

รายงานวิจัยฉบับสมบูรณ์

การพัฒนาวิธีการทำแผนที่โมเสคน้ำฝนประมาณค่าจากข้อมูลเรดาร์ตรวจ
อากาศภาคพื้นดินบริเวณลุ่มน้ำเจ้าพระยา

โดย

ดร. นัฐพล มหาวิค

ภาควิชาทรัพยากรธรรมชาติและสิ่งแวดล้อม

คณะเกษตรศาสตร์ทรัพยากรธรรมชาติและสิ่งแวดล้อม

มหาวิทยาลัยนเรศวร

กรกฎาคม พ.ศ. 2562

สำนักหอสมุด มหาวิทยาลัยนเรศวร
รับลงทะเบียน 06 ส.ค. 2564
เลขทะเบียน 1034755
เลขเรียกหนังสือ ๑ GA
102
.5
๒๓๒๕
๒๕๖๒

รายงานวิจัยฉบับสมบูรณ์

การพัฒนาวิธีการทำแผนที่โมเสคน้ำฝนประมาณค่าจากข้อมูลเรดาร์ตรวจ

อากาศภาคพื้นดินบริเวณลุ่มน้ำเจ้าพระยา



1. ดร.นัฐพล มหาวิค มหาวิทยาลัยนเรศวร
2. รศ.ดร.ศรินทิพย์ แทนธานี มหาวิทยาลัยนเรศวร

สนับสนุนโดย

งบประมาณแผ่นดินมหาวิทยาลัยนเรศวร

ประจำปี พ.ศ. 2561

กิตติกรรมประกาศ

(Acknowledgement)

โครงการการพัฒนาวิธีการทำแผนที่โมเสคน้ำฝนประมาณค่าจากข้อมูลเรดาร์ตรวจอากาศภาคพื้นดินบริเวณลุ่มน้ำเจ้าพระยา ได้รับทุนสนับสนุนจากสำนักงานการวิจัยแห่งชาติ (วช.) ผ่านการจัดสรรโดยมหาวิทยาลัยนเรศวร รหัสโครงการ R2561B063 โดยสามารถตีพิมพ์ผลงานวิจัยในวารสารที่อยู่ในฐานข้อมูลระดับนานาชาติได้ 1 ฉบับและรายงานสืบเนื่องในงานประชุมวิชาการระดับนานาชาติ 1 ฉบับ ผู้วิจัยขอขอบคุณสำนักงานการวิจัยแห่งชาติและมหาวิทยาลัยนเรศวรที่พิจารณาจัดสรรทุนในการศึกษาครั้งนี้ กรมอุตุนิยมวิทยา สถาบันสารสนเทศทรัพยากรน้ำและการเกษตร (องค์การมหาชน) และภาควิชาทรัพยากรธรรมชาติและสิ่งแวดล้อม คณะเกษตรศาสตร์ธรรมชาติและสิ่งแวดล้อม ที่ให้การสนับสนุนทั้งข้อมูลและเครื่องมือในการทำวิจัยเป็นอย่างดี

นัฐพล มหาวิค



คำนำ

โครงการการพัฒนาวิธีการทำแผนที่โมเสคน้ำฝนประมาณค่าจากข้อมูลเรดาร์ตรวจอากาศภาคพื้นดินบริเวณลุ่มน้ำเจ้าพระยา เป็นโครงการต้นแบบในการพัฒนากระบวนการโมเสคข้อมูลเรดาร์ ในงานวิจัยครั้งนี้ทุกกระบวนการได้พัฒนาจากภาษาไพธอนซึ่งนอกจากจะได้องค์ความรู้ของกระบวนการที่พัฒนาขึ้นมาแล้ว ยังสามารถนำโค้ดที่ได้พัฒนานี้ไปใช้ในการเรียนการสอนในระดับบัณฑิตศึกษา รวมทั้งสามารถนำไปสร้างเป็นแบบเรียนให้กับนิสิตที่สนใจในรายวิชาภูมิอากาศวิทยาและบรรยากาศวิทยาได้อีกทาง อันจะเป็นการสร้างบุคลากรของประเทศให้มีองค์ความรู้ที่พร้อมนำไปปรับใช้กับหน่วยงานจริงของประเทศหลังจากได้จบการศึกษาแล้ว นอกจากนี้คณะผู้วิจัยยังได้นำข้อมูลเรดาร์ตรวจอากาศภาคพื้นดินมาวิเคราะห์สภาพภาพของเมฆฝนในช่วงพายุและร่องมรสุมอันสามารถเป็นต้นแบบในการเฝ้าระวังและติดตามพายุฝนที่รุนแรงในช่วงมรสุมที่อาจสร้างความเสียหายต่อชีวิตและทรัพย์สินของประชาชนได้เนื่องจากคณะผู้วิจัยได้พัฒนาโค้ดต้นแบบทั้งกระบวนการจนสามารถนำมาปรับใช้กับข้อมูลภาพเรดาร์ทุกสถานีของกรมอุตุนิยมวิทยา อันจะเป็นการเปิดแนวทางการศึกษาใหม่เพื่อสร้างองค์ความรู้ด้านกายภาพน้ำฝนที่ใช้ข้อมูลรีโมทเซนซิงให้กับประเทศต่อไป

นัฐพล มหาวิค

บทคัดย่อ

ข้อมูลฝนที่ตรวจวัดด้วยเรดาร์ตรวจอากาศภาคพื้นดินนั้นมีจุดเด่นในด้านความละเอียดเชิงพื้นที่และเวลาซึ่งเป็นข้อได้เปรียบข้อมูลน้ำฝนประมาณค่าที่ได้จากข้อมูลดาวเทียม แต่ในประเทศไทยนั้นยังมีการนำข้อมูลมาใช้ในการวิเคราะห์ผลกระทบทางด้านสภาพอากาศที่มีต่อพื้นที่น้อยมาก อาจเนื่องมาจากการเข้าถึงข้อมูลดังกล่าวมีจำกัดและองค์ความรู้ในการจัดการข้อมูลยังไม่มีเผยแพร่ ในการวิจัยครั้งนี้คณะผู้วิจัยได้ทำการพัฒนากระบวนการเพื่อเป็นต้นแบบในการทำแผนที่โมเสคน้ำฝนจากข้อมูลเรดาร์ครอบคลุมพื้นที่ทั้งหมด 5 สถานีในพื้นที่ลุ่มน้ำเจ้าพระยา เนื่องจากการเข้าถึงข้อมูลดิบของเรดาร์เหล่านั้นไม่มีข้อจำกัด ดังนั้นงานวิจัยชิ้นนี้จึงได้นำข้อมูลที่เข้าถึงได้อย่างเสรีหรือ Opendata นำมาเป็นปัจจัยนำเข้า ข้อมูลที่ใช้คือข้อมูลภาพค่าการสะท้อนของเรดาร์ตรวจอากาศภาคพื้นดินที่ทำการตรวจวัดทุกชั่วโมงจากผู้เชี่ยวชาญด้านเรดาร์ตรวจอากาศของกรมอุตุนิยมวิทยา ซึ่งภาพเหล่านี้ได้ถูกเผยแพร่ให้ประชาชนได้ใช้แบบใกล้เวลาจริง โดยในครั้งนี้ได้ใช้คลังข้อมูลของภาพเรดาร์ที่จัดเก็บโดยหน่วยงาน สถาบันสารสนเทศทรัพยากรน้ำและการเกษตร (องค์การมหาชน) นำมาเข้ากระบวนการประมวลผลภาพเชิงเลขโดยใช้ภาษาไพธอน เพื่อทำการเตรียมข้อมูลโดยข้อมูลที่ผ่านการตรวจสอบแต่ละสถานีจะถูกนำมาโมเสกให้เป็นข้อมูลนำเข้าที่มีพื้นที่ครอบคลุมพื้นที่ลุ่มน้ำเจ้าพระยาแล้วได้ทำการเปรียบเทียบายุและร่องมรสุมที่ส่งผลให้เกิดเมฆ convective ในพื้นที่ศึกษาในปี พ.ศ.2561 พบว่าเมฆฝนในช่วงพายุเขินติญมีคุณสมบัติทางด้านกายภาพที่บ่งชี้ถึงความรุนแรงของตัวพายุ ทำให้เห็นเป็นแนวทางในการใช้ข้อมูลเรดาร์หลายสถานีในการตรวจวัดและเฝ้าระวังพายุฝนที่รุนแรงในการเตือนภัยให้ภาคประชาชนได้ นอกจากนี้ผู้วิจัยยังได้พัฒนากระบวนการในการวัดคุณภาพของการโมเสกเรดาร์ด้วยการใช้ระยะทาง มุมยก ความสูงของการตรวจวัดและการบดบังของเรดาร์บีมเพื่อสร้างสมการอย่างง่ายที่สามารถนำไปใช้เป็นข้อมูลในการวิเคราะห์ต่อไป ซึ่งจะเป็นแนวทางต่อการต่อยอดพัฒนาผลิตภัณฑ์ให้กับกรมอุตุนิยมวิทยาเมื่อนำข้อมูลแบบสามมิติมาใช้ในการโมเสกเรดาร์ตรวจอากาศต่อไปในอนาคต

นัฐพล มหาวิค

สารบัญ

	หน้า
กิตติกรรมประกาศ	ก
คำนำ	ข
บทคัดย่อ	ค
สารบัญ	ง
สารบัญตาราง	จ
สารบัญรูป	ฉ
บทที่ 1 บทนำ	1-1
1.1 ที่มาและความสำคัญของปัญหาทางวิจัย	1-1
1.2 วัตถุประสงค์ของโครงการวิจัย	1-4
1.3 ขอบเขตของโครงการวิจัย	1-4
1.4 ระเบียบวิธีวิจัย	1-5
1.5 ประโยชน์ที่คาดว่าจะได้รับ	1-5
1.6 ระยะเวลาดำเนินการวิจัย	1-6
บทที่ 2 วรรณกรรม/สารสนเทศ (Information) ที่เกี่ยวข้อง	
2.1 แผนที่โมเสคข้อมูลเรดาร์ตรวจอากาศ	2-1
2.2 หลักการเบื้องต้นเรดาร์ตรวจอากาศภาคพื้นดิน	2-2
2.3 การทำแผนที่ค่าการสะท้อนจากเรดาร์	2-3
2.4 การทบทวนวรรณกรรม/สารสนเทศ (Information) ที่เกี่ยวข้อง	2-5
บทที่ 3 วิธีดำเนินการวิจัย	
3.1 ที่นที่ศึกษา	3-1
3.2 ข้อมูล	3-2
3.3 เครื่องมือ ซอฟต์แวร์	3-5
3.4 วิธีการ	3-7
บทที่ 4 ผลการศึกษา	
4.1 การพัฒนากระบวนการโมเสคเรดาร์ 5 สถานี	4-1
4.2 การพัฒนาดัชนีชี้วัดคุณภาพการโมเสคโดยใช้ค่าการสะท้อนจากภาพเรดาร์สองสถานี	4-25
บทที่ 5 สรุปผลการศึกษาและข้อเสนอแนะ	
5.1 สรุปผลการศึกษา	5-1
5.2 ข้อเสนอแนะ	5-2
เอกสารอ้างอิง	6-1
ภาคผนวก ก, ข, ค, ง	



สารบัญตาราง

	หน้า
ตาราง 3.1 รายละเอียดเรดาร์ตรวจอากาศ 3-2	3-2
ตาราง 3.2 รายละเอียดพายุที่ทำการวิเคราะห์ 3-5	3-5



สารบัญญภาพ

	หน้า
ภาพที่ 1.1 แผนที่ตำแหน่งที่ตั้งสถานีเรดาร์ตรวจอากาศของกรมอุตุนิยมวิทยา	1-2
ภาพที่ 2.1 สถานีเรดาร์ที่เก็บข้อมูลทั่วภูมิภาคอินโดจีน	2-1
ภาพที่ 2.2 ค่าการสะท้อนของกลุ่มฝนที่เกิดจากการเคลื่อนเข้าสู่ตอนกลางของอินโดจีน	2-2
ภาพที่ 2.3 แผนที่เรดาร์ภาคพื้นดินทั้ง 19 สถานีครอบคลุมประเทศญี่ปุ่น	2-4
ภาพที่ 2.4 อัลกอริทึมในการควบคุมคุณภาพของข้อมูลเรดาร์	2-6
ภาพที่ 2.5 ผลลัพธ์แผนที่คุณภาพจากการใช้อัลกอริทึมควบคุมคุณภาพ	2-7
ภาพที่ 3.1 พื้นที่ศึกษา	3-1
ภาพที่ 3.2 คลังข้อมูลภาพเรดาร์ตรวจวัดโดยกรมอุตุนิยมวิทยา	3-2
ภาพที่ 3.3 สถานีเรดาร์ตรวจอากาศของกรมอุตุนิยมวิทยาจังหวัดเชียงราย	3-3
ภาพที่ 3-4 สถานีเรดาร์ตรวจอากาศจังหวัดพิษณุโลกและอุพกรณ์	3-3
ภาพที่ 3.5 การแลกเปลี่ยนข้อมูลและการเยี่ยมชมอุปกรณ์สถานีเรดาร์ตรวจอากาศ	3-3
ภาพที่ 3.6 เปรียบเทียบข้อมูล SRTM DEM 3Arc	3-4
ภาพที่ 3.7 โปรแกรม Anaconda	3-5
ภาพที่ 3.8 โปรแกรม Spyder ที่ใช้ในการเขียนพัฒนาโปรแกรมไพธอนใน Anaconda	3-6
ภาพที่ 3.9 เปรียบเทียบ OpenCV	3-6
ภาพที่ 3.10 การพัฒนาโปรแกรมไพธอนโดยเรียกใช้ OpenCV	3-8
ภาพที่ 3.11 กระบวนการ Morphology ของ OpenCV	3-9
ภาพที่ 3.12 ฟังก์ชัน fitEllipse	3-10
ภาพที่ 3.13 ตัวอย่างการประยุกต์ใช้ฟังก์ชัน fitEllipse	3-10
ภาพที่ 3.14 แนวคิด fitEllipse	3-11
ภาพที่ 3.15 ลักษณะการกวาดตามุมยก (Elevation)	3-12
ภาพที่ 4.1 กรอบแนวคิดกระบวนการโมเสคข้อมูลภาพเรดาร์	4-1
ภาพที่ 4.2 การสกัดค่าเรดาร์การสะท้อนจากภาพของกรมอุตุนิยมวิทยา	4-2
ภาพที่ 4.3 โค้ดการสกัดค่าสีจากข้อมูลเรดาร์	4-3
ภาพที่ 4.4 การสร้างค่าพิกัดให้กับข้อมูลภาพเรดาร์	4-4
ภาพที่ 4.5 ผลลัพธ์การสกัดค่าเรดาร์การสะท้อน	4-5
ภาพที่ 4.6 โค้ดการตรวจหา radar flare	4-6
ภาพที่ 4.7 อัตราส่วนการบดบังของเรดาร์บีมจากการจำลองการเคลื่อนที่ของเรดาร์บีมจ.พิษณุโลก	4-7
ภาพที่ 4.8 อัตราส่วนการบดบังของเรดาร์บีมจากการจำลองการเคลื่อนที่ของเรดาร์บีมจ.เชียงราย	4-8
ภาพที่ 4.9 อัตราส่วนการบดบังของเรดาร์บีมจากการจำลองการเคลื่อนที่ของเรดาร์บีมจ.ลำพูน	4-10
ภาพที่ 4.10 อัตราส่วนการบดบังของเรดาร์บีมจากการจำลองการเคลื่อนที่ของเรดาร์บีมจ.จันทนา	4-11
ภาพที่ 4.11 อัตราส่วนการบดบังของเรดาร์บีมจากการจำลองการเคลื่อนที่ของเรดาร์บีมสุวรรณภูมิ	4-11
ภาพที่ 4.12 โค้ดไพธอนวิเคราะห์อัตราส่วนพื้นที่บดบังเรดาร์บีมของสถานีเรดาร์ 5 สถานี	4-12
ภาพที่ 4.13 จำนวนแผนที่การบดบังเรดาร์บีมในแต่ละอัตราส่วนในรูปแบบเปอร์เซ็นต์	4-13
ภาพที่ 4.14 กรอบแนวคิดการพัฒนาโค้ดเพื่อคำนวณแผนที่การบดบังเรดาร์บีม	4-14

ภาพที่ 4.15 ได้ดการคำนวณการบดบังเรดาร์บีมในระบบพิกัดกริด	4-15
ภาพที่ 4.16 ได้ดการคำนวณพื้นที่บีมที่บดบัง	4-16
ภาพที่ 4.17 ได้ดโมเสคผลการคำนวณพื้นที่เรดาร์บีมที่ถูกภูมิประเทศบดบัง	4-17
ภาพที่ 4.18 ผลิตภันท์เรดาร์โมเสคทั้งห้าสถานี	4-18
ภาพที่ 4-19 ได้ดการโมเสคเรดาร์ค่าการสะท้อน	4-19
ภาพที่ 4-20 สถิติเปรียบเทียบระหว่างสี่เหตุการณ์ฝนในด้านความสมบูรณ์ของข้อมูลที่ใช้	4-20
ภาพที่ 4.21 สถิติเปรียบเทียบจำนวนของเมฆฝน	4-20
ภาพที่ 4.22 ค่าเฉลี่ยรายชั่วโมงลักษณะของเมฆฝน convective ทั้ง 4 เหตุการณ์	4-21
ภาพที่ 4.23 ได้ดการค้นหาเมฆฝนแบบ convective	4-22
ภาพที่ 4.24 ได้ดการค้นหาเมฆฝนแบบ convective (ต่อ)	4-22
ภาพที่ 4.25 สถิติจำแนกเปรียบเทียบ จำนวน ขนาด และความยาวของเมฆฝนแบบ convective	4-23
ภาพที่ 4.26 ผลการโมเสคค่าการสะท้อนเรดาร์	4-26
ภาพที่ 4.27 ข้อมูลที่ใช้ในแต่ละตำแหน่งของเรดาร์โมเสค	4-27
ภาพที่ 4.28 เรดาร์โมเสคการบดบังภูมิประเทศของการจำลองเรดาร์บีมในมมยกรแรก	4-28
ภาพที่ 4.29 ได้ดโมเสคเรดาร์โมเสคการบดบังภูมิประเทศของการจำลองเรดาร์บีมในมมยกรแรก	4-29
ภาพที่ 4.30 การตรวจพบ radar flares	4-30
ภาพที่ 4.31 แผนที่คุณภาพเรดาร์	4-31
ภาพที่ 4.32 ผลลัพธ์ Radar quality index	4-32

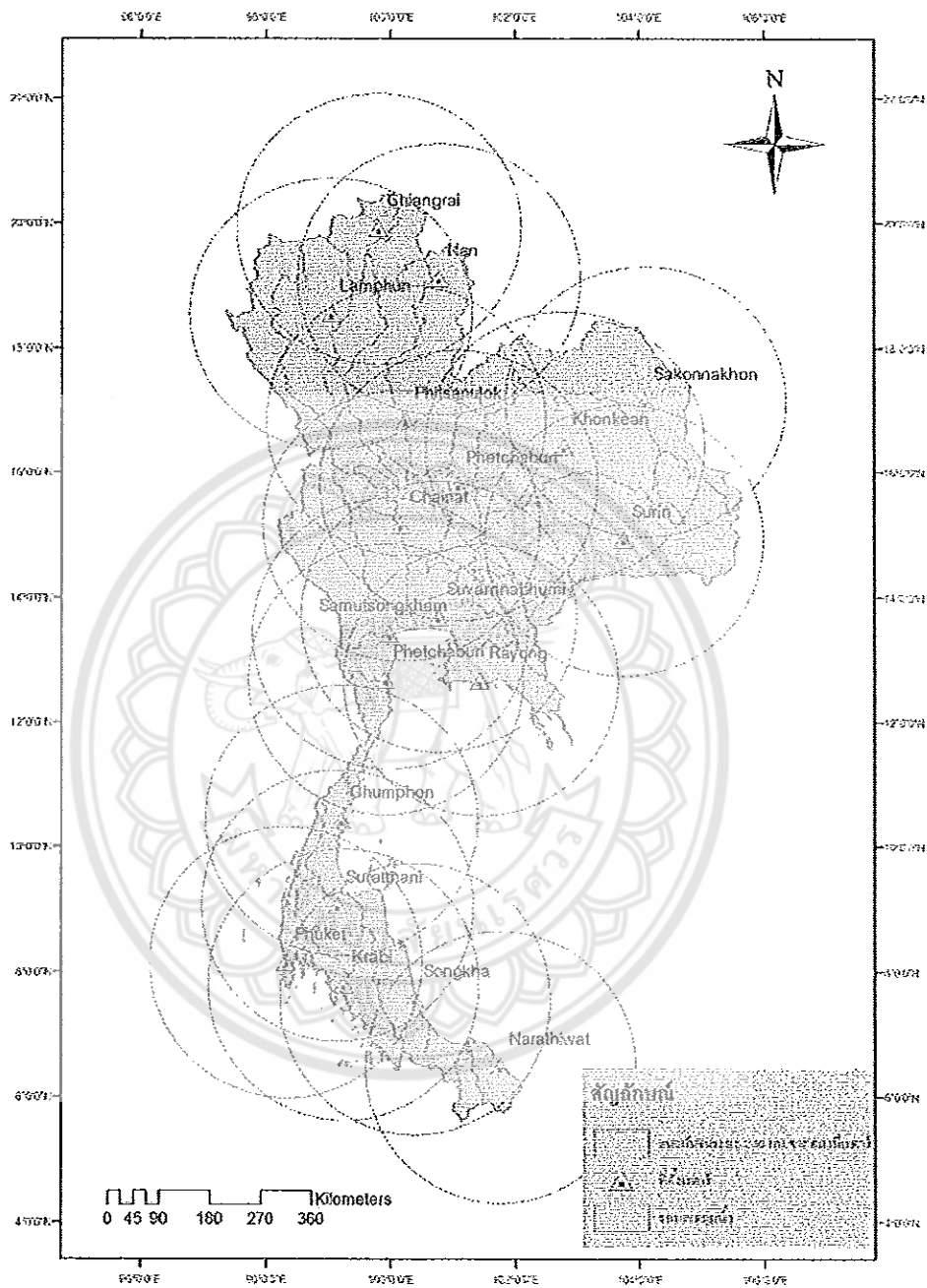
บทที่ 1 บทนำ

1.1 ที่มาและความสำคัญของปัญหาทางานวิจัย

เหตุการณ์อุทกภัยครั้งใหญ่ในปี 2554 ในพื้นที่ลุ่มน้ำเจ้าพระยาของประเทศไทย เป็นเหตุการณ์ที่มีลำดับความต่อเนื่องในช่วงระหว่างเดือนสิงหาคม ถึง ธันวาคม พ.ศ. 2554 สร้างความเสียหายหลายประการต่อประเทศไม่ว่าจะเป็น เศรษฐกิจ พื้นที่การเกษตร การสูญเสียชีวิตที่นับเป็นจำนวนทั้งสิ้น 813 คนและผู้สูญหาย 3 ราย (กระทรวงมหาดไทย, 2555) จากรายงานการประเมินความเสียหายของธนาคารโลกแบ่งเป็น ความเสียหายด้านทรัพย์สินและค่าเสียโอกาสที่ 660 และ 700 พันล้านบาท ตามลำดับ ทำให้เกิดการชะลอตัวของสภาพเศรษฐกิจและผลกระทบวงกว้างต่อประเทศ รวมไปถึงผลกระทบแบบลูกโซ่ต่อเศรษฐกิจโลกสืบเนื่องมาจากมีบริษัทที่เป็นส่วนหนึ่งของห่วงโซ่อุปทานในสายพานอุตสาหกรรมโลกตั้งโรงงานผลิตรวมทั้งสิ้น 804 บริษัท กระจายอยู่ในนิคม 7 แห่ง ได้รับผลกระทบจากน้ำท่วมในลุ่มน้ำเจ้าพระยารั้งนี้เป็นอย่างมาก

จากการศึกษาของ Komori et al. (2012) พบว่าน้ำท่วมในลุ่มน้ำเจ้าพระยานั้นเกิดจากการมีปริมาณน้ำฝนที่มากเกินไปถึง 143% ในช่วงฤดูฝนในปี 2554 โดยใช้ข้อมูลน้ำฝนที่สำรวจได้ ณ สถานีภาคพื้นดินและข้อมูลอัตราการไหลของกระแสน้ำมาศึกษาซึ่งเป็นข้อมูลที่ไม่มีความต่อเนื่องเชิงพื้นที่ (discrete spatial data) ดังนั้นทำให้เห็นว่าการศึกษารองการกระจายปริมาณน้ำฝนเชิงพื้นที่เพื่อเข้าใจถึงรูปแบบน้ำฝนเชิงพื้นที่และช่วงเวลาในลุ่มน้ำเจ้าพระยามีความสำคัญต่อการพัฒนาประเทศเป็นอย่างมาก ซึ่งหากมีข้อมูลเชิงพื้นที่แบบต่อเนื่อง (continuous spatial data) ที่มีความถูกต้องสูงแล้วนำมาวิเคราะห์เพื่อระบุตำแหน่งและสาเหตุ จะช่วยลดการสูญเสียชีวิตจากปัญหาอุทกภัยในพื้นที่เปราะบางได้ดีกว่าข้อมูลที่มีในปัจจุบัน แต่อุปสรรคสำคัญต่อการทำความเข้าใจลักษณะของฝนเชิงพื้นที่ของประเทศไทยนั้นคือ การขาดข้อมูลปริมาณน้ำฝนที่มีความถูกต้องสูงในประเทศที่พัฒนาแล้วใช้ในการวิเคราะห์สถานการณ์น้ำฝนเพื่อใช้ในการศึกษาวิจัยเชิงลึกเพื่อจะเข้าใจต่อปรากฏการณ์กายภาพน้ำฝนในอดีต

ข้อมูลปริมาณฝนภาคพื้นดินที่มีความถูกต้องสูงในเชิงปริมาณนั้นปัจจุบันเป็นที่ยอมรับกันว่าปริมาณฝนที่วัดจากเครื่องมือวัดน้ำฝนภาคพื้นดินให้ความถูกต้องสูงที่สุด แต่ก็ไม่สามารถอธิบายความต่อเนื่องของฝนเชิงพื้นที่ได้เนื่องจากเป็นข้อมูลแบบไม่ต่อเนื่องเชิงพื้นที่ ส่วนข้อมูลฝนเชิงพื้นที่ที่ได้จากดาวเทียมนั้นเป็นข้อมูลที่มีประโยชน์ที่สามารถมองเห็นได้ในบริเวณกว้างในระดับภูมิภาค เช่น ดาวเทียมวงโคจรค้างฟ้า (satellite stationary orbit) ที่โคจรอยู่สูงขึ้นไปประมาณสามหมื่นกว่ากิโลเมตรทำให้



ภาพที่ 1.1 แผนที่ตำแหน่งที่ตั้งสถานีเรดาร์ตรวจอากาศของกรมอุตุนิยมวิทยาและขอบเขตการตรวจอากาศเป็นรัศมีเท่ากับ 240 กิโลเมตร

สามารถถ่ายภาพได้ในความถี่ที่สูงในรัศมีมากกว่า 1 ชั่วโมงต่อหนึ่งภาพ แต่ปัญหาในเรื่องการวัดปริมาณฝนจากข้อมูลดาวเทียมนี้คือ ปริมาณที่คลาดเคลื่อนไปจากความเป็นจริงสืบเนื่องจากการวัดประมาณค่าปริมาณน้ำฝนจากค่าของอุณหภูมิยอดเมฆที่วัดได้ (Brightness temperature) โดยมีสมมติฐานว่าหากอุณหภูมิยอดเมฆที่สูงจะทำให้ค่าอุณหภูมิต่ำแล้วจะสอดคล้องกับปริมาณน้ำฝนที่ประมาณได้จากภาคพื้นดิน แต่ก็ไม่เสมอไปว่าอุณหภูมิเมฆที่ต่ำจะเป็นเมฆที่ให้น้ำฝนซึ่งเป็นการได้มาซึ่งข้อมูลที่ไม่ตรง (indirect observation) เท่าไรนัก ยิ่งเมื่อพบกับเมฆอุ่น (warm rain) ที่เกิดการกลั่นตัวเป็นหยดน้ำอาจจะเนื่องมาจากการยกตัวของก้อนเมฆตามลักษณะภูมิประเทศเกิดเป็นฝนภูเขา (orographic rainfall) ทำให้ค่าฝนที่ประมาณได้จากดาวเทียมกับฝนภาคพื้นดินมีโอกาสสูงที่จะไม่สอดคล้องกัน

ปัจจุบันนานาอารยประเทศยอมรับว่าการวัดปริมาณน้ำฝนจากค่าสะท้อนของคลื่นเรดาร์ที่ถูกปล่อยออกไปจากเรดาร์ตรวจอากาศภาคพื้นดินที่ไปกระทบกับกลุ่มฝนในก้อนเมฆนั้นเป็นการประมาณค่าน้ำฝนเชิงพื้นที่ที่มีประสิทธิภาพค่อนข้างสูง เมื่อนำมาประมวลผลโดยการแปลงค่าสะท้อนเป็นค่าปริมาณน้ำฝนอย่างถูกต้องแล้วจะทำให้ได้ข้อมูลปริมาณน้ำฝนเชิงปริมาตรที่เป็นตัวแทนของข้อมูลน้ำฝนเชิงพื้นที่ (spatial data) ได้เป็นอย่างดี โดยกรมอุตุนิยมวิทยาของประเทศไทยได้ให้บริการข้อมูลนี้อยู่เพื่อใช้ในการตรวจสอบสภาพอากาศในแต่ละชั่วโมงทั่วประเทศซึ่งอยู่ในรูปแบบภาพค่าการสะท้อนของกลุ่มฝนอย่างไรก็ตามการประมาณค่าน้ำฝนจากค่าการสะท้อนของเรดาร์ก็ยังมีโอกาสที่จะได้ปริมาณน้ำฝนที่คลาดเคลื่อนไปจากฝนภาคพื้นดิน (e.g. Brandes 1975; Anagnostou et al. 1998) เนื่องมาจากปัจจัยหลายประการ อาทิเช่น ชนิดของฝน ความละเอียดของการตรวจวัดทั้งเชิงพื้นที่และเวลา กระแสลม ความสัมพันธ์ที่ใช้ในการแปลง เป็นต้น ซึ่งการได้มาซึ่งข้อมูลปริมาณน้ำฝนจากเรดาร์ตรวจอากาศภาคพื้นดินนั้นจะตั้งอยู่บนสมมติฐานแทบทั้งสิ้นเพื่อให้เหมาะกับการประมาณค่า ซึ่งในประเทศที่พัฒนาแล้วได้นำการประมาณค่าปริมาณน้ำฝนที่ได้จากเรดาร์ซึ่งมีข้อได้เปรียบเชิงพื้นที่มาทำการปรับแก้ความคลาดเคลื่อนกับข้อมูลน้ำฝนสถานี ทำให้ได้ข้อมูลที่มีความถูกต้องสูงหลังการปรับแก้ โดยข้อมูลเรดาร์นั้นได้ทำเชื่อมต่อหลายสถานีเข้าด้วยกันเพื่อให้เป็นข้อมูลผืนเดียวกันอย่างเป็นระบบ ตัวอย่างเช่น Radar AmdAS (Makihara et al., 1996) ของญี่ปุ่น หรือ The Next Generation Weather Radar (NEXRAD) ของอเมริกา (e.g. Fulton et al. 1998) เป็นต้น ที่นักวิจัย นักวิชาการและประชาชนทั่วไปสามารถเข้าถึงได้เพื่อนำข้อมูลเหล่านี้ไปใช้ประโยชน์ในงานที่เกี่ยวข้อง

จากภาพที่ 1.1 พบว่าโครงข่ายเรดาร์ของกรมอุตุนิยมวิทยาที่มีทั้งหมด 18 สถานีตามข้อมูลที่ให้บริการแก่ประชาชนผ่านทางเว็บไซต์ของกรมอุตุนิยมวิทยา ครอบคลุมพื้นที่ทั้งประเทศไทยคืออยู่แล้ว ซึ่งลุ่มน้ำเจ้าพระยาก็ครอบคลุมด้วยเรดาร์อย่างเหมาะสมไปด้วย ดังนั้นในงานวิจัยชิ้นนี้คณะนักวิจัยซึ่งประกอบไปด้วยหน่วยงานจากภาคมหาวิทยาลัยและหน่วยงานของกรมอุตุนิยมวิทยาแห่งประเทศไทยมีความ

ตั้งใจจะพัฒนาต้นแบบของกระบวนการผลิตข้อมูลแผนที่น้ำฝนจากเรดาร์ตรวจอากาศภาคพื้นดินโดย
ทำพัฒนากระบวนการโมเสคของข้อมูลน้ำฝนประมาณค่าที่ได้จากค่าการสะท้อนของกลุ่มฝนจาก
เรดาร์ตรวจอากาศภาคพื้นดินครอบคลุมพื้นที่ลุ่มน้ำเจ้าพระยา และเพื่อจัดทำเป็นฐานข้อมูลเชิงพื้นที่
ในช่วงเวลาฤดูมรสุมของประเทศไทย โดยงานวิจัยชิ้นนี้ถือว่าเป็นชิ้นแรกที่จะได้ทำการศึกษ
กระบวนการทำแผนที่โมเสคน้ำฝนจากข้อมูลเรดาร์ตรวจอากาศหลายสถานีในลุ่มน้ำเจ้าพระยาแห่งนี้
อย่างเป็นระบบ

1.2 วัตถุประสงค์ของโครงการวิจัย

1. ศึกษาและพัฒนากระบวนการพัฒนาแผนที่น้ำฝนจากเรดาร์ตรวจอากาศภาคพื้นดินในพื้นที่ลุ่มน้ำ
เจ้าพระยาในช่วงฤดูมรสุมเขตร้อน
2. พัฒนาฐานข้อมูลน้ำฝนเชิงพื้นที่จากผลลัพธ์ของกระบวนการข้อที่1ให้เหมาะสมกับการนำไปใช้ใน
งานวิเคราะห์ปริมาณน้ำฝนเชิงพื้นที่ของระบบลุ่มน้ำ

1.3 ขอบเขตของโครงการวิจัย

1.3.1 กระบวนการพัฒนาแผนที่น้ำฝนจากเรดาร์ตรวจอากาศ

เนื่องจากพื้นที่ครอบคลุมการสังเกตการณ์ของเรดาร์ตรวจอากาศภาคพื้นดินมีรัศมีอยู่ที่ 240 กิโลเมตร
ทำให้ในพื้นที่ลุ่มน้ำเจ้าพระยานั้นมีเรดาร์ครอบคลุมอยู่หลายสถานี ซึ่งทำให้ต้องมีการแปลงข้อมูลจาก
การตรวจอากาศแบบเชิงรัศมี (radial observation) เป็นระบบแผนที่แบบคาร์ทีเซียนกริด
(Cartesian grid) เพื่อให้ง่ายต่อการนำไปเชื่อมต่อกับเรดาร์สถานีถัดไปเรียกว่า การโมเสคข้อมูลการ
ประมาณค่าน้ำฝนจากเรดาร์ตรวจอากาศภาคพื้นดินโดยในการศึกษาครั้งนี้จะทำการศึกษากการโมเสค
แบบ2มิติในเชิงราบ (Horizontal dimensions) เพื่อเป้าหมายการผลิตแผนที่การประมาณค่าน้ำฝน
ซึ่งหลังจากทำการศึกษาวิธีการในการโมเสคข้อมูลน้ำฝนแล้วจะต้องทำการวัดสอบค่าความถูกต้อง
(validation of estimated rainfall products) ซึ่งจะใช้ข้อมูลฝนสถานีภาคพื้นดินของกรม
อุตุนิยมวิทยาทำการเทียบสอบความถูกต้องออกมาในรูปของค่าสถิติ

1.3.2 พื้นที่ศึกษา

เนื่องจากเป็นงานวิจัยเชิงต้นแบบ คณะผู้วิจัยได้เลือกเอาเรดาร์ตรวจอากาศที่ได้มีการติดตั้งใหม่ของ
กรมอุตุนิยมวิทยา มาศึกษา ซึ่งจะประกอบไปด้วยเรดาร์สถานีพิษณุโลกและชัยนาท ที่ตรวจวัดสภาพ
อากาศด้วยความถี่ในระบบซีแบนด์ (C-band frequency) ทำการตรวจค่าทุกๆ 15 นาที
ประกอบด้วย 4 มุมยก (Plan Position Indicator: PPI) ครอบคลุมพื้นที่ 240 กิโลเมตรตาม
ระยะทางของแนวรัศมีออกไปจากสถานีเรดาร์ตรวจอากาศ อย่างไรก็ตามการกรมอุตุนิยมวิทยายังไม่
อนุญาตให้มีการเข้าถึงข้อมูลดิบซึ่งถือว่าเป็นระบบไฟล์รหัสปิดที่ทางบริษัทผู้จำหน่ายเรดาร์ไม่ได้

เผยแพร่โค้ดต้นฉบับให้นักวิจัยได้ทำการวิจัยต่อยอด ดังนั้นในงานวิจัยครั้งนี้จึงใช้ข้อมูลประเภท opendata ที่ประชาชนทุกคนสามารถเข้าถึงข้อมูลได้ ในรูปแบบของไฟล์ภาพซึ่งต้องพัฒนากระบวนการสกัดค่าการสะท้อนรวมถึงการพัฒนาอัลกอริทึมในการโมเสกข้อมูลเรดาร์

1.3.2 ระยะเวลาที่ศึกษา

ฤดูฝนตามบริเวณตอนกลางของอินโดจีนที่ศึกษาโดย Matsumoto et al. (1997) ระบุว่าฤดูฝนจะเริ่มต้นตั้งแต่กลางพฤษภาคมไปจนถึงปลายกันยายนซึ่งจะเรียกว่าฤดูมรสุมเขตร้อน (Tropical monsoon season) ทำให้ผู้วิจัยจะใช้ข้อมูลในช่วงฤดูมรสุมโดยเลือกปี 2561 ซึ่งเป็นปีที่มีพายุพัดผ่านตอนกลางของอินโดจีนนั่นคือ พื้นที่ลุ่มน้ำเจ้าพระยาเป็นหลักในการศึกษาครั้งนี้

1.4 ระเบียบวิธีวิจัย

- 1 ทบทวนวรรณกรรมที่เกี่ยวข้อง
- 2 รวบรวมข้อมูลข้อมูลภาพเรดาร์ตรวจอากาศภาคพื้นดินจากกรมอุตุนิยมวิทยา
- 3 ตรวจสอบความถูกต้องข้อมูลจากกรมอุตุนิยมวิทยา
- 4 ประมวลผลข้อมูลเรดาร์เบื้องต้น
- 5 ออกภาคสนามเพื่อสำรวจสถานีเรดาร์
- 6 ออกแบบและพัฒนาการอัลกอริทึมเพื่อโมเสกแผนที่เรดาร์ค่าการสะท้อน

1.5 ประโยชน์ที่คาดว่าจะได้รับ

โครงการวิจัยนี้มีเป้าประสงค์เพื่อเป็นต้นแบบในการจัดทำแผนที่เรดาร์โมเสกที่ได้มาจากข้อมูลการตรวจวัดอากาศจากเรดาร์ตรวจอากาศภาคพื้นดิน ซึ่งผลลัพธ์ที่ได้คือกระบวนการในการสร้างข้อมูลต้นแบบนี้ที่จะได้มาจากการพัฒนาหลังจากงานวิจัยชิ้นนี้สำเร็จ รวมทั้งฐานข้อมูลน้ำฝนเชิงพื้นที่ที่สามารถนำไปใช้ในการศึกษาปรากฏการณ์การกระจายตัวของปริมาณน้ำฝนในลุ่มน้ำเจ้าพระยา ผู้ที่ได้รับผลประโยชน์โดยตรงคือ นักวิชาการในกรมอุตุนิยมวิทยา กรมชลประทาน กรมพัฒนาที่ดิน กรมป่าไม้ เป็นต้น รวมไปถึงนักวิจัยที่จะนำกระบวนการและฐานข้อมูลเชิงพื้นที่ไปศึกษาในแต่ละปรากฏการณ์ ซึ่งคาดว่าจะเริ่มต้นในการขยายผลไปสู่การสร้างผลิตภัณฑ์น้ำฝนในระดับประเทศ โดยในท้ายที่สุดจะนำไปสู่การออกนโยบายซึ่งเป็นวาระแห่งชาติในการสร้างผลิตภัณฑ์น้ำฝนจากเรดาร์แบบใกล้เวลาจริง (near real time of gridded precipitation data) ดังที่มีให้บริการในประเทศที่พัฒนาแล้ว โดยกำหนดเป้าประสงค์ที่วัดได้ดังต่อไปนี้

- 1 การผลิตบทความวิจัยจำนวน 1 บทความเพื่อตีพิมพ์ในวารสารวิชาการนานาชาติ
- 2 การนำเสนอบทความวิชาการในการประชุมทางวิชาการในประเทศ 1 ครั้ง

- 3 การพัฒนานักวิจัยรุ่นใหม่ โดยโครงการนี้มีผู้เชี่ยวชาญเรื่องอุทก-ภูมิอากาศวิทยาเป็นนักวิจัยที่เลี้ยงซึ่งมีประสบการณ์ในการทำงานด้านการวิเคราะห์ข้อมูลน้ำฝนเชิงพื้นที่มาอย่างเชี่ยวชาญ และยังมีประสบการณ์ร่วมงานร่วมวิจัยกับนักวิจัยในกรมอุตุนิยมวิทยาและกรมชลประทานที่จะคอยให้คำแนะนำปรึกษาได้ตรงตามความต้องการของหน่วยงาน
- 4 การพัฒนานิสิตทั้งระดับปริญญาตรีและโทในฐานะผู้ช่วยนักวิจัย เพื่อเรียนรู้กระบวนการวิจัยด้านการประมวลผลข้อมูลเรดาร์ตรวจอากาศภาคพื้นดินเพื่อจัดทำเป็นฐานข้อมูลเชิงพื้นที่อย่างเป็นระบบ
- 5 การส่งเคราะห์งานวิจัยให้เป็นที่ศึกษา เพื่อนำไปเป็นหนึ่งในบทเรียนของตำราวิชาภูมิอากาศวิทยาและการรับรู้ระยะไกลในระดับปริญญาตรี

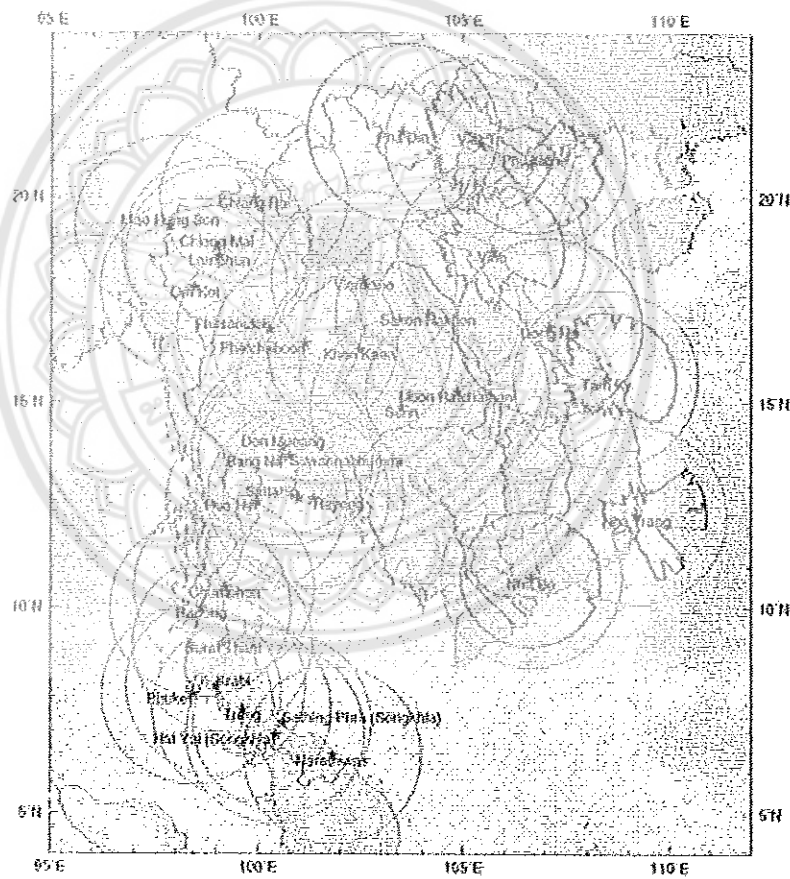
1.6 ระยะเวลาดำเนินการวิจัย

ปี	กิจกรรม	ค.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.	มิ.ย.	ก.ค.	ส.ค.	ก.ย.
2560	ทบทวนวรรณกรรมที่เกี่ยวข้อง	X	X	X									
2560	รวบรวมข้อมูลน้ำฝนสถานีและข้อมูลเรดาร์ตรวจอากาศ	X	X	X	X	X							
2561	ตรวจสอบความถูกต้องข้อมูล				X	X	X						
2561	ประมวลผลข้อมูลเรดาร์เบื้องต้น			X	X	X	X	X	X				
2561	ออกภาคสนาม						X	X	X				
2561	ออกแบบและพัฒนาการโมเสคแผนที่							X	X	X	X		
2561	จัดอบรมสัมมนาเผยแพร่องค์ความรู้											X	
2561	จัดทำรายงานฉบับสมบูรณ์												X

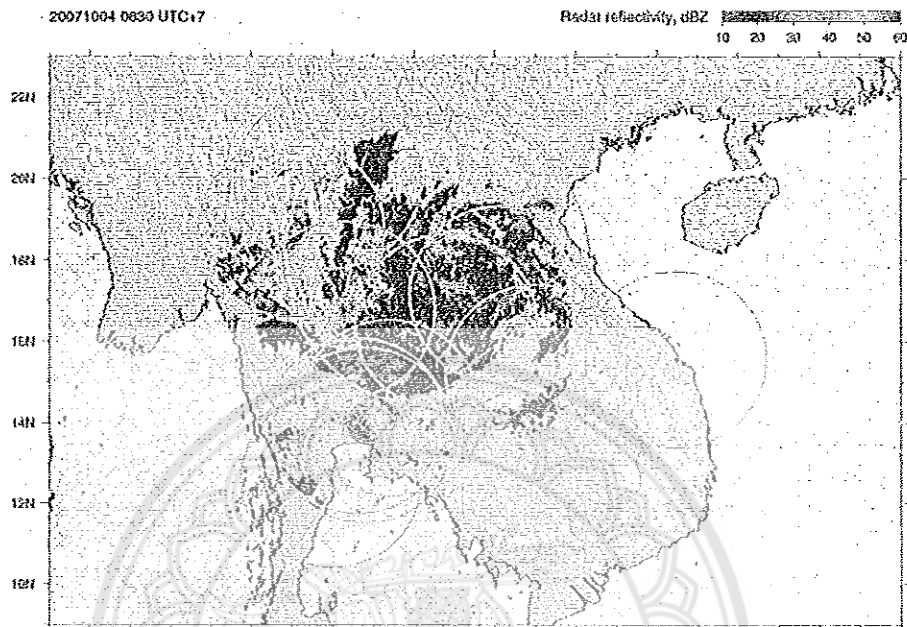
บทที่ 2 วรรณกรรม/สารสนเทศ (Information) ที่เกี่ยวข้อง

2.1 แผนที่โมเสคข้อมูลเรดาร์ตรวจอากาศ

ข้อมูลเรดาร์ตรวจอากาศถูกใช้ในประเทศกลุ่มอินโดจีนมาหลายทศวรรษแล้วรวมทั้งประเทศไทย (ภาพ 1.1) แต่ยังไม่ได้มีการนำมาโมเสคสร้างเป็นผลิตภัณฑ์ในรูปแบบเรียลไทม์เหมือนประเทศที่พัฒนาแล้ว Satomura et al. (2013) ได้สร้างข้อมูลเรดาร์โมเสคที่ถูกนำมาใช้ในงานวิจัยด้านอุตุนิยมวิทยาในภูมิภาคอินโดจีนเป็นกลุ่มแรกๆ โดยทำการทำโมเสคเรดาร์ในภูมิภาคอินโดจีนในช่วงที่พายุ Lekima สลายตัวเพื่อทำความเข้าใจเกี่ยวกับโครงสร้างของการสลายตัวของพายุ ดังภาพที่ 2.1



ภาพที่ 2.1 สถานีเรดาร์ที่เก็บข้อมูลทั่วภูมิภาคอินโดจีน แสดงเป็นลักษณะวงกลมรัศมีการสังเกตการณ์ของเรดาร์ตรวจอากาศ



ภาพที่ 2.2 ค่าการสะท้อนของกลุ่มฝนที่เกิดจากการเคลื่อนเข้าสู่ตอนกลางของอินโดจีนจากพายุ Lekima ในเดือนตุลาคม 2007

2.2 หลักการเบื้องต้นเรดาร์ตรวจอากาศภาคพื้นดิน

เรดาร์ (RADAR: Radio Detection And Ranging) ได้ถูกค้นพบในศตวรรษที่ 18 แล้วได้ถูกนำมาใช้
ในด้านการทหารในช่วงสงครามโลกครั้งที่สอง โดยมีหลายประเทศที่พัฒนาระบบและเครื่องมือเรดาร์
เพื่อเสริมสมรรถนะในด้านการรบให้กับกองทัพของตัวเองได้แก่ ประเทศสหรัฐอเมริกา อังกฤษ
ฝรั่งเศส อิตาลี เยอรมนี เนเธอร์แลนด์ รัสเซียและญี่ปุ่น หลังจากสงครามโลกครั้งที่สองเป็นต้นมา
ระบบเรดาร์ได้ถูกนำมาใช้ในด้านการตรวจอากาศ (Reinhart, 1997)

เรดาร์ใช้หลักการสังเกตการณ์ข้อมูลน้ำฝนในรูปแบบของการสัมผัสระยะไกล (Remote sensing) ใน
ที่นี้จะกล่าวถึงหลักการอย่างย่อเนื่องจากสามารถศึกษาได้จากตำราหลายแห่ง (e.g., Battan 1973;
Sauvaugot 1992; Doviak and Zrinc 1993; Rinehart 1997) โดยพลังงานแม่เหล็กไฟฟ้าจะถูก
ส่งออกไปจากจานเรดาร์ในช่วงคลื่นของไมโครเวฟโดยมีความยาวช่วงคลื่นอยู่ระหว่าง 1-10 ซม. ที่
ความถี่จาก 3 ถึง 30 GHz แล้วจะถูกสะท้อนกลับเมื่อเจอกลุ่มฝนเข้าสู่ตัวรับสัญญาณในรูปแบบความ
เข้มของพลังงาน โดยที่การความเข้มของพลังงานที่สะท้อนกลับจะขึ้นอยู่กับ ปริมาณกลุ่มฝน ขนาด
ของเม็ดฝนที่พลังงานแม่เหล็กไฟฟ้าไปกระทบซึ่งจะถูกวัดในรูปแบบของปริมาตร โดยค่าการสะท้อน
กลับของพลังงานนั้นสามารถคำนวณได้จาก

$$Z = \frac{P_B \cdot r^2}{C \cdot k^2}$$

โดยที่

Z	[mm ⁶ /m ³]	ค่าการสะท้อนของวัตถุเป้าหมายในรูปแบบปริมาตร
P _B	[W]	พลังงานในการสะท้อนกลับของคลื่นแม่เหล็กไฟฟ้า
r	[m]	ระยะทางของวัตถุเป้าหมาย
C	[Wm ⁵ /mm ⁶]	ค่าสัมปสิทธิ์เรดาร์
k	[-]	ค่าคงที่ของการสะท้อนกลับของเม็ดฝน

