลี่ยนเป็น 7 bit (0 – 128 ระดับ) สำหรับแบนด์ 4 , 5 และ 6

2.3.2.2 ระบบ Thematic Mapper (TM)

ระบบ TM ออกแบบให้มีความเหมาะสมยิ่งขึ้นโดยได้รับการปรับปรุงให้มีความไวเชิงคลื่นดีขึ้นในทุกแบนด์จึงสามารถบันทึกค่าสีเทาได้ 256 ระดับใน 7 ช่วงคลื่น นอกจากนี้แล้ว Optics ของกล้องเทเลสโคปและของกระจกที่ใช้กวาดจะช่วยลดการกระจายของแสงและความพร่ามัวลงได้คุณสมบัติที่ดีอีกข้อหนึ่งของระบบ TM คือ ได้รับการปรับปรุงความแตกต่างทางพื้นที่ 30 เมตรในทุกช่วงคลื่นยกเว้นช่วงคลื่นความร้อนซึ่งมีความแตกต่างทางพื้นที่ 120 เมตร การปรับปรุง

รายละเอียดของภาพให้ดีขึ้นนี้ ทำให้ผู้ใช้สามารถแยกรายละเอียดภาพพื้นดินได้ 900 ตารางเมตร (0.56 ไร่) ส่วนจำนวนช่วงคลื่นที่มีมากกว่าระบบ MSS ทำให้ผู้ใช้สามารถเลือกช่วงคลื่นที่เหมาะสมมาวิเคราะห์ได้ดียิ่งขึ้นข้อมูล TM จึงมีความสำคัญอย่างยิ่งต่อการศึกษาทรัพยากรโลกจากดาวเทียมสำรวจทรัพยากรโลก แสดงส่วนประกอบภายในของระบบ TM และคุณสมบัติของระบบ TM และการนำไปใช้ประโยชน์คุณลักษณะของระบบบันทึกภาพดาวเทียมด้วยระบบ TM สามารถแสดงได้ดังตารางที่ 2.2 ดังต่อไปนี้

ตารางที่ 2.2 แสดงคุณลักษณะระบบบันทึกภาพดาวเทียมในระบบ TM

ช่วงคลื่น	ความยาวช่วงคลื่น(μm)	การนำมาใช้ประโยชน์(Applications)
1. B/G	0.45-0.52	ผ่านทะลุน้ำ ดูความแตกต่างของพืชพันธุ์สีเขียว แสดงความแตกต่างระหว่างป่าผลัดใบกับป่าสนซึ่งไม่ผลัดใบ
2. G	0.52-0.60	แยกประเภทพืชพันธุ์สีเขียว เพราะมี การสะท้อนสูง มีประโยชน์ในการประเมินความแข็งแรงของพืช
3. R	0.63-0.69	มีการดูดกลืนคลอโรฟิลด์สูงมาก ใช้แยกชนิดของพืช
4. NI	0.76-0.90	แยกความแตกต่างระหว่างพื้นดินและพื้นน้ำมีการสะท้อนจากพืชสูงมาก
5. NI	1.55-1.75	รายละเอียดเกี่ยวกับปริมาณความชื้นของดิน ความแตกต่างระหว่าง หิมะกับเมฆ
6. ThermalInfrared	10.4-12.5	วิเคราะห์ความเครียดของพืชพรรณ (Plant stress) เช่น การขาดน้ำ ถูกแมลงทำลาย การหาแหล่งความร้อน การจำแนกความชื้นในดิน
7. MiddleInfrared	2.08-2.35	จำแนกชนิดของหิน การทำแผนที่แสดง แหล่งความร้อนใต้พื้นพิภพ

2.4 ระบบ GIS [1]

ระบบจีไอเอส GIS (Geographic Information System) หรือระบบสารสนเทศภูมิศาสตร์ หมายถึง ระบบที่มีกระบวนการทำงานเกี่ยวกับข้อมูลเชิงพื้นที่ (Spatial Data) ด้วยระบบคอมพิวเตอร์ โดยการกำหนดข้อมูลเชิงบรรยายหรือข้อมูลคุณลักษณะ (Attribute Data) และสารสนเทศ เช่น ที่อยู่บ้านเลขที่ ที่มีความสัมพันธ์กับตำแหน่งในเชิงพื้นที่ (Spatial Data) เช่น ตำแหน่งบ้าน ถนน แม่น้ำ ซึ่งแสดงในรูปของตารางข้อมูล และฐานข้อมูล เช่น แผนที่ กริดภูมิศาสตร์ ระบบพิกัดแนวระนาบ แบบจำลองข้อมูลเชิงพื้นที่

2.4.1 แผนที่

แผนที่ คือ เครื่องมือสำคัญของการบรรจุข้อมูลในระบบสารสนเทศภูมิศาสตร์ การเริ่มต้นของการผลิตแผนที่กล่าวได้ว่าเกิดก่อนเทคโนโลยีของระบบสารสนเทศภูมิศาสตร์โดยการผลิตแผนที่เริ่มจากการเดินทางสำรวจแล้วบันทึกข้อมูลภูมิลักษณ์ต่างๆที่ปรากฏพบเห็นจากการสำรวจ เช่น รูปร่างของทวีป การตั้ง ถิ่นฐาน และขอบเขตของการปรากฏพืชพรรณธรรมชาติในแผ่นทวีป เป็นต้น ในยุคเริ่มต้นของการทำแผนที่เริ่มจากการบันทึกย่อข้อมูลลงเป็นภาพเขียนในวัสดุธรรมชาติ

ในปัจจุบันพัฒนาการการทำแผนที่ได้บันทึกข้อมูลอยู่ในคอมพิวเตอร์ในรูปข้อมูลแผนที่เชิงเลข (Digital Map Data) ความหมายของแผนที่ได้มีผู้ให้คำจำกัดความแตกต่างกันอยู่บ้าง ซึ่งพอสรุปได้ว่าแผนที่ หมายถึง การแสดงข้อมูลภูมิลักษณ์ส่วนใดส่วนหนึ่งของผิวโลกที่ย่อส่วนลงมา จากความหมายที่กล่าวมานี้สามารถขยายความได้ดังนี้

1) **การแสดงข้อมูล** หมายถึง การจำลองแบบของจริงให้อยู่ในรูปลายเส้นรูปเชิงกริดหรือรูปทรวดทรงโดยการแสดงนี้อยู่ในรูปภาพหรือรูปทรง 2 มิติที่แสดงเฉพาะความกว้างและความยาว หรือรูปทรง 3 มิติที่แสดงทั้ง ความกว้าง ความยาว และความสูง

2) **ข้อมูลภูมิลักษณ์** หมายถึง สิ่งปรากฏบนโลกที่เกิดตามธรรมชาติ เช่น แม่น้ำ ความสูงต่ำ เป็นต้น หรือที่เกิดโดยมนุษย์ เช่น อาคาร ถนน และบ่อน้ำ เป็นต้น

3) **การย่อส่วน** หมายถึง การถ่ายทอดสิ่งปรากฏบนผิวโค้งโลกลงมาบนพื้นผิวราบโดยการย่อสัดส่วนด้วยการใช้มาตราส่วน (Scale) ระบบอ้างอิงตำแหน่ง ภูมิลักษณ์บนแผนที่ (Map Feature) แสดงถึงภูมิลักษณ์เชิงพื้นที่ (Spatial Feature) บนพื้นผิวโลกตำแหน่งของภูมิลักษณ์บนแผนที่ขึ้นอยู่กับระบบพิกัด (Coordinate System) ขณะที่ตำแหน่งของภูมิลักษณ์เชิงพื้นที่ขึ้นอยู่กับกริดภูมิศาสตร์ (Geographic Grid) ซึ่งอยู่ในค่าลองจิจูดและละติจูด ดังนั้น จึงเกิดการเปลี่ยนรูปจากกริดภูมิศาสตร์เป็นระบบพิกัด เรียกว่าเส้นโครงแผนที่ (Map Projection) พื้นฐานของการทำงานด้วยระบบสารสนเทศภูมิศาสตร์ คือ การใช้ชั้นข้อมูลแผนที่ (Map Layers) ร่วมกันได้จำเป็นต้องมีระบบพิกัดเดียวกัน การแปลงระบบอ้างอิงตำแหน่งของแผนที่จากค่าลองจิจูดและละติจูดไปเป็นพิกัด 2 มิติ (Two-Dimensional Coordinate) เรียกว่า โปรเจ็กชั่น (Projection) และการเปลี่ยนจากระบบพิกัดหนึ่งไปยังระบบพิกัดอื่น เรียกว่า รี-โปรเจ็กชั่น (Re-Projection) โดยทั่วไปแล้วไม่ว่าเป็นโปรเจ็กชั่นหรือรี -โปรเจ็กชั่นถือเป็นขั้นตอนเริ่มต้นในการทำงานในระบบสารสนเทศภูมิศาสตร์

2.4.2 กริดภูมิศาสตร์

กริดภูมิศาสตร์ หรือ ระบบอ้างอิงตำแหน่ง (Location Reference System) คือ สำหรับภูมิลักษณ์บนพื้นผิวโลก กริดภูมิศาสตร์ประกอบด้วยเส้นเมอริเดียน (Meridians) และเส้นพาราเรล (Parallels) เส้นเมอริเดียน คือ เส้นลองจิจูดสำหรับทิศทางตะวันออก-ตะวันตก โดยเส้นไพร์เม-อริเดียน (Prime Meridian) ซึ่งเป็นเส้นศูนย์องศาเมอริเดียนพาดผ่านเมืองกรีนนิช (Greenwich) ของประเทศอังกฤษเส้นเมอริเดียนมีค่าตั้ง แต่ 0 ถึง 180 องศาตะวันออก หรือตะวันตกของเส้นไพร์มเมอ

ริเดียน ส่วนเส้นพาราเรล คือ เส้นละติจูดสำหรับทิศทางเหนือ-ใต้ โดยมีเส้นศูนย์ สูตร (Equator) เป็นเส้นศูนย์องศาละติจูด เส้นละติจูดมีค่าตั้งแต่ 0 ถึง 90 องศาเหนือหรือใต้ของเส้นศูนย์สูตรจุดเริ่มต้นของกริดภูมิศาสตร์ หรือตำแหน่งที่มีค่าลองจิจูดและละติจูด 0 องศา คือ ตำแหน่งที่เส้นไพร์เมอริเดียนตัดกับเส้นศูนย์สูตร ค่าลองจิจูด คือ ค่า x ในระบบพิกัด ส่วนค่าละติจูด คือ ค่า y ในระบบพิกัด

2.4.3 ระบบพิกัดแบบระนาบ

ระบบพิกัดแบบระนาบ (Coordinate) คือมาตราใช้สำหรับการสร้างแผนที่ขนาดใหญ่ (Large-Scale Mapping) เช่น มาตราส่วน 1 : 24,000 หรือใหญ่กว่าความถูกต้องทางตำแหน่งสัมบูรณ์ (Absolute Position) ของภูมิลักษณ์และตำแหน่งสัมพัทธ์ (Relative Position) ของภูมิลักษณ์นั้น กับภูมิลักษณ์อื่น ๆ มีความสำคัญมากกว่าคุณสมบัติของเส้นโครงแผนที่ เพื่อรักษาระดับความถูกต้องสำหรับการวัดตำแหน่ง ดังนั้น ระบบพิกัดหนึ่งๆจึงนิยมแบ่งออกเป็นโซนต่างๆ โดยที่แต่ละโซนขึ้นอยู่กับเส้นโครงแผนที่ที่แยกออกจากกัน

ระบบพิกัดแบบยูทีเอ็ม UTM (Universal Transverse Mercator Coordinate System) นับเป็นระบบพิกัดที่นิยมใช้กันทั่วโลก โดยแบ่งพื้นที่ผิวโลกตั้งแต่ 840 N และ 800 S ออกเป็นจำนวน 60 โซนตามแนวเหนือ-ใต้ และ 20 โซนตาม แนวตะวันออก-ตก รวมจำนวนทั้งหมด 1,200 กริดโซน แต่ละโซนครอบคลุมลองจิจูด 6 องศาโดยโซนที่ 1 เริ่มต้นที่ 1800 ถึง 1740 ตะวันตก แต่ละโซนของระบบกริดยูทีเอ็ม (UTM) ถ่ายทอดลงบนเส้นโครงแผนที่รูปทรงกระบอกแบบ Transverse Mercator Projection โดยมี Scale Factor เท่ากับ 0.9996 ที่เมอริเดียนย่านกลาง (Central Meridian) ทั้งนี้ Scale factor คือ อัตราส่วนความยาวที่ผิว Spheroid ต่อความยาวที่ผิวทรงกระบอก ซึ่ง 1 Scale Factor มีค่าเท่ากับ 0.9996 เมอริเดียน (Standard Meridian) ค่าระยะพิกัดมีหน่วยเป็นเมตร สำหรับประเทศไทยตั้ง อยู่ในโซนยูทีเอ็ม (UTM) ที่ 47 และ 48

2.4.4 แบบจำลองข้อมูลเชิงพื้นที่

แบบจำลองข้อมูลที่ใช้แสดงแทนข้อมูลเชิงพื้นที่ แบ่งเป็น 2 ประเภท คือ แบบจำลองข้อมูลเวกเตอร์ และแบบจำลองข้อมูลราสเตอร์ ซึ่งสามารถเปลี่ยนแปลงรูปแบบระหว่างกันได้ การแปลงจากราสเตอร์เป็นเวกเตอร์เรียก Vectorization และการแปลงจากเวกเตอร์เป็นราสเตอร์เรียก Rasterization ประเภทของแบบจำลองมีความสำคัญในการกำหนดโครงสร้าง การจัดเก็บ การประมวลผล และการวิเคราะห์ข้อมูลในระบบสารสนเทศภูมิศาสตร์

2.4.4.1 แบบจำลองข้อมูลเวกเตอร์ (Vector Data Model) เป็นแบบจำลองข้อมูลที่ใช้รูปทรงทางเรขาคณิตแบ่งเป็น 3 ลักษณะคือ จุด (Point) เส้น (Line) และรูปหลายเหลี่ยม (Polygon) สำหรับแสดงข้อมูลเชิงพื้นที่ข้อมูลจุดเป็นข้อมูลที่มีพิกัดเพียงตำแหน่งเดียวไม่มีทิศทางและขนาดข้อมูลเส้นเป็นข้อมูลที่มีพิกัดมากกว่าหนึ่งจุด จึงมีทิศทางแต่ไม่มีขนาดส่วนข้อมูลโพลิกอนเป็นข้อมูลที่มีพิกัดมากกว่า 3 จุดขึ้นไป โดยที่จุดเริ่มต้นและจุดสุดท้ายต้องอยู่ที่ตำแหน่งเดียวกัน ข้อมูลโพลิกอนจึงเป็นพื้นที่ที่เส้นรอบรูปปิดที่มีทั้งตำแหน่ง ทิศทาง และขนาด ข้อมูลเวกเตอร์ถือเป็นวัตถุเรขาคณิตที่ไม่ต่อเนื่อง (Discrete Geometric Object) ของพื้นที่

ข้อมูลเวกเตอร์มีทั้งแบบมีโทโพโลจี (Topological) และไม่มีโทโพโลจี (Non-Topology) โทโพโลจีมีประโยชน์ต่อการตรวจหาความคลาดเคลื่อนหรือข้อผิดพลาดบนแผนที่เชิงเลข นอกจากมีประโยชน์ต่อการแก้ไขความคลาดเคลื่อนหรือข้อผิดพลาดของข้อมูลแล้วยังมีประโยชน์สำหรับการวิเคราะห์เชิงพื้นที่บางลักษณะ จึงเกิดการพัฒนาข้อมูลเวกเตอร์แบบไม่มีโทโพโลจี (Non-Topological Vector Data) ซึ่งสามารถแสดงบนหน้าจอคอมพิวเตอร์ได้เร็วกว่าข้อมูลแบบโทโพโลจี เช่นรูปแบบข้อมูลรูปร่าง (Shape File) ซึ่งไม่มีแฟ้มข้อมูลที่อธิบายความสัมพันธ์เชิงพื้นที่ระหว่างวัตถุทรงเรขาคณิตรูปทรงเรขาคณิตของรูปร่าง (Shape File) เก็บบันทึกในแฟ้มข้อมูลพื้นฐาน 2 รูปแบบ คือ แฟ้มข้อมูลนามสกุล .shp เก็บบันทึกรูปทรงเรขาคณิตของภูมิลักษณ์ และแฟ้มข้อมูลนามสกุล .shx เก็บรักษาดัชนีของรูปทรงเรขาคณิตของภูมิลักษณ์ ข้อมูลในรูปร่าง (Shape File) และ (Coverage) สามารถเปลี่ยนระหว่างกันได้แบบจำลอง

2.4.4.2 **แบบจำลองข้อมูลราสเตอร์** (Raster data model) เป็นแบบจำลองที่ใช้กริดหรือจุดภาพรูปร่างสี่เหลี่ยมที่มีขนาดเท่ากันเรียงตัวในรูปแถว (Row) ตามทิศตะวันออก-ตก และสดมภ์ (Column) ตามทิศเหนือ-ใต้ โดยเซลจุดภาพเริ่มต้นของข้อมูลอยู่ที่ตำแหน่งบนซ้ายของตำแหน่งแถวและสดมภ์ที่ (1,1) ภายในเซลของแต่ละกริดมีค่าของข้อมูลที่แสดงถึงคุณลักษณะที่แตกต่างกันของภูมิลักษณ์เชิงพื้นที่ขนาดของกริดมีผลต่อรายละเอียด (Resolution) ของแบบจำลองข้อมูลราสเตอร์กริดยิ่งมีขนาดเล็กแสดงถึงรายละเอียดที่เพิ่มขึ้นเช่น กริดขนาด 30 เมตรแทนพื้นที่ 900 ตารางเมตร ในขณะที่กริดขนาด 40 เมตรแทนเนื้อที่ 1,600 ตารางเมตรดังนั้นการที่กริดมีขนาดใหญ่ขึ้น หรือมีรายละเอียดน้อยลงนั้น ย่อมไม่สามารถแสดงตำแหน่งที่แม่นยำของภูมิลักษณ์เชิงพื้นที่อีกทั้งยังทำให้เกิดการผสมของภูมิลักษณ์หลายประเภทอยู่ภายในกริดเดียวกัน เช่นมีทั้งพื้นที่ป่าและแหล่งน้ำอยู่ในกริดเดียวกันข้อมูลราสเตอร์เหมาะสำหรับการแสดงภูมิลักษณ์เชิงพื้นที่ที่มีลักษณะข้อมูลแบบต่อเนื่องโดยเฉพาะด้านภูมิประเทศ เช่น DEM ข้อมูลราสเตอร์ที่ใช้ในระบบสารสนเทศภูมิศาสตร์มีรูปแบบต่างๆ ได้แก่ DEM ข้อมูลการสำรวจระยะไกล แผนที่เชิงเลขจากเครื่องกวาดภาพ รวมทั้งแฟ้มข้อมูลกราฟฟิก เช่น TIFF (Tag Image File Format) ข้อมูลราสเตอร์แสดงจุดภาพในรูปเซลเดี่ยว (Single Cells) หรือแสดงเส้นในรูปลำดับต่อเนื่องของเซลเพื่อนบ้าน (Sequences Of Neighboring Cells) หรือแสดงพื้นที่ในรูปกลุ่มของเซลที่ติดต่อกัน (Contiguous Cells) โดยที่ค่าของเซลอาจเป็นจำนวนเต็ม (Integer) หรือค่าทศนิยม (Floating) ในกรณีค่าจำนวนเต็มมักใช้กับข้อมูลนามกำหนดหรืออันดับ (Categorical Data) เช่น พื้นที่ในจังหวัดพิษณุโลกถ้าเป็นเมืองให้เป็น 1 ป่าให้เป็น 2 และแหล่งน้ำให้เป็น 3 ส่วนในกรณีค่าทศนิยมนิยมใช้กับข้อมูลต่อเนื่อง (Continuous Data) ข้อมูลราสเตอร์แตกต่างจากข้อมูลเวกเตอร์ประการหนึ่งคือ ไม่ได้แยกข้อมูลเชิงพื้นที่และข้อมูลเชิงคุณลักษณะ ดังนั้นข้อมูลราสเตอร์จึงไม่มีตารางข้อมูลคุณลักษณะ (Attribute Table) ยกเว้นข้อมูลราสเตอร์ที่มีค่าเป็นจำนวนเต็ม ซึ่งมีตารางคุณลักษณะแสดงค่าข้อมูลกริดสำหรับใช้หาค่าผลรวมและความถี่ทางสถิติ

2.5 ข้อมูล

2.5.1 เวิลด์ไฟล์ (World file) [4]

เวิลด์ไฟล์ (World file) คือระบบไฟล์ที่ใช้ในการแสดงผลข้อมูลในระบบสารสนเทศทางภูมิศาสตร์ในระบบ UTM (Universal Transverse Mercator coordinate system) มีนามสกุลไฟล์เป็น .tfw เป็นพิกัดที่อยู่ในรูปสี่เหลี่ยมพื้นผ้าที่ประกอบด้วยข้อมูลทั้งหมด 6 บรรทัด

ตัวอย่าง แสดงข้อมูลอยู่ใน 6 บรรทัด

```
32.0
0.0
0.0
-32.0
691200.0
4576000.0
```

บรรทัดที่ 1: A: ขนาดพิกเซลในทิศทางแกน x
บรรทัดที่ 2: D: การหมุนของภาพรอบแกน y
บรรทัดที่ 3: B: การหมุนของภาพรอบแกน x
บรรทัดที่ 4: E: ขนาดพิกเซลในทิศทางแกน y
บรรทัดที่ 5: C: พิกัดจุดในแกน x พิกเซลบนซ้ายของภาพ
บรรทัดที่ 6: F: พิกัดจุดในแกน y พิกเซลบนซ้ายของภาพ

สมการคำนวณระยะทางจาก จุดกำเนิดโซน จะเป็นไปตามสูตร

$$x' = Ax + By + C \quad (2.1)$$

$$y' = Dx + Ey + F \quad (2.2)$$

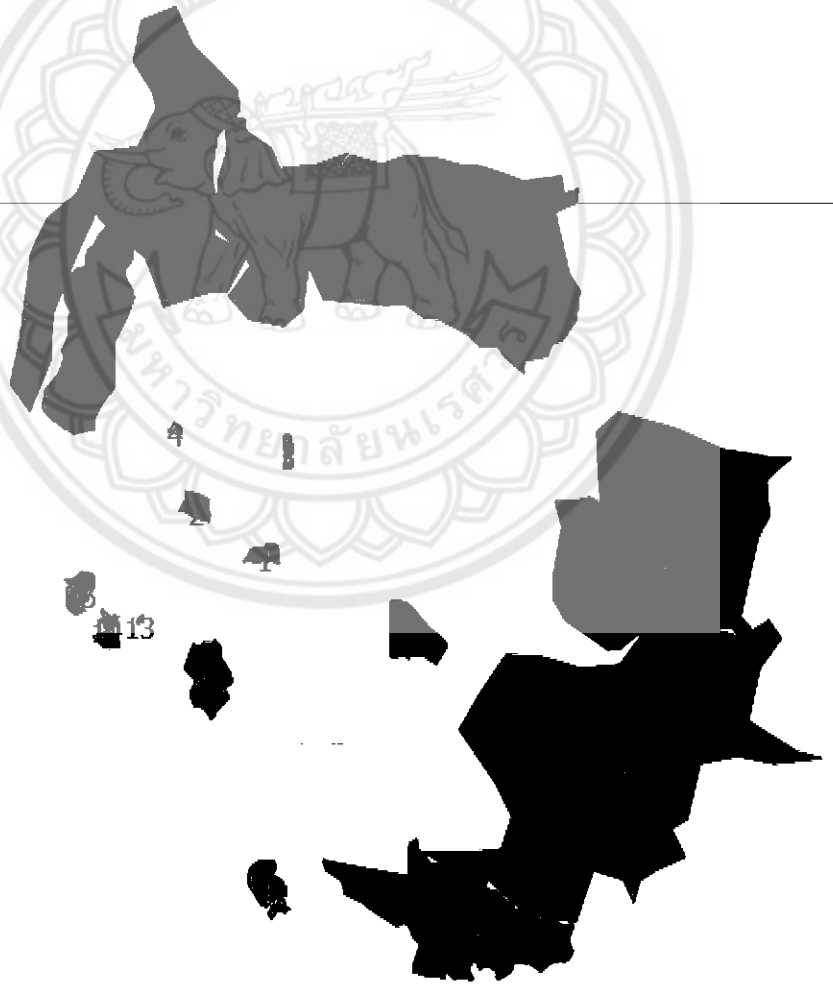
2.5.2 ไฟล์รูปร่าง (Shape File) [5]

ไฟล์รูปร่าง (Shape File) คือไฟล์ที่เก็บข้อมูลเวกเตอร์ และชั้นข้อมูล (Layer) แต่ละประเภทไว้อย่างใดอย่างหนึ่ง ตัวอย่างไฟล์ รูปร่าง (Shape File) ประเภทไฟล์รูปร่าง (Polygon) ไฟล์นั้นก็จะไม่สามารถแทรก เพิ่ม ข้อมูล ประเภทเส้น (Line) หรือจุด (Point) ได้ ซึ่งไฟล์รูปร่าง (Shape File) จะประกอบด้วยไฟล์อย่างน้อย 3 ไฟล์ที่มีการอ้างถึงกันและกันและไม่สามารถขาดไฟล์ใดไฟล์หนึ่งไปได้ ได้แก่

