การออกแบบ แบบจำลองในการรู้จำเสียงวรรณยุกต์สำหรับภาษาไทย โดยใช้เทคนิคการควอนไตซ์พิทช์ และ Hidden Markov Modeling

THE DESIGNING OF TONE RECOGNITION MODEL FOR THAI LANGUAGE USING PITCH QUANTIZATION AND HIDDEN MARKOV MODELING TECHNIQUES

จิตรลดา จารุมิศร์

CHITLADA CHARUMIT

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2542

ISBN 974-622-417-4

การออกแบบ แบบจำลองในการรู้จำเสียงวรรณยุกต์สำหรับภาษาไทย โดยใช้เทคนิคการควอนไตซ์พิทช์ และ Hidden Markov Modeling

THE DESIGNING OF TONE RECOGNITION MODEL FOR THAI LANGUAGE USING PITCH QUANTIZATION AND HIDDEN MARKOV MODELING TECHNIQUES

จิตรลดา จารุมิศร์

CHITLADA CHARUMIT

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ.2542

ISBN 974-622-417-4

THE DESIGNING OF TONE RECOGNITION MODEL FOR THAI LANGUAGE USING PITCH QUANTIZATION AND HIDDEN MARKOV MODELING TECHNIQUES

CHITLADA CHARUMIT

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN ELECTRICAL ENGINEERING

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

1999

ISBN 974-622-417-4

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองวิทยานิพนธ์

หัวข้อวิทยานิพนธ์ การออกแบบ แบบจำลองในการรู้จำเสียงวรรณยุกต์สำหรับภาษาไทย โดยใช้เทคนิคการควอนไตซ์พิทช์ และ Hidden Markov Modeling

THE DESIGNING OF TONE RECOGNITION MODEL FOR THAI LANGUAGE USING PITCH QUANTIZATION AND HIDDEN MARKOV MODELING TECHNIQUES

ชื่อนักศึกษา นางสาวจิตรลดา จารุมิศร์

รหัสประจำตัว 38061246

ปริญญา วิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชา วิศวกรรมไฟฟ้า

อาจารย์ผู้ควบคุมวิทยานิพนธ์ ผศ.อิทธิชัย อรุณศรีแสงไชย

คณะกรรมการสอบวิทยานิพนธ์		ลายมือชื่อ
ผศ.อิทธิชัย	อรุณศรีแสงไชย	
ผศ.ดร.ไกรสิน	ส่งวัฒนา	
รศ.ดร.ชม	กิ้มปาน	
ผศ.ดร.ยุทธพงษ์	รังสรรค์เสรี	
ดร.วิศิษฏ์	หิรัญกิตติ	

วัน/เดือน/ปี ที่สอบ 30 เมษายน 2542 เวลา 10.00 - 12.00 น.

สถานที่สอบ ห้องสอบวิทยานิพนธ์ คณะวิศวกรรมศาสตร์ ตึก 12 ชั้น ชั้น 4 ห้อง (E12-404)

บัณฑิตวิทยาลัยรับรองแล้ว

(รศ.ดร.มนัส สถิรกุล)

คณบดีบัณฑิตวิทยาลัย

วันที่ 19 เดือน พฤษภาคม พ.ศ. 2542

หัวข้อวิทยานิพนธ์	การออกแบบ แบบจำลองในการรู้จำเสียงวรรณยุกต์สำหรับภาษาไทย โดยใช้เทคนิคการควอนไตซ์พิทช์ และ Hidden Markov Modeling
นักศึกษา	นางสาวจิตรลดา จารุมิศร์
รหัสประจำตัว	38061246
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมไฟฟ้า
พ.ศ.	2542
อาจารย์ผู้ควบคุมวิทยานิพนธ์	ผศ.อิทธิชัย อรุณศรีแสงไชย
อาจารย์ผู้ควบคุมวิทยานิพนธ์ร่วม	ผศ.ดร.ไกรสิน ส่งวัฒนา