โดยค่า k เป็นค่าเชิงประจักษ์ที่อยู่ในช่วง k=0.964 สำหรับเม็ดฝน ไปจนถึง k=0.456 สำหรับหิมะ
ค่า Z โดยปรกติแล้วจะใช้หน่วย dBZ แทน [mm⁶/m³]

$$dBZ = 10 \log_{10}(Z)$$

ในเชิงปฏิบัตินั้นค่าการสะท้อนจากวัตถุเป้าหมาย (Z) ได้ถูกนำไปหาความสัมพันธ์กับค่าฝนจริงที่วัดได้
ในเชิงประจักษ์นำไปสู่ความสัมพันธ์ที่เรียกว่า Z-R คือความสัมพันธ์ระหว่างค่าการสะท้อนพลังงาน
แม่เหล็กไฟฟ้ากับปริมาณน้ำฝนที่วัดได้จริง

$$Z = A \cdot R^B$$

โดยที่

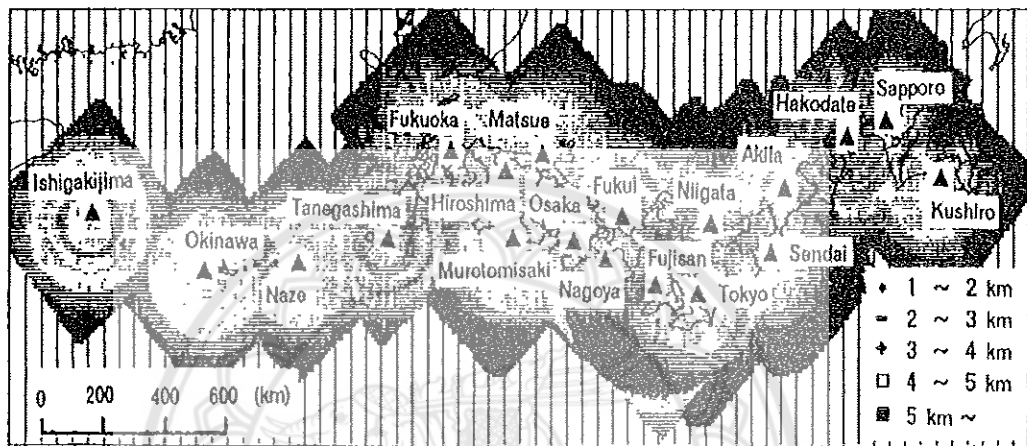
Z	[mm ⁶ /m ³]	ค่าการสะท้อนของวัตถุเป้าหมายในรูปแบบปริมาตร
R	[mm/h]	ค่าความเข้มของฝน
A	[-]	ค่าคงที่เชิงประจักษ์
B	[-]	ค่าคงที่เชิงประจักษ์

ค่าคงที่ A กับ B จะผันแปรไปตามปรากฏการณ์ของฝน ตามขนาด ชนิดของเม็ดฝน โดยจะได้อาจ
การเทียบค่าเฉลี่ยระหว่างความเข้มของฝนภาคพื้นดินกับค่าการสะท้อนพลังงานของกลุ่มฝน (e.g.
Brandes 1975; Anagnostou et al. 1998) ค่าความสัมพันธ์ Z-R ได้มีการกล่าวครั้งแรกโดย
Marshall and Palmer (1948) ซึ่งเป็นความสัมพันธ์ที่ได้มาจากฝนแบบ Stratiform คือ
 $Z = 200R^{1.6}$ ใช้กันอย่างแพร่หลายเช่น หน่วยงานกรมอุตุนิยมวิทยา ประเทศอังกฤษ และกรม
อุตุนิยมวิทยาในประเทศไทย

2.3 การทำแผนที่ค่าการสะท้อนจากเรดาร์

การสร้างแผนที่ค่าการสะท้อนจากเรดาร์นั้นสามารถนำข้อมูลสร้างจากข้อมูล PPI ซึ่งสามารถเลือกเอา
มุมยกใดก็ได้มาทำการประมาณค่าเพื่อสร้างแผนที่เรดาร์น้ำฝน หรือการใช้ข้อมูลที่ได้จากหลายมุมยก

PPIs (3D scan) นำมาสร้างแผนที่ค่าการสะท้อนแบบกำหนดความสูง CAPPI (Constant Altitude Plan Position Indicator) ซึ่งใช้กันอย่างแพร่หลาย (e.g. Fulton et al. 1998; Seo et al. 2010) ซึ่งในประเทศสหรัฐอเมริกาได้มีการนำหลักการ



ภาพที่ 2.3 แผนที่เรดาร์ภาคพื้นดินทั้ง 19 สถานีครอบคลุมประเทศญี่ปุ่นซึ่งแสดงตามระดับค่าความสูงตามสัญลักษณ์เหนือระดับน้ำทะเลปานกลาง ความสูงต่ำที่สุดจะถูกเลือกมาเป็นตัวแทนค่าความแรงของการประมาณค่าน้ำฝน

การสร้างแผนที่ค่าการสะท้อนโดยกำหนดความสูงของข้อมูลเรดาร์ ด้วยการให้ค่าน้ำหนักด้วยวิธีการเคอร์เนลฟังก์ชัน (Kernel function) ซึ่งค่าน้ำหนักจะถูกกำหนดให้กับค่าการสะท้อนของแต่ละมุมยกที่ถูกนำมาสร้างเป็นแผนที่ค่าการสะท้อน ค่าน้ำหนักที่ได้จากเคอร์เนลฟังก์ชันจะถูก normalization ให้มีค่ารวมเท่ากับ 1 ซึ่งวิธีการนี้จะช่วยให้แผนที่ค่าการสะท้อนน้ำฝนจากเรดาร์นั้นมีความต่อเนื่องในกรณีที่ต้องใช้ข้อมูลหลายมุมยก

จากภาพที่ 2.3 เป็นการนำเรดาร์ของกรมอุตุนิยมวิทยาประเทศญี่ปุ่น (Japan Meteorological Agency: JMA) มาทำการโมเสค (Makihara et al. 1996) ซึ่งรวมทั้ง 19 เรดาร์ ซึ่งใช้โครงข่ายของสถานีน้ำฝนภาคพื้นดินที่มีความหนาแน่นสูงในการช่วยปรับแก้ค่าประมาณค่าน้ำฝนจากเรดาร์ตรวจอากาศซึ่งมีจำนวนทั้งหมดประมาณ 1,300 สถานีทั่วประเทศ คิดเป็นค่าเฉลี่ยที่ทุกๆระยะทาง 17 กิโลเมตรต่อหนึ่งสถานีวัดน้ำฝนภาคพื้นดิน ซึ่งเมื่อทำการประมวลผลข้อมูลเรดาร์แล้วจะได้กริดที่มีความละเอียดที่ 5 กิโลเมตร ในการศึกษาครั้งนี้คณะนักวิจัยจะประยุกต์และพัฒนาวีธีการของ (Makihara et al. 1996) ในการประมาณค่าน้ำฝนและโมเสคข้อมูลน้ำฝนซึ่งประกอบด้วยขั้นตอนหลักสองขั้นได้แก่ การประมาณค่าแฟกเตอร์สำหรับการปรับแก้ทั่วทั้งรัศมีการตรวจเรดาร์ (Estimation of the calibration factor field over the entire radar detection range)

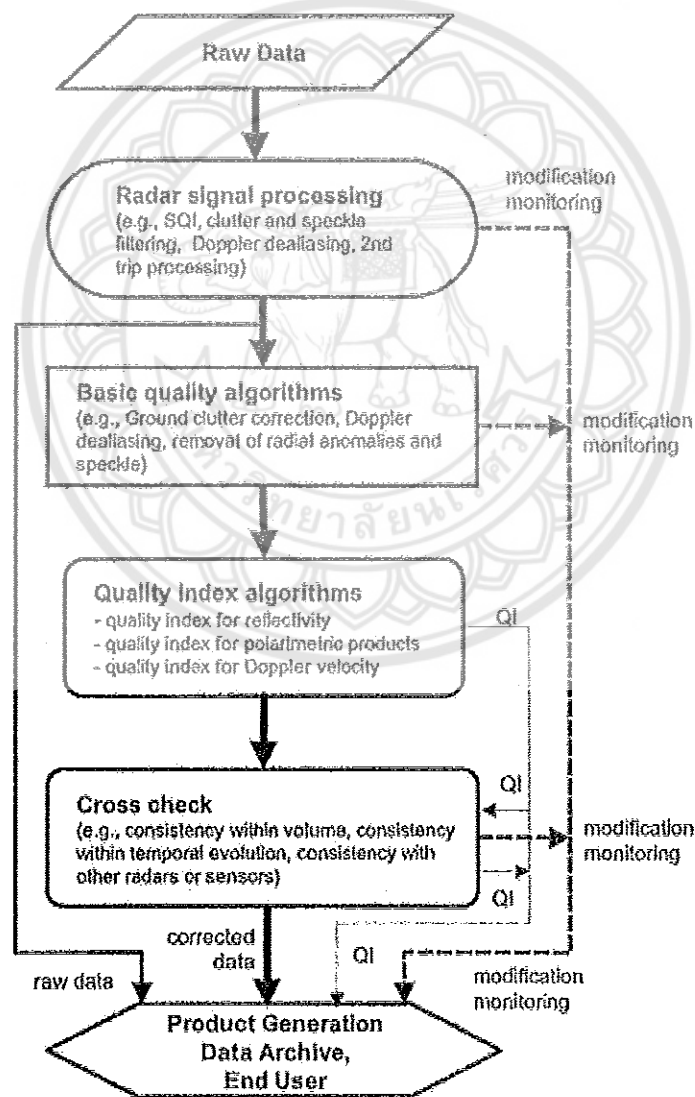
การให้ค่าน้ำหนักในการประมาณค่าความแรงของน้ำฝน (Weight for interpolation considering precipitation intensity) วิธีการหลักสองขั้นตอนเบื้องต้นจะถูกทำการทดลองโดยวิธีการทดสอบค่าความอ่อนไหวของแต่ละการทดสอบ (Sensitivity tests) เพื่อให้ได้ข้อสรุปในการนำมาทำแผนที่ประมาณค่าน้ำฝน การวัดค่าความถูกต้องเชิงสถิติที่ใช้ในการตรวจสอบค่าความถูกต้องของผลิตภัณฑ์การประมาณค่าจากน้ำฝน การประมาณค่าความถูกต้องของแผนที่เรดาร์น้ำฝนที่ได้นั้นจะประเมินโดยใช้ค่าสถิติโดยทำการเทียบสอบกับชุดของน้ำฝนสถานีที่กระจายอยู่ในขอบเขตพื้นที่ของแผนที่เรดาร์น้ำฝนซึ่งมีค่าสถิติดังต่อไปนี้ คือ ค่าสหสัมพันธ์ (R), ค่าการวิเคราะห์สมการถดถอย (R^2), ค่าความคลาดเคลื่อนรากที่สองของค่าเฉลี่ย (Root Mean Square Errors: RMSE), ค่าความคลาดเคลื่อนกำลังสองเฉลี่ย (Mean Square Error: MSE) (Veerakachen et al. 2014, Chokngamwong and Chiu 2008)

2.4 การทบทวนวรรณกรรม/สารสนเทศ (Information) ที่เกี่ยวข้อง

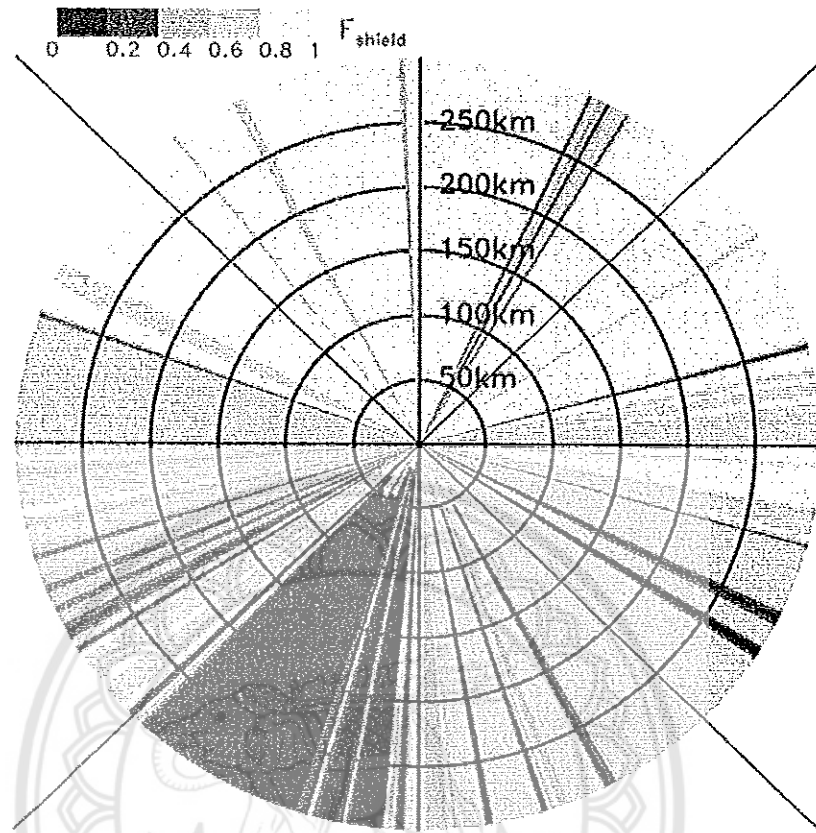
ในประเทศที่พัฒนาแล้วข้อมูลเรดาร์น้ำฝนถูกจัดสร้างเป็นฐานข้อมูลแผนที่ที่พร้อมให้นักวิจัยนำไปวิเคราะห์ในเรื่องที่เกี่ยวข้อง เช่นในประเทศสหรัฐอเมริกา อังกฤษ ฝรั่งเศส เป็นต้น ในกรณีของประเทศสหรัฐอเมริกามีการนำเรดาร์ตรวจอากาศมาใช้ในโครงการที่ชื่อ The Next Generation Weather Radar (NEXRAD) โดยใช้เรดาร์แบบ S-band ซึ่งใช้ทั้งหมดมากกว่า 160 สถานีทั่วประเทศสหรัฐอเมริกา โดยโครงการนี้เกิดขึ้นเพื่อที่จะพัฒนาระบบการให้บริการข้อมูลทางด้านสภาพอากาศของประเทศให้ดีขึ้น และยังมีส่วนช่วยในการพยากรณ์และเตือนอุทกภัยอีกด้วย (e.g. Fulton et al. 1998) โดยในโครงการ NEXRAD นี้ได้มีผลิตภัณฑ์ที่ถูกสร้างขึ้นมาอันได้แก่ ค่าการสะท้อน ความเร็วดอปเปลอร์และ ความกว้างของช่วงคลื่น ซึ่งทางทีมงานวิจัยได้สร้างอัลกอริทึมขึ้นมาเพื่อประมวลผลข้อมูลเรดาร์ในแต่ละขั้นตอนโดยเรียกระบบนี้ว่า the precipitation processing system (PPS) ซึ่งจะรวมอัลกอริทึมย่อยๆที่ได้มาจากผลงานวิจัยที่ผ่านมาในอดีต ในเรื่องของการสร้างแผนที่การสะท้อนจากข้อมูลเรดาร์ภาคพื้นดินนั้นได้มีการนำวิธีการที่เรียกว่า Hybrid scan นั่นคือ การสร้างแผนที่การสะท้อนค่ากลุ่มฝนนั้นจะพิจารณาเรดาร์บีมร่วมกับข้อมูลความสูงโดยรอบ หากเรดาร์บีมถูกกบฏเขาบดบัง (Beam Blockage: BB) ซึ่งเป็นข้อมูลที่ไต่มาจากแบบจำลองความสูงน้อยกว่า 50% และมุมยกที่ต่ำที่สุดของเรดาร์จะถูกพิจารณาก่อนเพื่อให้ได้ข้อมูลน้ำฝนที่ใกล้เคียงกับพื้นผิวโลกมากที่สุด

ในปัจจุบันมีการเสนอแนวคิดในการตรวจสอบคุณภาพของข้อมูลเรดาร์ตรวจอากาศอย่างหลากหลาย เนื่องจากมีการพัฒนาของเทคโนโลยีการตรวจวัด การนำข้อมูลดอปเปลอร์ของเรดาร์มาใช้ในการ

ตรวจสอบคุณภาพจะสามารถทำให้การจำแนกข้อมูลที่เป็นกลุ่มฝนจริงๆออกจากสิ่งที่ไม่ใช่กลุ่มฝน โดยใช้ความเร็วของการเคลื่อนที่ของวัตถุเป็นตัวกรองได้อย่างมีประสิทธิภาพ Friedrich and Hagen (2006) ได้เสนอแนวคิดเกี่ยวกับการควบคุมคุณภาพของข้อมูลเรดาร์ด้วยการใช้ค่าการสะท้อนพารามิเตอร์จาก polarization รวมถึงความเร็วของดอปเปอร์ในการกรองข้อมูลเรดาร์ในระบบ C band ซึ่งสามารถที่จะสร้างชุดผลลัพธ์ออกมาในรูปแบบจุดภาพ โดยมีรายละเอียดของผังอัลกอริทึม ดังภาพ 4



ภาพที่ 2.4 อัลกอริทึมในการควบคุมคุณภาพของข้อมูลเรดาร์ Friedrich and Hagen (2006)



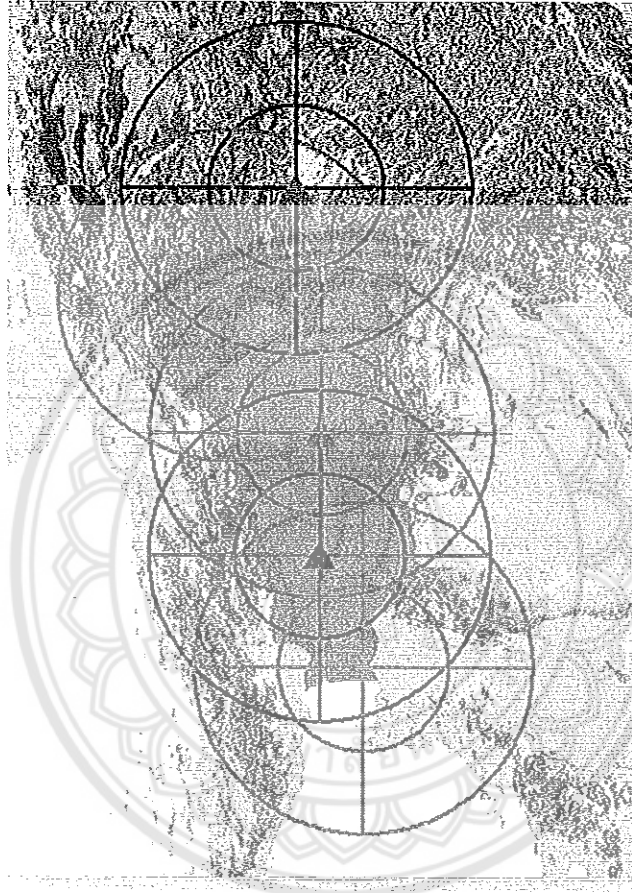
ภาพที่ 2.5 ผลลัพธ์แผนที่คุณภาพจากการใช้อัลกอริทึมควบคุมคุณภาพ Friedrich and Hagen (2006)

จากภาพที่ 2.5 พบว่าพื้นที่บริเวณด้านตะวันตกเฉียงใต้ของเรดาร์มีคุณภาพต่ำ เนื่องจากมีภูมิประเทศ
บดบังเรดาร์บีม ซึ่งทำให้ข้อมูลผลลัพธ์ชุดนี้บ่งบอกถึงคุณภาพการตรวจวัด ซึ่งจะทำให้ผู้ใช้ผลิตภัณฑ์
เรดาร์ได้ทราบถึงข้อจำกัดของข้อมูลที่ตนเองกำลังใช้อยู่

บทที่ 3 วิธีดำเนินการวิจัย

ในบทนี้จะกล่าวถึง พื้นที่ศึกษา ข้อมูลและเครื่องมือวิธีการที่ใช้ในการพัฒนากระบวนการโมเสครดาร์
ดังมีรายละเอียดดังหัวข้อต่อไปนี้

3.1 พื้นที่ศึกษา



ภาพที่ 3.1 พื้นที่ศึกษา

พื้นที่ศึกษาครอบคลุมลุ่มน้ำเจ้าพระยาดังภาพที่ 3.1 โดยมีเรดาร์ที่นำมาใช้ในการพัฒนากระบวนการ
ทั้งหมด 5 สถานี ทางตอนเหนือของลุ่มน้ำเจ้าพระยามีภูมิประเทศที่เป็นที่ราบสลับภูเขาส่วนตอนกลาง
และตอนล่างมีลักษณะเป็นที่ราบน้ำท่วมถึง

3.2 ข้อมูล

3.2.1 ข้อมูลเรดาร์

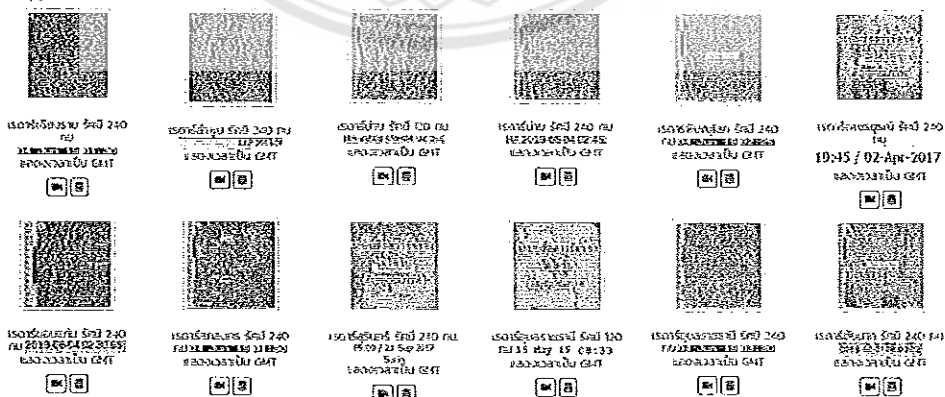
ตาราง 3.1 รายละเอียดเรดาร์ตรวจอากาศ

Radar	1 st Angle (°)	Latitude	Longitude	Height (MSL)
1. Chiangrai (CRI)	0.99°	19°57'41.919"N	99°52'53.845"E	387
2. Lamphun (LMP)	0.48°	18°33'59.652"N	99°2'20.38"E	301
3. Phitsanulok (PHK)	0.50°	16°46'31.311"N	100°13'4.756"E	45
4. Chainat (CHN)	0.50°	15°9'28.06"N	100°11'28.436"E	17
5. Suvarnabhumi (SVP)	0.70°	13°41'10.732"N	100°46'4.8"E	1

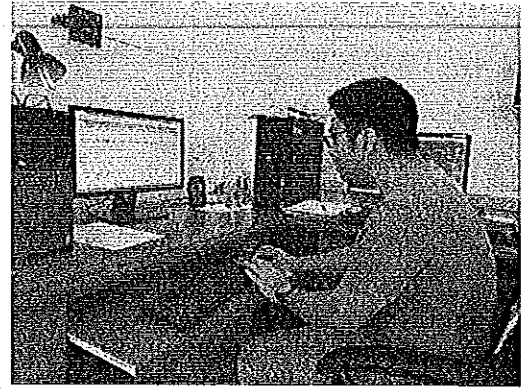
ภาพเรดาร์ตรวจอากาศของกรมอุตุนิยมวิทยาถูกนำมาใช้ในการพัฒนากระบวนการเรดาร์โมเสค เนื่องจากสามารถเข้าถึงข้อมูลได้ง่าย เรดาร์ที่ใช้ในการกวาดข้อมูลนั้นมีความถี่ในช่วยคลื่น C band ขอบเขตพื้นที่สังเกตการณ์มีรัศมี 240 กิโลเมตร ใช้ภาพมุกแรกของทุกสถานีตามรายละเอียดดังตารางด้านบน โดยมีหน่วยงานสถาบันสารสนเทศทรัพยากรน้ำองค์การมหาชน หรือ Hydro and Agro Informatics Institute (HAI) ได้ทำการเก็บข้อมูลในรูปแบบภาพนามสกุล *.gif โดยให้ดาวน์โหลดได้ที่เว็บไซต์ <http://thaiwater.net/v3/hydro>

ข้อมูลภาพเรดาร์ที่ทาง HAI ได้ให้บริการแก่ประชาชนนั้นมีความถี่ 1 ภาพต่อหนึ่งชั่วโมง ซึ่งไม่ครบตามจำนวนการกวาดภาพของกรมอุตุนิยมวิทยา ที่โดยส่วนใหญ่จะกวาดภาพที่ 4 ครั้งต่อชั่วโมง สถานภาพของประเทศไทยล่าสุดจากระบบเครือข่ายเรดาร์ ๖ คลังข้อมูลทรัพยากรน้ำประเทศไทย (๒๕7)

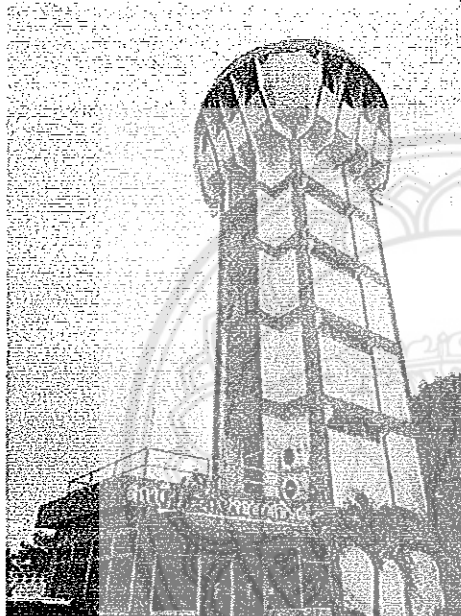
กรมอุตุนิยมวิทยา



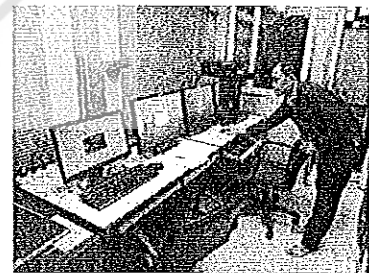
ภาพที่ 3.2 คลังข้อมูลภาพเรดาร์ตรวจวัดโดยกรมอุตุนิยมวิทยาและจัดเก็บโดยสถาบันสารสนเทศทรัพยากรน้ำองค์การมหาชน <http://thaiwater.net/v3/hydro>



ภาพที่ 3.3 สถานีเรดาร์ตรวจอากาศของกรมอุตุนิยมวิทยาจังหวัดเชียงราย



ภาพที่ 3-4 สถานีเรดาร์ตรวจอากาศจังหวัดพิษณุโลกและอุบลราชธานี



ภาพที่ 3.5 การแลกเปลี่ยนข้อมูลและการเยี่ยมชมอุปกรณ์สถานีเรดาร์ตรวจอากาศพิษณุโลก เชียงราย และสถานีเรดาร์กรมฝนหลวงอำเภอร่องขวาง จังหวัดแพร่

3.2.2 ข้อมูลแบบจำลองภูมิประเทศ

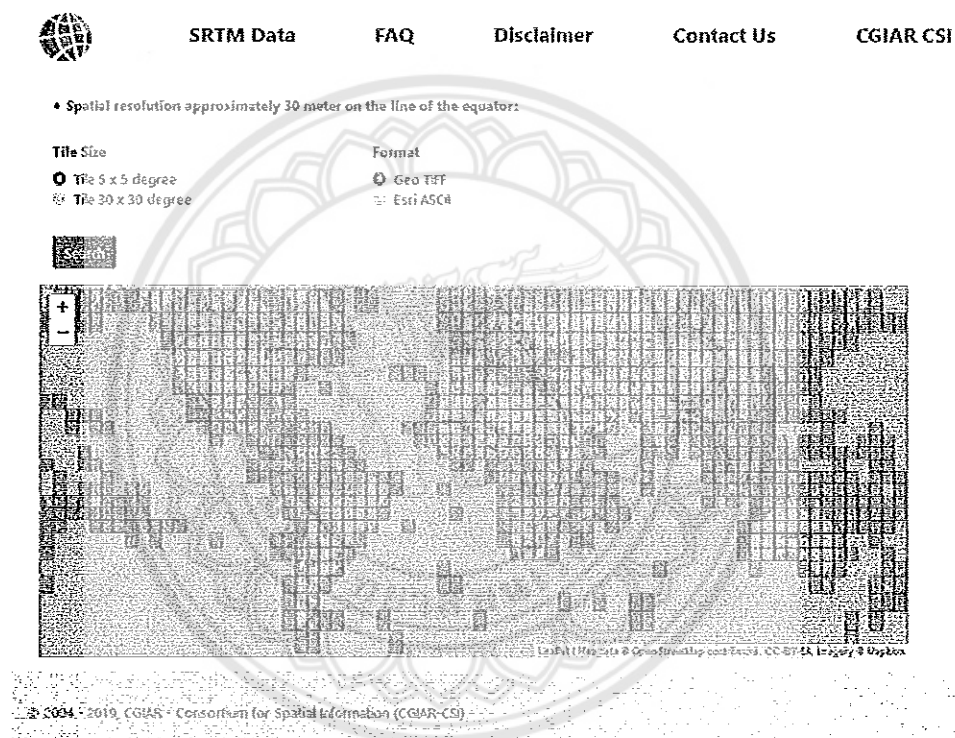
แบบจำลองภูมิประเทศ SRTM DEM เวอร์ชัน 4.1 ที่มีความละเอียดเชิงพื้นที่ระดับ 90 เมตรถูกนำมาใช้ในการวิเคราะห์การบดบังของเรดาร์บีม

ดาวน์โหลดจากเว็บไซต์

<https://cgiarcsi.community/data/srtm-90m-digital-elevation-database-v4-1/> แบบจำลอง

ภูมิประเทศจะทำการต่อและตัดให้ครอบคลุมพื้นที่สังเกตการณ์ของเรดาร์แต่ละสถานีเพื่อประหยัด

ทรัพยากรในการประมวลผล



ภาพที่ 3.6 เว็บไซต์ดาวน์โหลดข้อมูล SRTM DEM 3Arc <http://srtm.csi.cgiar.org/srtmdata/>

3.2.3 เหตุการณ์พายุ

ตาราง 3.2 รายละเอียดพายุที่ทำการวิเคราะห์

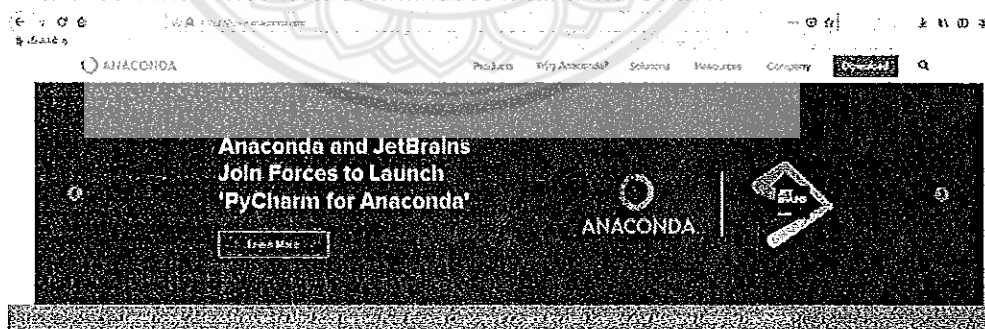
Event name	Period	No. of radar files
1. Monsoon trough (R1)	20180601-20180610	229
2. Sontihn (R2)	20180715-20180724	237
3. Bebinca (R3)	20180813-20180822	226
4. Barijat-Mangkut (R4)	20180910-20180921	252

ช่วงเวลาที่ได้ทำการวิเคราะห์ข้อมูลนั้นประกอบไปด้วย 4 ครอบคลุมฤดูฝนช่วงในปี 2561 ประกอบไปด้วยเหตุการณ์ฝนในช่วงร่องมรสุม และฝนในช่วงพายุที่มีแหล่งกำเนิดจากมหาสมุทรแปซิฟิกและทะเลจีนใต้ อีก 3 เหตุการณ์ โดยจำนวนภาพเรดาร์ที่ใช้มีจำนวนใกล้เคียงกันตามตาราง 3.2 อย่างไรก็ตาม เนื่องจากมีการพัฒนากระบวนการกรองข้อมูลของภาพเรดาร์ที่มีคุณภาพซึ่งได้อธิบายในส่วนผลของบทที่ 4 ทำให้จำนวนภาพที่ใช้จริงมีความแตกต่างกัน ช่วงเวลาที่ใช้ในการเลือกภาพนั้นมาจากการตรวจสอบการประกาศเตือนภัยของกรมอุตุนิยมวิทยา รวมถึงการพิจารณาวิเคราะห์ถึงข่าวความเสียหายที่เกี่ยวข้อง

3.3 เครื่องมือ ซอฟต์แวร์

3.3.1 ภาษาคอมพิวเตอร์ไพธอน

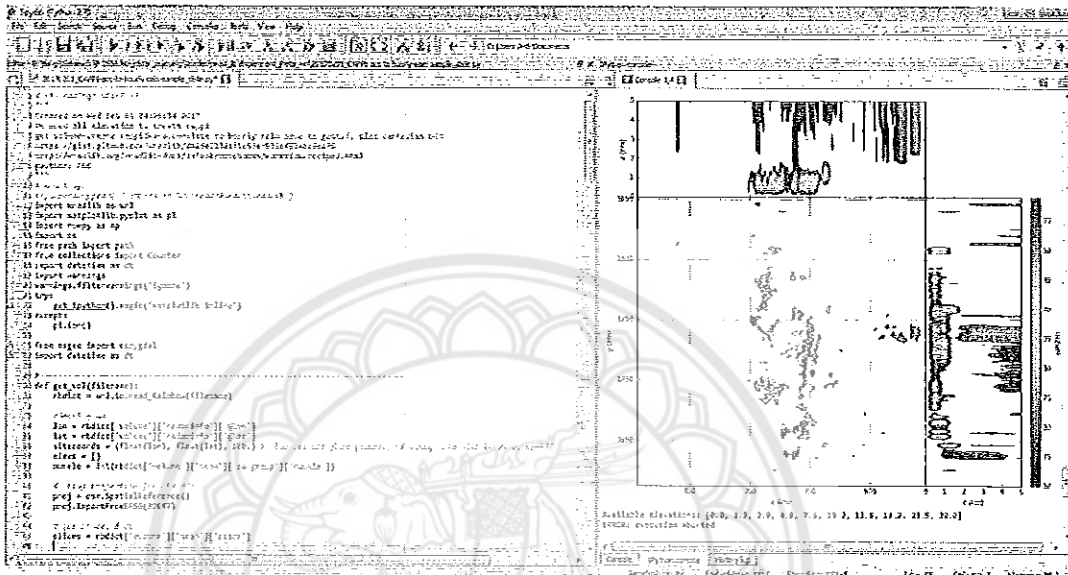
ภาษาไพธอน V2.7 ถูกนำมาใช้ในการพัฒนากระบวนการโมเสกเรดาร์ โดยใช้ซอฟต์แวร์ที่ชื่อว่า Anaconda with Python2 32bit for windows ซึ่งจะมีโปรแกรม Spyder ที่สามารถทำการเขียนโค้ดคำสั่งไพธอน โดยการเรียกใช้ไลบรารีที่เกี่ยวข้องในแต่ละกระบวนการ



ภาพที่ 3.7 โปรแกรม Anaconda <https://www.anaconda.com/>

3.3.2 ซอฟต์แวร์เรดาร์ที่สเปคเปิด wradlib

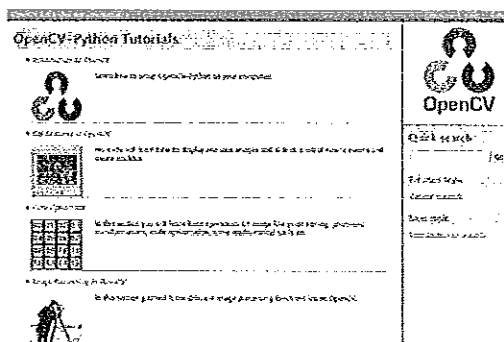
ไลบรารีเรดาร์ที่สเปคเปิดที่ถูกพัฒนาด้วยภาษาสคริปต์ไพธอนที่ชื่อ Wradlib (Heistermann et al. 2013) ถูกนำมาใช้ในการวิเคราะห์พื้นที่เรดาร์บีมที่ถูกบดบังด้วยภูมิประเทศ



ภาพที่ 3.8 โปรแกรม Spyder ที่ใช้ในการเขียนพัฒนาโปรแกรมไพธอนใน Anaconda

3.3.3 ซอฟต์แวร์ OpenCV

OpenCV ไลบรารีมีฟังก์ชันที่มุ่งเพื่อการพัฒนาในศาสตร์ของ computer vision ถูกพัฒนาขึ้นมาในภาษา C++ แต่สามารถที่จะถูกเรียกใช้ด้วยการ binding กับภาษาคอมพิวเตอร์อื่นเช่น ไพธอน จาวา หรือแมทแลป ในการศึกษาครั้งนี้ได้เรียกใช้ Opencv ในภาษาไพธอน โดยใช้ OpenCV V.2 ในการประมวลผลภาพเรดาร์



ภาพที่ 3.9 เว็บไซต์ OpenCV https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_tutorials.html

3.3.4 ซอฟต์แวร์ ARCGIS

ARCGIS ถูกนำมาใช้ในการเตรียมข้อมูลและทำแผนที่พื้นที่ศึกษา ข้อมูล SRTM DEM ที่ได้ดาวเทียมโพลดมาจะมีขนาดไฟล์และพื้นที่ขนาดใหญ่เกินกว่าขอบเขตที่ต้องการ จึงจำเป็นที่จะต้องตัดให้พอดีกับขอบเขตพื้นที่สังเกตการณ์ของเรดาร์ โดยทำการต่อข้อมูลแต่ละระวางที่ต้องการก่อน นอกจากนี้ยังนำมาใช้ในการปรับแก้ค่าพิกัดภาพเรดาร์จากพิกัดภาพเป็นพิกัดภูมิศาสตร์เพื่อระบุตำแหน่งของข้อมูลเรดาร์แต่ละสถานี

3.4 วิธีการ

ทุกกระบวนการในการศึกษาวิจัยครั้งนี้เกี่ยวข้องกับการวิเคราะห์ข้อมูลเชิงเลขทั้งหมด ดังนั้นจำเป็นที่จะต้องใช้แนวคิดของการประมวลผลข้อมูลเชิงเลขที่มีประสิทธิภาพเนื่องจากข้อมูลที่น่ามาวิเคราะห์ในครั้งนี้เป็นข้อมูลขนาดใหญ่มีมิติเชิงพื้นที่และเวลา นอกจากนี้การสร้างและพัฒนากระบวนการปรับแก้ข้อมูลรวมถึงการทำพื้นที่ฝน convective นั้นจำเป็นต้องประมวลผลบนพื้นฐานข้อมูลเชิงเลขซึ่งเป็นการลดกำลังคนในการประมวลผลและวิเคราะห์ข้อมูล อย่างไรก็ตามในแต่ละขั้นตอนของการพัฒนาต้องมีการตรวจสอบ ทวนผลลัพธ์ที่ได้ในแต่ละขั้นตอนของการพัฒนาโค้ด ก่อนนำไปใช้กับข้อมูลทั้งหมด ดังมีรายละเอียดในหัวข้อต่อไปนี้

3.4.1 การประมวลผลข้อมูลภาพเชิงเลข

หลักการประมวลผลข้อมูลภาพเชิงเลข หรือ Digital Image Processing ถูกนำมาใช้ในการประมวลผลข้อมูลภาพเรดาร์ในทุกขั้นตอนก่อนถูกนำไปใช้วิเคราะห์ โดยภาพไลบรารี OpenCV ซึ่งผู้วิจัยได้พัฒนาโค้ดเพื่อการอ่านค่าและแปลงฟอร์แมตภาพไปสู่ Jpeg ด้วยการเรียกใช้ OpenCV นอกจากนี้ยังได้ใช้ในการสกัดข้อมูลเรดาร์การสะท้อนจากภาพต้นฉบับเพื่อเตรียมเข้าสู่กระบวนการต่อไปในการปรับปรุงคุณภาพข้อมูลที่สกัดมาได้

การปรับปรุงคุณภาพของข้อมูลนั้นยังได้ใช้หลักการ Morphology ทั้งกระบวนการ erosion, dilation, opening และ closing

```
import numpy as np
import cv2
from path import path
import os
from PIL import Image
import sys
#-----
#plot opencv
def plotcv(label,image):
    #to plot image in opencv
    cv2.namedWindow(label, cv2.WINDOW_NORMAL) #-resize manually
    cv2.imshow(label,image)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

#-----
def convertGif2Jpg(inpgif_dir,radname):
    print "convert gif to JPG..."radname
    #-convert gif to jpeg
    for f in inpgif_dir.files(pattern='*.gif'):
        head, imfile = os.path.split(f)
        out_fn=imfile[0:16]+' .jpg'
        try:
            Image.open(inpgif_dir+imfile).convert('RGB').save(inpgif_dir+out_fn)
        except Exception:
            sys.exc_clear()
#    Image.open(inpgif_dir+imfile).convert('RGB').save(inpgif_dir+out_fn)
    print out_fn
```

ภาพที่ 3.10 การพัฒนาโปรแกรมไพธอนโดยเรียกใช้ OpenCV

Morphological Transformations

Goal

In this chapter,

- We will learn different morphological operations like Erosion, Dilation, Opening, Closing etc.
- We will see different functions like : `cv2.erode()`, `cv2.dilate()`, `cv2.morphologyEx()` etc.

Theory

Morphological transformations are some simple operations based on the image shape. It is normally performed on binary images. It needs two inputs, one is our original image, second one is called structuring element or kernel which decides the nature of operation. Two basic morphological operators are Erosion and Dilation. Then its variant forms like Opening, Closing, Gradient etc also comes into play. We will see them one-by-one with help of following image:



1. Erosion

The basic idea of erosion is just like soil erosion only, it erodes away the boundaries of foreground object (Always try to keep foreground in white). So what it does? The kernel slides through the image (as in 2D convolution). A pixel in the original image (either 1 or 0) will be considered 1 only if all the pixels under the kernel is 1, otherwise it is eroded (made to zero).

So what happens is that, all the pixels near boundary will be discarded depending upon the size of kernel. So the thickness or size of the foreground object decreases or simply white region decreases in the image. It is useful for removing small white noises (as we have seen in colorspace chapter), detach two connected objects etc.

Here, as an example, I would use a 5x5 kernel with full of ones. Let's see it how it works:



Quick search

Table Of Contents

- Morphological Transformations
 - Goal
 - Theory
 - 1. Erosion
 - 2. Dilation
 - 3. Opening
 - 4. Closing
 - 5. Morphological Gradient
 - 6. Top Hat
 - 7. Black Hat
 - Structuring Element
 - Additional Resources
 - Exercises
- Previous topic: [Smoothing Images](#)
- Next topic: [Image Thresholding](#)

ภาพที่ 3.11 กระบวนการ Morphology ของ OpenCV https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html#morphological-ops

นอกจากนี้ยังได้พัฒนาโค้ดในส่วนของการนับจำนวนก้อนเมฆ convective เพื่อให้เข้าใจลักษณะฝนในแต่ละช่วงเวลา รวมถึงการทำ Radar flares ที่เป็นค่าสัญญาณเรดาร์ที่ไม่มีส่วนเกี่ยวข้องกับกลุ่มฝน เพื่อให้เข้าใจคุณภาพของข้อมูลในแต่ละช่วงเวลา

3.4.2 วิธี Best Fitting Ellipses

ฟังก์ชัน `fitEllipse` ดังภาพ ใน OpenCV ถูกนำมาใช้ในการหาเมฆ convective และหา Radar flares ด้วยการสร้างอัลกอริทึมในการกำหนดเกณฑ์ตามงานวิจัยในอดีตที่เกี่ยวข้อง ฟังก์ชันนี้ถูกเรียกใช้ในภาษาไพธอนเพื่อสร้างวงรีครอบพื้นที่ที่กำหนด

fitEllipse

Fits an ellipse around a set of 2D points.

C++: `RotatedRect fitEllipse(InputArray points)`

Python: `cv2.fitEllipse(points) → retval`

C: `CvBox2D cvFitEllipse2(const CvArr* points)`

Python: `cv.fitEllipse2(points) → Box2D`

Parameters: `points` --

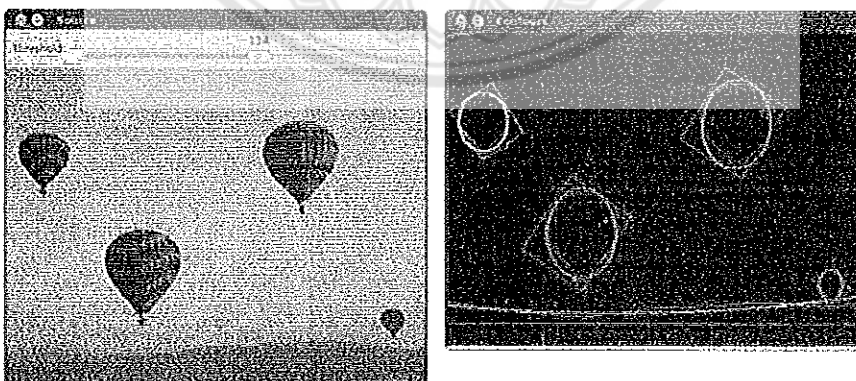
Input 2D point set, stored in:

- `std::vector<>` or `Mat` (C++ interface)
- `CvSeq*` or `CvMat*` (C interface)
- `Nx2` numpy array (Python interface)

ภาพที่ 3.12 ฟังก์ชัน `fitEllipse`

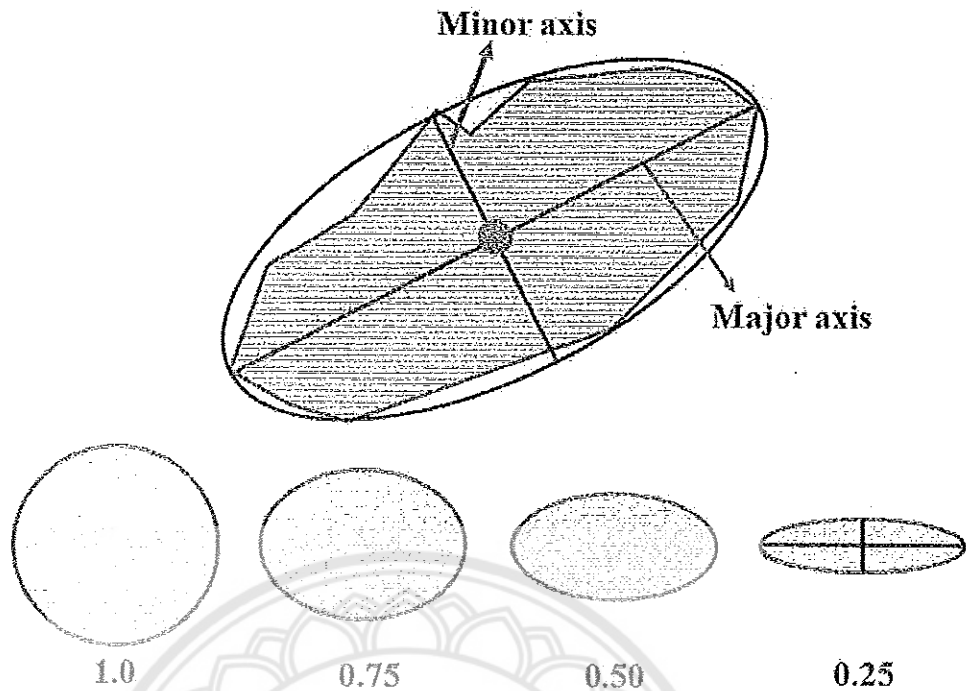
https://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html?highlight=fitellipse#fitellipse

วงรีที่ต้องพอดีมากที่สุดจะถูกสร้างขึ้นครอบวัตถุที่สนใจในภาพ



ภาพที่ 3.13 ตัวอย่างการประยุกต์ใช้ฟังก์ชัน `fitEllipse`

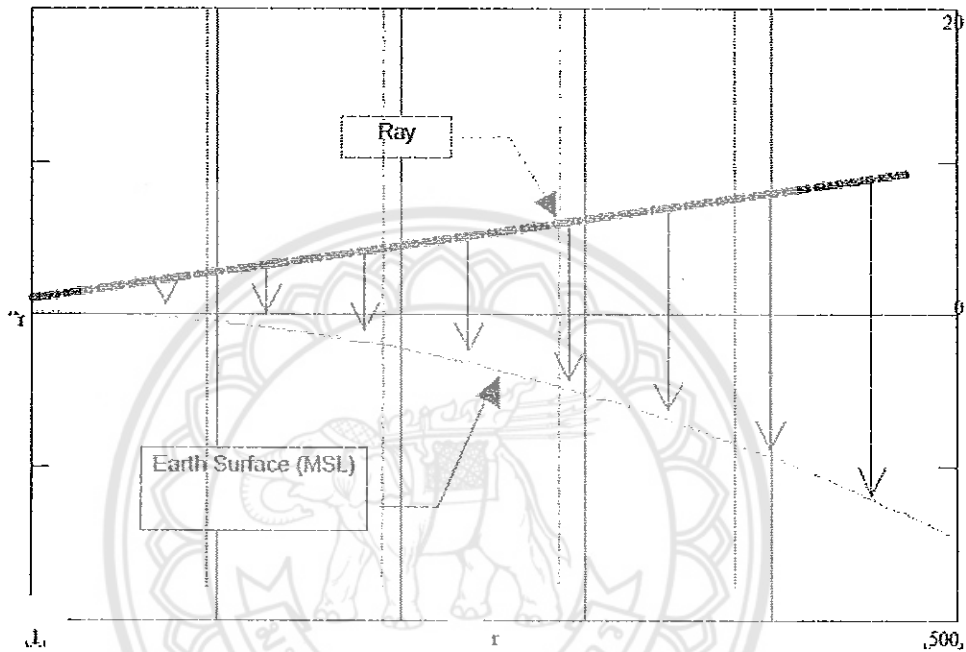
https://docs.opencv.org/2.4/doc/tutorials/imgproc/shapedescriptors/bounding_rotated_ellipses/bounding_rotated_ellipses.html



ภาพที่ 3.14 แนวคิด fitEllipse ตัวเลขสีแดงแสดงค่าสัดส่วนแกนสั้นต่อแกนยาวของวงรี ผู้วิจัยได้ดัดแปลงโค้ดด้วยการเตรียมข้อมูลให้พร้อมด้วยการจัดพื้นที่หลังของภาพเรดาร์ออกไป เหลือไว้แต่ข้อมูลค่าการสะท้อนที่ต้องการ แล้วจึงทำการปรับปรุงข้อมูลด้วยหลักการ Morphology เฉพาะค่าฝนที่รุนแรงที่แสดงเป็นสีส้มและแดงในค่าการสะท้อน จากนั้นเรียกใช้ fitEllipse เพื่อให้หาตำแหน่งของเมฆที่สนใจ โดยค่าพารามิเตอร์ที่ได้จากฟังก์ชัน fitEllipse นั้นได้แก่ ค่าความยาวของแกนวงรีทั้งด้านสั้นและด้านยาว พื้นที่วงรี การเอียงตัวของแกนยาววงรี จากนั้นทำการคำนวณค่าสัดส่วนความยาวด้านสั้นต่อด้านยาวของวงรี ถ้าด้านสั้นและด้านยาวมีขนาดใกล้เคียงกันจะมีค่าสัดส่วนที่เข้าใกล้ 1.0 ซึ่งสามารถนำไปเป็นเทียบกับเกณฑ์ที่ตั้งขึ้นเพื่อกรองเมฆ convective และ radar flare

3.4.3 การเคลื่อนที่ของเรดาร์บีมในชั้นบรรยากาศ

การจำลองการเคลื่อนที่ของเรดาร์บีมในชั้นบรรยากาศนั้นจะสามารถคำนวณถึงระยะทางและความสูงของเรดาร์บีมในแต่ละระยะกวาดตามมุมยกและมุมอิมพัลส์ที่กำหนดเข้าไปในโปรแกรมที่พัฒนาขึ้นมาได้



ภาพที่ 3.15 ลักษณะการกวาดตามมุมยก (Elevation) เป็นระยะทางใดๆ (Ray) จากแนวรัศมี โดยยิ่งระยะทางที่ไกลออกไปจากตัวสถานีเรดาร์จะทำให้ระดับความสูงจากพื้นโลกห่างไปเนื่องจากความโค้งของโลก (Edge software)

$$H = (r^2 + R'^2 + 2 r R' \sin(\phi))^{1/2} + R' + H_0$$

โดยที่ H คือ ความสูงของของ gate, r คือระยะทางของ gate, ϕ คือมุมยก, R' คือ 4/3ของรัศมีโลก, H₀ คือความสูงของสถานีเรดาร์

เมื่อนำระยะและความสูงใดๆที่คำนวณได้มาทำการเปรียบเทียบกับค่าความสูงจากแบบจำลองภูมิประเทศจะทำให้สามารถกำหนดค่าการบดบังออกมาได้ ซึ่งค่าที่ได้นี้จะบ่งบอกถึงคุณภาพของการตรวจวัดเรดาร์ในบริเวณพื้นที่นั้นๆได้เป็นอย่างดีว่าควรจะมีการปรับปรุงหรือไม่

