การรู้จำตัวเลขต่อเนื่องโดยการแมตชิ่งแบบไดนามิคโปรแกรมมิ่ง

พิพัฒน์ หิรัณย์วณิชชากร[1]

1. บทนำ

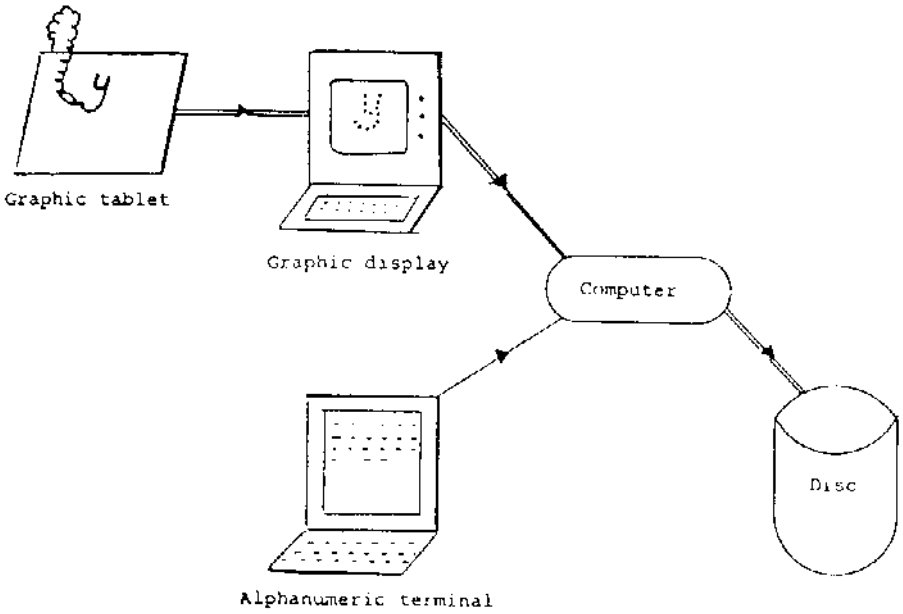
การรู้จำตัวอักขระ (Character recognition) ด้วยคอมพิวเตอร์ นับเป็นงานวิจัยที่สำคัญแขนงหนึ่ง เพราะผลการวิจัยสามารถนำไปใช้ประโยชน์มากมาย เช่น อุปกรณ์ช่วยในการป้อนข้อมูลสู่คอมพิวเตอร์ การประมวลใบเสร็จต่าง ๆ และ Points of Sales เป็นต้น ดังนั้น จึงมีการวิจัยเพื่อสร้างผลิตภัณฑ์ที่สามารถรู้จำตัวอักขระภาษาต่าง ๆ ด้วยคอมพิวเตอร์มากมาย สำหรับการรู้จำตัวอักขระนั้นสามารถแบ่งได้เป็น 2 ประเภท คือ การรู้จำแบบออฟไลน์ (Off-Line Recognition) และการรู้จำแบบออนไลน์ (On-Line Recognition) แบบแรกภาพของตัวอักขระที่ถูกพิมพ์หรือเขียนบนกระดาษจะถูกป้อนเข้าสู่คอมพิวเตอร์โดยผ่านเครื่องมือ Image Scanner เมื่อข้อมูลภาพตัวอักขระเข้าสู่คอมพิวเตอร์ ตัวอักขระแต่ละตัวจะถูกแยกออกมาจากข้อมูลภาพ แล้วตัวอักขระเหล่านั้นจะถูกรู้จำโดยอัลกอริทึมในการรู้จำต่อไป วิธีนี้เรียกทั่วไปว่า Optical Character Recognition (OCR) สำหรับการรู้จำแบบออนไลน์นั้น ผู้ใช้จะเขียนตัวอักขระเข้าสู่คอมพิวเตอร์โดยอาศัยเครื่องอ่านพิกัด (Tablet หรือ Digitizer) ดังแสดงในรูปที่ 1 แบบนี้ผลของการรู้จำจะถูกแสดงบนหน้าจอทันที ทำให้ผู้เขียนสามารถแก้ไขได้ทันทีเมื่อการรู้จำผิดพลาด ปัจจุบันการรู้จำแบบนี้มีการศึกษาวิจัยมากมายเนื่องจากสามารถนำไปใช้ประโยชน์ในทางธุรกิจได้ เช่น อุปกรณ์ OA การออกแบบโดยคอมพิวเตอร์ (CAD) การเรียนการสอนโดยคอมพิวเตอร์ (CAI) เป็นต้น การรู้จำแบบออนไลน์ยังแบ่งย่อยได้อีก 2 แบบ คือ การรู้จำตัวอักขระเขียนแยก (Isolated Characters) และการรู้จำตัวอักขระเขียนต่อเนื่อง สำหรับการรู้จำตัวอักขระเขียนแยกนั้น มีการวิจัยทั่วไปโดยเฉพาะตัวอักขระ จีน ญี่ปุ่น เกาหลี เพราะเป็นวิธีที่มีประสิทธิภาพในการป้อนข้อมูลเข้าระบบ Word Processor ดังแสดงในรูปที่ 2 แต่อย่างไรก็ตาม การรู้จำแบบนี้จะต้องมีการแยกตัวอักขระ

---

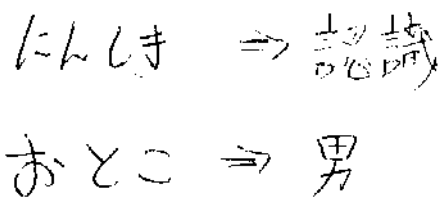
[1] คณะสถิติประยุกต์ สถาบันบัณฑิตพัฒนบริหารศาสตร์

(Segmentation) ซึ่งอาจทำได้โดยการเขียนให้ผู้เขียนส่งสัญญาณให้แก่ระบบ เช่น การจิ้มปากกาไปยังตำแหน่งที่กำหนดไว้บนแผ่น Tablet เมื่อเขียนจบตัวอักขระแต่ละตัว หรือมีการทิ้งเวลาไว้ช่วงหนึ่งเมื่อเขียนจบตัวอักขระ ซึ่งวิธีดังกล่าวทำให้เกิดความลำบากแก่ผู้ป้อนตัวอักขระ และทำให้เกิดความผิดพลาดได้ง่ายเมื่อต้องป้อนข้อมูลตัวอักขระจำนวนมาก ดังนั้นจึงมีการศึกษาวิจัยเกี่ยวกับการรู้จำตัวอักขระเขียนต่อเนื่อง [1] - [2] เพื่อให้การป้อนข้อมูลทำได้อย่างมีประสิทธิภาพและเป็นธรรมชาติ อีกทั้งสามารถนำไปใช้งานธนาคารและตลาดหุ้น สำหรับบทความนี้ได้เสนอผลการศึกษาวิจัยของการรู้จำตัวเลขเขียนต่อเนื่อง ซึ่งเป็นพื้นฐานของการรู้จำตัวอักขระเขียนต่อเนื่องตลอดจนการรู้จำเสียงพูดต่อเนื่องต่อไป