บทคัดย่อ

วิทยานิพนธ์นี้ได้เสนอการออกแบบ แบบจำลองเพื่อใช้ในการรู้จำหน่วยเสียงวรรณยุกต์สำหรับภาษาไทย โดยขั้นแรกเสียงพูดจะถูกแบ่งให้เป็นส่วนย่อยๆ ซึ่งแต่ละส่วนจะถูกนำมาคำนวณหาคาบเวลาพิทช์โดยใช้ออโตโครีเลชั่นที่มีขั้นตอนการคลิปยอดของสัญญาณ (Autocorrelation Method using Center Clipping) และจากคาบเวลาพิทช์นี้จะถูกแปลงเป็นค่าความถี่มูลฐานซึ่งเป็นตัวบ่งชี้ระดับสูง-ต่ำของเสียง ซึ่งลำดับของความถี่มูลฐานที่ได้จะถูกปรับปรุงให้มีความต่อเนื่องของข้อมูลโดยใช้มีเดียนฟิลเตอร์ จากนั้นทำการหาค่าการเปลี่ยนแปลงของความถี่มูลฐานของพิทช์นั้นๆเทียบกับเวลา โดยทำการควอนไตซ์การเบี่ยงเบนออกเป็น 3 ระดับตามทิศทางการเพิ่มขึ้น คงที่ หรือ ลดลง ของค่าความถี่มูลฐาน ซึ่งค่าที่ได้จากการควอนไตซ์นี้จะถูกนำไปใช้เป็นข้อมูลฝึกสอนให้กับการสร้างแบบจำลองการรู้จำของหน่วยเสียงวรรณยุกต์ทั้ง 5 ระดับด้วยวิธี Hidden Markov Model

งานวิจัยนี้ได้ทำการหาค่าความถี่มูลฐานจากคำพูดภาษาไทยพยางค์เดียว ซึ่งจากการจัดแบ่งข้อมูลฝึกสอนออกเป็น 3 ระดับนี้ทำให้ช่วงความถี่เสียงที่แตกต่างกันของชายและหญิงไม่มีผลต่อการสร้างแบบจำลองการรู้จำ ทำให้แบบจำลองที่สร้างขึ้นนี้ผู้ทดสอบทั้งชายและหญิงสามารถใช้ร่วมกันได้ นอกจากนี้ยังได้ทำการศึกษารูปแบบของ HMM ที่เหมาะสม ซึ่งจากการทดลองพบว่า HMM ขนาด 10 สเตทและมีการย้ายข้ามสเตทได้สูงสุดไม่เกิน 2 สเตท เป็นรูปแบบที่เหมาะสมกับอัลกอริธึมที่พัฒนาขึ้นมากที่สุด โดยในการทดสอบการรู้จำ ให้ผลการรู้จำระดับเสียงวรรณยุกต์ถูกต้องเฉลี่ยมากกว่า 90 เปอร์เซ็นต์

Thesis Title	The Designing of Tone Recognition Model For Thai Language Using Pitch Quantization and Hidden Markov Modeling Techniques
Student	Ms. Chitlada Charumit
Student ID.	38061246
Degree	Master of Engineering
Programme	Electrical Engineering
Year	1999
Thesis Advisor	Asst.Prof. Itthichai Arungsrisangchai
Thesis Co-Advisor	Asst.Prof.Dr.Kraisin songwattana

ABSTRACT

This Thesis presents the designing of tones level recognition modeling for spoken Thai language. First, the speech is divided into frames, and the autocorrelation method using center clipping is applied to each frame of speech to determine the pitch period and it fundamental frequency. The sequence of fundamental frequency is improved to make the connecting of data more smoother by using median filtered. The observation sequence of pitch levels are preprocessed to find the pitch differences and the sequence of pitch differences are then grouped into three quantized levels. The resultant sequence is used as bases for training a Hidden Markov Model and recognition of 5 tones.