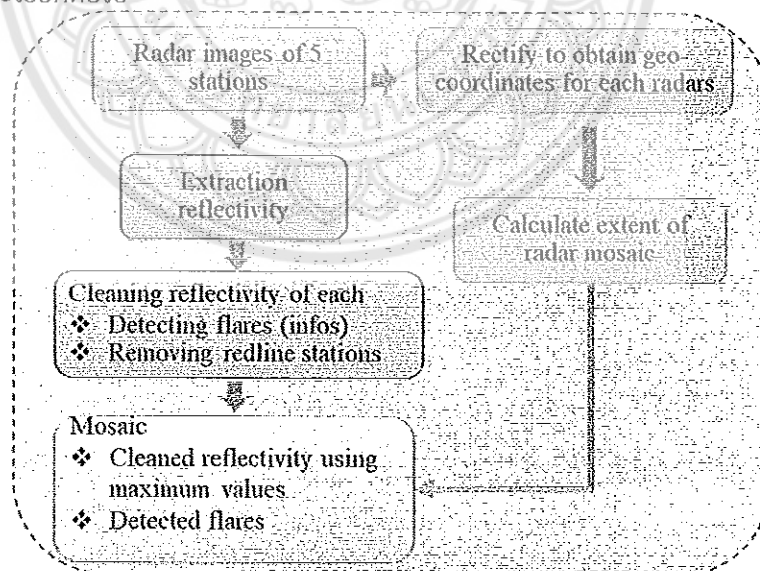
บทที่ 4 ผลการศึกษา

ในบทนี้จะกล่าวถึงผลการศึกษาที่ได้ทำการพัฒนากระบวนการโมเสคข้อมูลเรดาร์จากหน่วยงานกรมอุตุนิยมวิทยา ซึ่งได้แบ่งผลการพัฒนาเป็น 2 ส่วนหลักคือ การพัฒนากระบวนการโมเสคเรดาร์รวมถึงการวิเคราะห์ลักษณะฝนจากข้อมูลที่ได้พัฒนาขึ้นมา และการพัฒนาดัชนีชี้วัดคุณภาพการโมเสคโดยใช้ค่าการสะท้อนจากภาพเรดาร์สองสถานี ซึ่งพื้นที่ศึกษาได้ครอบคลุมพื้นที่ลุ่มน้ำเจ้าพระยา ดังมีรายละเอียดดังหัวข้อต่อไปนี้

4.1 การพัฒนากระบวนการโมเสคเรดาร์ 5 สถานี

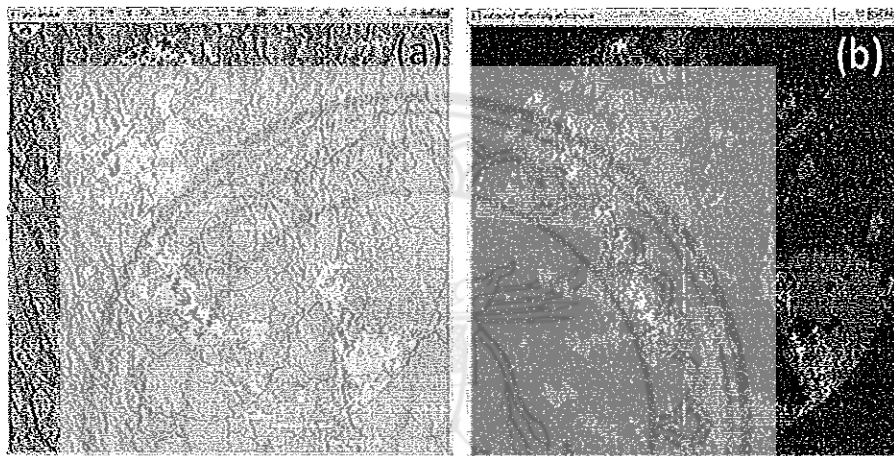
4.1.1 การพัฒนากระบวนการสกัดค่าการสะท้อนการตรวจวัดค่าฝนจากข้อมูลภาพของกรมอุตุนิยมวิทยา

การพัฒนากระบวนการโมเสคในครั้งนี้ได้ดำเนินการตามกรอบแนวคิดดังภาพที่ 4.1 ซึ่งเริ่มจากการสกัดภาพเรดาร์ทั้ง 5 สถานีเพื่อให้ได้ข้อมูลเชิงเลข จากนั้นทำการปรับแก้ข้อมูลให้เรียบร้อยด้วยการตรวจสอบข้อมูลเส้นที่ไม่ต้องการที่ปรากฏในภาพดั้งเดิม ในขณะที่เดียวกันทำการสร้างค่าพิกัดของแต่ละสถานีเพื่อนำมาคำนวณพิกัดให้กับสถานีที่ใช้ สุดท้ายนำภาพที่สกัดได้เหล่านั้นมาทำการโมเสคเข้าด้วยกันด้วยการใช้ค่าพิกัดทางภูมิศาสตร์ในการโยงยึดให้อยู่ในขอบเขตที่ได้ออกแบบไว้ในลุ่มน้ำเจ้าพระยา ซึ่งกระบวนการทั้งหมดจะมีการคำนวณซ้ำในแต่ละช่วงเวลาของข้อมูลที่ทวามา ซึ่งจะได้กล่าวในรายละเอียดต่อไป



ภาพที่ 4.1 กรอบแนวคิดกระบวนการโมเสคข้อมูลภาพเรดาร์

ข้อมูลการตรวจค่าการสะท้อนของฝนของกรมอุตุนิยมวิทยาจะแสดงออกมาเป็นช่วงค่าสีดังที่แสดงในภาพที่ 4.2(a) ซึ่งยากต่อการนำข้อมูลไปวิเคราะห์เนื่องจากเป็นแค่เพียงภาพ ดังนั้นจึงต้องมีการนำข้อมูลภาพเหล่านี้มาผ่านกระบวนการประมวลผลภาพ เพื่อสร้างข้อมูลให้อยู่ในรูปแบบที่พร้อมประมวลผลต่อไป ดังข้อมูลภาพที่ 4.2(b) ในการวิจัยครั้งนี้ได้พัฒนากระบวนการเหล่านี้ขึ้นมาจากภาษาไพธอน



ภาพที่ 4.2 การสกัดค่าเรดาร์การสะท้อนจากภาพของกรมอุตุนิยมวิทยา (a) ภาพเรดาร์ดั้งเดิมของเรดาร์สถานีพิษณุโลกทำการบันทึกภาพ เวลา 11:25 น. วันที่ 26 กรกฎาคม 2560 (b) ค่าการสะท้อนที่ถูกสกัดออกมาจากโปรแกรมที่พัฒนาบนภาษาไพธอน

การสกัดค่าการสะท้อนเพื่อให้ได้ผลลัพธ์ดังภาพที่ 4.2(b) นั้น มาจากหลักการเบื้องต้นที่ต้องการจัดภาพเบื้องหลังที่ไม่มีความเกี่ยวข้องกับค่าการสะท้อนเรดาร์ออกไป ด้วยการอ่านภาพดั้งเดิมที่มีมิติ 800×800 จุดภาพ จากนามสกุล GIF แล้วแปลงเป็นนามสกุล JPG ให้เข้ามาเก็บในรูปแบบของอาร์เรย์สองมิติด้วยการใช้ Numpy แพคเกจ เมื่อข้อมูลถูกเก็บในอาร์เรย์แล้ว ได้มีการพัฒนากระบวนการจัดพื้นหลังของภาพเรดาร์เพื่อเก็บไว้แต่ค่าการสะท้อนที่ต้องการ ในที่นี้วัตถุประสงค์ของการสร้างข้อมูลเรดาร์โมเสคเพื่อมุ่งนำข้อมูลไปใช้ในการวิเคราะห์ฝนที่รุนแรง ดังนั้นจึงมุ่งสกัดค่าสีในแต่ละช่วง ซึ่งจะแตกต่างกันกับวัตถุประสงค์ที่จะนำข้อมูลไปแปลงไปเป็นค่าฝน ทำให้ใช้ค่าข้อมูลเชิงเลขที่เก็บไว้ในอาร์เรย์เป็นข้อมูลตั้งต้น จากนั้นทำการสร้างช่วงค่าสี (Red Green Blue) ของช่วงข้อมูลค่าสีที่ต้องการเพื่อกำหนดให้ OpenCV ไลบรารีทำการสกัดค่าช่วงสีที่ต้องการเท่านั้น ซึ่งได้แสดงเป็นโค้ดดังภาพ 4.3

```

#-----
def extractRGB(img,text_rad):
    print "----->processing extractRGB--- ",text_rad
    #-nearly perfect color shades
    #-BGR
    boundaries = [
        ([0, 153, 0], [102, 255, 102]), #green
        ([0, 204, 204], [102, 255, 255]),#yellow
        ([0, 120, 230], [102, 178, 255]),#orange
        ([0, 0, 102], [51, 51, 255]),#red
    ]

    #-create combined img of reflectivity
    r,c,b=img.shape
    rf=np.zeros((r,c),dtype = "uint8")#-gray
    rfc=np.zeros((img.shape),dtype = "uint8")#-color

#-----
# to extract reflectivity by looping over the boundaries
i=0
for (lower, upper) in boundaries:
    i +=1
    # create Numpy arrays from the boundaries
    lower = np.array(lower, dtype = "uint8")
    upper = np.array(upper, dtype = "uint8")
    # find the colors within the specified boundaries and apply
    # the mask
    mask = cv2.inRange(img, lower, upper)
    output = cv2.bitwise_and(img, img, mask = mask)
    #-store detected color reflectivity to output in color
    rfc += output

    #-store detected color reflectivity to output in gray
    gray=cv2.cvtColor( output, cv2.COLOR_RGB2GRAY ) #-convert rgb to gray
    gray[gray>0]=i
    rf += gray

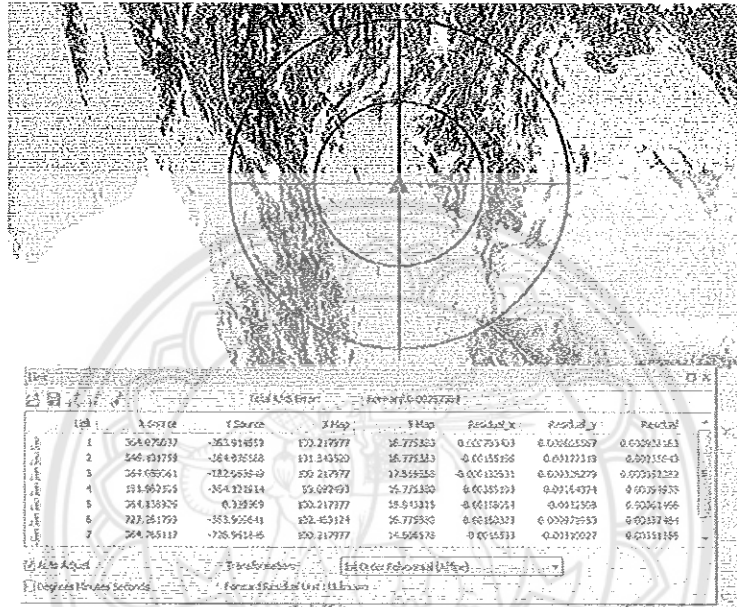
return rf,rfc
#-----

```

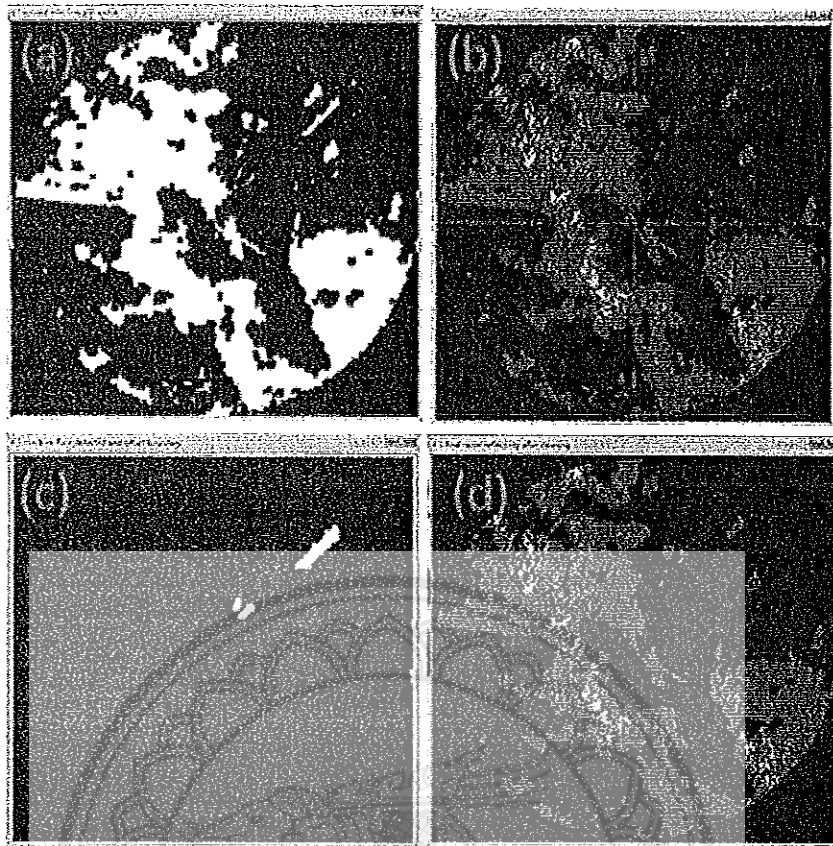
ภาพที่ 4.3 โค้ดการสกัดค่าสีจากข้อมูลเรดาร์ดั้งเดิมด้วยการใช้ไลบรารี OpenCV ในภาษาไพธอน

ขั้นตอนการกำหนดค่าสีนั้นจำเป็นต้องระบุค่าสีบนและล่างของช่วงค่าสีที่ต้องการ นั่นคือ ค่าสีแดง สีเขียวและสีน้ำเงิน ให้ใกล้เคียงภาพดั้งเดิมให้มากที่สุด เพื่อให้การสกัดค่าสีมีประสิทธิภาพได้ค่าการสะท้อนที่ครอบคลุมทุกช่วงที่ต้องการ จากโค้ดภาพที่ 4.3 จะพบว่ามีกำหนดค่าสีที่สกัดได้เหล่านั้นให้เป็นค่าเฉพาะก่อนเพื่อเก็บในอาเรย์ที่สร้างขึ้นไว้ ดังนั้นค่าพื้นหลังจะไม่ถูกสกัดเนื่องจากไม่มีการระบุขอบเขตบนและล่างให้กับการสกัดค่าของฟังก์ชัน cvtColor จึงทำให้ผลลัพธ์ที่ได้มีแต่เพียงค่าการสะท้อนเท่านั้น อย่างไรก็ตามยังมีปัญหาในการสกัดค่าที่พบอยู่ที่ทำให้ได้ค่าการสะท้อนไม่ครบถ้วน นั่นคือ เส้นขอบเขตการปกคลุม ป้ายชื่อจังหวัดและแนวเส้นระยะทางกำกับของเรดาร์ เป็นอุปสรรคที่ทำให้

ให้พื้นที่ที่เส้นเหล่านั้นทับจะไม่มีค่าเรดาร์การสะท้อน อย่างไรก็ตามพื้นที่ที่หายไปนี้ ถือว่าเป็นพื้นที่ส่วนน้อยซึ่งเมื่อนำมาพิจารณาวิเคราะห์ฝนที่รุนแรงที่มุ่งสกัดเมฆที่ก่อตัวในแนวตั้งก็ไม่ได้ส่งผลกระทบต่อวิเคราะห์เท่าใดนัก



ภาพที่ 4.4 การสร้างค่าพิกัดให้กับข้อมูลภาพเรดาร์ ยกตัวอย่างกรณีของสถานีพิษณุโลก ในขณะที่เดียวกันพิกัดภูมิศาสตร์ของภาพเรดาร์นั้นจำเป็นต้องนำมาคำนวณตำแหน่งของค่าการสะท้อนที่จะนำไปใช้ในขั้นตอนการโมเสค พิกัดของสถานีเรดาร์ทั้งห้าสถานีได้มาจากการออกภาคสนามเพื่อระบุค่าพิกัดด้วยเครื่องจีพีเอสแบบมือถือ พร้อมทั้งมีการตรวจสอบค่าตำแหน่งสถานีจากภาพถ่ายของกูเกิ้ล ซึ่งตำแหน่งสถานีที่ได้นี้ได้นำมาคำนวณพิกัดของเขตภาพเรดาร์จากการทำค่าแนวกันชนออกไปตามรัศมีในระยะ 240 กิโลเมตรซึ่งเป็นระยะทางไกลที่สุดในขอบเขตการกวาดรับสัญญาณของเรดาร์กรมอุตุนิยมวิทยาในระบบความถี่ C band จากภาพ 4 จะพบว่าได้มีการทำแนวกันชนออกไปในระยะ 120 และ 240 กิโลเมตรพร้อมกับการทำเส้นบอกระยะในแนวเหนือใต้เพื่อช่วยในการระบุพิกัด จากนั้นทำการหาค่าอ้างอิงให้กับภาพแล้วทำการแปลงค่าพิกัดจากพิกัดภาพเป็นพิกัดภูมิศาสตร์โดยใช้หลักการ Affine transformation



ภาพที่ 4.5 ผลลัพธ์การสกัดค่าเรดาร์การสะท้อน (a) ผลจากการใช้ erode morphology (b) radar flare และเส้นบอกระยะทางเรดาร์ยังคงหลงเหลืออยู่ในข้อมูลเรดาร์การสะท้อน (c) radar flares ที่ตรวจพบโดยอัลกอริทึม (d) ผลลัพธ์สุดท้ายหลังจากขจัด radar flare

ข้อมูลที่สกัดค่าการสะท้อนมานั้น ยังปะปนไปด้วยค่าการสะท้อนที่ไม่มีความเกี่ยวข้องกับฝน ดังนั้น จำเป็นที่จะต้องสร้างกระบวนการเพื่อลดทอนค่าการสะท้อนเหล่านั้นออกไปให้เหลือน้อยที่สุดเท่าที่จะทำได้ โดยในขั้นตอนนี้จะทำการขจัดค่า radar flare ซึ่งเป็นกลุ่มค่าการสะท้อนที่มีค่าคงที่และมีลักษณะยาวเรียวออกจากสถานี ซึ่งจากการพบปะแลกเปลี่ยนความรู้กับผู้เชี่ยวชาญเรดาร์ที่ปฏิบัติงาน ณ สถานีตรวจวัดเรดาร์ภาคพื้นดินของกรมอุตุนิยมวิทยา พบว่าค่าการสะท้อนเหล่านี้เกิดจากการรบกวนกันในช่วงคลื่น C band ที่ถูกนำไปใช้ในตำนานงานโทรคมนาคมด้วย นอกจากนี้เส้นกำกับระยะทางของภาพเรดาร์ยังคงปรากฏอยู่ในภาพเนื่องจากมีค่าสีแดง ในการวิจัยขั้นนี้จึงทำการประยุกต์การประมวลผลภาพเชิงเลข ด้วยการใช้ morphology operation เข้ามาช่วยในการขจัด ดังภาพที่ 4.5(a) เป็นผลลัพธ์จากการใช้ eroded operation เพื่อให้ได้พื้นผิวค่าการสะท้อนที่ต่อเนื่องของการตรวจวัด โดยจะพบว่าภาพที่ 4.5(b) นั้นจะยังมีเส้น radar flare และเส้นกำกับระยะทางปรากฏในทางทิศตะวันออกเฉียงเหนือและตอนกลางของค่าการสะท้อนที่สกัดได้

ดังนั้นจึงจำเป็นต้องพัฒนาโค้ดเพื่อหาพื้นที่เหล่านั้น พบว่าพื้นที่เหล่านั้นมีลักษณะยาวเรียว ดังนั้นผู้วิจัยจึงนำหลักการ fitted ellipse เข้ามาใช้เพื่อทำการตรวจค่าการสะท้อนที่ไม่ต้องการ และกำหนดให้การสะท้อนเหล่านั้นเป็นพื้นที่ที่ไม่ใช้ฝนด้วยการทำการ mask ออกไปจากค่าการสะท้อนดังกล่าวจะพบได้ในภาพ 5(c) ซึ่งเมื่อทำการประยุกต์ใช้กระบวนการที่สร้างมานี้กับข้อมูลค่าการสะท้อนที่สกัดได้ในครั้งแรกจะได้ผลลัพธ์ดังภาพ 5(d) ที่จะพบว่าค่ากลุ่มการสะท้อนเรดาร์ที่ไม่ใช้ฝนนั้นถูกขจัดไปเหลือไว้แต่ค่าการสะท้อนที่จะนำไปทำการโมเสกต่อไป

```

#-Step 1: apply automatically removing radar flare with criteria
#-calculate area of one pixel to find criteria of filtering only biggest possible flare
area_rad=450.0*450.0 #-unit in sq.km.
npix=eroded_images.size #-num of pix
res=area_rad/npix*1.0 #-unit of 1 pix per sq km.>>0.4365 sq.km/pixel at resolution of 800*800
#-create mask of possible flare
mask = np.zeros(eroded_images.shape, dtype=np.uint8)
#we try parameters for optimize flare detections in max_eccen:in_ellipse
max_eccenl=0.25 #-unitless 0.25 is suitable, 0.30 is fail due to inclusion of elongate rain
min_ellipse_h=20.0 #-unit of km 50.0
max_ellipse_h=250.0 #-unit of km 250.0
SumFA=0 #-to sum total area of detected flare
cnt=0 #-to count number of detected flare
for h,cnt in enumerate(contours):
    if cnt.shape[0]<5: #-prevent number of contour point less than 5
        continue
    #-apply rotated rectangle
    rect = cv2.minAreaRect(cnt) #-need to filter area of rect that more than defined
    box=cv2.boxPoints(rect)
    box = np.int0(box)
    area_rect=rect[1][0]*rect[1][1]
    #-apply fitting ellipse to contour
    ellipse = cv2.fitEllipse(cnt)
    eccentricity=ellipse[1][0]/ellipse[1][1] #-width/height
    ellipse_h=ellipse[1][1]*res #-height of ellipse in km

    #-to find radar flare best criteria
    if ellipse_h>min_ellipse_h and ellipse_h<max_ellipse_h and eccentricity<max_eccenl: #500,3000,0.1 need to be tuned
    if ellipse_h<min_ellipse_h and eccentricity>max_eccenl: #-try
    #-to avoid overlapping ellipse inside larger ellipse from "2018-0103CVextraction.py"
    M = cv2.moments(cnt) #-Moment of contour
    cx = int(M['m10']/M['m00']) #xgrid(cx), ygrid(401-cy)
    cy = int(M['m01']/M['m00'])
    if moc_rad(cy,cx)[0]!=255:
        print '----invalid ellipse'
        continue
    #-plot possible flare area, rotated rectangle, fitting ellipse
    cv2.drawContours(img,[cnt],0,(255,0,0),-1)
    #-plot rotated box over possible flare area
    cv2.drawContours(img,[box],0,(0,0,0),3)
    #-plot fitting ellipse over possible flare area
    ellipse = cv2.fitEllipse(cnt)
    cv2.ellipse(img,ellipse,(0,0,255),2)
    #-create mask
    roi = np.array(cnt, dtype=np.int32)
    #white = (255, 255, 255)
    cv2.fillPoly(mask, roi, 255) #-fill mask by desired color
    #plotcv('mask'+text_rad,mask)
    #-sum detected flare area
    SumFA += (area_rect*res)
    #-count number of detected flares
    cnt += 1

```

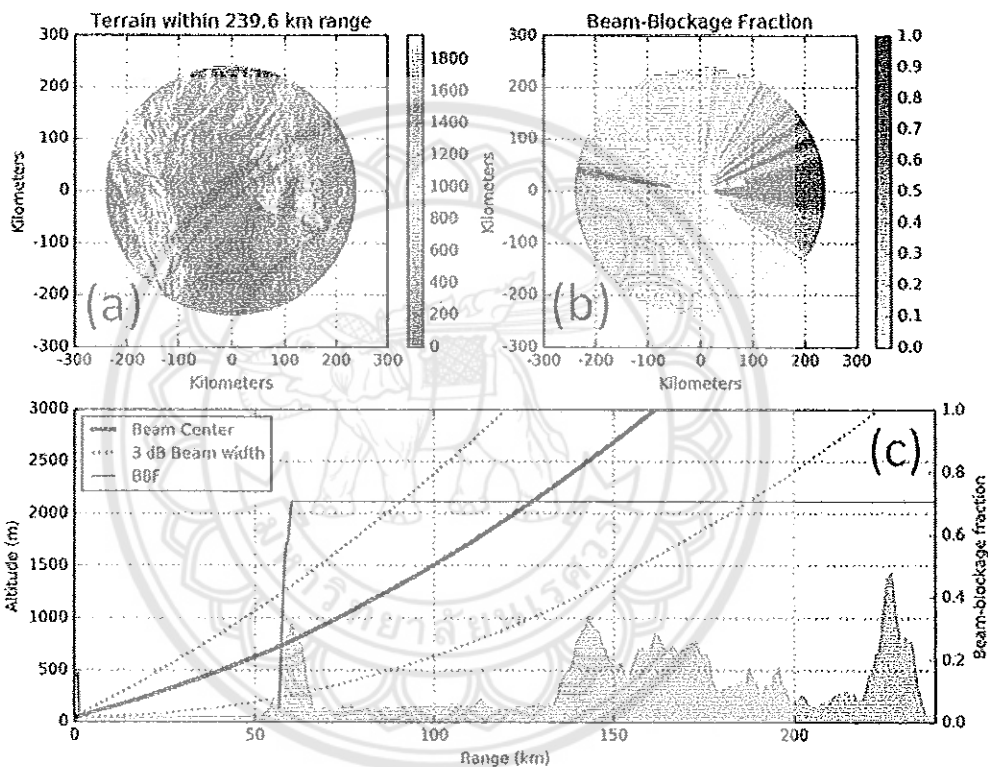
ภาพที่ 4.6 โค้ดการตรวจหา radar flare และทำการขจัดค่าการสะท้อนที่ไม่ใช่กลุ่มฝนด้วยหลักการ fitted ellipse

ภาพที่ 4.6 เป็นโค้ดที่ได้ทำการพัฒนาเพื่อตรวจหากลุ่มค่าการสะท้อน radar flare และ เส้นกำกับระยะทาง จะพบว่ามีการกำหนดค่าระยะของ flare ที่ต้องการผ่านความยาวของแกนยาววงรีที่ระยะทางระหว่าง 20-250 กิโลเมตร หมายความว่า radar flare ที่มีขนาดยาวและสั้นกว่าระยะที่กำหนดจะไม่ถูกตรวจพบ ซึ่งตัวเลขที่กำหนดนี้ได้มาจากการตรวจดูภาพเรดาร์ดั้งเดิมว่า radar flare ที่พบนั้นมีความยาวอยู่ในช่วงเท่าใด ซึ่งจากการตรวจดูภาพทั้งห้าสถานีจะพบว่า มี radar flare ทุกสถานี โดยสถานีเชียงรายและพิษณุโลกนั้นพบว่ามีปริมาณการเกิดมากกว่าสถานีอื่น นอกจากนี้จะ

นำหลักการของ fitted ellipse มาใช้ในการหา radar flare แล้ว ยังสามารถนำไปประยุกต์ใช้หากลุ่มฝนที่รุนแรงหรือ convective region ได้อีกด้วยซึ่งจะได้กล่าวต่อไป

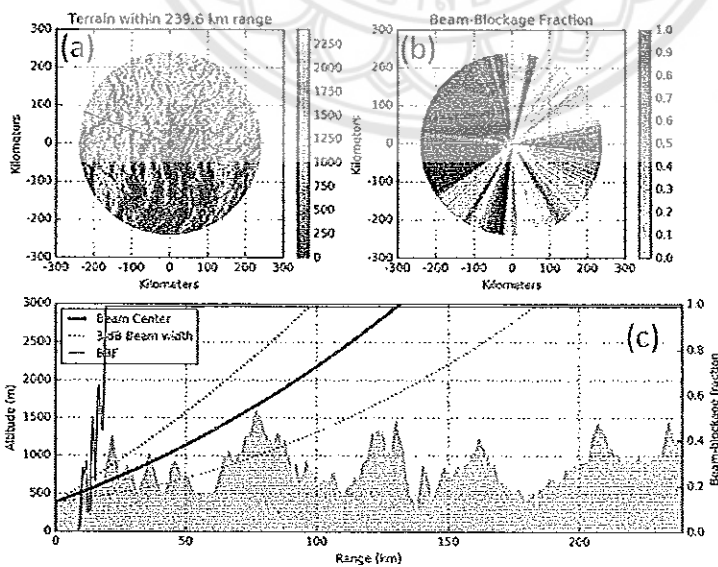
แม้ว่าจะทำการขจัดค่าที่ไม่ใช่กลุ่มฝนออกไปจากข้อมูลเรดาร์ที่สกัดได้แล้วก็ตาม ข้อมูลที่ได้มานั้น จำเป็นที่จะต้องรู้ถึงข้อจำกัดของข้อมูลเรดาร์ที่ได้ทำการกวาดตรวจวัด ปัจจัยที่เป็นตัวลดทอนความถูกต้องของการตรวจวัดเรดาร์ที่สำคัญประการหนึ่งคือ ลักษณะภูมิประเทศที่อาจบดบังการกวาดของเรดาร์บีมได้ ดังจะได้กล่าวในหัวข้อถัดไป

4.1.2 การวิเคราะห์การบดบังของเรดาร์บีมในแต่ละมุมอิมพัลส์สำหรับเรดาร์ที่ใช้ในการโมเสก 5 สถานี



ภาพที่ 4.7 อัตราส่วนการบดบังของเรดาร์บีมจากการจำลองการเคลื่อนที่ของเรดาร์บีมของเรดาร์พิชฌุโลก (a) แผนที่ดัชนีการจำลองเรดาร์บีมตามแนวมุมอิมพัลส์ แสดงโดยเส้นสีแดงเหนือภูมิประเทศรอบสถานีเรดาร์ในรัศมี 240 กิโลเมตร (b) อัตราส่วนการบดบังเรดาร์บีมตามแนวรัศมีที่มีการกวาดเรดาร์บีมในระยะทาง 240 กิโลเมตรโดยเทียบกับแบบจำลองภูมิประเทศ (c) แบบจำลองการเคลื่อนที่ในแนวศูนย์กลางของเรดาร์บีมแสดงเป็นเส้นสีน้ำเงินเข้มตามความกว้างของเรดาร์บีมที่ 3 เดซิเบลแสดงเป็นเส้นประสีน้ำเงิน จำลองเคลื่อนที่ในชั้นบรรยากาศภายใต้สมมติฐานว่าบรรยากาศไม่มีการหักกลับ โดยอัตราส่วนการบดบังในแต่ละระยะของการกวาดแสดงเป็นเส้นสีดำ

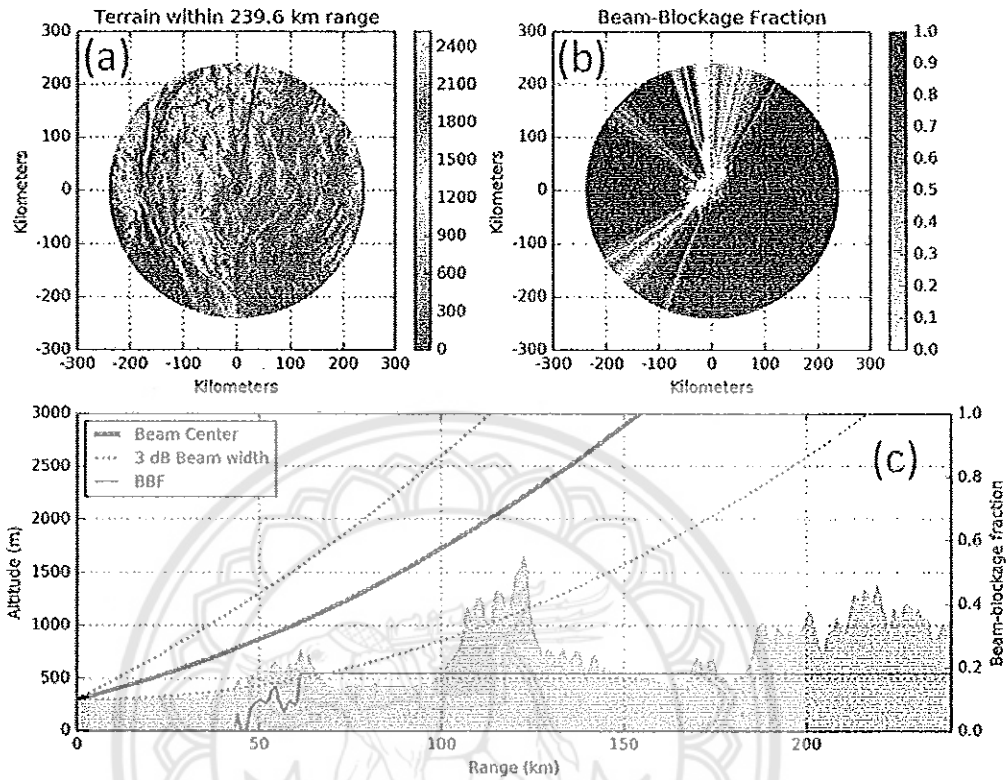
แม้ว่าข้อมูลที่ใช้ในการเป็นปัจจัยนำเข้าไปในกระบวนการโมเสคจะใช้มุมยกแรกๆของทุกสถานีเนื่องจากเป็นข้อมูลที่กรมอุตุนิยมวิทยาได้เผยแพร่แก่สาธารณะชน การทำความเข้าใจถึงข้อจำกัดเรื่องการบดบังของเรดาร์บีมจะช่วยให้การใช้ข้อมูลเป็นไปอย่างมีประสิทธิภาพ การใช้ข้อมูลจากมุมยกแรกๆในพื้นที่ภูมิประเทศแบบภูเขา เรดาร์บีมมักจะไม่สามารถหลีกเลี่ยงการบดบังของภูมิประเทศเหล่านั้นไปได้ หากมีข้อมูลการตรวจอากาศในหลายมุมยกจะสามารถหลีกเลี่ยงข้อจำกัดเหล่านี้ได้มีประสิทธิภาพยิ่งขึ้น อย่างไรก็ตามจากภาพที่ 4.7 จะพบว่าในแนวทิศตะวันตกเฉียงเหนือของเรดาร์พิชญ์โลกนั้นมี การบดบังของภูเขา นั่นก็คือเขาหลวงที่อยู่ในเขตจังหวัดสุโขทัย ทำให้ อัตราการบดบังเรดาร์บีม (Beam-Blockage Fraction:BBF) มีสัดส่วนที่สูงแสดงเป็นสีแดงเข้ม เข้าใกล้ค่า 1.0 ดังภาพที่ 4.7b ซึ่งผลลัพธ์ที่ได้ในภาพที่ 4.7 นี้ได้มาจากการคำนวณในไลบรารีเรดาร์รหัสเปิดที่ชื่อ wradlib ที่ใช้ python ในการประมวลผล ซึ่งเมื่อดูภาพ 4.7(c) จะพบว่าเมื่อเรดาร์บีมไปปะทะกับภูเขานั้นมีปริมาณ สัดส่วนน้อยกว่า 50% ของพื้นที่หน้าตัดเรดาร์บีมที่สามารถผ่านไปได้ ทำให้การตรวจวัดค่าการ สะท้อนในบริเวณนี้มีกำลังอ่อนหรืออาจไม่ได้รับการสะท้อนในพื้นที่ฝนหลังภูเขานี้ ดังจะเห็นว่า มี BBF ที่สูงถึง 0.7 หลังพื้นที่ภูเขาอันแสดงถึงพื้นที่ที่ไม่มีความน่าเชื่อถือของข้อมูล เช่นเดียวกับกับ บริเวณที่อยู่ทางทิศตะวันออกของพื้นที่ตรวจวัดของเรดาร์พิชญ์โลกก็พบว่ามีค่าการบดบังของเรดาร์ บีมที่สูงมากเนื่องเป็นพื้นที่ภูเขา ดังนั้นการนำข้อมูลเรดาร์หลายสถานีมาทำการโมเสคจะสามารถช่วย ลดข้อจำกัดการบดบังของเรดาร์บีมได้เป็นอย่างดี ดังเช่นประเทศที่พัฒนาแล้วได้ดำเนินการสร้าง ผลิตภัณฑ์โมเสคเรดาร์ขึ้นมาใช้ในหน่วยงานกรมอุตุนิยมวิทยาและหน่วยงานอุทกวิทยา การวิเคราะห์ พื้นที่บดบังของเรดาร์บีมนั้นได้ทำการวิเคราะห์ทุกสถานีที่ใช้



ภาพที่ 4.8 อัตราส่วนการบดบังของเรดาร์บีมจากการจำลองการเคลื่อนที่ของเรดาร์บีมจ. เชียงราย เช่นเดียวกับกับภาพที่ 4.7

ภาพที่ 4.8 แสดงการวิเคราะห์พื้นที่ที่บดบังของเรดาร์บีมของสถานีเรดาร์ตรวจอากาศเชียงรายในมุมมองแรก จะพบว่าภาพ 4.8 (a) แสดงภูมิประเทศแบบภูเขาล้อมรอบสถานีเรดาร์เชียงราย โดยมีพื้นที่ราบที่แสดงเป็นสีน้ำเงินอยู่ตรงกลางเป็นพื้นที่เพียงเล็กน้อยเท่านั้นใกล้ๆกับสถานีเรดาร์เชียงราย สถานีนี้ตั้งอยู่ในเขาสนามบินจังหวัดเชียงรายเพื่อทำการเฝ้าระวังด้านอากาศแก่การขึ้นลงของเครื่องบินโดยสารเป็นหลัก ดังนั้นวัตถุประสงค์นี้จึงไม่มีความจำเป็นที่จะต้องคำนึงถึงภูมิประเทศที่เป็นภูเขาเนื่องจากไม่ได้เป็นพื้นที่เฝ้าระวังในภารกิจหลักของกรมอุตุนิยมวิทยาของสถานีเรดาร์เชียงราย ดังนั้นการนำเรดาร์เชียงรายมาใช้ในการเตือนภัยสภาพฝนจึงมีข้อจำกัดอย่างมากเพราะค่าการสะท้อนจะมีค่าไม่ตรงกับความเป็นจริงของฝนที่กำลังเคลื่อนที่มา เนื่องจากสัญญาณเรดาร์นั้นไม่สามารถเคลื่อนที่ผ่านภูเขาได้ ทำให้ค่าการสะท้อนกลับมีค่าน้อยกว่าความเป็นจริง เมื่อพิจารณาถึง BBF นั้นจะพบว่าพื้นที่ส่วนใหญ่มีอัตราส่วนการบดบังที่สูงมากแสดงออกมาเป็นสีแดงดังภาพที่ 4.8(b) ซึ่งสอดคล้องกับการจำลองการเคลื่อนที่ของเรดาร์บีมในภาพที่ 4.8(c)

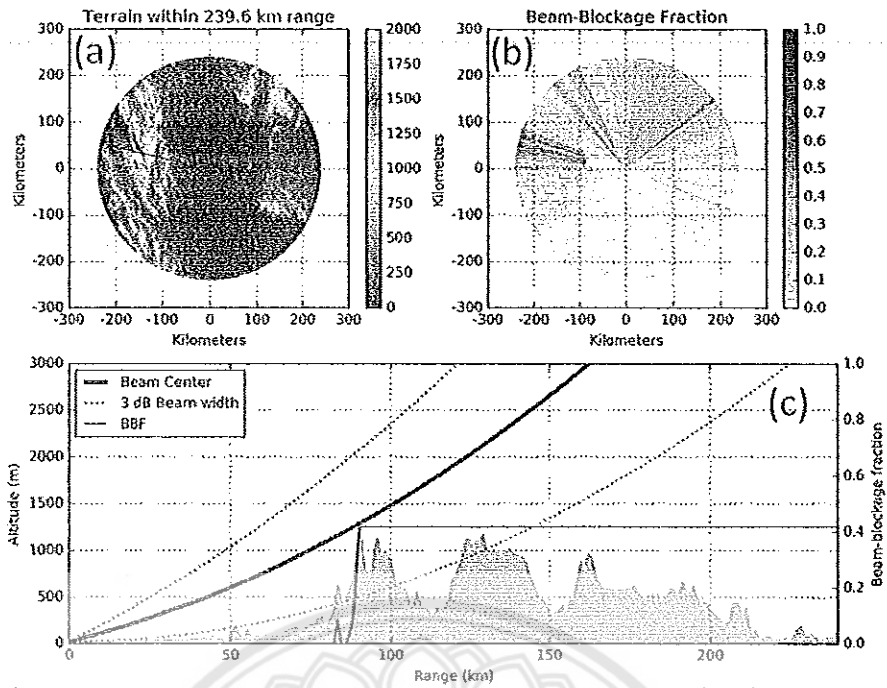




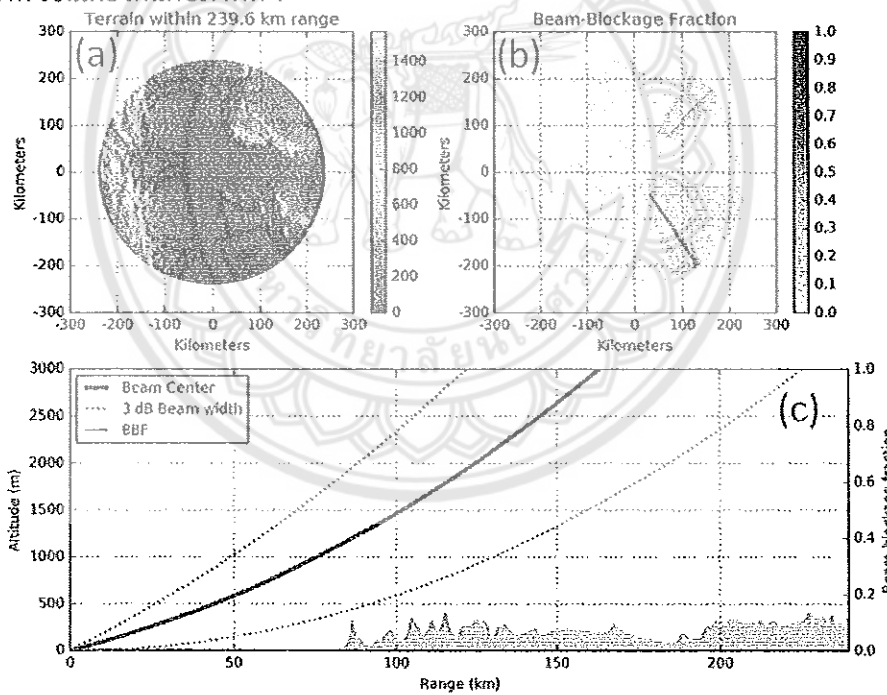
ภาพที่ 4.9 อัตราส่วนการบดบังของเรดาร์บีมจากการจำลองการเคลื่อนที่ของเรดาร์บีมของเรดาร์ จ. ลำพูน เช่นเดียวกับกับภาพที่ 7

ภาพที่ 4.9 เมื่อพิจารณาพื้นที่บดบังเรดาร์บีมของสถานีลำพูนจะพบว่าการบดบังที่รุนแรงของภูมิภาคนี้ โดยพื้นที่ใกล้เคียงกับสถานีเรดาร์เท่านั้นที่มี BBF ที่ต่ำ นั่นคือพื้นที่ในเขตแอ่งเชียงใหม่-ลำพูน ซึ่งเรดาร์สถานีลำพูนมีเป้าประสงค์เพื่อตรวจสอบสภาพอากาศในพื้นที่บริเวณเมืองทั้งสองแห่ง

ภาพที่ 4.10 แสดงการวิเคราะห์พื้นที่บดบังเรดาร์บีมของสถานีชัยนาท จะพบว่าอัตราการบดบังมีน้อยกว่าเรดาร์ที่กล่าวมาข้างต้น เนื่องจากสถานีเรดาร์ชัยนาทตั้งอยู่บนที่ราบลุ่มแม่น้ำเจ้าพระยา ทำให้สภาพภูมิประเทศรอบข้างส่วนใหญ่เป็นที่ราบ โดยจะมีเทือกเขาที่อยู่ห่างไกลออกไปซึ่งเป็นระยะที่เรดาร์บีมนั้นอยู่สูงกว่าภูมิประเทศแล้วทำให้อัตราส่วนการบดบังมีน้อย จึงสามารถนำพื้นที่การตรวจวัดมาใช้ประโยชน์ได้อย่างเต็มที่



ภาพที่ 4.10 อัตราส่วนการบดบังของเรดาร์บีมจากการจำลองการเคลื่อนที่ของเรดาร์บีมของเรดาร์ ชัยนาท เช่นเดียวกับกับภาพที่ 7



ภาพที่ 4.11 อัตราส่วนการบดบังของเรดาร์บีมจากการจำลองการเคลื่อนที่ของเรดาร์บีมของเรดาร์ สุวรรณภูมิ เช่นเดียวกับกับภาพที่ 4.7

ภาพที่ 4.11 แสดงการวิเคราะห์พื้นที่บดบังเรดาร์บีมของสถานีสุวรรณภูมิ ซึ่งตั้งอยู่ในสนามบินสุวรรณภูมิ โดยตำแหน่งที่ตั้งเป็นที่ราบสามเหลี่ยมปากแม่น้ำเจ้าพระยา ดังนั้นในระยะใกล้เคียงจึงไม่มีภูเขาสูงที่จะบดบังเรดาร์บีม ทำให้การตรวจวัดเรดาร์ในพื้นที่นี้สามารถนำข้อมูลมาใช้ได้อย่างเต็ม

ประสิทธิภาพในภาพมุกแรก

```

sitecoords = (100.768800,13.685315,1.0)#-alt from siteinfo #-setup Sunambobbox
angle = 310 # azimuth angle
nrays = 360 # number of rays
nbins = 480 # number of range bins
#Available elevations: [0.0, 1.5, 2.9, 4.9, 7.3, 10.2, 13.5, 18.2, 23.5, 30.0] for FMR
el = 0.5 # vertical antenna pointing angle (deg)
bw = 1.0 # half power beam width (deg)
range_res = 500. # range resolution (meters)
outputname=stainp.str(angle)+'deg.png'
#-Create the range, azimuth, and bearradius arrays.
r = np.arange(nbins) * range_res
bearradius = wrl.util.half_power_radius(r, bw)
#Calculate the spherical coordinates of the bin centroids and their longitude, latitude and altitude.
coord = wrl.georef.sweep_centroids(nrays, range_res, nbins, el)
lon, lat, alt = wrl.georef.polar2lonlatalt_n(
    coord[... , 0], np.degrees(coord[... , 1]),
    coord[... , 2], sitecoords)
polcoords = np.dstack((lon, lat))
print("lon,lat,alt:", lon.shape, lat.shape, alt.shape)

rlimits = (lon.min(), lat.min(), lon.max(), lat.max())
print("radar bounding box:", rlimits)

#Preprocessing the digital elevation model
rasterfile = wrl.util.get_wradlib_data_file(
'D:/Yang/Research/0.2561/MultipleRadarsPosAnalysis/IGTsworks/wrl_dea_'+stainp+''.tif')
rastercoords, rastervalues = wrl.io.read_raster_data(rasterfile)
# Clip the region inside our bounding box
ind = wrl.util.find_bbox_indices(rastercoords, rlimits)
rastercoords = rastercoords[ind[1]:ind[3], ind[0]:ind[2], ...]
rastervalues = rastervalues[ind[1]:ind[3], ind[0]:ind[2]]
# Map rastervalues to polar grid points
polarvalues = wrl.ipol.cart2irregular_spline(rastercoords, rastervalues,
    polcoords, order=3,
    prefilter=False)

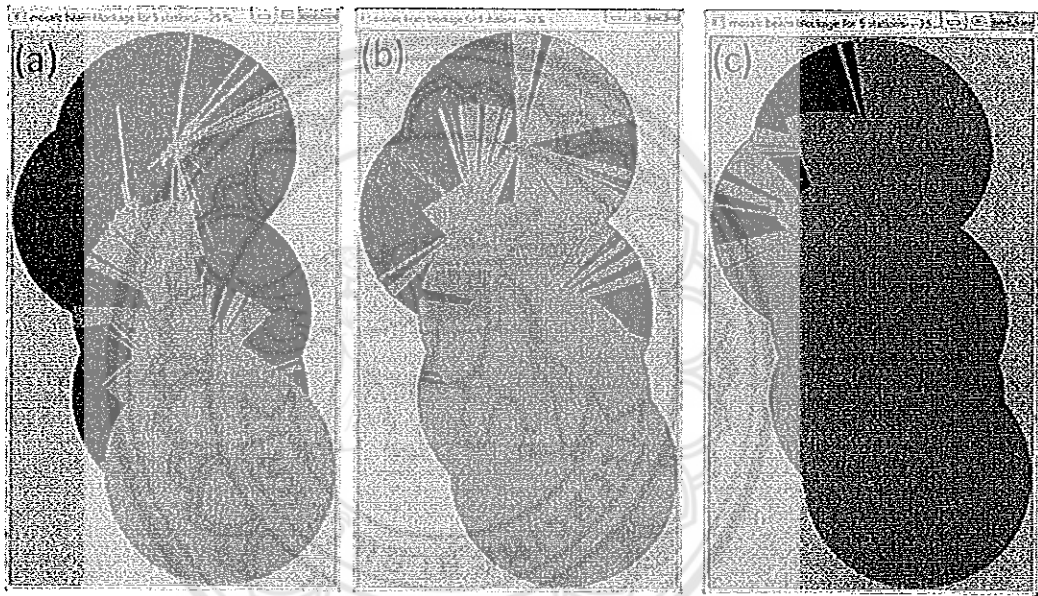
print(polarvalues.shape)
#Calculate beam blockage
PBB = wrl.qual.beam_block_frac(polarvalues, alt, bearradius)
PBB = np.ma.masked_invalid(PBB)
print(PBB.shape)
# calculate cumulative beam blockage CBB
ind = np.nanargmax(PBB, axis=1)
CBB = np.copy(PBB)
for ii, index in enumerate(ind):
    CBB[ii, 0:index] = PBB[ii, 0:index]
    CBB[ii, index:] = PBB[ii, index]
    
```

ภาพที่ 4.12 โค้ดไพธอนวิเคราะห์อัตราส่วนพื้นที่บดบังเรดาร์บีมของสถานีเรดาร์ 5 สถานีที่ดัดแปลงจากโค้ด wradlib ไลบราลี

ผู้วิจัยได้พัฒนาโค้ดไพธอนด้วยการดัดแปลงจาก wradlib ไลบราลีเพื่อนำมาใช้ในการวิเคราะห์อัตราส่วนพื้นที่บดบังเรดาร์บีม ซึ่งได้ทำการระบุค่ามุก ความกว้างของบีม ความละเอียดของระยะการตรวจวัด ค่าพิกัดสถานี โดยได้ทำการจำลองเรดาร์บีมเปรียบเทียบกับแบบจำลองภูมิประเทศ srtm การคำนวณหลักของโปรแกรมนี้จะพิจารณาในระบบพิกัด Polar coordinate ไม่ใช่ Cartesian coordinate หรือระบบพิกัดกริด ซึ่งผลลัพธ์จากโปรแกรมที่ได้นี้สามารถนำไปวิเคราะห์การบดบังของเรดาร์บีมในประเทศไทยได้ เพื่อเป็นประโยชน์ในการปฏิบัติงานของเจ้าหน้าที่เรดาร์ในด้านการจำลอง

ค่าพารามิเตอร์ก่อนการตรวจวัดจริงหรือหากมีการเคลื่อนย้ายสถานีเรดาร์ที่เป็นแบบโมบายระบบ X band ก็สามารถนำโปรแกรมที่พัฒนานี้ไปประยุกต์ใช้เพื่อวางแผนการตั้งสถานีในอนาคตได้ อันจะทำให้การตรวจวัดมีประสิทธิภาพแก่ประชาชนมากขึ้นไป หรืออาจนำแนวคิดนี้ไปคำนวณผลิตภัณฑ์โมเสครีดาร์แบบสามมิติที่ใช้หลายมุมยกเพื่อพัฒนาเป็นผลิตภัณฑ์ที่มีความถูกต้องแม่นยำและมีประสิทธิภาพกว่าข้อมูลในปัจจุบัน ดังที่มีใช้ในประเทศสหรัฐอเมริกาแล้วในปัจจุบัน อย่างไรก็ตามผู้วิจัยยังได้มีการพัฒนาการโค้ดไพธอนเพื่อคำนวณแผนที่การบดบังเรดาร์บีมในแต่ละอัตราส่วนในรูปแบบเปอร์เซ็นต์ขึ้นมา เพื่อนำผลลัพธ์มาสร้างเป็นแผนที่อันจะเป็นดัชนีบ่งบอกความสมบูรณ์ของการตรวจวัดตั้งรายละเอียดในหัวข้อถัดไป

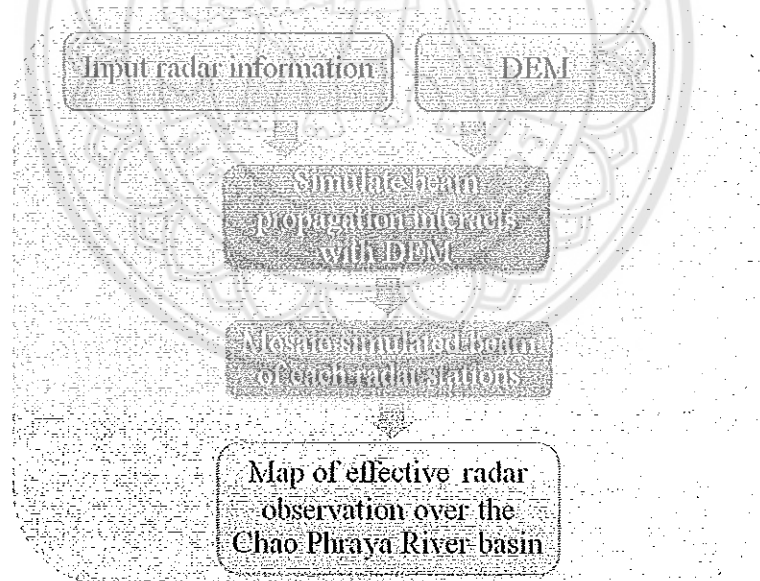
4.1.3 ผลิตภัณฑ์เรดาร์โมเสคค่าการสะท้อนเรดาร์ 5 สถานี