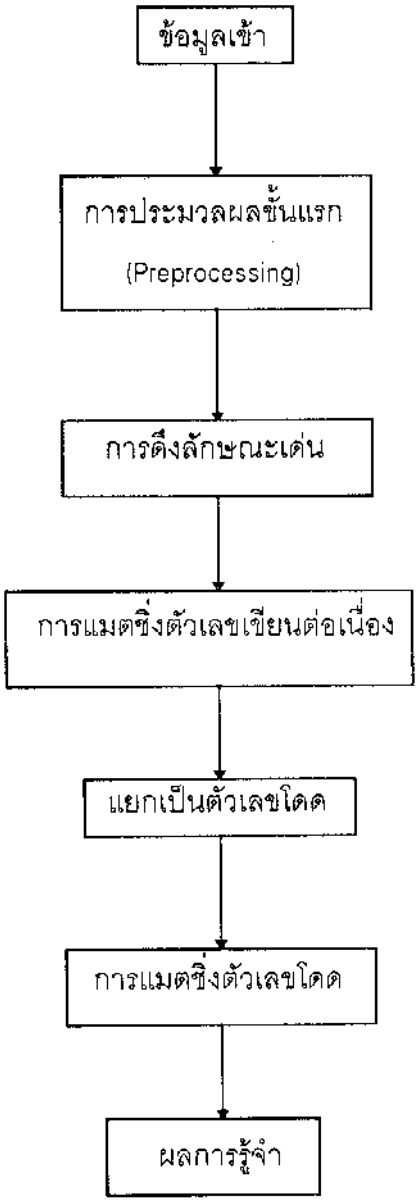
เนื่องจากการรู้จำตัวอักขระเขียนต่อเนื่องค่อนข้างยาก จึงยังมีการศึกษาวิจัยทางด้านนี้ไม่มากนัก แต่ในการรู้จำแบบอื่นนั้น เทคนิคอย่างหนึ่งที่ได้มีการพยายามนำมาใช้ในการรู้จำเสียงพูดต่อเนื่องได้ผลดีก็คือ เทคนิคของการแมตชิ่งแบบไดนามิคโปรแกรมมิ่ง (Dynamic Programming Matching) และก็ได้มีการนำเอาเทคนิคนี้มาใช้ในการรู้จำตัวอักขระเขียนแยกได้ผลดี การวิจัยนี้ได้นำเอาเทคนิคไดนามิคโปรแกรมมิ่งมาประยุกต์กับการรู้จำตัวเลขเขียนต่อเนื่องในการวิจัยเพื่อให้การรู้จำมีประสิทธิภาพจึงมีการแบ่งระบบของการรู้จำออกเป็นลำดับขั้นดังแสดงในรูปที่ 3 ในขั้นตอนแรกรูปของตัวเลขเขียนต่อเนื่องที่ถูกป้อนเข้ามานั้นจะถูกประมาณด้วยเส้นตรงเล็ก ๆ (Line Sigment) ซึ่งเชื่อมโยงกัน แล้วทิศทาง ความยาว รวมทั้งตำแหน่งของเส้นตรงเหล่านี้ถูกนำมาใช้เป็นลักษณะเด่นของตัวเลข หลังจากนั้นลักษณะเด่นของรูปตัวเลขเขียนต่อเนื่องนี้จะถูกนำมาแมตชิ่ง (Matching) กับโมเดลของตัวเลขต่อเนื่องโดยอาศัยหลักการเปรียบเทียบของไดนามิคโปรแกรมมิ่ง เมื่อผ่านขั้นตอนนี้ก็สามารถจะแยกรูปตัวเลขที่เขียนต่อเนื่องกันออกเป็นรูปตัวเลขโดด (Segmented Numeric) แต่ละตัวได้ หลังจากนั้นการแมตชิ่งแบบไดนามิคโปรแกรมมิ่งก็ถูกใช้อีกครั้งหนึ่งในการรู้จำตัวเลขโดดแต่ละตัว ในการทดลองเพื่อทดสอบวิธีการนี้ใช้ผู้เขียน 7 คน เขียนตัวเลขต่อเนื่อง 4 หลัก จำนวน 36 กลุ่ม รวมแล้วมีตัวเลขต่อเนื่อง 252 กลุ่ม (หรือตัวเลขโดด 1008 ตัว) จากการทดลองพบว่าเทคนิคที่ได้เสนอนี้มีประสิทธิภาพของการรู้จำตัวเลขโดดได้ถึง 99 %



รูปที่ 1 การรู้จำอักขระแบบออนไลน์



รูปที่ 2 การป้อนข้อมูลตัวอักขระญี่ปุ่น
รูปซ้ายมือใช้แป้นพิมพ์ป้อนด้วยตัว KANA
รูปขวามือใช้ Tablet เขียนตัว KANJI



รูปที่ 3 ขั้นตอนของการรู้จำตัวเลขเขียนต่อเนื่อง

2. การประมวลผลขั้นแรก (Pre-Processing)

ปกติแล้วเมื่อผู้ใช้ปากกาเขียนบนแผ่น Tablet โดยไม่มีการยกมือเลย จะเรียกสภาวะนี้ว่า Pen Down และเมื่อมีการยกมือขึ้นจะเรียกสภาวะนี้ว่า Pen Up ในแต่ละสภาวะ Pen Down จะได้หนึ่งเส้น (Stroke) ที่ประกอบเป็นตัวอักขระแต่ละเส้น (Stroke) ของตัวอักขระนี้เมื่อถูกป้อนเข้าสู่คอมพิวเตอร์จะเป็นลักษณะของจุดซึ่งต่อเนื่องกันเป็นรูปเส้น และในการป้อนข้อมูลแต่ละครั้งจะมีจุดรบกวน (Noise) เกิดขึ้นเสมอ เนื่องจากการเขียนไม่เรียบพอหรือจากการแปลงค่าเป็นดิจิตอล ดังนั้นจึงต้องมีการขจัดจุดรบกวนออกจากเส้นของตัวอักขระ สำหรับการวิจัยนี้ได้ใช้เทคนิคของการขจัดจุดรบกวนซึ่งเสนอโดยโยชิดะ (Yoshida) ในบทความ [3] นอกจากนั้นเนื่องจากตัวอักขระที่ถูกเขียนเข้าสู่คอมพิวเตอร์นั้นมีขนาดของตัวอักขระแตกต่างกันไป ดังนั้นหลังจากขจัดจุดรบกวนแล้วจึงต้องมีการปรับขนาดให้เป็นปกติของตัวอักขระ (Size Normalization) ซึ่งมีวิธีการดังต่อไปนี้

2.1 <u>การปรับขนาดให้เป็นปก</u>ติ (Size Normalization)

การปรับขนาดสามารถทำได้โดยการทำให้รัศมีโดยเฉลี่ยจากจุดศูนย์กลางของตัวอักขระหลังการปรับขนาดมีขนาดใกล้เคียงกันสำหรับทุก ๆ รูปแบบของตัวอักขระ นั่นคือเป็นการปรับขนาดด้วยรัศมีโดยเฉลี่ยของตัวอักษร

ให้ $P_n = (x_n , y_n)$ โดยที่ $n = 1 \sim N$ เป็นจุดต่อเนื่องของตัวอักขระก่อนการปรับขนาดแล้วจะได้ว่าจุดศูนย์กลางของตัวอักขระ (x_0 , y_0) และรัศมีโดยเฉลี่ย R_0 ของตัวอักขระนั้นสามารถคำนวณหาได้ดังสมการต่อไปนี้

$$x_0 = \sum_{n=1}^{N} x_n / N; y_0 - \sum_{n=1}^{N} y_n / N \qquad \text{------ (1)}$$

$$R_0 = \sum_{n=1}^{N} \sqrt{(x_n - x_0)^2 + (y_n - y_0)^2} / N \qquad \text{------ (2)}$$

ดังนั้น จะสามารถคำนวณหาจุดต่อเนื่องของตัวอักขระหลังจากการปรับขนาด $Pa_n = (xa_n , ya_n)$ โดยที่ $n = 1 \sim N$ ดังต่อไปนี้

$xa_n = C_0 \times (x_n - x_0) / R_0$

และ $ya_n = C_0 \times (y_n - y_0) / R_0$ ------- (3)

โดยที่ C_0 เป็นค่าคงที่สำหรับทุก ๆ ตัวอักขระ

สมการข้างต้นสามารถใช้ได้สำหรับการปรับขนาดของตัวอักขระเขียนแยกแต่ละตัว แต่อย่างไรก็ตามสำหรับตัวอักขระเขียนต่อกันนั้น จำนวนของตัวอักขระที่เขียนต่อกันตามแนวนอน (แกน X) ไม่แน่นอน ทำให้ระยะห่างของจุดต่อเนื่องจากจุดศูนย์กลางของรูปแบบของตัวอักขระเขียนต่อเนื่องตามแนวแกน X มีค่าแตกต่างกันมากสำหรับแต่ละรูปแบบ ดังนั้นการปรับขนาดของรูปแบบของตัวอักขระเขียนต่อเนื่องจึงมีการเปลี่ยนแปลงจากสมการข้างต้น โดยการทำให้ระยะห่างโดยเฉลี่ยตามแกน Y จากจุดศูนย์กลางรูปแบบของตัวอักขระมีค่าใกล้เคียงกันสำหรับทุก ๆ รูปแบบ กล่าวคือ ให้ $Q_k = (x_k , y_k)$ โดย $k = 1 \sim K$ เป็นจุดต่อเนื่องของรูปแบบตัวอักขระเขียนต่อเนื่องก่อนการปรับขนาด แล้วจะได้ว่าจุดศูนย์กลางของรูปแบบตัวอักขระ (x_c , y_c) และระยะห่างโดยเฉลี่ยตามแนวแกน Y คือ RY_c สามารถคำนวณได้ดังสมการต่อไปนี้

$$x_c = \sum_{k=1}^{K} x_k / K \ , \ y_c = \sum_{k=1}^{K} y_k / K \qquad \text{------ (4)}$$

$$RY_c = \sum_{k=1}^{K} \sqrt{(y_k - y_c)^2} / K \qquad \text{------ (5)}$$

ดังนั้น สามารถคำนวณจุดต่อเนื่องของรูปแบบตัวอักขระเขียนต่อเนื่องหลังจากการปรับขนาด $Qa_k = (xa_k , ya_k)$: $k = 1 \sim K$ ดังต่อไปนี้

$xa_k = C \times (x_k - x_c) / RY_c$

และ $ya_k = C \times (y_k - y_c) / RY_c$ ------ (6)