The purpose of this experiment is processed to find pitch period from isolated monosyllable Thai speech, which indicates a possibility of gender independent tone recognition. The quantization of three levels has the properties of frequency independent. Further more, the studying also concerned with the optimum forms of HMM, and formed that a Hidden Markov Model with 10-state double-transition is optimized with our developed algorithm. The experimental results also showed the average recognition accuracy more than 90 percent.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลงได้ด้วยความช่วยเหลืออย่างดียิ่งจากหลายๆฝ่าย ซึ่งผู้จัดทำใคร่ขอขอบคุณทุกๆท่านที่มีส่วนร่วมสนับสนุน ช่วยเหลือและแนะนำในทุกๆด้าน

ขอขอบพระคุณ ผศ.อิทธิชัย อรุณศรีแสงไชย และ ผศ.ดร. ไกรสิน ส่งวัฒนา อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ได้กรุณาเสียสละเวลา ให้คำปรึกษา และข้อเสนอแนะที่เป็นประโยชน์ตลอดจนห้องทดลองและการทำงาน ให้การทำวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยดี

ขอกราบขอบพระคุณบิดา มารดา ผู้ให้โอกาสและคอยให้กำลังใจเสมอมา

ขอขอบคุณ พี่ๆ และ เพื่อนๆ ห้อง A-404 ที่ช่วยเหลือให้คำปรึกษาและกำลังใจมาโดยตลอด

ขอขอบคุณ เจ้าของเสียงทุกท่านที่ให้ความร่วมมือและช่วยเหลือในการเก็บข้อมูล เพื่อทำการวิจัยเป็นอย่างดี

สุดท้ายขอขอบคุณบัณฑิตวิทยาลัย ที่ได้ให้ทุนสนับสนุนการทำวิทยานิพนธ์ครั้งนี้

คุณค่าและประโยชน์อันพึงมีจากวิทยานิพนธ์ฉบับนี้ ผู้วิจัยขอมอบแด่ผู้มีพระคุณทุกท่าน

จิตรลดา จารุมิศร์

สารบัญ

หน้า

บทคัดย่อภาษาไทย I

บทคัดย่อภาษาอังกฤษ II

กิตติกรรมประกาศ III

สารบัญ IV

สารบัญตาราง VII

สารบัญภาพ VIII

บทที่ 1 บทนำ 1

1.1 กล่าวนำ I

1.2 วัตถุประสงค์ในการทำวิทยานิพนธ์ 2

1.3 ข้อกำหนดในการทำวิทยานิพนธ์ 2

1.3 โครงประกอบของวิทยานิพนธ์ 3

บทที่ 2 ระบบเสียงในภาษาไทย 4

2.1 ทฤษฎีการสร้างเสียงพูด 4

2.1.1 อวัยวะที่ใช้ในการออกเสียงพูด 4

2.1.2 ลักษณะร่วมของเสียงพูด 6

2.2 หน่วยเสียงสำคัญในภาษาไทย 7

2.3 หน่วยเสียงสระ 8

2.3.1 ลักษณะของเสียงสระ 8

2.3.2 หน้าที่ของหน่วยเสียงสระในภาษาไทย 9

2.4 หน่วยเสียงพยัญชนะ 9

2.4.1 ลักษณะของเสียงพยัญชนะ 9

2.4.2 หน้าที่ของหน่วยเสียงพยัญชนะในภาษาไทย 10

2.5 หน่วยเสียงวรรณยุกต์ 12

2.5.1 ลักษณะของเสียงวรรณยุกต์ 12

2.6 ลักษณะพยางค์ ของคำไทย 13

2.6.1 คำจำกัดความของ พยางค์ และคำในภาษาไทย 13

สารบัญ (ต่อ)