ภาพที่ 4.13 จำนวนแผนที่การบดบังเรดาร์บีมในแต่ละอัตราส่วนในรูปแบบเปอร์เซ็นต์ (a) 25 % (b) 50% (c) 75%

จากการจำลองเรดาร์บีมด้วยการดัดแปลงโปรแกรมจาก wradlib ไลบรารีในหัวข้อก่อนหน้านี้ทำให้ได้ทราบถึงข้อจำกัดถึงการใช้ข้อมูลมุมยกแรกของเรดาร์ตรวจอากาศ อย่างไรก็ตามหากมีการพัฒนาสร้างแผนที่โมเสคตามสัดส่วนการบดบังเรดาร์บีมของภูมิประเทศรอบข้างทั้งห้าเรดาร์ จะทำให้สามารถนำไปต่อยอดสร้างเป็นหนึ่งในปัจจัยบ่งบอกถึงคุณภาพของของการตรวจวัดได้ หรือ Radar Quality Index ดังนั้นผู้วิจัยได้พัฒนาโค้ดไพธอนในการจำลองเรดาร์บีมที่ถูกบดบังโดยในครั้งนี้จะทำการจำลองเป็นพื้นที่เรดาร์ที่หลีกเลี่ยงการบดบัง นั่นคือ ภาพ 13 (a) แสดงถึงการโมเสคพื้นที่เรดาร์

จากการกำหนดค่าการขึ้นที่เรดาร์บีมถูกบดบังอย่างน้อย 25% ของพื้นที่หน้าตัดของเรดาร์บีม แล้วนำพื้นที่เหล่านั้นมาโมเสกทั้งห้าสถานี จะพบว่าพื้นที่ส่วนใหญ่ในภาคเหนือหรือตอนเหนือของกลุ่มน้ำเจ้าพระยานั้นเรดาร์บีมถูกบดบังอย่างน้อย 25% เสียเป็นส่วนใหญ่ อย่างไรก็ตามการกำหนดค่านี้ค่อนข้างจะเป็นค่าที่เคร่งครัดเกินไป เนื่องจากเรดาร์บีมในมุกแรกมักจะมีการปนเปื้อนค่าการสะท้อนของภูมิประเทศใน Beam width อยู่แล้วเนื่องจากอยู่ใกล้พื้นโลก ทำให้พื้นที่โมเสกนั้นน้อยมาก เมื่อเทียบกับ ภาพ 13(b) จะพบว่าเมื่อกำหนดให้เป็น 50% พื้นที่การตรวจวัดจะมีมากขึ้นในภาคเหนือ ส่วนหากนำค่า 75% มาใช้ดังภาพ 13(c) จะพบว่าพื้นที่การตรวจวัดอากาศในภาคเหนือจากเรดาร์สามารถนำมาใช้ได้ นั่นก็คืออัตราส่วนการบดบังที่จะใช้ในการโมเสกเพื่อบ่งบอกคุณภาพการตรวจวัดนั้นจำเป็นที่จะต้องพิจารณาค่ามุกแรกเป็นหลัก อย่างไรก็ตามหากมุกแรกถูกตั้งค่าให้สูงมากเกินไป อาจทำให้การตรวจวัดค่าฝนไม่ตรงกับปริมาณที่ฝนตกภาคพื้นดิน ดังนั้นในงานวิจัยหลายชิ้นจึงเสนอให้มีการใช้ PsoCAPPI นั่นคือการเลือกใช้ค่ามุกหลายมุกในการสร้างแผนที่ฝนในระดับความสูงคงที่ ซึ่งในอนาคตหากกรมอุตุนิยมวิทยามีการปล่อยข้อมูลเหล่านี้ให้นักวิจัยได้ทดลองพัฒนาจะสามารถต่อยอดผลิตภัณฑ์ให้แก่ภาครัฐและประชาชนได้ใช้ประโยชน์มากขึ้น



ภาพที่ 4.14 กรอบแนวคิดการพัฒนาโค้ดเพื่อคำนวณแผนที่การบดบังเรดาร์บีม

การพัฒนาโค้ดในไพธอนเพื่อคำนวณแผนที่การบดบังเรดาร์บีมนั้น ผู้วิจัยได้พัฒนาตั้งอัลกอริทึมในภาพที่ 4.14 โดยได้ทำการรับค่าพารามิเตอร์ของเรดาร์แต่ละสถานี จากนั้นทำการรับอ่านแบบจำลองภูมิประเทศที่ได้เตรียมไว้แต่ละสถานีเพื่อทำการสร้างฟังก์ชันจำลองการเคลื่อนที่ของเรดาร์บีมในแต่ละระยะการสังเกตการณ์นั้นคือตามแนวรัศมี 240 กิโลเมตร แล้วสร้างฟังก์ชันโมเสกพื้นที่บดบังเรดาร์ที่คำนวณได้

```

def cal_beasblockage(dem, coord_dem, coord_rad, bb_thresh, h_rad, bin_rad, elev_angle, rbin, rad_ait, const_x, const_y, rad_col, rad_row, cres, rres):
    print "--calculate beas blockage --"
    # direction of beam blockage by specify percent of block
    pi=3.14159265358979
    re=6374000 # m
    rr=4.0/3.0*re # earth effective radius
    # direction of beam blockage
    pts_dem=[]
    pts_rad=[]
    for i in range(angle): # azimuth dir angle
        for j in range(rbin):
            # not to be parallel in horizontal plane
            elev_rad=elev*pi/180.0 # elevation dir radian
            beas_dist_hrz = (j*1000.0)*np.cos(elev_rad) #m

            # real beam height & real radius in meter.
            beas_height= np.sqrt(((j*1000.0)**2+rr**2)-(2*(j*1000.0)*rr*np.sin(elev_rad)))-rrth_rdrad_ait #m

            # real beam diameter straight point, may can't know exactly beam width here i used 1 degree for beamwidth
            beas_width= 2*(np.sin(bin_rad)*(j*1000.0))
            beas_upper=beas_height+(beas_width/2.0)
            beas_lower=beas_height-(beas_width/2.0)

            # convert azimuth to azimuth rad
            if i >= 0 and i <= 90:
                cart_angle=(90-i)*pi/180
            if i > 90 and i < 360:
                cart_angle=(450-i) *pi/180

            # real polar coordinate of radar to cartesian (assume 1 spacing dem=1000m.)
            azeth_rad=cart_angle*(pi/180.0)

            # locate position of beas in cartesian coordinate of radar with system
            bin_x=np.int(((beas_dist_hrz*np.cos(azeth_rad))/1000.0)*const_x)
            bin_y=np.int(((beas_dist_hrz*np.sin(azeth_rad))/1000.0)*const_y)

            # locate beam position on azeth with coordinates to know check blockage line
            bin_col=bin_x+rad_col
            bin_row=bin_y+rad_row
    
```

ภาพที่ 4.15 โค้ดการคำนวณการบดบังเรดาร์บีมในระบบพิกัดกริด (ดูโค้ดเต็มในภาคผนวก)

ภาพที่ 4.15 เป็นโค้ดที่ทำการคำนวณการบดบังของเรดาร์บีม ซึ่งนำเข้าไปในระบบพิกัดแบบ polar coordinates จากนั้นจะทำการคำนวณค่าพิกัดในระบบพิกัดกริดเพื่อให้ง่ายต่อการนำไปคำนวณพื้นที่บดบังโดยวิเคราะห์ร่วมกันกับข้อมูลแบบจำลองภูมิประเทศ ซึ่งในขั้นตอนนี้ได้พัฒนาในรูปแบบของฟังก์ชันเพื่อให้มีประสิทธิภาพต่อการคำนวณทุกพื้นที่ของเรดาร์

```

#-----
def beamBlockArea(bb_thresh,radLon,radLat,im_rad,dea_cor,radRow,radCol,radAlt,radElv,deaFile,extrad):
    print "*****calculate beam blockage area*****"
    #variables for all calculation
    # bb_thresh=50.0 #at cost beamblock percentage
    pi=3.14159265358979
    h_rd=30.0 #radar tower height fixed by THD
    bw_rad=(1.0*pi/180.0) #beam width 1 degree
    # elv=radElv # degree 1st ppl from THD
    nbin=240 #-1 km/1bin
    angle=360 #-degree
    rad_alt,elv=radAlt,radElv
    lat_top,lat_bottom,lon_right,lon_left=dea_cor
    lon_phl,lat_phl=radLon,radLat
    num_row_pic,num_col_pic=radRow,radCol # get from original image after clipping
    const_x=num_col_pic/(nbin*2.0) #used for scale of bin_x
    const_y=num_row_pic/(nbin*2.0) #used for scale of bin_y
    grd_y_res=(lat_top-lat_bottom)/num_row_pic
    grd_x_res=(lon_right-lon_left)/num_col_pic
    rad_col=np.int((lon_phl-lon_left)/(grd_x_res))
    rad_row=np.int((lat_top-lat_phl)/(grd_y_res))
    rres,cres=grd_y_res,grd_x_res #row and col resolutions calculated after verified to grid
    coord_rad=im_rad #[top,bottom,right,left]
    print grd_x_res, ', ', grd_y_res
    #---den setting and reading-----
    filename=deaFile
    gdal_data = gdal.Open(filename)
    gdal_band = gdal_data.GetRasterBand(1)
    nodataval=32767
    # convert to a numpy array
    dem = gdal_data.ReadAsArray().astype(np.float) # flipped
    # plot('dem', dem)
    #---set webmercator from geotransform
    gt=gdal_data.GetGeoTransform()
    xres=gt[1]
    yres=gt[5]
    X=np.arange(gt[0],gt[0]+dem.shape[1]*xres, xres)
    Y=np.arange(gt[3],gt[3]+dem.shape[0]*yres, yres)
    xgrid,ygrid=np.meshgrid(X,Y)
    coord_dem=dea_cor #[top,bottom,right,left]
    #-----
    #---calculate beam blockage
    #---create mesh and plot beam-block on radar coordinates
    pts_rad=cal_BeamBlockage(dem,coord_dem,coord_rad,bb_thresh,h_rd,bw_rad,elv,
        angle,nbin,rad_alt,const_x,const_y,rad_col,rad_row,
        cres,rres)

```

ภาพที่ 4.16 ได้ทำการคำนวณพื้นที่บิมที่บดบัง (ดูโค้ดเต็มในภาคผนวก)

ภาพที่ 4.16 แสดงโค้ดที่พัฒนาเพื่อให้คำนวณพื้นที่บดบังของเรดาร์บิม ซึ่งฟังก์ชันจะคืนค่าออกมาเป็นอาร์เรย์สองมิติของเรดาร์ที่กำลังทำการวิเคราะห์อยู่ เพื่อนำไปทำการโมเสกต่อในฟังก์ชันต่อไป

```

#-----
#-function to mosaic 5 beam block
def mosaic_beamBlock(mocg,moci,r,c,rr,cr,coord_mo,coord_rad,rad_bb,radname): #
    # c,c is mosaic dimension in pixel
    # rr,cr is radar dimension in pixel
    #-loop inside radar dimension to transfer distance to mosaic dimension
    htop, botcom, right, left
    cres_mo=(coord_mo[2]-coord_mo[3])/c #-in geo
    rres_mo=(coord_mo[0]-coord_mo[1])/r #-in geo

    cres_rad=(coord_rad[2]-coord_rad[3])/cr #-in geo
    rres_rad=(coord_rad[0]-coord_rad[1])/rr #-in geo

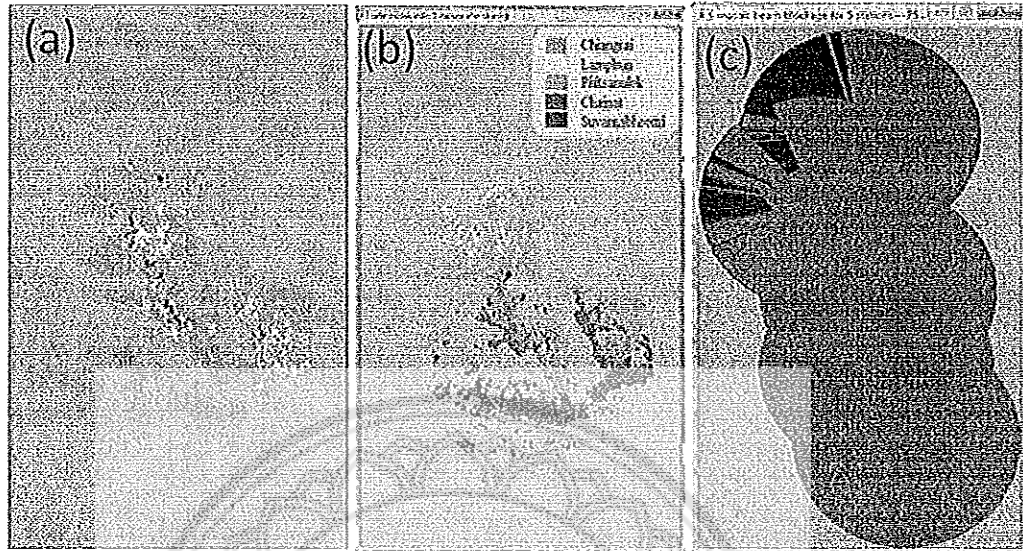
    print 'loop inside radar for mosaicking... '+'---'+radname
    print "---first loop to compute block and unblocked pixels...+++" +radname
    #-first loop to compute block and unblocked pixels
    for i in range(rr):
        for j in range(cr):
            #-transform radar pixel to mosaic pixels
            row_mo= round(((coord_mo[0]-(coord_rad[0]-(i*rres_rad)))/rres_mo)
            col_mo= round(((coord_rad[3]+(j*cres_rad))-coord_mo[3])/cres_mo)

            if col_mo>=c-1:continue
            if row_mo>=r-1:continue
            #-ocg>0,2,1 (nodata, unblock, block)
            if rad_bb[i,j]==1:
                if mocg[row_mo,col_mo]<=1:
                    mocg[row_mo,col_mo]=1
                    continue
                if mocg[row_mo,col_mo]==2:
                    mocg[row_mo,col_mo]=2
                    continue
            if rad_bb[i,j]==2:
                if mocg[row_mo,col_mo]>=0:
                    mocg[row_mo,col_mo]=2
                    continue

```

ภาพที่ 4.17 โค้ดโมเสคผลการคำนวณพื้นที่เรดาร์บีมที่ถูกภูมิประเทศบัง (ดูโค้ดเต็มในภาคผนวก)

เมื่อได้พื้นที่เรดาร์บีมที่ถูกบังจากทุกสถานีแล้ว ฟังก์ชันจากภาพที่ 4.17 จะทำการคำนวณตามขอบเขตพิกัดที่ต้องการเพื่อทำการโมเสคพื้นที่เรดาร์บีมที่ผ่านการคำนวณทั้งห้าสถานีในระบบพิกัดภูมิศาสตร์



ภาพที่ 4.18 ผลลัพธ์เรดาร์โมเสกทั้งห้าสถานี (a) ค่าการสะท้อนของเรดาร์ (b) เรดาร์โมเสก ตำแหน่งข้อมูลที่ใช้ในการโมเสก (c) พื้นที่บดบังของเรดาร์ที่กำหนดค่าเรดาร์บีมที่ถูกภูมิประเทศบังน้อยกว่า 75%

ผลลัพธ์ที่ได้พัฒนามาจากการพัฒนากระบวนการโมเสกในงานวิจัยครั้งนี้ นอกจากจะได้แผนที่เรดาร์ที่บดบังจากภูมิประเทศแล้วนั้น ยังได้ผลลัพธ์หลักคือ เรดาร์โมเสกของค่าการสะท้อนทั้งห้าสถานี (ภาพ 18 a) ที่สามารถนำไปวิเคราะห์หาลักษณะของฝนที่รุนแรงโดยมุ่งการวิเคราะห์ไปที่ฝนที่เกิดจากการก่อตัวของเมฆในแนวตั้ง นอกจากนี้ตำแหน่งจุดภาพโมเสกเรดาร์นั้นสามารถระบุถึงชื่อสถานีเรดาร์ที่ถูกใช้ในการโมเสกได้ดังภาพที่ 18 (b) ซึ่งจะมีความแตกต่างกันไปแต่ละครั้งของการโมเสก เนื่องจากการวิจัยครั้งนี้ใช้ค่าสูงสุดของค่าการสะท้อนในจุดภาพนั้นๆเป็นตัวกำหนดค่าที่จะจัดเก็บในอาร์เรย์ผลลัพธ์ ซึ่งจะมีค่าเปลี่ยนแปลงตามการตรวจวัดของแต่ละสถานี ซึ่งระยะทางใกล้ไกลจากสถานีตรวจวัดมีผลต่อพลังงานค่าการสะท้อน อีกทั้งความสูงของเรดาร์บีมยังมีผลต่อการตรวจวัดโดยตรง เมื่อนำข้อมูลเรดาร์โมเสกมาทำการซ้อนทับกับข้อมูลพื้นที่บดบังของเรดาร์ยังสามารถบอกนัยความเชื่อมั่นของผลลัพธ์เรดาร์โมเสกในพื้นที่นั้นๆได้ว่ามีความน่าเชื่อถือหรือไม่ ซึ่งถือว่าเป็นแนวทางให้ทางหน่วยงานที่เกี่ยวข้องได้สร้างเกณฑ์ในการรายงานผลความน่าเชื่อถือของข้อมูลที่เผยแพร่แก่สาธารณะชนซึ่งถือว่าเป็นเรื่องปกติของประเทศที่พัฒนาแล้วที่ควรจะให้ข้อมูลความเชื่อมั่นแก่ประชาชน

```

#convert multiple radar images
def mosaicRad(img, coord_rad, shape_rad, coord_geo, res, socg, text_rad, socg_index, num_rad):
    r,c=socg.shape
    rr,cr=img.shape
    print '---processing mosaicking mosaicRad---'+text_rad
    # r,c is mosaic dimension in pixel
    # rr,cr is radar dimension in pixel
    # loop inside radar dimension to transfer distance to mosaic dimension

    #top, border, right, left
    cres_geo=(coord_geo[2]-coord_geo[3])/c #in geo
    rres_geo=(coord_geo[0]-coord_geo[1])/r #in geo

    cres_rad=(coord_rad[2]-coord_rad[3])/cr #in geo
    rres_rad=(coord_rad[0]-coord_rad[1])/rr #in geo

    if num_rad==2:
        #to refine "soc_rad" by removing red guideline remaining from Loxphun radar
        #the kernel size will surely affect to rain pixel when compared to radar data
        # in the future we must refine this work because applying kernel will defect real size of rain
        kernel = np.ones((2,2), np.uint8)
        img = cv2.morphologyEx(img, cv2.MORPH_OPEN, kernel)

    for i in range(shape_rad[0]): # search in row
        for j in range(shape_rad[1]): # search in col
            # transfer radar pixel to mosaic pixels
            row_geo= round(((coord_geo[0]-(coord_rad[0]-(i*rres_rad)))/rres_geo)
            col_geo= round(((coord_rad[3]+(j*cres_rad))-coord_geo[3])/cres_geo)
            print "row, col geo: ", row_geo, " ", col_geo

            if col_geo>=c-1:continue
            if row_geo>=r-1:continue

            # assign only valid value to pixel of mosaiced result
            if img[i,j]>socg[row_geo,col_geo]:
                socg[row_geo,col_geo]=img[i,j] # for grey
                socg_index[row_geo,col_geo]=num_rad
            socg[row_geo,col_geo,:]=img[i,j,:] # assign extracted radar to color mosaic

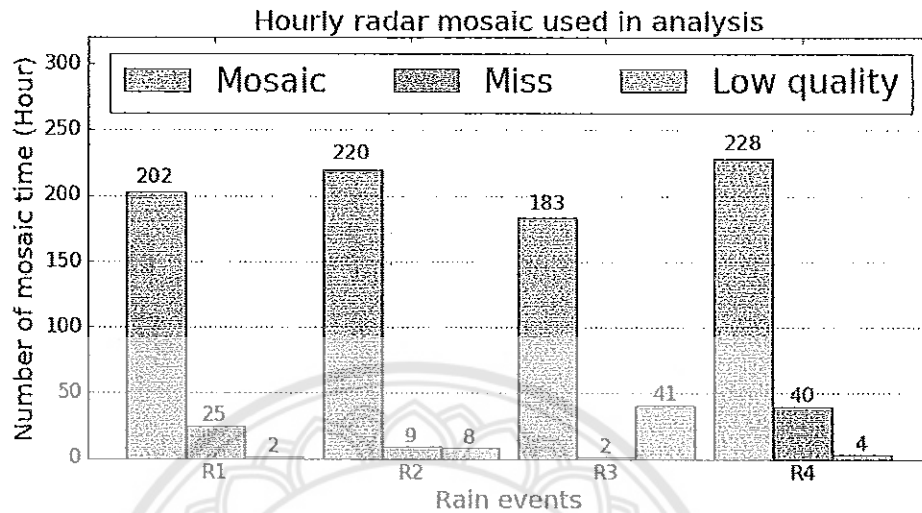
    # print -----
    return socg,socg_index

```

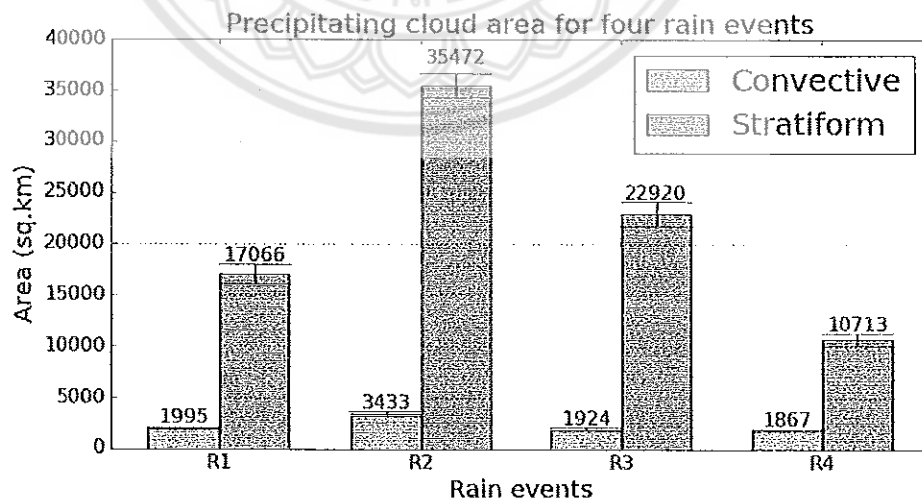
ภาพที่ 4-19 โค้ดการโมเสคเรดาร์ค่าการสะท้อน

การพัฒนากระบวนการโมเสคของค่าการสะท้อนได้ใช้แนวคิดเดียวกันกับการโมเสคพื้นที่เรดาร์บีมที่ถูกดบัง ดังภาพที่ 4-19 จะพบว่าในการศึกษาครั้งนี้ได้ใช้ค่าสูงสุดในการโมเสคผลิตภัณฑ์ทั้งสามชนิด อันได้แก่ ค่าการสะท้อน พื้นที่เรดาร์บีมที่ถูกดบัง และข้อมูลเรดาร์ที่ใช้ในแต่ละจุดภาพ

4.1.4 ผลการวิเคราะห์ลักษณะพายุที่รุนแรงจากผลิตภัณฑ์เรดาร์โมเสค

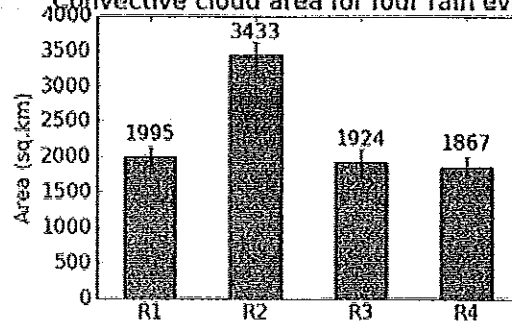


ภาพที่ 4.20 สถิติเปรียบเทียบระหว่างสี่เหตุการณ์ฝนในด้านความสมบูรณ์ของข้อมูลที่ใช้ ข้อมูลภาพเรดาร์ตรวจอากาศที่ใช้ในการสร้างแผนที่โมเสคนั้น ได้เลือกเหตุการณ์ฝนที่รุนแรงในปี 2561 มาทั้งหมด 4 เหตุการณ์นั้นคือ (R1) ร่องมรสุม (R2) การสลายตัวของพายุเชินติญ (R3) การสลายตัวของพายุเบบินคา และ (R4) พายุบาร์จัด-มังกุด จำนวนข้อมูลภาพโมเสคที่ได้ในแต่ละเหตุการณ์ฝนนั้นมีจำนวนที่ไม่เท่ากันดังภาพที่ 4.20 ซึ่งพายุบาร์จัด-มังกุดมีจำนวนภาพที่โมเสคมากกว่าเหตุการณ์อื่น รวมทั้งยังมีข้อมูลที่เสียหายที่ไม่สามารถนำมาเป็นข้อมูลนำเข้าในการโมเสคสูงถึง 40 ชั่วโมง ส่วนช่วงพายุเบบินคานั้นมีภาพที่มีคุณภาพต่ำอยู่ทั้งหมด 41 ภาพ

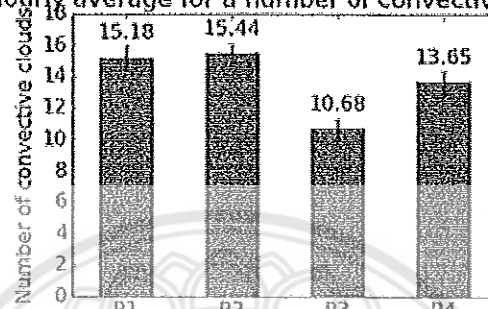


ภาพที่ 4.21 สถิติเปรียบเทียบจำนวนของเมฆฝนแบบ stratiform และ convective ทั้ง 4 เหตุการณ์

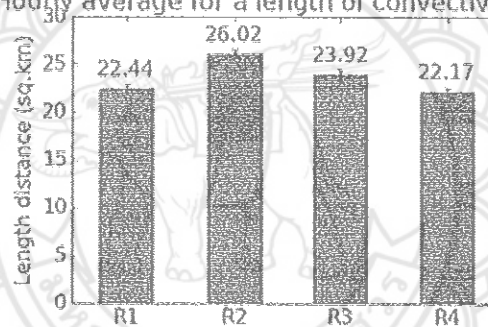
Convective cloud area for four rain events



Hourly average for a number of convective cloud



Hourly average for a length of convective cloud



ภาพที่ 4.22 ค่าเฉลี่ยรายชั่วโมงลักษณะของเมฆฝน convective ทั้ง 4 เหตุการณ์ (a) พื้นที่เมฆ (b) จำนวนเมฆ (c) ความยาวเมฆ

ภาพที่ 4.22 พบว่าจำนวนเมฆฝนแบบ convective ของเหตุการณ์พายุฝนเขตร้อนครอบคลุมพื้นที่บริเวณกว้างกว่าเหตุการณ์ฝนชนิดอื่นซึ่งพื้นที่ฝนแบบ stratiform ก็มีจำนวนที่สอดคล้องไปด้วย ทำให้ฝนในเหตุการณ์พายุฝนเขตร้อนส่งผลกระทบต่อพื้นที่การสลายตัวของพายุ ซึ่งหากหน่วยงานที่เกี่ยวข้องนำกระบวนการและวิธีการที่ได้พัฒนาในงานวิจัยครั้งนี้ไปสู่ภาคปฏิบัติแบบเรียลไทม์ จะทำให้งานเฝ้าระวังฝนที่รุนแรงสามารถเตือนภัยและให้ข้อมูลที่เป็ประโยชน์แก่ประชาชนได้ดียิ่งขึ้น ส่วนกายภาพของเมฆฝน convective นั้นได้ทำการสกัดค่าออกมาเพื่อวิเคราะห์ถึงพื้นที่ จำนวนและความยาวของเมฆดังภาพ 4.22 พบว่าพายุเขตร้อนนั้นสร้างเมฆฝน convective ที่มีค่ากายภาพสูงกว่าพายุอื่นอย่างชัดเจน


```

0-----
# find contour of convective storm
# conv program must calculate length and prop respect to latitude
res=0.73 deg for length one side of pixel measured in degree
sqrs=0.73*0.73 for the pixel side face 0.53425 degree

# create mask of possible flare
mask = np.zeros(close_in.shape, dtype=np.uint8)
# apply findContours
joker, contours, hierarchy = cv2.findContours(close_in, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

# setting parameters to find valid flare need to be located to find appropriate values
min_area_rect=10 # unit of sq km
max_area_rect=5000 # unit of number pixels
max_eccen=0.1 # constant
min_ellipse_h=5.0 # unit of km
max_ellipse_h=1000.0 # unit of km

cv_poly = np.zeros(close_in.shape, dtype=np.uint8) # to store roi of cv, will use in statistic analysis
ct=0 # to count number of detected flare
for h, cnt in enumerate(contours):
    # print 'number of contour points:', cnt.shape[0]
    if cnt.shape[0]<5: # prevent number of contour point less than 5
        continue
    print 'number of contour points:', cnt.shape[0]

    # apply closed rectangle
    rect = cv2.minAreaRect(cnt) # need to filter area of rect that over 1000 pixels
    box=cv2.boxPoints(rect)
    box = np.int0(box)
    area_rect=rect[1][0]*rect[1][1]*sqrs # unit of sq km

    # apply fitting ellipse to contour
    ellipse = cv2.fitEllipse(cnt)
    eccentricity=ellipse[1][0]/ellipse[1][1] # width/height
    ellipse_h=ellipse[1][1]*res # height of ellipse in km
    
```

ภาพที่ 4.23 ได้การค้นหาเมฆฝนแบบ convective

โค้ดไพธอนถูกพัฒนาขึ้นเพื่อประมวลผลข้อมูลภาพที่ได้โมเสคเพื่อทำการค้นหาบริเวณที่เป็นเมฆ convective ดังภาพที่ 4.23 โดยได้กำหนดค่าพารามิเตอร์ที่จำเป็นได้แก่ ขนาดพื้นที่ ความยาวของ เมฆ ซึ่งคำนวณค่าเหล่านี้ได้จากฟังก์ชัน cv2.fitEllipse ในไลบรารี OpenCV ซึ่งต้องการพื้นที่รูปปิดที่มีจำนวนหมุดของรูปปิดไม่น้อยกว่า 5 จุด จากนั้นจะทำการสร้างขอบเขตเมฆฝนตามเงื่อนไขดังภาพที่ 4.23

```

# apply criteria to find convective cloud
# create mask of invalidly detected cloud
mask = np.zeros(close_in.shape, dtype=np.uint8)
if ellipse_h>min_ellipse_h and ellipse_h<max_ellipse_h and eccentricity<max_eccen and area_rect<min_area_rect:
    # to avoid error due to ellipse center location ellipse
    # cv2.moments(cnt) # moment of contour
    cx = int((M['m10']/M['m00'])) # center (cx, cy) of (x0-y)
    cy = int((M['m01']/M['m00']))
    print 'cx, cy:', cx, cy, 'area_rect:', area_rect
    if ecc_rad[cy,cx][0]==255:
        print '----invalid ellipse'
        continue

    # plot possible cloud prop, rotated rectangle, fitting ellipse
    cv2.drawContours(ecc_rad, [cnt], 0, (255,0,0), -1) # by fitting center

# print 'cx, cy:', cx, cy, 'area_rect:', area_rect, 'ecc_rad:', ecc_rad

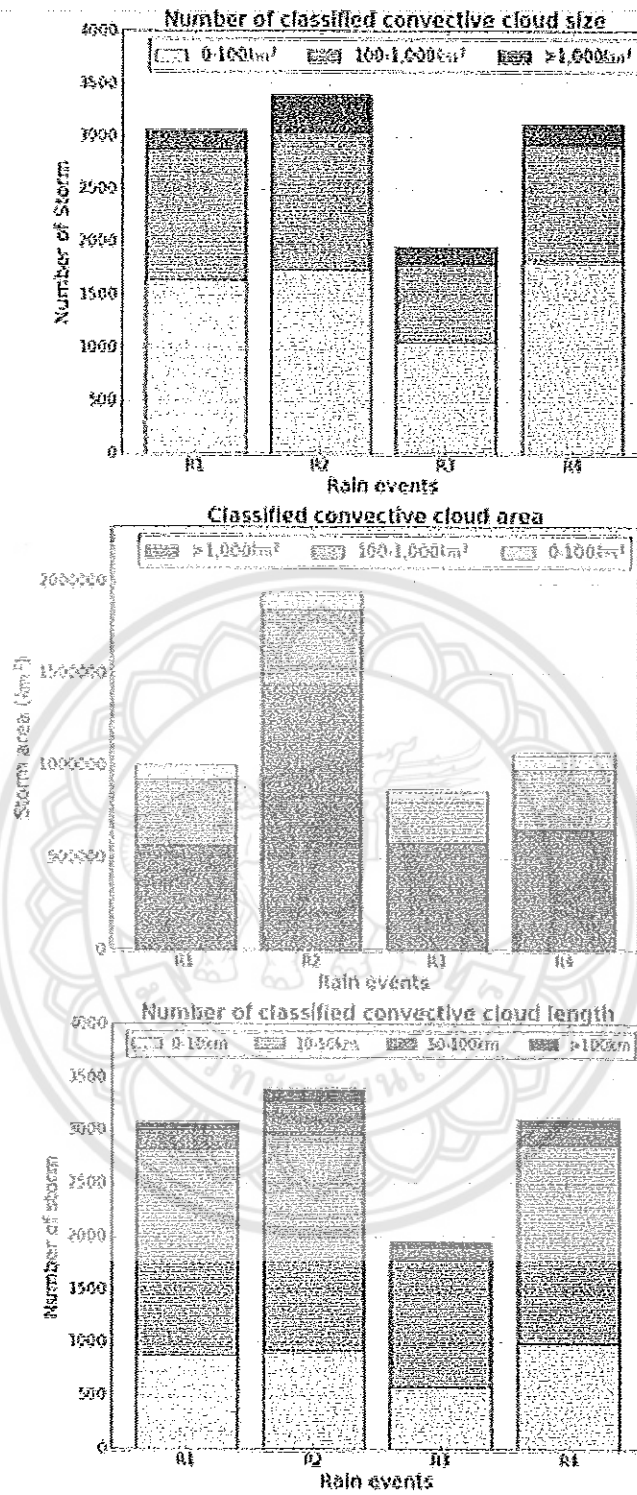
# to find center of convective cloud
h = cv2.eccen(cnt) # eccentricity of contour
cx = int((M['m10']/M['m00'])) # center (cx, cy) of (x0-y)
cy = int((M['m01']/M['m00']))
# distance center to detected flare
clen=coord_eu[3]+(cx*res_sq)
clat=coord_eu[0]-(cy*res_sq)
print 'clen, clat:', clen, clat, 'area_rect:', area_rect, 'ecc_rad:', ecc_rad

# print fitting ellipse over possible cloud shape
ellipse = cv2.fitEllipse(cnt)
cv2.ellipse(ecc_rad, ellipse, (150,20,255), 2) # flare cloud

# count number of detected convective cloud
ct +=1
print 'original HPI radar on detected possible flare showed by box and ellipse, ecc_rad:', ecc_rad, 'ecc_rad:') # plot to check detected cs

# create mask to extract convective blob
roi = np.array([cnt], dtype=np.int32)
cv2.fillPoly(mask, roi, 255) # fill each invalidly detected cv pixel with value 255
cv2.fillPoly(cv_poly, roi, 255) # fill cv flare pixel
    
```

ภาพที่ 4.24 ได้การค้นหาเมฆฝนแบบ convective (ต่อ)



ภาพที่ 4.25 สถิติจำแนกเปรียบเทียบ จำนวน ขนาด และความยาวของเมฆฝนแบบ convective

ภาพที่ 4.25 แสดงผลการวิเคราะห์สถิติของเมฆแบบ convective ในช่วง 4 เหตุการณ์ เมื่อวิเคราะห์ถึงจำนวนพื้นที่ที่ได้ถูกจำแนกเป็นระดับเพื่อให้เข้าใจถึงความรุนแรงของแต่ละเหตุการณ์จะพบว่า เหตุการณ์พายุเชินติญนั้นมีสัดส่วนจำนวนพื้นที่เมฆฝนแบบ convective ที่มากกว่า 1,000 ตร.กม. มากกว่าเหตุการณ์อื่น ครอบคลุมบริเวณกว้าง นอกจากนั้นจำนวนพายุที่มีความยาว 50 กม. มีสัดส่วนมากกว่าเหตุการณ์ฝนอื่นอย่างเห็นได้ชัดเจน



สำนักหอสมุด
06 ส.ค. 2561

1034753

4.2 การพัฒนาดัชนีชี้วัดคุณภาพการโมเสกโดยใช้ค่าการสะท้อนจากภาพเรดาร์สองสถานี แม้ว่า การพัฒนากระบวนการโมเสกเรดาร์นั้นมีความจำเป็น เนื่องจากสามารถนำไปพัฒนาต่อในการ เฝ้าระวังภัยสภาพอากาศแบบเรียลไทม์ได้ อย่างไรก็ตาม การพัฒนาดัชนีชี้วัดคุณภาพการโมเสกนั้นก็มีความสำคัญไม่น้อยไปกว่ากัน เนื่องจากข้อมูลที่ได้จากการสังเกตการณ์ด้วยเรดาร์อุตุนิยมวิทยา ภาคพื้นดินนั้นมีความคลาดเคลื่อนหลายประการโดยธรรมชาติ ทำให้เกิดข้อจำกัดต่อการนำไปใช้งาน แก่ผู้ใช้งานที่ไม่มีความรู้พื้นฐานด้านอุตุนิยมวิทยาและเรดาร์ ดังนั้นในหัวข้อต่อไปนี้จะวิจัยได้ทำการ พัฒนาสมการเชิงเส้นอย่างง่ายในการชี้วัดคุณภาพของผลิตภัณฑ์เรดาร์โมเสก เพื่อให้เป็นแนวทางแก่ กรมอุตุนิยมวิทยาที่จะนำไปพัฒนาต่อยอดในการสร้างผลิตภัณฑ์เพื่อบ่งชี้คุณภาพของข้อมูล

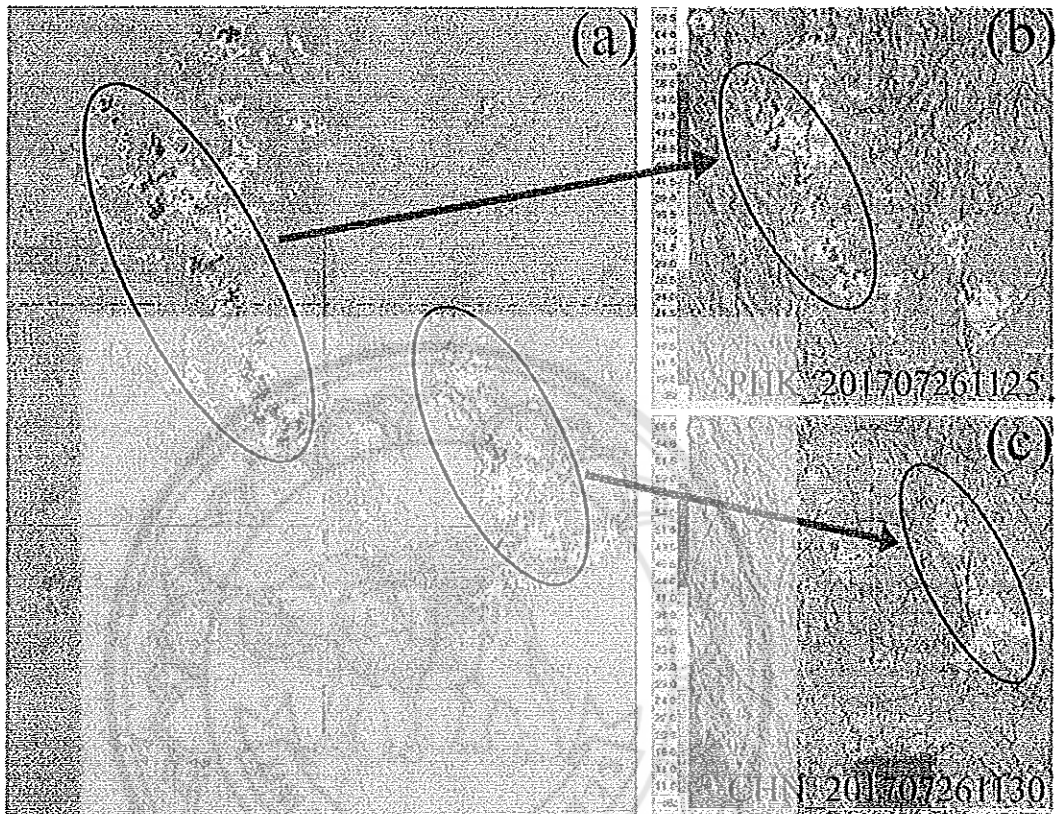
4.2.1 การพัฒนาสมการเพื่อชี้วัดคุณภาพของเรดาร์โมเสก

สมการที่จะชี้วัดคุณภาพของผลิตภัณฑ์เรดาร์โมเสกนี้ ได้นำตัวแปรมาจากการทบทวนวรรณกรรมซึ่ง จะมีปัจจัยหลักสามปัจจัยในงานวิจัยครั้งนี้ ได้แก่ ปัจจัยที่1ระยะทางของเรดาร์บีมจากสถานีไปยัง ตำแหน่งใด ๆ ในพื้นที่สังเกตการณ์ ปัจจัยที่2 ความสูงของเรดาร์บีมจากพื้นโลกซึ่งจะแปรเปลี่ยนไปตาม ระยะทางที่เพิ่มขึ้น และปัจจัยที่3การบดบังเรดาร์บีมจากภูมิประเทศที่ได้จากแบบจำลองภูมิประเทศ เรียกสมการนี้ว่า Radar Quality Index

$$RQI = (0.25 * \text{Distance}) + (0.25 * \text{Elevation}) + (0.5 * \text{Beam blockage})$$

สมการนี้จะมีการให้ค่าน้ำหนักของปัจจัยที่แตกต่างกัน นั่นคือการบดบังเรดาร์บีมจากภูมิประเทศถือว่ามีอิทธิพลมากที่สุด รองลงมาคือระยะทางและความสูงที่มีอิทธิพลเท่ากัน สมการชี้วัดคุณภาพของ เรดาร์โมเสกนี้จะถูกใช้กับทุกจุดภาพในพื้นโมเสกที่พิจารณา สมการที่ได้นี้จะให้ค่า RQI ที่ผันแปรไป แต่ช่วงเวลาของการสังเกตการณ์ด้วยเรดาร์ เนื่องจากในการวิจัยครั้งนี้ได้ใช้ค่าสูงสุดของการสะท้อน ในการยึดเป็นค่าที่จะนำไปโมเสก ซึ่งค่าสูงสุดที่ได้นี้จะผันแปรไปตามลักษณะกลุ่มฝนที่ทำการตรวจวัด ในการพัฒนาและทดสอบอัลกอริทึมครั้งนี้ได้ใช้เรดาร์สองสถานีที่ใกล้กันอันมีตำแหน่งที่ตั้งอยู่ในลุ่มน้ำ เจ้าพระยา นั่นคือสถานีเรดาร์ตรวจอากาศพิษณุโลกและชัยนาท ข้อมูลที่ใช้เป็นข้อมูลจากภาพเรดาร์ที่ จะต้องนำไปสกัดค่าการสะท้อนดังหัวข้อ 4.1.1 ด้วยการถ่ายภาพมุกแรกของทั้งสองสถานีเนื่องจาก เป็นภาพที่เข้าถึงได้อย่างเสรี เผยแพร่โดยกรมอุตุนิยมวิทยา

4.2.2 เรดาร์โมเสคสถานีพิฆณโลกและชัยนาท



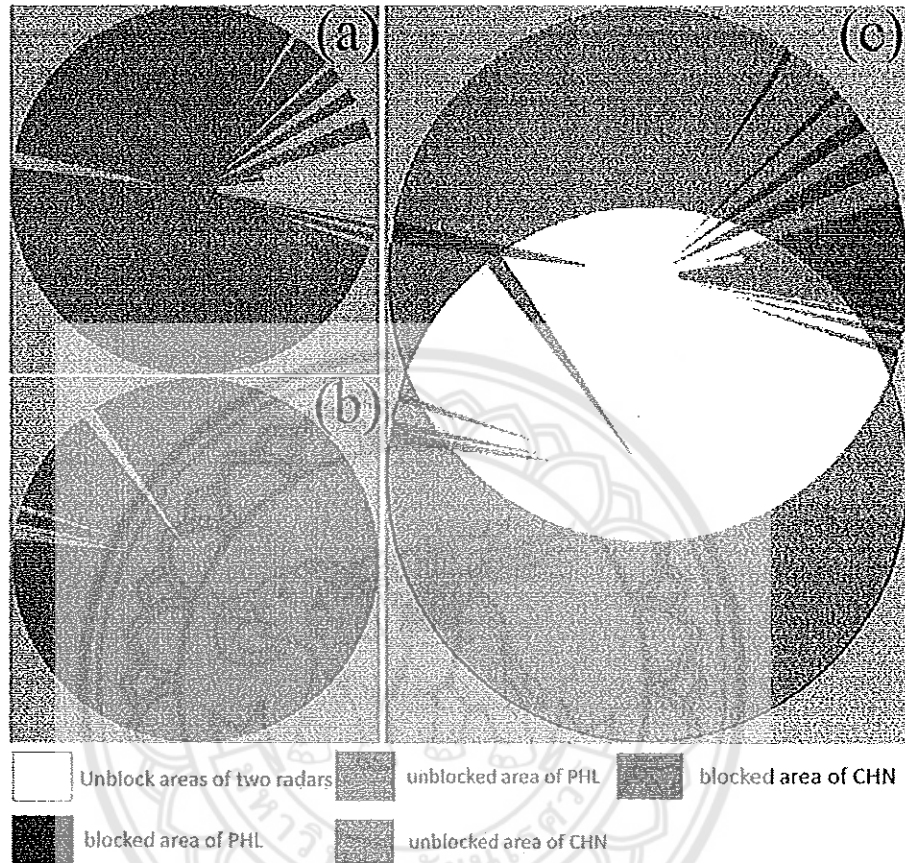
ภาพที่ 4.26 ผลการโมเสคค่าการสะท้อนเรดาร์วันที่ 26 กรกฎาคม 2560 เวลา 11:30 (a) ผลลัพธ์ภาพโมเสคสองสถานี (b) ภาพค่าการสะท้อนจากเรดาร์สถานีพิฆณโลก (c) ภาพค่าการสะท้อนจากเรดาร์สถานีชัยนาท วงรีแสดงถึงกลุ่มฝน mesoscale สองกลุ่มในช่วงเวลาที่ทำการบันทึกข้อมูลภาพ

เมื่อทำการสกัดข้อมูลค่าการสะท้อนในกระบวนการ 4.1.1 ข้อมูลภาพทั้งสองได้ถูกนำมาโมเสคกันโดยใช้ค่าสูงสุดของการสะท้อนในแต่ละตำแหน่งจุดภาพ ดังภาพ 4.26(a) จะพบว่ายังมีแนวเส้นกำกับระยะทางสีแดงที่ยังคงเหลือมาจากกระบวนการสกัดค่าการสะท้อน ทำให้จำเป็นต้องกำจัดเส้นเหล่านี้ออกไปด้วยฟังก์ชัน cv2.fitEllipse ในไลบรารี OpenCV นอกจากนี้ข้อมูลยังถูกกำจัดค่า radar flare ตามกระบวนการที่กล่าวมาในข้างต้น



ภาพที่ 4.27 ข้อมูลที่ใช้ในแต่ละตำแหน่งของเรดาร์โมเสค วันที่ 26 กรกฎาคม 2560 เวลา 11:30 จะเห็นได้ว่าการใช้ค่าสูงสุดของแต่ละเรดาร์วัดได้เป็นค่าตัวแทนในแต่ละจุดภาพของภาพโมเสคดังภาพที่ 4.27 นั้น ทำให้โอกาสของค่าการสะท้อนจากเรดาร์ชยันตสามารถนำมาใช้ในพื้นที่ใกล้เคียงกับตำแหน่งสถานีของเรดาร์พิษณุโลกได้เช่นกัน โดยค่าการสะท้อนที่ไม่เท่ากันนี้มีหลายสาเหตุ เช่น การตั้งค่าอุปกรณ์เรดาร์ไม่เท่ากัน อายุของเครื่องมือ ภายภาพของการตรวจวัด มุมยกของข้อมูล การอ่อนสัญญาณของเรดาร์เนื่องจากกลุ่มฝน เป็นต้น ซึ่งความไม่แน่นอนเหล่านี้ยังคงเป็นปัญหาในการวิจัยต่อไปในอนาคต

4.2.3 การวิเคราะห์การบดบังของเรดาร์บีมสถานีพิษณุโลกและชัยนาท



ภาพที่ 4.28 เรดาร์โมเสกการบดบังภูมิประเทศของการจำลองเรดาร์บีมในมุมมองแรกของเรดาร์สถานีพิษณุโลกและชัยนาท (a) พื้นที่บดบังเรดาร์บีมของสถานีพิษณุโลกแสดงด้วยสี่เทา (b) สถานีเรดาร์ชัยนาท (c) การโมเสกพื้นที่การวิเคราะห์การบดบังเรดาร์บีม