โดยที่ C เป็นค่าคงที่

2.2 <u>การแซมปลิ้งจุดบนเส้น (Stroke) ของตัวอักขระ</u>

ในการป้อนข้อมูลเข้าสู่คอมพิวเตอร์นั้น จุดต่อเนื่องที่ประกอบเป็นแต่ละเส้น (Stroke) ของตัวอักขระนั้นมีระยะห่างของจุดแตกต่างกันไปขึ้นอยู่กับความเร็วของการเขียนของ

ผู้ป้อนข้อมูล ระยะห่างซึ่งไม่คงที่ของจุดต่อเนื่องเหล่านี้ทำให้การหาลักษณะเด่นของแต่ละเส้น (Stroke) ทำได้ยากและได้ผลที่ไม่ดีเท่าที่ควร อีกทั้งทำให้ประสิทธิภาพของการแมตชิ่งระหว่างรูปแบบตัวอักขระเขียนต่อเนื่องกับโมเดลไม่ดี ดังนั้น หลังจากการปรับขนาดตามสมการ (6) แล้วจึงมีการแซมปิ้งจุดต่อเนื่องของแต่ละเส้นของตัวอักขระ โดยให้ระยะห่างของแต่ละจุดบนเส้นมีค่าใกล้เคียงกัน กล่าวคือ ถ้าให้ L_e เป็นความยาวของเส้น (Stroke) ใด ๆ ของตัวอักขระรูปแบบหนึ่ง ให้หาค่า m ซึ่งสอดคล้องตามเงื่อนไขที่ 1

<u>เงื่อนไข</u> (1) $(m - 1) \times Th < L_e < m \times Th$

โดยที่ m = 1, 2, 3 ... เป็นเลขจำนวนเต็มบวก และ Th เป็นค่าขีดแบ่ง (Threshold)

สำหรับทุกเส้นของรูปแบบตัวอักขระให้หาค่า m ตามเงื่อนไข (1) และค่า m ที่มีค่าน้อยที่สุดระหว่างเส้น (Storke) ต่าง ๆ ที่ประกอบเป็นรูปแบบตัวอักขระให้มีค่าเป็น MN เมื่อได้ค่า MN แล้วจึงทำการแบ่งแต่ละเส้น (Storke) ของรูปแบบตัวอักขระออกเป็น m x MN ส่วน โดยที่ m เป็นค่าที่หาได้ตามเงื่อนไขที่ (1) สำหรับแต่ละเส้นของตัวอักขระ

อนึ่ง สำหรับสภาวะ Pen up ซึ่งเป็นสภาวะตั้งแต่จุดสุดท้ายของเส้น (Stroke) หนึ่งไปยังจุดเริ่มต้นของเส้น (Storke) ถัดไป จะทำให้เกิดเป็นลักษณะเส้นตรง เส้นตรงนี้จะถูกเรียกว่าเส้นตรงที่มีสภาวะ Pen up (Pen up Line) สำหรับเส้นที่มีสภาวะ Pen up นี้จะไม่ถูกแบ่งย่อย กล่าวคือ จะถูกประมาณด้วยเส้นตรงเส้นเดียวจากจุดสุดท้ายของเส้น (Storke) ไปยังจุดเริ่มต้นของเส้น (Storke) ถัดไป

เมื่อได้ทำการประมาณเส้น (Storke) ด้วยเส้นตรงเล็ก ๆ (Line segment) ต่อเชื่อมกันแล้วก็จะสามารถหาลักษณะเด่นของรูปแบบตัวอักขระได้ดังหัวข้อต่อไปนี้

3. ลักษณะเด่นของตัวอักขระ

ในการรู้จำตัวอักขระแบบออนไลน์นั้น ทิศทางของการเขียนตัวอักขระมักจะถูกใช้เป็นลักษณะเด่นของตัวอักขระ แต่อย่างไรก็ตามทิศทางของการเขียนตัวอักขระนี้ไม่สามารถแยกแยะตัวอักขระ 6 กับ 0 ได้ ในการวิจัยนี้จึงมีการใช้พารามิเตอร์หลายตัวต่อไปนี้ประกอบเป็นลักษณะเด่นของตัวอักขระเพื่อให้การรู้จำมีประสิทธิภาพ

สำหรับแต่ละเส้นตรงย่อย (Line Segment) มีการใช้พารามิเตอร์เหล่านี้แสดงลักษณะเด่นของเส้นตรงย่อย

(1) ทิศทางของส้นตรงย่อย v_i : $0 \le v_i \le 360^\circ$

(2) ความยาวของเส้นตรงย่อย l_i

(3) สภาวะ Pen up หรือ Pen down ของเส้นตรงย่อย s_i กล่าวคือ
หากเส้นตรงย่อยมีสภาวะ Pen up ค่า $s_i = 0$
หากเส้นตรงย่อยมีสภาวะ Pen down ค่า $s_i = 1$

(4) ตำแหน่งกึ่งกลางของเส้นตรงย่อยตามแนวแกน Y , ค่า y_i , สำหรับค่า y_i นี้เป็นระยะห่างตามแนวแกน Y จากจุดศูนย์กลางของรูปแบบตัวอักขระไปยังจุดกึ่งกลางของเส้นตรงย่อย

ดังนั้น แต่ละเส้นตรงย่อย a_i สามารถแสดงได้ดังต่อไปนี้

$$a_i = (v_i , l_i , s_i , y_i) \qquad \text{------ (7)}$$

และรูปแบบตัวอักขระ A (หรือรูปแบบตัวอักขระเขียนต่อเนื่อง A) จะสามารถแสดงได้โดย

$$A = a_1 , a_2 , a_3 , \ldots , a_i , \ldots , a_l \qquad \text{------ (8)}$$

เมื่อ l เป็นจำนวนเส้นตรงย่อยที่ประกอบเป็นรูปแบบของตัวอักขระ (หรือรูปแบบตัวอักษรเขียนต่อเนื่อง)

4. การสร้างโมเดล

ในการรู้จำตัวอักขระต้องมีการสร้างโมเดลเพื่อใช้ในการแมตซิ่งกับรูปแบบของตัวอักขระเขียนต่อเนื่องที่ถูกป้อนเข้ามา และเนื่องจากการวิจัยนี้มีการแมตซิ่งทั้งในขั้นตอนของรูปแบบตัวอักขระเขียนต่อเนื่องและขั้นตอนของตัวอักขระโดด ดังนั้น จึงมีการสร้างโมเดลของตัวอักขระโดดและโมเดลสำหรับตัวอักขระเขียนต่อเนื่อง

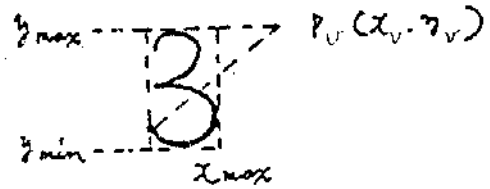
สำหรับโมเดลของตัวอักขระโดดนั้น ได้มาจากการเขียนตัวเลข 0 ถึง 9 แบบบรรจง จำนวน 1 ชุด แล้วจากตัวเลขแต่ละตัวก็ถูกนำมาผ่านขั้นตอนต่าง ๆ เพื่อหาพารามิเตอร์ที่แสดงเส้นตรงย่อยแต่ละเส้น a_i ตามสมการ (7) และจะได้โมเดลสำหรับแต่ละตัวเลขตามสมการ (8) ส่วนโมเดลสำหรับตัวเลขเขียนต่อเนื่องนั้น เนื่องจากจะต้องมีส่วนของเส้นตรงที่มีสภาวะ Pen up เชื่อมโยงระหว่างตัวเลขที่เขียนติดกัน ดังนั้น จึงต้องมีการสร้างส่วนของเส้นตรงนี้เพิ่มจาก