1. **ไฟล์ประเภท (.shp)** ไฟล์นี้จะประกอบไปด้วยข้อมูลเวกเตอร์แต่ละประเภทไว้ซึ่งแต่ละเวกเตอร์ประกอบเป็น Shape File นั้นจะอ้างอิงพิกัด UTM

2. **ไฟล์ประเภท (.dbf)** ไฟล์นี้จะประกอบไปด้วยข้อมูลในรูปแบบตารางฐานข้อมูลเพื่อแสดงรายละเอียดของแต่ละเวกเตอร์

3. **ไฟล์ประเภท (.shx)** ไฟล์นี้จะทำหน้าที่เชื่อมไฟล์ (.shp) และ (.dbf) เข้าด้วยกัน



รูปที่ 2.1 แสดงตัวอย่างข้อมูลใน Shape File

2.6 หลักการที่ใช้ในการคำนวณ

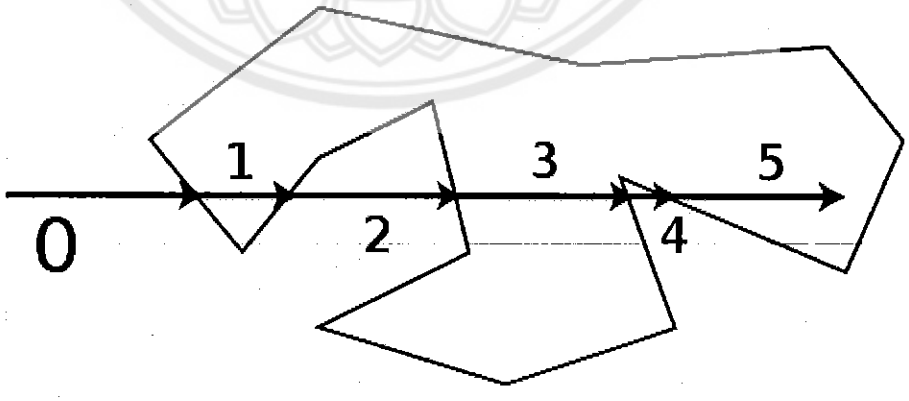
2.6.1 หลักการการหาจุดในรูปหลายเหลี่ยม (Point in Polygon) [6]

การหาจุดในรูปหลายเหลี่ยม (Point in Polygon) คือ การค้นหาจุดที่กำหนดในระนาบว่าอยู่ภายในหรือภายนอกขอบเขตของรูปหลายเหลี่ยม (Polygon) ซึ่งใช้มากในงานคอมพิวเตอร์กราฟฟิก คอมพิวเตอร์วิชชั่น ระบบสารสนเทศทางภูมิศาสตร์ จีไอเอส (GIS) ในการค้นหาจุดว่าอยู่ในหรือนอกรูปหลายเหลี่ยม (Polygon)

อัลกอริทึมที่เลือกใช้ในการหาจุดในรูปหลายเหลี่ยม (Point in Polygon) นั้นคือหลักการของเรย์แครสติ่ง (Ray Casting Algorithm) ซึ่งมีหลักการคำนวณเริ่มจากจุดใดจุดหนึ่งลากไปในแกน x และตัดขอบเขตของรูปหลายเหลี่ยม (Polygon) ที่ลากผ่านและนับจำนวนของจุดที่ตัดขอบเขตของรูปหลายเหลี่ยม (Polygon) ทุกจุดรวมกันถ้าจุดที่นับได้เป็นจำนวนเป็นเลขคู่จะถือว่าจุดนั้นไม่ได้อยู่ในรูปหลายเหลี่ยม (Polygon) แต่ถ้าทุกจุดรวมกันถ้าจุดที่นับได้เป็นจำนวนเป็นเลขคี่จะถือว่าจุดนั้นอยู่ในรูปหลายเหลี่ยม (Polygon)

ข้อดีของการใช้การหาจุดในรูปหลายเหลี่ยม (Point in Polygon) โดยหลักการของ เรย์แครสติ่ง (Ray Casting Algorithm) เป็นหลักการที่ง่ายต่อการทำความเข้าใจและการใช้งาน และหลักการเรย์แครสติ่ง (Ray Casting Algorithm) การใช้งานกับรูปหลายเหลี่ยม (Polygon) อย่างง่ายและเหมาะสมกับการใช้งานในรูปแบบ 2 มิติ

ข้อเสียของการใช้การหาจุดในรูปหลายเหลี่ยม (Point in Polygon) โดยหลักการของเรย์แครสติ่ง (Ray Casting Algorithm) นั้นถ้าเป็นรูปที่มีวงกลมหรือรูปหลายมิติจะทำให้การคำนวณมีความผิดพลาดได้ง่าย ซึ่งหลักการการคำนวณของหลักการเรย์แครสติ่ง (Ray Casting Algorithm) ตัวอย่างในรูปที่ 2.2



รูปที่ 2.2 แสดงหลักการคำนวณของเรย์แครสติ่ง (Ray Casting Algorithm) เพื่อหาการหาจุดในรูปหลายเหลี่ยม (Point in Polygon) [6]

2.6.2 ยุคลิดดิสแทนซ์ (Euclidean Distance) [7]

ยุคลิด ดิสแทนซ์ (Euclidean Distance) คือ การหาระยะทางเซตของจุด โดยเซตของจุดต้องมีจำนวนเท่ากัน ถูกประยุกต์ใช้ในงานหลายๆ ด้านในการคำนวณหาระยะทางจากจุดสองจุดแบบง่ายๆ ซึ่งมีสมการดังนี้

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad (2.3)$$

2.6.3 เฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) [8]

เฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) คือ หลักการการหาระยะทางเซตของจุดโดยที่เซตของจุดนั้นไม่จำเป็นต้องเท่ากัน หรือขอบเขตของพื้นที่รูปภาพ ถูกนำมาประยุกต์ใช้มากในงานคอมพิวเตอร์กราฟฟิก คอมพิวเตอร์วิชชั่นในการเปรียบเทียบภาพหรือการระยะทางจากจุดใดจุดหนึ่งระหว่างรูปหลายเหลี่ยม (Polygon)

ข้อดีของหลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) นั้นคือการทำงานที่รวดเร็ว การคำนวณที่เซตของจุดทั้งสองรูปไม่เท่ากัน และยังสามารถคำนวณได้ในรูปแบบ 2 มิติ 3 มิติ โดยสมการของหลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) ยังสามารถทำความเข้าใจได้ง่าย

ข้อเสียของหลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) การคำนวณมีความคาดเคลื่อนง่าย ไม่มีความถูกต้อง 100 %

ซึ่งมีสมการการคำนวณดังนี้

กำหนดเซตของจุด A

$$A = a_1, \ldots, a_m \qquad (2.4)$$

กำหนดเซตของจุด B

$$B = b_1, \ldots, b_n \qquad (2.5)$$

สมการ Hausdorff Distance

$$H(A,B) = \max(h(A,B), h(B,A)) \qquad (2.6)$$

โดยที่

$$h(A,B) = \max_{a \in A} \min_{b \in B} \|a - b\| \qquad (2.7)$$

$\| . \|$ หมายถึง สมการ การหาระยะทางโดยใช้หลักการเฮาซดอรฟดิสแทนซ์ (Euclidean Distance)

2.6.4 ฟังก์ชันตรีโกณมิติ (Trigonometric Function) [7]

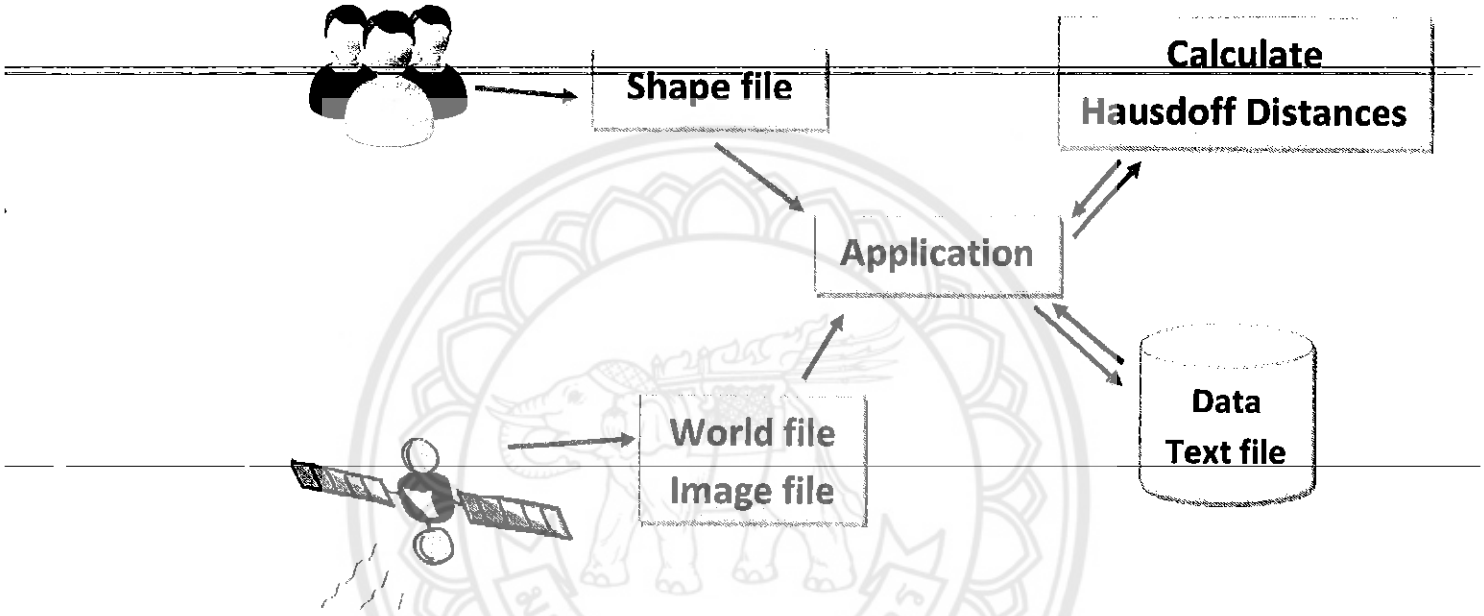
ฟังก์ชันตรีโกณมิติ (Trigonometric Function) คือ ฟังก์ชันของมุม ซึ่งมีความสำคัญในการศึกษารูปสามเหลี่ยมและปรากฏการณ์ในลักษณะเป็นคาบ ฟังก์ชันอาจนิยามด้วยอัตราส่วนของด้าน 2 ด้านของรูปสามเหลี่ยมมุมฉาก หรืออัตราส่วนของพิกัดของจุดบนวงกลมหนึ่งหน่วย หรือนิยามในรูปทั่วไปเช่น อนุกรมอนันต์ หรือสมการเชิงอนุพันธ์ รูปสามเหลี่ยมที่นำมาใช้จะอยู่ในระนาบแบบยุคลิด ดังนั้น ผลรวมของมุมทุกมุมจึงเท่ากับ 180° เสมอ ปัจจุบันฟังก์ชันตรีโกณมิติมีฟังก์อยู่ 6 ฟังก์ชันที่นิยมใช้กันดังตารางที่ 2.4

ตารางที่ 2.4 แสดงฟังก์ชันตรีโกณมิติ [9]

ฟังก์ชัน	ตัวย่อ	ความสัมพันธ์
ไซน์ (Sine)	sin	$\sin\theta = \cos\left(\frac{\pi}{2} - \theta\right)$
โคไซน์ (Cosine)	cos	$\cos\theta = \sin\left(\frac{\pi}{2} - \theta\right)$
แทนเจนต์ (Tangent)	tan (หรือ tg)	$\tan\theta = \frac{1}{\cot\theta} = \frac{\sin\theta}{\cos\theta} = \cot\left(\frac{\pi}{2} - \theta\right)$
โคแทนเจนต์ (Cotangent)	cot (หรือ ctg หรือ ctn)	$\cot\theta = \frac{1}{\tan\theta} = \frac{\cos\theta}{\sin\theta} = \tan\left(\frac{\pi}{2} - \theta\right)$
ซีแคนต์ (Secant)	sec	$\sec\theta = \frac{1}{\cos\theta} = \csc\left(\frac{\pi}{2} - \theta\right)$
โคซีแคนต์ (Cosecant)	csc (หรือ cosec)	$\csc\theta = \frac{1}{\sin\theta} = \sec\left(\frac{\pi}{2} - \theta\right)$

บทที่ 3
ขั้นตอนการดำเนินงาน

ในบทนี้จะกล่าวถึงรายละเอียดของการออกแบบระบบขั้นตอนการดำเนินงาน ซึ่งสามารถอธิบายขั้นตอนได้ตามรูปที่ 3.1



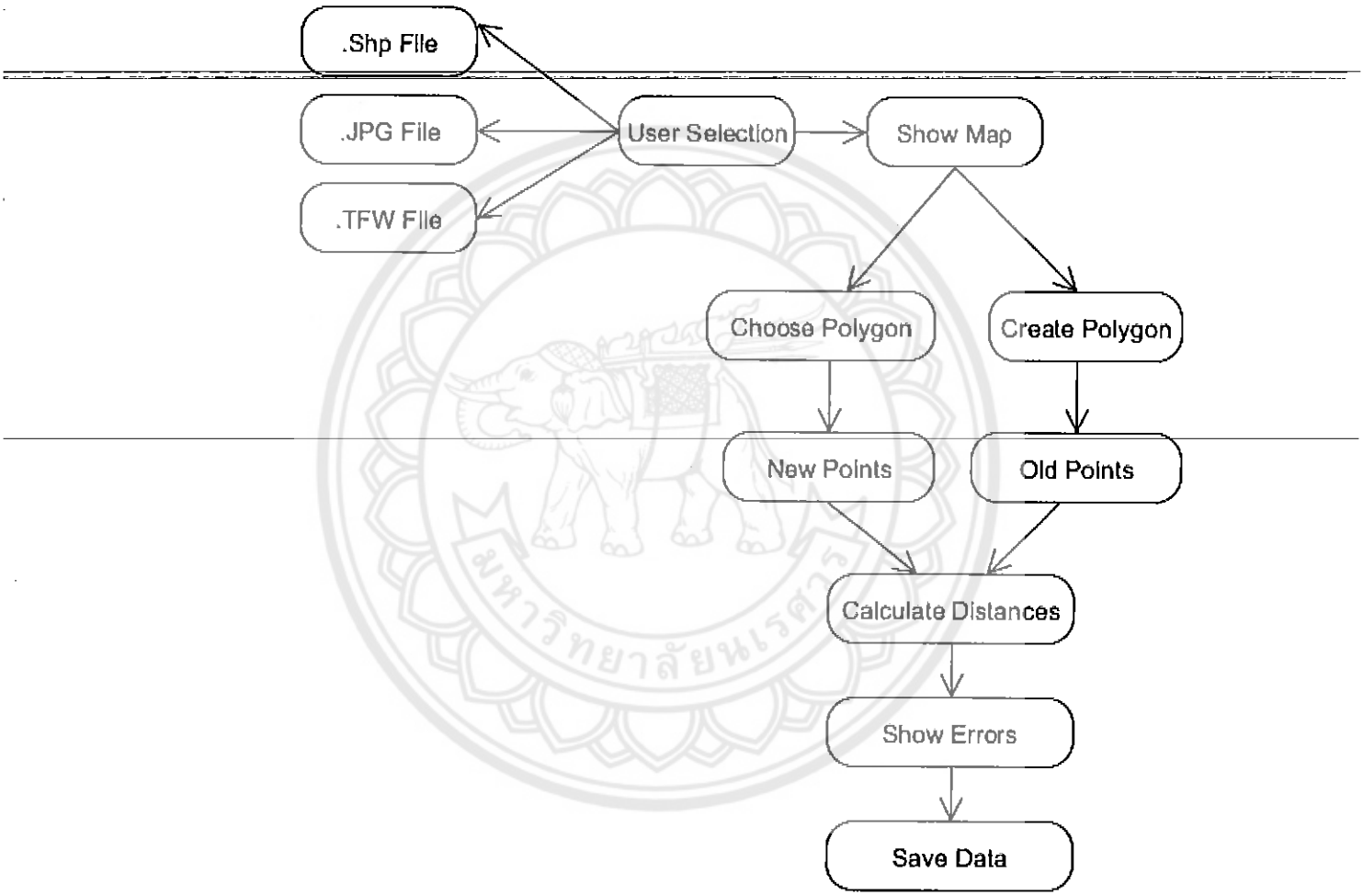
รูปที่ 3.1 แสดงการวางแผนการปฏิบัติงาน

จากรูปที่ 3.1 แสดงการทำงานของระบบการวิเคราะห์ภาพถ่ายจากดาวเทียมโดยการนำข้อมูลเข้ามาใช้งานทั้งหมด 3 ส่วนโดยส่วนที่ 1 เป็นข้อมูลเชปไฟล์ (Shape file) ที่ประกอบด้วยข้อมูลรูปหลายเหลี่ยม (Polygon) ที่ได้จากการสำรวจพื้นที่จริง ส่วนที่ 2 และ 3 คือข้อมูลที่ได้จากดาวเทียมประกอบด้วยข้อมูลภาพ (Image file) และพิกัดของภาพ (World file) นำข้อมูลมาซ้อนทับกันเพื่อแสดงผลในโปรแกรมที่ได้ออกแบบขึ้น โดยให้ผู้ใช้งานสามารถออกแบบพื้นที่ป่าจากภาพถ่ายดาวเทียมเพื่อเปรียบเทียบกับพื้นที่ป่าที่จากการสำรวจพื้นที่จริง และแสดงค่าความคลาดเคลื่อนด้วยหลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) และยังสามารถเก็บข้อมูลเพื่อใช้งานในรูปแบบของไฟล์ข้อความโดยใช้โปรแกรมแมทแลป (MATLAB) ในการทำงานของโปรแกรมทั้งหมด

3.1 การออกแบบระบบ

3.1.1 โครงสร้างของโปรแกรม (Program Structure)

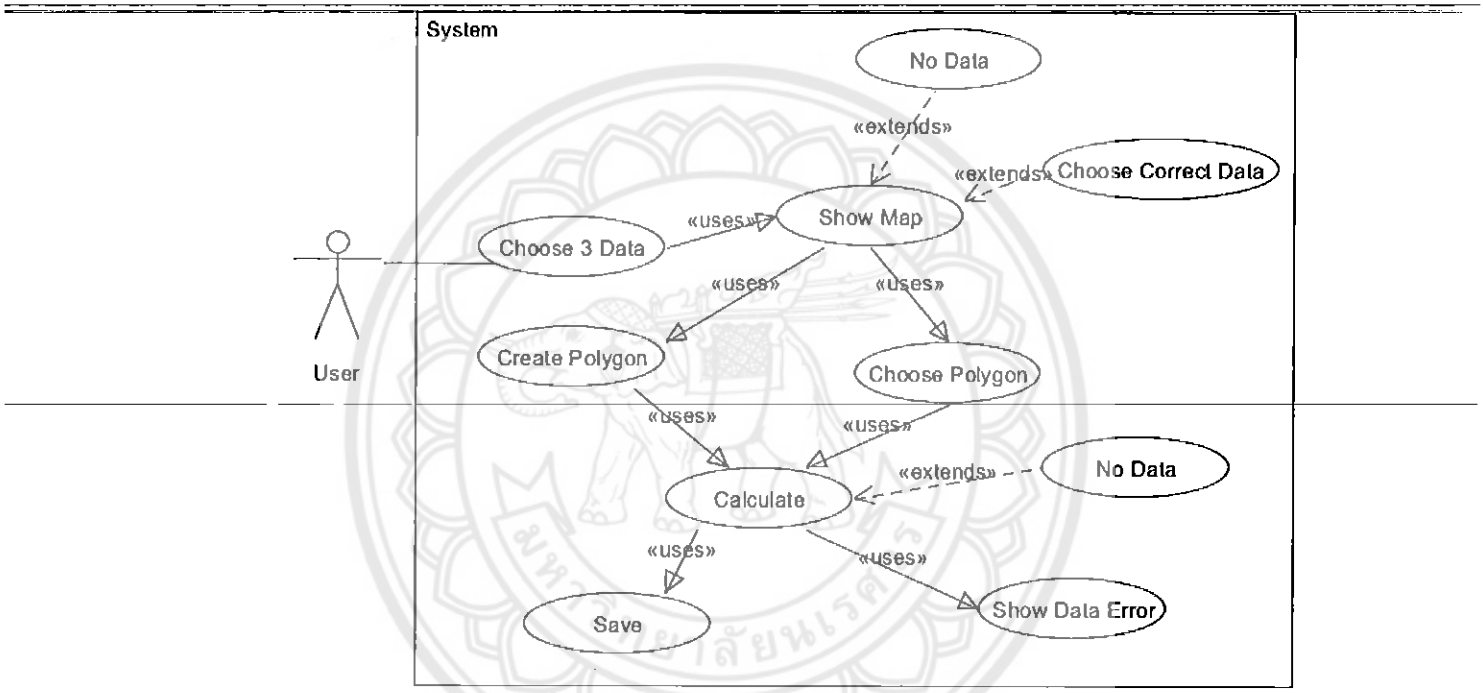
ในการออกแบบโครงสร้างของโปรแกรมเพื่อหาความคลาดเคลื่อนของข้อมูลพื้นที่ป่าจากดาวเทียม เปรียบเทียบกับข้อมูลพื้นที่ป่าจากการสำรวจพื้นที่จริง เพื่อตรวจสอบความคลาดเคลื่อนของข้อมูล ซึ่งมีการมีการออกแบบโครงสร้างของโปรแกรมดังต่อไปนี้



รูปที่ 3.2 แสดงถึงโครงสร้างของระบบ

3.1.2 Use Case Diagram

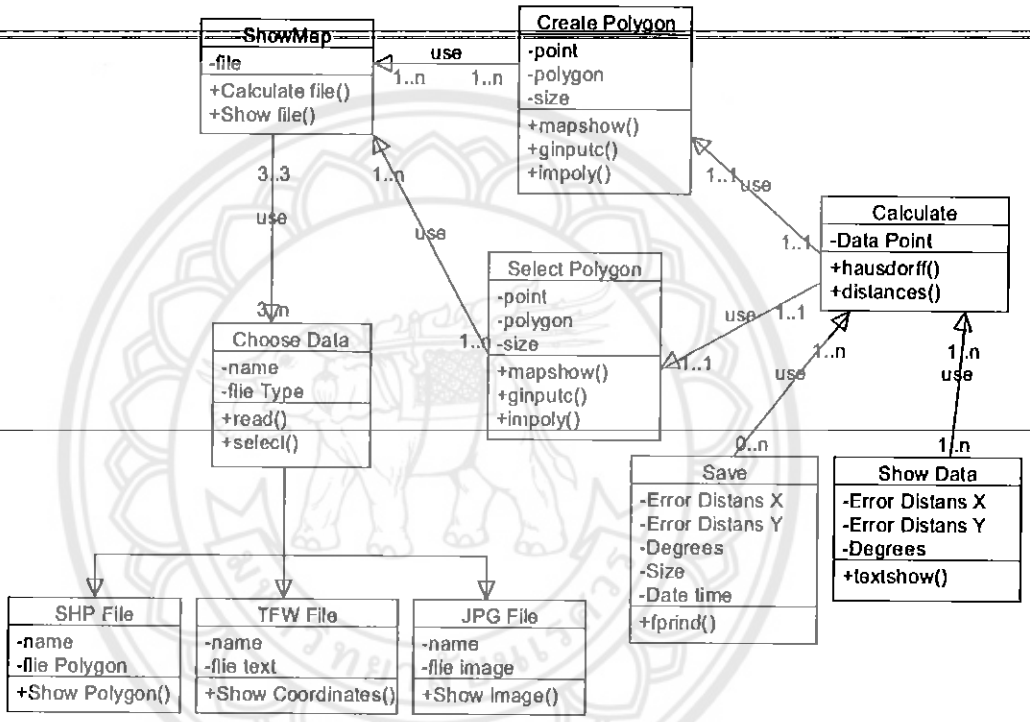
Use Case Diagram แสดงการทำงานของระบบเพื่อให้มีความเข้าใจภาพรวมของระบบ โดยผู้ใช้งาน (User) จะเป็นคนเลือกข้อมูลทั้ง 3 ส่วนเพื่อใช้งานในระบบถ้าไม่มีข้อมูลหรือเลือกข้อมูลผิดโปรแกรมก็ไม่สามารถทำงานได้ และเมื่อได้ข้อมูลครบทั้ง 3 ส่วนแล้วโปรแกรมจะให้เลือกรูปหลายเหลี่ยม (Polygon) และสร้างรูปหลายเหลี่ยม (Polygon) มาเปรียบเทียบซึ่งกัน และคำนวณค่าความคาดเคลื่อนแสดงผล และนำข้อมูลไปใช้งานต่อไปได้ ซึ่งแสดงในรูปที่ 3.3



