

เทคนิคการปรับแต่งภาพลายเส้น

สัญชัย น้อยจันทร์ และ นพพร โชติกกำธร

คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

Email: sanchai.modnoi@gmail.com, nopporn@it.kmitl.ac.th

บทคัดย่อ

บทความนี้นำเสนอเกี่ยวกับการปรับแต่งลายเส้นของภาพที่วาดด้วยมือ ซึ่งเป็นกระบวนการที่ปรับเปลี่ยนรูปร่างลักษณะของภาพลายเส้นที่ถูกวาดด้วยอุปกรณ์วาดเส้นอิเล็กทรอนิกส์ให้มีความสวยงาม ถูกต้อง หรือตรงตามความต้องการมากยิ่งขึ้น โดยจะอธิบายเปรียบเทียบความแตกต่างระหว่างการปรับแต่งลายเส้น การรู้จำลายเส้น และการสังเคราะห์ลายเส้น และยังได้อธิบายพื้นฐานการเก็บข้อมูลลายเส้นแบบดิจิทัล พร้อมรวบรวมแนวทางหลักๆ ในการปรับปรุงลายเส้นที่มีการใช้งานกันอยู่ในปัจจุบัน สรุปมานำเสนอในบทความนี้ 3 แนวทาง ได้แก่ การใช้การปรับปรุงลายเส้นระดับล่าง การเฉลี่ยลายเส้นที่ซ้ำซ้อนกัน และการกำหนดขอบเขตของลายเส้น นอกจากนี้ยังได้ยกตัวอย่างเทคนิคที่เกี่ยวข้องกับการปรับปรุงลายเส้น อันได้แก่ การทดสอบความเป็นรูปปิด, พื้นที่ความคลาดเคลื่อน และการค้นหาความสมมาตร เพื่อให้เป็นแนวทางในการศึกษาค้นคว้าต่อยอดด้วย

คำสำคัญ – sketch beautification; sketch recognition; handwriting beautification; computer graphics

Abstract

This paper reviews techniques for sketch beautification, a process of beautifying strokes drawn from electronic drawing device to meet the intention of a creator. The paper discusses the difference between sketch beautification, sketch recognition, and sketch synthesis. Three means of sketch beautification, namely low-level sketch recognition, repeated stroke averaging, and sketch domain scoping, are next described. Additionally, other novel and interesting techniques related to sketch beautification, such as close-shape testing, feature area, and symmetry detection, are briefly discussed for guiding future study.

Keywords – sketch beautification; sketch recognition; handwriting beautification; computer graphics

1. บทนำ

การวาดและเขียนเป็นวิธีการที่มนุษย์ใช้บันทึกข้อมูลและสื่อสารความคิด มีการเริ่มใช้งานกันมาช้านาน ที่พบเห็นได้มากมักจะอยู่ในรูปของการวาดเขียนลงบนกระดาษหรือวัสดุอื่นๆ แต่ปัญหาของวิธีการนี้มีอยู่ว่า วัสดุดังกล่าวนั้น

ส่วนมากมักจะมีขนาดไม่เล็กพอที่จะทำให้จัดเก็บได้อย่างสะดวก และหากข้อมูลเพิ่มปริมาณมากขึ้น ก็ต้องใช้วัสดุเหล่านั้นมากขึ้น นั่นหมายถึงการที่ต้องมีพื้นที่จัดเก็บมากขึ้น นอกจากนี้วิธีการดังกล่าวก็ยังไม่เอื้ออำนวยต่อการสำเนาข้อมูลและนำข้อมูลไปใช้ได้อีก อีกทั้งตัววัสดุยังมี

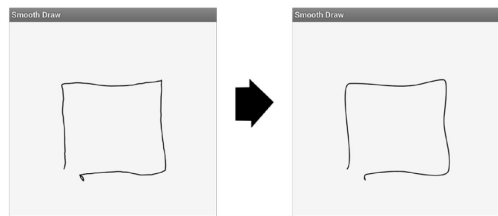
โอกาสที่จะเสื่อมสลายหรือชำรุดได้ง่ายอีกด้วย ในปัจจุบัน การเก็บข้อมูลในรูปแบบดิจิทัลเข้ามามีบทบาทสำคัญ ปัญหาต่างๆ ดังที่กล่าวมาแล้วนั้นจึงได้รับการแก้ไข การวาดและเขียนเพื่อบันทึกข้อมูลและสื่อสารในรูปแบบดิจิทัลช่วยลดพื้นที่ในการจัดเก็บข้อมูลลงได้หลายเท่าตัวเมื่อเปรียบเทียบกับวิธีการเดิม สามารถนำข้อมูลไปสำเนาได้อย่างไม่จำกัดด้วยค่าใช้จ่ายที่ต่ำมาก จึงสามารถทำสำเนาได้มาก ป้องกันปัญหาสื่อที่เก็บข้อมูลชำรุดหรือเสื่อมสภาพ และข้อมูลยังอยู่ในรูปแบบที่สามารถนำไปประยุกต์ใช้งานต่อได้สะดวกอีกด้วย

เมื่อปัญหาในระดับแรกได้รับการแก้ไขแล้ว ประเด็นปัญหาในระดับถัดไปจึงถูกนำมาพิจารณาแทน ปัญหาที่จะหยิบยกมาอธิบายในบทความนี้ได้แก่ 1) คอมพิวเตอร์จะช่วยให้การวาดและเขียนสวยงามหรือมีรูปลักษณะที่ดี (ตรงตามความต้องการ) มากขึ้นได้อย่างไร ด้วยปัญหานี้จึงทำให้เกิดการศึกษาวิจัยในเรื่อง **การปรับปรุงลายเส้น (sketch beautification)** ขึ้นมา 2) คอมพิวเตอร์จะสามารถรับรู้และเข้าใจสิ่งที่มนุษย์วาดและเขียนได้อย่างไร ด้วยปัญหานี้จึงทำให้เกิดการศึกษาวิจัยในเรื่อง **การรู้จำลายเส้น (sketch recognition)** ขึ้นมา และ 3) คอมพิวเตอร์จะช่วยวาดและเขียนแทนมนุษย์ได้อย่างไร ด้วยปัญหานี้จึงทำให้เกิดการศึกษาวิจัยในเรื่อง **การสังเคราะห์ลายเส้น (sketch synthesis)** ขึ้นมา

ในบทความนี้จะมุ่งประเด็นไปที่การปรับปรุงลายเส้นอธิบายถึงความแตกต่างระหว่างการปรับปรุงลายเส้นและกระบวนการเกี่ยวกับลายเส้นอื่นๆ นำเสนอแนวทางการปรับปรุงลายเส้น รวมไปถึงตัวอย่างเทคนิคต่างๆ ที่เกี่ยวข้อง

2. การปรับปรุงลายเส้น การรู้จำลายเส้น และการสังเคราะห์ลายเส้น

การปรับปรุงลายเส้น คือการที่คอมพิวเตอร์พยายามปรับปรุงลายเส้นให้มีลักษณะที่สวยงามหรือดูดี (ตรงตาม