รูปแบบของตัวเลขโดด ดังแสดงในรูปที่ 4 กล่าวคือ จะมีการกำหนดจุด P_v (x_v , y_v) ขึ้นมา เพื่อที่จะให้เป็นจุดปลายของเส้นตรงที่มีสภาวะ Pen up เชื่อมระหว่างตัวเลขที่ติดกัน โดยที่จุด P_v ถูกกำหนดตามสมการข้างล่างนี้

$$x_v = x_{max} + (y_{max} - y_{min}) / R_v$$

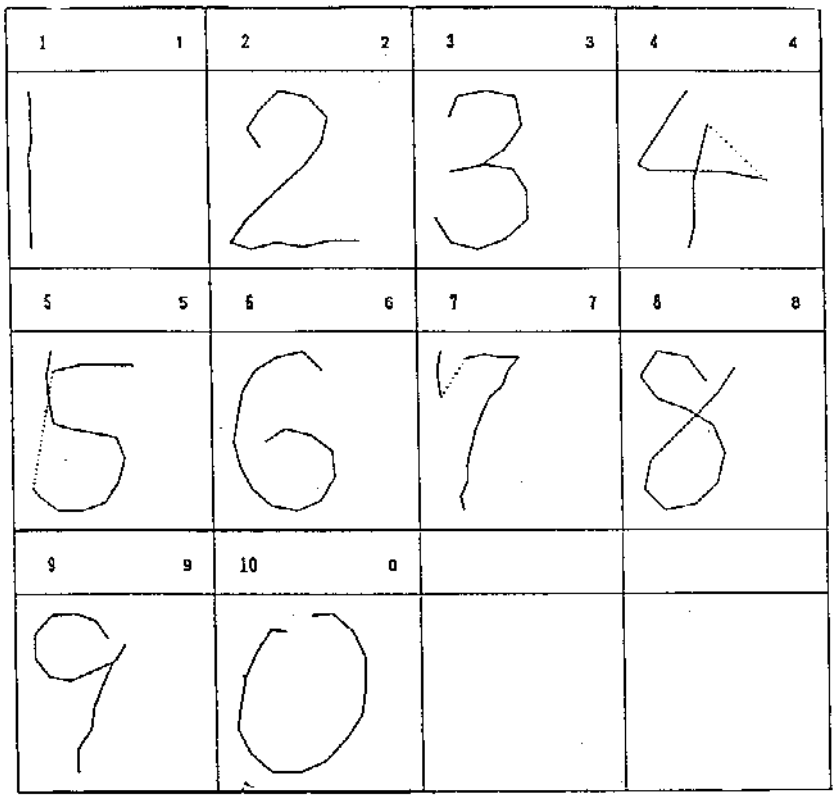
และ $$y_v = y_{max} \qquad \text{------ (9)}$$

โดยที่ R_v เป็นค่าคงที่ สำหรับการทดองนี้ให้มีค่าเท่ากับ 3 รูปที่ 5 (a) แสดงรูปแบบของตัวเลขที่ใช้เป็นโมเดลสำหรับการแมตซิ่งตัวเลขโดด ส่วนรูป 5 (b) แสดงรูปแบบของตัวเลขที่ใช้เป็นโมเดลสำหรับตัวเลขเขียนต่อเนื่อง กล่าวคือ โมเดลจะเกิดจากการนำรูปแบบตัวเลขในรูป 5 (b) มาเขียนต่อกัน โดยที่ตัวเลขตำแหน่งสุดท้ายจะใช้รูปแบบตัวเลขในรูป 5 (a) หลังจากนั้นจึงหาพารามิเตอร์สำหรับแสดงโมเดลตามสมการ (7) และ (8) ดังได้กล่าวข้างต้น

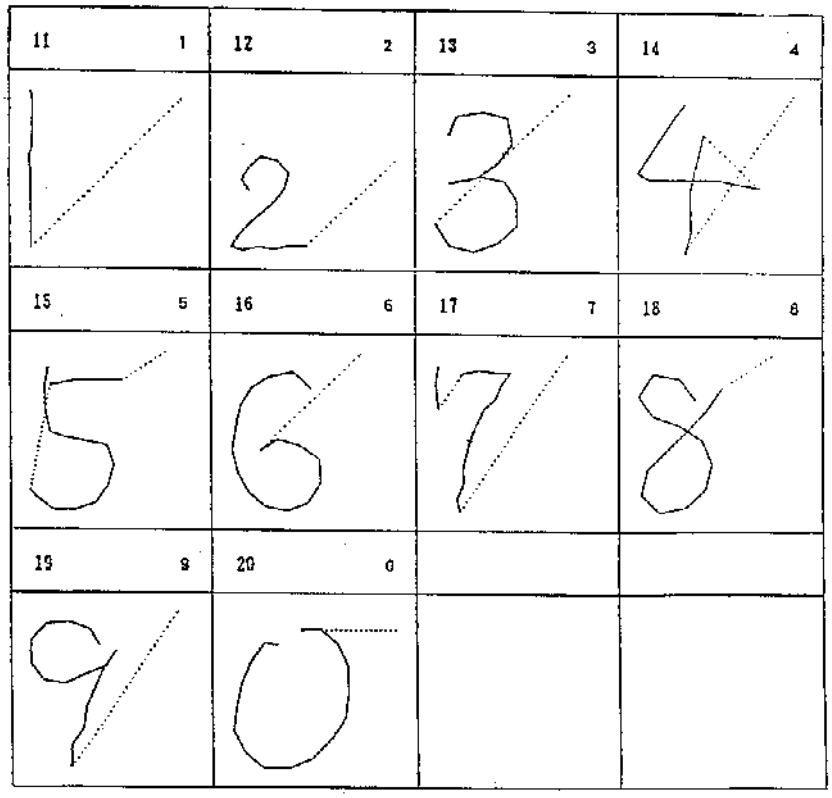


รูปที่ 4 การสร้างโมเดลสำหรับตัวเลขเขียนต่อเนื่อง

(a)



b)



รูปที่ 5 (a) โมเดลสำหรับตัวเลขโดด

(b) โมเดลสำหรับตัวเลขเขียนต่อเนื่อง

5. การแมตชิ่งแบบไดนามิคโปรแกรมมิ่ง

ไดนามิคโปรแกรมมิ่งเป็นวิธีการที่ถูกนำมาใช้ทั่วไปในการรู้จำเสียงพูด [4] - [6] เนื่องจากเป็นวิธีการที่สามารถปรับการแมตชิ่งได้ตามความยาวของการออกเสียงของผู้พูด นอกจากนั้นวิธีการนี้ยังได้ถูกนำมาใช้ในการรู้จำตัวอักขระทั้งแบบออนไลน์และออฟไลน์ [7] ในการวิจัยนี้ไดนามิคโปรแกรมมิ่งได้ถูกนำมาประยุกต์ใช้ในการแมตชิ่งตัวเลขเขียนต่อเนื่องได้อย่างมีประสิทธิภาพ และเพื่อแก้ปัญหาตำแหน่งที่ไม่แน่นอนของตัวเลขขึ้นลงที่ถูกเขียนต่อเนื่องกัน ตลอดจนปัญหาขอบเขตระหว่างตัวอักขระที่ติดกัน ดังนั้น ไดนามิคโปรแกรมมิ่งแมตชิ่งจึงถูกนำมาใช้สองขั้นตอน ขั้นตอนแรกใช้ในการแมตชิ่งรูปแบบตัวเลขเขียนต่อเนื่องกับรูปแบบโมเดลเพื่อที่จะแยกตัวเลขเขียนต่อเนื่องออกมาเป็นตัวเลขโดด หลักจากนั้นจึงทำการแมตชิ่งตัวเลขโดดกับโมเดลอีกครั้งซึ่งทำให้ผลการรู้จำดีขึ้นมาก ต่อไปนี้จะอธิบายถึงแต่ละขั้นตอนโดยละเอียด

5.1 การแมตชิ่งรูปแบบตัวเลขเขียนต่อเนื่อง

ให้ $A = a_1, a_2, \ldots, a_i, \ldots, a_I$ เป็นรูปแบบของตัวอักขระเขียนต่อเนื่องที่ถูกป้อนเข้าสู่คอมพิวเตอร์และถูกแสดงด้วยเส้นตรงเล็ก ๆ a_i ต่อเนื่องกันไปโดยที่ I เป็นจำนวนเส้นตรงที่ประกอบเป็นรูปแบบตัวอักขระเขียนต่อเนื่อง และ a_i สามารถแสดงด้วยพารามิเตอร์ต่อไปนี้

$$a_i = (v_i, l_i, s_i, y_i)$$

ส่วนรูปแบบของโมเดลหนึ่งสำหรับตัวอักขระเขียนต่อเนื่อง (B) สามารถแสดงได้ต่อไปนี้

$$B = b_1, b_2, \ldots, b_j, \ldots, b_J$$

$$b_j = (v_j, l_j, s_j, y_j)$$

โดย J เป็นจำนวนของเส้นตรงที่ประกอบเป็นโมเดล ดังนั้นสามารถหาค่าความแตกต่าง d (i , j) ระหว่างเส้นตรง a_i ของข้อมูลเข้าและเส้นตรง b_j ของโมเดลได้ตามนิยามต่อไปนี้

นิยามที่ 1 $d(i, j) = dv \times wv + ws + dy \times wy$ _____ (10)

โดยที่ $dv = \min\{ |v_i - v_j|, 360 - |v_i - v_j| \}$

$dy = |y_i - y_j|$

สำหรับ wv และ wy เป็นน้ำหนักถ่วงสำหรับพารามิเตอร์ v และ y ตามลำดับ ส่วน ws เป็นค่าปรับ (Penalty) ในกรณีที่เส้นตรง a_i และ a_j มีสภาวะของปากกาขึ้นลงแตกต่างกัน กล่าวคือ