รูปที่ 3.3 แสดงถึง Use Case Diagram ของระบบ

3.1.3 Class Diagram

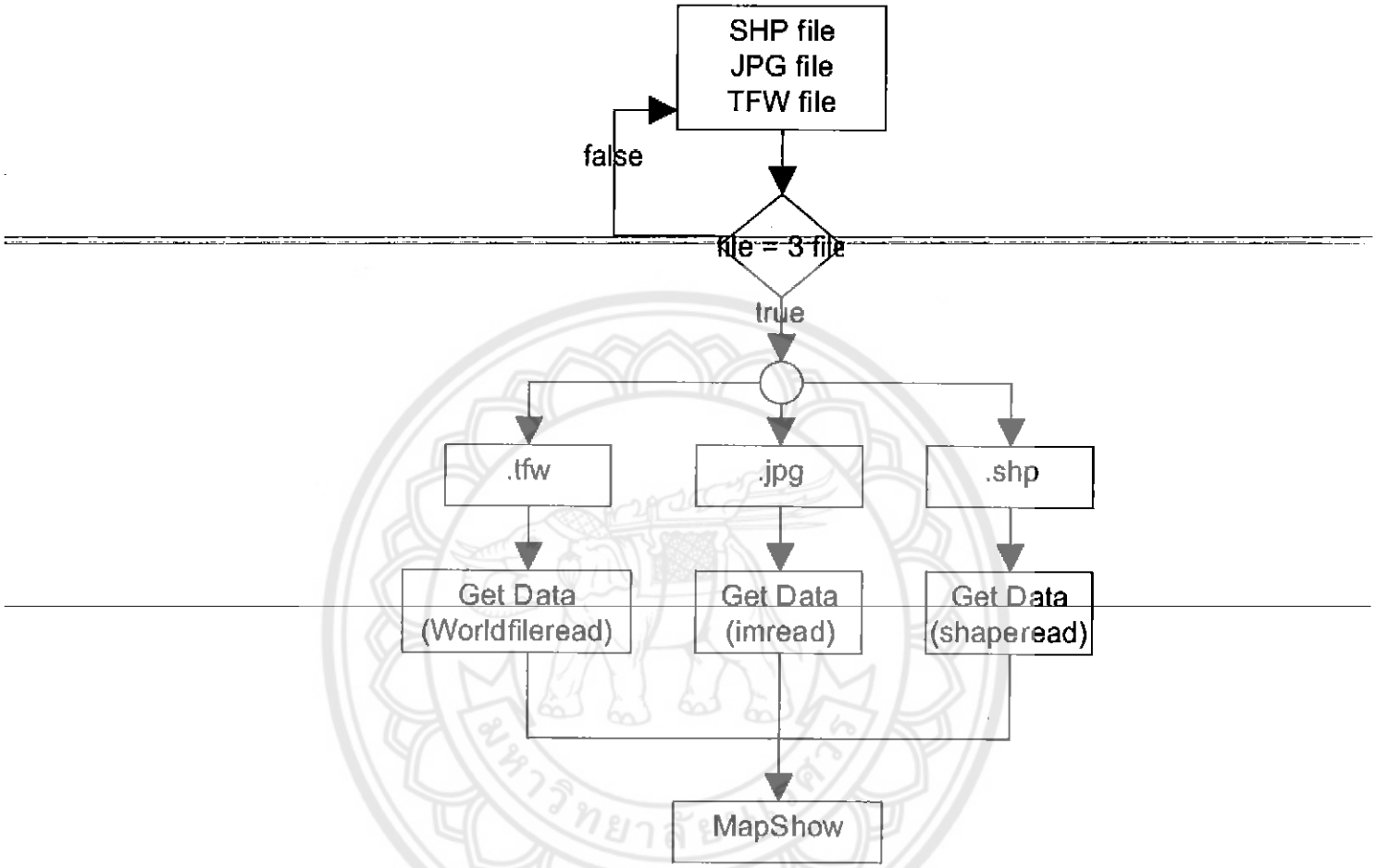
Class Diagram แสดงการทำงานของระบบเพื่อให้มีความเข้าใจภาพรวมของระบบโดยละเอียดตั้งแต่ผู้ใช้งาน (User) จะเป็นคนเลือกข้อมูลทั้ง 3 ส่วนเพื่อใช้งานในระบบถ้าไม่มีข้อมูลหรือเลือกข้อมูลผิดโปรแกรมก็ไม่สามารถทำงานได้ และเมื่อได้ข้อมูลครบทั้ง 3 ส่วนแล้วโปรแกรมจะให้เลือกรูปหลายเหลี่ยม (Polygon) และสร้างรูปหลายเหลี่ยม (Polygon) มาเปรียบเทียบซึ่งกัน และคำนวณค่าความคาดเคลื่อนแสดงผล และนำข้อมูลไปใช้งานต่อไปได้ ซึ่งแสดงในรูปที่ 3.4



รูปที่ 3.4 แสดงถึง Class Diagram ของระบบ

3.2 การออกแบบโปรแกรม

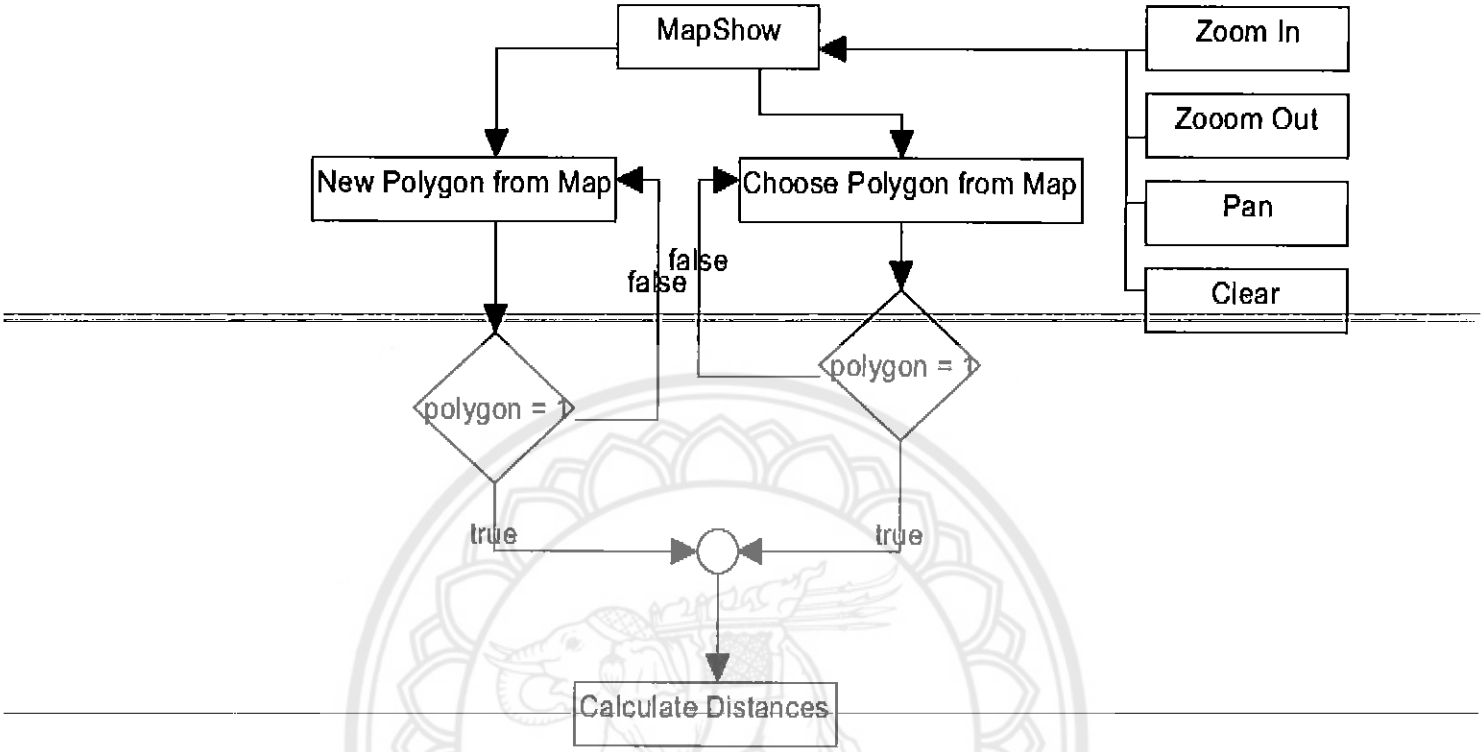
3.2.1 การออกแบบโปรแกรมในส่วนการรับข้อมูล และแสดงผลข้อมูล



รูปที่ 3.5 แสดงถึง Flow Chart ของการรับข้อมูลและแสดงผลข้อมูล

การออกแบบโปรแกรมในส่วนการรับข้อมูลเข้ามาใช้งานทั้งหมด 3 ส่วนซึ่งข้อมูลทั้ง 3 ส่วนนั้นประกอบด้วยข้อมูลรูปภาพ (Image file) เวิลด์ไฟล์ (World file) ไฟล์รูปร่าง (Shape File) โดยมีนามสกุล .jpg .shp .tfw ถ้าข้อมูลมีไม่ครบทั้ง 3 ส่วน ให้โปรแกรมกลับไปเริ่มทำใหม่ โดยจะให้เลือกข้อมูลจนกว่าข้อมูลจะครบทั้ง 3 ส่วน เมื่อเลือกข้อมูลทั้ง 3 ส่วนครบแล้วโปรแกรมจะใช้ คำสั่ง Worldfileread ในการอ่านข้อมูลจากเวิลด์ไฟล์ (World file) คำสั่ง Imread ในการอ่านข้อมูลจากไฟล์ภาพ (Image file และคำสั่ง Shaperead ในการอ่านข้อมูลไฟล์รูปร่าง (Shape File) และเมื่อโปรแกรมอ่านข้อมูลทั้ง 3 ส่วนครบแล้วโปรแกรมจะนำข้อมูลมาซ้อนทับกันและแสดงผลออกมาซึ่งจะอธิบายเป็นเป็นรูปภาพในหัวข้อที่ 3.4

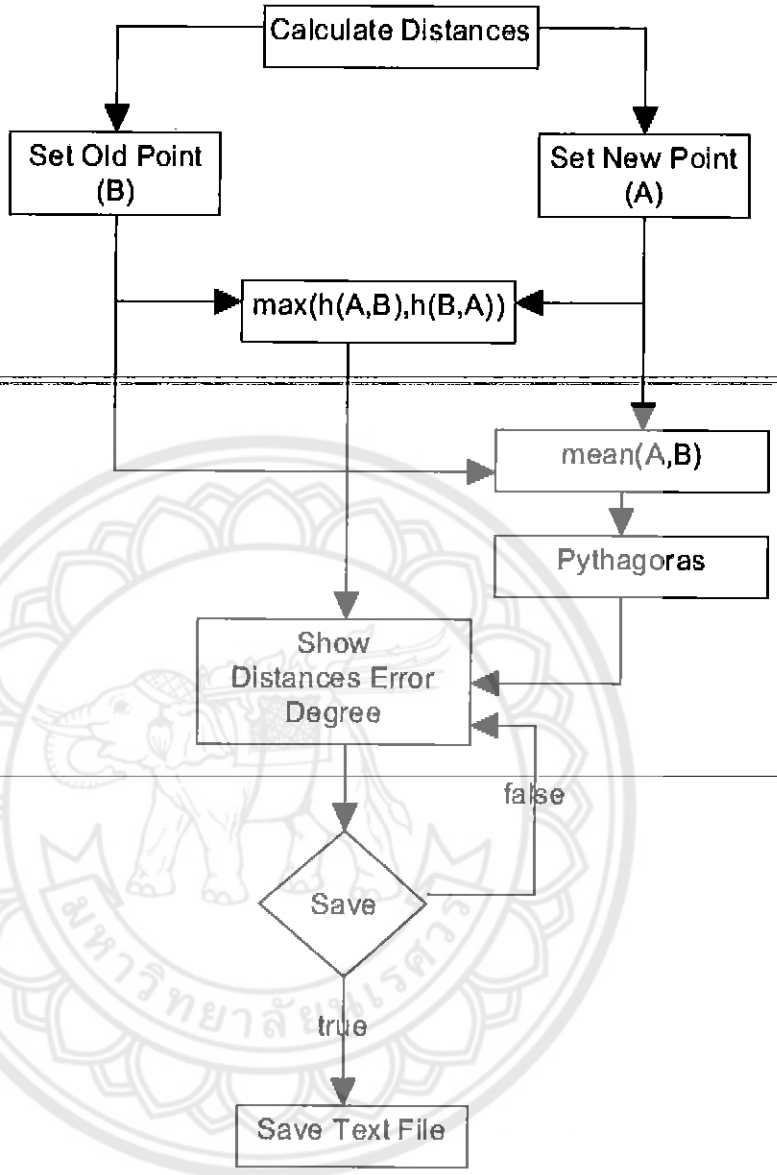
3.2.2 การออกแบบโปรแกรมในส่วนเลือกข้อมูลและสร้างข้อมูล



รูปที่ 3.6 แสดงถึง Flow Chart เลือกข้อมูลและสร้างข้อมูล

เมื่อโปรแกรมแสดงผลตามหัวข้อ 3.2.1 แล้วผู้ใช้งานสามารถเลือกย่อภาพ ขยายภาพ และเลื่อนภาพเพื่อให้การเลือกรูปหลายเหลี่ยม (Polygon) สามารถทำได้สะดวกมากยิ่งขึ้น และเลือกรูปหลายเหลี่ยม (Polygon) 1 รูปจากหลายเหลี่ยม (Polygon) ทั้งหมด (จากข้อมูลไฟล์รูปร่าง Shape File) เมื่อเลือกรูปหลายเหลี่ยม (Polygon) แล้วโปรแกรมจะให้ผู้ใช้งานสร้างรูปหลายเหลี่ยม (Polygon) ใหม่ออกมา 1 รูป (จากการมองด้วยตา) โปรแกรมจะนำเอารูปหลายเหลี่ยม (Polygon) ทั้ง 2 รูปที่สร้างขึ้นมาคำนวณหาระยะทางความคลาดเคลื่อน และทิศทางที่เคลื่อนที่ไปใช้หลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) และปิทาโกรัส (Pythagoras)

3.2.3 การออกแบบโปรแกรมในส่วนการคำนวณค่าความคลาดเคลื่อนของข้อมูล



รูปที่ 3.7 แสดงถึง Flow Chart การคำนวณและการบันทึกข้อมูล

จากรูปที่ 3.7 เมื่อโปรแกรมได้รูปหลายเหลี่ยม (Polygon) ทั้ง 2 รูปแล้วจะได้ค่าของจุดในรูปหลายเหลี่ยม (Polygon) แต่ละรูป และนำเซ็ทของจุดของรูปหลายเหลี่ยม (Set Point in Polygon) นั้นมาคำนวณหาระยะทางความคลาดเคลื่อน และทิศทางที่เคลื่อนที่ไปโดยใช้หลักการเฮา-ซดอรฟดิสแทนซ์ (Hausdorff Distance) และปิทาโกรัส (Pythagoras) แล้วแสดงผลโดยถ้าผู้ใช้งานเลือกเซพก็ข้อมูลโปรแกรมก็จะบันทึกค่าที่ได้จากการแสดงผลทั้งหมดในรูปของข้อมูลอักษร (text file)

3.3 ระบบการคำนวณ

3.3.1 การคำนวณพิกัดจากเวิลด์ไฟล์ (World file)

ตัวอย่าง ข้อมูลในเวิลด์ไฟล์ (World file)

32.0

0.0

0.0

-32.0

691200.0

4576000.0

บรรทัดที่ 1: A: ขนาดพิกเซลในทิศทางแกน x A มีค่า = 32

บรรทัดที่ 2: D: การหมุนของภาพรอบแกน y D มีค่า = 0

บรรทัดที่ 3: B: การหมุนของภาพรอบแกน x B มีค่า = 0

บรรทัดที่ 4: E: ขนาดฟิกเซลในทิศทางแกน y E มีค่า = -32

บรรทัดที่ 5: C: พิกัดจุดในแกน x ฟิกเซลบนซ้ายของภาพ C มีค่า = 691200

บรรทัดที่ 6: F: พิกัดจุดในแกน y ฟิกเซลบนซ้ายของภาพ F มีค่า = 4576000

สมการที่ใช้ในคำนวณระยะทางจาก จุดกำเนิดโซน จะเป็นไปตามสูตร

$$x' = Ax + By + C \quad (3.1)$$

$$y' = Dx + Ey + F \quad (3.2)$$

ค่า x คือ ค่าของแถวที่ต้องการ ตัวอย่าง แถวที่ 10

ค่า y คือ ค่าของหลักที่ต้องการ ตัวอย่าง หลักที่ 20

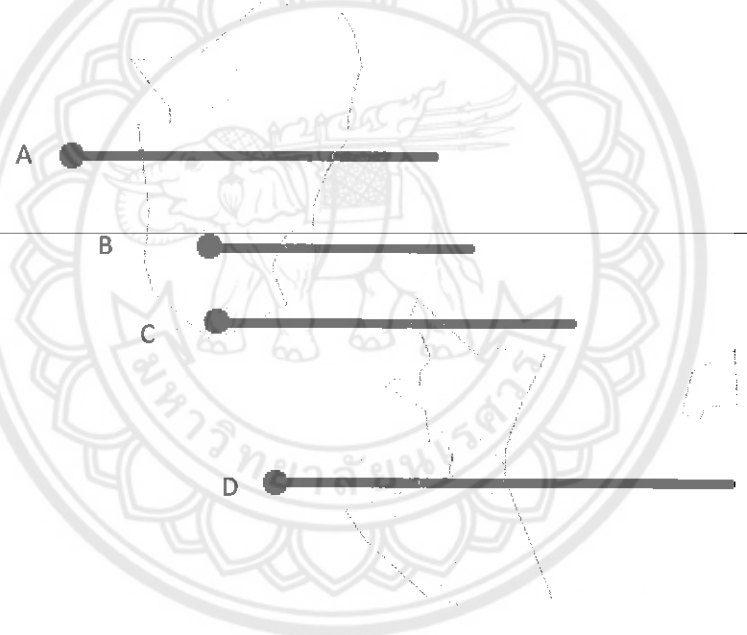
x` = 35(10) + 0(20) + 691200 = 350 + 69120 = 69420

y` = 0(10) + (-32)(10) + 4576000 = -320 + 4576000 = 4675680

จากตัวอย่างการคำนวณจึงเป็นหลักการที่เอามาใช้ในการใส่ค่าพิกัดลงไปในภาพถ่ายจากดาวเทียมทั้งหมดโดยโปรแกรมจะคำนวณค่าและใส่พิกัดเข้าไปในรูปภาพทั้งหมดโดยอัตโนมัติ

3.3.2 การคำนวณเลือกรูปหลายเหลี่ยม (Polygon)

การเลือกรูปหลายเหลี่ยม (Polygon) นั้นใช้หลักการของ เรย์แครสติ่ง (Ray Casting Algorithm) การเลือกจุดในรูปหลายเหลี่ยม (Point in Polygon) โดยการเลือกจุดใดจุดหนึ่งของรูป และใช้หลักการเทียบจุดจากตำแหน่งที่จุดเลื่อนไปตามแนวระนาบ เมื่อตัดกับขอบของรูปหลายเหลี่ยม (Polygon) ใดๆ จะทำการเก็บจุดตัดขอบของรูปนั้นๆไว้ ถ้าผลรวมของจำนวนจุดที่ลากผ่านนั้นมีค่าเป็นเลขคี่ แสดงว่า จุดนั้นอยู่ในรูปหลายเหลี่ยม (Polygon) แต่ถ้าผลรวมของจำนวนจุดที่ลากผ่านนั้นมีค่าเป็นเลขคู่แสดงว่า จุดนั้นไม่ได้อยู่ในรูปหลายเหลี่ยม (Polygon) ซึ่งแสดงตัวอย่างการคำนวณตามรูปที่ 3.8



รูปที่ 3.8 แสดงถึงการคำนวณจุดของโปรแกรม

จากรูปกำหนดจุดทั้ง 4 ได้แก่ จุด A, B, C และ D โดยกำหนดให้แต่ละจุดลากผ่านตามแนวระนาบ ซึ่งจะเกิดจุดตัด (จุดสีฟ้า) ขอบเขตของรูปหลายเหลี่ยม (Polygon) ในจุดที่ต่างกันและนับผลรวมของจำนวนจุดที่ตัดขอบเขตของแต่ละตัวไว้ ดังนี้

จุด A มีจุดตัดทั้งหมด 2 จุด เพราะฉะนั้น จุด A ไม่ได้อยู่ข้างในรูปหลายเหลี่ยม (Polygon)

จุด B มีจุดตัดทั้งหมด 1 จุด เพราะฉะนั้น จุด B อยู่ข้างในรูปหลายเหลี่ยม (Polygon)

จุด C มีจุดตัดทั้งหมด 3 จุด เพราะฉะนั้น จุด C อยู่ข้างในรูปหลายเหลี่ยม (Polygon)

จุด D มีจุดตัดทั้งหมด 2 จุด เพราะฉะนั้น จุด D จึงไม่ได้อยู่ข้างในรูปหลายเหลี่ยม (Polygon)

จากหลักการคำนวณจึงเป็นหลักการที่นำมาใช้ในการเลือกขอบเขตของรูปหลายเหลี่ยม (Polygon) จากภาพถ่ายดาวเทียมเพื่อนำพิกัดจากรูปหลายเหลี่ยม (Polygon) มาใช้ในการคำนวณหาระยะทาง และทิศทาง

3.3.3 การคำนวณระยะทางจากเฮาซดอรฟดิสแทนซ์ (Hausdorff Distances)

การคำนวณหาระยะทางเฮาซดอรฟดิสแทนซ์ (Hausdorff Distances) นั้นใช้การคำนวณระยะทางจากจุดของรูปหลายเหลี่ยม (Polygon) ทุกจุดเทียบกับจุดใดจุดหนึ่งของรูปหลายเหลี่ยม (Polygon) ที่ถูกเปรียบเทียบ และเลือกระยะทางที่สั้นที่สุด แล้วนำระยะทางที่สั้นที่สุดทั้งหมดมาเลือกค่าที่มากที่สุด สามารถใช้งานกับรูปที่มีเซตของจุดเท่ากันหรือไม่เท่ากันก็ได้

กำหนดเซตของจุด A

$$A = a_1, \dots, a_m \tag{3.3}$$

กำหนดเซตของจุด B

$$B = b_1, \dots, b_n \tag{3.4}$$

สมการ Hausdorff Distance

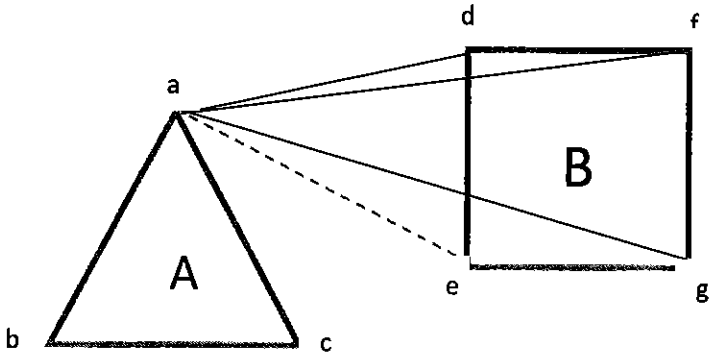
$$H(A,B) = \max(h(A,B), h(B,A)) \tag{3.5}$$

โดยที่

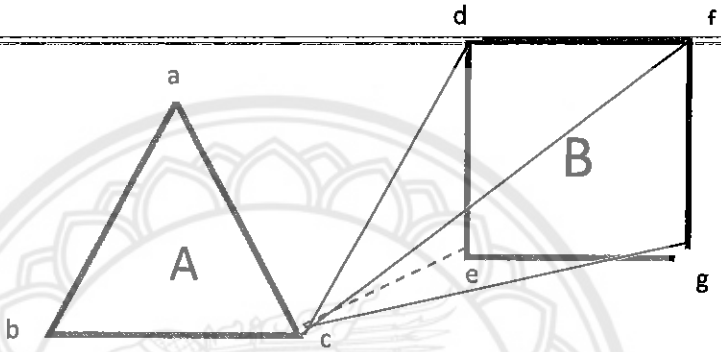
$$h(A,B) = \max_{a \in A} \min_{b \in B} \|a - b\| \tag{3.6}$$

$\| . \|$ หมายถึง สมการ การหาระยะทางโดยใช้หลักการยุคลิดดิสแทนซ์ (Euclidean Distance)