หน้า

2.6.2 ลักษณะโครงสร้างของคำพยางค์เดียวต่อการผันเสียงวรรณยุกต์ 14

บทที่ 3 การหาค่าความถี่มูลฐานของสัญญาณเสียงพูด 17

3.1 กล่าวนำ 17

3.2 การวิเคราะห์ในโดเมนเวลา 17

3.3 ทฤษฎีการประมาณค่าพิทช์ โดยใช้ออโตโครีเลชั่น ฟังก์ชั่น 17

3.3.1 การจัดแบ่งการวิเคราะห์สัญญาณออกเป็นช่วงสั้นๆ 18

3.3.2 การกำจัดผลของโครงสร้างฟอร์แมนท์ด้วยวิธี Center Clipping 21

3.4 สรุป 26

บทที่ 4 การเตรียมข้อมูลเพื่อสร้างแบบจำลอง 27

4.1 กล่าวนำ 27

4.2 การปรับปรุงความต่อเนื่องของข้อมูลด้วยวิธีการกรองมัธยฐาน 27

4.3 การควอนไตซ์ทิศทางการเปลี่ยนแปลงของความถี่มูลฐาน 29

4.4 สรุป 31

บทที่ 5 การสร้างแบบจำลองการรู้จำด้วยวิธี Hidden Markov Model (HMM) 32

5.1. กล่าวนำ 32

5.2 ส่วนประกอบของแบบจำลองมาร์คอฟ 32

5.3 ชนิดของ HMM 33

5.4 ปัญหาพื้นฐานของแบบจำลองมาร์คอฟ 35

5.5 การปรับปรุงค่าพารามิเตอร์ของ HMM 43

5.5.1 การสเกลลิง 43

5.5.2 ลำดับของค่าปรากฏหลายลำดับ 47

5.6 การสร้างแบบจำลองอ้างอิง 48

5.7 แบบจำลองฮิดเดนมาร์คอฟ ในการรู้จำเสียงวรรณยุกต์ภาษาไทย 50

สารบัญ (ต่อ)

หน้า

บทที่ 6 การทดลอง และผลการทดลอง 52

6.1 กล่าวนำ 52

6.2 การกำหนดขอบเขตของพยางค์ หรือ คำ 52

6.3 ขั้นตอนในการวิเคราะห์ และพัฒนาอัลกอริธึมในการสร้างแบบจำลองการรู้จำ 54

6.3.1 การหาค่าพิทช์ 54

6.3.2 การเตรียมข้อมูล 56

6.3.3 การสร้างแบบจำลองการรู้จำด้วย HMM 57

6.4 การสร้างแบบจำลองอ้างอิง เพื่อใช้ในการรู้จำระดับเสียงวรรณยุกต์ 60

6.5 การทดสอบแบบจำลองอ้างอิง 68

6.5.1 การทดสอบแบบจำลองอ้างอิง โดยใช้กลุ่มคำต้นแบบ 68

6.5.2 การทดสอบแบบจำลองอ้างอิง โดยใช้กลุ่มคำใหม่ 69

บทที่ 7 สรุปผล และ ข้อเสนอแนะ 73

7.1 การทดลอง 73

7.2 ข้อสังเกต ปัญหาที่พบในการทดลอง และข้อเสนอแนะ 74

เอกสารอ้างอิง 76

ภาคผนวก 78

ภาคผนวก ก. ตัวอย่างการตัดคำที่มีพยัญชนะต้น และพยัญชนะสะกด 79

ภาคผนวก ข. โปรแกรมที่พัฒนาขึ้นในการวิจัย 82

ภาคผนวก ค. ผลงานวิจัยที่ได้รับการตีพิมพ์ 123

ประวัติผู้เขียน 132

สารบัญตาราง

ตารางที่ หน้า

2.1 เสียงพยัญชนะในภาษาไทย 10

2.2 แสดงลักษณะของคำพยางค์เดียวในภาษาไทย 14

2.3 อักษรไตรยางค์ 15

2.4 ตัวอย่างการผันเสียงอักษรต่ำคู่ กับอักษรสูง 16

2.5 การจับคู่ในการผันเสียงวรรณยุกต์ 16

6.1 กลุ่มคำที่ 1 ใช้ในการทดสอบหาแบบจำลอง HMM ที่เหมาะสม 59

6.2 กลุ่มคำที่ 2 ใช้ในการสร้างแบบจำลองอ้างอิง 61

6.3 ผลการทดสอบแบบจำลองอ้างอิง โดยใช้กลุ่มคำและเสียงต้นแบบ จำนวน 925 เสียง 68

6.4 กลุ่มคำที่ 3 ใช้ในการทดสอบแบบจำลองอ้างอิง 69

6.5 ผลการรู้จำระดับเสียงวรรณยุกต์ จากการทดสอบโดยใช้เสียงจากผู้ออกเสียงต้นแบบ 70

6.6 ผลการรู้จำระดับเสียงวรรณยุกต์ จากการทดสอบโดยใช้เสียงจากผู้ออกเสียงกลุ่มใหม่ 70