ถ้า $s_i = s_j$ แล้วจะให้ $ws = 0$

แต่ถ้า $s_i = 1$ และ $s_j = 0$ แล้วจะให้ $ws = ws1$

แต่ถ้า $s_i = 0$ และ $s_j = 1$ แล้วจะให้ $ws = ws2$

โดยที่ ws1 และ ws2 เป็นน้ำหนักถ่วงของความแตกต่างของสภาวะของปากกาขึ้นลง

เนื่องจากขนาด I ของรูปแบบ A อาจจะแตกต่างกับขนาด J ของรูปแบบ B ดังนั้นจึงเป็นการยากที่จะทราบว่าเส้นตรง a_i เส้นใดของ A จึงควรจะเปรียบเทียบได้พอดีกับเส้นตรง b_j ของ B เพื่อที่จะหาความแตกต่างโดยรวมระหว่าง A กับ B แต่อย่างไรก็ตามโดยการประยุกต์ของไดนามิคโปรแกรมมิ่งทำให้สามารถคำนวณหาความแตกต่างสะสมที่น้อยที่สุด g (i , j) เมื่อเปรียบเทียบเส้นตรงตั้งแต่ a_1 , a_2 ... จนถึง a_i ของ A และเส้นตรง b_1 , b_2 ... จนถึง B ได้ตามสมการไดนามิคโปรแกรมมิ่งต่อไปนี้

$$g(i,j) = min \begin{cases} g(i,j) + l_i \times d(i,j), \\ g(i-1,j-1) + l_i \times d(i,j), \\ g(i,j-1) + l_j \times d(i,j) \end{cases}$$

_____ (11)

ซึ่งเมื่อ i = I และ j = J แล้ว g (I , J) คือความแตกต่างสะสมที่น้อยที่สุดของการเปรียบเทียบเส้นตรง a_1 , a_2 ,.... จนถึง a_I ของรูปแบบ A และเส้นตรง b_1 , b_2 ... จนถึง b_J ของรูปแบบ B ซึ่งก็คือความแตกต่างของรูปแบบ A และ B นั่นเอง

นอกจากนั้นสามารถหาเส้นทางของการแมตชิ่งระหว่างเส้นตรงต่าง ๆ h(i , j) ของข้อมูลกับโมเดลได้ดังต่อไปนี้

$$h(i,j) = h(\hat{i},\hat{j}) \qquad \text{_____ (12)}$$

โดยที่ $(\hat{i},\hat{j})$ เป็นค่า i,j ที่ทำให้ค่า g(i ,j) ในสมการ (11) มีค่าต่ำสุด

ดังนั้นจะเห็นได้ว่าด้วยสมการ (11) และ (12) ทำให้สามารถทราบได้ว่าเส้นตรงชุดใด (เส้นตรงเส้นเดียวหรือหลายเส้น) ของข้อมูลเข้าที่แมตชิ่งกับเส้นตรงชุดใด (เส้นตรงเส้นเดียวหรือหลายเส้น) ของโมเดล ซึ่งก็เป็นเส้นทางของการแมตชิ่งตั้งแต่ a_1 ถึง a_i และ b_1 ถึง b_j เมื่อทราบเส้นทางของการแมตชิ่งแล้วโดยการย้อนรอยกลับ (Back Tracking) ก็สามารถที่แยกตัวเลขที่เขียนต่อเนื่องออกเป็นตัวเลขโดดได้เนื่องจากเราทราบทั้งจุดเริ่มต้นและจุดสุดท้ายของแต่ละโมเดล (ดูรูป 5 (b))

นอกจากสามารถแยกตัวเลขเขียนต่อเนื่องเป็นตัวเลขโดดได้แล้วก็ยังสามารถรู้จำตัวเลขเขียนต่อเนื่องที่ถูกป้อนเข้ามานั้นได้ โดยที่โมเดลตัวเลขเขียนต่อเนื่อง B_M ใดที่ทำให้ค่า $g(I , J_M)$ มีค่าน้อยที่สุดก็จะถูกระบุเป็นรูปแบบของข้อมูลเข้านั้น แต่อย่างไรก็ตามการรู้จำในขั้นตอนนี้ได้ผลที่ไม่ดีนัก จึงทำการแมตชิ่งอีกครั้งสำหรับตัวเลขโดด

5.2 <u>การแมตชิ่งตัวเลขโดด</u>

เมื่อทำการแยกตัวเลขเขียนต่อเนื่องเป็นตัวเลขโดดได้แล้ว ตัวเลขโดดแต่ละตัวจะถูกปรับขนาดให้เป็นปกติ (Size Normalization) โดยคำนวณหาจุดต่าง ๆ ของเส้น (Stroke) ใหม่โดยอาศัยสมการ (3) หลังจากนั้นตัวเลขโดดที่ถูกปรับขนาดแล้วจึงถูกนำมาแมตชิ่งกับโมเดลซึ่งสร้างไว้สำหรับตัวเลขโดดที่แสดงในรูป 5 (a) ในขั้นตอนของการแมตชิ่งนี้ สมการ (10) ยังถูกนำมาใช้เพื่อหาค่าความแตกต่าง d (i, j) ระหว่างเส้นตรงย่อยของรูปแบบตัวเลขทั้งสอง และสมการ (11) ก็ถูกใช้เพื่อหาความแตกต่างระหว่างรูปแบบทั้งสอง สำหรับโมเดล MM ที่ทำให้ความแตกต่าง g (I , J) (โดยที่ I และ J เป็นจำนวนเส้นตรงของตัวเลขโดดและโมเดลตามลำดับ) มีค่าต่ำที่สุดก็จะถูกระบุเป็นตัวเลขโดดนั้น ๆ

6. ผลการทดลอง

ในการทดลองการรู้จำตัวเลขเขียนต่อเนื่องนั้นได้ใช้ผู้เขียน 7 คนเขียนตัวเลข 4 หลัก จำนวน 36 กลุ่ม ดังแสดงในตารางที่ 1 รวมเป็นตัวเลข 4 หลัก ที่ใช้ในการทดลองจำนวนทั้งหมด 252 กลุ่ม (1008 ตัว) ในตัวเลขเขียนต่อเนื่องกันนี้ปรากฏว่ามีผู้เขียน 3 คนที่เขียนตัวเลขติดกันโดยไม่ยกมือจำนวนทั้งสิ้น 18 กลุ่ม (จาก 252 กลุ่ม) ซึ่งเมื่อพิจารณาจุดที่เขียนติดกันแล้วนับได้ 22 จุด นอกจากข้อมูลที่ใช้ในการทดลองแล้วยังใช้ผู้เขียนอีกหนึ่งคน (นอกเหนือจาก 7 คนข้างต้น) เขียนตัวเลขที่ใช้เป็นโมเดล

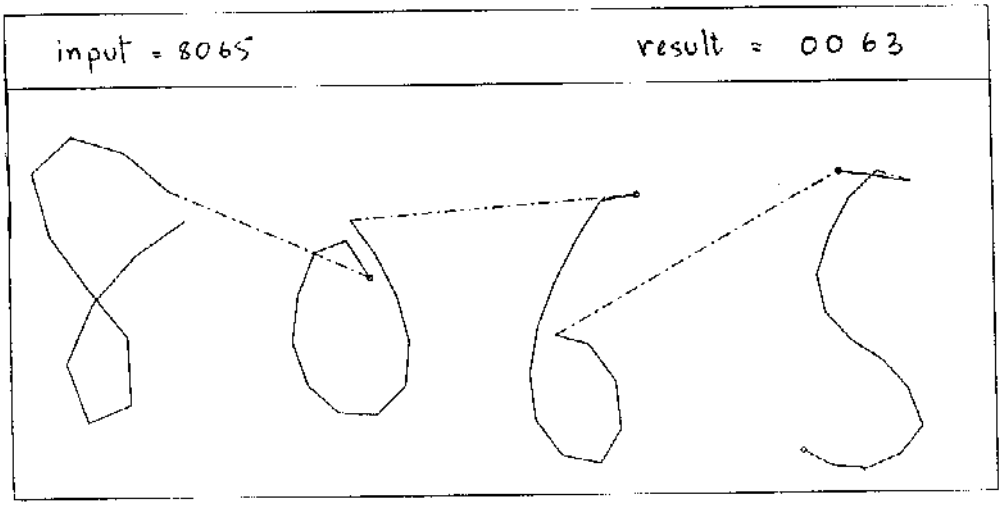
การทดลองนี้ได้แยกออกเป็น 2 แบบ แบบแรกทำการทดลองโดยทำการแมตชิ่งรูปแบบตัวเลขเขียนต่อเนื่องกับโมเดลเพียงขั้นตอนเดียว ในการทดลองนี้ได้กำหนดน้ำหนักของ

wv เท่ากับ 0.1 ส่วน ws1 และ ws2 ให้มีค่า 2 และ 6 ตามลำดับ แล้วเปลี่ยนค่า wy ไป สำหรับผลการทดลองนี้ได้แสดงในตารางที่ 2 ซึ่งจะเห็นได้ว่าเมื่อค่า wy = 0.025 ทำให้ประสิทธิภาพของการรู้จำกลุ่มตัวเลขเขียนต่อเนื่องมีค่าสูงสุดเท่ากับ 85.3% นอกจากนั้นรูปที่ 6 แสดงตัวอย่างรูปแบบกลุ่มตัวเลขเขียนต่อเนื่องที่รู้จำผิดพลาด ตำแหน่งของ ∘ ในรูปแสดงถึงตำแหน่งที่แยกตัวเลขต่อเนื่องออกเป็นเลขโดด จากตารางที่ 2 และรูปที่ 6 จะเห็นได้ว่าผู้เขียน E นั้นมีลำดับของการเขียนตัวเลข 5 และ 8 แตกต่างกับโมเดลซึ่งเป็นสาเหตุทำให้เกิดความผิดพลาดของการรู้จำถึง 60% ส่วนอีก 40% ของความผิดพลาดเกิดจากตำแหน่งของตัวเลขบางตัวในรูปแบบตัวเลขที่เขียนต่อเนื่องนั้นผิดแปลกไปอีกทั้งตัวเลขที่เขียนติดกันทำให้รูปแบบของตัวเลขเพี้ยนไป เป็นต้น