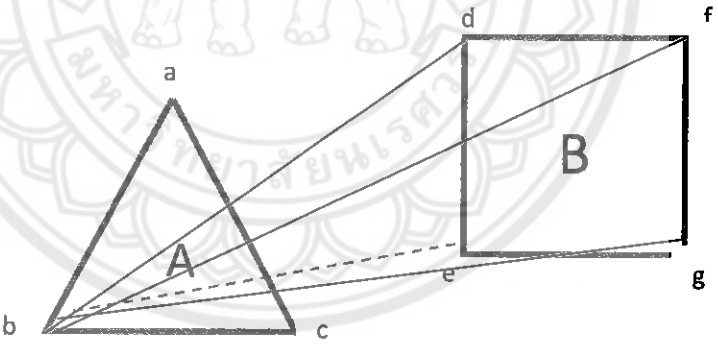
แสดงในตัวอย่างการเลือกระยะทางที่สั้นที่สุดจากรูป A กับรูป B ในรูปที่ 3.9 – 3.11 และตัวอย่างการเลือกระยะทางที่สั้นที่สุดจากรูป B กับรูป A ในรูปที่ 3.12 – 3.15



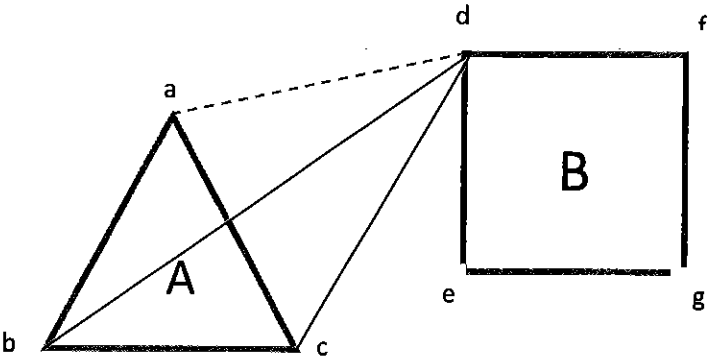
รูปที่ 3.9 แสดงการหาระยะทางเส้นที่สั้นที่สุดคือ (เส้นประ) a ไป e ของรูป A และรูป B



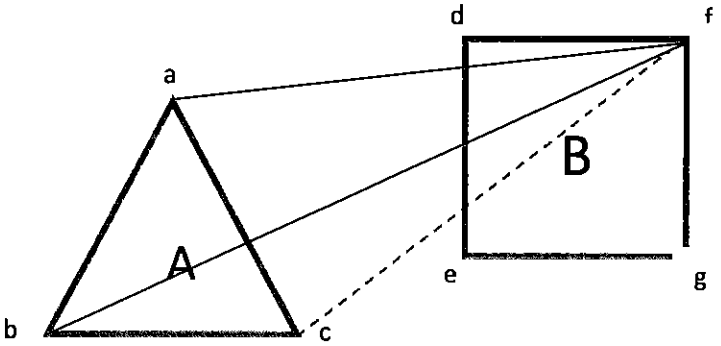
รูปที่ 3.10 แสดงการหาระยะทางเส้นที่สั้นที่สุดคือ (เส้นประ) c ไป e ของรูป A และรูป B



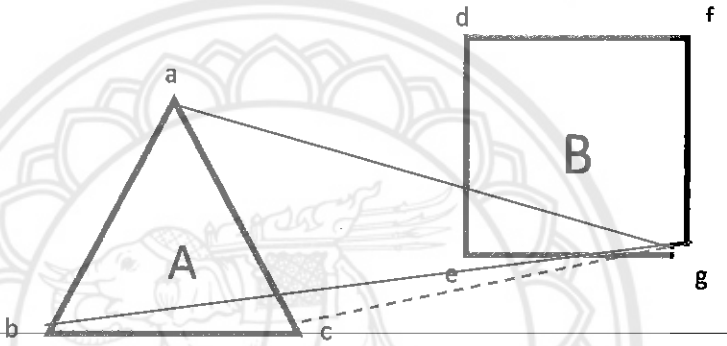
รูปที่ 3.11 แสดงการหาระยะทางเส้นที่สั้นที่สุดคือ (เส้นประ) b ไป e ของรูป A และรูป B



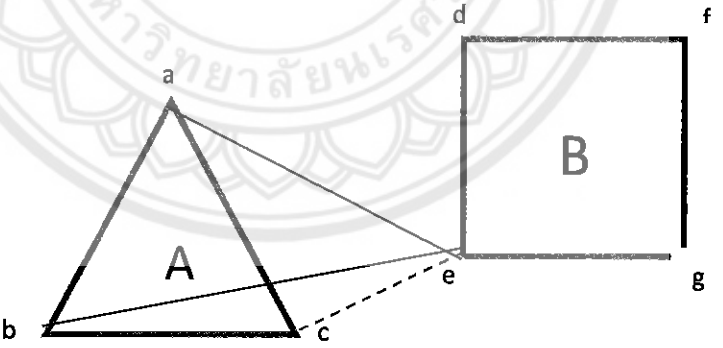
รูปที่ 3.12 แสดงการหาระยะทางเส้นที่สั้นที่สุดคือ (เส้นประ) d ไป a ของรูป B และรูป



รูปที่ 3.13 แสดงการหาระยะทางเส้นที่สั้นที่สุดคือ (เส้นประ) f ไป c ของรูป B และรูป A



รูปที่ 3.14 แสดงการหาระยะทางเส้นที่สั้นที่สุดคือ (เส้นประ) g ไป c ของรูป B และรูป A



รูปที่ 3.15 แสดงการหาระยะทางเส้นที่สั้นที่สุดคือ (เส้นประ) e ไป c ของรูป B และรูป A

จากรูปที่ 3.9 – 3.11 ระยะทางที่สุดของของภาพ A และ B ประกอบด้วย

- เส้นทาง a ไป e
- เส้นทาง c ไป e
- เส้นทาง b ไป e

เลือกระยะทางที่มีค่ามากที่สุดของภาพ A และ B คือ เส้นทาง b ไป e

จากรูปที่ 3.12 – 3.15 ระยะทางที่สุดของของภาพ B และ A ประกอบด้วย

- เส้นทาง d ไป a
- เส้นทาง f ไป c
- เส้นทาง g ไป c
- เส้นทาง e ไป c

เลือกระยะทางที่มีค่ามากที่สุดของภาพ B และ A คือ เส้นทาง f ไป c

เลือกเส้นทางที่มากที่สุดระหว่าง เส้นทาง b ไป e และ เส้นทาง f ไป c คือ เส้นทาง f ไป c

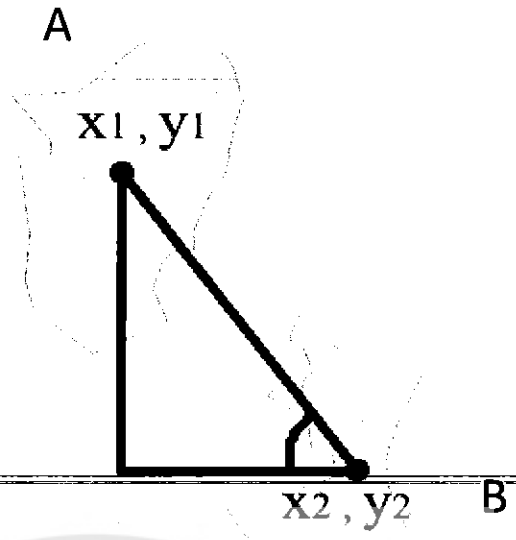
จากหลักการคำนวณเฮาซดอรฟดิสแทนซ์ (Hausdorff Distances) ถูกนำไปใช้ในการเปรียบเทียบรูปหลายเหลี่ยม (Polygon) ในการใช้งานโปรแกรมในหน้าที่ 35 โดยการคำนวณค่าความคลาดเคลื่อนใน แกน x และแกน y ที่มีเซตจุดพิกัดที่แตกต่างกันเพื่อหาระยะทางที่คลาดเคลื่อนของรูปหลายเหลี่ยม (Polygon) ที่แตกต่างกันได้

3.3.4 การหาระยะกระจัดและทิศทาง

หลักการการหาระยะกระจัดนั้นใช้การหาจุดศูนย์กลางของรูปหลายเหลี่ยม (Polygon) เพื่อเปรียบเทียบกัน และใช้การหาค่าของมุมเพื่อบอกทิศทางของรูปหลายเหลี่ยม (Polygon) ที่เปรียบเทียบว่าอยู่ในทิศทางใดซึ่งมีหลักการการคำนวณดังนี้

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad (3.7)$$

โดยที่ x และ y คือเซตทุกจุดในรูปรูปหลายเหลี่ยม (Polygon) ซึ่งแสดงในรูปที่ 3.16



รูปที่ 3.16 แสดงจุดกึ่งกลางของรูปหลายเหลี่ยม (Polygon)

โดยที่เซตของรูปหลายเหลี่ยม (Polygon) A = {x1, y1} เซตของรูปหลายเหลี่ยม (Polygon) B = {x2,y2} แสดงในรูปที่ 3.16

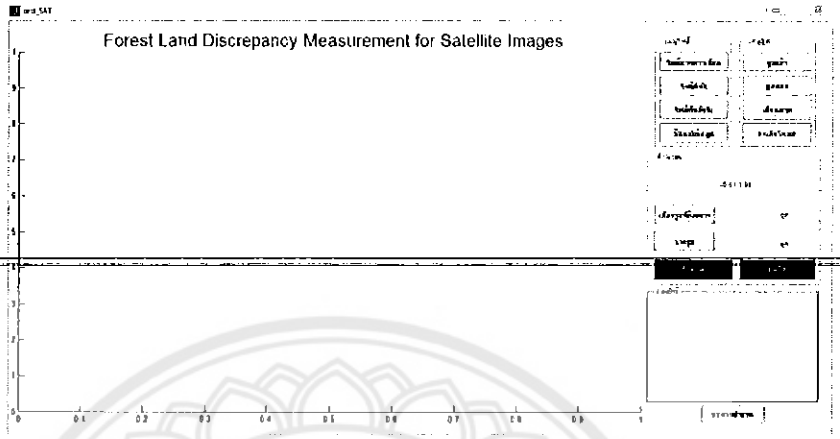
ระยะกระจัด = $\sqrt{(x2-x1)2 + (y2-y1)2}$

ทิศทางการเคลื่อนที่ = arc tan{(y2-y1)/(x1-x2)}

จากหลักการคำนวณหาระยะกระจัดและทิศทางถูกนำไปใช้ในการเปรียบเทียบรูปหลายเหลี่ยม (Polygon) ถูกสร้างขึ้นจากภาพถ่ายดาวเทียมและรูปหลายเหลี่ยม (Polygon) ที่เก็บจากพื้นที่จริงเพื่อนำค่าความคลาดเคลื่อนในแนวกระจัด ในหน้าที่ 35 และทิศทางในหน่วย องศา เพื่อนำค่าความคลาดเคลื่อนไปใช้ต่อไป

3.4 การออกแบบโปรแกรมและหลักการทำงานของโปรแกรม

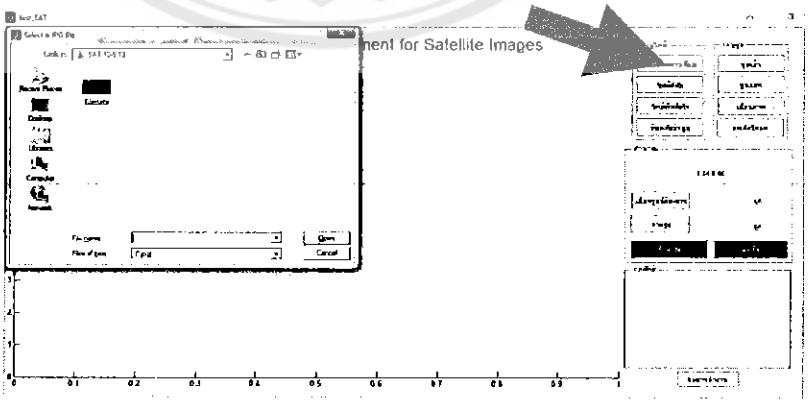
การออกแบบในส่วนของ GUI เพื่อให้ผู้ใช้งานสามารถใช้งานได้อย่างสะดวก



รูปที่ 3.17 แสดงถึงปุ่มนำเข้าข้อมูลของโปรแกรมใน Menu File ได้แก่ ไฟล์ภาพจากดาวเทียม, ไฟล์พิกัด, ไฟล์พื้นที่จริง

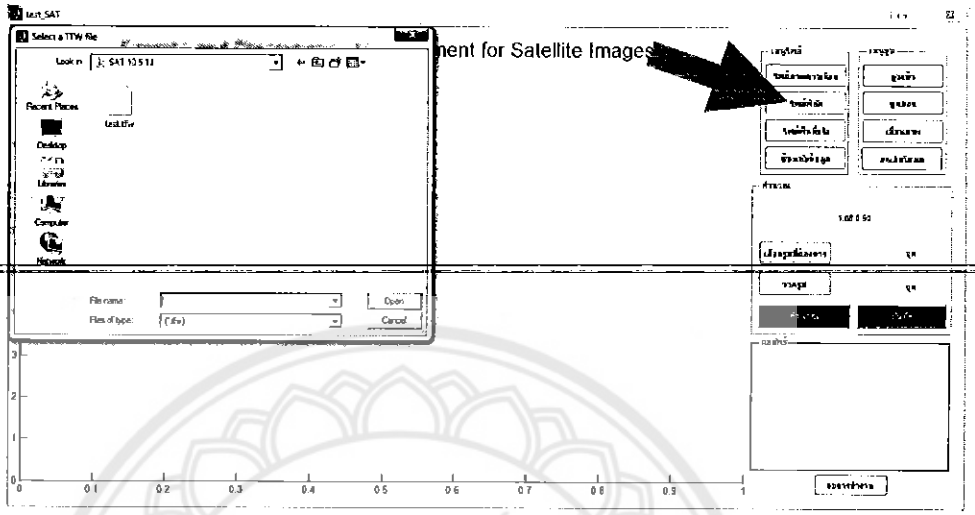
ในการรับข้อมูลมีทั้งหมด 3 รูปแบบ ซึ่งประกอบด้วย ไฟล์ไฟล์ภาพจากดาวเทียม, ไฟล์พิกัด , ไฟล์พื้นที่จริง ซึ่งต้องเลือกไฟล์ทั้งหมด 3 ไฟล์ในส่วนของเมนูไฟล์และปุ่มซ้อนทับภาพทั้ง 3 แบบเข้าด้วยกัน ตามรูปที่ 3.17

การเลือกไฟล์ภาพจากดาวเทียม เมื่อผู้ใช้งานทำการเลือกไฟล์ แล้วโปรแกรมจะใช้คำสั่ง imread() ในการอ่านภาพ และนำข้อมูลมาแปลงโดยใช้คำสั่ง imfilter() เพื่อลดการแตกของภาพเวลาซูมเข้าเพื่อดูภาพระยะใกล้ ทำให้ภาพดูคมชัดมากยิ่งขึ้นตามรูปที่ 3.18



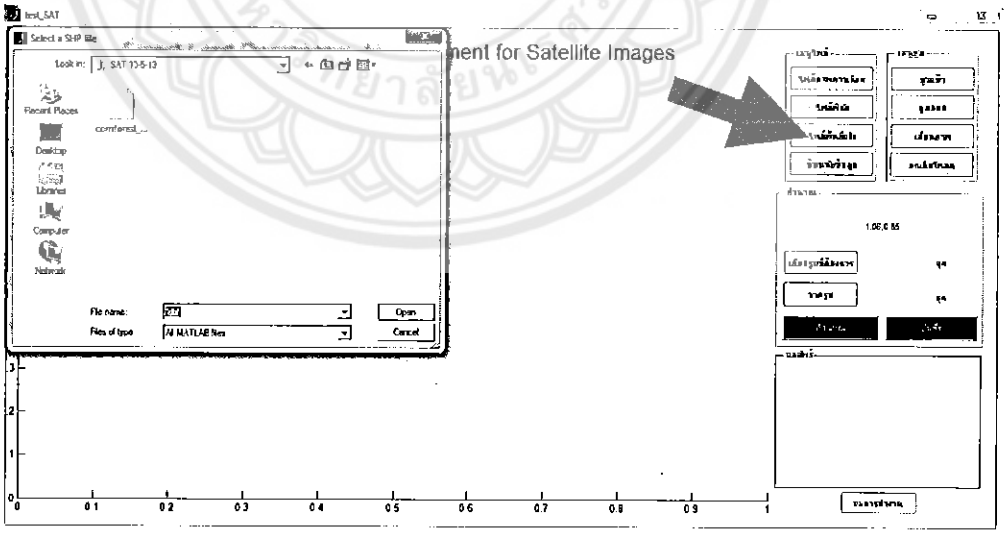
รูปที่ 3.18 แสดงถึงปุ่มใช้ข้อมูลของโปรแกรมในเมนูไฟล์ในส่วนการเลือกไฟล์ภาพจากดาวเทียม

การเลือกไฟล์พิกัด (World File) นั้นเมื่อผู้ใช้งานทำการเลือกไฟล์แล้วโปรแกรมจะใช้คำสั่ง worlfileread() เพื่ออ่านข้อมูลในไฟล์ข้อมูลตัวเลข ตามรูปที่ 3.19



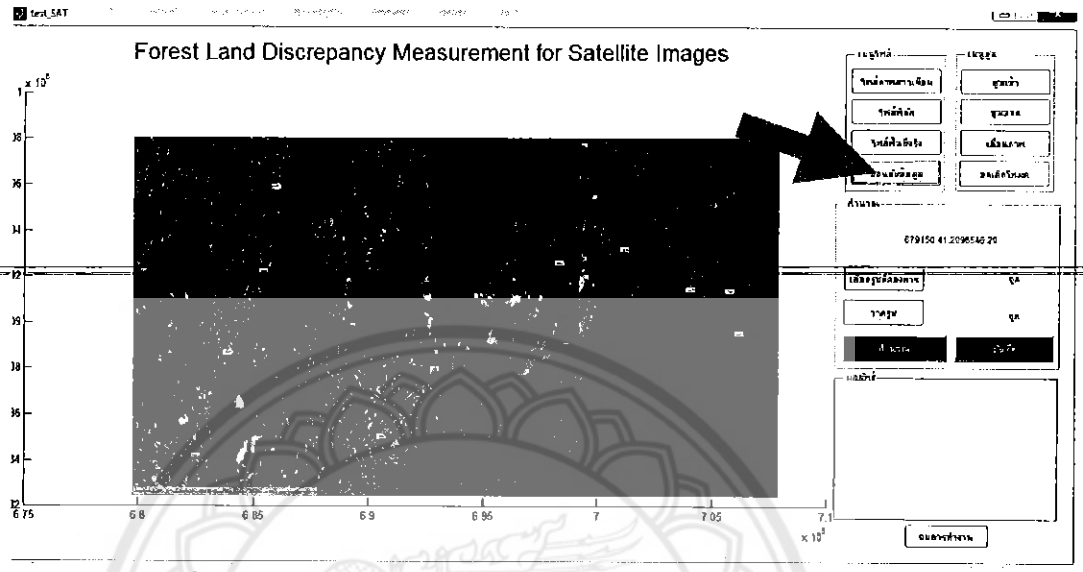
รูปที่ 3.19 แสดงถึงปุ่มใช้ข้อมูลของโปรแกรมในเมนูไฟล์ในส่วนการเลือกไฟล์พิกัด (World File)

การเลือกไฟล์พื้นที่จริง (Shape File) นั้นเมื่อผู้ใช้งานทำการเลือกไฟล์แล้วโปรแกรมจะใช้คำสั่ง shaperead() เพื่ออ่านข้อมูลในรูปแบบรูปหลายเหลี่ยม (Polygon) ตามรูปที่ 3.20

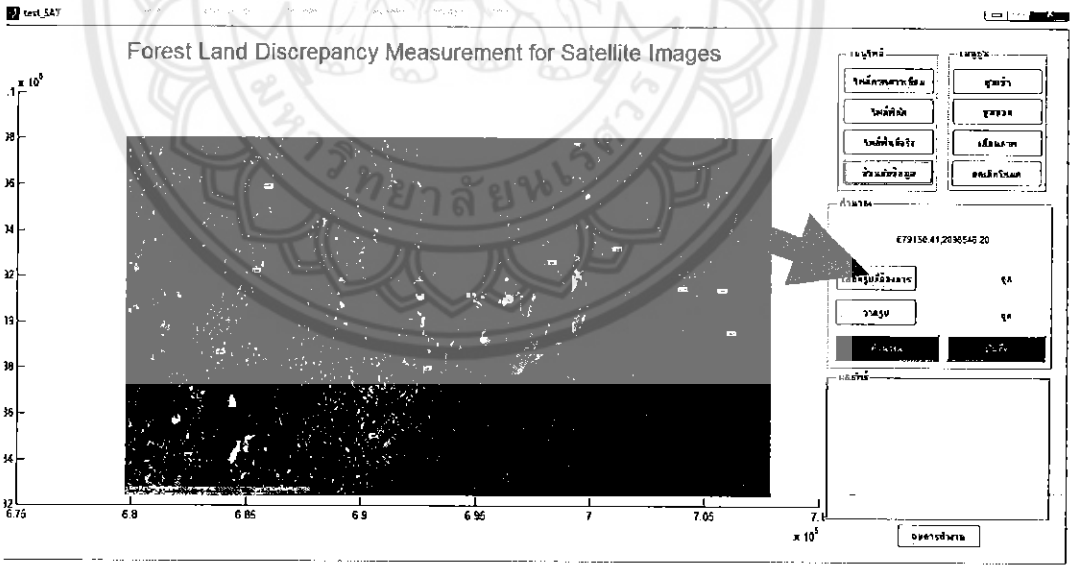


รูปที่ 3.20 แสดงถึงปุ่มใช้ข้อมูลของโปรแกรมในเมนูไฟล์
ในส่วนการเลือกไฟล์พื้นที่จริง (Shape File)

เมื่อทุกไฟล์ได้ข้อมูลครบแล้วทำการกดปุ่มซ้อนทับข้อมูล โปรแกรมจะใช้คำสั่ง mapshow() โดยการเรียกทุกไฟล์ ขึ้นมาซ้อนทับกันซึ่งจะแสดงในรูปที่ 3.21



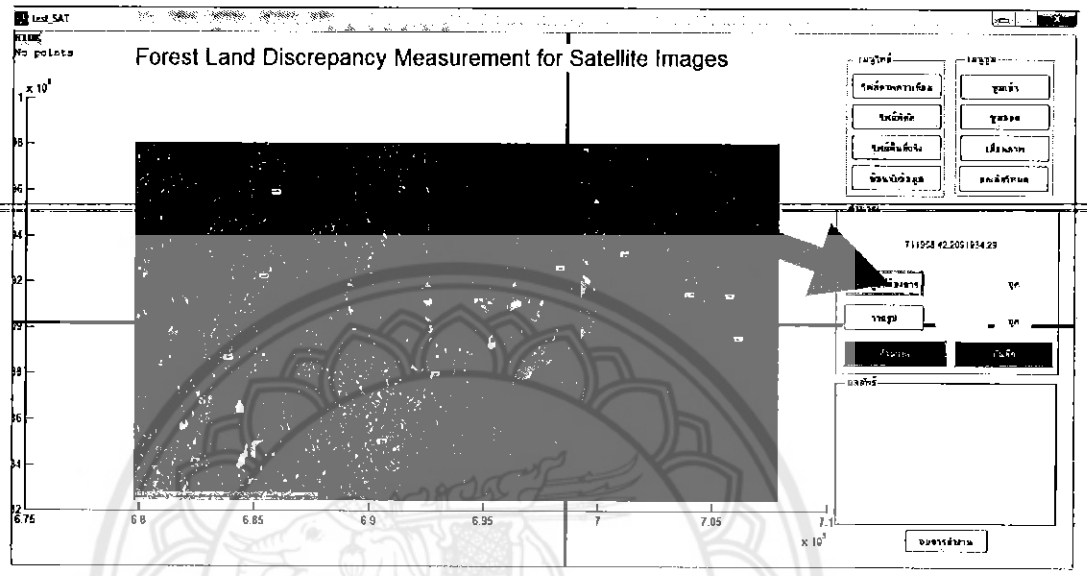
รูปที่ 3.21 แสดงถึงการเรียกใช้ข้อมูลของโปรแกรมเมื่อกดปุ่มซ้อนทับข้อมูล



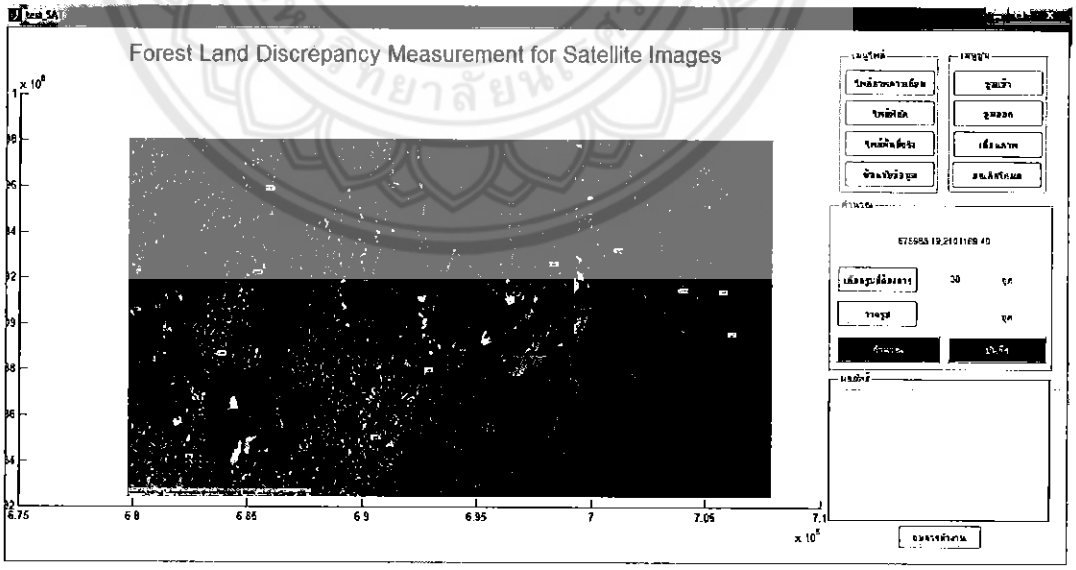
รูปที่ 3.22 แสดงถึงปุ่มเลือกข้อมูลรูปหลายเหลี่ยม (Polygon) สีเขียวบนแผนที่