สารบัญภาพ

ภาพที่ หน้า

1.1 ส่วนประกอบของระบบการรู้จำภาษาไทยโดยวิธีแยกจำลักษณะของหน่วยเสียง 1

2.1 ภาพตัดขวางแสดงอวัยวะในระบบการพูดของมนุษย์ 4

2.2 องค์ประกอบของพยางค์ในภาษาไทย 14

3.1 ออโตโครีเลชั่น ฟังก์ชั่น 19

3.2 ออโตโครีเลชั่น ฟังก์ชั่น สำหรับเสียงก้อง โดยใช้ค่า S ที่แตกต่างกัน 20

3.3 ฟังก์ชั่น เซนเตอร์คลิปปิ้ง และตัวอย่างแสดงการคลิปสัญญาณเสียงพูด 21

3.4 ฟังก์ชั่น เซนเตอร์คลิปปิ้ง แบบ 3 ระดับ 22

3.5 อัลกอริธึม การหาค่าพิทช์ 23

3.6 ตัวอย่างสัญญาณเสียง และฟังก์ชั่นโครีเลชั่น 25

3.7 ค่าความถี่มูลฐานของคำ อา อ่า อ้า อ๊า อ๋า จากผู้ออกเสียงเพศหญิง 26

4.1 การจัดแบ่งความถี่มูลฐานออกเป็นชุดข้อมูล 28

4.2 แสดงระดับความถี่มูลฐานที่แตกต่างกันระหว่างชาย และหญิง 29

4.3 แสดงการจัดแบ่งค่าความถี่มูลฐานออกเป็น 3 ระดับตามทิศทาง
การเปลี่ยนแปลงต่อเวลาที่เพิ่มขึ้น จากผู้ออกเสียงที่เป็นชายและหญิง.....30

5.1 แบบจำลองต่างๆของ HMM 33

5.2 กระบวนการไปข้างหน้า 37

5.3 กระบวนการย้อนกลับ 39

5.4 ลำดับการคำนวณ การเกิดค่าปรากฏร่วมซึ่งจะอยู่ที่สเตท i ที่เวลา t
และอยู่ที่สเตท j ที่เวลา t+1 41

5.5 โฟลว์ชาร์ต การคำนวณหาค่าพารามิเตอร์ของแบบจำลองอ้างอิง 49

5.6 บล็อคไดอะแกรม ของการรู้จำระดับเสียงวรรณยุกต์ด้วยแบบจำลองฮิดเดนมาร์คอฟ 50

6.1 ตัวอย่างสัญญาณเสียงพูดของคำว่า "อ๋อ" จากผู้ออกเสียงเพศชาย 53

6.2 ขั้นตอนในการวิเคราะห์ 54

6.3 ตัวอย่างสัญญาณในขั้นตอนต่างๆของการหาพิทช์ 55

6.4 ลักษณะการเปลี่ยนแปลงความถี่มูลฐานในวรรณยุกต์ทั้ง 5 เสียง 55

6.5 ตัวอย่างข้อมูลที่นำมาผ่านตัวกรองมัธยฐาน 56

6.6 การควอนไตซ์ข้อมูลออกเป็น 3 ระดับตามทิศทางการเปลี่ยนแปลงของความถี่มูลฐาน 57

6.7 Left-Right Model 4 สเตท 58

สารบัญภาพ (ต่อ)

ภาพที่ หน้า

6.8 เปอร์เซ็นต์ความถูกต้องเมื่อมีการเปลี่ยนแปลงสเตท และการย้ายข้ามสเตทของ HMM 59

6.9 แบบจำลอง HMM ที่เหมาะสมกับอัลกอริธึมที่พัฒนาขึ้น 60

6.10 ตัวอย่างทางเดินเสียงวรรณยุกต์ของคำที่เป็นเสียง สามัญ เอก โท ตรี และ จัตวา 71