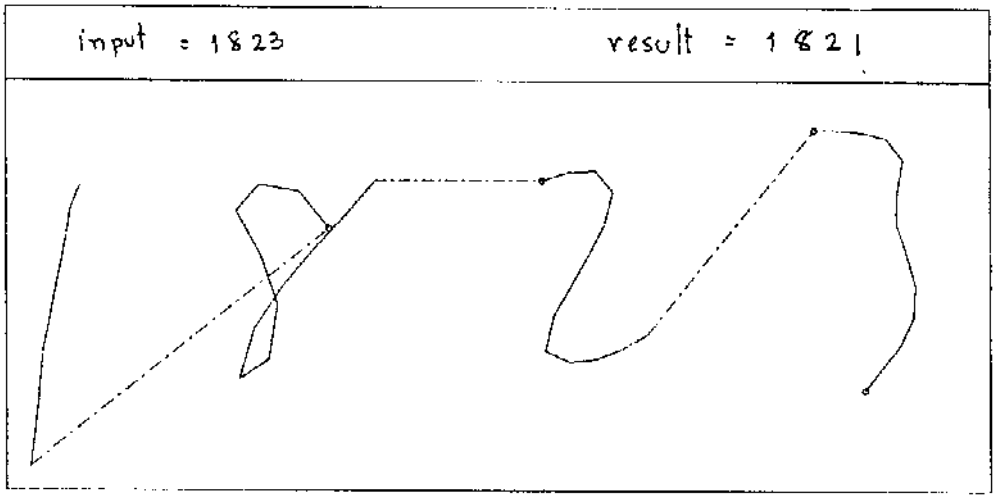
สำหรับการทดลองแบบที่สองได้แบ่งการแมตชิ่งออกเป็น 2 ขั้นตอน ขั้นตอนแรกใช้เพื่อแยกแยะข้อมูลออกเป็นตัวเลขโดด หลังจากนั้นในขั้นตอนที่สองจึงทำการแมตชิ่งตัวเลขโดดเพื่อทำการรู้จำตัวเลขแต่ละตัว อนึ่งในขั้นตอนของการแยกแยะข้อมูลเป็นตัวเลขโดดนั้น หากตัวเลขโดดที่ได้นั้นเส้นตรงส่วนท้ายสุดมีสภาวะเป็น Pen up ก็จะตัดส่วนที่เป็น Pen up ออก แล้วจึงนำตัวเลขโดดที่ได้มาทำการปรับขนาดแล้วทำการแมตชิ่งกับโมเดลในขั้นตอนที่ 2 ต่อไป

เนื่องจากผู้เขียน E มีลำดับของการเขียนตัวเลข 5 และ 8 แตกต่างไปจากโมเดล ดังนั้น ในการทดลองนี้จึงได้เพิ่มรูปแบบของตัวเลข 5 และ 8 ของผู้เขียน E เป็นโมเดล ในการทดลองนี้ได้กำหนดน้ำหนักถ่วง wv เป็น 0.1 และเปลี่ยนแปลงค่า wy เพื่อหาผลลัพธ์ของการรู้จำ นอกจากนั้นยังได้ทดลองเปลี่ยนค่า ws1 และ ws2 เพื่อลดปัญหาที่เส้นตรงที่มีสภาวะ Penup ไปแมตชิ่งกับเส้นตรงที่สภาวะ Pendown โดยได้ทดลองใช้ค่า ws1 เป็น 2 และ ws2 เป็น 6 หรือ ws1 เป็น 6 และ ws2 เป็น 12 หรือ ws1 เป็น 9 และ ws2 เป็น 18 จากผลของการทดลองพบว่าในขั้นตอนของการแมตชิ่งรูปแบบตัวเลขเขียนต่อเนื่องนั้น เมื่อค่า ws1 เป็น 6 และ ws2 เป็น 12 ให้ผลดีที่สุด ส่วนขั้นตอนของการแมตชิ่งตัวเลขโดดนั้น เมื่อ ws1 เป็น 2 และ ws2 เป็น 6 ให้ผลการรู้จำดีที่สุด

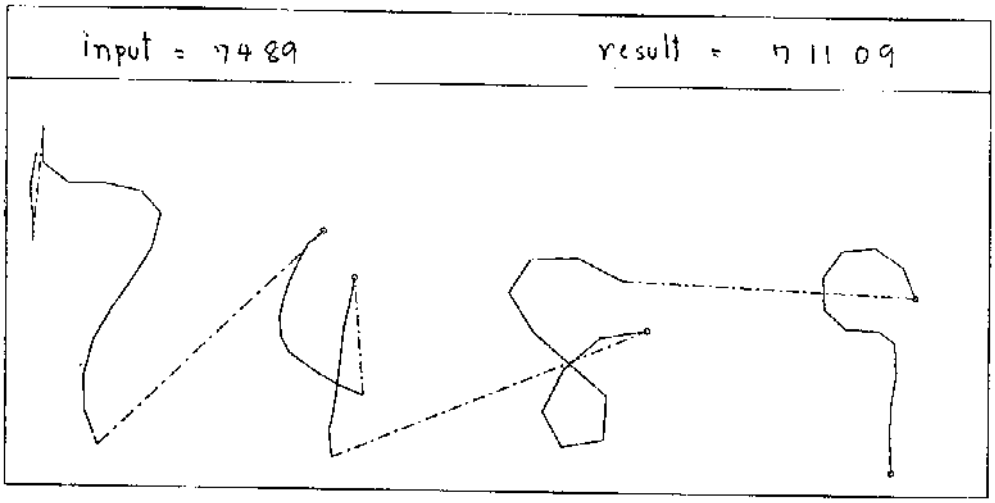
สำหรับผลของการรู้จำของการทดลองแบบที่สองได้แสดงในตารางที่ 3 (a) , (b) และ (c) ซึ่งระบุทั้งผลของการรู้จำกลุ่มของตัวเลข และผลของการรู้จำตัวเลขโดดแต่ละตัว อนึ่งท้ายตารางในวงเล็บแสดงถึงผลของการรู้จำในกรณีที่ใช้การแมตชิ่งสำหรับตัวเลขเขียนต่อเนื่องในขั้นตอนแรกเท่านั้น จากตารางจะเห็นได้ชัดว่าผลของการรู้จำเมื่อมีการใช้แมตชิ่ง 2 ขั้นตอนสูงกว่าการแมตชิ่งขั้นตอนเดียวมาก สำหรับรูปแบบตัวเลขที่การรู้จำผิดพลาดได้แสดงในรูปที่ 7 ส่วนสาเหตุของความผิดพลาดเกิดจากรูปแบบตัวเลขที่เขียนแตกต่างกับโมเดลมาก จึงทำให้การแมตชิ่งของตัวเลขได้ผลผิดพลาดไป (รูป (a) , (b) , (d) และ (f)) หรือทำให้การแยกแยะออกเป็นตัวโดดผิดพลาดไป (รูป (c) และ (e))



(a)