จากนั้นทำการกดปุ่มเลือกรูปหลายเหลี่ยม (Polygon) แล้วโปรแกรมจะไปเรียกฟังก์ชันในการเลือกจุดเพื่อนำค่าจากจุดไปคำนวณว่าเป็นจุดที่อยู่ในรูปหลายเหลี่ยม (Polygon) หรือไม่ตามหลักการการหาจุดในรูปหลายเหลี่ยม (Point In Polygon) ถ้าจุดนั้นอยู่ในรูปที่รูปหลายเหลี่ยม

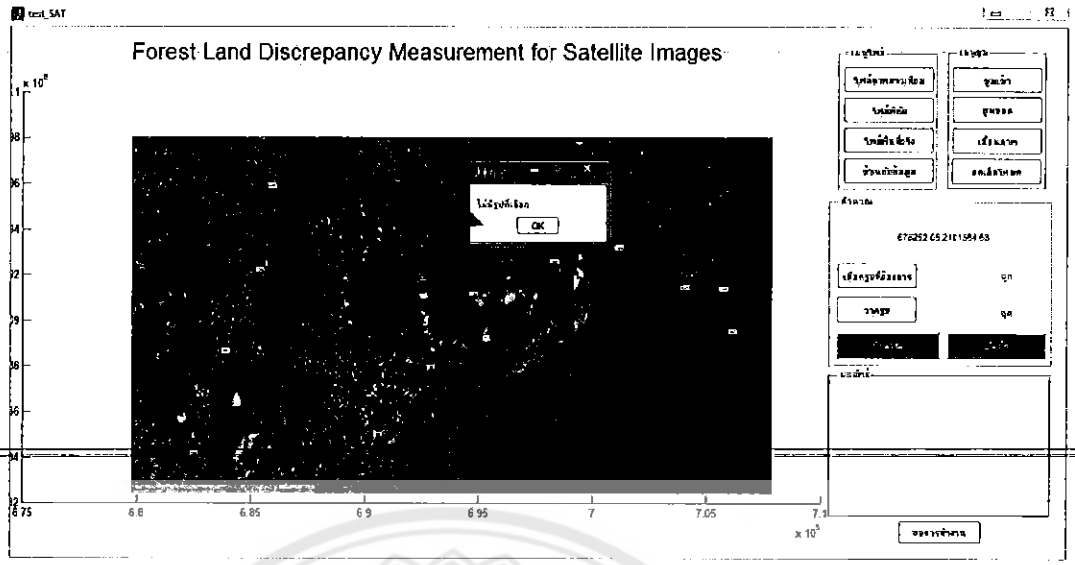
(Polygon) โปรแกรมจะแสดงค่าจุดทุกจุดที่เป็นขอบของรูปหลายเหลี่ยม (Polygon) ออกมาให้เห็นเป็นจุดสีแดง ตามรูปที่ 3.24 แต่ถ้าจุดไม่อยู่ในรูปรูปหลายเหลี่ยม (Polygon) ใดรูปหลายเหลี่ยม (Polygon) หนึ่งโปรแกรมจะแสดง ข้อความบอกว่าไม่มีรูป ที่เลือกตามรูปที่ 3.25



รูปที่ 3.23 แสดงถึงปุ่มเลือกข้อมูลรูปหลายเหลี่ยม (Polygon) สีเขียวบนแผนที่

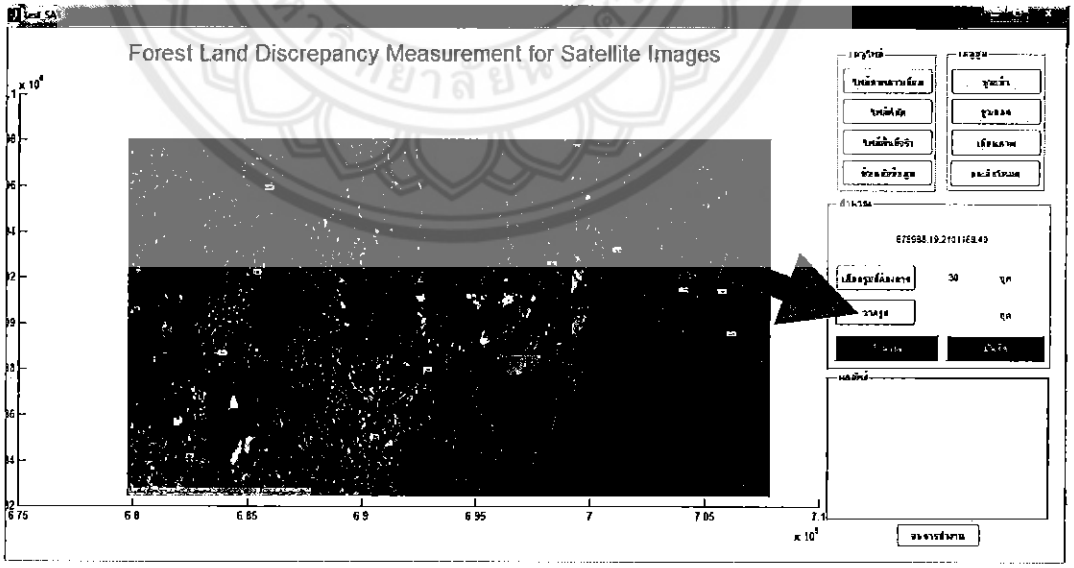


รูปที่ 3.24 แสดงถึงข้อมูลรูปหลายเหลี่ยม (Polygon) สีเขียวและจุดสีแดงบนแผนที่

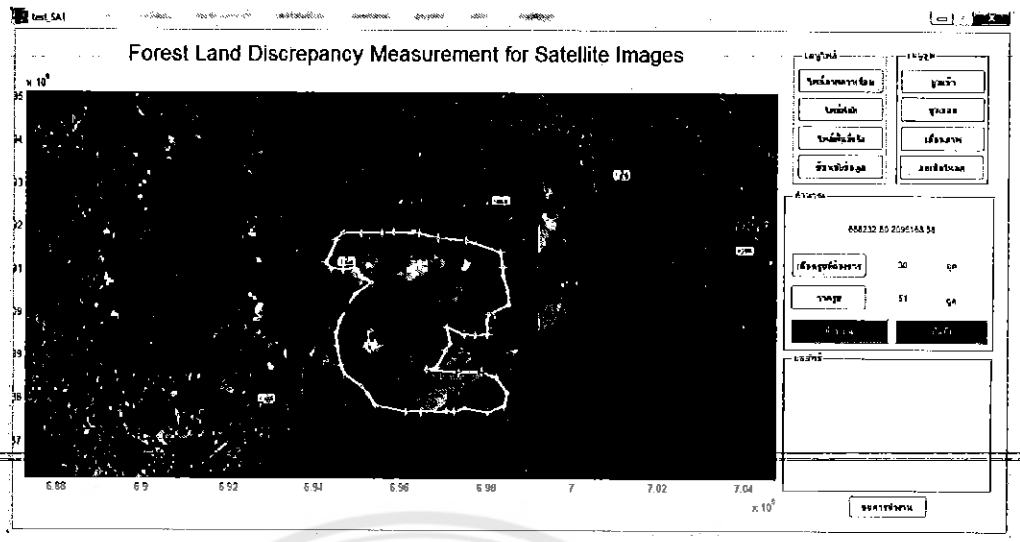


รูปที่ 3.25 แสดงถึงความผิดพลาดเมื่อคลิกจุดในแผนที่ไม่มีรูปที่เลือก

เมื่อเราต้องการรูปหลายเหลี่ยม (Polygon) ใหม่ขึ้นมาเปรียบเทียบให้กดปุ่มวาดรูป โปรแกรมจะให้ผู้ใช้งานเลือกจุดที่ต้องการ เพื่อสร้างรูปหลายเหลี่ยม (Polygon) ใหม่ สามารถเลือกกี่จุดตามต้องการก็ได้ ตามรูปที่ 3.26 และเมื่อเลือกเสร็จแล้วให้กดปุ่ม Enter เพื่อสิ้นสุดการเลือกจุด และโปรแกรมจะแสดงเส้นเชื่อมต่อระหว่างจุด ตามรูปที่ 3.27

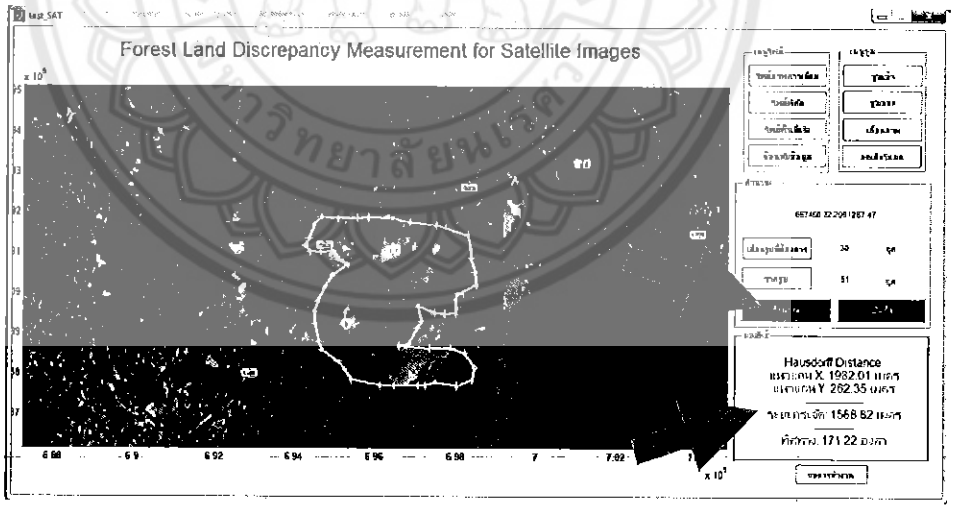


รูปที่ 3.26 แสดงถึงการสร้างข้อมูลรูปหลายเหลี่ยม (Polygon) ใหม่บนแผนที่



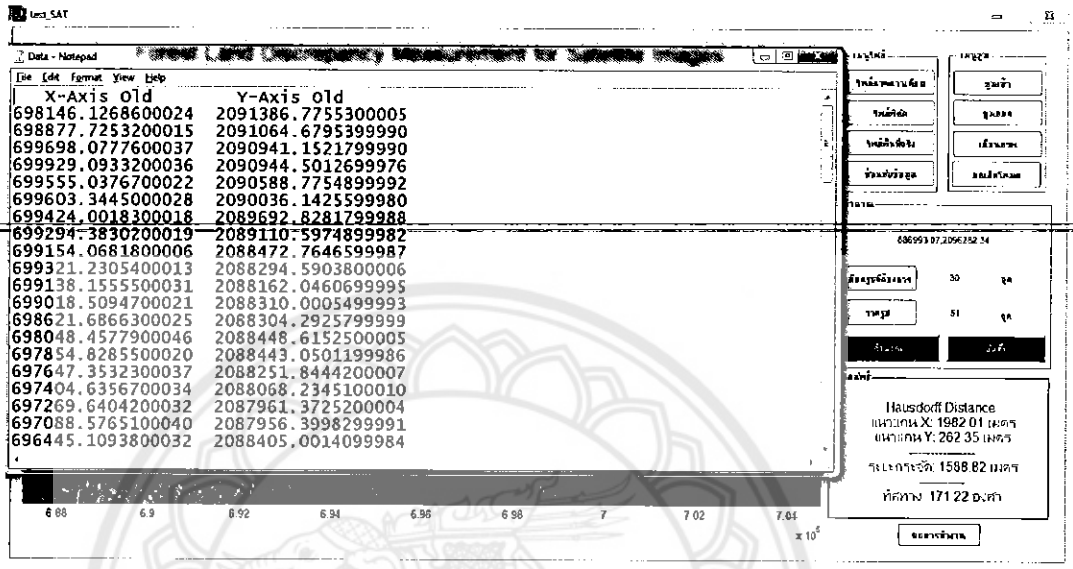
รูปที่ 3.27 แสดงถึงการสร้างข้อมูลรูปหลายเหลี่ยม (Polygon) ใหม่บนแผนที่

เมื่อได้ Polygon เพื่อมาเปรียบเทียบกันแล้ว ผู้ใช้งานสามารถกดปุ่มคำนวณเพื่อคำนวณหาค่าความ โปรแกรมจะแสดงข้อมูลค่าความคลาดเคลื่อนของ Polygon ทั้ง 2 รูป ออกมาในรูปค่าความคลาดเคลื่อนแบบการกระจัด ทิศทาง จากหลักการพีทาโกรัส ค่าความคลาดเคลื่อนในแกน x ค่าความคลาดเคลื่อนในแกน y จากหลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distances)

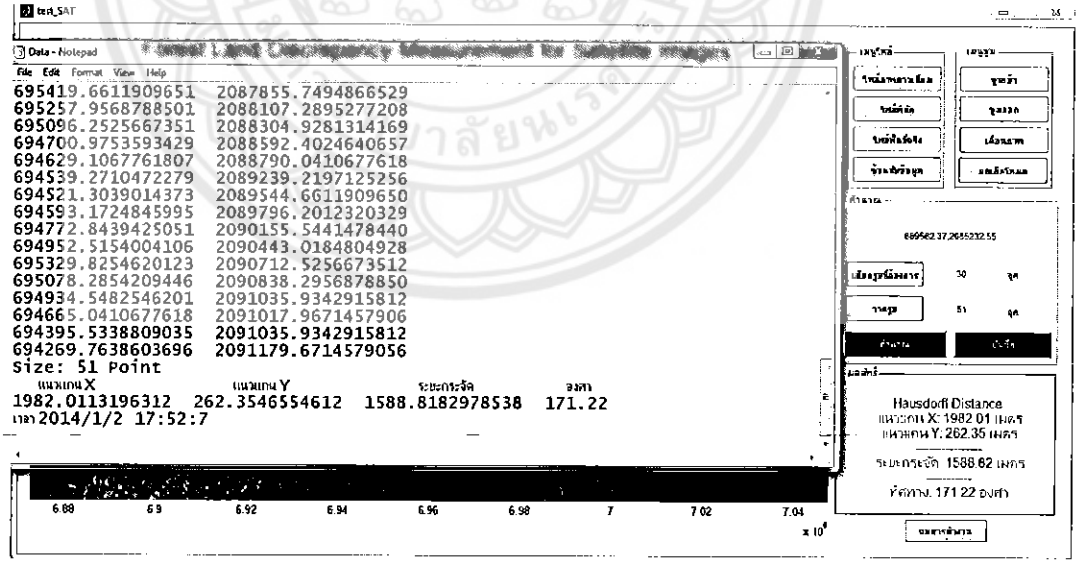


รูปที่ 3.28 แสดงถึงคำนวณค่าความคลาดเคลื่อนของรูปหลายเหลี่ยม (Polygon) บอกค่า ความคลาดเคลื่อนในระยะกระจัด ทิศทาง แกน x และแกน y

เมื่อต้องการบันทึกข้อมูลจากการคำนวณโปรแกรมสามารถบันทึกจุดของรูปหลายเหลี่ยม (Polygon) ทั้งใหม่และเก่าและค่าที่คำนวณ และจำนวนจุดทั้งหมดของแต่ละรูปหลายเหลี่ยม (Polygon) เพื่อให้ผู้ใช้งานสามารถนำค่าไปใช้งานได้ต่อไป ตามรูปที่ 3.30 และรูปที่ 3.31



รูปที่ 3.30 แสดงถึงการบันทึกข้อมูล



รูปที่ 3.31 แสดงถึงการบันทึกข้อมูล

3.5 ความต้องการระบบ (System Requirement)

3.5.1 Hardware requirement specification

1. หน่วยประมวลผลกลาง (CPU)

   Any Intel or AMD x86 processor supporting SSE2 instruction

2. หน่วยความจำหลัก (RAM)

   1024 MB ขึ้นไป

3. หน่วยความจำหลัก (Disk Space)

   1 GB for MATLAB only

   3-4 GB for a typical installation

3.5.2 Software requirement specification

1. Windows XP x64 Edition Service Pack 2 ขึ้นไป
2. MATLAB

บทที่ 4

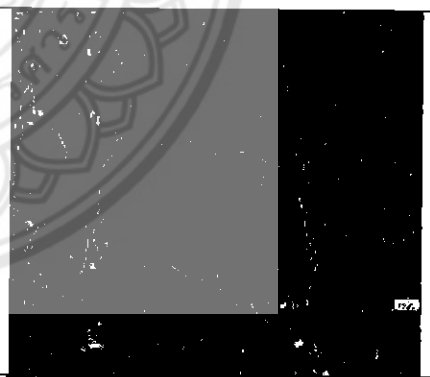
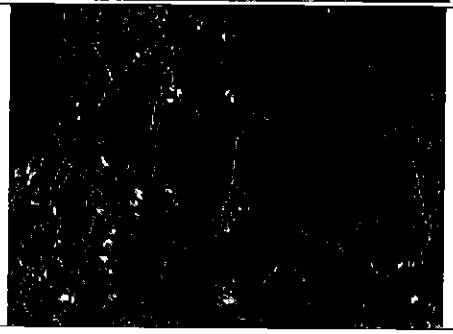
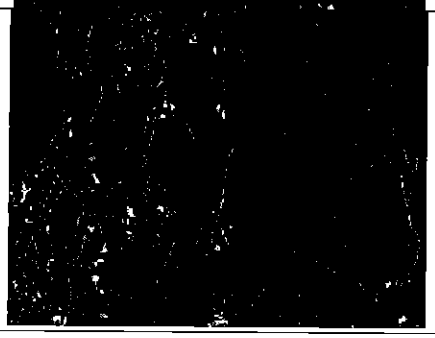
ผลการทดลอง

การทดลองที่ใช้ในการตรวจสอบความถูกต้องของโปรแกรมคือหลักการการหาตำแหน่งของจุดเพื่อเลือกรูปหลายเหลี่ยม (Polygon) หลักการของ การสร้างรูปหลายเหลี่ยม (Polygon) เพื่อเปรียบเทียบและหลักการหาความคลาดเคลื่อนของรูปหลายเหลี่ยม (Polygon) ซึ่งผลการทดลองเป็นดังนี้

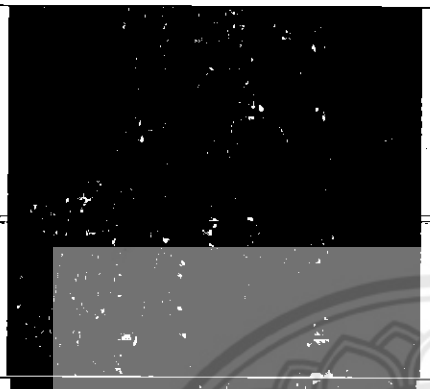
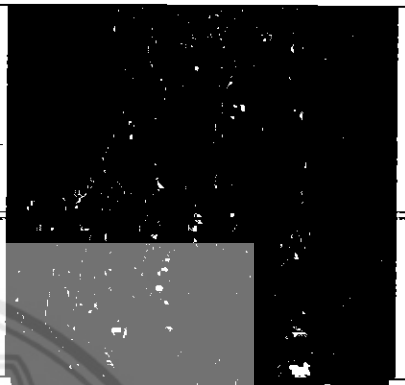
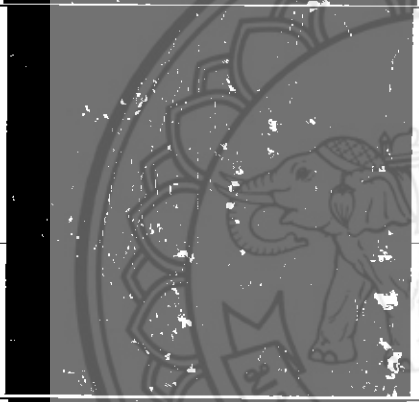
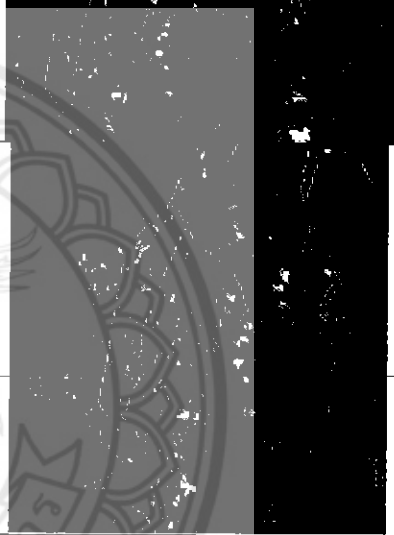
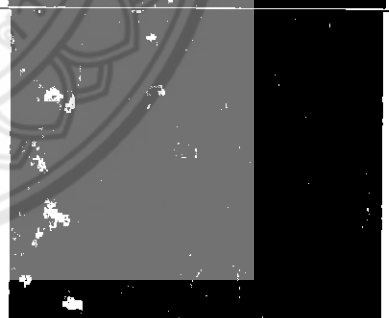
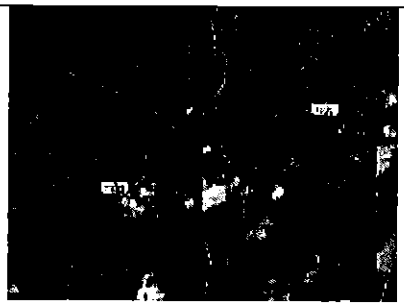
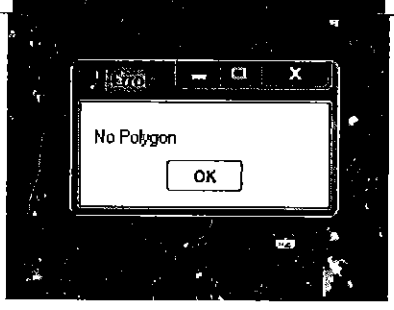
4.1 ทดสอบความถูกต้องในการเลือกข้อมูลรูปหลายเหลี่ยม (Polygon)

ในการเลือกทดสอบเมื่อเลือกจุดใดจุดหนึ่งในขอบเขตของแต่ละรูปหลายเหลี่ยม (Polygon) นั้น โปรแกรมจะต้องแสดงจุดแต่ละจุดของแต่ละรูปหลายเหลี่ยม (Polygon) ตามข้อมูลที่มีได้อย่างถูกต้อง เพื่อที่จะสามารถให้ผู้ใช้งานได้เลือกใช้รูปหลายเหลี่ยม (Polygon) ได้อย่างถูกต้องตามต้องการ แต่ถ้าเลือกจุดใดจุดหนึ่งที่อยู่นอกขอบเขตรูปหลายเหลี่ยม (Polygon) โปรแกรมจะต้องแสดงให้เห็นว่าไม่มีรูปหลายเหลี่ยม (Polygon) ที่เลือก

ตารางที่ 4.1 ตารางตรวจสอบความถูกต้องของผลการเลือกรูปหลายเหลี่ยม (Polygon) ตามลำดับ

ลำดับ	Input	ผลรวมจุดตัด	Output	ผลการทดลอง
1		1		ถูกต้อง
2		3		ถูกต้อง

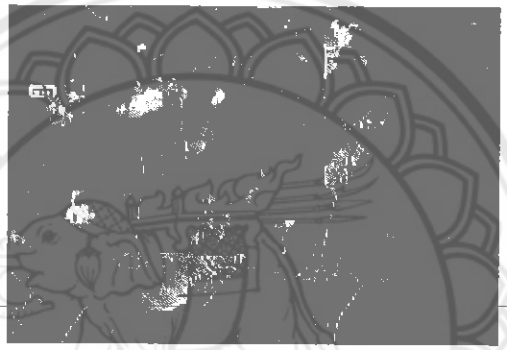
ตารางที่ 4.1 (ต่อ) ตารางตรวจสอบความถูกต้องของผลการเลือกรูปหลายเหลี่ยม (Polygon) ตามลำดับ

ลำดับ	Input	ผลรวมจุดตัด	Output	ผลการทดลอง
3		5		ถูกต้อง
4		7		ถูกต้อง
5		1		ถูกต้อง
6		0		ถูกต้อง