จากผู้ออกเสียง 16 คน โดย

(a-e) เป็นทางเดินเสียงวรรณยุกต์ของผู้ออกเสียงเพศหญิง 8 คน และ

(f-j) เป็นทางเดินเสียงวรรณยุกต์จากผู้ออกเสียงเพศชาย 8 คน

บทที่ 1
บทนำ

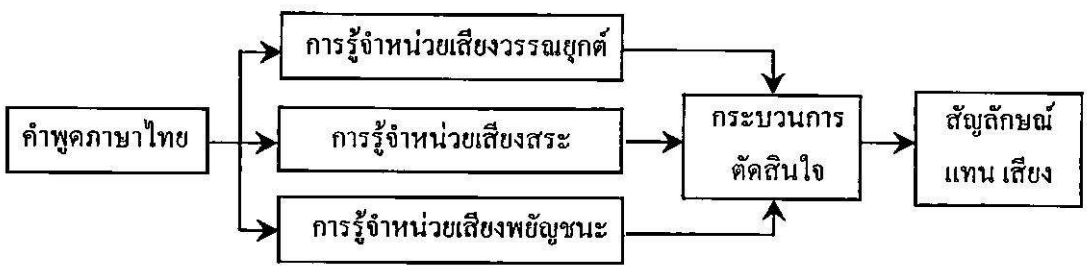
1.1 กล่าวนำ

ปัจจุบันเทคโนโลยีทางด้านคอมพิวเตอร์ได้ถูกพัฒนาให้มีขีดความสามารถมากขึ้น และนำมาใช้ร่วมกับเทคโนโลยีในด้านต่างๆ เพื่อทำให้การประมวลผลเป็นไปอย่างรวดเร็วและถูกต้อง ซึ่งโดยปกติการติดต่อสื่อสารระหว่างเครื่องคอมพิวเตอร์กับมนุษย์จะทำได้โดยการป้อนคำสั่งผ่านทางคีย์บอร์ดและเมาส์ ในขณะที่ได้มีความต้องการที่จะหาวิธีการอื่นที่สะดวกและเป็นธรรมชาติมากกว่า การพัฒนาให้คอมพิวเตอร์สามารถรับรู้คำสั่งจากเสียงพูดของมนุษย์ได้ จึงเป็นอีกเทคโนโลยีหนึ่งที่น่าสนใจซึ่งจะทำให้การติดต่อสื่อสารระหว่างมนุษย์กับคอมพิวเตอร์ทำได้ง่ายและสะดวกขึ้น

จากความต้องการให้เครื่องคอมพิวเตอร์สามารถรับรู้เสียงพูดได้ จึงทำให้เกิดศาสตร์แขนงหนึ่งเรียกว่า "Speech Recognition" แต่เนื่องจากการพูดของมนุษย์มีความซับซ้อน และมีความแตกต่างกันในแต่ละบุคคลจึงทำให้การพัฒนาเป็นไปอย่างล่าช้า โดยสามารถแบ่งวิธีการรับรู้เสียงพูดออกได้เป็น 2 วิธี คือ

1. พิจารณาทั้งหน่วยภาษาที่เปล่งเสียงออกมาทั้งชุด มีทั้งระบบการรู้จำคำเดี่ยว[1]-[2] (Isolated word Recognition) และระบบรู้จำคำพูดต่อเนื่อง (Continuous word Recognition) ซึ่งข้อดีของระบบเหล่านี้คือ ง่าย เนื่องจากมีการหลีกเลี่ยงผลกระทบอันเนื่องจากฐานของเสียงภายในคำหรือกลุ่มคำนั้น แต่ข้อเสีย คือสามารถรู้จำคำได้ในจำนวนคำที่จำกัด เนื่องจากต้องใช้เนื้อที่จำนวนมากในการจัดเก็บแบบจำลองอ้างอิง และต้องใช้เวลาในการคำนวณเพื่อเปรียบเทียบมากตามจำนวนของแบบจำลองอ้างอิงที่มีอยู่

2. พิจารณาโดยการแยกแยะรายละเอียดของหน่วยเสียง (Phonetic Recognition)[3]-[5] วิธีนี้จะพิจารณาลักษณะของหน่วยเสียงที่มีขนาดเล็กลงไป เช่น หน่วยเสียงพยัญชนะ หน่วยเสียงสระ และหน่วยเสียงวรรณยุกต์ ดังแสดงในรูปที่ 1.1 โดยจะใช้หน่วยเสียงย่อยเหล่านี้เป็นหลักในการรู้จำเสียงพูด ซึ่งวิธีนี้เหมาะสำหรับการพัฒนาไปสู่ระบบการรู้จำคำจำนวนมาก