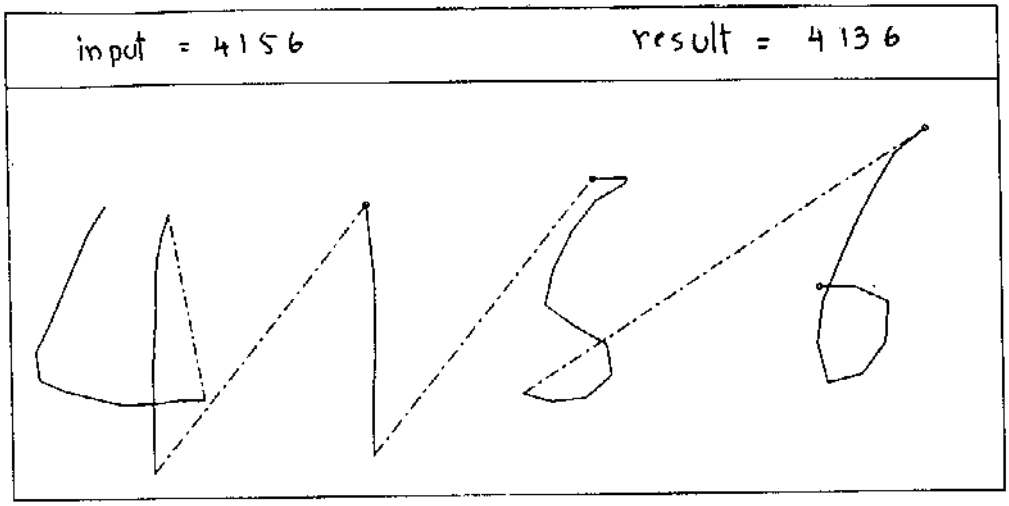


(b)

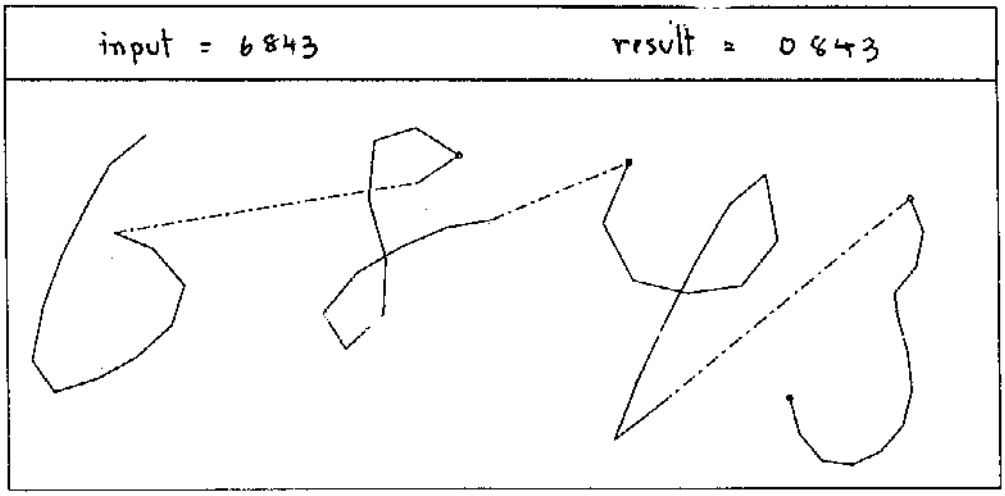


(c)

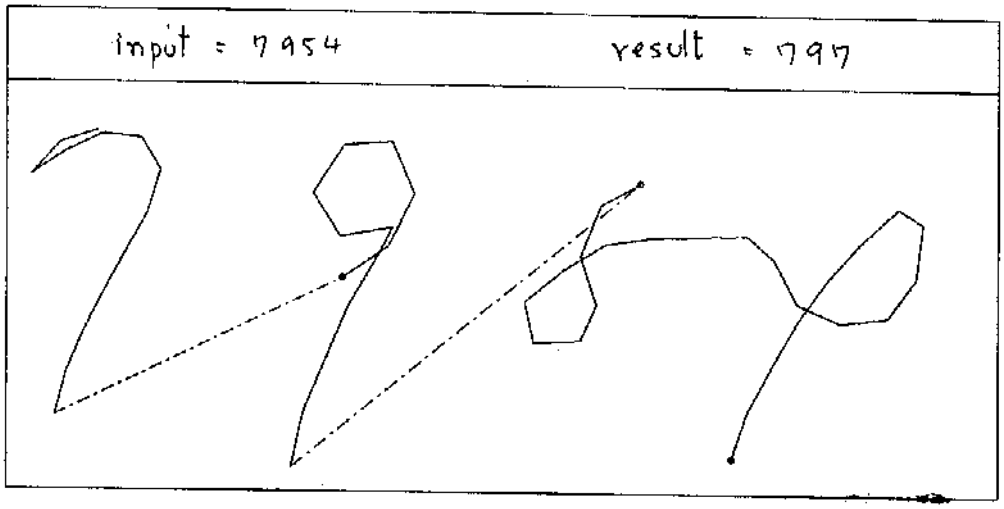
รูปที่ 6 ตัวอย่างรูปแบบที่รู้จำผิดพลาดเมื่อแมตชิ่งขั้นตอนเดียว



(a)

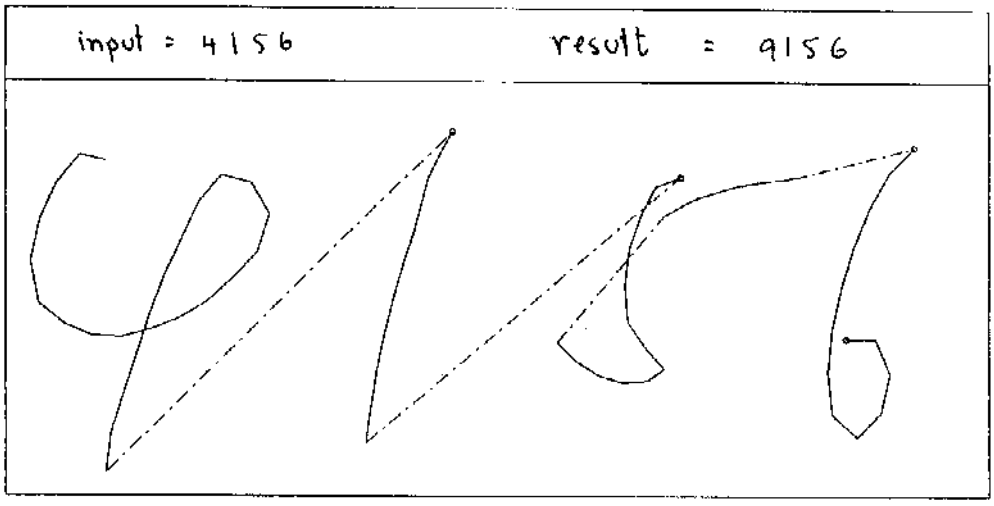


(b)

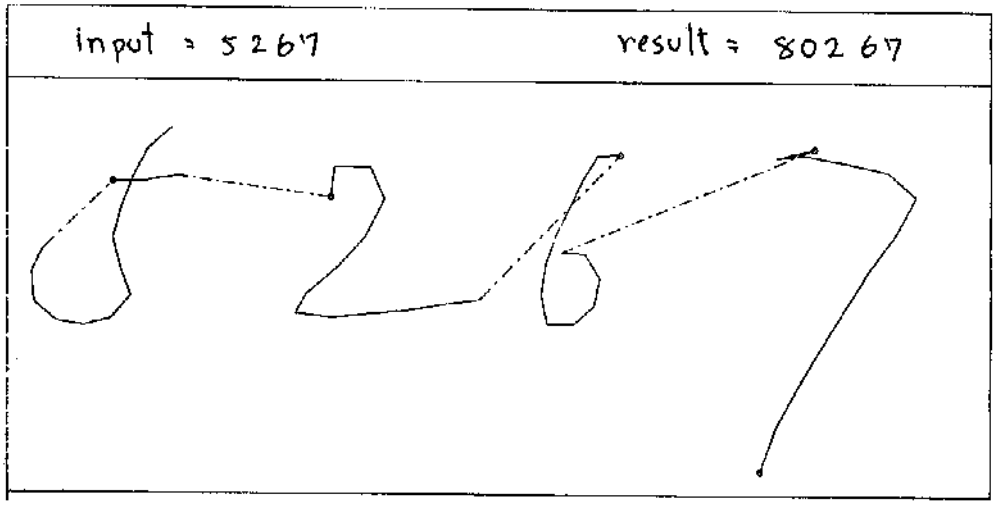


(c)

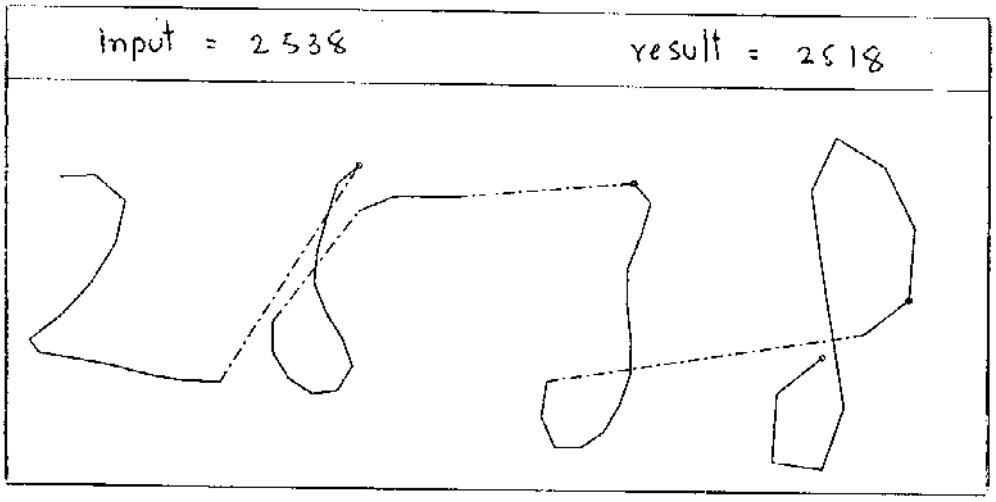
รูปที่ 7 รูปแบบที่รู้จำผิดพลาด



(d)