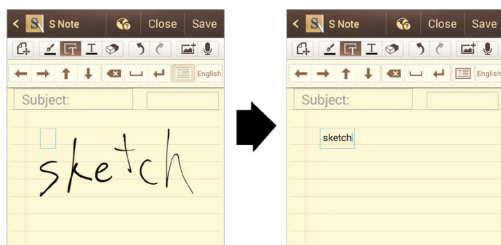


รูปที่ 1. หน้าจอแสดงฟังก์ชันการปรับปรุงลายเส้นในโปรแกรม Smooth Draw

ความต้องการ) ยิ่งขึ้น โดยมีความเป็นไปได้ว่าคอมพิวเตอร์อาจจะต้องรู้หรือไม่รู้เลยว่าลายเส้นนั้นๆ คือรูปหรือตัวอักษรอะไรก็ได้ ตัวอย่างของการนำเอาการปรับปรุงลายเส้นไปประยุกต์ใช้งานเช่น การปรับเส้นที่ผู้ใช้วาดซึ่งมีลักษณะขยุกขยิกให้มีความเรียบเนียน อย่างเช่นฟังก์ชันการวาดในโปรแกรม Smooth Draw โดย Ashim Prasad ที่จะช่วยปรับเส้นที่ผู้ใช้วาดให้มีความเรียบเนียน มีความโค้งมนบริเวณรอยหัก (ดูรูปที่ 1)

ปัญหาสำคัญของการปรับปรุงลายเส้นก็คือ ทำอย่างไรจึงจะสามารถปรับปรุงลายเส้นได้ตรงตามความต้องการของผู้ใช้ และรักษารายละเอียดที่สำคัญเอาไว้ได้อย่างครบถ้วนมากที่สุด รวดเร็วที่สุด และใช้ทรัพยากรน้อยที่สุด และนี่ก็เป็นโจทย์ที่นักค้นคว้าวิจัยพยายามค้นหาคำตอบ

การรู้จำลายเส้น คือการที่คอมพิวเตอร์พยายามที่จะวิเคราะห์ลายเส้น เพื่อรับรู้ว่าลายเส้นนั้นคือรูปอะไร โดยการพยายามที่จะรับรู้จะต้องอาศัยความรู้ทั้งในด้านปัญญาประดิษฐ์ (artificial intelligence) และการปฏิสัมพันธ์ระหว่างคอมพิวเตอร์กับมนุษย์ (human computer interaction) สิ่งที่จะวัดความสำเร็จของการรู้จำการวาดนั้นก็คือ ความถูกต้องของผลการรู้จำเมื่อเปรียบเทียบกับสิ่งที่ผู้วาดตั้งใจจะวาดจริงๆ ตัวอย่างของการนำการรู้จำลายเส้นมาประยุกต์ใช้งานก็คือ การวิเคราะห์ลายเส้นที่ผู้ใช้วาด แล้วแทนที่ลายเส้นนั้นด้วยตัวอักษร อย่างเช่นฟังก์ชันการรู้จำตัวอักษรในโปรแกรม S Note เมื่อผู้ใช้เขียนตัวหนังสือด้วยลายมือเสร็จ โปรแกรม



รูปที่ 2. หน้าจอแสดงฟังก์ชันการรู้จำลายเส้นในโปรแกรม S Note

จะพยายามวิเคราะห์ลายเส้นว่าเป็นตัวอักษรอะไร แล้วแทนที่ลายเส้นเหล่านั้นด้วยตัวอักษรแบบตัวพิมพ์แทน (ดูรูปที่ 2)

ปัญหาสำคัญของการรู้จำลายเส้นก็คือ ทำอย่างไรให้ผลการรู้จำลายเส้นมีความถูกต้อง แม่นยำ และรวดเร็วมากที่สุด แต่สิ้นเปลืองทรัพยากรน้อยที่สุด และนี่ก็เป็นโจทย์ที่นักค้นคว้าวิจัยพยายามค้นหาคำตอบ

การสังเคราะห์ลายเส้น เป็นการสร้างวัตถุใหม่ขึ้นมาโดยการรับคำสั่งหรือตัวแปรบางอย่างที่บอกว่าจะสร้างวัตถุอะไร คำสั่งหรือตัวแปรเหล่านี้ได้มาจากข้อมูลลายเส้น ซึ่งอาจจะอาศัยการรู้จำลายเส้นเข้ามาช่วยเพื่อที่จะได้รับรู้วาลายเส้นนั้นคือรูปหรือตัวอักษรอะไร จากนั้นจึงสังเคราะห์วัตถุใหม่ขึ้นมาตามผลลัพธ์ของการรู้จำลายเส้นนั่นเอง ยกตัวอย่างการประยุกต์ใช้งานจากรูปที่ 2 หลังจากที่ใช้วาดเส้นด้วยลายมือ โปรแกรมจะใช่วิธีการทางการรู้จำลายเส้นเพื่อวิเคราะห์ว่าผู้ใช้งานวาดเส้นอะไร จากนั้นจึงเข้าสู่ขั้นตอนการสังเคราะห์ลายเส้น นั่นคือการสร้างตัวอักษรในรูปแบบของตัวพิมพ์ขึ้นมาใหม่ แทนที่ลายเส้นที่วาดโดยผู้ใช้งาน

โดยตัวการสังเคราะห์ลายเส้นเองนั้นไม่ค่อยมีปัญหาสำคัญอะไร เนื่องจากการสังเคราะห์ลายเส้นจะรับคำสั่งหรือตัวแปรที่ระบุว่าจะสังเคราะห์อะไรแล้วสร้างวัตถุตามนั้น ปัญหาสำคัญจึงตกไปอยู่ที่กระบวนการที่ทำให้ได้มาซึ่งคำสั่งหรือตัวแปรดังกล่าวนั่นเอง

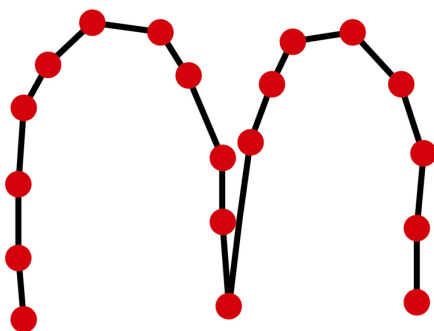
สิ่งที่แตกต่างกันอย่างชัดเจนระหว่างการรู้จำลายเส้น การปรับปรุงลายเส้น และการสังเคราะห์ลายเส้นก็คือ เป้าหมาย การรู้จำลายเส้นมีเป้าหมายหลักคือเพื่อรับรู้วาลายเส้นเป็นรูปหรือตัวอักษรอะไร เป้าหมายหลักของการปรับปรุงลายเส้นคือการทำให้ลายเส้นมีรูปลักษณะที่ดีขึ้น หรือตรงตามความต้องการมากขึ้น ส่วนเป้าหมายหลักของการสังเคราะห์ลายเส้นคือการสร้างรูปร่างขึ้นมาตามคำสั่งหรือตัวแปรที่ได้รับ ซึ่งอาจได้มาจากกระบวนการรู้จำลายเส้นอีกทีหนึ่งก็เป็นได้ ถึงแม้ว่าเป้าหมายของแต่ละกระบวนการจะต่างกัน แต่รายละเอียดทางเทคนิคในการสร้างกระบวนการแต่ละกระบวนการขึ้นมา อาจมีบางส่วนที่ใช้เทคนิควิธีเดียวกัน หรืออาศัยผลลัพธ์จากกระบวนการอื่นๆ เข้ามาช่วยด้วยก็เป็นได้ ท้ายที่สุดแล้วทุกกระบวนการก็จะมีจุดประสงค์เดียวกันคือการทำให้ลายเส้นที่ผู้ใช้ป้อนเข้าสู่ระบบถูกนำไปทำให้อยู่ในรูปที่สามารถนำไปใช้ประโยชน์ต่อได้ดีกว่าเดิม