จากภาพที่ 4.28 แสดงให้เห็นถึงการวิเคราะห์การบดบังของเรดาร์บีมเปรียบเทียบกับแบบจำลองภูมิประเทศที่ได้มาจาก SRTM ภาพที่ 4.28 (c) จะพบว่าในทางตะวันออกของเรดาร์พิษณุโลกนั้นมีพื้นที่ที่ถูกภูเขาบดบังทำให้เป็นพื้นที่ที่มีคุณภาพการตรวจวัดที่ต่ำ แม้ว่าจะนำเรดาร์ชัยนาทมาใช้ในการโมเสกก็ยังไม่สามารถที่จะแก้ปัญหาในจุดนี้ได้ ส่วนพื้นที่บริเวณที่เป็นสีขาวตอนกลางแสดงถึงพื้นที่ที่ไม่มีมีการบดบังของภูมิประเทศทำให้มีโอกาสเลือกใช้ค่าการสะท้อนจากทั้งสองเรดาร์ พื้นที่เรดาร์บีมที่ถูกบดบังของสถานีชัยนาทนั้นมีน้อยกว่าพิษณุโลก โดยทางทิศตะวันตกของสถานีสามารถใช้เรดาร์พิษณุโลกเติมเต็มในส่วนที่ถูกภูเขาบังได้

```

def mosaick_beamBlock(r,c,rr,cr,coord_moc,coord_phl,coord_chn,phl_bb,chn_bb,rres,cres):
    # moc=np.zeros((r,c), dtype=np.ints)#-create array output of grey mosaic
    mocg=np.zeros((r,c,1))#-create array output of grey mosaic
    so_index=np.zeros((r,c), dtype=np.ints)#-create array output of used radar (red,green-phl,chn)
    #-loop inside PHL radar dimension to transfer distance to mosaic dimension
    print 'loop inside PHL radar for mosaicking...'
    for i in range(rr):
        for j in range(cr):
            row_no=round((coord_moc[0]-coord_phl[0])/rres)+i
            col_no=round((coord_phl[3]-coord_moc[3])/cres)+j
            if phl_bb[i,j]==1:
                mocg[row_no,col_no]=1 #-block
                so_index[row_no,col_no]=11 #-11 scan radar PHL and block

            if phl_bb[i,j]==2:
                mocg[row_no,col_no]=2 #-unblock
                so_index[row_no,col_no]=12 #-12 scan radar PHL and unblock

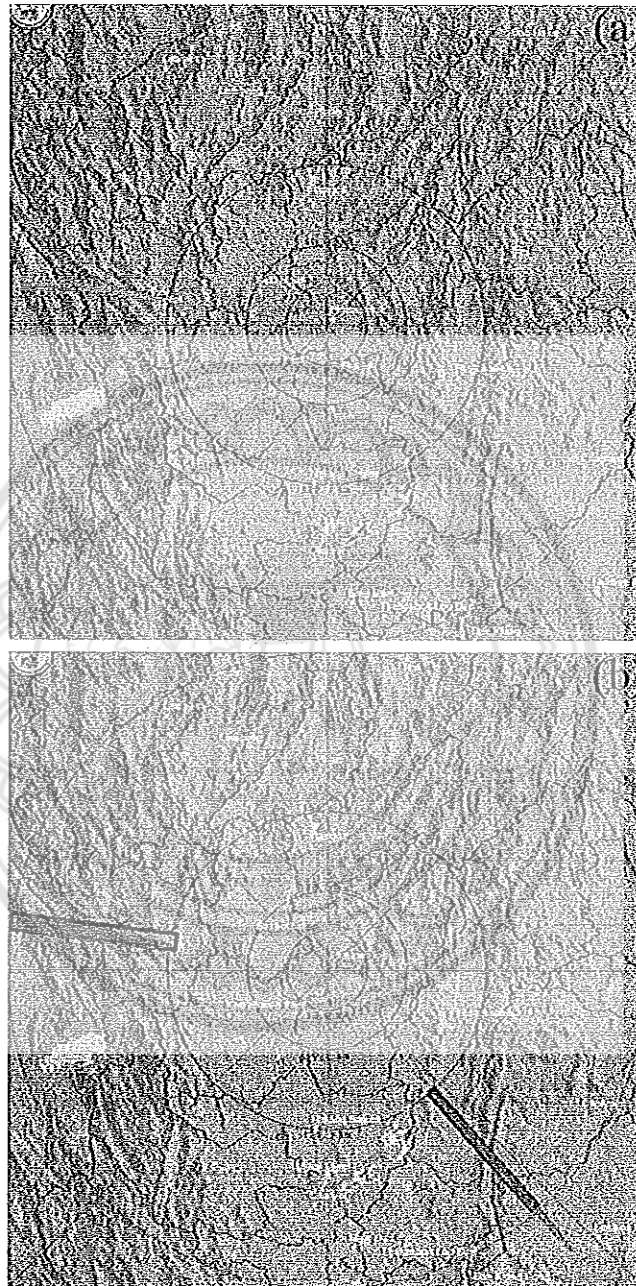
    #-loop inside CHN radar dimension to transfer distance to mosaic dimension
    print 'loop inside CHN radar for mosaicking...'
    for i in range(rr):
        for j in range(cr):
            row_no=round((coord_moc[0]-coord_chn[0])/rres)+i
            col_no=round((coord_chn[3]-coord_moc[3])/cres)+j
            if row_no>=392:break #-just for safe
            if mocg[row_no,col_no]==1 and chn_bb[i,j]==1:
                mocg[row_no,col_no]=1
                so_index[row_no,col_no]=11#-block the radar
                continue
            if mocg[row_no,col_no]==1 and chn_bb[i,j]==2:
                mocg[row_no,col_no]=2
                so_index[row_no,col_no]=22 #-using chainat and unblock
                continue
            if mocg[row_no,col_no]==2 and chn_bb[i,j]==1:
                mocg[row_no,col_no]=2
                so_index[row_no,col_no]=12 #-using PHL and unblock
                continue
            if mocg[row_no,col_no]==2 and chn_bb[i,j]==2:
                mocg[row_no,col_no]=2
                so_index[row_no,col_no]=12#-unblock the radar
                continue
            if mocg[row_no,col_no]==0 and chn_bb[i,j]==1:
                mocg[row_no,col_no]=1
                so_index[row_no,col_no]=21 #-chn and block
                continue
            if mocg[row_no,col_no]==0 and chn_bb[i,j]==2:
                mocg[row_no,col_no]=2
                so_index[row_no,col_no]=22 #-using chainat and unblock
                continue

    return mocg,so_index
    
```

ภาพที่ 4.29 โค้ดโมเสคเรดาร์โมเสคการบดบังภูมิประเทศของการจำลองเรดาร์บีมในมุกแรกๆของเรดาร์สถานีพิษณุโลกและชัยนาท

หลังจากได้คำนวณพื้นที่เรดาร์บีมที่ถูกบดบังจากภูมิประเทศแล้ว จะนำเข้ามาสู่ฟังก์ชันที่พัฒนาขึ้นมาเพื่อทำการโมเสคดังภาพที่ 4.29

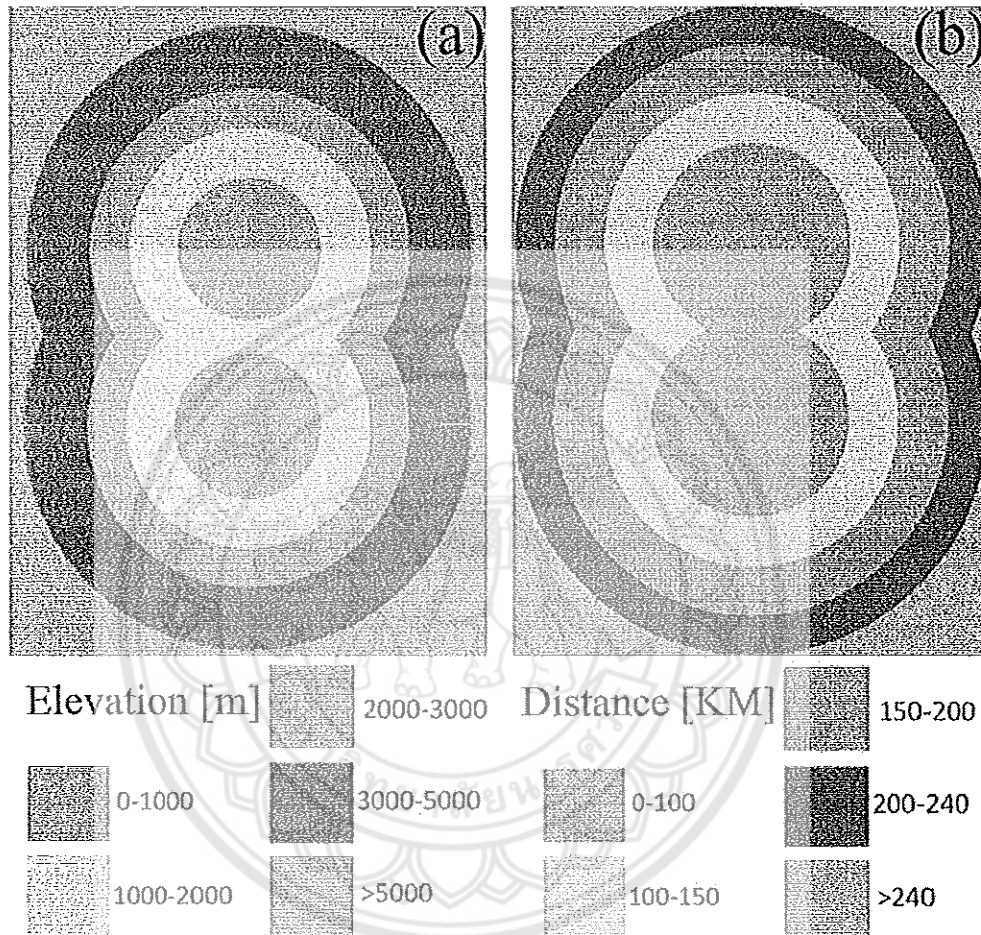
4.2.4 การตรวจ Radar Flares



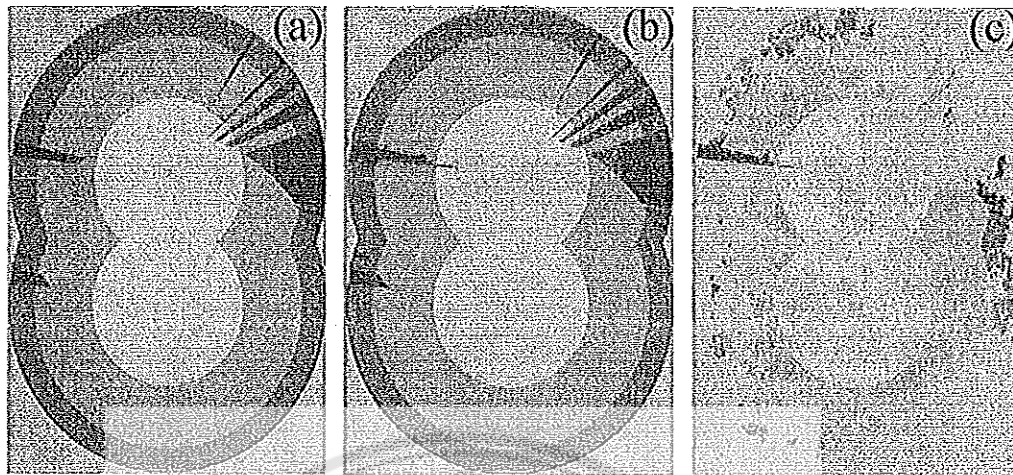
ภาพที่ 4.30 การตรวจพบ radar flares (a) ภาพการสะท้อนจากเรดาร์ดั้งเดิมตรวจวัดในวันที่ 23 มิถุนายน 2562 เวลา 12:25 a.m. (b) การตรวจเจอ radar flares ด้วย best-fitting ellipses และ best-fitting rectangles.

พื้นที่ที่มี radar flares จะเปลี่ยนแปลงแตกต่างกันไปในแต่ละช่วงเวลา ดังนั้นพื้นที่นี้จะเป็นพื้นที่ที่ทำให้คุณภาพของการตรวจเรดาร์นั้นต่ำเนื่องจากไม่ใช่กลุ่มฝนจริง ดังภาพที่ 4.30 มีการตรวจพบ radar flares สองตำแหน่ง สองตำแหน่งนี้จะถูกนำไปรวมกับค่าคุณภาพของจุดภาพที่คำนวณได้ต่อไป

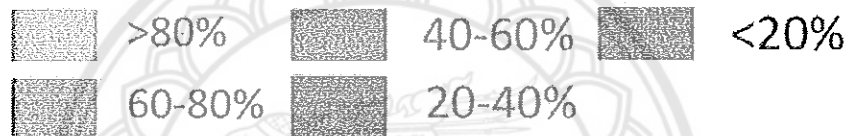
4.1.5 แผนที่ Radar Quality Index Fields



ภาพที่ 4.31 แผนที่คุณภาพเรดาร์ที่คำนวณจาก (a) ความสูงของเรดาร์บีมขณะที่เคลื่อนที่ไปในบรรยากาศ (b) ระยะทางของเรดาร์บีมที่ห่างจากสถานีรับส่งสัญญาณเรดาร์
คุณภาพของข้อมูลเรดาร์โมเสกขึ้นอยู่กับระยะทางและความสูงของเรดาร์บีม ดังภาพที่ 4.31 แสดงถึงผลลัพธ์การคำนวณระยะทางและความสูงของเรดาร์บีมที่เคลื่อนที่ไปในชั้นบรรยากาศ



Quality of mosaicked radar reflectivity (%)



ภาพที่ 4.32 ผลลัพธ์ Radar quality index (a) ดัชนีชี้วัดคุณภาพของเรดาร์โมเสคจากสามปัจจัย (b) ผลลัพธ์การรวมดัชนีชี้วัดกับพื้นที่ radar flares (c) คุณภาพของค่าการสะท้อนตามดัชนีชี้วัดวันที่ 26 กรกฎาคม 2560 เวลา 11.30.

ภาพที่ 4.32 แสดงผลลัพธ์ของดัชนีชี้วัดคุณภาพผลิตภัณฑ์เรดาร์โมเสค โดยเมื่อนำผลลัพธ์จาก radar flares มาใช้กับค่าการสะท้อนเรดาร์โมเสคในภาพ 4.32(c) จะพบว่าสามารถบอกคุณภาพในระดับร้อยละ ซึ่งทำให้ผู้ใช้งานได้มีความตระหนักว่าควรเชื่อถือข้อมูลในระยะขอบเขตเท่าใด นอกจากนี้ บริเวณที่ไม่ใช่พื้นที่ฝนจะพบว่ามีค่าคุณภาพค่อนข้างต่ำนั้นคือต่ำกว่า 20% ซึ่งจะพบในทางทิศตะวันตกและตะวันออกของเรดาร์พิชฌุโลก

บทที่ 5 สรุปผลการศึกษาและข้อเสนอแนะ

5.1 สรุปผลการศึกษา

งานวิจัยครั้งนี้มุ่งศึกษาและพัฒนากระบวนการทำแผนที่โมเสคน้ำฝนจากข้อมูลเรดาร์ตรวจอากาศจำนวน 5 สถานีครอบคลุมพื้นที่ลุ่มน้ำเจ้าพระยา ข้อมูลที่ใช้เป็นข้อมูลภาพค่าการสะท้อนเรดาร์ตรวจอากาศในมุกยกแรกที่ตรวจวัดโดยกรมอุตุนิยมวิทยาในช่วงฤดูมรสุม อันเป็นข้อมูลที่เข้าถึงได้อย่างสาธารณะ ได้มีการพัฒนากระบวนการที่เป็นอัลกอริทึมเพื่อที่จะนำไปพัฒนาต่อในการปฏิบัติการแบบเรียลไทม์ของหน่วยงานที่เกี่ยวข้องได้ โดยภาษาคอมพิวเตอร์ที่ใช้ในการพัฒนากระบวนการโมเสคนั้นคือภาษาไพธอน ซึ่งเป็นภาษาที่มีไวยากรณ์ที่เข้าใจได้ง่าย เจ้าหน้าที่ในกรมอุตุนิยมวิทยาสามารถเรียนรู้และดัดแปลงให้เข้ากับข้อมูลเรดาร์ทั่วประเทศได้

5.1.1 ศึกษาและพัฒนากระบวนการพัฒนาแผนที่น้ำฝนจากเรดาร์ตรวจอากาศภาคพื้นดินในพื้นที่ลุ่มน้ำเจ้าพระยาในช่วงฤดูมรสุมเขตร้อน

การพัฒนากระบวนการทำแผนที่โมเสคนั้นใช้ภาษาไพธอนในการพัฒนาทั้งหมด โดยได้ใช้ข้อมูลภาพที่เผยแพร่โดยเว็บไซต์ของกรมอุตุนิยมวิทยาและถูกจัดเก็บเชิงระบบโดยหน่วยงาน สถาบันสารสนเทศทรัพยากรน้ำและการเกษตรองค์การมหาชนซึ่งภาพเหล่านี้ต้องทำการพัฒนากระบวนการสกัดข้อมูลค่าการสะท้อนของภาพดั้งเดิมนั้นก่อน ถึงจะสามารถนำมาประมวลผลต่อไปได้ ผู้วิจัยได้พัฒนากระบวนการเหล่านี้เพื่อเตรียมข้อมูลให้พร้อมก่อนการโมเสค โดยมีการใช้ฟังก์ชัน `fittEllipse` ในไลบรารี `OpenCV` ซึ่งเป็นไลบรารีที่ใช้จัดการข้อมูลภาพเพื่อจัดกลุ่มข้อมูลที่ไม่มีความเกี่ยวข้องกับค่าการสะท้อนของกลุ่มฝน นอกจากนั้นการวิเคราะห์ข้อมูลเรดาร์บีมที่ถูกบดบังโดยภูมิประเทศเพื่อให้ทราบถึงคุณภาพของข้อมูลโดยใช้ไลบรารี `Wradlib` พบข้อจำกัดของข้อมูลเรดาร์ในพื้นที่บริเวณภูเขา

5.1.2 พัฒนารูปร่างข้อมูลน้ำฝนเชิงพื้นที่จากผลลัพธ์ของกระบวนการข้อที่ 1 ให้เหมาะสมกับการนำไปใช้ในงานวิเคราะห์ปริมาณน้ำฝนเชิงพื้นที่ของระบบลุ่มน้ำ

ผลิตภัณฑ์ที่ได้จากงานวิจัยชิ้นนี้นั้นได้แก่ แผนที่เรดาร์โมเสคค่าการสะท้อน แผนที่คุณภาพการโมเสคเรดาร์ แผนที่ตำแหน่งข้อมูลเรดาร์ที่ถูกใช้ในแต่ละจุดภาพและแผนที่เรดาร์บีมที่ถูกบดบังจากภูมิประเทศ โดยผลิตภัณฑ์ทั้งหมดเหล่านี้สร้างขึ้นมาจากการพัฒนากระบวนการในภาษาไพธอน ข้อมูลแผนที่ที่เป็นผลลัพธ์จากการพัฒนากระบวนการนี้อยู่ในรูปของ `Tif` ที่สามารถนำข้อมูลไปจัดทำเป็นแผนที่เพื่อนำไปวิเคราะห์เรื่องลักษณะฝนต่อไปได้ นอกจากนั้นผู้วิจัยยังได้นำข้อมูลผลิตภัณฑ์ที่ได้

พัฒนามาวิเคราะห์ถึงลักษณะฝนที่เกิดขึ้นในสี่เหตุการณ์ฝนที่สำคัญของปี 2561 พบว่าฝนที่เกิดในช่วงพายุซินติญันมีลักษณะของจำนวนเมฆ convective ที่มากกว่าและมีคุณลักษณะของเมฆที่โดดเด่นในด้านขนาดพื้นที่และความยาวของเมฆที่มากกว่าฝนจากเหตุการณ์อื่น ซึ่งผลจากการค้นพบนี้สามารถนำไปเป็นต้นแบบการวิเคราะห์และเฝ้าติดตามในสภาวะที่ประเทศไทยมีพายุเคลื่อนที่ผ่านเข้ามาได้ เนื่องจากโค้ดที่ได้พัฒนานี้สามารถนำไปสร้างข้อมูลแบบเรียลไทม์ ซึ่งจะปรับปรุงยับยั้งต่อการวิเคราะห์เพื่อเฝ้าระวังพายุที่รุนแรงจากการสลายตัวของพายุที่เกิดจากพายุเขตร้อนหรือร่องมรสุมได้อย่างมีประสิทธิภาพ

5.2 ข้อเสนอแนะ

1. ข้อมูลที่ใช้เป็นข้อมูลในมุกแรกทำให้การพัฒนากระบวนการโมเสครดาร์สามารถทำได้อย่างจำกัด เนื่องจากมุกแรกนั้นจะปนเปื้อนค่าการสะท้อนที่ไม่ใช่กลุ่มฝนด้วยหรืออาจจะได้รับผลกระทบของการบดบังจากภูมิประเทศ ดังนั้นในการวิจัยครั้งต่อไปควรใช้ข้อมูลมุกอื่นๆเพื่อสร้างข้อมูลโมเสคแบบสามมิติ
2. ข้อมูลภาพเรดาร์นั้นเป็นเพียงค่าการสะท้อนของกลุ่มฝนที่ยังไม่มีความละเอียด ซึ่งเพียงพอต่อการพัฒนากระบวนการโมเสครดาร์ตามวัตถุประสงค์ของงานวิจัยครั้งนี้เพื่อนำไปประยุกต์ใช้ในการหา ลักษณะเมฆฝนในช่วงเหตุการณ์ที่สนใจ อย่างไรก็ตามหากต้องการพัฒนากระบวนการโมเสคเพื่อวัดเทียบปริมาณฝนจากข้อมูลการสำรวจระยะไกลกับข้อมูลฝนภาคพื้นดินจำเป็นต้องใช้ข้อมูลเรดาร์ที่สามารถเข้าถึงโครงสร้างของการจัดเก็บข้อมูลหรือข้อมูลลักษณะ Volume file ที่มีหลายมุก ซึ่งควรมีการวิจัยและพัฒนากระบวนการต่อไป
3. การวิจัยครั้งต่อไปควรนำโค้ดที่ได้ไปพัฒนาระบบภูมิสารสนเทศแบบเรียลไทม์ที่คอยเฝ้าระวัง ลักษณะกลุ่มฝนที่อันตรายที่สามารถให้บริการแก่ประชาชน นอกจากนี้ควรสร้างระบบจัดเก็บฐานข้อมูลเชิงพื้นที่นี้ด้วยการสร้างอัลกอริทึมอัตโนมัติเพื่อสกัดข้อมูลกลุ่มฝนให้เป็นข้อมูลที่สามารถเรียกใช้ในระบบสารสนเทศภูมิศาสตร์อันจะเป็นประโยชน์ต่อการวิเคราะห์ข้อมูล Bigdata จากคลังข้อมูลที่สร้างขึ้นนี้เพื่อลดภัยพิบัติได้

เอกสารอ้างอิง

- Anagnostou, P., W. F. Krajewski, D. J. Seo, and E. R. Johnson, 1998: Mean-field rain-fall bias studies for WSR-88D. *J. Hydrol. Eng.*, 3, 149–159.
- Battan, L.J., Radar Observation of the Atmosphere, The University of Chicago Press, 1973.
- Brandes, E., 1975: Optimizing rainfall estimates with the aid of radar. *J. Appl. Meteor.*, 14, 2038–2052.
- Chokngamwong, R. and Chiu, S. L., 2008: Thailand Daily Rainfall and Comparison with TRMM Products. *J. Hydromet.*, 9, 256-266.
- Chumchean, S., A. Sharma, and A. Seed, 2003: Radar rainfall error variance and its impact on radar rainfall calibration. *J. Phys. Chem. Earth.*, 28, 27–39.
- Chumchean, S., P. Aungsuratana, A. Khommuang and R. Hanchoo Wong, 2009: Study of Rain-Cloud Characteristics Using Weather Radar Data, 18th World IMACS / MODSIM Congress, Cairns, Australia, 3682-3689.
- Doviak, R.J., and D.S. Zmic, Doppler Radar and Weather Observations, Academic Press, 1993.
- Friedrich, K., M. Hagen, and T. Einfalt, 2006: A quality control concept for radar reflectivity, polarimetric parameters, and Doppler velocity. *J. Atmos. Oceanic Technol.*, 23, 865–887.
- Fulton, R. A., Breidenbach, J. P., Seo, D.-J., Miller, D. A. & O'Bannon, T., 1998: The WSR-88D rainfall algorithm. *Wea. Forecasting.*, 13(2), 377–395.
- Hanchoo Wong, R., U. Weesakul, S. Chumchean, 2012: Bias correction of radar rainfall estimates based on a geostatistical technique. *ScienceAsia.*, 38, 373-385.
- Komori, D., S. Nakamura, M. Kiguchi, D. Yamazaki, S. Suzuki, A. Kawasaki, K. Oki and T. Oki, 2012: Characteristics of the 2011 Chao Phraya River flood in Central Thailand. *HRL* 6, 41–46 (2012).
- Mahavik, N., T. Satomura and S. Baimoung, 2011: Investigation ZR relationship by using weather radar in Thailand. *International symposium on Southeast Asian water environment*, 9 (part II), 212-217.

- Mahavik, N., T. Satomura, S. Shige, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2014: Rainfall pattern over the middle of Indochina Peninsula during 2009–2010 summer monsoon. *HRL.*, 8, 57–63.
- Mahavik, N., 2015: Study on rainfall over the middle of the Indo-China Peninsula during summer monsoon by producing gauge-calibrated ground-based radar data, Ph.D. Dissertation, Kyoto University, Japan, 114pp.
- Marshall, J., and W. Palmer, 1948: The distribution of raindrops with size. *J. Meteorol.*, 5, 165–166.
- Okumura, K., T. Satomura, T. Oki, and W. Khantiyanan, 2003: Diurnal variation of precipitation by moving mesoscale systems: radar observations in northern Thailand. *Geophys. Res. Lett.*, 30.
- Rinehart, R.E., *Radar for Meteorologists*, Rinehart Publishing, 1997.
- Satomura, T., 2000: Diurnal variation of precipitation over the Indo-China Peninsula: two-dimensional numerical simulation. *J. Meteor. Soc. Japan.*, 78, 461–475.
- Satomura, T., M. Katsumata, S. Mori, S. Yokoi, J. Matsumoto, S. Ogino, and Y. Kamimura, 2013: To understand typhoons' behavior over Indochina. *JDR*, 8, 153–154.
- Satomura, T., K. Yamamoto, B. Sysouphanthavong, and S. Phonevilay, 2011: Diurnal variation of radar echo area in the middle of Indochina. *J. Meteor. Soc. Japan.*, 89a, 299–305.
- Seo, B.-C., W.F. Krajewski, A. Kruger, P. Domaszczynski, J.A. Smith, and M. Steiner, 2010: Radar-rainfall estimation algorithms of Hydro-NEXRAD. *J. Hydroinformatics.*, 13(2), 277–291.
- Tantanee, S. and Prakarnrat, S., 2008: Using GIS Buffer Technique to Improve Rainfall-Radar Reflectivity Relationship Estimation. *WSEAS TRANSACTIONS on FLUID MECHANICS*, 4(3), 304–313.
- Veerakachen, W., Raksapatcharawong, M. and Seto, S. (2014). Performance evaluation of Global Satellite Mapping of Precipitation (GSMaP) products over the Chaophraya River basin, Thailand. *HRL.*, 8(1), 39–44.

Yokoi, S., and T. Satomura, 2008: Geographical distribution of variance of intraseasonal variations in western Indochina as revealed from radar reflectivity data. *J. Climate.*, 21, 5154–5161.

กระทรวงมหาดไทย. 2555. http://disaster.go.th/dpm/flood/news/flood_lastnews.html.





1.กระบวนการโมเสคข้อมูลภาพเรดาร์

-*- coding: utf-8 -*-

'''

Created on Thu Jan 10 11:13:07 2019

We do complete all process of storm analysis:

we do mosaic of 5 radars both reflectivity and flares by detect and cleaning flares

we report quality of mosaic by detect and reject bad radar images and report number of radar used

incase not available radars as well

we screen broken images by rejecting those files out of mosaicking process using

threshold of exceeding 10,000 square km.

we do extract convective and stratiform clouds. we report cloud size, lenght, area, center storm in geographic

spatio-temporal can be anylazed from the extracted data.

during the ellipse fitting process, we add criteria to avoid overlapping small ellipse

inside of larger ellipse."20190103CVextraction.py"

@author: Dr.Nattapon Mahavik, Naresuan University, Thailand

'''

import numpy as np

import cv2

from path import path

import os

from PIL import Image

import sys

#-----

#plot opencv

def plotcv(label,image):

 #to plot image in opencv

 cv2.namedWindow(label, cv2.WINDOW_NORMAL) #-resize manually

 cv2.imshow(label,image)

 cv2.waitKey(0)

 cv2.destroyAllWindows()

#-----

def convertGif2Jpg(inpgif_dir,radname):

```

print "convert Gif to JPG..." + radname

#-convert gif to jpg
for f in inpgif_dir.files(pattern="*.gif"):
    head, imfile = os.path.split(f)
    out_fn=imfile[0:16]+'jpg'
    try:
        Image.open(inpgif_dir+imfile).convert('RGB').save(inpgif_dir+out_fn)
    except Exception:
        sys.exc_clear()
#    Image.open(inpgif_dir+imfile).convert('RGB').save(inpgif_dir+out_fn)
    print out_fn

#-----
def extractRGB(img,text_rad):
    print "---->processing extractRGB--- ",text_rad
    #-nearly perfect color shades
    #-BGR
    boundaries = [
        ([0, 153, 0], [102, 255, 102]), #green
        ([0, 204, 204], [102, 255, 255]),#yellow
        ([0, 120, 230], [102, 178, 255]),#orange
        ([0, 0, 102], [51, 51, 255]),#red
    ]

    #-create combined img of reflectivity
    r,c,b=img.shape
    rf=np.zeros((r,c),dtype = "uint8")#-gray
    rfc=np.zeros((img.shape),dtype = "uint8")#-color

#-----
# to extract reflectivity by looping over the boundaries
i=0
for (lower, upper) in boundaries:
    i +=1
    # create NumPy arrays from the boundaries

```

```

lower = np.array(lower, dtype = "uint8")
upper = np.array(upper, dtype = "uint8")
# find the colors within the specified boundaries and apply
# the mask
mask = cv2.inRange(img, lower, upper)
output = cv2.bitwise_and(img, img, mask = mask)
#-store detected color reflectivity to output in color
rfc += output

#-store detected color reflectivity to output in gray
gray=cv2.cvtColor( output, cv2.COLOR_RGB2GRAY ) #-convert rgb to gray
gray[gray>0]=i
rf += gray

return rf,rfc
#-----
#-----
def auto_remove_flare(eroded_images,img,rf,contours,text_rad):
    print "---->processing auto_remove_flare---",text_rad
    #-----
    #-Step 1: apply automatically removing radar flare with criteria
    #-calculate area of one pixel to find criteria of filtering only biggest possible flare
    area_rad=480.0*480.0 #-unit in sq.km.
    npix=eroded_images.size #-num of pix
    res=area_rad/npix*1.0 #-unit of 1 pix per sq km.>>0.4365 sq.km/pixel at resolution of 800*800
    #-create mask of possible flare
    mask = np.zeros(eroded_images.shape, dtype=np.uint8)
    # min_area_rect=500 #- unit of number pixels
    # max_area_rect=6000 #- unit of number pixels
    #-we try parameters for optimize flare detections in max_eccen1&min_ellipse
    max_eccen1=0.25 #-unitless 0.25 is suitable, 0.30 is fail due to inclusion of elongate rain
    min_ellipse_h=20.0 #-unit of km 50.0
    max_ellipse_h=250.0 #-unit of km 250.0

```

```

SumFA=0 #-to sum total area of detected flare
ct=0 #-to count number of detected flare
for h,cnt in enumerate(contours):
    if cnt.shape[0]<5: #-prevent number of contour point less than 5
        continue

    #-apply rotated rectangle
    rect = cv2.minAreaRect(cnt) #-need to filter area of rect that more than defined
    box=cv2.boxPoints(rect)
    box = np.int0(box)
    area_rect=rect[1][0]*rect[1][1]

    #-apply fitting ellipse to contour
    ellipse = cv2.fitEllipse(cnt)
    eccentricity=ellipse[1][0]/ellipse[1][1] #-width/height
    ellipse_h=ellipse[1][1]*res #-height of ellipse in km

    #-to find radar flare meet criteria
    # if ellipse_h>min_ellipse_h and ellipse_h<max_ellipse_h and eccentricity<max_eccen1:
    #500,3000,0.1 need to be tuned
    if ellipse_h>min_ellipse_h and eccentricity<max_eccen1: #-try
        #-to avoid overlapping ellipse inside larger ellipse from "20190103CVextraction.py"
        M = cv2.moments(cnt) #-Monment of contour
        cx = int(M['m10']/M['m00']) #xgrid[cx], ygrid[481-cy]
        cy = int(M['m01']/M['m00'])
        # print 'cx,cy: ',cx,': ',cy,moc_rad[cy,cx]
        if moc_rad[cy,cx][0]==255:
            print '----invalid ellipse'
            continue

        # print 'possible flare with area: ', area_rect*res , ' sq.km.'
        # print cnt.shape[0], ';',area_rect,';',eccentricity,': ', ellipse_h
        #-plot possible flare area, rotated rectangle, fitting ellipse
        cv2.drawContours(img,[cnt],0,(255,0,0),-1)

        #-plot rotated box over possible flare area
    
```

```

cv2.drawContours(img,[box],0,(0,0,0),3)

#-plot fitting ellipse over possible flare area
ellipse = cv2.fitEllipse(cnt)
cv2.ellipse(img,ellipse,(0,0,255),2)
#   plotcv('Original 1PPI radar vs detected possible flare bounded by box and ellipse'+text_rad,
#         np.hstack([img, img]))

#-create mask
roi = np.array([cnt], dtype=np.int32)
#white = (255, 255, 255)

cv2.fillPoly(mask, roi, 255) #-fill mask by desired color
#plotcv('mask'+text_rad,mask)

#-sum detected flare area
SumFA += (area_rect*res)

#-count number of detected flares
ct +=1

print '----->number of detected flare: ',ct
print '----->total area of detected flare: ',SumFA,' sq.km.'
# print '-----'

#--apply mask and use the result as input in the next process
#-apply mask resulting from automatic flare detection
#-we add dilate step to mask sure that our mask is large enough to cover flares
#****in the future, before apply flare to rf, we need to look for homogenous of green and red
#****if there are more percent of red, we don't apply that roi to rf
kernel = np.ones((9,9),np.uint8) #-15,15 try and error to remove the flare
mask = cv2.dilate(mask,kernel,iterations = 1)
mask_inv = cv2.bitwise_not(mask) #-inverse mask
masked1st_rf = cv2.bitwise_and(rf, mask_inv)
# plotcv('masked1st_rf'+text_rad,masked1st_rf*50)

```

```

flare_detected=ct

# return masked1st_rf,flare_detected
return masked1st_rf,mask,flare_detected,SumFA

#-----
def cleaning_rad(r,c,eroded_images,final_rf,flare_detected,text_rad):
    print "---processing final cleaning_rad--- ",text_rad
    #-create offset-bound to combine with eroded used for masking reflectivity area
    img_offset_bnd=np.ones((r,c),dtype = "uint8")#-gray
    offpix=5
    img_offset_bnd[0:offpix,:]=0 #-upper row
    img_offset_bnd[r-offpix:r,:]=0 #-bottom row
    img_offset_bnd[:,0:offpix]=0 #-left column
    img_offset_bnd[:,c-offpix:c]=0 #-left column
    #plotcv("offset boundary used for mask"+text_rad,img_offset_bnd*200)

    #-combining eroded_images with img_offset (cleaning process)
    mask_er_off=img_offset_bnd*eroded_images
    mask_er_off[mask_er_off>0]=1
    #plotcv("offset boundary used for mask"+text_rad,(mask_er_off*255))

    #-apply mask_er_off to clean redline of radar will be used for masiack
    final_rf=final_rf*mask_er_off
    # plotcv("final radar reflectivity after cleaning"+text_rad,(final_rf*50))
    #-----

    #-combining flare_dected with img_offset will be used in mosaic ***
    offset=img_offset_bnd
    offset[offset==0] = 255
    offset[offset==1]=0
    # plotcv("offset boundary "+text_rad,offset*255)

    #-combine offset boundary and detected flares will be used in mosaic ***
    flare_offset=offset+flare_detected
    # plotcv("combine flare_detected and offset boundary"+text_rad,flare_offset)

```



```
return flare_offset,final_rf,flare_detected
#-----
#-----
def main_cleaning_program(img,text_rad):
    nf=1 #-mean we will use this flare
    print "---processing main_cleaning_program--- "+text_rad
    #-----
    #this main programm will do the following processes
    #1. extractRGB 2.preparing detecting flares 3.Apply flares detection
    #4. Apply cleaning flares and red lines
    #-----
    #-Apply extract radar reflectivity
    rf,rfc=extractRGB(img,text_rad)
#    plotcv('extracted reflectivity phitsanulok',rf*50)
    #-----
    ##### Apply detecting flares
    #-1.apply morphology to combined reflectivity to find contour
    #Gray Scaling
    gray_images =cv2.cvtColor(rfc, cv2.COLOR_RGB2GRAY)
    #threshold need to adjust
    ret,thresh = cv2.threshold(gray_images,0,255,0)
    #-apply close kernel 7x7
    kernel = np.ones((7,7), np.uint8)
    closed_images = cv2.morphologyEx(thresh, cv2.MORPH_CLOSE,kernel)
    #-apply erosion, output will be used in the detection of flare
    kernel = np.ones((3,3),np.uint8)
    eroded_images = cv2.erode(closed_images,kernel,iterations = 1)
    #plot('closed_images 7x7 vs eroded_images 7x7',np.hstack([closed_images, eroded_images]))
#-50 is just factor for plotting
    #-apply findcontours
    im2, contours, hierarchy =
cv2.findContours(eroded_images,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
#    plotcv('eroded reflectivity phitsanulok',eroded_images)
    #-----
```

```

#-2.Apply flares detection
final_rf,flare_detected,ct,SumFA=auto_remove_flare(eroded_images,img,rf,contours,text_rad)

#-final result
if SumFA>13000.00: #-13000 sq.km. is severe flare, we should not use in process
    print "!!!---- radar: ",text_rad, "-> seem to be broken due to lots of flares !!! not used"
    final_rf=np.zeros((img.shape[0],img.shape[1]), np.uint8)
    flare_detected=np.zeros((img.shape[0],img.shape[1]), np.uint8)
    nf=9 #-to be flag of broken radar, we mean don't use this flare

#-----

#-3.Apply cleaning flares and red lines
r,c,b=img.shape

flare_offset,final_rf,flare_detected=cleaning_rad(r,c,eroded_images,final_rf,flare_detected,text_rad)
# plotcv("final radar reflectivity after cleaning"+text_rad,(final_rf*50))
# plotcv("combine flare_detected and offset boundary"+text_rad,flare_offset)
#-these flare_offset,final_rf will be used in mosaic 5 stations
#-----
return flare_offset,final_rf,ct,SumFA,flare_detected,nf

#-----
#-----

#-Mosaic multiple radar images
def mosaicRad(img,coord_rad, shape_rad, coord_mo,res, mocg,text_rad,mocg_index,num_rad):
    r,c=mocg.shape
    rr,cr=img.shape
    print '---processing mosaicking mosaicRad---'+text_rad
    # r,c is mosaic dimension in pixel
    # rr,cr is radr dimension in pixel
    #-loop inside radar dimension to transfer distance to mosaic dimension

    #top, bottom, right, left
    cres_mo=(coord_mo[2]-coord_mo[3])/c #-in geo
    rres_mo=(coord_mo[0]-coord_mo[1])/r #-in geo

    cres_rad=(coord_rad[2]-coord_rad[3])/cr #-in geo
    
```

```

rres_rad=(coord_rad[0]-coord_rad[1])/rr #in geo

if num_rad==2:
    #-to refine "mo_radg" by removing red guideline remaining from Lamphun radar
    #-the kernel size will surely affect to rain pixel when compared to raw data
    #-in the future we must refine this work because applying kernel will defect real size of rain
    kernel = np.ones((2,2), np.uint8)
    img = cv2.morphologyEx(img, cv2.MORPH_OPEN, kernel)

for i in range(shape_rad[0]): #-search in row
    for j in range(shape_rad[1]): #-search in col
        #-transform radar pixel to mosiack pixels
        row_mo= round((coord_mo[0]-(coord_rad[0]-(i*rres_rad)))/rres_mo)
        col_mo= round(((coord_rad[3]+(j*cres_rad))-coord_mo[3])/cres_mo)
#         print "row,col mo:",row_mo,'---',col_mo

        if col_mo>=c-1:continue
        if row_mo>=r-1:continue

        #-assign only maimum value to pixel of mosaiced result
        if img[i,j]>mocg[row_mo,col_mo]:
            mocg[row_mo,col_mo]=img[i,j] #-for grey
            mocg_index[row_mo,col_mo]=num_rad
#         mocc[row_mo,col_mo,:]=imc[i,j,:] #-assign extracted radar to color mosaic

#     print '-----'
    return mocg,mocg_index
#-----
def
extract_storm(inpath,coord_mo,res_mo,img,moc_rad,st,outpath,outtext,outtext_area,cv_file_name,st_
file_name,file_name):
    print '---processing extract convective storm for mosaicked--- '
    #This program is to extract convective properties
    # write to output file for convective area in images and text file for cv properties
    #-----

```

```

#-----
#-apply morphology to combined reflectivity to find contour
#-green, yellow, orange, red; 1, 2, 3, 4 very important
cv=img
cv[cv<2]=0
cv[cv>=2]=255
# plotcv('cv',cv)

#threshold need to adjust
ret,thresh = cv2.threshold(cv,200,255,0)
# plotcv('thresh',thresh)

#-apply close kernel 15x15
kernel = np.ones((15,15), np.uint8)
close_im = cv2.morphologyEx(thresh, cv2.MORPH_CLOSE,kernel)
# plotcv('result after apply close morphology15x15...',close_im)

#-add to try find contour
kernel = np.ones((3,3), np.uint8)
close_im = cv2.morphologyEx(close_im, cv2.MORPH_OPEN, kernel)
# plotcv('result after apply opening morphology3x3...',close_im)

#-----
# find contour of convective storm
#-next program must calculate length and area respect to latitude
res=0.73 #km for length one side of pixel measured in Arcgis
sqkm=0.730*0.730 #sq.km per pixel side from 0.00695 degree

#-create mask of possible flare
mask = np.zeros(close_im.shape, dtype=np.uint8)
#-apply findcontours
joker,contours, hierarchy = cv2.findContours(close_im, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)

#-setting parameters to find radar flare need to be tuned to find appropriate values

```

```
min_area_rect=10 #- unit of sq km
max_area_rect=6000 #- unit of number pixels
max_eccen=0.1 #-unitless
min_ellipse_h=5.0 #-unit of km
max_ellipse_h=1000.0 #-unit of km

cv_poly = np.zeros(close_im.shape, dtype=np.uint8) #-to store roi of cv, will use in spatial analysis
ct=0 #-to count number of detected flare
for h,cnt in enumerate(contours):
    # print 'number of contour points:', cnt.shape[0]
    if cnt.shape[0]<5: #-prevent number of contour point less than 5
        continue
    print 'number of contour points:', cnt.shape[0]

    #-apply rotated rectangle
    rect = cv2.minAreaRect(cnt) #-need to filter area of rect that more than defined
    box=cv2.boxPoints(rect)
    box = np.int0(box)
    area_rect=rect[1][0]*rect[1][1]*sqkm #-unit in sq km

    #-apply fitting ellipse to contour
    ellipse = cv2.fitEllipse(cnt)
    eccentricity=ellipse[1][0]/ellipse[1][1] #-width/height
    ellipse_h=ellipse[1][1]*res #-height of ellipse in km

    #-apply criteria to find convective cloud
    #-create mask of individual detected cloud
    mask = np.zeros(close_im.shape, dtype=np.uint8)
    if ellipse_h>min_ellipse_h and ellipse_h<max_ellipse_h and eccentricity>max_eccen and
area_rect>min_area_rect: #500,3000,0.1 need to be tuned
        #-to avoid overlapping ellipse inside larger ellipse
        M = cv2.moments(cnt) #-Monment of contour
        cx = int(M['m10']/M['m00']) #xgrid[cx], ygrid[481-cy]
        cy = int(M['m01']/M['m00'])
    # print 'cx,cy: ',cx,' ',cy,moc_rad[cy,cx]
```

```

if moc_rad[cy,cx][0]==255:
    print '----invalid ellipse'
    continue

#-plot possible cloud area, rotated rectangle, fitting ellipse
cv2.drawContours(moc_rad,[cnt],0,(255,0,0),-1) #-by filling color

#-plot rotated box over possible cloud area
#   cv2.drawContours(img,[box],0,(0,0,0),3) #-just show
#-to find center x,y of convective contour
M = cv2.moments(cnt) #-Monment of contour
cx = int(M['m10']/M['m00']) #xgrid[cx], ygrid[481-cy]
cy = int(M['m01']/M['m00'])
#-storm center in geographic
clon=coord_mo[3]+(cx*res_mo)
clat=coord_mo[0]-(cy*res_mo)
#   print "cx, cy: ", cx,"-",cy,"coord in geo lon,lat: ",clon,"-",clat

#-plot fitting ellipse over possible cloud area
ellipse = cv2.fitEllipse(cnt)
cv2.ellipse(moc_rad,ellipse,(150,20,255),2) #-just show

#-count number of detected convective cloud
ct +=1

#   plotcv('Original 1PPI radar vs detected possible flare bounded by box and ellipse',
#   np.hstack([moc_rad, moc_rad])) #-plot to check detected cv

#-create mask to extract convective area
roi = np.array([cnt], dtype=np.int32)
cv2.fillPoly(mask, roi, 255) #-fill each individual cv to find area
#   plot('mask',mask)
cv2.fillPoly(cv_poly, roi, 255) #-fill cv into file

#-calculate geometry of ellipse
cv_area=mask[mask==255].size*res

```

```
# print 'Area of convective cloud no.',ct,' ', cv_area, ' sq.km.' #-unit in sq.km.
# print '--major axis length: ',ellipse_h, ' km.' #-unit in sq.km.

#https://www.quora.com/How-can-I-write-text-in-the-first-line-of-an-existing-file-using-Python
#-write cv_area to file for further analysis of temporal analysis
src= open(outpath+'infos_stormproperties.txt','r')
#We read the existing text from file in READ mode
fline= filen[4:16]+' '+str(cv_area)+' '+str(ct)+' '+str(ellipse_h)+' '+str(clon)+' '+str(clat)+'\n' #-
problem here
write_infos_text('infos_stormproperties.txt',rddir,fline)

#-write convective area to image file
# plotcv('convective area',cv)
cv2.imwrite(outpath+cv_file_name,cv) #-convective regions from original extraction
cv[cv==0]=1
cv[cv==255]=0
st=st*cv
st[st==1]=255
#-write stratiform image file
cv2.imwrite(outpath+st_file_name,st) #-startiform regions from original extraction
# plotcv('startiform area',st)
total_area_cv=cv[cv==0].size*sqkm
total_area_st=st[st==255].size*sqkm
print "-->covective and statiform areas: ", str(total_area_cv),',',str(total_area_st)

#-write cv/st to information file for further analysis of temporal analysis
# src= open(outpath+'infos_stormcv_st_area.txt','r')
#We read the existing text from file in READ mode
fline= file_name[4:16]+' '+str(total_area_cv)+' '+str(total_area_st)+'\n'
write_infos_text('infos_stormcv_st_area.txt',rddir,fline)

#-----
def write_infos_text(fileInfos,rddir,fline):
#-to write extracted infos to file
src= open(rddir+fileInfos,'r')
```

```

#We read the existing text from file in READ mode
# fline= str(filer[:10])+','+str(num_rad)+' '+ ' 0'+'\n' #-0 mean not found radar
oline=src.readlines()

#Here, we prepend the string we want to on first line
# print fline
oline.insert(0,fline)
src.close()

#We again open the file in WRITE mode
src=open(rddir+fileInfos,'w')
src.writelines(oline)
src.close()

#-----
def convert_mosaicGrey2rgb(mocg,noRowMo,noColMo):
    #-convert grayscale to RGB for method of maximum value
    print 'convert grayscale to RGB for method of maximum value'
    r,c,b=np.int(noRowMo),np.int(noColMo),3 #-r,c from mosaic dimension get from gis calculation
    mo_col_max=np.zeros((r,c,b),dtype = "uint8")#-create array output of color mosaic

    for i in range(r):
        for j in range(c):
            if mocg[i,j]==0:#-black
                mo_col_max[i,j,0]=160
                mo_col_max[i,j,1]=160
                mo_col_max[i,j,2]=160

            # #-assign index of over unused radar pixels
            # mo_index[i,j,0]=160
            # mo_index[i,j,1]=160
            # mo_index[i,j,2]=160

            if mocg[i,j]==1:#-green
                mo_col_max[i,j,0]=0
                mo_col_max[i,j,1]=255
                mo_col_max[i,j,2]=0

```



```

if mocg[i,j]==2:#-yellow
    mo_col_max[i,j,0]=0
    mo_col_max[i,j,1]=255
    mo_col_max[i,j,2]=255

if mocg[i,j]==3:#-orange
    mo_col_max[i,j,0]=0
    mo_col_max[i,j,1]=128
    mo_col_max[i,j,2]=255

if mocg[i,j]==4:#-red
    mo_col_max[i,j,0]=0
    mo_col_max[i,j,1]=0
    mo_col_max[i,j,2]=255

if mocg[i,j]==5:#-purple
    mo_col_max[i,j,0]=164
    mo_col_max[i,j,1]=73
    mo_col_max[i,j,2]=163

return mo_col_max
#-----
#-----

mainStormFd='D:/Yang/1Reseach/0.2561.MultipleRadarsMosAnalysis/1Data/1storm2561/'
lst_folder_main=[x[1] for x in os.walk(mainStormFd)]
lst_folder_main=lst_folder_main[0] #-get sub folder only

#-to collect filename of radar in processing list
#dir_files='D:/Yang/1Reseach/0.2561.MultipleRadarsMosAnalysis/1Data/1storm2561/99Testloop/1SontihnTest'
#rddir='D:/Yang/1Reseach/0.2561.MultipleRadarsMosAnalysis/1Data/1storm2561/9SontihnTest/'
#rddir='D:/Yang/1Reseach/0.2561.MultipleRadarsMosAnalysis/1Data/1radar_trial/1test_prog/' #-test
3 time
#rddir='D:/Yang/1Reseach/0.2561.MultipleRadarsMosAnalysis/1Data/1radar_trial/1test_prog/' #-test
single time

```

```

#-loop main folder of all storm folders here!
for substormfd in lst_folder_main:
    print "+"
    print "+"
    print "+"
    print
    "++++"
    print "->Do all process of storm in: ",substormfd,"-----"
    print
    "++++"
    #-----
    #ic=0 #-counter to call folder name
    #-convert gif to jpg for PHK and SVP
    print "---processing convert gif to jpg"
    fd_gif=["3phk","5svp"] #-2 folders contained gif
    inpgif_dir=path(mainStormFd+substormfd+"/"+fd_gif[0]+"/")
    convertGif2Jpg(inpgif_dir,str(fd_gif[0]))
    inpgif_dir=path(mainStormFd+substormfd+"/"+fd_gif[1]+"/")
    convertGif2Jpg(inpgif_dir,str(fd_gif[1]))
    #-----
    #create list file used for main loop
    print "---create list file used"

#mydir='D:/Yang/1Reseach/0.2561.MultipleRadarsMosAnalysis/1Data/1radar_trial/1soncaTry/3phk/'

mydir=mainStormFd+substormfd+"/"+fd_gif[0]+"/"
rddir=mainStormFd+substormfd+"/"
outtext= open(rddir+'infos_radFileNames.txt','w')#-create file
outtext.close #-close file
for file in os.listdir(mydir):
    if file.endswith(".jpg"):
        print file
    
```

```
#-write cv/st to information file for further analysis of temporal analysis
fline= file[4:14]+'\\n'
write_infos_text('infos_radFileNames.txt',rddir,fline)

#-read line by line of "radFileNames.txt"
with open(rddir+'infos_radFileNames.txt') as fp:
    radfiles = fp.readlines()

#-----
#-*****main program loop to do mosaik 5 radars
#rddir='D:/Yang/1Reseach/0.2561.MultipleRadarsMosAnalysis/1Data/1radar_trial/1soncaTry/'
#-----
#-----
#---setting mosaick boundary
#-geocoordinates derived from GIS rectify
cor_cri=[22.1407994189, 17.7900994189, 102.176617698, 97.5826676985]#top, bottom, right, left
cor_imp=[20.7466347659, 16.3889847659, 101.275708473, 96.7999084731]#top, bottom, right,
left
cor_phk=[18.9508102196, 14.6001102196, 102.473979248, 97.9564792477]#top, bottom, right,
left
cor_chn=[17.3324365695, 12.9817365695, 102.425502987, 97.9497029868]#top, bottom, right,
left
cor_svp=[15.9059776576, 11.4788276576, 102.960974698, 98.5755246984]#top, bottom, right,
left
#-mosaic boundary
res=0.00695 #-in unit of geographic
#top, bottom, right, left
coord_mo=[cor_cri[0],cor_svp[1],cor_svp[2],cor_imp[3]]#top, bottom, right, left

#-mosaic boundary
noRowMo=np.ceil((cor_cri[0]-cor_svp[1])/res)
noColMo=np.ceil((cor_svp[2]-cor_imp[3])/res)

#-----
```

```

#-preparation file for function "extract_storm"
outtext= open(rddir+'infos_stormproperties.txt','w')
outtext_area= open(rddir+'infos_stormcv_st_area.txt','w')
outtext_flare= open(rddir+'infos_flare_infos.txt','w')
outtext_radBrokeUsed= open(rddir+'infos_usedRad_broken_infos.txt','w')
#-----
lst_folder=[x[1] for x in os.walk(rddir)]
lst_folder=lst_folder[0] #-get sub folder only
for filer in radfiles: #-loop list of radar file names
    radName=int(filer)
    print "-----Mosaicking radar time GMT: ", radName,"-----"
    ct = 0 #-counter number of radar file found
    new_mo=0 #-index to check when starting new mosiack
    #-loop radar folder to find same radar name

#-----
#-setting for mosaick reflectivity and flare
mocg=np.zeros((noRowMo,noColMo))#-for grey reflectivity
mocc=np.zeros((noRowMo,noColMo,3),dtype = 'uint8')#-for color mosaic reflectivity
mocg_index=np.zeros((noRowMo,noColMo))#-for grey reflectivity

mocfg=np.zeros((noRowMo,noColMo))#-for grey flare
mocg_joker=np.zeros((noRowMo,noColMo))#-for grey flare
#-----
for subf in lst_folder:
    print "+++++-----",subf,"-----+++++"
    #-loop inside sub folder
    nf = 0 #- indicator not found 0=not found; 1=found
    for filen in os.listdir(rddir+subf+*"):
        if filen.endswith(".jpg") and int(filen[4:14])==radName:
            print subf,">>>>",filen
#         nf = 1 #- indicator found
            #-read radar each file
            img = cv2.imread(rddir+subf+*"+filen)