(e)



(f)

รูปที่ 7 รูปแบบที่รู้จำผิดพลาด

ตารางที่ 1 ชุดตัวเลขที่ใช้ในการทดลอง

1.	0287	2.	1398	3.	2409	4.	3510
5.	4621	6.	5732	7.	6843	8.	7954
9.	8065	10.	9176	11.	9601	12.	0712
13.	1823	14.	2934	15.	3045	16.	4156
17.	5267	18.	6378	19.	7489	20.	8590
21.	4750	22.	5861	23.	6972	24.	7083
25.	8194	26.	9205	27.	0316	28.	1427
29.	2538	30.	3649	31.	1199	32.	6633
33.	8877	34.	2244	35.	5500	36.	1000

ตารางที่ 2 ผลการรู้จำเมื่อมีการแมตชิ่งขั้นตอนเดียว

ค่า wy / ผลการรู้จำ		0.100	0.050	0.033	0.025	0.020	0.000
ผู้เขียน	A	66.66	86.11	88.88	91.66	91.66	63.88
	B	69.44	97.22	97.22	97.22	94.44	77.77
	C	50.00	86.11	91.66	94.44	91.66	63.88
	D	63.88	88.88	100.0	100.0	91.66	80.55
	E	13.88	33.33	38.88	38.88	36.11	22.22
	F	33.33	55.55	77.77	80.55	83.33	55.55
	G	66.66	80.55	91.66	94.44	83.33	55.55
ค่าเฉลี่ย		51.98	75.79	83.73	85.31	82.14	60.32

ตารางที่ 3 ผลของการรู้จำสำหรับการแมตชิ่งสองขั้นตอน

ค่า wy	CODE				DIGIT		
ผลการรู้จำ	0.033	0.025	0.020		0.033	0.025	0.020
A	100.0	100.0	100.0		100.0	100.0	100.0
B	100.0	100.0	100.0		100.0	100.0	100.0
ผู้เขียน C	100.0	100.0	100.0		100.0	100.0	100.0
D	100.0	100.0	100.0		100.0	100.0	100.0
E	97.22	97.22	97.22		99.30	99.30	99.30
F	83.33	86.11	86.11		95.14	95.83	95.83
G	100.0	100.0	100.0		100.0	100.0	100.0
ค่าเฉลี่ย	97.22	97.62	97.62		99.21	99.31	99.31

(a) ค่า wy = 0.033 สำหรับการแมตชิ่งขั้นตอนแรก

(ในการแมตชิ่งขั้นตอนแรกได้ผลการรู้จำของชุดตัวเลขเท่ากับ 84.13 %)

ค่า wy	CODE				DIGIT		
ผลการรู้จำ	0.033	0.025	0.020		0.033	0.025	0.020
A	97.22	97.22	97.22		98.61	98.61	98.61
B	100.0	100.0	100.0		100.0	100.0	100.0
ผู้เขียน C	100.0	100.0	100.0		100.0	100.0	100.0
D	100.0	100.0	100.0		100.0	100.0	100.0
E	97.22	97.22	97.22		99.30	99.30	99.30
F	83.33	86.11	86.11		95.14	95.83	95.83
G	97.22	97.22	97.22		99.30	99.30	99.30
ค่าเฉลี่ย	96.43	96.83	96.83		98.91	99.00	99.00

(b) ค่า wy = 0.025 สำหรับการแมตชิ่งในขั้นตอนแรก

(ในการแมตชิ่งขั้นตอนแรกได้ผลการรู้จำของชุดตัวเลขเท่ากับ 86.51 %)

ค่า wy	CODE			DIGIT		
ผลการรู้จำ	0.033	0.025	0.020	0.033	0.025	0.020
A	100.0	100.0	100.0	100.0	100.0	100.0
B	100.0	100.0	100.0	100.0	100.0	100.0
ผู้เขียน C	100.0	100.0	100.0	100.0	100.0	100.0
D	100.0	100.0	100.0	100.0	100.0	100.0
E	97.22	97.22	97.22	99.30	99.30	99.30
F	83.33	86.11	86.11	95.14	95.83	95.83
G	97.22	97.22	97.22	99.30	99.30	99.30
ค่าเฉลี่ย	96.82	97.22	97.22	99.11	99.21	99.21

(c) ค่า wy = 0.020 สำหรับการแมตชิ่งในขั้นตอนแรก

(ในการแมตชิ่งขั้นตอนแรกได้ผลการรู้จำของชุดตัวเลขเท่ากับ 84 92%)

7. บทสรุป

บทความนี้ได้เสนอการศึกษาพร้อมผลการทดลองการรู้จำตัวเลขเขียนต่อเนื่อง โดยอาศัยวิธีการแมตชิ่งแบบไดนามิคโปรแกรมมิ่ง ในงานวิจัยนี้เส้น (Stroke) ที่ประกอบเป็นรูปแบบตัวเลขถูกประมาณด้วยเส้นตรงเล็กเชื่อมต่อกัน แล้วทิศทาง ความยาว ตำแหน่ง ตลอดจนสภาวะการเขียนของเส้นตรงถูกใช้เป็นลักษณะเด่นในการรู้จำ นอกจากนั้นเพื่อให้ประสิทธิภาพของการรู้จำดีขึ้น การแมตชิ่งแบบไดนามิคโปรแกรมมิ่งได้ถูกใช้ 2 ขั้นตอน ขั้นตอนแรกใช้เพื่อแยกแยะรูปแบบตัวเลขเขียนต่อเนื่องออกเป็นตัวเลขโดด ซึ่งตัวเลขโดดจะถูกรู้จำโดยการแมตชิ่งในขั้นตอนที่สอง โดยการทดลองวิธีการนี้กับตัวเลข 4 หลัก ซึ่งเขียนโดยผู้เขียน 7 คน รวมจำนวนกลุ่มทั้งสิ้น 252 กลุ่ม หรือ 1008 ตัวสำหรับตัวเลขโดด ได้ผลการรู้จำถึง 99% สำหรับตัวเลขโดด จากการศึกษาทดลองนี้พอสรุปได้ว่าการแมตชิ่งแบบไดนามิคโปรแกรมมิ่งสามารถใช้งานได้ดีสำหรับการรู้จำตัวเลขเขียนต่อเนื่อง ซึ่งอาจจะนำแนวคิดของการศึกษานี้ไปประยุกต์ในการรู้จำตัวอักขระเขียนต่อเนื่องและการรู้จำเสียงพูดต่อเนื่องกัน ตลอดจนสามารถนำแนวคิดไปดัดแปลงเพื่อใช้ในการรู้จำตัวพิมพ์และตัวเขียนสำหรับระบบ OCR ต่อไป

8. เอกสารอ้างอิง

(1) Murase H., Wakahara T. and Umeda M. : "Online handwritten character-string recognition by candidate character lattice method", Paper of Technical Group, TGPRL 84-13, IECE Japan (1984) (in Japanese).

(2) Sato Y. and Ichihara T. "Feasibility study on online character recognition of cursive writing using feature of writing pressure" Paper of Technical Group, TGPRL 83-23, IECE Japan (1983) (in Japanese)

(3) Yoshida : "Model making for online recognition of alphabet characters", Research paper of NEC Research and Development center Japan, 8GB-906030 (in Japannese)

(4) Sokoe, H. and Chiba, S : "Dynamic programming algorithm optimization for spoken word recognition", IEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-26, pp. 43-49, Feb.1978.

(5) Sakoe, H. and Chiba, S. : "A dynamic programming approach to continuous speech recognition", in 1971 Proc. 7th ICA, Paper 20 C13, Aug. 1971.

(6) Sakoe, H. and Chiba, S. : "Comparative study of DP-pattern matching techniques for speech recognition", (in Japanese), in 1973 Tech. Group Meeting Speech, Acoust. Soc Japan, Preprints (S73-22), Dec. 1973.

(7) Hiranvanichakorn P. and Boonsuwan M. : "Recognition of multifont Thai characters by concave and convex features and DP matching algorithm", Computer Processing of Asia Languages, CPAL-2, pp.123-141 (March 1992)