3. ความรู้เบื้องต้นเกี่ยวกับข้อมูลลายเส้น

ข้อมูลลายเส้น หรือที่เรียกกันว่า ink data ในที่นี้คือข้อมูลที่ถูกบันทึกจากอุปกรณ์วาดเส้นแบบดิจิทัล เช่น อุปกรณ์จำพวกแท็บเล็ตและสมาร์ตโฟนต่างๆ โดยผู้ใช้งานวาดเส้นผ่านหน้าจอหรือพื้นผิวสัมผัสสำหรับรับข้อมูลของตัวอุปกรณ์ด้วยปากกาดิจิทัล (stylus) หรือนิ้วมือ

หลักการทำงานของอุปกรณ์วาดเส้นดังที่กล่าวมาก็คือ เมื่อผู้ใช้งานจรดปากกาลงบนพื้นผิวสัมผัสสำหรับรับข้อมูล (surface) ของตัวอุปกรณ์ แล้วลากปากกาเพื่อให้เกิดเป็นลายเส้น อุปกรณ์จะทำการเก็บบันทึกตำแหน่งที่หัวปากกาสัมผัสกับพื้นผิวสัมผัสสำหรับรับข้อมูล และอาจจะมีการบันทึกข้อมูลอื่นๆ ที่เกี่ยวข้องด้วย เช่น แรงกดปากกา (pressure) ความเอียงของปากกา (tilt) เป็นต้น โดยข้อมูลเหล่านี้จะถูกบันทึกทุกๆ ช่วงเวลาหนึ่ง ข้อมูลที่ถูกบันทึกแต่ละครั้งจะถูกเรียกว่า sample (แสดงด้วยจุดวงกลมในรูปที่ 3) อัตราความถี่ของการบันทึกข้อมูลจะ

เรียกว่า sample rate เช่น บันทึกข้อมูล 100 ครั้งในทุกๆ วินาที ก็หมายความว่า sample rate เป็น 100 เฮิรตซ์ (hertz ย่อเป็น Hz) โดยอุปกรณ์ที่ต่างกันก็อาจจะมีอัตรา sample rate ที่แตกต่างกันไป ยิ่งอุปกรณ์ที่มีอัตรา sample rate ที่สูง ก็จะทำให้ได้ข้อมูลที่มีความละเอียดมากยิ่งขึ้น แต่ถ้าอัตรา sample rate สูงเกินไปก็อาจทำให้ได้ข้อมูลที่ซ้ำซ้อนเกินความจำเป็น ข้อมูลที่ถูกเก็บตั้งแต่ผู้ใช้เริ่มจรดปากกาถึงสัมผัสกับพื้นผิวสัมผัสสำหรับรับข้อมูล ไปจนกระทั่งผู้ใช้ยกปากกาขึ้นจากผิวสัมผัสสำหรับรับข้อมูล รวมกันจะทำให้ได้ลายเส้น 1 เส้น เรียกลายเส้นนี้ว่า stroke (แสดงด้วยเส้นดำในรูปที่ 3)



รูปที่ 3. เส้น stroke และจุด sample

4. แนวทางการปรับปรุงลายเส้น

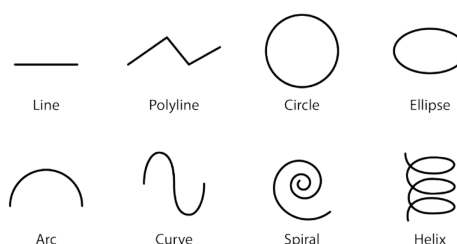
4.1 การรู้จำลายเส้นระดับล่าง (low-level sketch recognition)

ถึงแม้ว่าอย่างที่ได้อธิบายไปแล้วว่าการปรับปรุงลายเส้นอาจจะไม่จำเป็นต้องอาศัยการรู้จำลายเส้นก็ได้ แต่การใช้การรู้จำลายเส้นระดับล่างก็เป็นอีกแนวทางหนึ่งที่สามารถนำมาใช้ให้เป็นประโยชน์ต่อการปรับแต่งลายเส้นได้ การรู้จำลายเส้นระดับล่างนี้หมายถึงการวิเคราะห์เพื่อรับรู้ว่าลายเส้นหรือส่วนหนึ่งส่วนใดของลายเส้นเป็นรูปร่างพื้นฐานหรือไม่ เช่น วงกลม วงรี เส้นโค้ง สี่เหลี่ยม เป็นต้น ถ้าเราสามารถรู้ได้ว่าลายเส้นหรือส่วนหนึ่งส่วนใดของ

ลายเส้นเป็นรูปร่างพื้นฐานอะไร เราก็จะสามารถปรับลายเส้นให้มีลักษณะเหมือนหรือใกล้เคียงกับรูปร่างพื้นฐานมากขึ้นกว่าเดิม แต่ยังคงพยายามรักษารูปร่างและขนาดให้ใกล้เคียงกับลายเส้นต้นฉบับมากที่สุด

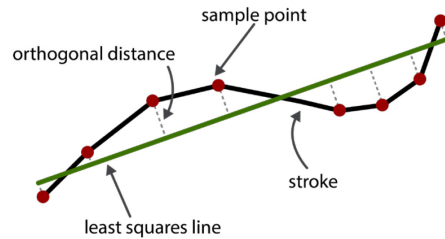
ในปัจจุบันมีการคิดค้นวิธีการต่างๆ ขึ้นมาเพื่อใช้สำหรับการรู้จำรูปร่างพื้นฐาน ยกตัวอย่างเช่น ใน PaleoSketch [1] สามารถรู้จำรูปร่างพื้นฐานได้ทั้งหมด 8 แบบ ได้แก่ เส้นตรง เส้นที่ประกอบด้วยเส้นตรงเชื่อมต่อกัน (polyline) วงกลม วงรี ส่วนของวงกลม (arc) เส้นโค้ง (curve) เส้นวงก้นหอย (spiral) และเส้นเกลียว (helix) (ดูภาพที่ 4) โดยขั้นตอนการทำงานคร่าวๆ ของ PaleoSketch จะเริ่มจากขั้นตอน pre-recognition ซึ่งเป็นขั้นตอนของการจัดเตรียมเส้น stroke ให้พร้อมสำหรับขั้นตอนถัดๆ ไป ในขั้นตอนนี้โปรแกรมจะทำการลบจุด sample ที่ซ้ำซ้อนออกเพื่อประสิทธิภาพในการประมวลผล และมีการลบเส้นปลายทั้งสองด้านของเส้น stroke ที่คาดว่าจะเกิดจากความไม่ได้ตั้งใจ เนื่องจากโดยปกติบริเวณปลายเส้นทั้งสองด้านมักจะมียอยหยักที่ผู้วาดไม่ได้ตั้งใจให้เกิดขึ้น เรียกว่า noise ซึ่งอาจส่งผลให้การรู้จำผิดพลาดหรือคลาดเคลื่อนได้ และนอกจากนี้โปรแกรมก็จะคำนวณค่าต่างๆ ที่จะถูกนำไปใช้ต่อในขั้นตอนถัดๆ ไปเตรียมไว้อีกด้วย