```

```
# plotcv("radar image "+filen,img)

#-cut img
#-resize cut img to be same
#-do cut and resize and parameter setting for mosaic
if subf=="1cri":
    num_rad=1 #-to use in mosaic
    cor_rad=cor_cri
    img=img[28:757, 72:802] #-thaiwater.net dimension
    img=cv2.resize(img, (800, 800))
    nf=1
# num_rad=5 #-to use in mosaic
if subf=="2lmp":
    num_rad=2
    cor_rad=cor_lmp
    img=img[22:764, 72:802]
    img=cv2.resize(img, (800, 800))
    num_rad=2 #-to use in mosaic
    nf=1
if subf=="3phk": #-to use in mosaic
    num_rad=3
    cor_rad=cor_phk
    img=img[28:757, 72:802]
    img=cv2.resize(img, (800, 800))
    nf=1
if subf=="4chn": #-to use in mosaic
    num_rad=4
    cor_rad=cor_chn
    img=img[28:757, 72:802]
    img=cv2.resize(img, (800, 800))
    nf=1
if subf=="5svp": #-to use in mosaic
    num_rad=5
    cor_rad=cor_svp
    img=img[0:511, 0:511]
```

```

img=cv2.resize(img, (800, 800))

nf=1

if nf==0: break #-break when not found

# plotcv("radar image "+filen,img)
#-call flare cleaning function [extract reflectivity inside]
flare_offset,final_rf,flare_ct,flareArea,flare_detected,nf=main_cleaning_program(img,subf)

#-write flare information to file for further analysis of temporal analysis
src= open(rddir+'infos_flare_infos.txt','r')
#We read the existing text from file in READ mode
fline= subf[0]+' '+filen[4:14]+' '+str(flare_ct)+' '+str(flareArea)+'\n'
write_infos_text('infos_flare_infos.txt',rddir,fline)

#-cleaning red dot for svp station
if subf=="5svp":
    mask_img = np.ones((800,800), dtype = "uint8")
    cr,cc=[np.round(img.shape[0]/2),np.round(img.shape[1]/2)]
    radius=35 #-radius of circle from center of station in unit of pixel
    circle=cv2.circle(mask_img,(cr,cc), radius, 0, -1)
    final_rf=final_rf*circle #-apply masking circle of suvannabhoom reflectivity

#-[row,col] for clipped radar images
img_shp=[img.shape[0], img.shape[1]]

#-call to do mosaic function reflectivity and flares
if num_rad==1 and nf==1:# and new_mo==0:
    mo_radg,mo_index= mosaicRad(final_rf,cor_rad, img_shp, coord_mo, res, mocg,
subf,mocg_index,num_rad)#-mosaick reflectivity
    mo_Fradg,mocg_joker= mosaicRad(flare_detected,cor_rad, img_shp, coord_mo,res,
mocfg, subf,mocg_joker,num_rad)#-mosaick flare
    if num_rad>1 and nf==1:
        mo_radg,mo_index= mosaicRad(final_rf,cor_rad, img_shp, coord_mo, res, mo_radg,
subf, mocg_index,num_rad)#-mosaick reflectivity

```

```
mo_Flradg,mocg_joker= mosaicRad(flare_detected,cor_rad, img_shp, coord_mo,res,  
mo_Flradg, subf,mocg_joker,num_rad)#-mosaick flare
```

```
#-----counter
```

```
if nf==1:
```

```
    ct +=1
```

```
    break #-use when found case
```

```
#     nf = 1 #- indicator found
```

```
#     new_mo +=1
```

```
#
```

```
#-check if idicator tell not found means unavailable file at that time
```

```
#-then we create zeros array to inject to mosaic function
```

```
if nf==0:
```

```
    print "---not found file--->" + filer
```

```
    #-write not found radars information to file for further analysis of temporal analysis
```

```
    src= open(rddir+'infos_usedRad_broken_infos.txt','r')
```

```
    #-We read the existing text from file in READ mode
```

```
    fline= str(filer[:10])+','+subf[0]+' '+ '00'+'\n' #-00 mean not found radar
```

```
    write_infos_text('infos_usedRad_broken_infos.txt',rddir,fline)
```

```
if nf==9:
```

```
    print "---broken file by exceeding flare threshold--->" + filer
```

```
    #-write not found radars information to file for further analysis of temporal analysis
```

```
    src= open(rddir+'infos_usedRad_broken_infos.txt','r')
```

```
    #-We read the existing text from file in READ mode
```

```
    fline= str(filer[:10])+','+subf[0]+' '+ '99'+'\n' #-99 mean flare exceeding threshold as broken
```

```
radar
```

```
    write_infos_text('infos_usedRad_broken_infos.txt',rddir,fline)
```

```
#-write used radar information to file for further analysis of temporal analysis
```

```
src= open(rddir+'infos_usedRad_broken_infos.txt','r')
```

```
#We read the existing text from file in READ mode
```

```
fline= filen[4:14]+' '+str(ct)+' '+ '11'+'\n' #-ending 11 means used radar code
```

```
write_infos_text('infos_usedRad_broken_infos.txt',rddir,fline)
```

```

print "--number of rad found at GMT time of ",radName,">> ",ct
print '-----'

#-----
#-save "mo_radg" to binary file
#-convert mosaicked radar reflectivity to color for plotting
moc_rad=np.zeros((noRowMo,noColMo,3),dtype = "uint8")#-create array output of color mosaic
moc_rad=convert_mosaicGrey2rgb(mo_radg,noRowMo,noColMo)
cv2.imwrite(rddir+"moc_rad_"+filen[4:16]+".tif",moc_rad) #-reflectivity mosaick in color
cv2.imwrite(rddir+"mo_radg_"+filen[4:16]+".tif",mo_radg) #-reflectivity mosaick in grey
cv2.imwrite(rddir+"moc_flareg_"+filen[4:16]+".tif",mocfg) #-reflectivity mosaick in grey

# plotcv("",mo_radg)
# plotcv('radar mosaic 5 stations after cleaning--'+filen,moc_rad)

#-----
#-extracting cv from "moc_rad"
#-parameter setting function "extract_storm"
print "--extracting cv"
inpath=rddir.
outpath=inpath
img= cv2.imread(rddir+"mo_radg_"+filen[4:16]+".tif",1)#-color
img= cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
st=cv2.imread(rddir+"mo_radg_"+filen[4:16]+".tif",1) #-color
st= cv2.cvtColor(st, cv2.COLOR_BGR2GRAY) #-for startiform cloud

cv_file_name="cv_moc_"+filen[4:16]+".tif"
st_file_name="st_moc_"+filen[4:16]+".tif"
#-cal function to extract storm characteristics

extract_storm(inpath,coord_mo,res,img,moc_rad,st,outpath,outtext,outtext_area,cv_file_name,st_file_
name,filen)#-work

#-----
print '-----'

```



```
# break #-just test prog to break loop "filer"  
# print filer  
#-----  
#-convert mosaicked radars to color for plotting  
#r,c,b=993,731,3 #-r,c from mosaic dimension get from gis calculation  
#result_c=np.zeros((noRowMo,noColMo,3),dtype = "uint8")#-create array output of color mosaic  
#result_c=convert_mosaicGrey2rgb(mo_radg,noRowMo,noColMo)  
#plotcv('radar mosaic 5 stations after cleaning',result_c)
```



2. ทบทวนคุณภาพของการโมเสค

```
# -*- coding: utf-8 -*-
```

```
'''
```

Created on Thu Oct 07 12:31:17 2018

This program is to

-mosaic radars

-detect radar flares

-combine the extracted radar flares with QI from "20181001_qualityIndexMosaic.py"

input: combination thress files

1.D:\Yang\1Reseach\0.2561.ACRES_ConvectiveExtraction\20180629_AutoIRemoveFlareRois.py

2.D:\Yang\1Reseach\0.2561.conf.GMSARN.radarMosaic\0codeForMosaic\20181001_qualityIndexMosaic.py

3.D:\Yang\1Reseach\0.2561.conf.GMSARN.radarMosaic\0codeForMosaic\20180918_tmpSimpleMosaicChnPhl1.py

output: radar flare+QI to indicate a quality of mosaic radar

Next: we will apply algorithm to extract convective clouds

@author: Nattapon Mahavik, Naresuan University, Thailand

```
'''
```

```
import numpy as np
```

```
import cv2
```

```
#from matplotlib import pyplot as plt
```

```
#-----
```

```
def plot(label,image):
```

```
    #to plot image in opencv
```

```
    cv2.namedWindow(label, cv2.WINDOW_NORMAL) #-resize manually
```

```
    cv2.imshow(label,image)
```

```
    cv2.waitKey(0)
```

```
    cv2.destroyAllWindows()
```

```
#-----
```

```
#-----
```

```
#-Radar mosaic
```

```
#-read phitsanulok image
```

```

def mosaic_rads(pimg,cimg,coord_phl,coord_chn,rres,cres,r,c):
    print 'mosaicking two radars...'
    #-----
    # to extract reflectivity and combined its for phitsanulok
    #-perfect color shades
    boundaries = [
        ([0, 153, 0], [102, 255, 102]), #green
        ([0, 204, 204], [102, 255, 255]),#yellow
        ([0, 120, 230], [102, 178, 255]),#orange
        ([0, 0, 102], [51, 51, 255]),#red
    #   ([86, 31, 4], [220, 88, 50]),#blue
    #   ([25, 146, 190], [62, 174, 250]),
    #   ([103, 86, 65], [145, 133, 128])
    ]

    #-create combined img of reflectivity
    r,c,b=pimg.shape
    rfp1=np.zeros((r,c),dtype = "uint8")#-gray for phitsanulok
    rfc=np.zeros((pimg.shape),dtype = "uint8")#-color

    i=0
    # loop over the boundaries
    for (lower, upper) in boundaries:
        i +=1
        # create NumPy arrays from the boundaries
        lower = np.array(lower, dtype = "uint8")
        upper = np.array(upper, dtype = "uint8")
        # find the colors within the specified boundaries and apply
        # the mask
        mask = cv2.inRange(pimg, lower, upper)
        output = cv2.bitwise_and(pimg, pimg, mask = mask)
        #-store detected color reflectivity to output in color
        rfc += output

        #-store detected color reflectivity to output in gray

```

```

gray=cv2.cvtColor( output, cv2.COLOR_RGB2GRAY ) #-convert rgb to gray
gray[gray>0]=i
rfpl += gray

# plot('org vs detected color',np.hstack([pimg, output]))
print 'finishing extraction of radar reflectivity'
print '-----'
#-----

#-create array of mosaic
r,c,b=993,731,3 #-r,c from mosaic dimension get from gis calculation
moc=np.zeros((r,c,b),dtype = "uint8")#-create array output of color mosaic
#mocg=np.zeros((r,c,1),dtype = "uint8")#-create array output of grey mosaic
mocg=np.zeros((r,c))
rr,cr,br=pimg.shape #-row, col for original image
rres,cres=0.0060,0.0062 #-row and col resolutions calculated after rectified in gis
coord_phl=[18.94491603,14.60527603,102.4735853,97.95719026]#top, bottom, right, left
coord_moc=[18.94491603,12.98396803,102.4735853,97.93910342]#top, bottom, right, left
mo_index=np.zeros((r,c,b),dtype = "uint8")#-create array output of used radar [red,green=phl,chn]
#-loop inside PHL radar dimension to transfer reflectivity to mosaic dimension
for i in range(rr):
    for j in range(cr):
        row_mo=round((coord_moc[0]-coord_phl[0])/rres)+i
        col_mo=round((coord_phl[3]-coord_moc[3])/cres)+j
        moc[row_mo,col_mo,:]=rfc[i,j,:] #-assign extracted radar to color mosaic
        mocg[row_mo,col_mo]=rfpl[i,j]#-assign extracted radar to grey mosaic

#-assign index of used radar Phitsanulok
if rfpl[i,j]>0:
    mo_index[row_mo,col_mo,0]=204
    mo_index[row_mo,col_mo,1]=0
    mo_index[row_mo,col_mo,2]=204

#-----

# to extract reflectivity and combined its for chainat

```

```

#-create combined img of reflectivity
r,c,b=cimg.shape
rfcn=np.zeros((r,c),dtype = "uint8")#-gray for chainat
rfc=np.zeros((cimg.shape),dtype = "uint8")#-color

i=0
# loop over the boundaries
for (lower, upper) in boundaries:
    i +=1
    # create NumPy arrays from the boundaries
    lower = np.array(lower, dtype = "uint8")
    upper = np.array(upper, dtype = "uint8")
    # find the colors within the specified boundaries and apply
    # the mask
    mask = cv2.inRange(cimg, lower, upper)
    output = cv2.bitwise_and(cimg, cimg, mask = mask)
    #-store detected color reflectivity to output in color
    rfc += output

    #-store detected color reflectivity to output in gray
    gray=cv2.cvtColor( output, cv2.COLOR_RGB2GRAY ) #-convert rgb to gray
    gray[gray>0]=i
    rfcn += gray

# plot('org vs detected color',np.hstack([cimg, output]))
print 'finishing extraction of radar reflectivity'
print '-----'

#-----
rr,cr,br=cimg.shape #-row, col for original image
# rres,cres=0.0060,0.0062 #-row and col resolutions calculated after rectified in gis
# coord_chn=[17.33080003,12.98396803,102.4381354,97.93910342]#top, bottom, right, left
# coord_moc=[18.94491603,12.98396803,102.4735853,97.93910342]#top, bottom, right, left

```

```

#-loop inside CHN radar dimension to transfer reflectivity to mosaic dimension
for i in range(rr):
    for j in range(cr):
        row_mo=round((coord_moc[0]-coord_chn[0])/rres)+i
        if row_mo>=moc.shape[0]:break #-just for safe

        col_mo=round((coord_chn[3]-coord_moc[3])/cres)+j
        moc[row_mo,col_mo,:]=rfcn[i,j,:] #-assign extracted radar to color mosaic

        #-assign maximum value of extracted radar to grey mosaic
        if rfcn[i,j]>mocg[row_mo,col_mo]:
            mocg[row_mo,col_mo]=rfcn[i,j]

            #-assign index of used radar Chainat
            mo_index[row_mo,col_mo,0]=255
            mo_index[row_mo,col_mo,1]=51
            mo_index[row_mo,col_mo,2]=51
    return mocg,mo_index
#-Finishing radar mosaic
#-----
#-----
#-----
def detect_flares(img,org_img): #-to detect radar flares
    print 'removing radar flares...'
    #-read mask that have been calculated
    r,c,b=img.shape
    ma=np.zeros((r,c,6),dtype = "uint8")#-gray to store mask

    ma[:,:,0] = cv2.imread(r'D:\Yang\1 Reseach\0.2561.ACRS_ConvectiveExtraction\maskfq1.jpg',0)
    ma[:,:,1] = cv2.imread(r'D:\Yang\1 Reseach\0.2561.ACRS_ConvectiveExtraction\maskfq2.jpg',0)
    ma[:,:,2] = cv2.imread(r'D:\Yang\1 Reseach\0.2561.ACRS_ConvectiveExtraction\maskfq3.jpg',0)
    ma[:,:,3] = cv2.imread(r'D:\Yang\1 Reseach\0.2561.ACRS_ConvectiveExtraction\maskfq4.jpg',0)
    ma[:,:,4] = cv2.imread(r'D:\Yang\1 Reseach\0.2561.ACRS_ConvectiveExtraction\maskfq5.jpg',0)
    ma[:,:,5] = cv2.imread(r'D:\Yang\1 Reseach\0.2561.ACRS_ConvectiveExtraction\maskfq6.jpg',0)
    #-----

```

```

# to extract reflectivity and combined its
#-perfect color shades
boundaries = [
    ([0, 0, 102], [51, 51, 255]),#red
    ([0, 204, 204], [102, 255, 255]),#yellow
    ([0, 120, 230], [102, 178, 255]),#orange
    ([0, 153, 0], [102, 255, 102]), #green
#    ([86, 31, 4], [220, 88, 50]),#blue
#    ([25, 146, 190], [62, 174, 250]),
#    ([103, 86, 65], [145, 133, 128])
]

#-create combined img of reflectivity
r,c,b=img.shape
rf=np.zeros((r,c),dtype = "uint8")#-gray
rfc=np.zeros((img.shape),dtype = "uint8")#-color

i=0
# loop over the boundaries
for (lower, upper) in boundaries:
    i +=1
    # create NumPy arrays from the boundaries
    lower = np.array(lower, dtype = "uint8")
    upper = np.array(upper, dtype = "uint8")
    # find the colors within the specified boundaries and apply
    # the mask
    mask = cv2.inRange(img, lower, upper)
    output = cv2.bitwise_and(img, img, mask = mask)
    #-store detected color reflectivity to output in color
    rfc += output

    #-store detected color reflectivity to output in gray
    gray=cv2.cvtColor( output, cv2.COLOR_RGB2GRAY ) #-convert rgb to gray
    gray[gray>0]=i
    rf += gray

```

```

# plot('org vs detected color',np.hstack([img, output]))
print 'finishing extraction of radar reflectivity'
print '-----'
#-----
#-----
#-----

#-This part is to detect and remove radar flares
#-apply morphology to combined reflectivity to find contour
#Gray Scaling
gray_images =cv2.cvtColor(rfc, cv2.COLOR_RGB2GRAY)
#threshold need to adjust
ret,thresh = cv2.threshold(gray_images,0,255,0)
#-apply close kernel 7x7
kernel = np.ones((7,7), np.uint8)
closed_images = cv2.morphologyEx(thresh, cv2.MORPH_CLOSE,kernel)
#-apply erosion, output will be used in the detection of flare
kernel = np.ones((3,3),np.uint8)
eroded_images = cv2.erode(closed_images,kernel,iterations = 1)
#plot('closed_images 7x7 vs eroded_images 7x7',np.hstack([closed_images, eroded_images])) #-
50 is just factor for plotting
#-apply findcontours
im2 , contours, hierarchy =
cv2.findContours(eroded_images,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
#-----
def auto_remove_flare(eroded_images):
#-----
#-Step 1: apply automatically removing radar flare with criteria
#-calculate area of one pixel to find criteria of filtering only biggest possible flare
area_rad=480.0*480.0 #-unit in sq.km.
npix=eroded_images.size #-num of pix
res=area_rad/npix*1.0 #-unit of 1 pix per sq km.>>0.4365 sq.km/pixel at resolution of 800*800
#-create mask of possible flare
mask = np.zeros(eroded_images.shape, dtype=np.uint8)
# min_area_rect=500 #- unit of number pixels

```



```
# max_area_rect=6000 #- unit of number pixels
max_eccen1=0.1 #-unitless
min_ellipse_h=50.0 #-unit of km
max_ellipse_h=250.0 #-unit of km

ct=0 #-to count number of detected flare
for h,cnt in enumerate(contours):
    if cnt.shape[0]<5: #-prevent number of contour point less than 5
        continue

    #-apply rotated rectangle
    rect = cv2.minAreaRect(cnt) #-need to filter area of rect that more than defined
    box=cv2.boxPoints(rect)
    box = np.int0(box)
    area_rect=rect[1][0]*rect[1][1]

    #-apply fitting ellipse to contour
    ellipse = cv2.fitEllipse(cnt)
    eccentricity=ellipse[1][0]/ellipse[1][1] #-width/height
    ellipse_h=ellipse[1][1]*res #-height of ellipse in km

    #-to find radar flare meet criteria
    if ellipse_h>min_ellipse_h and ellipse_h<max_ellipse_h and eccentricity<max_eccen1:
#500,3000,0.1 need to be turned

        print 'possible flare with area: ', area_rect*res , ' sq.km.'
        print cnt.shape[0], ';',area_rect,';',eccentricity,';', ellipse_h
        #-plot possible flare area, rotated rectangle, fitting ellipse
        cv2.drawContours(img,[cnt],0,(255,0,0),-1)

        #-plot rotated box over possible flare area
        cv2.drawContours(img,[box],0,(0,0,0),3)

        #-plot fitting ellipse over possible flare area
        ellipse = cv2.fitEllipse(cnt)
```

```

cv2.ellipse(img,ellipse,(0,0,255),2)
plot('Original 1PPI radar vs detected possible flare bounded by box and ellipse',
      np.hstack([org_img, img]))

#-create mask
roi = np.array([cnt], dtype=np.int32)
#white = (255, 255, 255)

cv2.fillPoly(mask, roi, 255) #-fill mask by desired color
# plot('mask',mask)

#-count number of detected flares
ct +=1

print 'number of detected flare: ',ct
#--apply mask and use the result as input in the next process
#-apply mask resulting from automatic flare detection
#-we add dilate step to mask sure that our mask is large enough to cover flares
kernel = np.ones((15,15),np.uint8) #-15,15 try and error to remove the flare
mask = cv2.dilate(mask,kernel,iterations = 1)
mask_inv = cv2.bitwise_not(mask) #-inverse mask
masked1st_rf = cv2.bitwise_and(rf, mask_inv)
# plot('masked1st_rf',masked1st_rf*50)

flare_detected=ct
return masked1st_rf,flare_detected

#-----
#-----

def remove_6rois(ma):
    # Step 2: apply 6 roi mask to the number
    #-loop mask 6 rois to compare the number of pixel under each roi
    ct=0 #-to count initial removing flare
    for i in range(6):
        print 'checking, removing flare over ROI: ', i+1

```

```

tmp_ma=ma[:,i]

#-we add dilate step to mask sure that our mask is large enough to cover flares
kernel = np.ones((15,15),np.uint8) #-15,15 try and erray to remove the flare
tmp_ma = cv2.dilate(tmp_ma,kernel,iterations = 1)
# plot('dilate masking: ',tmp_ma)
rfm=cv2.bitwise_and(eroded_images, tmp_ma)
# plot(",np.hstack([tmp_ma,rfm]))

#-----#
#- step 2.1 find geometry of fitting ellipse over individual mark #
#-----#
#-apply to find contours over individual mask
ret,thresh = cv2.threshold(tmp_ma,200,255,0)
# plot('thresh: ',thresh)
im2 , cont_ma, hierarchy =
cv2.findContours(thresh,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
# num_point=0 #-to find the largest polygon
max_eccen2=0.1 #-unitless we use 0.1 as same as it has been set before
min_ellipse_h=50.0 #-unit of km
max_ellipse_h=250.0 #-unit of km
for h,cnt in enumerate(cont_ma):

    if cnt.shape[0]<5: #-prevent number of contour point less than 5
        continue

    #-apply fitting ellipse to contour
    ellipse_m = cv2.fitEllipse(cnt)
    eccentricity_m=ellipse_m[1][0]/ellipse_m[1][1] #-width/height
    ellipse_mh=ellipse_m[1][1]*res #-major axis of ellipse in km

    print 'ellipse_mh,eccentricity_m: ',ellipse_mh,' ',eccentricity_m
    if ellipse_mh>min_ellipse_h and ellipse_mh<max_ellipse_h and
eccentricity_m<max_eccen2:
        print 'number of contour points for main individual roi:', cnt.shape[0]
        break

```

```

#-----#
#- step 2.2 find geometry of fitting ellipse inside over individual mark #
# compare geometry of step 1 and 2 if it is within our acceptable threshold#
#-----#
#-loop cont to get geometry of ellipse fitting to criteria based on eacg individual part
#-1.find contour of mask to find geometry of ellipse 2.find geometry of ellipse inside mask
#-3.loop compare geometry 2 to 1 if close to one we apply mask to original reflectivity
#-check fitting ellipse with major axis withing criteria less than 5.0%
#-apply to find contours to masked area
im2, conts, hierarchy = cv2.findContours(rfm,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
#   pct=50.0 #-percent threshold wheter to decide to use mark
max_eccen3=0.5 #-unitless we used 0.5 to assure we can remove much of flare

for h,cnt in enumerate(conts):

    if cnt.shape[0]<5: #-prevent number of contour point less than 5
        continue

    #-apply fitting ellipse to contour
    ellipse = cv2.fitEllipse(cnt)
    eccentricity=ellipse[1][0]/ellipse[1][1] #-width/height
    ellipse_h=ellipse[1][1]*res #-major axis of ellipse in km

    #-to find radar flare meet setting criteria
    if eccentricity< max_eccen3:
#   if ((dif_eh<pct) and (dif_ec<pct)):
        print 'inside eccentricity,ellipse_h',eccentricity,ellipse_h
#   print 'inside dif_eh,dif_ec: ',dif_eh,' ',dif_ec
#   print 'number of contour points:', cnt.shape[0]
        #-apply mask to original combined radar
        mask_inv = cv2.bitwise_not(tmp_ma) #-inverse mask
#   plot('inverse mask',mask_inv)

    if ct==0:

```

```

        #-apply individual mask to remove radar flare
        masked_rf = cv2.bitwise_and(eroded_images, mask_inv)
    else:
        masked_rf = cv2.bitwise_and(masked_rf, mask_inv)
    ct += 1
#     plot('org vs mask',np.hstack((eroded_images, masked_rf)))
    print '-----'

    return masked_rf
#-----
#-find area of rain
area_sq_rad=480.0*480.0
npix=img.shape[0]*img.shape[1]
res=area_sq_rad/npix*1.0 #-unit of 1 pix per sq km.
rad_area=(3.14*np.power(240,2))/res #-unit in number of pixels

#-find rain area
#-if rain over threshold we use only automatic, otherwise we will use both auto+rois
area_rain=eroded_images[eroded_images>0].size #-in pixels
pc_rain=area_rain/rad_area*100.0
print '-----'
print 'rain coverage : ', pc_rain
print '-----'
#-----
#-apply method to remove radar flare depend on percent of rain area
#-here we have three cases
min_pc_rain=15.0
if pc_rain>min_pc_rain: #-case I
    print '----apply the automatic remove only----'
    final_rf,flare_detected=auto_remove_flare(eroded_images) #-final result
else:
    print '----apply the automatic remove+6ROIs----' #case II
    masked1st_rf=auto_remove_flare(eroded_images)
    masked_rf=remove_6rois(ma)
    final_rf = cv2.bitwise_and(masked1st_rf , masked_rf) #-final result

```

```

#-here is the polish process when automatic algorithm fail
#-we will apply only western mask
#-read mask 40%
ma40= cv2.imread('D:\Yang\1 Reseach\0.2561.ACRS_ConvectiveExtraction\mask_western.jpg')
#plot('mask 40 percent',ma40)
if pc_rain>min_pc_rain and flare_detected==0: #-case III
    print '----apply the western mark when automatic detection fail----'
    gray_images =cv2.cvtColor(ma40, cv2.COLOR_RGB2GRAY)
    ret,thresh = cv2.threshold(gray_images,200,255,0)
    kernel = np.ones((15,15),np.uint8) #-15,15 try and error to remove the flare
    thresh = cv2.dilate(thresh,kernel,iterations = 1)
    ma40_inv = cv2.bitwise_not(thresh) #-inverse mask
    final_rf = cv2.bitwise_and(rf, ma40_inv)

#-----
# plot('org vs after applying masked_rf',np.hstack([rf*50,final_rf*50]))
print 'finishing detecting and removing radar flares part.'
print '-----'
#-----

#-find radar flares
final_rf[(rf>0) & (final_rf==0)]=255 #-for detecting radar flare results
final_rf[final_rf<10]=0 #-10 is set to remove reflectivity data
# plot('org vs after applying masked_rf',np.hstack([rf*50,final_rf]))

return final_rf

#-finishing radar flares
#-----
#-----

def convert_grey2color(rs_sm,rs_c,r,c):
    #-convert grayscale to RGB
    print 'convert grayscale to RGB ...'

    for i in range(r):

```

```

for j in range(c):
#   print result[i,j]
    if rs_sm[i,j]==0: #-black
        rs_c[i,j,0]=160
        rs_c[i,j,1]=160
        rs_c[i,j,2]=160

    if rs_sm[i,j]>0 and rs_sm[i,j]<20 : #-strong red
        rs_c[i,j,0]=0
        rs_c[i,j,1]=0
        rs_c[i,j,2]=153

    if rs_sm[i,j]>20 and rs_sm[i,j]<40: #-red
        rs_c[i,j,0]=0
        rs_c[i,j,1]=0
        rs_c[i,j,2]=255

    if rs_sm[i,j]>40 and rs_sm[i,j]<60: #-red
        rs_c[i,j,0]=51
        rs_c[i,j,1]=51
        rs_c[i,j,2]=255

    if rs_sm[i,j]>60 and rs_sm[i,j]<80: #-red
        rs_c[i,j,0]=102
        rs_c[i,j,1]=102
        rs_c[i,j,2]=255

    if rs_sm[i,j]>80: #-light red
        rs_c[i,j,0]=153
        rs_c[i,j,1]=153
        rs_c[i,j,2]=255

return rs_c

#-----
#-----

```

```

def convert_mosaicGrey2rgb(mocg):
    #-convert grayscale to RGB for method of maximum value
    print 'convert grayscale to RGB for method of maximum value'
    r,c,b=993,731,3 #-r,c from mosaic dimension get from gis calculation
    mo_col_max=np.zeros((r,c,b),dtype = "uint8")#-create array output of color mosaic
    for i in range(r):
        for j in range(c):
            if mocg[i,j]==0:#-black
                mo_col_max[i,j,0]=160
                mo_col_max[i,j,1]=160
                mo_col_max[i,j,2]=160

            #-assign index of over unused radar pixels
            mo_index[i,j,0]=160
            mo_index[i,j,1]=160
            mo_index[i,j,2]=160

            if mocg[i,j]==1:#-green
                mo_col_max[i,j,0]=0
                mo_col_max[i,j,1]=255
                mo_col_max[i,j,2]=0

            if mocg[i,j]==2:#-yellow
                mo_col_max[i,j,0]=0
                mo_col_max[i,j,1]=255
                mo_col_max[i,j,2]=255

            if mocg[i,j]==3:#-orange
                mo_col_max[i,j,0]=0
                mo_col_max[i,j,1]=128
                mo_col_max[i,j,2]=255

            if mocg[i,j]==4:#-red
                mo_col_max[i,j,0]=0
                mo_col_max[i,j,1]=0

```



```

mo_col_max[i,j,2]=255
return mo_col_max
#-----
#-----
# find radar flare Phitsanulok
#-read image
img =
cv2.imread(r'D:\Yang\1Reseach\0.2561.ACRS_ConvectiveExtraction\src_img\SonCa\phk\PHK_20170
7281325.jpg',1)#-more rain not work well
img=img[30:755, 72:800] #-thaiwater.net dimension
org_img =
cv2.imread(r'D:\Yang\1Reseach\0.2561.ACRS_ConvectiveExtraction\src_img\SonCa\phk\PHK_20170
7281325.jpg',1)#-more rain not work well
org_img=org_img[30:755, 72:800] #-thaiwater.net dimension
#plot('cut radar of PHK_201706011825',img)
flare_phs=detect_flares(img,org_img)

#-read QI results
path='D:\Yang\1Reseach\0.2561.conf.GMSARN.radarMosaic\0codeForMosaic/results_calculated/'
fn = path+'qa_bb.bin'
qa_bb=np.fromfile(fn, dtype=float) #-read
qa_bb=qa_bb.reshape(993,731)
#-convert to color for plotting
r,c,b=993,731,3 #-r,c from mosaic dimension get from gis calculation
result_c=np.zeros((r,c,b),dtype = "uint8")#-create array output of color mosaic
result_c=convert_grey2color(qa_bb,result_c,r,c)
#plot('quality index map in color',result_c)

#-----
#-combine flare_phs+QI
#-1.to create array of flare_phs into mosaic dimension
r,c,b=993,731,3 #-r,c from mosaic dimension get from gis calculation
moc=np.zeros((r,c,b),dtype = "uint8")#-create array output of color mosaic
mocg=np.zeros((r,c,1))#-create array output of grey mosaic
coord_phl=[18.94491603,14.60527603,102.4735853,97.95719026]#top, bottom, right, left

```

```

coord_moc=[18.94491603,12.98396803,102.4735853,97.93910342]#top, bottom, right, left
num_col_pic=728 #-get from original image after clipping
num_row_pic=725 #-get from original image after clipping
grd_y_res=(coord_phl[0]-coord_phl[1])/num_row_pic
grd_x_res=(coord_phl[2]-coord_phl[3])/num_col_pic
rres,cres=grd_y_res,grd_x_res #-row and col resolutions calculated after rectified in gis
#mo_index=np.zeros((r,c,b),dtype = "uint8")#-create array output of used radar [red,green=phl,chn]
#-loop inside flare_phs dimension to combine within mosaic dimension
for i in range(num_row_pic):
    for j in range(num_col_pic):
        row_mo=round((coord_moc[0]-coord_phl[0])/rres)+i
        col_mo=round((coord_phl[3]-coord_moc[3])/cres)+j
        if flare_phs[i,j]==255:
            mocg[row_mo,col_mo]=255#-assign extracted radar to grey mosaic

#-convert mosaicked flare_phs to color for plotting
r,c,b=993,731,3 #-r,c from mosaic dimension get from gis calculation
result_c=np.zeros((r,c,b),dtype = "uint8")#-create array output of color mosaic
result_c=convert_grey2color(mocg,result_c,r,c)
plot('mosaicked flare of Phitsanulok',result_c)
flareQl=qa_bb
#-2.combine Ql+Flare_phs
for i in range(r):
    for j in range(c):
        if mocg[i,j]==255:
            flareQl[i,j]=10 #-combined Ql+flare_phs *** final results

#-convert mosaicked Ql+flare_phs to color for plotting
r,c,b=993,731,3 #-r,c from mosaic dimension get from gis calculation
result_c=np.zeros((r,c,b),dtype = "uint8")#-create array output of color mosaic
result_c=convert_grey2color(qa_bb,result_c,r,c)
plot('mosaicked flare of Phitsanulok and Ql',result_c)
#-----
#-Apply mosaic radars
rres,cres=0.0060,0.0062 #-row and col resolutions calculated after rectified in gis

```

```

coord_phi=[18.94491603,14.60527603,102.4735853,97.95719026]#top, bottom, right, left
coord_chn=[17.33080003,12.98396803,102.4381354,97.93910342]#top, bottom, right, left
coord_moc=[18.94491603,12.98396803,102.4735853,97.93910342]#top, bottom, right, left
rres,cres=0.0060,0.0062 #-row and col resolutions calculated after rectified in gis
r,c,b=993,731,3 #-r,c from mosaic dimension get from gis calculation

#-input radars
#-read phitsanulok image
pimg =
cv2.imread(r'D:\Yang\1Reseach\0.2561.conf.GMSARN.radarMosaic\1z_testCodes\1data_trial\PHL\or
g.jpg\PHK_201707261125.jpg',1)
pimg=pimg[30:755, 72:800] #-thaiwater.net dimension

#-read chainat image
cimg =
cv2.imread(r'D:\Yang\1Reseach\0.2561.conf.GMSARN.radarMosaic\1z_testCodes\1data_trial\CHN\or
g.jpg\CHN_201707261130.jpg',1)
cimg=cimg[30:755, 72:800] #-thaiwater.net dimension
mocg,mo_index=mosaic_rads(pimg,cimg,coord_phi,coord_chn,rres,cres,r,c) #-apply function
moc=convert_mosaicGrey2rgb(mocg)
plot('mosaicked radar phitsanulok and chinat',moc)
#-----
#-----

#-apply QI+flare_phs to original reflectivity
#-mask
mask=mocg
mask[mask<>0]=1

#-apply mask
mo_flareQI=np.multiply(flareQI,mask)
###convert grayscale to RGB
r,c,b=993,731,3 #-r,c from mosaic dimension get from gis calculation
result_c=np.zeros((r,c,b),dtype = "uint8")#-create array output of color mosaic
result_c=convert_grey2color(mo_flareQI,result_c,r,c)
plot('quality index map of radar mosaic in color',result_c)

```

3. กระบวนการวิเคราะห์การบดบังเรดาร์บีม

```
# -*- coding: utf-8 -*-
```

```
'''
```

```
Created on Sat Oct 10 13:11:42 2017
```

```
Phitsanulok
```

```
PBB http://wradlib.org/wradlib-docs/0.9.0/notebooks/beamblockage/wradlib\_beamblock.html
```

```
Using Gtopo30 can run the program, while srtm3arc is too high resolution causing crash in numpy?
```

```
@author: ANS
```

```
'''
```

```
import wradlib as wrl
```

```
import matplotlib.pyplot as pl
```

```
import matplotlib as mpl
```

```
import warnings
```

```
warnings.filterwarnings('ignore')
```

```
try:
```

```
    get_ipython().magic("matplotlib inline")
```

```
except:
```

```
    pl.ion()
```

```
import numpy as np
```

```
#-setup phetchabun radar
```

```
sifecoords = (100.217977, 16.775380, 47.0) #-alt from gtopo30
```

```
nrays = 360 # number of rays
```

```
nbins = 480 # number of range bins
```

```
#Available elevations: [0.0, 1.3, 2.9, 4.9, 7.3, 10.2, 13.8, 18.2, 23.5, 30.0] for PHB
```

```
el = 0.50 # vertical antenna pointing angle (deg)
```

```
bw = 1.0 # half power beam width (deg)
```

```
range_res = 500. # range resolution (meters)
```

```
 #-Create the range, azimuth, and beamradius arrays.
```

```
r = np.arange(nbins) * range_res
```

```
beamradius = wrl.util.half_power_radius(r, bw)
```

```
 #-calculate the spherical coordinates of the bin centroids and their longitude, latitude and altitude.
```

```
coord = wrf.georef.sweep_centroids(nrays, range_res, nbins, el)
lon, lat, alt = wrf.georef.polar2lonlatalt_n(
    coord[..., 0], np.degrees(coord[..., 1]),
    coord[..., 2], sitecoords)
polcoords = np.dstack((lon, lat))
print("lon,lat,alt:", lon.shape, lat.shape, alt.shape)

rlimits = (lon.min(), lat.min(), lon.max(), lat.max())
print("radar bounding box:", rlimits)

#Preprocessing the digital elevation model
#rasterfile =
wrf.util.get_wradlib_data_file("D:/Yang/1 Geoinformatic_data/DEM/SRTM_3arc_v4/srtm_57_09.tif") #-
too high resolution
#rasterfile =
wrf.util.get_wradlib_data_file("D:/Yang/1 Geoinformatic_data/DEM/GTOPO30/gtopo30_clip.tif")
rasterfile =
wrf.util.get_wradlib_data_file("D:/Yang/1 Reseach/0.2561.conf.GMSARN.radarMosaic/1GIS_works/DE
M_resampling_srtm16b_ph1.tif")
rastercoords, rastervalues = wrf.io.read_raster_data(rasterfile)

# Clip the region inside our bounding box
ind = wrf.util.find_bbox_indices(rastercoords, rlimits)
rastercoords = rastercoords[ind[1]:ind[3], ind[0]:ind[2], ...]
rastervalues = rastervalues[ind[1]:ind[3], ind[0]:ind[2]]

# Map rastervalues to polar grid points
polarvalues = wrf.ipol.cart2irregular_spline(rastercoords, rastervalues,
    polcoords, order=3,
    prefilter=False)

print(polarvalues.shape)

#Calculate Beam-Blockage
PBB = wrf.qual.beam_block_frac(polarvalues, alt, beamradius)
```

```

PBB = np.ma.masked_invalid(PBB)
print(PBB.shape)

# calculate cumulative beam blockage CBB
ind = np.nanargmax(PBB, axis=1)
CBB = np.copy(PBB)
for ii, index in enumerate(ind):
    CBB[ii, 0:index] = PBB[ii, 0:index]
    CBB[ii, index:] = PBB[ii, index]

#Visualize Beamblockage
# just a little helper function to style x and y axes of our maps
def annotate_map(ax, cm=None, title=""):
    ticks = (ax.get_xticks()/1000).astype(np.int)
    ax.set_xticklabels(ticks)
    ticks = (ax.get_yticks()/1000).astype(np.int)
    ax.set_yticklabels(ticks)
    ax.set_xlabel("Kilometers")
    ax.set_ylabel("Kilometers")
    if not cm is None:
        pl.colorbar(cm, ax=ax)
    if not title=="":
        ax.set_title(title)
    ax.grid()

fig = plt.figure(figsize=(10, 8))

# create subplots
ax1 = pl.subplot2grid((2, 2), (0, 0))
ax2 = pl.subplot2grid((2, 2), (0, 1))
ax3 = pl.subplot2grid((2, 2), (1, 0), colspan=2, rowspan=1)

# azimuth angle
angle = 280.0

```

```
# Plot terrain (on ax1)
ax1, dem = wrf.vis.plot_ppi(polarvalues,
                           ax=ax1, r=r,
                           az=np.degrees(coord[:,0,1]),
                           cmap=mpl.cm.terrain, vmin=0.)
ax1.plot([0,np.sin(np.radians(angle))*2.4e5],
         [0,np.cos(np.radians(angle))*2.4e5],*r-")
ax1.plot(sitecoords[0], sitecoords[1], 'ro')
annotate_map(ax1, dem, 'Terrain within {0} km range'.format(np.max(r / 1000.) + 0.1))

# Plot CBB (on ax2)
ax2, cbb = wrf.vis.plot_ppi(CBB, ax=ax2, r=r,
                           az=np.degrees(coord[:,0,1]),
                           cmap=mpl.cm.PuRd, vmin=0, vmax=1)
annotate_map(ax2, cbb, 'Beam-Blockage Fraction')

# Plot single ray terrain profile on ax3
bc, = ax3.plot(r / 1000., alt[angle, :], '-b',
              linewidth=3, label='Beam Center')
b3db, = ax3.plot(r / 1000., (alt[angle, :] + beamradius), '-b',
                linewidth=1.5, label='3 dB Beam width')
ax3.plot(r / 1000., (alt[angle, :] - beamradius), '-b')
ax3.fill_between(r / 1000., 0.,
                polarvalues[angle, :],
                color='0.75')
ax3.set_xlim(0., np.max(r / 1000.) + 0.1)
ax3.set_ylim(0., 3000)
ax3.set_xlabel('Range (km)')
ax3.set_ylabel('Altitude (m)')
ax3.grid()

axb = ax3.twinx()
bbf, = axb.plot(r / 1000., CBB[angle, :], '-k',
               label='BBF')
axb.set_ylabel('Beam-blockage fraction')
```

```
axb.set_ylim(0., 1.)
axb.set_xlim(0., np.max(r / 1000.) + 0.1)

legend = ax3.legend((bc, b3db, bbf),
                    ('Beam Center', '3 dB Beam width', 'BBF'),
                    loc='upper left', fontsize=10)

fig.savefig('D:/Yang/1Reseach/0.2561.conf.GMSARN.radarMosaic/0codeForMosaic/1results_PBB/phi
_280deg.png',dpi=300)
plt.close(fig)
```



ภาคผนวก ข
ผลงานการตีพิมพ์ในวารสารงานวิจัยระดับนานาชาติ 1 ฉบับ
และผลงานการตีพิมพ์ในรายงานสืบเนื่องการประชุมวิชาการระดับนานาชาติ 1
ฉบับ



International Journal of GEOINFORMATICS



ISSN 1686-6576

03 May, 2019

Dr. Nattapon Mahavik
Department of Natural Resources and Environment
Naresuan University
Muang, Phitsanulok, 65000, Thailand
E-mail: nattaponm@nu.ac.th

Dear Dr. Nattapon Mahavik,

I am very happy to inform you that your paper "*Spatio-Temporal Analysis of Convective Cloud Properties Deriving from Weather Radar Reflectivity during the Decaying Stage of Tropical Storm over the Lower Northern Thailand*" (Author Name: Nattapon Mahavik and Sarintip Tantanee) has been found very interesting and is accepted for publication in *International Journal of Geoinformatics*, 2019.

You shall be receiving a copy of the journal after printing.

We appreciate your excellent contribution.

With best regards,

Nitin Spathi

Prof. Nitin Kumar Tripathi
Editor-in-Chief
E-mail: nitinkt@ait.asia



Spatio-Temporal Analysis of Convective Cloud Properties Deriving from Weather Radar Reflectivity during the Decaying Stage of Tropical Storm over the Lower Northern Thailand

Mahavik, N.¹ and Tantance, S.^{2,3}

¹Department of Natural Resources and Environment, Faculty of Agriculture Natural Resources and Environment, Naresuan University, Phitsanulok, 65000, Thailand, E-mail: nattaponm@nu.ac.th

²Department of Civil Engineering, Faculty of Engineering, Naresuan University, Phitsanulok, 65000, Thailand

³Centre of Excellence on Energy Technology and Environment, Naresuan University, Phitsanulok, 65000, Thailand

Abstract

Disaster Management is one of the most important responsibilities of the governments all over the world. The systems of disaster prevention, preparedness and mitigation have been well established only in developed countries. For the developing countries including Thailand, people still have suffered from disaster since they are lacking of adequate information to cope with disaster. The weather radar is one of the tools that can provide spatio-temporal information for Nowcast which is useful for hydro-meteorological disasters warning and mitigation system. The extremes of precipitation are usually detected by convective cloud where the updrafts and downdrafts have been strengthened in vertical wind motions. These extremes are the vital threat to people during wet season over Thailand. In this study, therefore, we have developed algorithm to extract convective cloud information that is necessary for prediction of precipitation extremes. The algorithm is constructed in Python script using OpenCV library to extract the radar reflectivity of the 1st Plan Position Indicator of radar image. The reflectivity data with the measurement frequency of once per hour has been obtained from the website of Thaiwater.net of which original data provided by Thai Meteorological Department (TMD). This study has been done by using reflectivity data from Phitsanulok radar station which locates in Lower Northern Thailand. The convective cloud regions have been detected by the developed algorithm. In addition, the fitting ellipses (Fitzgibbon and Fisher, 1995) have been applied to derive the properties of convective clouds during Sonca tropical storm in July of 2017. Results show that the number of convective cloud regions has abruptly increased during Sonca period compared to before the storm passing. The average number of convective cloud region has shown clearly of peak time at 17 LST. However, several peaks of total area on the convective region at hourly scale have also been found. The spatio-temporal analysis of the extracted storm information demonstrates the severe pattern of the high frequency convective storm during Sonca event over windward slope. The number of convective storm doubled increases comparing with period prior to Sonca storm.

1. Introduction

Flash flood and landslide are mainly triggered by exceeding rainfall amount above the maximum capacity of soil resistance over the risky areas. Analysis of mesoscale precipitation areas is considerable meteorological significance because the features of subsynoptic-scale air motions including convective and stratiform regions of cloud will improve our understanding on mechanism of larger and smaller scales of the atmosphere (Austin and Houze, 1970). Mesoscale Convective Systems (MCSs) are the largest convective storms which are taken into account for a large proportion of precipitation in both the tropics and warmer

latitudes (Houze, 2004). There are many factors of MCSs that produce contiguous precipitation area of about 100 km. Many studies have investigated MCSs contribution to extreme rainfall implying disaster related to their events over the world (e.g. Rigo and Llasat 2004, Rigo and Llasat 2005, Schumacher and Johnson, 2006, Rigo and Llasat, 2007, Carbone and Tuttle, 2008 and He et al., 2016). Schumacher and Johnson (2006) had studied extreme rain events during 1999-2003 in U.S. They used National composite radar reflectivity to classify each event as the MCSs, Synoptic, or the tropical system and then to the sub-classifications based on their

organizational structures. They found 74% of warm-season events are associated with MCSs producing their peak rainfall time between 2100 and 2300 Local Standard Time (LST).

Radar reflectivity has been widely used to study characteristics of MCSs due to its advantages on spatial and temporal resolution (e.g. Rigo and Llasat, 2004, Schumacher and Johnson, 2006, Rigo and Llasat, 2007, Lang et al., 2007, He et al., 2016 and Satomura et al., 2011). The storm morphology has been detected by employing an ellipse-fitting technique of which the major and minor axes lengths are extracted from the geometry of the fitted ellipses (Lang et al., 2007). In addition, to improve convective parameterizations, the ellipse-fitting technique is also applied to radar sensor on satellite of the Tropical Rainfall Measuring Mission (TRMM) Precipitation Radar (PR) or TRMM/PR to detect the storm morphology and rainfall characteristics (Nesbitt et al., 2006). Finding the contribution of storm morphology characteristics in land and ocean will provide more understanding on the importance of rainfall modes difference regionally. Using radar sensor on satellite board, MCSs are found that they are the cause of rainfall over selected land regions up to 90%. In tropics-wide extent, MCSs are responsible for more than 50% of rainfall in almost all regions with average annual rainfall exceeding 3 mm day^{-1} .

Risky areas during tropical storm events over the Indochina have been regularly affected during the boreal summer. In the latter half of July 2017, tropical storm named Sonca coming from South China Sea had affected to northern and northeastern regions of Thailand when it was dissipating to tropical depression (Bangkok Post, 2017). Phitsanulok province located in the Lower Northern Thailand had also been affected which caused large area of paddy field inundation and the severe flashflood attacked more than 50 houses over the upper part of the province. However, there is no research about the situation of MCSs during the Sonca over the Lower Northern Thailand. Satomura et al., (2013) proposed the radar echo composite map using multiple weather radars crossing the nation boundaries over the Indochina during Typhoon Lekima in October 2007 to understand typhoon structures of tropical disturbance. Compared with the radar echo composite map, mesoscale numerical model simulation reproduced the typhoon center position very well, and it also caught the characteristics of internal structure of decaying typhoon in the Indochina. Nevertheless, the spatio-temporal analysis of decaying tropical storm is not yet well described using the high-resolution data.

Therefore, the objectives of this study are; to detect the convective cloud regions by extracting the radar data over Phitsanulok province, to spatio-temporal analyze during the storm event in July 2017, and to describe the implemented methodology of the radar image data process. In the developing countries including Thailand, official mosaicked radar products are unavailable to be used for research purpose and real-time monitoring of severe weather. Thus, this study uses free-online radar data providing as images that have been observed at Phitsanulok radar, in the middle Thailand.

2. Data and Study areas

Radar reflectivity data used in this study has been observed by Phitsanulok weather radar station. The radar site is located in the Lower Northern Thailand (Figure 1) at the geographic coordinates at latitude of $16^{\circ}46'30.358''\text{N}$ and longitude of $100^{\circ}13'4.312''\text{E}$ on the height of the terrain at 47 m above mean sea level with tower height of 30 m. The radar data is observed by Thai Meteorological Department (TMD) at four time per hour. While the data has been archived as images by Hydro and Agro Informatics Institute (HAI) at frequency of once per hour in format of GIF. The active radius of the Phitsanulok radar is 240 km observing in C-band frequency with beam width at 1° . In this study, the Plan Position Indicator (PPI) images of the first elevation at 0.5° from horizontal line have been collected and used in the analysis.

The topography of radar coverage area varies from flood plain in the central and southern part to mountainous area in the surrounding of the western, northern and eastern part. Over the radar coverage area, there are also the important river tributaries of Chao Phraya river basin which are Ping, Yom and Nan river basin. During southwesterly monsoon season, the rainfall abruptly increases over the Indochina region ranging from May to the mid of October. The amount of rainfall over this region is under the influences of the convergence of prevailing wind that brings the moisture to the low depression at the location of Intertropical Convergence Zone. In addition, the tropical disturbances from north western Pacific Ocean usually bring a large amount of rainfall in terms of tropical depression. In this study, radar data during influencing period of Sonca tropical storm during 24 to 29 July 2017 according to TMD warning announcement has been processed and analyzed.

3. Methodology

3.1 Extraction of Radar Reflectivity

The radar images provided by TMD are shown in color ranges superimposed over terrain as its

background. The color ranges indicate intensity of radar reflectivity in ascending order of rainfall intensity of green (slight rain), yellow to oranges (medium rain), and red to white (heavy rain). The images cannot be instantly processed or analyzed in digital image processing (DIP) because it is needed pre-process to transform image to digital information. Therefore, the implemented methodology of radar reflectivity extraction from radar images shown in Figure 2 has been developed

using Python script as shown the workflow in Figure 3. First, the image has been cropped to select only radar coverage area. Then, radar reflectivity shown in Figure 2a has been extracted following the pre-defined color ranges as shown in Figure 2b. The threshold method to classify convective storm using radar reflectivity has been used in many studies (e.g. Steiner et al., 1995, Rigo and Llasat 2004 and 2007).