4.2 เปรียบเทียบข้อมูลโดยการสมมุติค่าแกน x และแกน y

ในการเลือกทดสอบเมื่อคลิกจุดใดจุดหนึ่งในขอบเขตของแต่ละรูปหลายเหลี่ยม (Polygon) นั้น โปรแกรมจะต้องแสดงจุดแต่ละจุดของแต่ละรูปหลายเหลี่ยม (Polygon) ตามข้อมูลที่มีได้ และสร้างรูปหลายเหลี่ยม (Polygon) ขึ้นมาเพื่อเปรียบเทียบกับรูปหลายเหลี่ยม (Polygon) ที่เลือกไว้ และตรวจสอบระยะกระจัด ทิศทาง และความคลาดเคลื่อนตามแกน x และแกน y การทดลองใช้การสมมุติค่าขึ้นมาทดสอบความถูกต้องการคำนวณของโปรแกรม

4.2.1 การสมมุติให้รูปหลายเหลี่ยม (Polygon) สีน้ำเงินเพิ่มค่าแกน x = +1000 เมตร และ y = +0 เมตร



รูปที่ 4.1 แสดงค่าความคลาดเคลื่อนโดยการสมมุติค่าของรูปหลายเหลี่ยม (Polygon) **สีน้ำเงินใน** แกน x = +1000 เมตร และแกน y = +0 เมตร ซึ่งแสดงค่าในตารางที่ 4.2

ตารางที่ 4.2 ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) **สีน้ำเงิน** โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = +1000 เมตร และแกน y = +0 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)		รูปหลายเหลี่ยม (Polygon) **สีน้ำเงิน** (x , y)	
698146.1268600024	2091386.7755300005	699146.1268600024	2091386.7755300005
698877.7253200015	2091064.6795399990	699877.7253200015	2091064.6795399990
699698.0777600037	2090941.1521799990	700698.0777600037	2090941.1521799990
699929.0933200036	2090944.5012699976	700929.0933200036	2090944.5012699976
699555.0376700022	2090588.7754899992	700555.0376700022	2090588.7754899992
699603.3445000028	2090036.1425599980	700603.3445000028	2090036.1425599980
699424.0018300018	2089692.8281799988	700424.0018300018	2089692.8281799988
699294.3830200019	2089110.5974899982	700294.3830200019	2089110.5974899982

ตารางที่ 4.2 (ต่อ) ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = +1000 เมตร และแกน y = +0 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
699154.0681800006 2088472.7646599987	700154.0681800006 2088472.7646599987
699321.2305400013 2088294.5903800006	700321.2305400013 2088294.5903800006
699138.1555500031 2088162.0460699995	700138.1555500031 2088162.0460699995
699018.5094700021 2088310.0005499993	700018.5094700021 2088310.0005499993
698621.6866300025 2088304.2925799999	699621.6866300025 2088304.2925799999
698048.4577900046 2088448.6152500005	699048.4577900046 2088448.6152500005
697854.8285500020 2088443.0501199986	698854.8285500020 2088443.0501199986
697647.3532300037 2088251.8444200007	698647.3532300037 2088251.8444200007
697404.6356700034 2088068.2345100010	698404.6356700034 2088068.2345100010
697269.6404200032 2087961.3725200004	698269.6404200032 2087961.3725200004
697088.5765100040 2087956.3998299991	698088.5765100040 2087956.3998299991
696445.1093800032 2088405.0014099984	697445.1093800032 2088405.0014099984
696262.6002900035 2088712.7733499985	697262.6002900035 2088712.7733499985
696251.7751800008 2089087.8150699993	697251.7751800008 2089087.8150699993
696382.8663000024 2089866.2745999990	697382.8663000024 2089866.2745999990
696275.7912700028 2090250.3058699979	697275.7912700028 2090250.3058699979
696605.2028300014 2090270.5928099989	697605.2028300014 2090270.5928099989
696795.5330200032 2090316.4501799997	697795.5330200032 2090316.4501799997
696969.3110000026 2090236.9687200005	697969.3110000026 2090236.9687200005
696897.6310200024 2091340.7701900003	697897.6310200024 2091340.7701900003
697235.5291000061 2091640.5005899996	698235.5291000061 2091640.5005899996
698146.1268600024 2091386.7755300005	699146.1268600024 2091386.7755300005

จากตารางที่ 4.2 เมื่อนำค่าแกน x และแกน y ของทั้ง 2 รูปหลายเหลี่ยม (Polygon) มาคำนวณโดยใช้หลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) ค่าที่ได้ตรงกับค่าที่สมมุติขึ้นโดยระยะทางแกน x = 1000 เมตร แกน y = 0 เมตร ระยะกระจัด 1000 เมตร และทิศทาง 0 องศา โดยแสดงผลลัพธ์ในรูปที่ 4.2

ผลลัพธ์

Hausdorff Distance
แนวแกน X: 1000.00 เมตร
แนวแกน Y: 0.00 เมตร

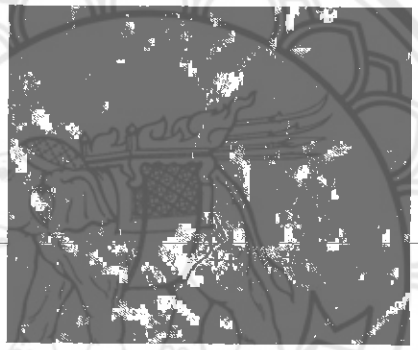
ระยะกระจัด: 1000.00 เมตร

ทิศทาง: 0.00 องศา

รูปที่ 4.2 แสดงผลลัพธ์ในการคำนวณโดยการสมมุติค่า

แกน x = 1000 เมตร แกน y = 0 เมตร ระยะกระจัด = 1000 เมตร และทิศทางมีค่า 0 องศา

4.2.2 การสมมุติให้รูปหลายเหลี่ยม (Polygon) สีน้ำเงินเพิ่มค่าแกน x = -500 เมตร และ y = +0 เมตร



รูปที่ 4.3 แสดงค่าความคลาดเคลื่อนโดยการสมมุติให้ค่าของรูปหลายเหลี่ยม (Polygon) สีน้ำเงินใน แกน x = -500 เมตร และแกน y = +0 เมตร ซึ่งแสดงค่าในตารางที่ 4.2

ตารางที่ 4.3 ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = -500 เมตร และแกน y = +0 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
691082.3372900027 2087970.2589899998	690582.3372900027 2087970.2589899998
691072.3372900025 2087888.2589900012	690572.3372900025 2087888.2589900012
691070.3372900041 2087823.2589899986	690570.3372900041 2087823.2589899986
691243.3372900030 2087553.2589900005	690743.3372900030 2087553.2589900005
691234.3372900034 2087526.2589899981	690734.3372900034 2087526.2589899981
691244.3372900034 2087484.2589899995	690744.3372900034 2087484.2589899995

ตารางที่ 4.3 (ต่อ) ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = -500 เมตร และแกน y = +0 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
691269.3372900026 2087465.2589899984	690769.3372900026 2087465.2589899984
691236.3372900023 2087400.2589900007	690736.3372900023 2087400.2589900007
691135.5542200023 2087336.1006399996	690635.5542200023 2087336.1006399996
691218.6279100040 2087178.0095599997	690718.6279100040 2087178.0095599997
691158.9620000036 2087143.5258099998	690658.9620000036 2087143.5258099998
691100.8123400026 2087072.9010999994	690600.8123400026 2087072.9010999994
691036.7125700046 2087013.9010999990	690536.7125700046 2087013.9010999990
690997.9208100006 2086968.0095600008	690497.9208100006 2086968.0095600008
690975.0704700012 2086892.3848499993	690475.0704700012 2086892.3848499993
690924.0118900021 2086848.9097899981	690424.0118900021 2086848.9097899981
690832.3372900023 2087001.2589900012	690332.3372900023 2087001.2589900012
690751.3372900016 2087039.2589900002	690251.3372900016 2087039.2589900002
690668.3372900022 2087051.2589899988	690168.3372900022 2087051.2589899988
690630.3372900026 2087138.2589899993	690130.3372900026 2087138.2589899993
690597.3372900018 2087263.2589899988	690097.3372900018 2087263.2589899988
690638.3372900022 2087343.2589899991	690138.3372900022 2087343.2589899991
690689.3372900023 2087460.2589899977	690189.3372900023 2087460.2589899977
690561.3372900020 2087566.2589899991	690061.3372900020 2087566.2589899991
690580.3372900041 2087583.2589899995	690080.3372900041 2087583.2589899995
690508.3372900023 2087634.2589899977	690008.3372900023 2087634.2589899977
690542.3372900046 2087754.2589899974	690042.3372900046 2087754.2589899974
690572.3372900034 2087820.2589900012	690072.3372900034 2087820.2589900012
690603.3372900044 2087927.2589900005	690103.3372900044 2087927.2589900005
690646.3372900017 2088017.2589899995	690146.3372900017 2088017.2589899995
690763.3372900005 2087976.2589899981	690263.3372900005 2087976.2589899981
690799.3372900029 2088043.2589900000	690299.3372900029 2088043.2589900000
690869.3372900038 2088038.2589899995	690369.3372900038 2088038.2589899995
690946.3372900024 2088040.2589899986	690446.3372900024 2088040.2589899986

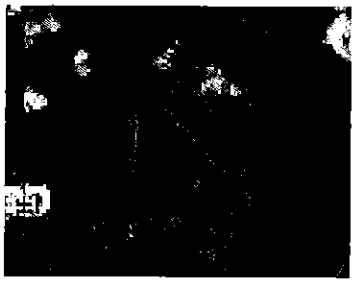
ตารางที่ 4.3 (ต่อ) ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = -500 เมตร และแกน y = +0 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
690965.3372900019 2088065.2589899991	690465.3372900019 2088065.2589899991
690978.3372900009 2088081.2589899981	690478.3372900009 2088081.2589899981
691082.3372900027 2087970.2589899998	690582.3372900027 2087970.2589899998

จากตารางที่ 4.3 เมื่อนำค่าแกน x และแกน y มาคำนวณโดยใช้หลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) ค่าที่ได้ตรงกับค่าที่สมมุติขึ้นโดยระยะทางแกน x = 500 เมตร แกน y = 0 เมตร ระยะกระจัด 500 เมตร และทิศทางมีค่า 180 องศา โดยแสดงผลลัพธ์ในรูปที่ 4.4

รูปที่ 4.4 แสดงผลลัพธ์ในการคำนวณโดยการสมมุติค่า แกน x = 500 เมตร แกน y = 0 เมตร ระยะกระจัด = 500 เมตร และทิศทางมีค่า 180 องศา

4.2.3 การสมมุติให้รูปหลายเหลี่ยม (Polygon) สีน้ำเงินเพิ่มค่า แกน x = +0 เมตร และ y = +400 เมตร



รูปที่ 4.5 แสดงค่าความคลาดเคลื่อนโดยการสมมุติให้ค่ารูปหลายเหลี่ยม (Polygon) สีน้ำเงินใน แกน x = +0 เมตร และแกน y = +400 เมตรซึ่งแสดงค่าในตารางที่ 4.4

ตารางที่ 4.4 ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = +0 เมตร และแกน y = +400 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
693976.2202600043 2088669.2863199983	693976.2202600043 2089069.2863199983
694104.2202600037 2088545.2863200007	694104.2202600037 2088945.2863200007
694190.2202600020 2088436.2863199981	694190.2202600020 2088836.2863199981
694252.2202600039 2088371.2863199995	694252.2202600039 2088771.2863199995
694298.2202600043 2088301.2863199976	694298.2202600043 2088701.2863199976
694408.2202600020 2088218.2863200002	694408.2202600020 2088618.2863200002
694537.2202600031 2088103.2863199981	694537.2202600031 2088503.2863199981
694597.2202600023 2088011.2863199997	694597.2202600023 2088411.2863199997
694427.2202600022 2087721.2863200023	694427.2202600022 2088121.2863200023
694317.2202600031 2087788.2863200000	694317.2202600031 2088188.2863200000
694234.2202600020 2087854.2863200009	694234.2202600020 2088254.2863200009
694155.2202600037 2087865.2863199981	694155.2202600037 2088265.2863199981
694028.2202600008 2087841.2863199972	694028.2202600008 2088241.2863199972
694008.2202600024 2087803.2863200007	694008.2202600024 2088203.2863200007
693975.2202600029 2087826.2863199997	693975.2202600029 2088226.2863199997
693885.2202600041 2087857.2863200000	693885.2202600041 2088257.2863200000
693750.2202600017 2087807.2863200004	693750.2202600017 2088207.2863200004
693712.2202600024 2087853.2863199988	693712.2202600024 2088253.2863199988
693721.2202600025 2088701.2863199995	693721.2202600025 2089101.2863199995
693834.2202600027 2088719.2863199995	693834.2202600027 2089119.2863199995
693976.2202600043 2088669.2863199983	693976.2202600043 2089069.2863199983

จากตารางที่ 4.4 เมื่อนำค่าแกน x และแกน y มาคำนวณโดยใช้หลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) ค่าที่ได้ตรงกับค่าที่สมมุติขึ้นโดยระยะทางแกน x = 0 เมตร แกน y = 400 เมตร ระยะกระจัด 400 เมตร และทิศทางมีค่า 90 องศา โดยแสดงผลลัพธ์ในรูปที่ 4.6

ผลลัพธ์

Hausdorff Distance
แนวแกน X: 0.00 เมตร
แนวแกน Y: 400.00 เมตร

ระยะกระจัด: 400.00 เมตร

ทิศทาง: 90.00 องศา

รูปที่ 4.6 แสดงผลลัพธ์ในการคำนวณโดยการสมมุติค่า

แกน x = 0 เมตร แกน y = 400 เมตร ระยะกระจัด = 400 เมตร และทิศทางมีค่า 90 องศา

4.2.4 การสมมุติให้รูปหลายเหลี่ยม (Polygon) น้ำเงินเพิ่มค่า

แกน x = +0 เมตร และ y = -300 เมตร

รูปที่ 4.7 แสดงค่าความคลาดเคลื่อนโดยการสมมุติให้ค่าของรูปหลายเหลี่ยม (Polygon) สีน้ำเงินใน แกน x = +0 เมตร และแกน y = -300 เมตรซึ่งแสดงค่าในตารางที่ 4.5

ตารางที่ 4.5 ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = +0 เมตร และแกน y = -300 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
691858.0924100017 2089568.1506299993	691858.0924100017 2089268.1506299993
691875.0924100044 2089577.1506299996	691875.0924100044 2089277.1506299996
691909.0924100028 2089569.1506299987	691909.0924100028 2089269.1506299987
691950.0924100027 2089554.1506299982	691950.0924100027 2089254.1506299982
691967.0924100041 2089497.1506299993	691967.0924100041 2089197.1506299993
691967.0924100034 2089338.1506300003	691967.0924100034 2089038.1506300003
691971.0924100035 2089276.1506299993	691971.0924100035 2088976.1506299993
691884.0924100030 2089304.1506299973	691884.0924100030 2089004.1506299973
691814.0924100012 2089307.1506299975	691814.0924100012 2089007.1506299975
691762.0924100014 2089283.1506299982	691762.0924100014 2088983.1506299982
691728.0924100034 2089327.1506299996	691728.0924100034 2089027.1506299996
691696.0924100024 2089294.1506299993	691696.0924100024 2088994.1506299993
691600.0924100033 2089286.1506299980	691600.0924100033 2088986.1506299980
691527.0924100025 2089314.1506299984	691527.0924100025 2089014.1506299984
691418.6974700025 2089302.1594899991	691418.6974700025 2089002.1594899991
691503.0924100047 2089403.1506300000	691503.0924100047 2089103.1506300000
691520.0924100027 2089466.1506300010	691520.0924100027 2089166.1506300010
691527.0924100033 2089491.1506299975	691527.0924100033 2089191.1506299975
691610.0924100013 2089524.1506299993	691610.0924100013 2089224.1506299993
691725.0924100038 2089531.1506299993	691725.0924100038 2089231.1506299993
691806.0924100023 2089549.1506299989	691806.0924100023 2089249.1506299989
691778.0924100027 2089589.1506299996	691778.0924100027 2089289.1506299996
691858.0924100017 2089568.1506299993	691858.0924100017 2089268.1506299993

จากตารางที่ 4.5 เมื่อนำค่าแกน x และแกน y มาคำนวณโดยใช้หลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) ค่าที่ได้ตรงกับค่าที่สมมุติขึ้นโดยระยะทางแกน x = 0 เมตร แกน y = 300 เมตร ระยะกระจัด 300 เมตร และทิศทางมีค่า 270 องศา โดยแสดงผลลัพธ์ในรูปที่ 4.8

ผลลัพธ์

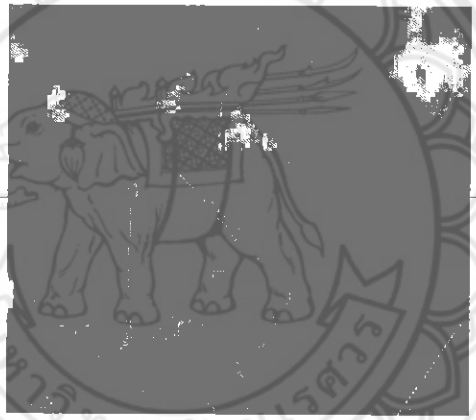
Hausdorff Distance
แนวแกน X: 0.00 เมตร
แนวแกน Y: 300.00 เมตร

ระยะกระจัด: 300.00 เมตร

ทิศทาง: 270.00 องศา

รูปที่ 4.8 แสดงผลลัพธ์ในการคำนวณโดยการสมมุติค่า
แกน x = 0 เมตร แกน y = 300 เมตร ระยะกระจัด = 300 เมตร และทิศทางมีค่า 270 องศา

4.2.5 การสมมุติให้รูปหลายเหลี่ยม (Polygon) สีน้ำเงินเพิ่มค่า
แกน x = +500 เมตร และ y = +500 เมตร



รูปที่ 4.9 แสดงค่าความคลาดเคลื่อนโดยการสมมุติให้ค่าของรูปหลายเหลี่ยม (Polygon) สีน้ำเงินใน
แกน x = +500 เมตร และแกน y = +500 เมตรซึ่งแสดงค่าในตารางที่ 4.6

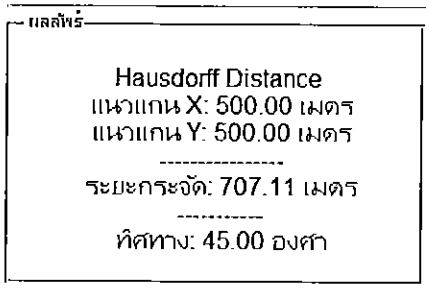
ตารางที่ 4.6 ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน
โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = +500 เมตร และแกน y = +500 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
693976.2202600043 2088669.2863199983	694476.2202600043 2089169.2863199983
694104.2202600037 2088545.2863200007	694604.2202600037 2089045.2863200007
694190.2202600020 2088436.2863199981	694690.2202600020 2088936.2863199981
694252.2202600039 2088371.2863199995	694752.2202600039 2088871.2863199995
694298.2202600043 2088301.2863199976	694798.2202600043 2088801.2863199976
694408.2202600020 2088218.2863200002	694908.2202600020 2088718.2863200002

ตารางที่ 4.6 (ต่อ) ตารางแสดงรูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = +500 เมตร และแกน y = +500 เมตร

รูปหลายเหลี่ยม (Polygon)สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon)สีน้ำเงิน (x , y)
694537.2202600031 2088103.2863199981	695037.2202600031 2088603.2863199981
694597.2202600023 2088011.2863199997	695097.2202600023 2088511.2863199997
694427.2202600022 2087721.2863200023	694927.2202600022 2088221.2863200023
694317.2202600031 2087788.2863200000	694817.2202600031 2088288.2863200000
694234.2202600020 2087854.2863200009	694734.2202600020 2088354.2863200009
694155.2202600037 2087865.2863199981	694655.2202600037 2088365.2863199981
694028.2202600008 2087841.2863199972	694528.2202600008 2088341.2863199972
694008.2202600024 2087803.2863200007	694508.2202600024 2088303.2863200007
693975.2202600029 2087826.2863199997	694475.2202600029 2088326.2863199997
693885.2202600041 2087857.2863200000	694385.2202600041 2088357.2863200000
693750.2202600017 2087807.2863200004	694250.2202600017 2088307.2863200004
693712.2202600024 2087853.2863199988	694212.2202600024 2088353.2863199988
693721.2202600025 2088701.2863199995	694221.2202600025 2089201.2863199995
693834.2202600027 2088719.2863199995	694334.2202600027 2089219.2863199995
693976.2202600043 2088669.2863199983	694476.2202600043 2089169.2863199983

จากตารางที่ 4.6 เมื่อนำค่าแกน x และแกน y มาคำนวณโดยใช้หลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) ค่าที่ได้ตรงกับค่าที่สมมุติขึ้นโดยระยะทางแกน x = 500 เมตร แกน y = 500 เมตร ระยะกระจัด 707.11 เมตร และทิศทางมีค่า 45 องศา โดยแสดงผลลัพธ์ในรูปที่ 4.10



รูปที่ 4.10 แสดงผลลัพธ์ในการคำนวณโดยการสมมุติค่า
แกน x = 500 เมตร แกน y = 500 เมตร ระยะกระจัด = 707.11 เมตร และทิศทางมีค่า 45 องศา

4.2.6 การสมมุติให้รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน เพิ่มค่าแกน x = -300 เมตร และ y = 500 เมตร

รูปที่ 4.11 แสดงค่าความคลาดเคลื่อนโดยการสมมุติให้ค่าของรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน ในแกน x = -300 เมตร และแกน y = +500 เมตรซึ่งแสดงค่าในตารางที่ 4.7

ตารางที่ 4.7 ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และ รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = -300 เมตร และแกน y = +500 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
690579.0000000036 2090382.9999999988	690279.0000000036 2090882.9999999988
690684.0000000018 2090360.0000000009	690384.0000000018 2090860.0000000009
690753.0000000022 2090337.0000000009	690453.0000000022 2090837.0000000009
690883.0000000024 2090263.9999999995	690583.0000000024 2090763.9999999995
690867.0000000026 2090196.0000000012	690567.0000000026 2090696.0000000012
690879.0000000033 2090112.0000000009	690579.0000000033 2090612.0000000009
690889.0000000018 2090033.0000000005	690589.0000000018 2090533.0000000005
690887.0000000015 2089984.9999999979	690587.0000000015 2090484.9999999979
690880.0000000030 2089928.9999999993	690580.0000000030 2090428.9999999993
690806.0000000028 2089948.9999999986	690506.0000000028 2090448.9999999986
690730.0000000037 2089960.0000000000	690430.0000000037 2090460.0000000000
690693.0000000033 2089978.9999999995	690393.0000000033 2090478.9999999995
690634.0000000042 2089995.9999999998	690334.0000000042 2090495.9999999998
690539.0000000019 2090064.0000000007	690239.0000000019 2090564.0000000007
690507.0000000022 2090079.0000000009	690207.0000000022 2090579.0000000009
690417.0000000034 2090089.0000000000	690117.0000000034 2090589.0000000000
690469.0000000033 2090188.9999999995	690169.0000000033 2090688.9999999995

ตารางที่ 4.7 (ต่อ) ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = -300 เมตร และแกน y = +500 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
690533.0000000014 2090291.0000000007	690233.0000000014 2090791.0000000007
690505.0000000023 2090383.9999999986	690205.0000000023 2090883.9999999986
690579.0000000036 2090382.9999999988	690279.0000000036 2090882.9999999988

จากตารางที่ 4.7 เมื่อนำค่าแกน x และแกน y มาคำนวณโดยใช้หลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) ค่าที่ได้ตรงกับค่าที่สมมุติขึ้นโดยระยะทางแกน x = 300 เมตร แกน y = 500 เมตร ระยะกระจัด 583.10 เมตร และทิศทางมีค่า 120.96 องศา โดยแสดงผลลัพธ์ในรูปที่ 4.12

รูปที่ 4.12 แสดงผลลัพธ์ในการคำนวณโดยการสมมุติค่าแกน x = 300 เมตร แกน y = 500 เมตร ระยะกระจัด = 583.10 เมตร และทิศทางมีค่า 120.96 องศา

4.2.7 การสมมุติให้รูปหลายเหลี่ยม (Polygon) สีน้ำเงินเพิ่มค่าแกน x = -800 เมตร และ y = -400 เมตร

รูปที่ 4.13 แสดงค่าความคลาดเคลื่อนโดยการสมมุติให้ค่าของรูปหลายเหลี่ยม (Polygon) สีน้ำเงินในแกน x = -800 เมตร และแกน y = -400 เมตรซึ่งแสดงค่าในตารางที่ 4.8