ต่อจากนั้นก็เข้าสู่ขั้นตอนการทดสอบรูปร่าง (shape tests) โปรแกรมจะทำการทดสอบเส้น stroke ว่า

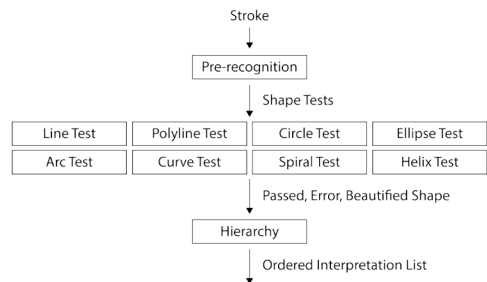


รูปที่ 4. รูปร่างพื้นฐานทั้งหมดที่ PaleoSketch สามารถรู้จำได้

ตรงกับรูปร่างใดบ้างจาก 8 รูปร่างที่โปรแกรมรองรับ และโปรแกรมก็จะให้ผลลัพธ์ของลายเส้นที่ถูกปรับปรุงแล้วออกมาด้วย ยกตัวอย่างการทดสอบ line test คือทดสอบว่าลายเส้นเป็นเส้นตรงหรือไม่ โปรแกรมจะใช้วิธีหา least squares line คือเส้นตรงที่มีตำแหน่ง ทิศทาง และความยาวใกล้เคียงกับลายเส้นมากที่สุดโดยที่ยังคงความเป็นเส้นตรงสมบูรณ์แบบเอาไว้ (แสดงด้วยเส้นสีตรงยาวในรูปที่ 5) จากนั้นจึงหา orthogonal distance (ใช้วิธีการเดียวกันกับใน [2]) ซึ่งก็คือผลรวมของระยะห่างระหว่างจุด sample แต่ละจุดบนเส้น stroke ไปยังเส้น least squares line (แสดงระยะห่างดังกล่าวด้วยเส้นประสีเทาในรูปที่ 5) นำค่าผลรวมที่ได้ไปหารด้วยความยาวของเส้น stroke ก็จะได้ค่าความคลาดเคลื่อนออกมา หากค่าความคลาดเคลื่อนนี้ไม่เกินค่าที่กำหนดเอาไว้ (ในกรณีของ PaleoSketch คือ 2.0) ก็จะผ่านเข้าสู่ขั้นตอนถัดไป นั่นก็คือการหา feature area ตามวิธีการใน [3] (สามารถอ่านเพิ่มเติมได้จากหัวข้อ 5.2 พื้นที่ความคลาดเคลื่อน (feature area)) นำค่า feature area ที่ได้มาหารด้วยความยาวของเส้น stroke ก็จะได้ค่าความคลาดเคลื่อนออกมา หากค่าความคลาดเคลื่อนไม่เกินค่าที่กำหนดไว้ (ในกรณีของ PaleoSketch คือ 10.25) ก็จะผ่านเข้าสู่ขั้นตอนสุดท้ายคือการทดสอบว่าเส้น stroke นั้นมีการหมุนวน (overtraced) หรือไม่ โดยใช้วิธีการเดียวกันกับใน [4] ถ้าหากค่า rotation (อธิบายไว้ในหัวข้อ 5.1 การทดสอบความเป็นรูปปิด) ที่ได้ไม่เกินค่าที่กำหนดไว้ (ในกรณีของ PaleoSketch คือ 1.31) ก็จะถือว่าผ่านการทดสอบ ก็จะสามารถสรุปได้ว่าเส้น stroke นี้เป็นเส้นตรง ผลของการทดสอบรูปร่างทั้งหมดจะถูกนำมาจัดเรียงในขั้นตอน hierarchy เพื่อหาผลที่มีความเป็นไปได้มากที่สุด และสรุปออกมาเป็นผลการรู้จำ ดูแผนผังแสดงขั้นตอนการทำงานของ PaleoSketch ได้จากรูปที่ 6 แต่ในมุมมองของการปรับปรุงลายเส้นนั้น สิ่งที่เราสนใจไม่ใช่ผลของการทดสอบว่าเส้น stroke เป็นรูปอะไร แต่หากเป็นวิธีการ



รูปที่ 5. ลายเส้นและเส้น least squares line

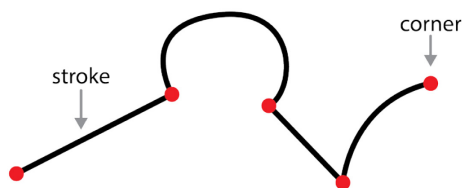


รูปที่ 6. ขั้นตอนการทำงานของ PaleoSketch (รูปจาก [1] หน้า 2)

สร้างรูปร่างสมบูรณ์แบบเทียบกับเส้น stroke โดยพยายามรักษาความสมบูรณ์แบบและรักษารูปร่างและขนาดให้ใกล้เคียงกับเส้น stroke เอาไว้ ซึ่งถือเป็นสาระสำคัญของการปรับปรุงลายเส้น

หากมองในระดับที่สูงขึ้นไป ในเส้น stroke หนึ่งเส้น นั้นมีความเป็นไปได้ว่าจะประกอบไปด้วยรูปร่างพื้นฐานมากกว่า 1 รูปก็เป็นได้ เพราะฉะนั้นจึงต้องมีการแบ่งเส้น stroke ออกเป็นเส้น stroke ย่อยๆ (segment) เพื่อจะแยกแยะรูปร่างพื้นฐานแต่ละรูปออกจากกัน ยกตัวอย่าง MergeCF [5] ซึ่งเป็นวิธีการค้นหาจุดมุม (corner) ของลายเส้น จุดมุมในที่นี้หมายถึงเส้นที่เป็นจุดเชื่อมระหว่าง 2 รูปร่างพื้นฐานย่อยในลายเส้น วิธีการของ MergeCF จะหาจุดมุมชุดแรกโดยอาศัยค่าความโค้ง (curvature) และค่าความเร็ว (speed) โดยตั้งสมมติฐานว่า บริเวณที่มีความโค้งมากๆ มีโอกาสเป็นไปได้สูงที่จะ

เป็นจุดมุม เนื่องจากอาจเป็นจุดที่ผู้วาดเปลี่ยนจากการวาดรูปร่างหนึ่งไปสู่อีกรูปร่างหนึ่ง และบริเวณที่มีความเร็วในการลากเส้นต่ำก็มีโอกาสเป็นไปได้สูงที่จะเป็นจุดมุมเช่นกัน เนื่องจากอาจมีการเปลี่ยนรูปร่างของเส้นที่จะวาดหรือเปลี่ยนทิศทางการวาด ผู้วาดจึงลดความเร็วการวาดลงจนหยุดนิ่งเพื่อเปลี่ยนทิศทาง (ดูรูปที่ 7 ประกอบ) แต่จุดมุมชุดแรกที่ได้มานี้อาจจะมีจุด sample ที่ไม่ใช่จุดมุมรวมมาด้วย (false positive) จึงต้องมีขั้นตอนการลบจุดที่ไม่ใช่จุดมุมออกจากผลการค้นหาจุดมุม ตามวิธีที่อธิบายไว้ใน [5] เมื่อเราทราบจุดมุม เราก็จะสามารถแบ่งเส้น stroke ออกเป็น segment และใช้วิธีการเกี่ยวกับการรู้จำลายเส้นกับ segment ได้ตามปกติ ซึ่งก็รวมไปถึงวิธีการเกี่ยวกับการปรับปรุงลายเส้นด้วย