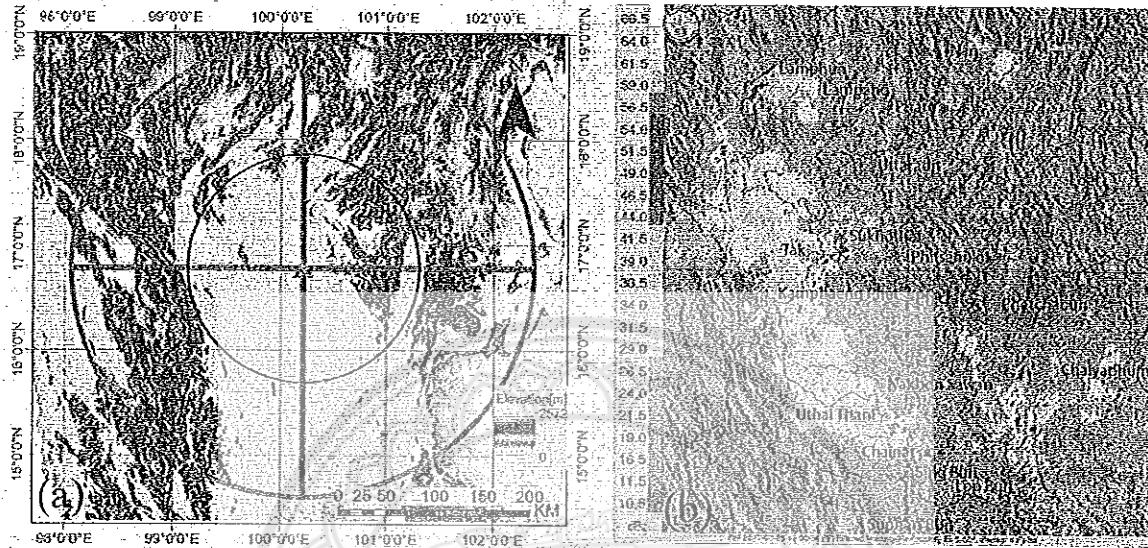


Figure 1: Study area over Phitsanulok, the middle of Thailand (a) active radius of Phitsanulok with two observation ranges at 120 and 240 km (b) radar image of 1st PPI at elevation angle of 0.5° on on 26 July 2017 at 20.25 LST provided by Thai Meteorological department

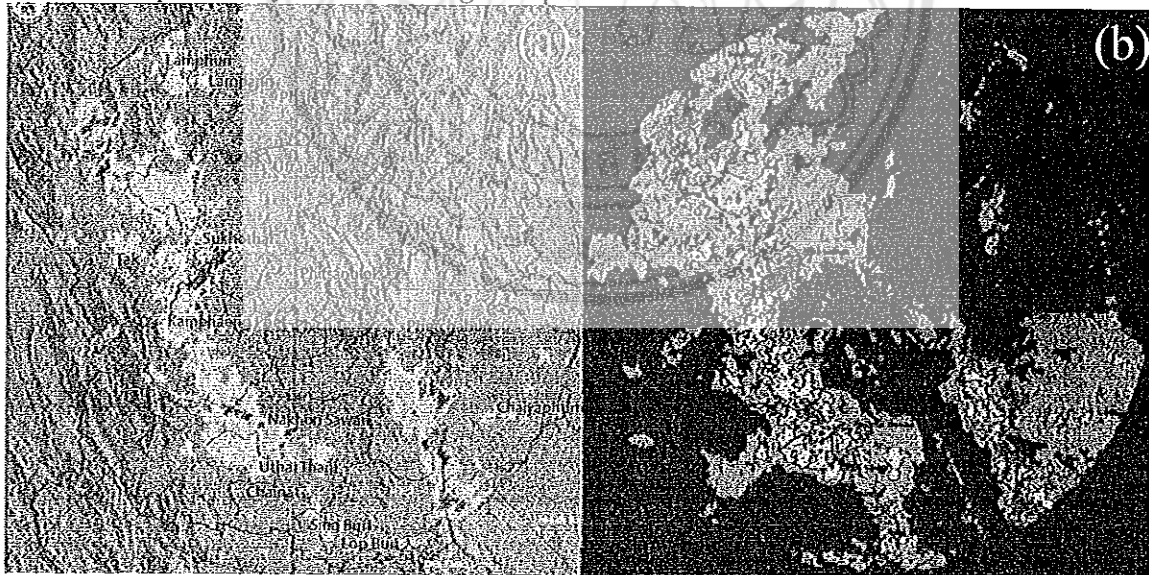


Figure 2: Radar extraction result (a) radar reflectivity with permanent flare to the west of the radar station (b) the extracted radar reflectivity on 26 July 2017 at 13.25 LST

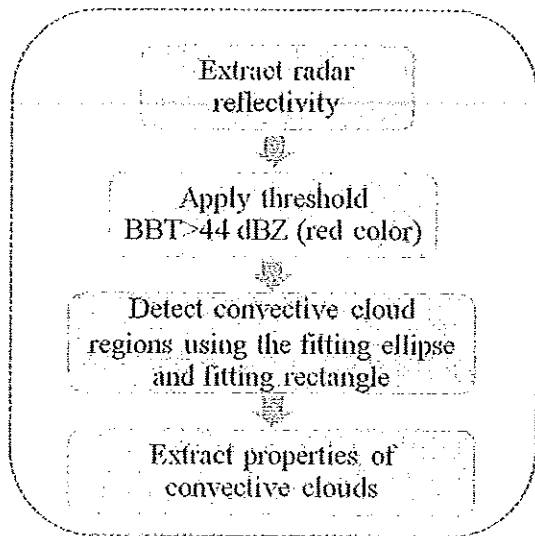


Figure 3: workflow of detection convective cloud properties

Using the raw radar reflectivity, the partition of cloud precipitation can be classified using the 2 and 3 dimensional information of radar scanning (Steiner et al., 1995 and Biggerstaff and Listemaa, 2000). However, without the raw data, the present study simply classifies the convective storm using the extracted radar information from TMD radar images by background-exceedence technique (BET) to identify the core of the convective precipitation (Austin and Houze, 1972 and Houze, 1973). The convective storm has been simply defined the threshold according to the red color ranges at the reflectivity greater than 44 dBZ. The next step is to use the obtained radar reflectivity in the process of radar flares removing.

3.2 Detection of Convective Cloud Properties

In the case of convective cloud detection, we have developed a method to detect convective cloud regions in the hourly radar images using Digital Image Processing (DIP) technique in computer vision field by employing OpenCV-Python library V.3.4.1.15. The morphological transformations have been applied to those extracted reflectivity to detect the convective region. At the beginning, the extracted reflectivity has been converted to grey scale image to be ready to apply kernel convolution of close morphology at size of 15x15 window which is derived from several trials to produce an appropriate result of the processed reflectivity. The morphological close is used to eliminate small holes inside foreground objects following with application of erosion morphology to separate each possible flare apart from the group of rain pixels. Later, edge detection using Canny operator has been firstly

applied to detect unwanted line appeared on the reflectivity and, the feature extraction technique of Hough transform (Duda and Hart, 1972), then, was applied to detect those lines from the results of canny application. In order to apply the process of convective cloud detection, the eroded image in greyscale has been searched through for delineating the contour of the pixel objects. The contoured image is used as the main input in the next process. The fitting ellipse (Fitzgibbon and Fisher, 1995) and the fitting rectangle have been applied the extracted possible regions as contour lines. The results of detected convective regions by the fitting ellipse provide convective cloud properties such as major ellipse axis, the ratio between minor and major ellipse axis lengths or AR as shown in Figure 4, as well as the center of convective cloud region from the fitted ellipse.

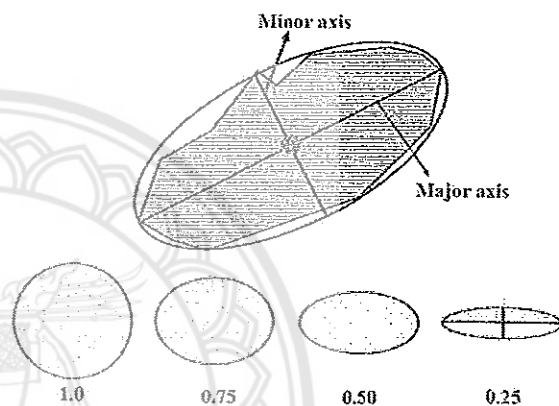


Figure 4: schematic of the fitting ellipse shown in the biggest ellipse envelopes of the convective region in blue region. The axis ratio (AR), the ratio between minor and major fitting ellipse length, is shown as a number

To avoid confusion between radar flares and convective clouds, the criteria of automate convective cloud detection has been set by using the AR exceeding 0.1, the length of major fitting ellipse axis must be in range of 5-480 km and area of the fitting rectangle is greater than 10 sq.km. MCSs are the regions of both convective and stratiform precipitation (Houze, 2004). Steiner et al., (1995) detected radar echo images with convective cloud at horizontal radius of 11 km. However, in this study, to assure the convective detection, we strictly apply both the major of the fitting ellipse and convective size compared to previous study. The results of detected convective cloud regions are shown in Figure 5. The next step is to use the results of detected convective cloud to analyze spatio-temporal variability for searching dominant spatio-temporal pattern over the Sonca storm period.

3.3 Terrain Effect on Radar Beam Observation

To discuss about the terrain effect on propagating radar beam, the simulated beam propagation is needed to reconstruct in the analysis with the terrain. To reconstruct of the propagating radar beam through the atmosphere, the spherical coordinate radar reflectivity was considered with the elevation above mean sea level (MSL) which obtained from the SRTM Digital Elevation Model (DEM) V4 with resolution of 3 arc second (Rabus et al., 2003) for each range bin at resolution of 1 km. The height of each range bin (H) was calculated using the standard refraction relation from Rinehart (1999):

$$H = \sqrt{r^2 + R'^2 + 2rR' \sin \phi} - R' + H_0$$

Equation 1

where r is the range from the radar to the point of interest, ϕ is the elevation angle of the radar beam, H_0 is the height of the radar antenna, $R' = 4/3R$, and R is the earth's radius (approximately 6374 km).

4. Results

4.1 Radar Reflectivity Extraction

The reflectivity from 1st Plan Position Indicator (PPI) of the original image is successfully extracted using the predefined color ranges as shown in Figure 2. The radar reflectivity in red color is better delineated and extracted comparing with the original image because among the color ranges the red color is easiest to be defined. The extracted radar reflectivity in the red color ranges will be used as the input for the delineated convective cloud in this study. However, the radar extraction from the simply implementation will be the new path way of radar meteorology in 2-Dimension analysis in the middle of Indochina peninsula due to lacking of raw digital radar data provided for the research community. Although, the extracted reflectivity derived from PPI may be either overshooting or undershooting the convective cloud at some distances, the PPI data is also useful to monitor the severity of storm for developing countries such Thailand. We also realized that the results of using PPI are not be consistent with using Constant altitude plan position indicator (CAPPI) from previous studies of Satomura et al., (2011) and Mahavik et al., (2014). PPI uses information from single elevation to observe the cloud which may subject to precipitating clouds in the far range. In contrast, the CAPPI uses information from several

angles during the interpolation processing in the Cartesian grid system using weighting techniques to mitigate either underestimate or overestimate precipitating cloud.

4.2 Detection of Convective Clouds

Convective clouds have been well extracted compared with the original image as shown in Figure 5. Since the beam width is 1°, the far range of radar observation has been increased their horizontal resolution larger than 5 km. Therefore, it is assured that the small convective cloud can be detected within the limitation of minimum convective area at greater than 10 km². In the future, the minimum size of convective cloud should be adjusted when the higher resolution of radar observation is in operation. Moreover, with the higher resolution of radar, the suddenly severe storm in smaller size over wider area can be detected, as well.

4.3 Terrain effect on Radar Beam Observation

The terrain effect of radar beam on cloud observation is needed to be discussed to evaluate the quality of the observation. The Wradlib library V.0.9.0 (Heistermann et al., 2013) is used in the analysis using with Python script. The terrain used in beam blockage analysis shows variation in altitude (Figure 6). The east side of Phitsanulok radar occupies rugged terrain with high elevation that obstructs the broadening beam over 50% of beam width. Using the beam-blockage fraction (BBF), the areas of blocked beam by terrain mainly locate in the east side of the radar. The blocked beam areas indicate the reliability of the returned radar power to estimate group of precipitation. The higher BBF represents the higher uncertainty of precipitation estimation over the area. As shown in Figure 6, we simply define the BBF at 50%. The cross section of propagating beam trough atmosphere by considering 1° of beam width according to equation 1 has shown the center of beam obstructed by the mountain range name Phetchabun range in the part of Phitsanulok province (Figure 6c). However, the blocked area is not large in this study and the results of the detected convective clouds are not affected. The suggestion from the beam blockage analysis is that TMD should consider using multiple elevation angles for the analysis and making a radar composite map by mosaicking nearby radar for warning the severe storm that may trigger disaster over the rugged terrain areas.

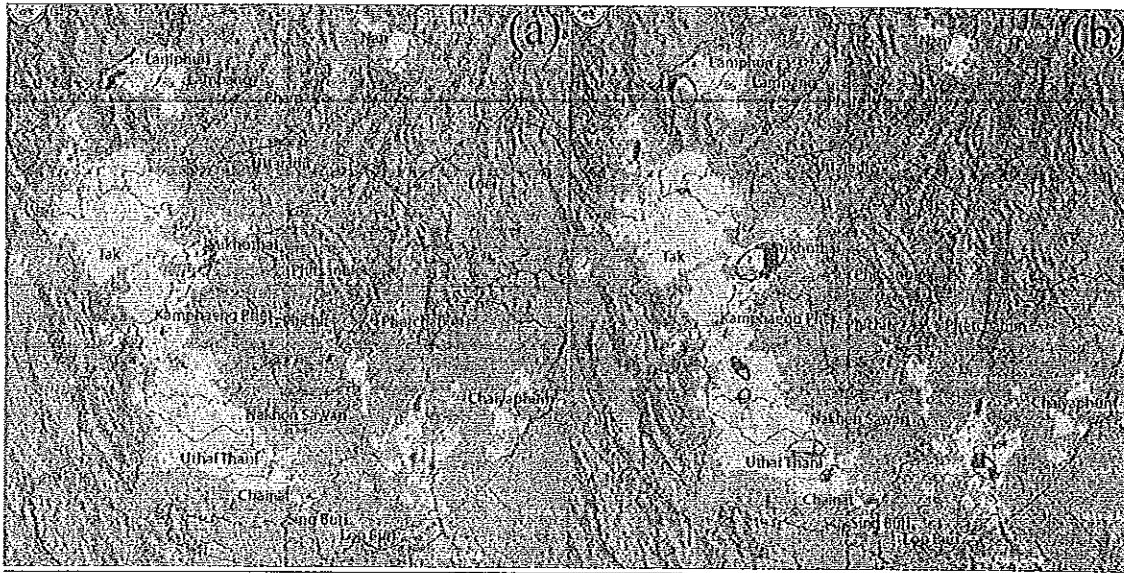


Figure 5: The detected convective storms on 26 July 2017 at 20.25 LST (a) original 1st PPI radar reflectivity (b) detected convective storms superimposed by the fitting ellipse in purple colors

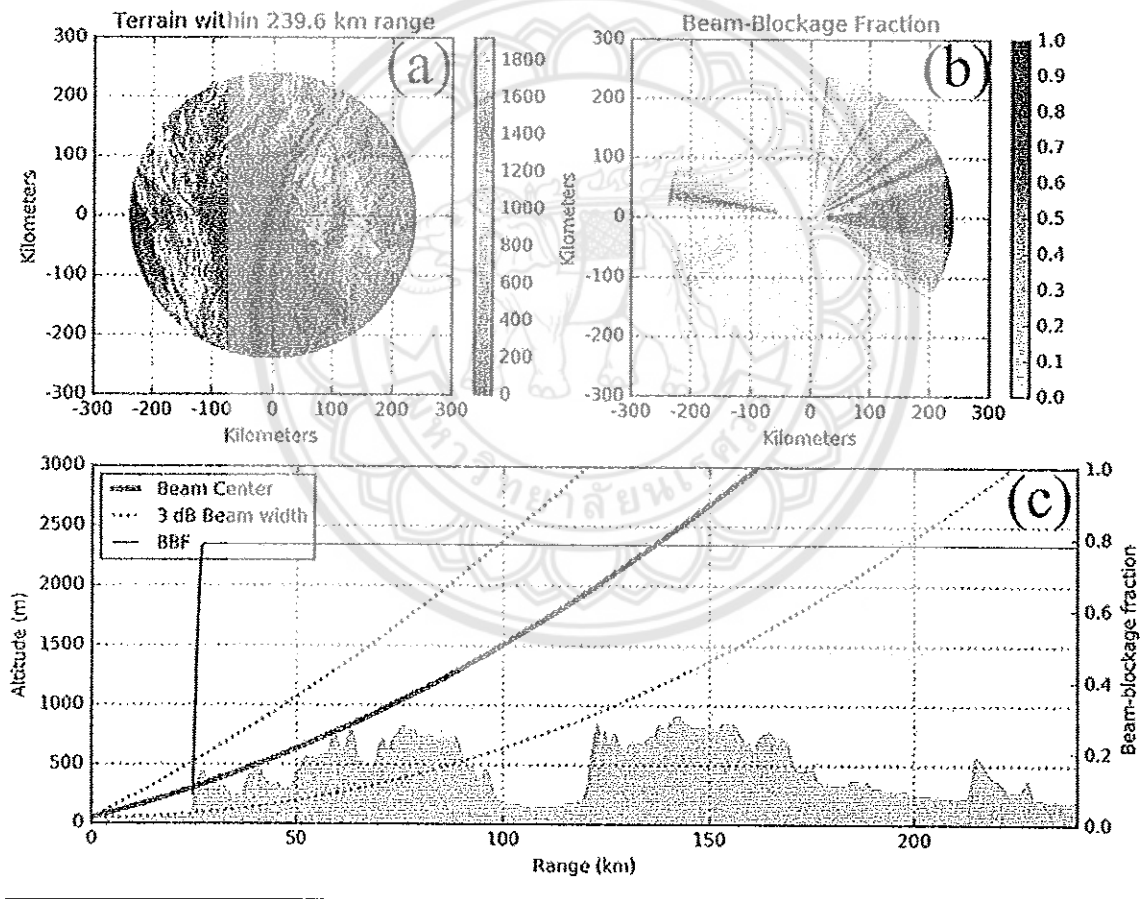


Figure 6: simulation of beam-blockage fraction using beam propagation equation implemented in Wradlib library over the terrain surrounding Phitsanulok radar using elevation angle at 0.5 deg. (a) terrain over the observed area and pointing azimuth direction shown in red line (b) beam-blockage fraction of each radar observation range (c) simulated result of propagating beam at azimuth of 90°

Table 1: summary of convective cloud statistics comparing between during Sonca and before Sonca periods

Properties	Sonca period	Before Soca period
1 Total area (km ²)	50,810.23	31,973.57
2 Average area (km ²)	102.65	91.88
3 S.D. area (km ²)	151.19	120.34
4 Total number of CV	2,651.00	1,112.00
5 Average number of CV	5.36	3.20
6 S.D. number of CV	4.35	2.26
7 Total length of CV (km)	6,754.69	4,737.52
8 Average length of CV (km)	13.65	13.61
9 S.D. length of CV (km)	12.55	11.11

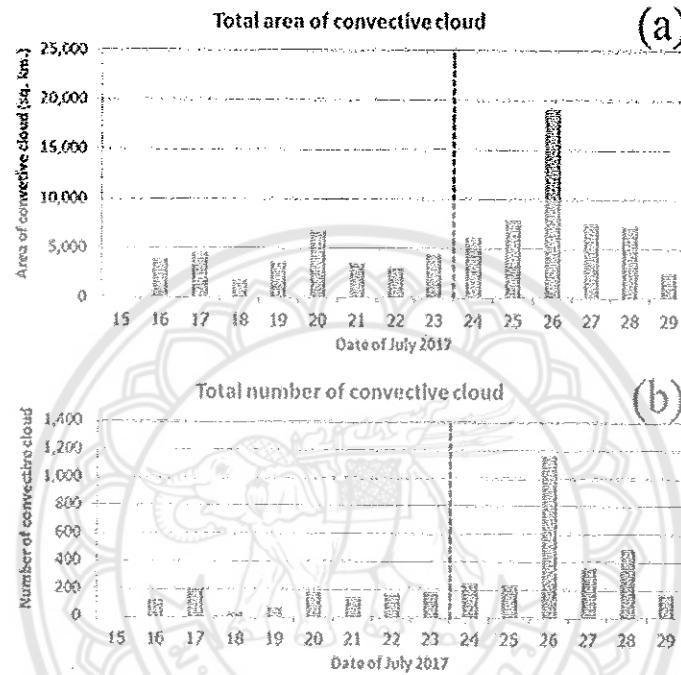


Figure 7: daily statistics of convective clouds (a) total area of convective cloud (b) total number of convective cloud. Dash lines separate during Sonca and before Sonca periods

4.4 Spatio-Temporal Analysis of Convective Cloud Occurrence

Spatio-temporal analysis of convective cloud occurrence will improve the understanding in severity impact of the Sonca over the middle Thailand. As shown in Table 1, comparison of period before and during Sonca event has shown severity in various parameters. The before Sonca period which accounted for 9 days from 15 to 23 July, has shown the small area of convective cloud coverage almost half of the coverage during Sonca period of 6 days from 24 to 29 July. The average lengths of convective cloud for both periods are not different, while the total number of convective cloud during Sonca is larger than double of before the storm period. Total area and number of convective

cloud during Sonca are greater than the before storm period especially on 26 July as shown in Figure 7a and 7b.

Average number of convective cloud during Sonca has been analyzed to find the severe hour as diurnal variation. The maximum of average number of the convective cloud at 10 UTC (17 LST) has been found as shown in Figure 8a. However, area average of diurnal variation convective peak has shown the highest peak at 23UTC as shown in Figure 8b. The quasi-stationary convective systems as well as speed and direction of convective storms during Sonca have not been considered in this study. Therefore, the characteristics of storm movement will be further investigated in the future study to classify the severity of the storm.

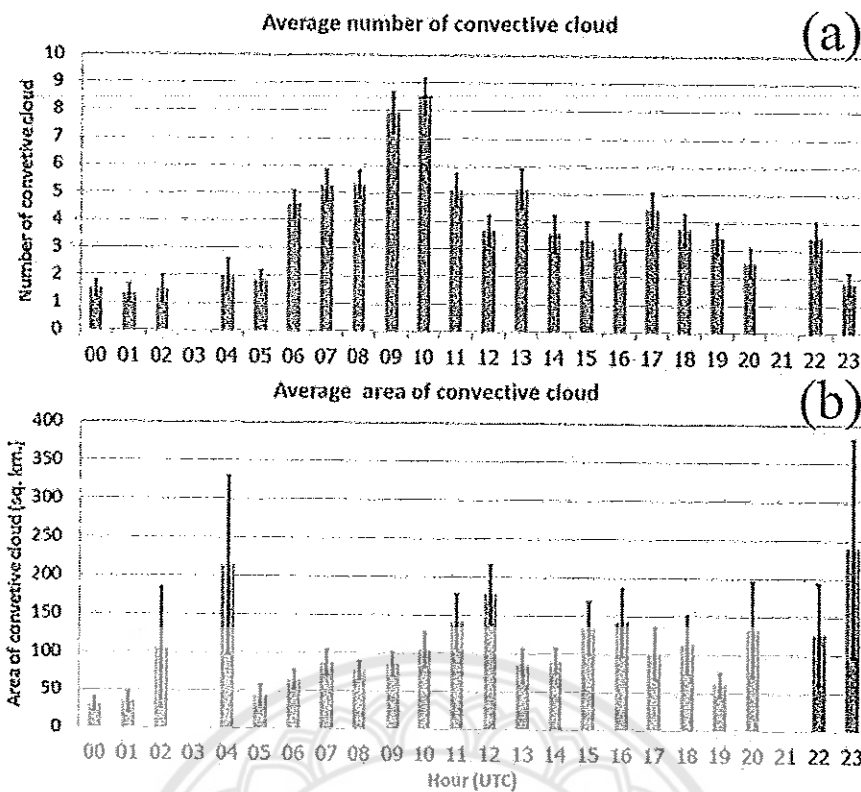


Figure 8: the diurnal variation of convective cloud (a) average number of convective clouds (b) average area of convective clouds. Standard error is shown as error bars

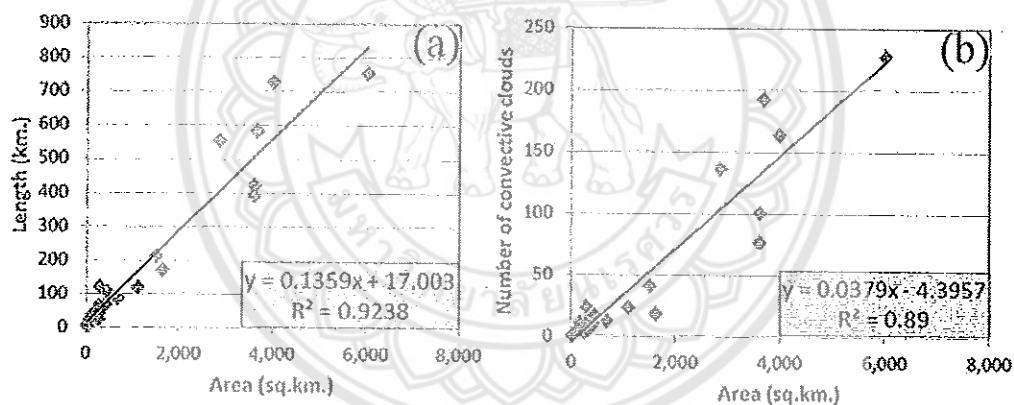


Figure 9: the relation of total area of hourly convective cloud with (a) total of hourly length (b) total number of convective clouds

The total area of hourly convective storm has been analyzed during the Sonca period to find relationship with the extracted convective cloud properties. These properties are total length and total number of which the relationships are as shown in Figure 9. Figure 9(a) shows the strong relationship between the total area and the total length of the convective strong which can summarize that during decaying stage of Sonca storm over the study area, the morphology of strong convective clouds has shown the elongate shapes at certain hour. For

example at 11 UTC (18 LST), the total area of convective storm has reach over 6000 km², associated with the summation of the longest major fitting ellipse axis of over 700 km as shown in Figure 9a. In addition, the total area of convective storm is also well related to the total number of convective clouds as shown in the Figure 9b. This result can be the suggestion for TMD to monitor the decaying severe storm in both space and time in their analysis before announcing disaster warnings to the people in the future.

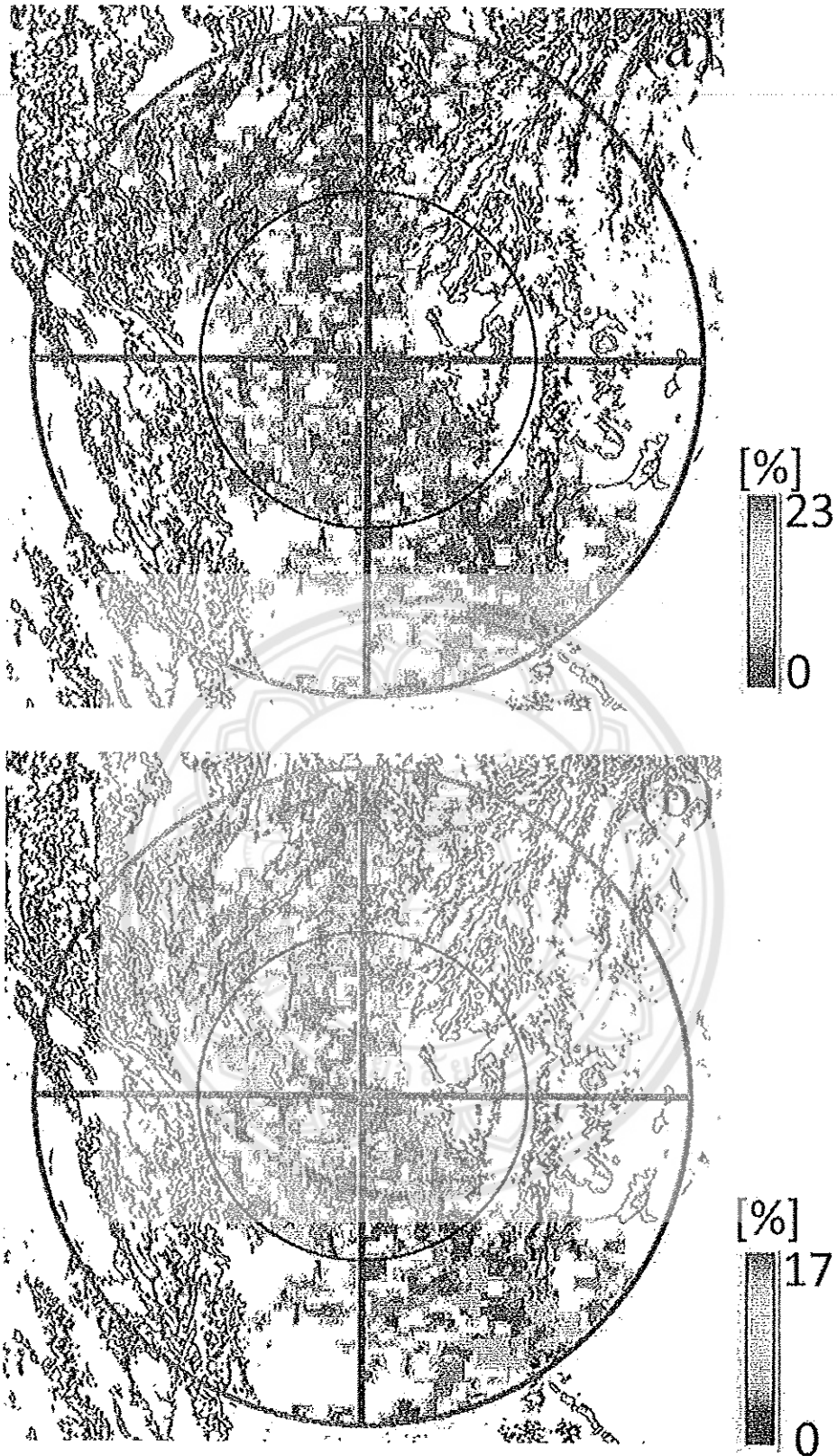


Figure 10: spatial distribution of convective storm frequency occurrence (a) composite of convective storm for 26 July 2017 (b) composite of the highest frequency of area average hour at 23, 12, 04 UTC

Spatial analysis of frequency occurrence for the storm can demonstrate the locations of severe convective storm occurrences as shown in Figure 10. The spatial pattern of convective cloud are located in both northwest and southeast of radar station ranging from mountainous areas to flood plain of Chao Phraya river basin. The high frequency of convective occurrence has been observed over the windward slope of mountain as shown in the composite map of 24-29 Jul 2017 especially on 26 Jul 2017. Composite of spatial convective cloud for the highest peak time of storm area occurs in the location of high frequency in the windward side as shown in Figure 10 (a) and Figure 10 (b). The Mae pool in Uttaradit province, north of the Phitsanulok radar station, where had historical suffered from landslide and flashflood, has shown frequency of convective storms at 17% during the storm period. In 2006, Asian Disaster Preparedness Center (ADPC) had observed flashflood/landslide damaging area over the northern region of Phitsanulok radar coverage in Uttaradit and Sukhothai provinces, after low pressure causing by severe tropical storm named Chanchu during 21-23 May 2006. The unusual rainfall intensity induced by low pressure causing from the severe storm Chanchu in early monsoon had largely affected the observed areas by destabilizing the slope over the risky area. The inappropriate landuse over high slope area in risky area had also been observed. ADPC suggested to have the monitoring the evolution of low pressure during dissipation of the tropical storm to issue some early warning over risky area that could decrease the damage from the flashflood and landslide. Therefore, the extracted radar information as convective region is one of the useful information during monsoon season in Thailand to monitor hydro-meteorological disasters especially for risky area. Moreover, to predict the inland decay of storms, understanding decaying typhoon characteristics is needed over the mid of ICP after the landfalls of Typhoon by using both observations and models. Not only rainfall induced by the decaying Typhoon, but also the producing wind during the Typhoon events are also needed to be elucidated. There is a model for predicting the decay of tropical cyclone winds after landfall that had been developed by Kaplan and DeMaria (1995). The model is developed a simple empirical model based on the least squares fit of an exponential decay equation to the NHC best-track 1-min maximum sustained surface wind estimates for all tropical storms and hurricanes that caused landfall in the south of 37°N United States for the period of 1967-93. The model shows that the maximum winds inland are a function of the maximum winds

at landfall and the time after landfall. It can estimate the maximum inland penetration of winds of a given speed, where the storm's landfall intensity and speed of motion are known. The radial velocity measured by Doppler radar is also needed to analyze because it can be used to validate the wind pattern from mesoscale numerical model such as the study of decaying Typhoon Lekima in 2007 done by Satomura et al., (2013).

5. Conclusion

The tropical storm named Sonca has been analyzed to find the spatio-temporal pattern over the Phitsanulok radar coverage area, in the middle of Thailand. The storm occurred over the study area during 24-29 July 2017 causing the large rainfall amount over the study area. The radar reflectivity from images operating by Thai Meteorological Department (TMD) has been downloaded through website of the Hydro and Agro Informatics Institute. The radar images at frequency of once per hour have been extracted by the developed method. The extracted reflectivity has been processed to detect convective cloud regions. Then, the spatio-temporal analysis of detected convective cloud has been done. The method and analysis have been developed using Python script and OpenCV, Computer Vision package to process image data. The developed method is practical to apply in reanalysis of storm in archiving images to understand the characteristics and mechanism of convective cloud which is the significant part to induce the disaster events such as flash flood and landslide over the middle of Thailand. In the spatio-temporal analysis of storm, it discovered the high frequency occurrence of the storm over the windward slope of mountain. The result of beam blockage analysis using Wraddlib library has shown the east side of Phitsanulok radar severely suffering from terrain blocking the propagating beam. The usage of multiple radars to monitor over risky area is needed to overcome the beam blockage by terrain problem. The further study should be included the analysis of stratiform cloud to compare with the convective clouds. In addition, the atmospheric circulation before and during the storm events are needed to be investigate to understand the atmospheric circulation and structures of the event.

Acknowledgements

This research was supported by National Research Council of Thailand (NRCT) through the Naresuan University R2561B062 and R2561B063. We are really appreciated to "Advancing Co-design of Integrated Strategies with Adaptation to Climate Change in Thailand (ADAP-T)" supported by the

Science and Technology Research Partnership for Sustainable Development (SATREPS), JST-JICA for providing research fund to support the project. reflectivity images. In addition, we gratefully thank to developer of OpenCV and Wrادlib libraries for providing source codes used in Python script to be adapted in the processing of radar reflectivity images.

References

- Asian Disaster Preparedness Center, 2006, Rapid Assessment: Flashflood and Landslide Disaster in the Provinces of Uttaradit and Sukhothai, Northern Thailand, May 2006, *Urban Disaster Risk Management*.
- Austin, P. M. and Houze, Jr., R. A., 1970, Analysis of Mesoscale Precipitation Areas. *Proceedings, 14th Conference on Radar Meteorology, Boston, American Meteorological Society*, 329-334. Abstract in *Bull. American Meteor. Society*, 51, 80.
- Austin, P. M. and Houze, Jr., R. A., 1972, Analysis of the Structure of Precipitation Patterns in New England. *Journal of Applied Meteorology*, Vol. 11, 926-935.
- Bangkok Post., 2017, North Suffers Effects of storm 'Sonca'. Available at: <https://www.bangkokpost.com/news/general/1294527/north-suffers-effects-of-storm-sonca>. [Accessed 17 Jul. 2018].
- Biggerstaff, M. I. and Listemaa, S., 2000, An Improved Scheme for Convective/Stratiform Echo Classification using Radar Reflectivity, *Journal of Applied Meteorology*, Vol. 39, 2129-2150.
- Carbone, R. E. and Tuttle, J. D., 2008, Rainfall Occurrence in the U.S. Warm Season: the Diurnal Cycle. *Journal of Climate*, Vol. 21, 4132-4146.
- Duda, R. O. and Hart, P. E., 1972, Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Communications of the ACM*, Vol. 15, 11-15.
- Fitzgibbon, A. W. and Fisher, R. B., 1995, A Buyer's Guide to Conic Fitting. *Proc. 5th British Machine Vision Conference*, Birmingham, 513-522.
- He, Z., Zhang, Q. and Sun, J., 2016, The Contribution of Mesoscale Convective Systems to Intense Hourly Precipitation Events during the Warm Seasons over Central East China. *Advances in Atmospheric Sciences*, Vol. 33, 1233-1239.
- Heistermann, M., Jacobi, S. and Pfaff, T., 2013, Technical Note: An Open Source Library for Processing Weather Radar Data (wradlib). Special thanks to the Thai Meteorological Department and Hydro and Agro Informatics Institute for providing free downloadable radar *Hydrology and Earth System Sciences*, Vol. 17, 863-871.
- Houze, Jr., R. A., 1973, A climatological study of vertical transports by cumulus-scale convection. *Journal of the Atmospheric Sciences*, 30, 1112-1123.
- Houze, Jr., R. A., 2004, Mesoscale Convective Systems. *Reviews of Geophysics*, Vol. 42, 10.1029/2004RG000150, 43.
- Kaplan, J. and DeMaria, M., 1995, A Simple Empirical Model for Predicting the Decay of Tropical Cyclone Winds After Landfall. *Journal of Applied Meteorology*, Vol. 34, 2499-2512.
- Lang, T. J., Ahijevych, D. A., Nesbitt S. W. and Carbone, R. E., 2007, Radar-Observed Characteristics of Precipitating Systems During NAME 2004. *Journal of Climate*, Vol. 20, 1713-1733.
- Mahavik, N., T. Satomura, S., Shige, B., Sysouphanthavong, S., Phonevilay, Wakabayashi, M. and Baimoung, S., 2014, Rainfall Pattern over the Middle of Indochina Peninsula During 2009-2010 Summer Monsoon. *Hydrological Research Letters*, Vol. 8, 57-63.
- Nesbitt, S. W., Cifelli, R. and Rutledge, S. A., 2006, Storm Morphology and Rainfall Characteristics of TRMM Precipitation Features. *Monthly Weather Review*, Vol. 134, 2702-2721.
- Rigo, T. and Llasat, M. C., 2004, A Methodology for the Classification of Convective Structures Using Meteorological Radar: Application to Heavy Rainfall Events on the Mediterranean coast of the Iberian Peninsula. *Natural Hazards and Earth System Sciences*, Vol. 4, 59-68.
- Rabus, B., Eineder M., Roth A. and Bamler R., 2003, The shuttle radar topography mission - a new class of digital elevation models acquired by space borne radar. *Journal of Photogrammetry and Remote Sensing*, Vol. 57, 241-262.
- Rigo, T. and Llasat, M. C., 2005, Radar analysis of the life cycle of Mesoscale Convective Systems during the 10 June 2000 event. *Natural Hazards and Earth System Sciences*, Vol. 5, 959-970.
- Rigo, T. and Llasat, M. C., 2007, Analysis of Mesoscale Convective Systems in Catalonia Using Meteorological Radar for the Period 1996-2000. *Atmospheric Research*, Vol. 83(2-4), 458-472.
- Rinehart, R. E., 1999, Radar for Meteorologists 3rd ed, Rinehart publication, P.O.Box6124, Grand Forks, ND 58206-6124, USA, 428.

Satomura, T., Yamamoto, K., Sysouphanthavong, B. and Phonevilay, S., 2011, Diurnal Variation of Radar Echo Area in the Middle of Indochina. *Journal of the Meteorological Society of Japan*, Vol. 89a, 299-305.

Satomura, T., Katsumata, M., Mori, S., Yokoi, S., Matsumoto, J., Ogino, S. and Kamimera, Y., 2013, To Understand Typhoons' Behavior Over Indochina. *Journal of Disaster Research*, Vol. 8, 153-154.

Schumacher, S. R. and Johnson, R. H., 2006, Characteristics of U.S. Extreme Rain Events during 1999–2003. *Weather and Forecasting*, Vol. 21, 69-85.

Steiner, M., Houze, Jr., R. A. and Yuter, S. E., 1995, Climatological Characterization of Three-Dimensional Storm Structure from Operational Radar and Rain Gauge Data. *Journal of Applied meteorology*, Vol. 34, 1978-2007.



Quality Assessment of mosaicked weather radars over The Chao Phraya river basin, Thailand

Nattapon Mahavik and Sarintip Tantanee

Abstract— The developing countries including Thailand, people still have suffered from disaster since they are lacking of adequate information to cope with disaster. Well preparedness for disaster management is one of the most important responsibilities of the governments all over the world. However, appropriate information is needed to cope with spatial and temporal variations of destructive disaster. The weather radar is one of the tools that can provide spatio-temporal information for Nowcast which is useful for hydrological disaster warning and mitigation system. The ground-based weather radar can provide spatial and temporal information to monitor severe storm over the risky area. However, the usage of multiple radars can provide more effective information over large study area where single radar beam may be blocked by surrounding terrain. However, the mosaicked radar product is not yet officially available for Thai researcher to be investigated the physical characteristics of the severe storm.

In this study, algorithm of mosaicked radar reflectivity has been developed by using data from ground-based radar of Thai Meteorological Department over the Chao Phraya river basin in the middle of Thailand. The Python script associated with OpenCV libraries are used in our investigations of the mosaicking processes. The radar quality index (RQI) field has been developed by implementing an equation of a quality radar index to identify the reliability of each mosaicked radar reflectivity pixels. First, the percentage of beam blockage is computed to understand the radar beam propagation obstructed by surrounding topography in order to clarify the limitations of the observed beam on producing radar reflectivity maps. Second, the elevation of beam propagation associated with distance field has been computed. Then, these two parameters and the obtained percentage of beam blockage are utilized as the parameters in the equation of the quality index (QI). Finally, the detected radar flare, non-precipitating radar area, has been included to the QI field of which obtained result is the QIF. Then, QIF has been applied to the extracted radar reflectivity. The developed QIF field will be improved further toward using in real-time operation of radar mosaicked processing over the Chao Phraya river basin.

Keywords— Radar mosaicking, Quality index, Thailand, Chao Phraya river basin, Python.

1. INTRODUCTION

The deployment of the weather radar network from Thai Meteorological Department (TMD) has provided meteorologists with critical information toward the issuance of warning for severe weathers, severe storms and flash flood. In addition, the information of moving precipitation derived only from the single radar has provided information to public over the radar coverage. The disadvantage of using single radar has been addressed such as spatial differences in the sampling properties for both horizontal and vertical, beam blockage, range-dependent biases [1]. However, the mosaicking of multiple radars will provide more information for meteorologists to observe an evolution of severe storm which can increase the accuracy on forecasting and warning system.

In developed countries such the United State, the algorithm of the national mosaicking of multiple radar has been developed by using the operational WSR-88D for producing real time radar-derived rainfall product to

support the warning and forecasting mission of the National Weather Service more than 20 years [2]. Recently, the national mosaic integrating radar, rain gauge, satellite and numerical weather prediction data have been automate fused into a seamless national 3D radar mosaic product [3]. The depiction and rendering of storm structure from the 3D radar mosaic products can provide more insightful information which US meteorologists can apply to their warning systems. This can help reducing the losses from the hydro-meteorological disasters. However, in the developing countries including Thailand, official mosaicked radar products is unavailable to be used for research purpose and real-time monitoring of severe weather.

Zhang et al. [4] have introduced the Radar Quality Index (RQI) by a combined measure for beam blockage and the vertical profile of reflectivity (VPR) effects in a national radar mosaic network. The quality of the next-generation multi-sensor quantitative precipitation estimation (QPE) varies in space and in time due to a number of factors, which includes: (1) errors in measuring radar reflectivity; (2) segregation of precipitation and non-precipitation echoes; (3) uncertainties in Z-R relationships; and (4) variability in VPR. Radar Quality Index (RQI) field is developed to describe the radar QPE uncertainty associated with VPRs. The RQI field accounts for radar beam sampling characteristics (blockage, beam height and width) and their relationships with respect to the freezing level. However, the VPRs information is hard to derive from TMD data due to limitation elevation angles used in the operational observation. In fact, the

Nattapon Mahavik is with Naresuan University, Thailand. He is now with the department of natural resources and environment, faculty of agriculture natural resources and environment, 65000, Thailand (phone: 66-94-927-4488; e-mail: nattaponm@nu.ac.th)

Sarintip Tantanee is with Naresuan University, Thailand. She is now with department of civil engineering, faculty of engineering and also with centre of excellence on energy technology and environment, Naresuan University, e-mail: sarintipt@nu.ac.th

radar network over the middle of Indochina Peninsula mainly in Thailand is quite densely distributed, but rainfall products derived from these radars are not yet publicly available. In addition, these radars are not unified in terms of their set parameters, scheduling of observation times, and data formats [5].

In this study, the RQI for TMD radar mosaicking product has been developed. First, we developed simple method to obtain mosaicked radar over the Chao Phraya River basin which locates in the middle of Thailand. The developed RQI field has been integrated with detected radar flares which are the non-precipitating rain pixel before applying to the mosaicked radar reflectivity.

2. DATA AND STUDY AREA

Radar reflectivity data of two stations, which are Phitsanulok and Chainat, has been used in mosaicking process to find radar flares. The Phitsanulok and Chainat weather radar stations are located at the Lower Northern Thailand (Figure 1).

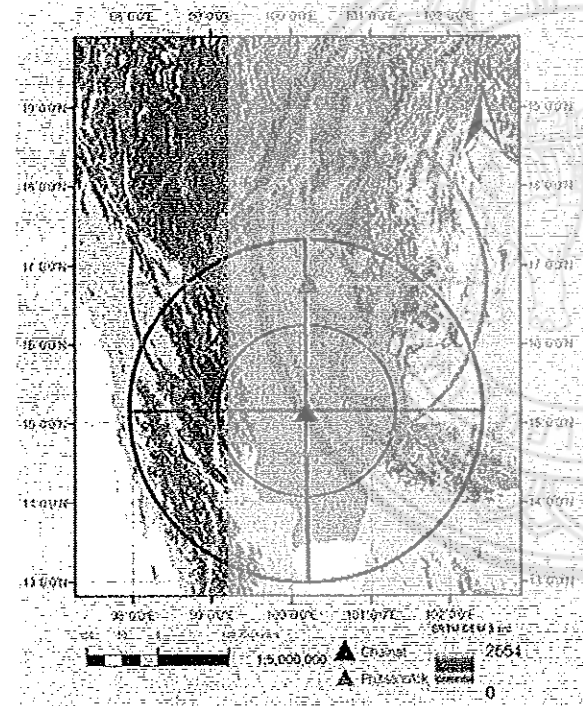


Figure 1 Study area over Chao Phraya River basin (shown in blue area) located in the middle of Thailand. Radar observation radius at Phitsanulok and Chainat stations with two observation ranges of 120 and 240 km, respectively, in the mode of 1st PPI (0.5°) provided by Thai Meteorological department.

The Phitsanulok radar site is located at latitude of 16°46'30.358"N and longitude of 100°13'4.312"E over the height of 47 m above mean sea level with tower height of 30 m. Relatively flat compared to Phitsanulok, The Chainat radar is located in the middle of Chao Phraya River basin over the height of 18 m above mean sea level at latitude of 15°9'27.898"N and longitude of 100°11'24.242"E which is in relative flat area comparing to Phitsanulok radar. Both radar data are observed by Thai Meteorological Department (TMD) at four time per hour, While the data has been archived as images by Hydro and Agro Informatics Institute (HAI) at frequency of once per

hour in format of GIF. The observation radius of the Phitsanulok and Chainat radar is 240 km observing in C-band frequency with beam width of 1°. In this study, the Plan Position Indicator (PPI) image of the first elevation at 0.5° from horizontal line have been collected and used in the analysis. In this study, radar data during influencing period of Sonca tropical storm on 26th July 2017 according to TMD warning announcement has been processed to find RQI.

3. METHODOLOGY

The workflow of this study is shown in Figure 2. Quality index of mosaicked radars is developed to describe the quality of mosaicked radar reflectivity pixels. Three factors have been considered to develop the quality index (QI).

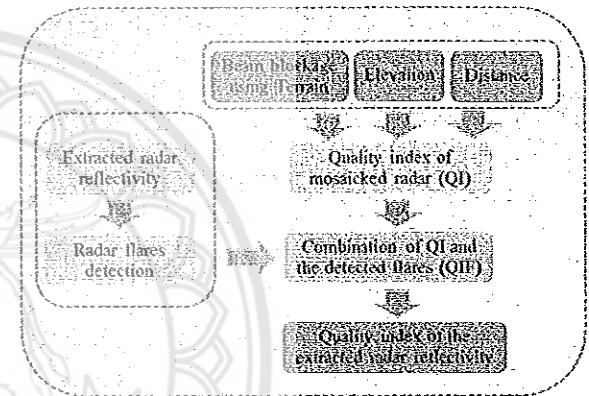


Figure 2 Flowchart of the developed radar quality index in assessment of the mosaicked reflectivity of two radars.

First factor is the geometry of radar beam propagation which has been used to find area where beam blockage over 50% of beam width by considering with terrain data from SRTM DEM V4 with resolution of 3 arc second. Second factor is the elevation of center radar beam as propagating through the air which has been included to the QI calculation. Last factor is the distance of propagating radar beam which is also considered to realize the beam range to the end users. The combination of those mentioned three factors are simply integrated in the linear equation with initial weighting based on reviews as shown in equation 1. The beam blockage has been considered as the highest weights compared to the other two factors because the beam blockage can diminish the returned radar reflectivity and it may increase uncertainty of radar rainfall estimates. The computation of those three factors is done on the basis of single station afterward mosaicking the calculated factors.

$$QI = (0.25 * Distance) + (0.25 * Elevation) + (0.5 * Beam \text{ blockage}) \quad (1)$$

The additional information of radar flares has been included to the QI to inform the severely affected areas of radar reflectivity. The flares are not rain areas but it occurs because of the interfering from other signal in communication sectors. The locations of radar flares usually change in space and time. However, the radar flares are obviously observed over the Phitsanulok radar and the permanent flare location in the west of radar site is also observed after the radar beam crosses over the mountain in Sukhothai province. The computed radar

flares of the Phitsanulok radar have been included with the QI, then, the quality index of QIF has been obtained.

4. RESULTS AND DISCUSSION

To study of rainfall characteristics in mesoscale, the mosaicked multiple radars is needed because of the high spatiotemporal information. This information can be used in rainfall analysis over watershed such the Chao Phraya river basin. However, the mosaicked radar product is not available for research in Thailand. Therefore, the public accessible radar reflectivity is used in this study. The Reflectivity data from these two stations are simply mosaicked using maximum value of extracted radar reflectivity as shown in Figure 3a.

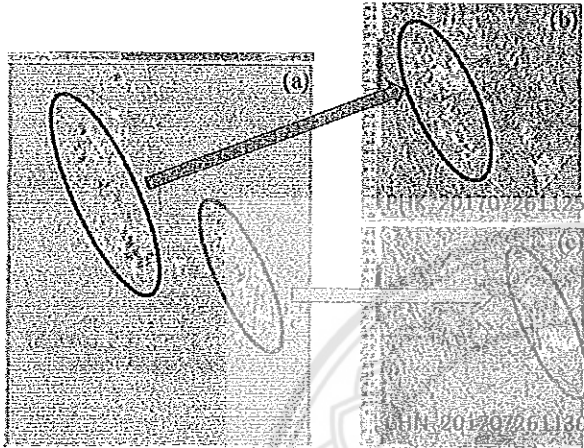


Figure 3 Results of mosaicked radar reflectivity of the 1st PPI on 26 July 2017 at 11.30. (a) the mosaicked map of the extracted radar reflectivity using Phitsanulok and Chainat stations (b) the original image of radar reflectivity at Phitsanulok (c) the original image of radar reflectivity at Chainat.

The two areas of mesoscale moving clouds which consisted of stratiform and convective clouds are obviously shown. Those reflectivity data are observed at 5 minutes of time difference at 11 o'clock of local time as shown in Figure 3b and 3c. In addition to mosaicking the extracted radar reflectivity, the information of used cloud pixels in the mosaicking process has also been provided which is not shown in the Figure. The index map of used radar varies in time depending on the criteria of maximum intensity of considering pixels. It can be concluded that the reliability of mosaicked radar reflectivity is not yet provided in the index map of used radars. Therefore, the radar quality index must be further developed for end user.

The highest impact factor for the radar quality index is the beam blockage information. In this study, the area of beam blockage at 50% by terrain information from SRTM DEM has been computed at pixel-based basis for each single radar as shown in Figure 4. The blocked beam by terrain for the first elevation at 0.5° is clearly shown over the west and east of Phitsanulok radars in Figure 4a, while the Chainat radar has small areas of blocked beam over the west side of radar coverage as shown in Figure 4b. The index map of beam blockage after mosaicking the two radars as shown in Figure 4c can be used to identify the unaffected area of beam blockage for both radars. The large area of beam blockage in the east of Phitsanulok site does not get the advantage from the mosaicking. In the

east of Phitsanulok radar, several pixels indicate the unreliable returned radar reflectivity. This mosaicked beam blockage map is useful to fill the gap of beam blockage for increasing the reliability of mosaicked radar products.

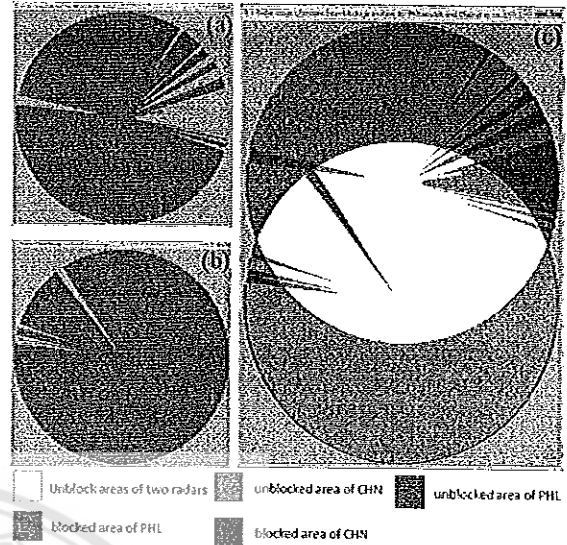


Figure 4 Beam blockage analysis with the surrounding terrain from SRTM DEM using threshold at 50% of beam width (a) Phitsanulok (b) Chainat (c) The mosaicked map of beam blockage from two radars.

The beam distance and elevation have been repeatedly computed for each radar site and following by mosaicking mosaicked in the dimension of two radar coverage as shown in Figure 5. The far range of radar station may be either underestimate or overestimate of rainfall when it is validated by gauge data at ground level. These two information are important for radar quality during the mosaicking process as weighting factors in the radar quality equation.