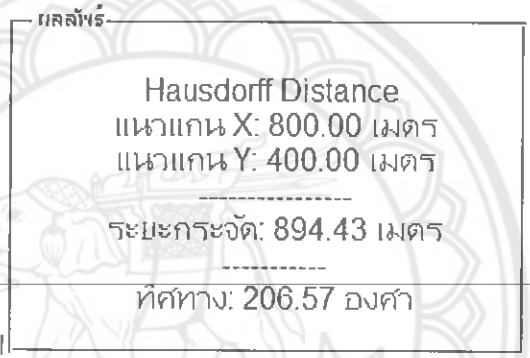
ตารางที่ 4.8 ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = -800 เมตร และแกน y = -400 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
698146.1268600024 2091386.7755300005	697346.1268600024 2090986.7755300005
698877.7253200015 2091064.6795399990	698077.7253200015 2090664.6795399990
699698.0777600037 2090941.1521799990	698898.0777600037 2090541.1521799990
699929.0933200036 2090944.5012699976	699129.0933200036 2090544.5012699976
699555.0376700022 2090588.7754899992	698755.0376700022 2090188.7754899992
699603.3445000028 2090036.1425599980	698803.3445000028 2089636.1425599980
699424.0018300018 2089692.8281799988	698624.0018300018 2089292.8281799988
699294.3830200019 2089110.5974899982	698494.3830200019 2088710.5974899982
699154.0681800006 2088472.7646599987	698354.0681800006 2088072.7646599987
699321.2305400013 2088294.5903800006	698521.2305400013 2087894.5903800006
699138.1555500031 2088162.0460699995	698338.1555500031 2087762.0460699995
699018.5094700021 2088310.0005499993	698218.5094700021 2087910.0005499993
698621.6866300025 2088304.2925799999	697821.6866300025 2087904.2925799999
698048.4577900046 2088448.6152500005	697248.4577900046 2088048.6152500005
697854.8285500020 2088443.0501199986	697054.8285500020 2088043.0501199986
697647.3532300037 2088251.8444200007	696847.3532300037 2087851.8444200007
697404.6356700034 2088068.2345100010	696604.6356700034 2087668.2345100010
697269.6404200032 2087961.3725200004	696469.6404200032 2087561.3725200004
697088.5765100040 2087956.3998299991	696288.5765100040 2087556.3998299991
696445.1093800032 2088405.0014099984	695645.1093800032 2088005.0014099984
696262.6002900035 2088712.7733499985	695462.6002900035 2088312.7733499985
696251.7751800008 2089087.8150699993	695451.7751800008 2088687.8150699993
696382.8663000024 2089866.2745999990	695582.8663000024 2089466.2745999990
696275.7912700028 2090250.3058699979	695475.7912700028 2089850.3058699979
696605.2028300014 2090270.5928099989	695805.2028300014 2089870.5928099989
696795.5330200032 2090316.4501799997	695995.5330200032 2089916.4501799997
696969.3110000026 2090236.9687200005	696169.3110000026 2089836.9687200005
696897.6310200024 2091340.7701900003	696097.6310200024 2090940.7701900003

ตารางที่ 4.8 (ต่อ) ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = -800 เมตร และแกน y = -400 เมตร

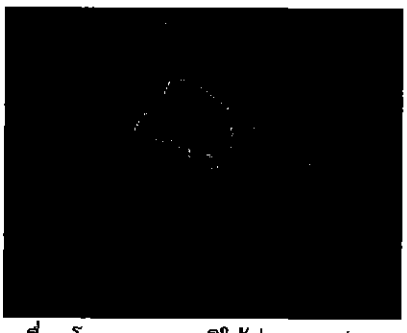
รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
697235.5291000061 2091640.5005899996	696435.5291000061 2091240.5005899996
698146.1268600024 2091386.7755300005	697346.1268600024 2090986.7755300005

จากตารางที่ 4.8 เมื่อนำค่าแกน x และแกน y มาคำนวณโดยใช้หลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) ค่าที่ได้ตรงกับค่าที่สมมุติขึ้นโดยระยะทางแกน x = 800 เมตร แกน y = 400 เมตร ระยะกระจัด 894.43 เมตร และทิศทางมีค่า 206.57 องศา โดยแสดงผลลัพธ์ในรูปที่ 4.13



รูปที่ 4.14 แสดงผลลัพธ์ในการคำนวณโดยการสมมุติค่า แกน x = 800 เมตร แกน y = 400 เมตร ระยะกระจัด = 894.43 เมตร และทิศทางมีค่า 206.57 องศา

4.2.8 การสมมุติให้รูปหลายเหลี่ยม (Polygon) สีน้ำเงินเพิ่มค่าแกน x= +100 เมตร และ y = -200 เมตร

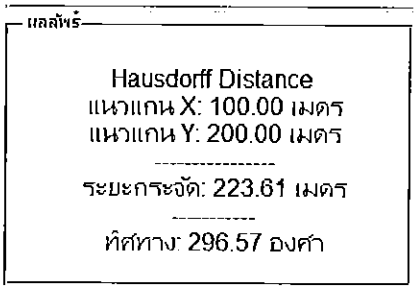


รูปที่ 4.15 แสดงค่าความคลาดเคลื่อนโดยการสมมุติให้ค่าของรูปหลายเหลี่ยม (Polygon) สีน้ำเงินในแกน x = +100 เมตร และแกน y = -200 เมตรซึ่งแสดงค่าในตารางที่ 4.9

ตารางที่ 4.9 ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และรูปหลายเหลี่ยม (Polygon) สีน้ำเงิน โดยการสมมุติค่าความคลาดเคลื่อนโดยกำหนดค่าแกน x = +100 เมตร และแกน y = -200 เมตร

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีน้ำเงิน (x , y)
690359.0000000019 2091441.9999999977	690459.0000000019 2091241.9999999977
690397.0000000029 2091422.9999999986	690497.0000000029 2091222.9999999986
690462.0000000035 2091375.0000000007	690562.0000000035 2091175.0000000007
690462.0000000042 2091361.9999999984	690562.0000000042 2091161.9999999984
690465.0000000030 2091296.9999999984	690565.0000000030 2091096.9999999984
690423.0000000020 2091267.0000000002	690523.0000000020 2091067.0000000002
690398.0000000033 2091276.9999999995	690498.0000000033 2091076.9999999995
690361.9999999997 2091295.0000000016	690461.9999999997 2091095.0000000016
690316.0000000012 2091298.9999999988	690416.0000000012 2091098.9999999988
690239.0000000014 2091326.0000000000	690339.0000000014 2091126.0000000000
690258.0000000022 2091355.9999999974	690358.0000000022 2091155.9999999974
690272.0000000023 2091370.9999999972	690372.0000000023 2091170.9999999972
690301.0000000037 2091403.0000000005	690401.0000000037 2091203.0000000005
690326.0000000031 2091451.0000000005	690426.0000000031 2091251.0000000005
690359.0000000019 2091441.9999999977	690459.0000000019 2091241.9999999977

จากตารางที่ 4.9 เมื่อนำค่าแกน x และแกน y มาคำนวณโดยใช้หลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) ค่าที่ได้ตรงกับค่าที่สมมุติขึ้นโดยระยะทางแกน x = 100 เมตร แกน y = 200 เมตร ระยะกระจัด 223.61 เมตร และทิศทางมีค่า 296.57 องศา โดยแสดงผลลัพธ์ในรูปที่ 4.15



รูปที่ 4.16 แสดงผลลัพธ์ในการคำนวณโดยการสมมุติค่า แกน x = 100 เมตร แกน y = 200 เมตร ระยะกระจัด = 223.61 เมตร และทิศทางมีค่า 296.57 องศา

4.3 การเปรียบเทียบข้อมูลโดยการสร้างข้อมูลจากแผนที่

ในการเลือกทดสอบเมื่อเลือกจุดใดจุดหนึ่งในขอบเขตของแต่ละรูปหลายเหลี่ยม (Polygon) นั้น โปรแกรมจะต้องแสดงจุดแต่ละจุดของแต่ละรูปหลายเหลี่ยม (Polygon) ตามข้อมูลที่มีได้ และสร้าง รูปหลายเหลี่ยม (Polygon) ขึ้นมาเพื่อเปรียบกับรูปหลายเหลี่ยม (Polygon) ที่เลือกไว้ และตรวจสอบระยะกระจัด ทิศทาง และความคลาดเคลื่อนตามแกน x และแกน y

4.3.1 การสร้าง Polygon แบบจำนวนจุดเท่ากัน

รูปที่ 4.17 ภาพแสดงการสร้าง Polygon ที่จำนวนจุดเท่ากัน

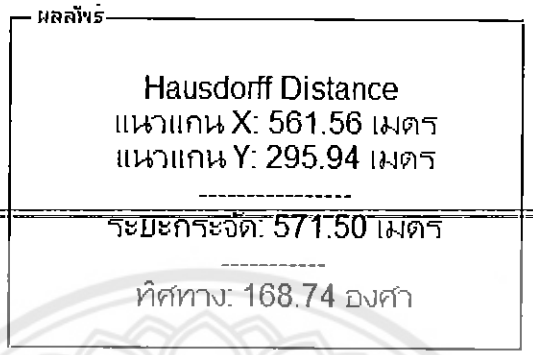
ตารางที่ 4.10 ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และ สี**เหลือง**

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)		รูปหลายเหลี่ยม (Polygon) **สีเหลือง** (x , y)	
691082.3372900027	2087970.2589899998	690528.1057494861	2087252.6570039499
691072.3372900025	2087888.2589900012	690456.2371663239	2087252.6570039499
691070.3372900041	2087823.2589899986	690411.3193018476	2087225.7062852641
691243.3372900030	2087553.2589900005	690357.4178644759	2087171.8048478924
691234.3372900034	2087526.2589899981	690348.4342915806	2087144.8541292066
691244.3372900034	2087484.2589899995	690303.5164271042	2087171.8048478924
691269.3372900026	2087465.2589899984	690276.5657084184	2087252.6570039499
691236.3372900023	2087400.2589900007	690195.7135523609	2087270.6241497404
691135.5542200023	2087336.1006399996	690123.8449691987	2087279.6077226358
691218.6279100040	2087178.0095599997	690105.8778234081	2087342.4927329028
691158.9620000036	2087143.5258099998	690105.8778234081	2087378.4270244839

ตารางที่ 4.10 (ต่อ) ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และ สีเหลือง

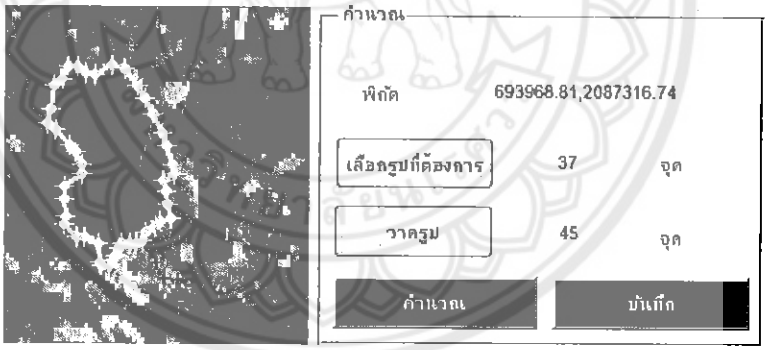
รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)		รูปหลายเหลี่ยม (Polygon) สีเหลือง (x , y)	
691100.8123400026	2087072.9010999994	690105.8778234081	2087513.1806179129
691036.7125700046	2087013.9010999990	690096.8942505128	2087576.0656281798
690997.9208100006	2086968.0095600008	690177.7464065703	2087594.0327739704
690975.0704700012	2086892.3848499993	690195.7135523609	2087683.8685029233
690924.0118900021	2086848.9097899981	690159.7792607799	2087764.7206589808
690832.3372900023	2087001.2589900012	690087.9106776175	2087854.5563879334
690751.3372900016	2087039.2589900002	690051.9763860365	2087899.4742524100
690668.3372900022	2087051.2589899988	690042.9928131412	2087962.3592626769
690630.3372900026	2087138.2589899993	690042.9928131412	2088052.1949916296
690597.3372900018	2087263.2589899988	690087.9106776175	2088124.0635747919
690638.3372900022	2087343.2589899991	690114.8613963034	2088213.8993037445
690689.3372900023	2087460.2589899977	690177.7464065703	2088267.8007411163
690561.3372900020	2087566.2589899991	690204.6971252562	2088267.8007411163
690580.3372900041	2087583.2589899995	690312.4999999995	2088249.8335953257
690508.3372900023	2087634.2589899977	690411.3193018476	2088240.8500224305
690542.3372900046	2087754.2589899974	690492.1714579051	2088079.1457103156
690572.3372900034	2087820.2589900012	690590.9907597532	2087953.3756897815
690603.3372900044	2087927.2589900005	690662.8593429154	2087845.5728150383
690646.3372900017	2088017.2589899995	690707.7772073917	2087701.8356487139
690763.3372900005	2087976.2589899981	690707.7772073917	2087576.0656281798
690799.3372900029	2088043.2589900000	690680.8264887059	2087504.1970450177
690869.3372900038	2088038.2589899995	690698.7936344964	2087432.3284618554
690946.3372900024	2088040.2589899986	690698.7936344964	2087396.3941702745
690965.3372900019	2088065.2589899991	690626.9250513342	2087369.4434515885
690978.3372900009	2088081.2589899981	690582.0071868579	2087351.4763057979
691082.3372900027	2087970.2589899998	690555.0564681720	2087288.5912955310

จากตารางที่ 4.10 เมื่อนำค่าแกน x และแกน y มาคำนวณโดยใช้หลักการเฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) มีค่าระยะทางแกน x = 561.56 เมตร แกน y = 295.94 เมตร ระยะกระจัด 571.50 เมตร และทิศทางมีค่า 168.74 องศา โดยแสดงผลลัพธ์ในรูปที่ 4.17



รูปที่ 4.18 แสดงผลลัพธ์ในการคำนวณ แกน x = 561.56 เมตร แกน y = 295.94 เมตร ระยะกระจัด 571.50 เมตร และทิศทางมีค่า 168.74 องศา

4.3.2 การสร้างรูปหลายเหลี่ยม (Polygon) แบบจำนวนจุดไม่เท่ากัน



รูปที่ 4.19 ภาพแสดงการสร้างรูปหลายเหลี่ยม (Polygon) ที่จำนวนจุดไม่เท่ากัน

ตารางที่ 4.11 ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และ สีเหลือง

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีเหลือง (x , y)
691082.3372900027 2087970.2589899998	690305.7623203285 2087137.0636550307
691072.3372900025 2087888.2589900012	690292.2869609856 2087240.3747433266
691070.3372900041 2087823.2589899986	690211.4348049281 2087285.2926078029
691243.3372900030 2087553.2589900005	690171.0087268994 2087280.8008213553

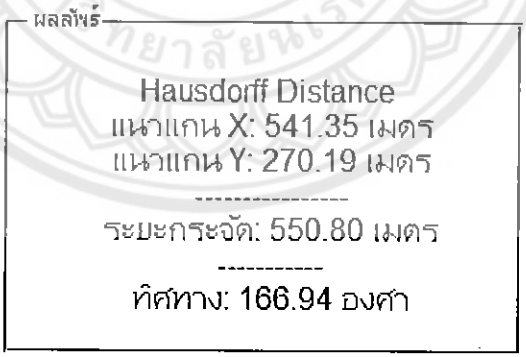
ตารางที่ 4.10 (ต่อ) ตารางแสดงค่า รูปหลายเหลี่ยม (Polygon) สีเขียว และ สีเหลือง

รูปหลายเหลี่ยม (Polygon) (x , y)		รูปหลายเหลี่ยม (Polygon) สีเหลือง (x , y)	
691234.3372900034	2087526.2589899981	690099.1401437372	2087352.6694045174
691244.3372900034	2087484.2589899995	690103.6319301848	2087397.5872689937
691269.3372900026	2087465.2589899984	690099.1401437372	2087496.4065708418
691236.3372900023	2087400.2589900007	690099.1401437372	2087568.2751540041
691135.5542200023	2087336.1006399996	690193.4676591376	2087644.6355236140
691218.6279100040	2087178.0095599997	690193.4676591376	2087698.5369609855
691158.9620000036	2087143.5258099998	690162.0251540041	2087770.4055441478
691100.8123400026	2087072.9010999994	690126.0908624230	2087806.3398357290
691036.7125700046	2087013.9010999990	690040.7469199179	2087896.1755646816
690997.9208100006	2086968.0095600008	689995.8290554414	2087927.6180698152
690975.0704700012	2086892.3848499993	690018.2879876797	2088012.9620123203
690924.0118900021	2086848.9097899981	690036.2551334703	2088143.2238193019
690832.3372900023	2087001.2589900012	690081.1729979466	2088179.1581108829
690751.3372900016	2087039.2589900002	690117.1072895278	2088251.0266940452
690668.3372900022	2087051.2589899988	690166.5169404517	2088251.0266940452
690630.3372900026	2087138.2589899993	690211.4348049281	2088219.5841889116
690597.3372900018	2087263.2589899988	690269.8280287475	2088210.6006160164
690638.3372900022	2087343.2589899991	690368.6473305955	2088260.0102669403
690689.3372900023	2087460.2589899977	690440.5159137577	2088246.5349075976
690561.3372900020	2087566.2589899991	690507.8927104723	2088147.7156057495
690580.3372900041	2087583.2589899995	690557.3023613963	2088071.3552361396
690508.3372900023	2087634.2589899977	690597.7284394250	2087981.5195071870
690542.3372900046	2087754.2589899974	690651.6298767967	2087878.2084188911
690572.3372900034	2087820.2589900012	690678.5805954826	2087801.8480492814
690603.3372900044	2087927.2589900005	690719.0066735114	2087698.5369609855
690646.3372900017	2088017.2589899995	690727.9902464065	2087640.1437371664
690763.3372900005	2087976.2589899981	690660.6134496920	2087604.2094455853
690799.3372900029	2088043.2589900000	690678.5805954826	2087545.8162217659
690869.3372900038	2088038.2589899995	690692.0559548255	2087473.9476386036
690946.3372900024	2088040.2589899986	690710.0231006161	2087393.0954825461

ตารางที่ 4.11 (ต่อ) ตารางแสดงค่ารูปหลายเหลี่ยม (Polygon) สีเขียว และ สีเหลือง

รูปหลายเหลี่ยม (Polygon) สีเขียว (x , y)	รูปหลายเหลี่ยม (Polygon) สีเหลือง (x , y)
690965.3372900019 2088065.2589899991	690647.1380903490 2087375.1283367556
690978.3372900009 2088081.2589899981	690620.1873716633 2087339.1940451746
691082.3372900027 2087970.2589899998	690566.2859342916 2087312.2433264886
	690548.3187885011 2087258.3418891171
	690512.3844969199 2087240.3747433266
	690476.4502053388 2087249.3583162217
	690431.5323408624 2087249.3583162217
	690413.5651950719 2087217.9158110884
	690391.1062628337 2087177.4897330594
	690359.6637577002 2087119.0965092403
	690314.7458932238 2087119.0965092403

จากตารางที่ 4.11 เมื่อนำค่าแกน x และแกน y มาคำนวณโดยใช้หลักการ**เฮาซดอรฟดิสแทนซ์** (Hausdorff Distance) มีค่าระยะทางแกน x = 541.35 เมตร แกน y = 270.19 **เมตร ระยะกระจัด** 550.80 เมตร และทิศทางมีค่า 166.94 องศา โดยแสดงผลลัพธ์ในรูปที่ 4.19



รูปที่ 4.20 แสดงผลลัพธ์ในการคำนวณ แกน x = 541.35 เมตร แกน y = 270.19 เมตร ระยะกระจัด 550.80 เมตร และทิศทางมีค่า 166.94 องศา

บทที่ 5
บทสรุปและข้อเสนอแนะ

5.1 สรุปผลการดำเนินงาน

ผู้วิจัยได้ทำการวัดความคลาดเคลื่อนของพื้นที่ป่าสำหรับภาพถ่ายดาวเทียม ซึ่งได้ออกแบบโปรแกรมเพื่อหาความคลาดเคลื่อนของพื้นที่ป่าระหว่างข้อมูลพื้นที่ป่าจากภาพถ่ายดาวเทียมกับข้อมูลพื้นที่ป่าจากการสำรวจพื้นที่จริง แสดงเป็นข้อมูลความคาดเคลื่อนโดยใช้หลักการของ เฮาซดอรฟดิสแทนซ์ (Hausdorff Distance) ในการคำนวณเป็นหลัก สามารถนำข้อมูลที่ได้จากการคำนวณไปใช้งานต่อไปได้ เพื่อช่วยลดการทำงานผู้สำรวจพื้นที่จริง

5.2 ปัญหาและอุปสรรคที่พบในการดำเนินงาน

- ภาพถ่ายจากดาวเทียมที่ใช้ในการสร้างรูปหลายเหลี่ยม (Polygon) มีข้อจำกัดในการมองเห็น
- ข้อมูลที่ได้จากการสำรวจมีข้อมูลจากแหล่งที่มาเดียว ทำให้โปรแกรมไม่สามารถเปรียบข้อมูลจากหลายๆแหล่งได้
- ในการทำงานบางครั้งขาดการวางแผนที่ดีทำให้งานล่าช้า และเสร็จไม่ทันตามกำหนด

5.3 แนวทางการแก้ไขปัญหา

- ภาพที่นำมาใช้งานควรมีการใช้ภาพที่ชัดเจน และเป็นภาพถ่ายในระยะสายตามองเห็นได้ชัด
- หาข้อมูลเพิ่มเติมเพื่อที่จะสามารถใช้เป็นหลักเกณฑ์ในเปรียบเทียบโปรแกรมได้
- วางแผนการทำงานและเผื่อเวลาที่แน่นอนให้งานออกมาทันตามกำหนด

5.4 ข้อเสนอแนะสำหรับงานในอนาคต

ปัจจุบันข้อมูลภาพถ่ายดาวเทียมมีแหล่งข้อมูลที่จำกัด ทำให้การออกแบบรูปหลายเหลี่ยม (Polygon) จากภาพถ่ายดาวเทียมเพื่อเปรียบเทียบรูปหลายเหลี่ยม (Polygon) จากพื้นที่จริงนั้นมีข้อจำกัดทางด้านข้อมูล เพราะฉะนั้นต้องมีการใช้ภาพที่สมบูรณ์เพื่อมาใช้ในการทดลองต่อไปให้ดียิ่งขึ้น อาจจะเป็นภาพจากดาวเทียมโดยตรงแบบอัตโนมัติโดยไม่ต้องใช้เป็นภาพที่ถ่ายจากดาวเทียม เพื่อจะได้เกิดความแม่นยำและถูกต้องกับการใช้งาน และเพิ่มอัลกอลิทึมให้มีความหลากหลายในการหาความคลาดเคลื่อน เพราะยังมีอีกหลายอัลกอลิทึมที่สามารถหาความคลาดเคลื่อนของพื้นที่ป่าได้

เอกสารอ้างอิง

[1] ชฎา ณรงค์ฤทธิ์. 2547. **ระบบสารสนเทศภูมิศาสตร์ด้านสิ่งแวดล้อม (Environmental Geographic Information System).** พิมพ์ครั้งที่ 1. พิษณุโลก: ภาควิชาทรัพยากรธรรมชาติและสิ่งแวดล้อม คณะเกษตรศาสตร์ ทรัพยากรธรรมชาติและสิ่งแวดล้อม มหาวิทยาลัยนเรศวร.

[2] อณุศร พุ่มพวง. **งานแผนที่ในระบบ GIS ความคลาดเคลื่อนตลอดกาล.** กรุงเทพฯ: กองสารสนเทศภูมิศาสตร์ สำนักยุทธศาสตร์และประเมินผล.

[3] วีระภาส คุณรัตนสิริ. **การปรับแก้ความคลาดเคลื่อนของภาพดาวเทียมทางเรขาคณิต.** กรุงเทพฯ: ภาควิชาการจัดการป่าไม้ คณะวนศาสตร์ มหาวิทยาลัยเกษตรศาสตร์.

[4] Harvey Greenberg. **TFW File.** สืบค้นเมื่อวันที่ 20 สิงหาคม 2556 จาก http://gis.ess.washington.edu/data/raster/drg/tfw.html.

[5] Traffy. **SHP File.** สืบค้นเมื่อวันที่ 20 สิงหาคม 2556 จาก http://its.nectec.or.th/2011/07/07/คู่มือการใช้งานโปรแกรม

[6] Wiki. **Point in Polygon.** สืบค้นวันที่ 16 ตุลาคม 2556. จาก http://en.wikipedia.org/wiki/Point_in_polygon.

[7] Wiki. **Euclidean Distance.** สืบค้นเมื่อวันที่ 16 ตุลาคม 2556 จาก http://en.wikipedia.org/wiki/Euclidean_distance.

[8] Wiki. **Hausdorff_distance.** สืบค้นเมื่อวันที่ 20 กันยายน 2556. จาก http://en.wikipedia.org/wiki/Hausdorff_distance.

[9] Wiki. **Trigonometric Function.** สืบค้นเมื่อวันที่ 20 กันยายน 2556. จาก http://en.wikipedia.org/wiki/ Trigonometric Function.

[10] Mathworks. **Point in Polygon.** สืบค้นวันที่ 13 พฤศจิกายน 2556. จาก http://www.mathworks.com/help/matlab/ref/inpolygon.html.