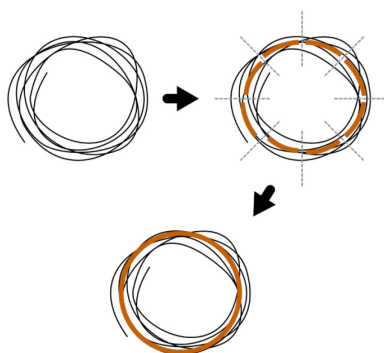


รูปที่ 7. จุดมุม (corner) ตามแนวของลายเส้น

4.2 การเฉลี่ยลายเส้นที่ซ้ำซ้อนกัน

ในบางครั้งผู้ใช้วาดเส้นด้วยการย่ำเส้น คือลากเส้นทับไปทับมาตรงบริเวณเดียวกัน เพราะในการลากเส้นครั้งแรกอาจจะได้เส้นที่ไม่ตรงตามความต้องการ จึงลากเส้นย่ำเพื่อให้ได้เส้นที่ตรงตามความต้องการมากขึ้นกว่าเดิม ในการปรับปรุงลายเส้นลักษณะนี้จึงต้องทำการเฉลี่ยลายเส้นทั้งหมดที่ผู้ใช้ตั้งใจให้เป็นเส้นเดียวกัน เพื่อหาตำแหน่งและรูปร่างของเส้นที่ผู้ใช้ต้องการ วิธีการที่เป็นไปได้นั้นก็มียหลายวิธี แต่สำหรับกรณีที่จะยกเป็นตัวอย่างนี้ จะใช้วิธีการแบ่งเส้น stroke ออกเป็นส่วนๆ จากนั้นใช้วิธีการทางคณิตศาสตร์บางอย่างเพื่อเฉลี่ยหาเส้นที่เป็นตัวแทนของส่วนของเส้น stroke นั้น จากนั้นจึงนำเส้นตัวแทนแต่

ละเส้นมาเฉลี่ยรวมเป็นเส้นเดียวกัน (ดูรูปที่ 8 ประกอบ) และนอกจากนี้ยังมีวิธีการปรับปรุงลายเส้นโดยอาศัยการเฉลี่ยลายเส้นเข้าด้วยกันในรูปแบบอื่นๆ อีก เช่น เฉลี่ยจากลายเส้นที่ผู้วาดตั้งใจวาดให้เป็นรูปเดียวกันใน [8] เฉลี่ยจากลายเส้นที่ซ้ำซ้อนกันโดยแบ่งลายเส้นที่ซ้ำซ้อนกันออกเป็นกลุ่มๆ ใน [9]

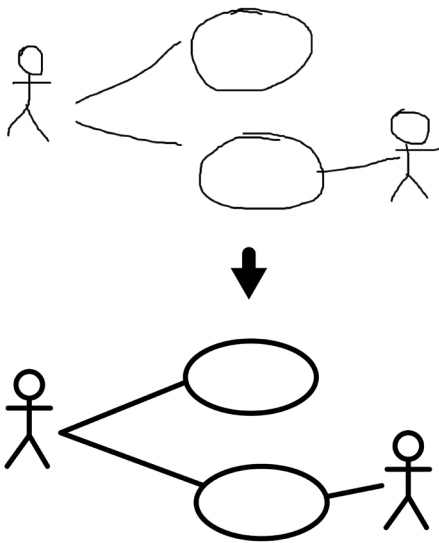


รูปที่ 8. ขั้นตอนการเฉลี่ยลายเส้นที่ซ้ำซ้อน

4.3 การกำหนดขอบเขตของลายเส้น

ในการปรับปรุงลายเส้น ถ้าเราไม่ทราบอะไรเลยเกี่ยวกับรูปที่ผู้ใช้วาดเลย โอกาสของการปรับปรุงลายเส้นที่ผิดพลาดก็จะเพิ่มมากขึ้น หรือถึงแม้จะถูกต้อง ก็อาจจะต้องใช้เวลาในการประมวลผลมาก แต่ถ้าหากมีการกำหนดขอบเขต (domain) ของสิ่งที่ผู้ใช้วาด โอกาสของการปรับปรุงลายเส้นที่ผิดพลาดก็จะมัน้อยลง และอาจจะทำให้ใช้เวลาในการประมวลผลน้อยลงด้วย อย่างเช่น ถ้าขอบเขตของการวาดคือ use case diagram เราก็จะทราบว่ารูปร่างทั้งหมดที่เป็นไปได้มีอะไรบ้าง มีลักษณะอย่างไร ถ้าเราค้นพบลายเส้นที่มีลักษณะเหมือนหรือใกล้เคียงกับสิ่งใดใน use case diagram ก็สามารถปรับลายเส้นให้อยู่ในรูปแบบที่ดีขึ้นและถูกต้องมากขึ้นได้ (ดูรูปที่ 9 ประกอบ)

วิธีการหนึ่งที่ถูกนำมาใช้เก็บคำอธิบายของวัตถุต่างๆ ที่เป็นไปได้ภายใต้ขอบเขตหนึ่งๆ นั่นก็คือ คำอธิบายขอบเขต (domain description) ซึ่งจะประกอบไปด้วย



รูปที่ 9. การปรับปรุงลายเส้นโดยมีขอบเขตเป็น use case diagram

คำอธิบายรูปร่าง (shape description) ยกตัวอย่าง LADDER [6] ซึ่งเป็นภาษาเชิงโครงสร้างที่ใช้สำหรับเก็บคำอธิบายเกี่ยวกับวัตถุและข้อกำหนดต่างๆ ของรูปร่างที่อยู่ภายใต้ขอบเขตใดขอบเขตหนึ่ง ด้านล่างนี้เป็นคำอธิบายรูปร่างสำหรับรูปลูกศร ในรายละเอียดจะระบุว่าลูกศรนี้จะประกอบไปด้วยรูปร่างย่อยๆ อะไรบ้าง และข้อกำหนดสำหรับรูปร่างย่อยแต่ละอันเป็นอย่างไร

```
(define shape Arrow
  (comment "An arrow with an open head.")
  (components
    (Line shaft)
    (Line head1)
    (Line head2))
  (constraints
    (coincident shaft.p1 head1.p1)
    (coincident shaft.p1 head2.p1)
    (equalLength head1 head2)
    (acuteMeet head1 shaft)
    (acuteMeet shaft head2))
  (aliases
    (Point head shaft.p2)
    (Point tail shaft.p1))
```