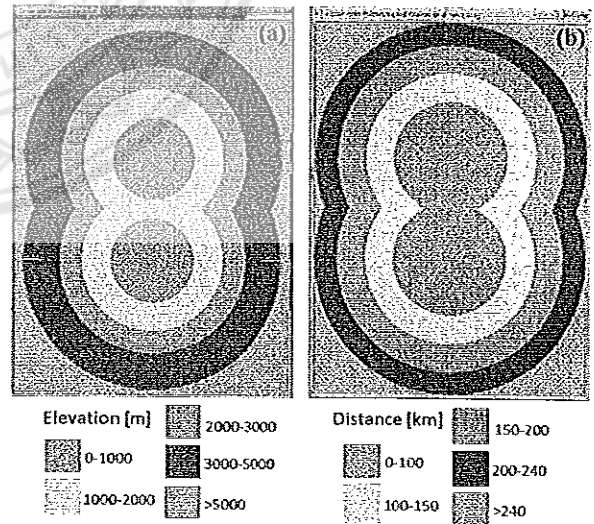


Figure 5 The mosaicked radar map for (a) elevation of radar beam propagation (b) distance of radar beam

The three factors are combined into the linear equation of the radar quality index using initial weighting and it is defined as the QI as shown in Figure 6a. The interval of quality index indicates the reliability of mosaicked radar reflectivity in the sample scanning time. The west side of

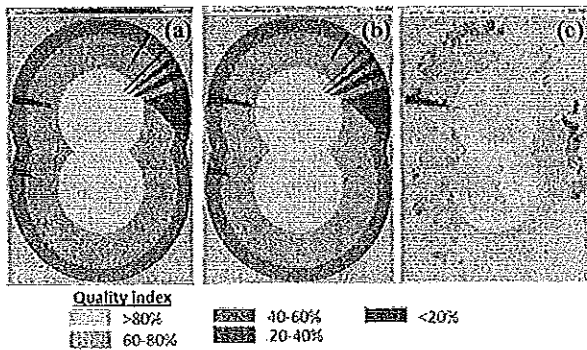


Figure 6 Radar quality index of (a) Quality index (QI) from the three factors (b) including the detected radar flares to the QI (c) results of applied the QIF to the extracted radar reflectivity on 26 July 2017 at 11.30.

Phitsanulok radar is obviously recognized the low reliability at less than 20% due to beam blockage. However, the majority of the radar reflectivity pixels are in higher reliability at greater than 60%.

Finally, QIF which is the combination of the detected radar flares to the QI, has been done to include the severely affected of radar flares as shown in Figure 6b. The result after applying QIF to the extracted radar reflectivity is recognized by the extended triangle toward the radar station of Phitsanulok as shown in Figure 6c. However, the QIF varies from scan to scan due to unpredictable flare in their locations except for the permanent flare located in the west of Phitsanulok radar.

5. CONCLUSION

A radar quality index (RQI) of mosaicked radar reflectivity was developed in this study over area of Chao Phraya River basin, Thailand. The radar data from Phitsanulok and Chainat radar of Thai Meteorological Department has been used in the study. The geometry of beam propagation has been considered with the digital elevation model to find the beam blockage area. The beam blockage area is the highest impact factor to the developed linear radar quality equation. The elevation and distance of beam propagation has been calculated in basis of gridded Cartesian system to include initial weighting factors in the radar quality equation. The initial weight of the equation can be used to assess each returned radar reflectivity of particular scanning time. The extracted radar reflectivity has been used as input information to detect a radar flare, non-precipitating area, over the single radar. Then, the radar flare can be added into the RQI field as the lowest reliability of mosaicked radar areas which named as QFI field. From scan to scan, the QFI is repeatedly computed for each scanning data that is identical field due to the changing in space and time of the detected radar flares. The result of applied QFI indicates the reliability of the mosaicked radar reflectivity for each pixel. In addition, the QFI field indicates radar coverage voids which is for the future process of gap-filling radars. Future work will include the validation process using a gauge network to assess the mosaicked radar products as one of factors. The developed radar quality equation will also be investigated the weight of each factors to realize the effect of its changing. The error-related RQI field will provide more information to the mosaicked radar product

toward for the real-time operation and reanalysis data implementation over the Chao Phraya river basin.

ACKNOWLEDGMENT

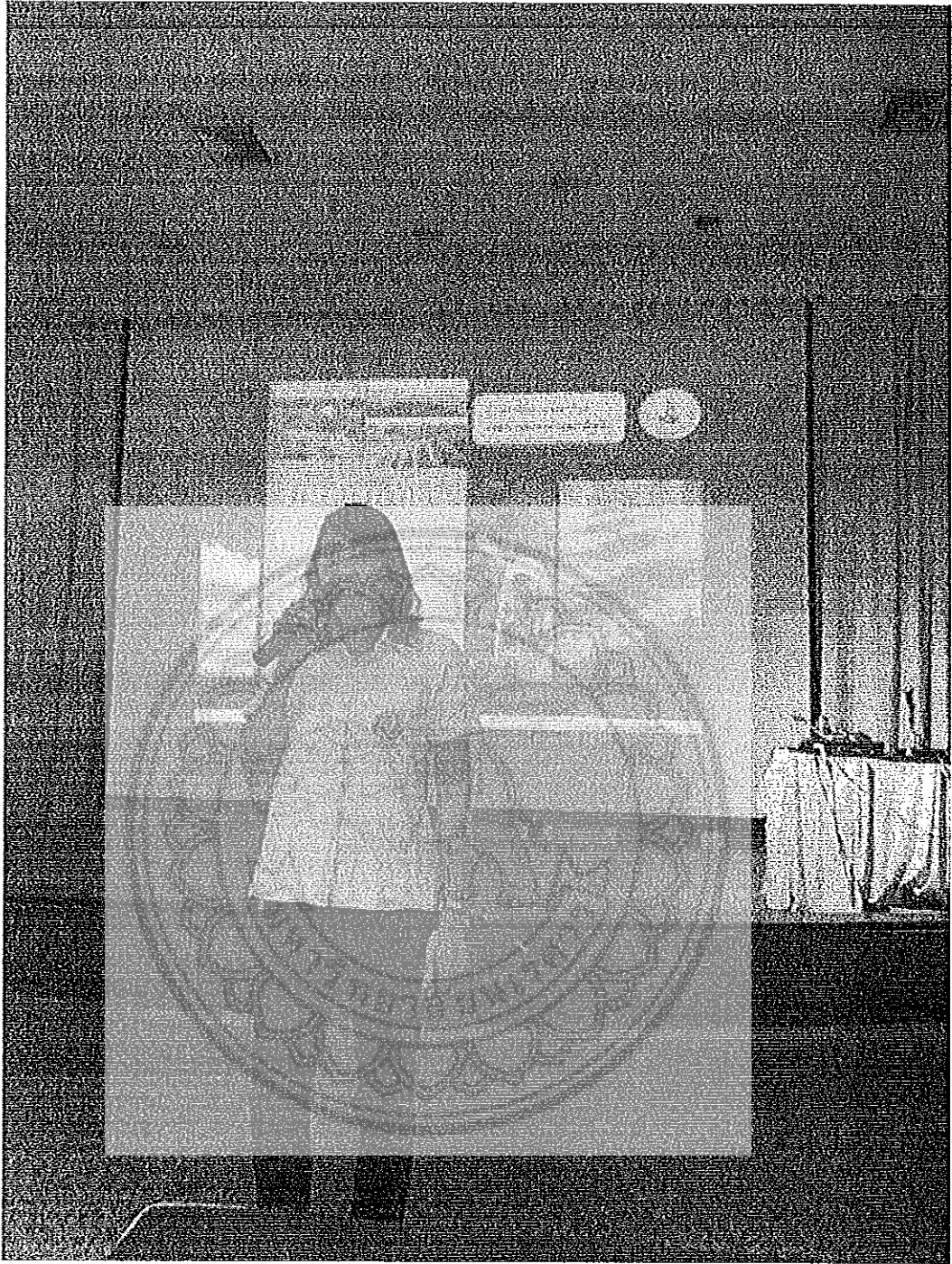
This research was supported by National Research Council of Thailand (NRCT) through the Naresuan University R2561B062 and R2561B063. We are really appreciated to Advancing Co-Design of Integrated Strategies with Adaptation to Climate Change in Thailand project (ADAP-T) in ST2-W1 group supported by JICA for providing additional fund to support the project. Special thanks to the Thai Meteorological Department and Hydro and Agro Informatics Institute for providing free downloadable radar reflectivity images. In addition, we gratefully thank to developer of OpenCV library for providing source codes used in Python script to be adapted in the processing of radar reflectivity images.

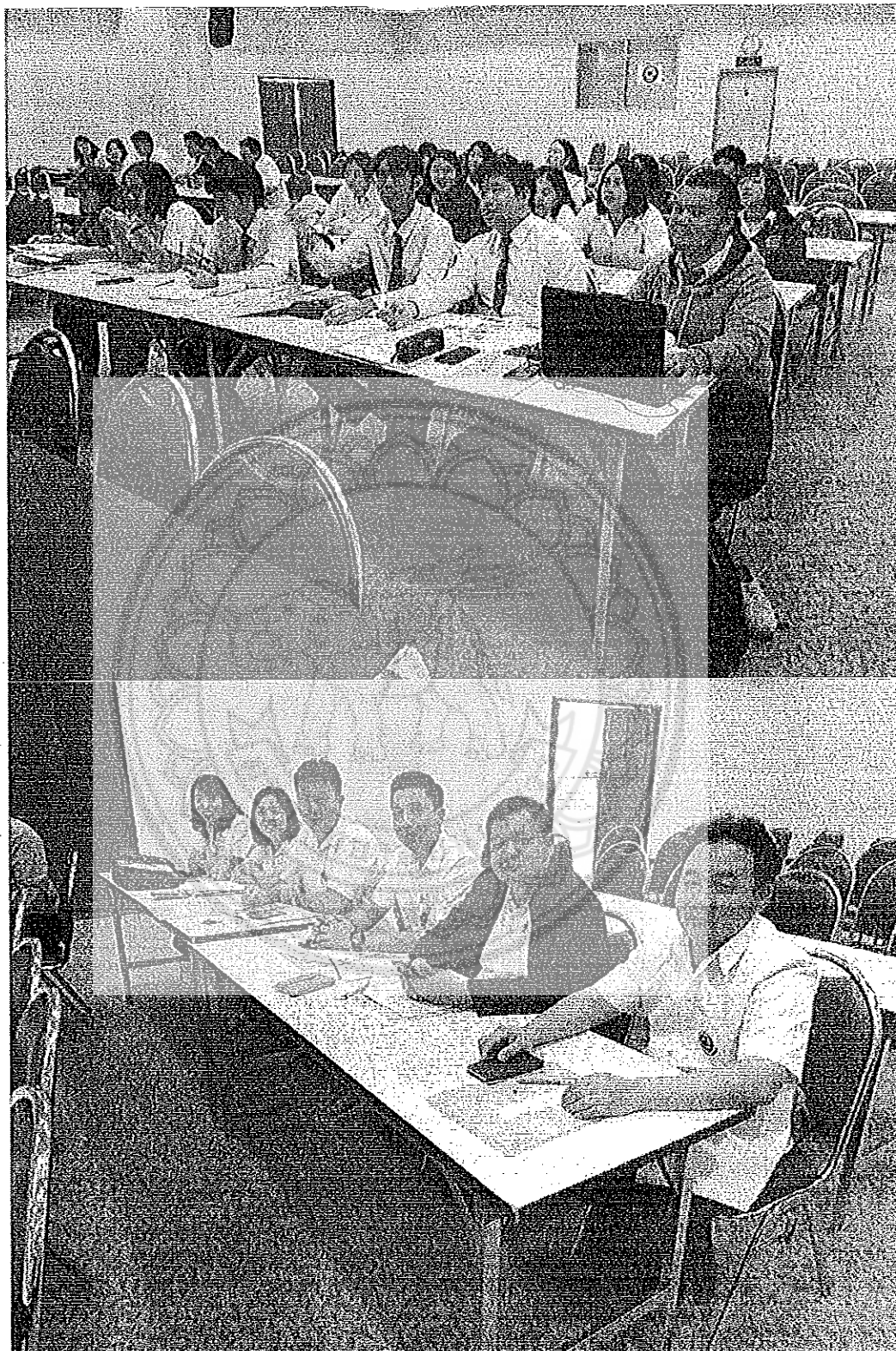
REFERENCES

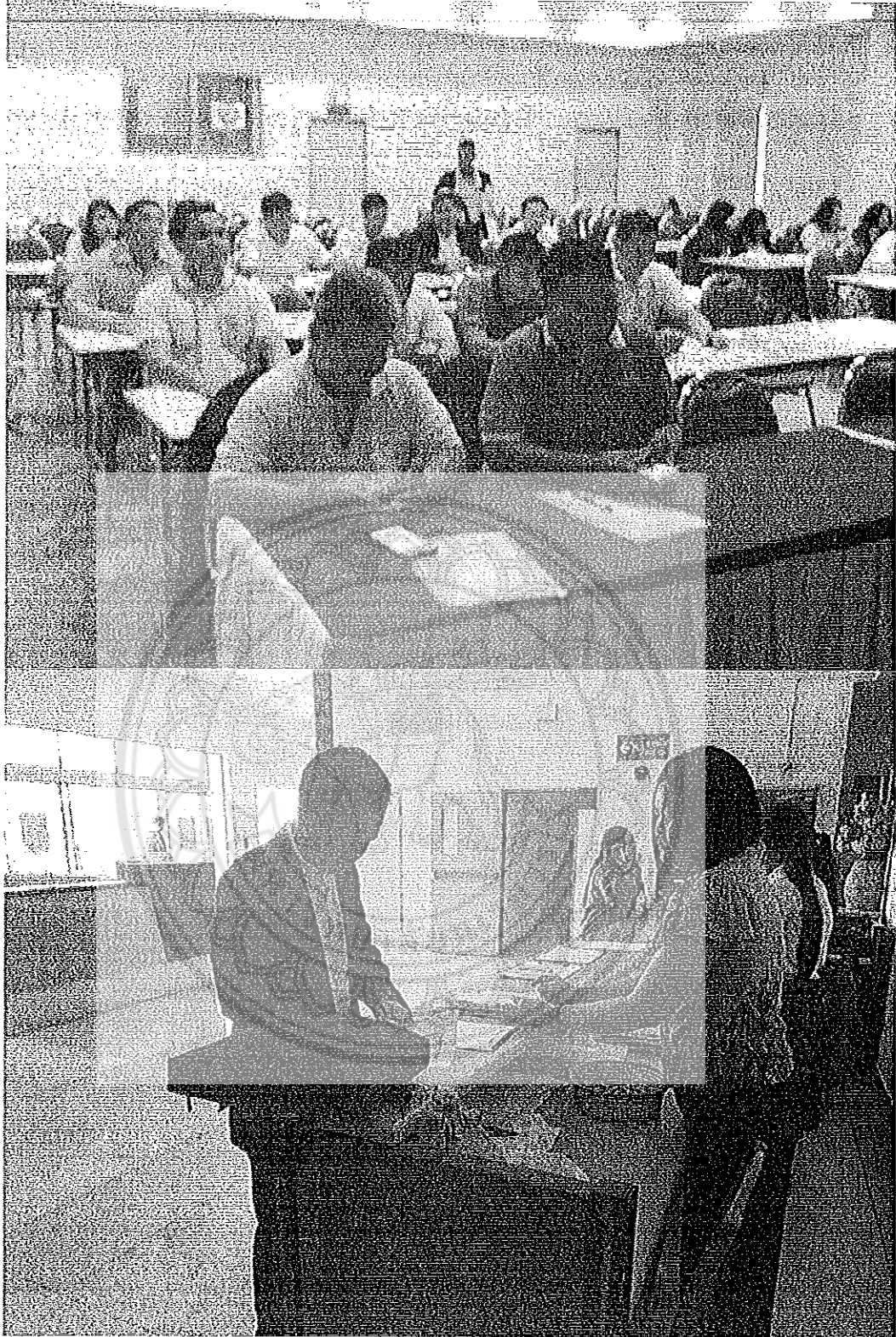
- [1] Mcroberts, D. B. and Nielsen-Gammon, J. W. 2017. Detecting Beam Blockage in Radar-Based Precipitation Estimates. *Journal of Atmospheric and Oceanic Technology* 34: 1407–1422.
- [2] Fulton, R., Breidenbach, J., Seo, D.-J., Miller, D. and O'Bannon, T. 1998. The WSR-88D Rainfall Algorithm. *Weather and Forecasting* 13: 377–395.
- [3] Zhang J. et al., 2011. National Mosaic and multi-sensor QPE (NMQ) system: Description, results, and future plans. *Bulletin of the American Meteorological Society*. 92: 1321–1338.
- [4] Zhang, J., Youcun, Q. L., Carrie, L. and Kaney, B. 2011. Radar Quality Index (RQI) – a combined measure for beam blockage and VPR effects in a national network. *Weather Radar and Hydrology (IAHS Proceeding and Reports)* 351, 2011.
- [5] Satomura, T., Katsumata, M., Mori, S., Yokoi, S., Matsumoto, J., Ogino, S. and Kamimera, Y. 2013. To understand typhoons' behavior over Indochina. *Journal of Disaster Research* 8: 153–154.

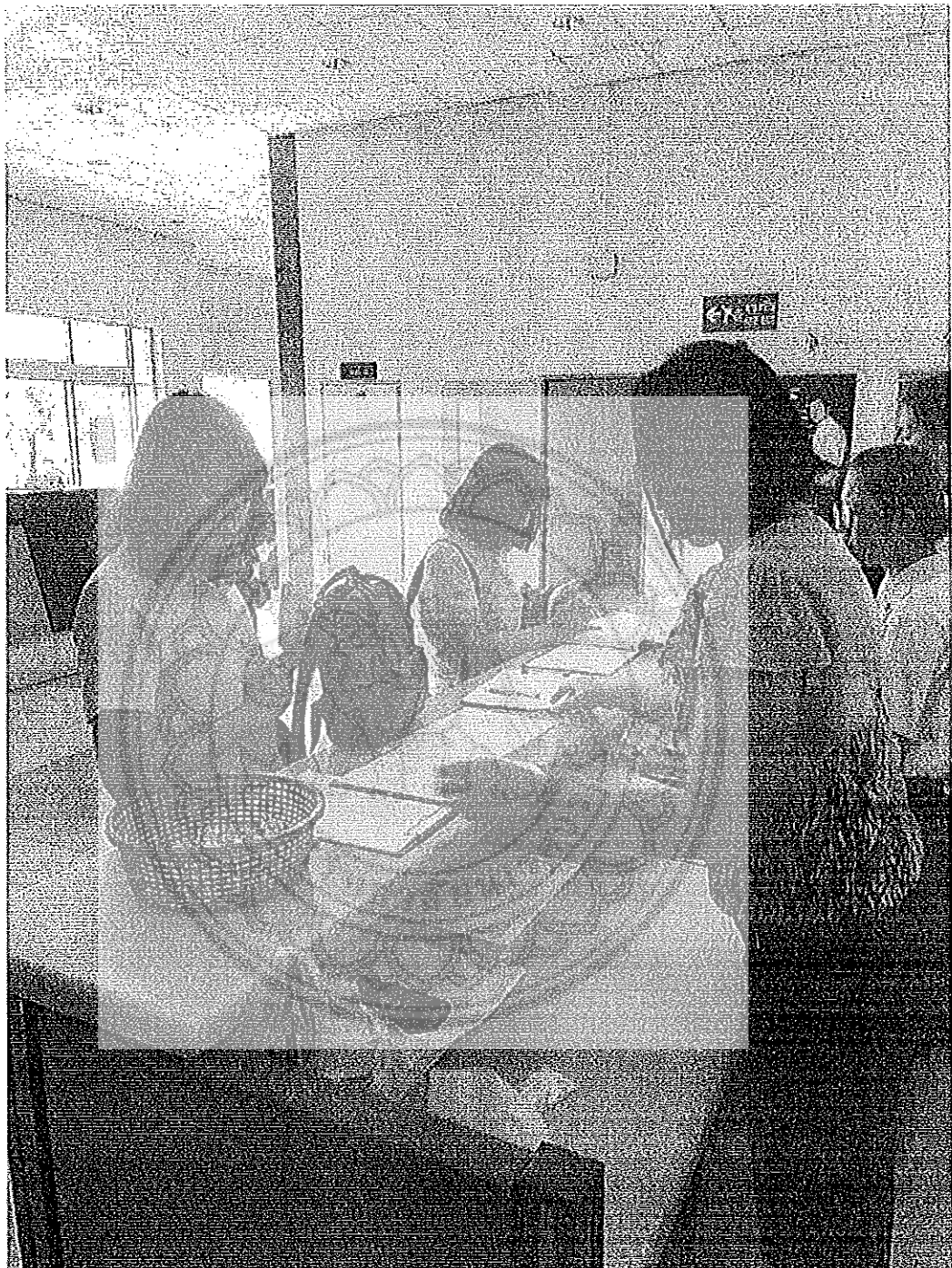


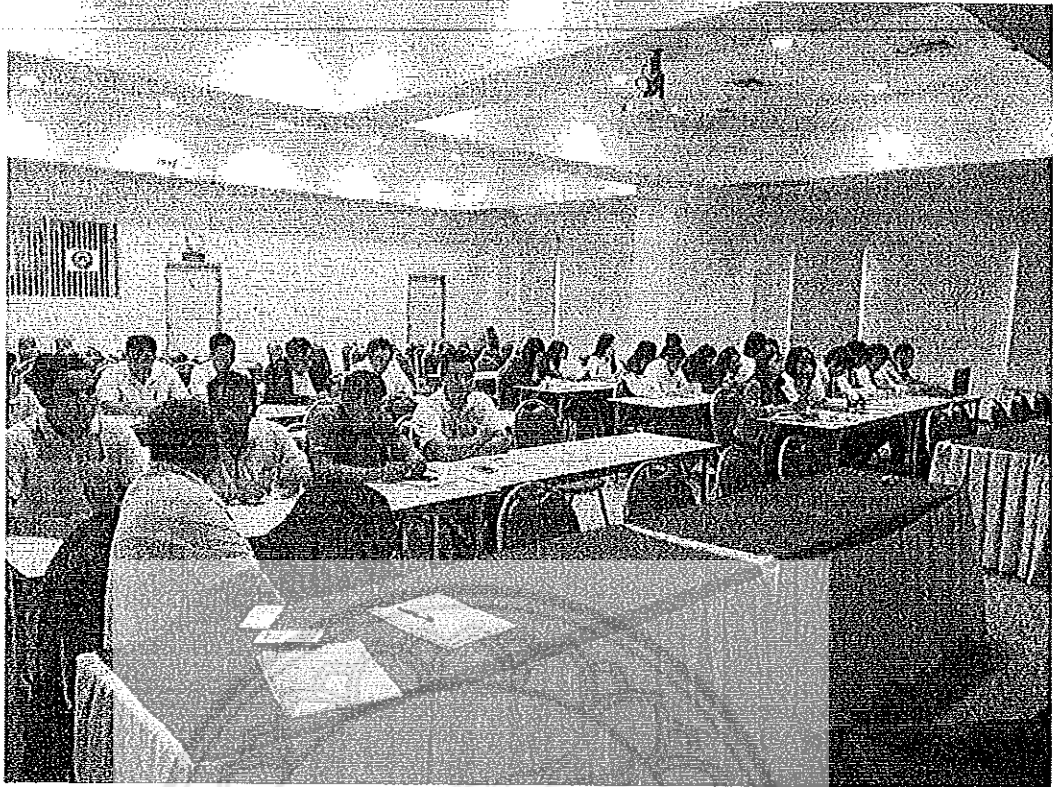














บันทึกข้อความ

ส่วนราชการ คณะเกษตรศาสตร์ฯ ภาควิชาทรัพยากรธรรมชาติและสิ่งแวดล้อม โทร. 2757

ที่ อว 0603.07.01/

วันที่ มิถุนายน 2562

เรื่อง ขออนุมัติจัดโครงการอบรมปิดโครงการวิจัยงบประมาณแผ่นดินรหัสโครงการ R2561B063

เรียน คณบดีคณะเกษตรศาสตร์ฯ

ตามที่ มหาวิทยาลัยขอนแก่น ได้อนุมัติจัดสรรงบประมาณแผ่นดิน ประจำปีงบประมาณ 2561 เพื่อเป็นค่าใช้จ่ายในการดำเนินงานโครงการวิจัยงบประมาณ 2561 ให้กับ ดร.รัฐพล มหาวิค หัวหน้าโครงการ “การพัฒนาวิธีการทำแผนที่โมเสคน้ำฝนประมาณค่าจากข้อมูลเรดาร์ตรวจอากาศภาคพื้นดินบริเวณลุ่มน้ำเจ้าพระยา” รหัสโครงการ R2561B063 วงเงินงบประมาณ 583,000 บาท (ห้าแสนแปดหมื่นสามพันบาทถ้วน) ระยะเวลาดำเนินงาน ตั้งแต่วันที่ 9 ตุลาคม 2560 ถึงวันที่ 8 ตุลาคม 2561 นั้น บัดนี้โครงการได้บรรลุวัตถุประสงค์ที่ตั้งไว้แล้ว

ดังนั้น ข้าพเจ้า ดร.รัฐพล มหาวิค จึงใคร่ขออนุมัติโครงการอบรมเผยแพร่องค์ความรู้ด้านอุตุนิยามวิทยาและการใช้ข้อมูลเรดาร์ในการวิเคราะห์สภาพอากาศ ในวันพฤหัสบดีที่ 11 กรกฎาคม 2562 ตั้งแต่เวลา 09.00 – 16.00 น. ณ ห้องประชุมเมนคอนเฟอร์เร้นท์ อาคาร CITCOM โดยขอเบิกค่าใช้จ่ายดังนี้

- | | |
|--|-----------|
| 1. ค่าอาหารผู้เข้าร่วมโครงการ (200บาทX40 คน) | 8,000 บาท |
| 2. ค่าอาหารว่าง (50บาทX2มื้อX40คน) | 4,000 บาท |
| 3. ค่าวัสดุสำนักงาน | 4,400 บาท |
| 4. ค่าวิทยากร (600บาทX2คนX3ชั่วโมง) | 3,600 บาท |

รวมเป็นเงิน 20,000 บาท

หมายเหตุ ถ้าวางได้ทุกรายการ

ทั้งนี้โดยเบิกจ่ายจากงบประมาณแผ่นดิน ประจำปีงบประมาณ ๒๕๖1 โครงการการพัฒนาวิธีการทำแผนที่โมเสคน้ำฝนประมาณค่าจากข้อมูลเรดาร์ตรวจอากาศภาคพื้นดินบริเวณลุ่มน้ำเจ้าพระยา

จึงเรียนมาเพื่อโปรดพิจารณาอนุมัติ

(ดร.รัฐพล มหาวิค)

หัวหน้าโครงการฯ

โครงการอบรมเผยแพร่องค์ความรู้ด้านอุตุนิยมวิทยาและการใช้ข้อมูลเรดาร์ในการวิเคราะห์สภาพ
อากาศภาควิชาทรัพยากรธรรมชาติและสิ่งแวดล้อม
คณะเกษตรศาสตร์ ทรัพยากรธรรมชาติและสิ่งแวดล้อม มหาวิทยาลัยนเรศวร

ชื่อโครงการ :

อบรมเผยแพร่องค์ความรู้ด้านอุตุนิยมวิทยาและการใช้ข้อมูลเรดาร์ในการวิเคราะห์สภาพ
อากาศ

หลักการและเหตุผล

ตามที่ มหาวิทยาลัยนเรศวร ได้อนุมัติจัดสรรงบประมาณแผ่นดิน ประจำปีงบประมาณ 2561 เพื่อเป็นค่าใช้จ่ายในการดำเนินงานโครงการวิจัยงบประมาณแผ่นดิน ประจำปีงบประมาณ 2561 ให้กับ ดร.นัฐพล มหาวิค หัวหน้าโครงการ “การพัฒนาวิธีการทำแผนที่โมเสคน้ำฝนประมาณค่าจากข้อมูลเรดาร์ตรวจอากาศภาคพื้นดินบริเวณลุ่มน้ำเจ้าพระยา” รหัสโครงการ R2561B063 บัดนี้โครงการได้บรรลุตามวัตถุประสงค์ที่ได้ระบุไว้ในสัญญา เพื่อให้มีการเข้าใจและตระหนักถึงองค์ความรู้ทางด้านอุตุนิยมวิทยาและการตรวจวัดเรดาร์ตรวจอากาศ คณะผู้วิจัยจึงได้จัดการอบรมเพื่อเผยแพร่องค์ความรู้ในชื่อโครงการอบรมเผยแพร่องค์ความรู้ด้านอุตุนิยมวิทยาและการใช้ข้อมูลเรดาร์ในการวิเคราะห์สภาพอากาศ โดยเชิญผู้เชี่ยวชาญจากกรมอุตุนิยมวิทยาและผู้เชี่ยวชาญจากกรมพัฒนาที่ดินมาเป็นวิทยากรในการบรรยายให้กับผู้เข้าประชุม

วัตถุประสงค์

เพื่อเผยแพร่องค์ความรู้ด้านอุตุนิยมวิทยาและการใช้ข้อมูลเรดาร์ในการวิเคราะห์สภาพ
อากาศ

รูปแบบการจัดประชุม

บรรยายองค์ความรู้ทางทฤษฎีและปฏิบัติ รวมถึงแลกเปลี่ยนประสบการณ์ระหว่างหน่วยงาน
ที่เข้าร่วมอบรม

ผู้เข้าประชุม

เจ้าหน้าที่หน่วยงานที่เกี่ยวข้องจากภาครัฐในพื้นที่จังหวัดพิษณุโลกและสุโขทัยและนิสิตที่
สนใจ จำนวน 40 คน

การฝึกอบรม


การเผยแพร่องค์ความรู้ด้านอุตุนิยามวิทยาและการใช้ข้อมูลเรดาร์ในการวิเคราะห์สภาพอากาศ
ภาควิชาทรัพยากรธรรมชาติและสิ่งแวดล้อม คณะเกษตรศาสตร์ มหาวิทยาลัยขอนแก่น
วันพฤหัสบดีที่ 11 กรกฎาคม 2562

08.30-09.00 น.	ลงทะเบียน (ห้องเมนคอนเฟอร์เร็นซ์ อาคาร CITCOM)
09.00-10.45 น.	การบรรยายเรื่อง การตรวจสภาพอากาศด้วยเรดาร์ตรวจอากาศภาคพื้นดิน วิทยากร: คุณนนท์ธวัช เนื่องจำนง ผู้เชี่ยวชาญเรดาร์ตรวจอากาศสถานี อุตุนิยามวิทยาจังหวัดพิษณุโลก <ul style="list-style-type: none">▪ หลักการทฤษฎีเรดาร์ตรวจอากาศ▪ ผลกระทบจากข้อมูลเรดาร์▪ หลักการพยากรณ์เพื่อการเตือนภัยด้วยเรดาร์ตรวจอากาศ▪ ตัวอย่างกรณีศึกษาจากภาคปฏิบัติ
10.45-11.00 น.	พักรับประทานอาหารว่าง
11.00-12.00 น.	การบรรยายเรื่อง การตรวจสภาพอากาศด้วยเรดาร์ตรวจอากาศภาคพื้นดิน วิทยากร: คุณนนท์ธวัช เนื่องจำนง <ul style="list-style-type: none">▪ การปฏิบัติงานและกรณีศึกษาตัวอย่างภาคเหนือตอนล่าง▪ การฝึกฝนการแจ้งเตือนภัย ภัยแล้ง ฝนจากร่องมรสุม ฝนจากพายุ พายุฤดูร้อน ลูกเห็บ พื้นที่ภาคเหนือตอนล่าง▪ ปัญหาและโจทย์งานวิจัยที่ต้องการพัฒนา (ถาม-ตอบอภิปราย)
12.00-13.00 น.	พักรับประทานอาหารกลางวัน
13.00-14.30 น.	การบรรยายเรื่อง งานกรมพัฒนาที่ดินกับสภาพอากาศ วิทยากร: คุณกชกร กานตารัมภ์ นักสำรวจดินชำนาญการพิเศษ สำนักพัฒนาที่ดินเขต 8 <ul style="list-style-type: none">▪ งานในกรมพัฒนาที่ดินและส่วนที่เกี่ยวข้อง▪ หลักการและเครื่องมือสำรวจดิน▪ ตัวอย่างกรณีศึกษาจากภาคปฏิบัติ
14.30-14.45 น.	พักรับประทานอาหารว่าง
14.45-16.00 น.	การบรรยายเรื่อง งานกรมพัฒนาที่ดินกับสภาพอากาศ วิทยากร: : คุณกชกร กานตารัมภ์ <ul style="list-style-type: none">▪ สภาพอากาศที่มีผลต่องานสำรวจดินของกรมพัฒนาที่ดิน กรณีศึกษา▪ ตัวอย่างกรณีศึกษาจากภาคปฏิบัติ▪ ปัญหาและโจทย์งานวิจัยที่ต้องการพัฒนา (ถาม-ตอบอภิปราย)
16.00-16.30 น.	สรุปผลการฝึกอบรมและมอบประกาศนียบัตร

ภาคผนวก ง
ประวัตินักวิจัย (Curriculum Vitae)



ประวัติและผลงานทางวิชาการ [update June 13 , 2019]

	<p>Dr. Nattapon Mahavik</p> <ul style="list-style-type: none"> ▪ D.Sc. in Atmospheric Science, Kyoto University, 2015 , Japan [MEXT] ทุนรัฐบาลญี่ปุ่น ▪ M.Sc. in Photogrammetry and Geoinformatics, Stuttgart University of Applied Sciences, 2009, Germany [DAAD] ทุนรัฐบาลเยอรมัน ▪ ฑบ. ภูมิศาสตร์ (เกียรตินิยม), มหาวิทยาลัยเชียงใหม่, 2545
	<p>+66(55)962752, nataponm@nu.ac.th</p>
<p>Research interests</p>	<ul style="list-style-type: none"> ▪ Ground-based radar rainfall estimation ▪ Remote sensing in Urban Heat Island ▪ Rainfall estimation from satellite products

Awards and Honour

- 2018 Shunji Murai Award for best paper on Asian Conference on Remote Sensing 2018 for the title *“The convective cloud properties extraction from weather radar reflectivity during SONCA tropical storm over the Lower Northern Thailand”*
- 2010-2013 Japanese government scholarship (monbukagakusho: MEXT) ทุนนักศึกษาวิจัยปริญญาเอก
- 2007-2009 German Academic Exchange Service (DAAD) ทุนปริญญาโท
- 1999-2001 The Shell Company of Thailand Limited

Work experiences

Date	Position	Organization
25 May 2009 - Present	Lecturer responsible for Geography and Geoinformatics Curriculum	Department of Natural Resources and Environment, Naresuan University

Specialization skills

- Ground-based weather radar and Satellite rainfall product processing
- Fortran, Python Scripting Language, GrADS Scripting Language, NCL Scripting Language, R Scripting Language
- Linux systems (Centos, Ubuntu)
- ArcGIS, QGIS, ENVI, ERDAS IMAGINE, MATLAB
- Drone photogrammetry (DJI, Syma) for mapping

ผลงานทางวิชาการ

บทความวิจัย/บทความวิชาการที่ตีพิมพ์

ระดับนานาชาติ

1. **Mahavik, N.,** Tantanee, S., (2019). Developed Radar Quality Index of Mosaicked Weather Radars over The Chao Phraya River Basin, Thailand. *GMSARN International Journal*. (Accepted in Scopus)
2. **Mahavik, N.,** Tantanee, S., (2019). Spatio-Temporal Analysis of Convective Cloud Properties Deriving from Weather Radar Reflectivity during the Decaying Stage of Tropical Storm over the Lower Northern Thailand, *International Journal of Geoinformatics* (Accepted in Scopus).
3. **Mahavik, N.,** Tantanee, S., (2019). Precipitating clouds analysis based on the developed radar mosaic products over the Chao Phraya River basin (Submitted in Scopus).
4. **Mahavik, N.,** Tantanee, S., (2019). Convective systems observed by ground-based radar during seasonal march of Asian summer monsoon in the middle of Indochina Peninsula, *Engineering and applied science research* (Accepted in Scopus).
5. Seejata, K., Yodying, A., Wongthadam, T., **Mahavik, N.,** Tantanee, S., (2018). Assessment of flood hazard areas using Analytical Hierarchy Process over the Lower Yom Basin, Sukhothai Province. *Procedia Engineering*, 212, 340-347. (scopus)
6. **Mahavik, N.,** (2017). Spatial Seasonal Distribution of Climatological Precipitation over the Middle of the Indochina Peninsula. *Applied Environmental Research*, 39(3), 63-76.
7. **Mahavik, N.,** (2017). Extreme precipitation trends over the middle of Indochina Peninsula during the period from 1978-2007. *International Journal of Earth Sciences and Engineering*, 10(3), 595-603.
8. **Mahavik, N.,** (2017). Bias Adjustments of Radar Rainfall during Seasonal March of the Summer Monsoon in the Middle of Thailand. *International Journal of Applied Environmental Sciences*, 12(4), 577-594.

9. **Mahavik, N.,** T. Satomura, S. Shige, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, S. Baimoung (2014). Rainfall Pattern over the Middle of Indochina Peninsula during 2009–2010 Summer Monsoon. *Hydrological Research Letter*, 8, 57-63.
10. **Mahavik, N.,** T. Satomura, S. Baimoung (2013). Radar rainfall analysis in the middle of Indochina peninsula. *J. Disaster Res*, 8, 187-188. (SCOPUS)

ตีพิมพ์ในรายงานสืบเนื่องจากการประชุมวิชาการ (Proceedings) ระดับนานาชาติ

1. **Mahavik, N.,** Tantanee, S. 2018: *Quality Assessment of mosaicked weather radars over the Chao Phraya river basin, Thailand.* Proceeding of GMSARN Int. Conf. on Energy, Environment, and Development in GMS, SD44, 1-4.
2. **Mahavik, N.,** Tantanee, S. 2018: The convective cloud properties extraction from weather radar reflectivity during SONCA tropical storm over the Lower Northern Thailand. *Proceeding of Asian Conference on Remote Sensing 2018*, 3, 1604-1612.
3. **Mahavik, N.,** Tantanee, S. 2018: Comparison of spatial error structures between TRMM products and APHRODITE over the Indochina Peninsula *Proceeding of Asian Conference on Remote Sensing 2018*, 3, 1754-1763.
4. **Mahavik, N.,** T. Satomura, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2013: Rainfall pattern observed by radar calibrated by rain gauges in Indochina. *Proceeding of The Third International MAHASRI/HyARC Workshop on Asian Monsoon and Water Cycle*, 44-49.

ตีพิมพ์ในรายงานสืบเนื่องจากการประชุมวิชาการ (Proceedings) ระดับชาติ

1. มินตรา รอดจันทร์และ**รัฐพล มหาวิศ.** (2560). การวิเคราะห์ห้วงวันเริ่มต้นและวันสิ้นสุดของฤดูมรสุมในพื้นที่ภาคเหนือของประเทศไทย. เรื่องเติมการประชุมวิชาการทรัพยากรธรรมชาติและสิ่งแวดล้อม ครั้งที่ 2, 15 ธันวาคม 2560, มหาวิทยาลัยนเรศวร, หน้า 170-179.
2. ศศิธร ฉัตรสุตารัตน์และ**รัฐพล มหาวิศ.** (2560). การศึกษาและวิเคราะห์การปลดปล่อยก๊าซเรือนกระจกในพื้นที่ปลูกอ้อย โดยเทคนิคการจำแนกเชิงวัตถุจากข้อมูลภาพถ่ายดาวเทียม Landsat-8 กรณีศึกษา อำเภอไทรงาม จังหวัดกำแพงเพชร. เรื่องเติมการประชุม

- วิชาการทรัพยากรธรรมชาติ สารสนเทศภูมิศาสตร์ และสิ่งแวดล้อมนเรศวร ครั้งที่ 2, 15 ธันวาคม 2560, มหาวิทยาลัยนเรศวร, หน้า 72-82.
3. อมินตรา เถลิ้มวัดและนัฐพล มหาวิค. (2560). การศึกษาและวิเคราะห์การตรวจพบไฟฟ้าของข้อมูลดาวเทียมกรณีศึกษาอุทยานแห่งชาติในเขตจังหวัดแม่ฮ่องสอนและเชียงใหม่: เรื่องเติมการประชุมวิชาการทรัพยากรธรรมชาติ สารสนเทศภูมิศาสตร์ และสิ่งแวดล้อมนเรศวร ครั้งที่ 2, 15 ธันวาคม 2560, มหาวิทยาลัยนเรศวร, หน้า 180-189.
 4. กมลฉัตร ศรีจะตะและนัฐพล มหาวิค. (2559). การศึกษาความหนาแน่นของเมฆจากข้อมูลแบบจำลองความสูงไลดาร์ : กรณีศึกษาเขตดาวทาวน์และเขตสตรีป รัฐเพนซิลวาเนีย ประเทศสหรัฐอเมริกา. เรื่องเติมการประชุมวิชาการทรัพยากรธรรมชาติ สารสนเทศภูมิศาสตร์ และสิ่งแวดล้อมนเรศวร ครั้งที่ 1, 3 พฤศจิกายน 2559, มหาวิทยาลัยนเรศวร, หน้า 158-165.
 5. เนตรนภา หงษ์ทองและนัฐพล มหาวิค. (2559). การศึกษาเปรียบเทียบเทคนิคการจำแนกเชิงจุดภาพของพื้นที่การเกษตรจากข้อมูลภาพถ่ายดาวเทียมไทยโชต : กรณีศึกษา อำเภอพร้าวกง จังหวัดเชียงใหม่. เรื่องเติมการประชุมวิชาการทรัพยากรธรรมชาติ สารสนเทศภูมิศาสตร์ และสิ่งแวดล้อมนเรศวร ครั้งที่ 1, 3 พฤศจิกายน 2559, มหาวิทยาลัยนเรศวร, หน้า 92-98.
 6. พิมพ์ผกา อ่องแก้วและนัฐพล มหาวิค. (2559). การศึกษาปฏิทินการเพาะปลูกพืชจากข้อมูลภาพถ่ายดาวเทียมหลายช่วงเวลา : กรณีศึกษา จังหวัดอุดรดิตถ์. เรื่องเติมการประชุมวิชาการทรัพยากรธรรมชาติ สารสนเทศภูมิศาสตร์ และสิ่งแวดล้อมนเรศวร ครั้งที่ 1, 3 พฤศจิกายน 2559, มหาวิทยาลัยนเรศวร, หน้า 85-91.
 7. ทับทิม วงศ์ทะคำและนัฐพล มหาวิค. (2559). การวิเคราะห์พื้นที่เสี่ยงน้ำท่วมโดยกระบวนการลำดับชั้นเชิงวิเคราะห์ บริเวณลุ่มน้ำยมตอนล่าง จังหวัดสุโขทัย. เรื่องเติมการประชุมวิชาการทรัพยากรธรรมชาติ สารสนเทศภูมิศาสตร์ และสิ่งแวดล้อมนเรศวร ครั้งที่ 1, 3 พฤศจิกายน 2559, มหาวิทยาลัยนเรศวร, หน้า 115-122.
 8. ศิริภัทร เอี่ยมละออและนัฐพล มหาวิค. (2559). พื้นที่เสี่ยงต่อการเกิดไฟฟ้าจากฐานข้อมูลตำแหน่งไฟฟ้าปี 2547-2557 : กรณีศึกษา อุทยานแห่งชาติน้ำหนาว. เรื่องเติมการประชุมวิชาการทรัพยากรธรรมชาติ สารสนเทศภูมิศาสตร์ และสิ่งแวดล้อมนเรศวร ครั้งที่ 1, 3 พฤศจิกายน 2559, มหาวิทยาลัยนเรศวร, หน้า 174-182.
 9. อภิษฐา ยอดยิ่งและนัฐพล มหาวิค. (2559). การเปรียบเทียบปริมาณน้ำฝนรายวันระหว่างผลิตภัณฑ์น้ำฝนจากข้อมูลดาวเทียมกับข้อมูลน้ำฝนภาคพื้นดิน พื้นที่ภาคเหนือของประเทศไทย. เรื่องเติมการประชุมวิชาการทรัพยากรธรรมชาติ สารสนเทศภูมิศาสตร์ และสิ่งแวดล้อมนเรศวร ครั้งที่ 1, 3 พฤศจิกายน 2559, มหาวิทยาลัยนเรศวร, หน้า 99-106.

การนำเสนอภาคบรรยาย (Oral presentation)

1. Mahavik, N., Tantanee, S. 2018: Quality Assessment of Mosaicked Weather Radars Over the Chao Phraya River Basin, Thailand. *The Grand GMSARN International Conference 2018 on "Energy, Environment, and Development in GMS"*, 28-30 November 2018 at Ramada Encore Hotel Nanning, Nanning City, Guangxi Province, China.
2. Mahavik, N., Tantanee, S. 2018: The convective cloud properties extraction from weather radar reflectivity during SONCA tropical storm over the Lower Northern Thailand. *Asian Conference on Remote Sensing 2018*, 15-19 October 2018, Renaissance Hotel Kuala Lumpur, Kuala Lumpur, Malaysia.
3. Mahavik, N., Tantanee, S. 2018: Comparison of spatial error structures between TRMM products and APHRODITE over the Indochina Peninsula. *Asian Conference on Remote Sensing 2018*, 15-19 October 2018, Renaissance Hotel Kuala Lumpur, Kuala Lumpur, Malaysia.
4. Seejata, K., Yodying, A., Wongthadam, T., Mahavik, N., Tantanee, S., (2017). Assessment of flood hazard areas using Analytical Hierarchy Process over the Lower Yom Basin, Sukhothai Province. 7th International Conference on Building Resilience Using scientific knowledge to inform policy and practice in Disaster Risk Reduction 27-29 November, 2018- Swissotel Le Concorde, Bangkok, Thailand.
5. Mahavik, N., S. Shige, T. Hayashi, M. K. Yamamoto, 2015: Orographic propagating rain systems over the middle of Indochina observed by gauge-calibrated ground-based radar data. MAHASRI workshop, 4 March 2015, Nagoya, Japan.
6. Mahavik, N., T. Satomura, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2013: Calibration of rainfall observed by

weather radars and rainfall pattern study in Indochina peninsular. IMPAC-T final workshop, 11-12 November 2014, Sendai, Japan.

7. Mahavik, N., T. Satomura, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2013: Rainfall pattern in the middle of Indochina Peninsular during 2009-2010 summer monsoon. IGU regional conference, 4-9 August 2013, Kyoto, Japan.
8. Mahavik, N., T. Satomura, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2013: Rainfall pattern in the middle of Indochina peninsular using Vientiane radar. MAHASRI/HyARC Workshop, 14 March 2013, Nagoya, Japan.
9. Mahavik, N., T. Satomura, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2013: Rainfall pattern in the middle of Indochina peninsular. IMPAC-T workshop, 28-29 January 2013, Bangkok, Thailand.

การนำเสนอภาคโปสเตอร์ (Poster presentation)

1. Mahavik, N., T. Satomura, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2013: Rainfall Estimated by Radar Reflectivity Calibrated by Rain Gauges and Rainfall Patterns in Indochina. GCOE final symposium, 2-3 December 2013, Kyoto, Japan.
2. Mahavik, N., T. Satomura, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2013: Application of Ground Based Radar and TRMM Rainfall for Hydrological Analysis, Part of Calibration of rainfall observed by weather radars and rainfall pattern study in Indochina peninsular. IMPAC-T final workshop, 11-12 November 2013, Sendai, Japan.
3. Mahavik, N., T. Satomura, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2013: Rainfall pattern in the middle of

- Indochina Peninsular during 2009-2010 summer monsoon. DACA-13, 8-12 July 2013, Davos, Switzerland.
4. Mahavik, N., T. Satomura, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2013: Rainfall pattern in the middle of Indochina Peninsula during 2009-2010 summer monsoon. JPGU 2013, 19-24 May 2013, Chiba, Japan.
 5. Mahavik, N., T. Satomura and S. Baimoung, 2012: Z-R parameter variations using conventional weather radar in the middle of Indochina peninsular. Meteorological Society of Japan (MSJ), 3-6 October 2012, Hokkaido, Japan.
 6. Mahavik, N., T. Satomura, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2012: Radar rainfall analysis in the middle of Indochina peninsular. International symposium on GCOE ARS, 3-4 August 2012, Kyoto, Japan.
 7. Mahavik, N., T. Satomura and S. Baimoung, 2012: Calibration of weather radar for analyzing precipitation distribution in Thailand. Meteorological Society of Japan (MSJ), 25-28 May 2012, Tsukuba, Japan.
 8. Mahavik, N., T. Satomura and S. Baimoung, 2011: Investigation ZR relationship by using weather radar in Thailand. IMPAC-T Joint conference with SEA water 9th conference, 1-3 December 2012, Bangkok, Thailand.
 9. Mahavik, N., T. Satomura, B. Sysouphanthavong, S. Phonevilay, M. Wakabayashi, and S. Baimoung, 2011: Precipitation characteristics in Thailand using weather radars and surface meteorological observations. Meteorological Society of Japan (MSJ), 16-19 May 2011, Tokyo, Japan.

ประสบการณ์ในงานวิจัย

1. 2558-2559 (หัวหน้าโครงการ) โครงการการพัฒนากระบวนการทำแผนที่ข้อมูลน้ำฝนเชิงเลขจากเรดาร์ตรวจอากาศภาคพื้นดินด้วยซอฟต์แวร์ระบบภูมิสารสนเทศรหัสเปิด ทุนพัฒนาบุคลากร มหาวิทยาลัยนเรศวร (งบประมาณ 220,000 บาท)
2. 2560-2561(หัวหน้าโครงการ) โครงการการพัฒนาวิธีการทำแผนที่โมเสคน้ำฝนประมาณค่าจากข้อมูลเรดาร์ตรวจอากาศภาคพื้นดินบริเวณลุ่มน้ำเจ้าพระยา ทุนงบประมาณแผ่นดิน วช. มหาวิทยาลัยนเรศวร (งบประมาณ 583,000 บาท)
3. 2560-2561(หัวหน้าโครงการ) โครงการความผันแปรโครงสร้างกายภาพระบบฝนจากข้อมูลดาวเทียม TRMM บริเวณอินโดจีนตอนกลาง ทุนงบประมาณแผ่นดิน วช. มหาวิทยาลัยนเรศวร (งบประมาณ 599,500.00)
4. 2562-2563(หัวหน้าโครงการ) โครงการการวิเคราะห์รูปแบบฝนบนพื้นฐานข้อมูลภูมิสารสนเทศเรดาร์ตรวจอากาศภาคพื้นดินรายละเอียดสูงในลุ่มน้ำยมและน่าน ทุนงบประมาณแผ่นดิน วช. มหาวิทยาลัยนเรศวร (งบประมาณ 460,000 บาท)
5. 2560-2564(ผู้ร่วมวิจัย) โครงการ Advancing Co-design of Integrated Strategies with Adaptation to Climate Change in Thailand (ADAP-T) สนับสนุนโดย Science and Technology Research Partnership for Sustainable Development (SATREPS), JST-JICA (6,000,000 เยน)
6. 2553-2557 (ผู้ร่วมวิจัย) โครงการ Integrated Study Project on Hydro-Meteorological Prediction and Adaptation to Climate Change in Thailand (IMPAC-T) สนับสนุนโดย JICA/JST
7. 2559-2561 (ผู้ร่วมวิจัย) โครงการศึกษาความเป็นธรรมทางสุขภาพ สำนักงานกองทุนสนับสนุนการสร้างเสริมสุขภาพ (สสส.)

การเข้ารับฝึกอบรม (Academic training)

1. Aerosol Modeling Workshop organized by National Astronomical Research Institute of Thailand (NARIT), University of Phayao (UP) & Hydro and Agro Informatics Institute (HAI), , February 19 – 21, 2019, Kantary Hills Hotel, Chiang Mai, Thailand.
2. Lidar data for biomass estimation by Silva Carbon supported by USAID, 5-11 February 2017, The Bazaar Hotel, Bangkok, Thailand

3. Landuse and Landcover change using remote sensing by SERVIR Mekong and ADPC supported by USAID, October 2015, Pak Nai Lert Hotel, Bangkok, Thailand.
4. Capacity building and brainstorming by SERVIR Mekong and ADPC supported by USAID, November 2015, Pak Nai Lert Hotel, Bangkok, Thailand.
5. IHP 22nd Precipitation measured by Satellite, 18 November – 1 December 2012 at Nagoya university, Nagoya, Japan.
6. Climate change downscaling approaches and applications, 9-14 November 2011 at United Nations University, Tokyo, Japan.
7. TRMM and GSMaP training, 29 August-2 September 2011, by JAXA at Kyoto university and visited JAXA, Kyoto-Chiba-Tokyo, Japan