ภาคผนวก

Functions Main ในการออกแบบ GUI

```
function varargout = test_SAT(varargin)
% TEST_SAT MATLAB code for test_SAT.fig
% TEST_SAT, by itself, creates a new TEST_SAT or raises the existing
% singleton*.
% H = TEST_SAT returns the handle to a new TEST_SAT or the handle to
% the existing singleton*.
% TEST_SAT('CALLBACK',hObject,eventData,handles,...) calls the local
% function named CALLBACK in TEST_SAT.M with the given input arguments.
% TEST_SAT('Property','Value',...) creates a new TEST_SAT or raises the
% existing singleton*.  Starting from the left, property value pairs are
% applied to the GUI before test_SAT_OpeningFcn gets called.  An
% unrecognized property name or invalid value makes property application
% stop.  All inputs are passed to test_SAT_OpeningFcn via varargin.
% *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
% instance to run (singleton)".
% See also: GUIDE, GUIDATA, GUIHANDLES
% Edit the above text to modify the response to help test_SAT
% Last Modified by GUIDE v2.5 22-Sep-2013 22:29:44
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
```

```
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @test_SAT_OpeningFcn, ...
                   'gui_OutputFcn',  @test_SAT_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before test_SAT is made visible.
function test_SAT_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to test_SAT (see VARARGIN)

% Choose default command line output for test_SAT
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
% UIWAIT makes test_SAT wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% --- Outputs from this function are returned to the command line.
function varargout = test_SAT_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
zoom on
```

```
% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[filename, pathname] = uigetfile('*.jpg', 'Select a JPG file');
if isequal(filename,0)
   disp('User selected Cancel');
else
   disp(['User selected', fullfile(pathname, filename)]);
   assignin('base', 'JPG', fullfile(pathname, filename));
end
guidata(hObject, handles);
% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[filename, pathname] = uigetfile('*.tfw', 'Select a TFW file');
if isequal(filename,0)
   disp('User selected Cancel');
else
   disp(['User selected', fullfile(pathname, filename)]);
```

```
    assignin('base', 'TFW', fullfile(pathname, filename));
end
% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[filename, pathname] = uigetfile('*dbf', 'Select a SHP file');
if isequal(filename,0)
    disp('User selected Cancel');
else
    disp(['User selected', fullfile(pathname, filename)]);
    assignin('base', 'DBF', fullfile(pathname, filename));
end
guidata(hObject, handles);


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
JPG = evalin('base', 'JPG');
```

```
TFW = evalin('base', 'TFW');
SHP = evalin('base', 'DBF');
axes(handles.axes1);
[X, cmap] = imread(JPG);
r = X(:,:,1);
r = medfilt2(r, [3 3]);
H = fspecial('unsharp');
r = imfilter(r,H,'replicate');
g = X(:,:,2);
g = medfilt2(g, [3 3]);
H = fspecial('unsharp');
g = imfilter(g,H,'replicate');
b = X(:,:,3);
b = medfilt2(b, [3 3]);
H = fspecial('unsharp');
b = imfilter(b,H,'replicate');
X=cat(3,r,g,b);
R_orig = worldfileread(TFW);
mapshow(X,R_orig);
landareas = shaperead(SHP);
blueRoads = makesymbolspec('Polygon',{'Default','FaceColor','none','EdgeColor','g'});
mapshow(landareas,'SymbolSpec',blueRoads,'Visible','on');
handles.land = landareas;
```

```
set(handles.text2,'string','');
set(handles.text10,'string','');
set(handles.text12,'string','');
guidata(hObject, handles);
% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
zoom off
pan off
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
c = handles.c;
r = handles.r;
%[x, y] = ginputc(Lc, 'Color', 'g', 'LineWidth', 3,'ShowPoints', true, 'ConnectPoints', true);
%[x, y] = ginputc('ShowPoints', true, 'ConnectPoints', true);
% [x1, y1] = ginputc('Color', 'b', 'LineWidth', 3,'ShowPoints', true, 'ConnectPoints', true);
[x1, y1] = ginputc('Color', 'b', 'LineWidth', 3,'ShowPoints', true, 'ConnectPoints', true);
h = impoly(gca, [x1,y1]);
setColor(h,'y');
addNewPositionCallback(h,@(p) title(mat2str(p,3)));
fcn = makeConstrainToRectFcn('impoly',get(gca,'XLim'),...
get(gca,'YLim'));
setPositionConstraintFcn(h,fcn);
```

```
l = length(x1);
set(handles.text12,'string',l);
meanxnew = mean(x1);
meanynew = mean(y1);
% % min max valu polygon two
% mitwo = min(x1);
% mxtwo = max(x1);
% valutwo = (mxtwo-mitwo);
% rtwo = valutwo/2;
% %  ============================
% %----------------------------------------------------------------
%
% handles.rtwo = rtwo;
handles.meanxnew = meanxnew;
handles.meanynew = meanynew;
handles.c = c;
handles.r = r;
handles.x1 = x1;
handles.y1 = y1;
guidata(hObject, handles);
% --- Executes on button press in pushbutton7.
```

```
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% rone = handles.rone;
% rtwo = handles.rtwo;
a = handles.c;
b = handles.x1;
c = handles.r;
d = handles.y1;
%dist = sqrt((abs(b(1,1) - a(1,1)))^2 + abs((d(1,1) - c(1,1)))^2);
aa = a(:);
cc = c(:);
% size (cc);
% size (b);
% size (ff);
% size (d);
[distx] = hausdorff(aa,b);
[disty] = hausdorff(cc,d);
% keep the minimum of the two distances
% frist point use cal
% sone = ((distx / rone)*180)/pi;
% stwo = ((distx / rtwo)*180)/pi;
```

```
% valuedata = (stwo - sone);



meanxnew = handles.meanxnew;

meanynew = handles.meanynew;

meanxold = handles.meanxold;

meanyold = handles.meanyold;

distan = sqrt(((abs(meanxold-meanxnew))^2) + ((abs(meanyold-meanynew))^2));

% distan = sqrt((abs(distx)^2 + abs(disty)^2));

deg = atand(  (abs(meanyold-meanynew)) / (abs(meanxold-meanxnew)) );

msgx = sprintf('Lat: %.02f M\nLong: %.02f M\nDistance: %.02f M \nDegrees:
%.02f°',distx,disty,distan,deg);

handles.distx = distx;

handles.disty = disty;

handles.distan = distan;

handles.deg = deg;

set(handles.text2,'string',msgx);

guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.

function edit1_CreateFcn(hObject, eventdata, handles)

% hObject    handle to edit1 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in pushbutton11.
function pushbutton11_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
zoom out
% --- Executes on button press in pushbutton12.
function pushbutton12_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close();
% --- Executes on button press in pushbutton13.
function pushbutton13_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
pan on

% --- Executes on mouse motion over figure - except title and menu.
function figure1_WindowButtonMotionFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
xy = get(handles.axes1,'CurrentPoint');
msg = sprintf('%.02f,%.02f',xy(1),xy(3));
set(handles.text7,'string',msg);

% --- Executes on button press in pushbutton16.
function pushbutton16_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
zoom off
pan off
h = handles.land;
L = length(h);
blueRoads = makesymbolspec('Polygon',{'Default','FaceColor','none','EdgeColor','g'});
mapshow(h,'SymbolSpec',blueRoads);
[x, y] = ginputc(1, 'Color', 'g', 'LineWidth', 3,'ShowPoints', true, 'ConnectPoints', true);
```

```
for i = 1:L
c = h(i,:).X;
r = h(i,:).Y;
in = inpolygon(x,y,c,r);
if in == 1
    mapshow(c,r,'DisplayType','MultiPoint');
    L = length(c);
    set(handles.text10,'string',L);
    break;
elseif i == L && in == 0
    h = msgbox('No have Polygon','Error');
    set(handles.text12,'string','');
    set(handles.text10,'string','');
    set(handles.text2,'string','');
end
%
% distx(i) = hausdorff(meanx(i),x);
% disty(i) = hausdorff(meany(i),y);
end
% [Euc_dist_minx, xRecognized_index] = min(distx);
% [Euc_dist_miny, yRecognized_index] = min(disty);
% %lo_min_realx = find(find_minx == min_realx && find_miny == min_realy)
%
```

```
% if Euc_dist_minx < Euc_dist_miny
% c = h(xRecognized_index,:).X;
% r = h(xRecognized_index,:).Y;
%
% elseif  Euc_dist_minx > Euc_dist_miny
% c = h(yRecognized_index,:).X;
% r = h(yRecognized_index,:).Y;
%
% end
% min max valu polygon one
% mione = min(c);
% mxone = max(c);
% Remove NAN (c)
sz = size(c);
c = reshape(c', size(c,1)*size(c,2), 1);
c(isnan(c)) = [];
c = reshape(c, sz(2)-1, sz(1))';
% Remove NAN (r)
sz = size(r);
r = reshape(r', size(r,1)*size(r,2), 1);
r(isnan(r)) = [];
r = reshape(r, sz(2)-1, sz(1))';
meanx = mean(c);
```

```
meany = mean(r);

% valuone = (mxone-mione);
% rone = valuone/2;
% ============================
% mapshow(c,r,'DisplayType','MultiPoint')
handles.meanxold = meanx;
handles.meanyold = meany;
handles.c = c;
handles.r = r;
handles.x = x;
handles.y = y;
guidata(hObject, handles);
% --- Executes on button press in pushbutton17.
function pushbutton17_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
zoom off
pan off
% --- Executes on button press in pushbutton19.
function pushbutton19_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton19 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% --- Executes on button press in pushbutton20.
function pushbutton20_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
a = handles.c;
b = handles.x1;
c = handles.r;
d = handles.y1;
distx= handles.distx;
disty= handles.disty;
distan= handles.distan;
deg = handles.deg;
siz = length(a);
siz2 = length(b);
fileID = fopen('Data.txt','a+');
fprintf(fileID,'%15s %17s\r\n','Lat Origin','Long Origin');
for i = 1:siz
fprintf(fileID,'%9.10f  %17.10f\r\n',a(i),c(i));
end
fprintf(fileID,'%s %d %s \r\n\n','Size:',siz , 'Point');
```

```
fprintf(fileID,'%12s %17s\r\n','Lat New','Long New');
for j = 1:siz2
fprintf(fileID,'%9.10f  %17.10f\r\n',b(j),d(j));
end
fprintf(fileID,'%s %d %s \r\n\n','Size:',siz2 , 'Point');
fprintf(fileID,'%10s %17s %17s %11s\r\n','Lat Error','Long Error','Distan Error','Degrees');
fprintf(fileID,'%6.10f  %6.10f  %6.10f  %6.2f \r\n\n',distx,disty,distan,deg);
format shortg;
c = clock;
fprintf(fileID,'%s %d/%d/%d %d:%d:%.f \r\n\n','DateTime',c);
fclose(fileID);
winopen('Data.txt')
guidata(hObject, handles);
```

Functions การสร้างจุดและการเลือกจุด

```
function [x, y, button, ax] = ginputc(varargin)
%GINPUTC Graphical input from mouse.
%   GINPUTC behaves similarly to GINPUT, except you can customize the
%   cursor color, line width, and line style.
%
%   [X,Y] = GINPUTC(N) gets N points from the current axes and returns
%   the X- and Y-coordinates in length N vectors X and Y.  The cursor
%   can be positioned using a mouse.  Data points are entered by pressing
%   a mouse button or any key on the keyboard except carriage return,
%   which terminates the input before N points are entered.
%       Note: if there are multiple axes in the figure, use mouse clicks
%             instead of key presses. Key presses may not select the axes
%             where the cursor is.
%
%   [X,Y] = GINPUTC gathers an unlimited number of points until the return
%   key is pressed.
%
%   [X,Y] = GINPUTC(N, PARAM, VALUE) and [X,Y] = GINPUTC(PARAM, VALUE)
%   specifies additional parameters for customizing. Valid values for PARAM
%   are:
```

```
%       'FigHandle'     : Handle of the figure to activate. Default is gcf.
%       'Color'         : A three-element RGB vector, or one of the MATLAB
%                         predefined names, specifying the line color. See
%                         the ColorSpec reference page for more information
%                         on specifying color. Default is 'k' (black).
%       'LineWidth'     : A scalar number specifying the line width.
%                         Default is 0.5.
%       'LineStyle'     : '-', '--', '-.', ':'. Default is '-'.
%       'ShowPoints'    : TRUE or FALSE specifying whether to show the
%                         points being selected. Default is false.
%       'ConnectPoints' : TRUE or FALSE specifying whether to connect the
%                         points as they are being selected. This only
%                         applies when 'ShowPoints' is set to TRUE. Default
%                         is true.
%
%   [X,Y,BUTTON] = GINPUTC(...) returns a third result, BUTTON, that
%   contains a vector of integers specifying which mouse button was used
%   (1,2,3 from left) or ASCII numbers if a key on the keyboard was used.
%
%   [X,Y,BUTTON,AX] = GINPUTC(...) returns a fourth result, AX, that
%   contains a vector of axes handles for the data points collected.
%
%   Requires MATLAB R2007b or newer.
```

```
%
%   Examples:
%       [x, y] = ginputc;
%
%       [x, y] = ginputc(5, 'Color', 'r', 'LineWidth', 3);
%
%       [x, y, button] = ginputc(1, 'LineStyle', ':');
%
%       subplot(1, 2, 1); subplot(1, 2, 2);
%       [x, y, button, ax] = ginputc;
%
%       [x, y] = ginputc('ShowPoints', true, 'ConnectPoints', true);
%
%   See also GINPUT, GTEXT, WAITFORBUTTONPRESS.

% Jiro Doke
% October 19, 2012
% Copyright 2012 The MathWorks, Inc.

try
    if verLessThan('matlab', '7.5')
        error('ginputc:Init:IncompatibleMATLAB', ...
            'GINPUTC requires MATLAB R2007b or newer');
```

```
    end
catch %#ok<CTCH>
    error('ginputc:Init:IncompatibleMATLAB', ...
        'GINPUTC requires MATLAB R2007b or newer');
end

% Check input arguments
p = inputParser();

addOptional(p, 'N', inf, @(x) validateattributes(x, {'numeric'}, ...
    {'scalar', 'integer', 'positive'}));
addParamValue(p, 'FigHandle', [], @(x) numel(x)==1 && ishandle(x));
addParamValue(p, 'Color', 'k', @colorValidFcn);
addParamValue(p, 'LineWidth', 0.5 , @(x) validateattributes(x, ...
    {'numeric'}, {'scalar', 'positive'}));
addParamValue(p, 'LineStyle', '-' , @(x) validatestring(x, ...
    {'-', '--', '-.', ':'}));
addParamValue(p, 'ShowPoints', false, @(x) validateattributes(x, ...
    {'logical'}, {'scalar'}));
addParamValue(p, 'ConnectPoints', true, @(x) validateattributes(x, ...
    {'logical'}, {'scalar'}));

parse(p, varargin{:});
```

```
N = p.Results.N;
hFig = p.Results.FigHandle;
color = p.Results.Color;
linewidth = p.Results.LineWidth;
linestyle = p.Results.LineStyle;
showpoints = p.Results.ShowPoints;
connectpoints = p.Results.ConnectPoints;

%--------------------------------------------------------------------
    function tf = colorValidFcn(in)
        % This function validates the color input parameter

        validateattributes(in, {'char', 'double'}, {'nonempty'});
        if ischar(in)
            validatestring(in, {'b', 'g', 'r', 'c', 'm', 'y', 'k', 'w'});
        else
            assert(isequal(size(in), [1 3]) && all(in>=0 & in<=1), ...
                'ginputc:InvalidColorValues', ...
                'RGB values for "Color" must be a 1x3 vector between 0 and 1');
            % validateattributes(in, {'numeric'}, {'size', [1 3], '>=', 0, '<=', 1})
        end
        tf = true;
```

```
    end
%--------------------------------------------------------------------

if isempty(hFig)
    hFig = gcf;
end

% Try to get the current axes even if it has a hidden handle.
hAx = get(hFig, 'CurrentAxes');
if isempty(hAx)
    allAx = findall(hFig, 'Type', 'axes');
    if ~isempty(allAx)
        hAx = allAx(1);
    else
        hAx = axes('Parent', hFig);
    end
end

% Handle interactive properites of HG objects. Save the current settings so
% that they can be restored later
allHG = findall(hFig);
propsToChange = {...
    'WindowButtonUpFcn', ...
```

```
    'WindowButtonDownFcn', ...
    'WindowButtonMotionFcn', ...
    'WindowKeyPressFcn', ...
    'WindowKeyReleaseFcn', ...
    'ButtonDownFcn', ...
    'KeyPressFcn', ...
    'KeyReleaseFcn', ...
    'ResizeFcn'};
validObjects = false(length(allHG), length(propsToChange));
curCallbacks = cell(1, length(propsToChange));

% Save current properties and set them to ''
for id = 1:length(propsToChange)
    validObjects(:, id) = isprop(allHG, propsToChange{id});
    curCallbacks{id} = get(allHG(validObjects(:, id)), propsToChange(id));
    set(allHG(validObjects(:, id)), propsToChange{id}, '');
end

% Save current pointer
curPointer = get(hFig, 'Pointer');
curPointerShapeCData = get(hFig, 'PointerShapeCData');

% Change window functions
```

```
    set(hFig, ...
        'WindowButtonDownFcn', @mouseClickFcn, ...
        'WindowButtonMotionFcn', @mouseMoveFcn, ...
        'KeyPressFcn', @keyPressFcn, ...
        'ResizeFcn', @resizeFcn, ...
        'Pointer', 'custom', ...
        'PointerShapeCData', nan(16, 16));

    % Create an invisible axes for displaying the full crosshair cursor
    hInvisibleAxes = axes(...
        'Parent', hFig, ...
        'Units', 'normalized', ...
        'Position', [0 0 1 1], ...
        'XLim', [0 1], ...
        'YLim', [0 1], ...
        'HitTest', 'off', ...
        'HandleVisibility', 'off', ...
        'Visible', 'off');

    % Create line object for the selected points
    if showpoints
        if connectpoints
            pointsLineStyle = '-';
```

```
        else
            pointsLineStyle = 'none';
        end

        selectedPoints = [];
        hPoints = line(nan, nan, ...
            'Parent', hInvisibleAxes, ...
            'HandleVisibility', 'off', ...
            'HitTest', 'off', ...
            'Color', [1 0 0], ...
            'Marker', 'o', ...
            'MarkerFaceColor', [1 .7 .7], ...
            'MarkerEdgeColor', [1 0 0], ...
            'LineStyle', pointsLineStyle);
    end

    % Create tooltip for displaying selected points

    hTooltipControl = text(0, 1, 'HIDE', ...
        'Parent', hInvisibleAxes, ...
        'HandleVisibility', 'callback', ...
        'FontName', 'FixedWidth', ...
        'VerticalAlignment', 'top', ...
```

```
    'HorizontalAlignment', 'left', ...
    'BackgroundColor', [.5 1 .5]);
hTooltip = text(0, 0, 'No points', ...
    'Parent', hInvisibleAxes, ...
    'HandleVisibility', 'off', ...
    'HitTest', 'off', ...
    'FontName', 'FixedWidth', ...
    'VerticalAlignment', 'top', ...
    'HorizontalAlignment', 'left', ...
    'BackgroundColor', [1 1 .5]);

% Call resizeFcn to update tooltip location
resizeFcn();

% Create full crosshair lines
hCursor = line(nan, nan, ...
    'Parent', hInvisibleAxes, ...
    'Color', color, ...
    'LineWidth', linewidth, ...
    'LineStyle', linestyle, ...
    'HandleVisibility', 'off', ...
```

```
    'HitTest', 'off');

% Prepare results
x = [];
y = [];
button = [];
ax = [];

% Wait until enter is pressed.
uiwait(hFig);

%--------------------------------------------------------------------
    function mouseMoveFcn(varargin)
        % This function updates cursor location based on pointer location

        cursorPt = get(hInvisibleAxes, 'CurrentPoint');

        set(hCursor, ...
            'XData', [0 1 nan cursorPt(1) cursorPt(1)], ...
            'YData', [cursorPt(3) cursorPt(3) nan 0 1]);
    end
%--------------------------------------------------------------------
```

```
%------------------------------------------------------------------------

    function mouseClickFcn(varargin)

        % This function captures mouse clicks.

        % If the tooltip control is clicked, then toggle tooltip display.

        % If anywhere else is clicked, record point.

        if isequal(gco, hTooltipControl)

            tooltipClickFcn();

        else

            updatePoints(get(hFig, 'SelectionType'));

        end

    end

%------------------------------------------------------------------------


%------------------------------------------------------------------------

    function keyPressFcn(obj, edata) %#ok<INUSL>

        % This function captures key presses.

        % If "return", then exit.

        % If "delete" (or "backspace"), then delete previous point.

        % If any other key, record point.


        key = double(edata.Character);
```

```
if isempty(key)
    return;
end

switch key
    case 13  % return
        exitFcn();

    case {8, 127}  % delete or backspace
        if ~isempty(x)
            x(end) = [];
            y(end) = [];
            button(end) = [];
            ax(end) = [];

            if showpoints
                selectedPoints(end, :) = [];
                set(hPoints, ...
                    'XData', selectedPoints(:, 1), ...
                    'YData', selectedPoints(:, 2));
            end

            displayCoordinates();
```

```
            end

        otherwise
            updatePoints(key);

    end
end
%-------------------------------------------------------------------

%-------------------------------------------------------------------
function updatePoints(clickType)
    % This function captures the information for the selected point

    hAx = gca;
    pt = get(hAx, 'CurrentPoint');
    x = [x; pt(1)];
    y = [y; pt(3)];
    ax = [ax; hAx];

    if ischar(clickType)   % Mouse click
        switch lower(clickType)
            case 'open'
                clickType = 1;
```

```
            case 'normal'
                clickType = 1;
            case 'extend'
                clickType = 2;
            case 'alt'
                clickType = 3;
        end
    end
    button = [button; clickType];

    displayCoordinates();

    if showpoints
        cursorPt = get(hInvisibleAxes, 'CurrentPoint');
        selectedPoints = [selectedPoints; cursorPt([1 3])];
        set(hPoints, ...
            'XData', selectedPoints(:, 1), ...
            'YData', selectedPoints(:, 2));
    end

    % If captured all points, exit
    if length(x) == N
        exitFcn();
```

```
        end
    end
%-----------------------------------------------------------------------

%-----------------------------------------------------------------------
    function tooltipClickFcn()
        % This function toggles the display of the tooltip

        if strcmp(get(hTooltipControl, 'String'), 'SHOW')
            set(hTooltipControl, 'String', 'HIDE');
            set(hTooltip, 'Visible', 'on');
        else
            set(hTooltipControl, 'String', 'SHOW');
            set(hTooltip, 'Visible', 'off');
        end
    end
%-----------------------------------------------------------------------

%-----------------------------------------------------------------------
    function displayCoordinates()
        % This function updates the coordinates display in the tooltip
%
        if isempty(x)
```

```
            str = 'No points';
        else
            str = sprintf('%d: %0.3f, %0.3f\n', [1:length(x); x'; y']);
            str(end) = '';
        end
        set(hTooltip, ...
            'String', str);

    end
%---------------------------------------------------------------------

%---------------------------------------------------------------------
    function resizeFcn(varargin)
        % This function adjusts the position of tooltip when the figure is
        % resized

        sz = get(hTooltipControl, 'Extent');
        set(hTooltip, 'Position', [0 sz(2)]);
    end
%---------------------------------------------------------------------

%---------------------------------------------------------------------
```

```
function exitFcn()
    % This function exits GINPUTC and restores previous figure settings

    for idx = 1:length(propsToChange)
        set(allHG(validObjects(:, idx)), propsToChange(idx), curCallbacks{idx});
    end

    % Restore window functions and pointer
    %       set(hFig, 'WindowButtonDownFcn', curWBDF);
    %       set(hFig, 'WindowButtonMotionFcn', curWBMF);
    %       set(hFig, 'WindowButtonUpFcn', curWBUF);
    %       set(hFig, 'KeyPressFcn', curKPF);
    %       set(hFig, 'KeyReleaseFcn', curKRF);
    %       set(hFig, 'ResizeFcn', curRF);

    % Restore pointer
    set(hFig, 'Pointer', curPointer);
    set(hFig, 'PointerShapeCData', curPointerShapeCData);

    % Delete invisible axes and return control
    delete(hInvisibleAxes);
    uiresume(hFig);
end
```

```
%-------------------------------------------------------------------------

end
```

Functions ในการเปรียบเทียบข้อมูล

```
%% Hausdorff Distance: Compute the Hausdorff distance between two point clouds.

% Let A and B be subsets of a metric space (Z,dZ),

% The Hausdorff distance between A and B, denoted by dH (A, B), is defined by:

% dH (A, B)=max{sup dz(a,B), sup dz(b,A)}, for all a in A, b in B,

% dH(A, B) = max(h(A, B),h(B, A)),

% where h(A, B) = max(min(d(a, b))),

% and d(a, b) is a L2 norm.

% dist_H = hausdorff( A, B )

% A: First point sets.

% B: Second point sets.

% ** A and B may have different number of rows, but must have the same number of
columns. **

% Hassan RADVAR-ESFAHLAN; Universit du Qubec; TS; Montral; CANADA

% 15.06.2010

%%

function [dist] = hausdorff( A, B)
```

```
if(size(A,2) ~= size(B,2))
    fprintf( 'WARNING: dimensionality must be the same\n' );
    dist = [];
    return;
end
dist = max(compute_dist(A, B), compute_dist(B, A));
```

```
%% Compute distance
function[dist] = compute_dist(A, B)
m = size(A, 1);
n = size(B, 1);
dim= size(A, 2);
for k = 1:m
    C = ones(n, 1) * A(k, :);
    D = (C-B) .* (C-B);
    D = sqrt(D * ones(dim,1));
    dist(k) = min(D);
end
dist = max(dist);
```