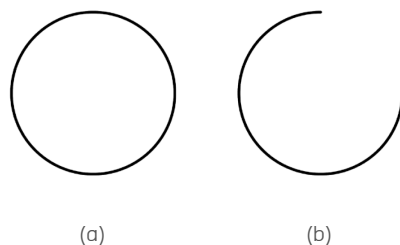
```
(editing
  ((trigger (holdDrag head))
    (action (rubberBand this tail head))
    ((trigger (holdDrag tail))
      (action (rubberBand this head tail))
      ((trigger (holdDrag this))
        (action (move this)))
      (display
        (originalStrokes shaft)
        (cleanedStrokes head1 head2)
        (color red)))
  )
```

5. ตัวอย่างเทคนิคอื่นๆ ที่เกี่ยวข้อง

5.1 การทดสอบความเป็นรูปปิด (closed shape testing)

โดยทั่วไปถ้าแบ่งลายเส้นโดยใช้ตำแหน่งของจุดปลายของเส้น stroke เป็นเกณฑ์ จะแบ่งได้เป็น 2 ประเภท ได้แก่

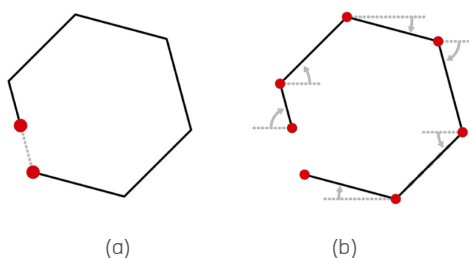
- 1) รูปปิด (closed shape) คือเส้น stroke ที่มีจุดปลายทั้งสองด้านอยู่ในตำแหน่งเดียวกันพอดี (ดูรูปที่ 10a) และ
- 2) รูปเปิด (open shape) คือเส้น stroke ที่มีจุดปลายทั้งสองด้านอยู่คนละตำแหน่งกัน (ดูรูปที่ 10b) ในทางปฏิบัติสำหรับการเก็บข้อมูลลายเส้นแบบดิจิตอลนี้ มีความเป็นไปได้ไม่น้อยมากที่จะได้เส้น stroke ที่เป็นรูปปิดโดยสมบูรณ์ เนื่องจากต้องอาศัยความแม่นยำของผู้วาดเป็นอย่างสูง เพราะฉะนั้นแทนที่จะตรวจสอบว่าจุดปลายทั้งสองด้านของเส้น stroke อยู่ที่ตำแหน่งเดียวกันหรือไม่ จึงเปลี่ยนมาตรวจสอบว่าจุดปลายทั้งสองด้านของเส้น stroke อยู่ห่างกันมากน้อยเพียงใดแทน



รูปที่ 10. ลายเส้นที่เป็น a) รูปปิด และ b) รูปเปิด

ยกตัวอย่างวิธีการทดสอบความเป็นรูปปิดใน PaleoSketch [1] จะเริ่มจากการตรวจสอบความใกล้เคียงของจุดปลายทั้งสองของเส้น stroke (ดูรูปที่ 11a ประกอบ) โดยการคำนวณหาระยะห่างระหว่างจุดปลายทั้งสองด้านของเส้น stroke จากนั้นนำมารหารด้วยความยาวของเส้น stroke หากผลลัพธ์ที่ได้มีค่าน้อยกว่าค่าที่กำหนดไว้ (ในกรณีของ PaleoSketch คือ 0.16) ก็จะถือว่าผ่านขั้นตอนนี้ไปสู่ขั้นตอนถัดไป คือขั้นตอนการตรวจสอบการวนรอบ (revolution) ของลายเส้น (ดูรูปที่ 11b ประกอบ) โดยการจับคู่จุด sample ที่อยู่ติดกัน มองแต่ละคู่ของจุด sample เป็นเส้นตรง (ในที่นี้จะขอเรียกเส้นตรงนี้ว่า substroke) แล้วนำมาคำนวณหามุมที่เส้นตรงนั้นทำกับแนวแกน X เราเรียกค่ามุมเหล่านี้ว่า direction นำค่า direction ของแต่ละคู่ของ substroke ที่อยู่ติดกันมาหาค่าความต่าง (นำมาลบกัน) ก็จะได้เป็นค่าการเปลี่ยนแปลงของ direction ของ substroke แต่ละคู่ ซึ่งจะขอเรียกว่าค่า rotation จากนั้นนำเอาค่า rotation ทั้งหมดมารวมกัน แล้วหารด้วย 2π ก็จะได้ผลลัพธ์เป็นค่า revolution ออกมา หากค่า revolution มากกว่าค่าที่กำหนดไว้ (ในกรณีของ PaleoSketch คือ 0.75) ก็จะถือว่าเส้น stroke นี้เป็นรูปปิด

การทดสอบความเป็นรูปปิดนี้สามารถนำไปใช้ประโยชน์ในการปรับปรุงลายเส้นโดยอาศัยการรู้จำระดับล่างได้ เช่น หากเส้น stroke เป็นรูปปิด ก็มีโอกาasเป็นไปได้

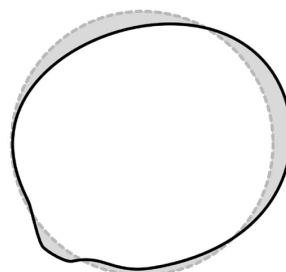


รูปที่ 11. การทดสอบความเป็นรูปปิดใน PaleoSketch

ว่าจะเป็นวงกลม หรือรูปเหลี่ยมใดๆ (polygon) เช่น สี่เหลี่ยม สามเหลี่ยม เป็นต้น และนั่นก็หมายความว่าสามารถนำไปใช้ประโยชน์สำหรับการปรับปรุงลายเส้นได้ด้วย

5.2 พื้นที่ความคลาดเคลื่อน (feature area)

พื้นที่ความคลาดเคลื่อนสามารถนำมาใช้ในการทดสอบรูปร่าง (shape testing) ซึ่งก็คือกระบวนการที่ทดสอบว่าเส้น stroke เป็นรูปร่างหนึ่งๆ หรือไม่ เช่น เป็นวงกลมหรือไม่ เป็นสี่เหลี่ยมหรือไม่ เป็นสามเหลี่ยมหรือไม่ เป็นต้น หรือแม้กระทั่งเปรียบเทียบว่าเป็นรูปร่างใดๆ ที่กำหนดหรือไม่ ซึ่งเป็นประโยชน์ต่อการปรับปรุงลายเส้นโดยอาศัยการรู้จำระดับล่าง โดยหลักการของพื้นที่ความคลาดเคลื่อนก็คือ เริ่มต้นจากการสร้างรูปร่างต้นแบบที่สมบูรณ์ (ideal shape) เช่น หากต้องการจะทดสอบว่าเส้น stroke เป็นวงกลมหรือไม่ ก็จะมีการสร้างเส้นวงกลมที่สมบูรณ์แบบ แต่พยายามรักษารูปร่างและขนาดของเส้นวงกลมให้ใกล้เคียงกับเส้น stroke มากที่สุดเท่าที่จะเป็นไปได้ (แสดงด้วยเส้นปะสีเทาในรูปที่ 12) จากนั้นจึงคำนวณหาพื้นที่ระหว่างเส้น stroke และเส้น ideal shape นั้น (แสดงด้วยพื้นที่สีเทาในรูปที่ 6) หากพื้นที่นี้มีมากเกินไป แสดงว่าเส้น stroke มีความผิดเพี้ยนไปจากเส้น ideal shape มาก ก็มีโอกาasเป็นไปได้ว่าเส้น stroke จะไม่ได้เป็นเส้นรูปร่างที่ต้องการ

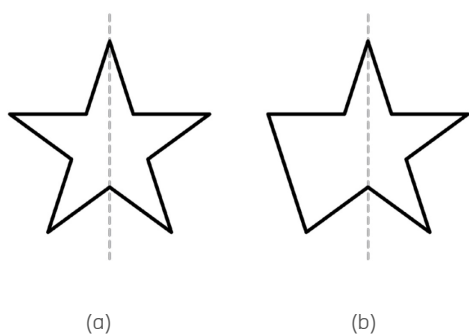


รูปที่ 12. ตัวอย่างการทดสอบพื้นที่คลาดเคลื่อนสำหรับรูปร่างกลม

5.3 การค้นหาความสมมาตร (Symmetry Detection)

ความสมมาตรเป็นคุณสมบัติอย่างหนึ่งของรูปร่าง จึงสามารถนำไปใช้ในการปรับปรุงลายเส้นโดยอาศัยการรู้จำระดับล่างได้ รูปร่างที่มีความสมมาตรตามแนวแกนสมมาตรใดๆ หมายถึงรูปร่างที่เมื่อนำมาพับตามแนวแกนสมมาตรแล้วจะได้รูปที่ทับกันสนิทพอดี จากรูปที่ 13 แกนสมมาตรแสดงด้วยเส้นประสีเทา รูปที่ 13a จะเป็นรูปที่สมมาตรตามแนวแกนสมมาตร ส่วนรูปที่ 13b นั้นไม่สมมาตรตามแนวแกนสมมาตร

ความสมมาตรเป็นคุณสมบัติหนึ่งของลายเส้น มีรูปที่มีความสมมาตรอยู่เป็นจำนวนมาก เพราะฉะนั้นหากเราสามารถตรวจพบได้ว่าลายเส้นมีความสมมาตรอยู่ ก็สามารถใช้การปรับปรุงลายเส้นทำให้ลายเส้นมีความสมมาตรอย่างสมบูรณ์แบบได้ เนื่องจากในทางปฏิบัตินั้นมีความเป็นไปได้ยากที่จะสามารถวาดรูปสมมาตรโดยสมบูรณ์แบบได้ด้วยมือ เป้าหมายของการค้นหาความสมมาตรจึงต้องรองรับข้อจำกัดนี้ด้วย คือมีความสามารถมากพอที่จะตรวจจบลายเส้นที่ไม่สมมาตรแต่ผู้วาดตั้งใจจะวาดให้สมมาตรได้



รูปที่ 13. ตัวอย่างรูปที่สมมาตรและไม่สมมาตรตามแนวแกนสมมาตร

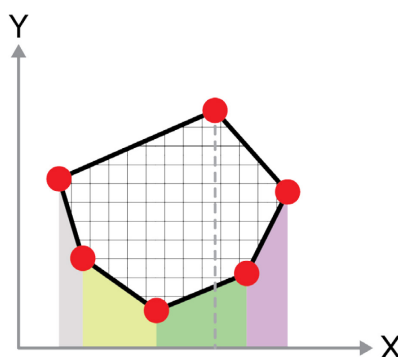
ยกตัวอย่างวิธีการตรวจจับความสมมาตรจาก [7] เริ่มจากการหาพื้นที่สี่เหลี่ยมภายใต้เส้น substroke แต่ละเส้น (สี่เหลี่ยมคางหมูหรือเรียกว่า trapezium ดังแสดงเป็นพื้นที่สีส้มต่างๆ ในรูปที่ 14) จากนั้นนำพื้นที่ของ trapezium ทั้งหมดมารวมกัน (ดูสมการที่ 1)

$$A_i = \sum_{i=1}^n \frac{x_i y_{i+1} - x_{i+1} y_i}{2} \quad (1)$$

จากสมการข้างต้น n คือจำนวนของเส้น trapezium ทั้งหมด (x_i, y_i) และ (x_{i+1}, y_{i+1}) เป็นตำแหน่งของจุดปลายทั้งสองของเส้น substroke ในลำดับที่ i สำหรับในกรณีที่เส้น stroke เป็นรูปเปิด ให้พิจารณาเส้นเชื่อมจุดปลายทั้งสองข้างของเส้น stroke เป็นหนึ่ง substroke จากนั้นหาจุดศูนย์กลางถ่วง (gravity centre) ของเส้น stroke (x_g, y_g) (ดูสมการที่ 2 และ 3)

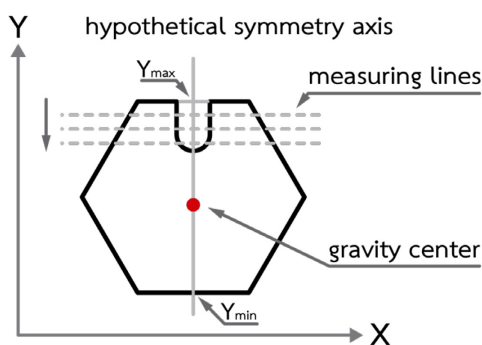
$$x_g = \sum_{i=1}^n (x_i + x_{i+1})(x_i y_{i+1} - x_{i+1} y_i) \quad (2)$$

$$y_g = \sum_{i=1}^n (y_i + y_{i+1})(x_i y_{i+1} - x_{i+1} y_i) \quad (3)$$

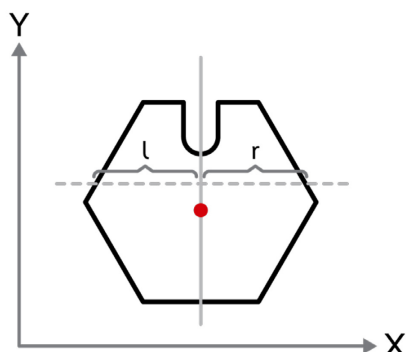


รูปที่ 14. พื้นที่ของ trapezium ของเส้น stroke (รูปจาก [7] หน้า 525)

ต่อมาจะเข้าสู่ขั้นตอนของการ scanning ทำได้โดยการแบ่งเส้น measuring lines เป็นเส้นแนวนอนจำนวนหนึ่งที่อยู่ห่างกันในระยะที่เท่าๆ กัน ตัดผ่านเส้น stroke (ดูรูปที่ 15) จะทำให้ได้ระยะ measuring stretch ของแต่ละเส้น measuring line มา ซึ่งก็คือระยะที่วัดจากเส้นแกนสมมาตรไปยังจุดที่เส้น measuring line ตัดกับเส้น stroke ทั้งฝั่งซ้าย (แสดงด้วย l ในรูปที่ 16) และขวา (แสดงด้วย r ในรูปที่ 16) ของเส้นสมมาตร ยิ่งระยะห่างระหว่างเส้น measuring lines แต่ละเส้นน้อย ผลของการสแกนจะยิ่งแม่นยำมากขึ้น แต่ก็ต้องใช้ระยะเวลาในการคำนวณมากขึ้นเช่นกัน



รูปที่ 15. เส้น measuring line (รูปจาก [7] หน้า 526)



รูปที่ 16. ระยะ measuring stretch (รูปจาก [7] หน้า 526)

จากนั้นทำการคำนวณค่า symmetry parameter เพื่อตรวจสอบความสมมาตร (ดูสมการที่ 4) (จาก [7] สมการ (8) หน้า 526)

$$Z_k = \frac{1}{n} \sum_{k=1}^n 1 - \frac{(l_i - r_i)}{L_{max}} \quad (4)$$

Z_k คือค่า symmetry parameter ของเส้น measuring line ลำดับที่ k ส่วน l คือระยะ measuring stretch ทางฝั่งซ้ายของเส้นสมมาตร r คือระยะ measuring stretch ทางฝั่งขวาของเส้นสมมาตร n คือจำนวนคู่ (l, r) ของระยะ measuring stretch ในแต่ละ measuring line (มีความเป็นไปได้ว่าเส้น measuring line 1 เส้นสามารถตัดกับเส้น stroke ได้มากกว่า 1 คู่) จากค่า Z_k เราสามารถสรุปได้ว่า ถ้าค่า Z_k เป็น 1 หมายความว่าเส้น stroke มีความสมมาตรในระดับของ measuring line ที่ k ถ้า Z_k เป็น 0 หมายความว่าที่ระดับ measuring line ที่ k ไม่มี measuring stretch ที่สมมาตรกันในฝั่งตรงข้ามของเส้นแกนสมมาตร ส่วนค่า Z_k ที่นอกเหนือจากนั้นหมายความว่าเส้น stroke ไม่มีความสมมาตรในระดับของ measuring line ที่ k จากนั้นจึงหาค่าเฉลี่ยของค่า Z_k แต่ละค่า (ดูสมการที่ 5) (จาก [7] สมการ (9) หน้า 527)

$$Z = \frac{\sum_{k=1}^N Z_k}{N} \quad (5)$$

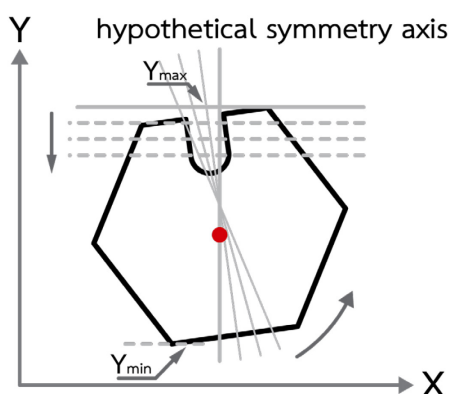
จากสมการข้างต้น N คือจำนวนของเส้น measuring lines ทั้งหมด เมื่อได้ค่า Z มา เราก็จะสามารถสรุปได้ดังนี้คือ ถ้าค่า Z มีค่าเท่ากับ 1 หมายความว่าเส้น stroke มีความสมมาตร แต่ถ้าค่า Z น้อยกว่า 1 ก็หมายความว่าเส้น stroke ไม่มีความสมมาตร แต่อย่างที่ได้อธิบายเอาไว้แล้วว่าเป็นการยากที่จะสามารถวาดรูปสมมาตรโดย

สมบูรณ์ด้วยมือ ในการพิจารณาว่าเส้น stroke มีความสมมาตรในแนวเส้นสมมาตรหรือไม่ ก็จะพิจารณาว่าค่า Z มีค่าเท่ากับ 1 หรือน้อยกว่า 1 เล็กน้อย ถือว่ามีความสมมาตร จากนั้นทำการหมุนเส้น stroke รอบจุดศูนย์กลางลงไปทีละน้อย เพื่อตรวจสอบหาแกนสมมาตรอื่นๆ ด้วย (ดูรูปที่ 17 ประกอบ) ยิ่งระยะของมุมที่หมุนเส้น stroke แต่ละรอบน้อยเท่าใด ผลการค้นหาค่าความสมมาตรก็มีความละเอียดเท่านั้น แต่ก็จะต้องใช้เวลาในการคำนวณมากขึ้นด้วยเช่นกัน

นอกจากวิธีการที่ยกตัวอย่างมาทั้งหมดแล้วนั้น ก็ยังมีวิธีการอื่นๆ ที่น่าสนใจอีกมากมาย แต่ละวิธีการก็สามารถนำมาใช้ประโยชน์ในด้านที่แตกต่างกัน สิ่งที่น่าสนใจคือ หากสามารถนำเอาวิธีการต่างๆ เหล่านี้มาประยุกต์ทำงานร่วมกันได้ โดยหยิบยกเอาส่วนที่เป็นข้อดีของแต่ละวิธีการ มาปรับปรุงส่วนที่เป็นข้อเสีย ก็จะทำให้ได้ผลลัพธ์การปรับปรุงลายเส้นที่ตรงตามความต้องการมากยิ่งขึ้นได้

6. บทสรุป

ในปัจจุบันมีการคิดค้นวิธีการต่างๆ ขึ้นมามากมายเพื่อตอบโจทย์การปรับแต่งลายเส้น เพื่อให้ได้ผลลัพธ์ที่ถูกต้องตรงตามความต้องการที่สุด โดยคำนึงถึงเวลาและทรัพยากร



รูปที่ 17. การหมุนเส้น stroke เพื่อคำนวณหาความสมมาตรในแนวแกนสมมาตรอื่นๆ (รูปจาก [7] หน้า 528)

ที่ใช้ แต่ละวิธีการที่ถูกคิดค้นขึ้นมายังมีข้อจำกัดที่ต่างกันออกไป ปัญหาที่ยังต้องการคำตอบที่เหมาะสมคือ ทำอย่างไรจึงจะสามารถปรับปรุงลายเส้นได้ตรงตามความต้องการของผู้ใช้ โดยการปรับปรุงการคาดการณ์ที่แม่นยำขึ้นจากข้อมูลลายเส้นและข้อมูลแวดล้อมอื่น ๆ ที่มี

เอกสารอ้างอิง

- [1] Brandon Paulson and Tracy Hammond. PaleoSketch: Accurate Primitive Sketch Recognition and Beautification. IUI '08', 2008, pp. 1-10.
- [2] Sezgin, T.M., Stahovich, T. and Davis, R. Sketch Based Interfaces: Early Processing for Sketch Understanding. ACM Press, 2001, pp. 1-8.
- [3] Yu, B. and Cai, S. A Domain-Independent System for Sketch Recognition. ACM Press, 2003, pp. 141-146.
- [4] Rubine, D. Specifying Gestures by Example. ACM Press, 1991, pp. 329-337.
- [5] Aaron David Wolin. Segmenting Hand-drawn Strokes, 2010, pp. 36-62.
- [6] Tracy Hammond and Randall Davis. LADDER, a Sketching Language for User Interface Developers. 2005, pp. 518-532.
- [7] Z. Szakal and I. Zsoldos. The Symmetry-diagram as a Tool of the Pattern Recognition. 2008, pp. 523-532.
- [8] C. Lawrence Zitnick. Handwriting Beautification Using Token Means. SIGGRAPH, 2013, pp. 1-8.
- [9] P. Barla, J.Thollot and F. X. Sillion. Geometric Clustering for Line Drawing Simplification. The Eurographics Association, 2005, pp. 1-10.