รูปที่ 1.1 ส่วนประกอบของระบบการรู้จำภาษาไทยโดยวิธีแยกจำลักษณะของหน่วยเสียง

ด้วยเหตุนี้ เพื่อพัฒนาไปสู่ระบบการรู้จำเสียงพูดภาษาไทยทั้งภาษาซึ่งมีคำจำนวนมาก วิทยานิพนธ์นี้จึงทำการพัฒนาระบบการรู้จำแบบแยกแยะหน่วยเสียง โดยมุ่งหวังที่จะสร้างระบบการรู้จำหน่วยเสียงวรรณยุกต์ ซึ่งเป็นส่วนสำคัญที่ทำให้คำที่มีส่วนประกอบแวดล้อมอื่นๆเหมือนกัน คือมีเสียงพยัญชนะต้น สระ และพยัญชนะสะกดอย่างเดียวกันมีความหมายต่างกัน ดังนั้นจะเห็นว่าหน่วยเสียงวรรณยุกต์จึงมีหน้าที่ๆจะทำให้เกิดคำขึ้นใช้ในภาษามากขึ้น เป็นวิธีการสร้างคำขึ้นใช้เพิ่มขึ้นในภาษาเป็นวิธีแรก ทั้งนี้เพราะถ้าเปลี่ยนเสียงวรรณยุกต์ก็จะทำให้คำเกิดความหมายเพิ่มขึ้นใหม่นั่นเอง

1.2 วัตถุประสงค์ในการทำวิทยานิพนธ์

วิทยานิพนธ์ฉบับนี้ เป็นการสร้างแบบจำลองการรู้จำระดับเสียงวรรณยุกต์ของภาษาไทยคำโดด (Monosyllabic)หรือคำพยางค์เดียว โดยใช้การหาค่าคาบเวลาพิทช์ในโดเมนของเวลา แล้วนำตัวอย่างข้อมูลเสียงมาสร้างแบบจำลองการรู้จำด้วยฮิดเดนมาร์คอฟโมเดล โดยมีวัตถุประสงค์ดังนี้

1. เพื่อศึกษารูปแบบของทางเดินเสียงในแต่ละระดับเสียงวรรณยุกต์ของพยางค์
2. เพื่อศึกษาและออกแบบระบบการรู้จำหน่วยเสียงวรรณยุกต์ในภาษาไทย โดยมุ่งเน้นให้แบบจำลองการรู้จำที่สร้างขึ้นนี้สามารถรู้จำเสียงพูดแบบต่างบุคคล และสามารถรู้จำระดับเสียงในคำที่ต่างออกไปจากคำต้นแบบได้ โดยแบบจำลองที่พัฒนาขึ้นนี้สามารถใช้ร่วมกันได้ทั้งผู้ออกเสียงที่เป็นชายและหญิง
3. เพื่อหาฮิดเดนมาร์คอฟโมเดลที่เหมาะสม สำหรับการเพิ่มอัตราความถูกต้องในการรู้จำหน่วยเสียงวรรณยุกต์
4. เพื่อเป็นองค์ประกอบในการพัฒนาไปสู่ระบบการรู้จำคำพูดภาษาไทยทั้งภาษา

1.3 ข้อกำหนดในการทำวิทยานิพนธ์

1. งานวิจัยนี้มุ่งศึกษาพัฒนาอัลกอริธึมบนเครื่องคอมพิวเตอร์ส่วนบุคคล โดยใช้โปรแกรมภาษา Borland C++ โดยมีอุปกรณ์เพิ่มเติมได้แก่ การ์ดเสียง (Sound Blaster AWE64) ไมโครโฟน และลำโพง
2. ข้อมูลเสียงที่ใช้ในวิทยานิพนธ์ฉบับนี้ เป็นเสียงที่มีสำเนียงภาคกลางเท่านั้น
3. ในการทดลองได้แบ่งกลุ่มของคำทดสอบออกเป็น 3 กลุ่มจากผู้ออกเสียง 18 คน ที่เป็นชาย 9 คน และหญิง 9 คน ได้แก่
   - กลุ่มที่ 1 ประกอบด้วย หน่วยเสียง อา อี อู เอ โอ ที่ผันระดับวรรณยุกต์ทั้ง 5 เสียง (25 คำ) จำนวน 250 เสียง เพื่อใช้ในการทดสอบหาแบบจำลองของ HMM (Hidden Markov Model) ที่เหมาะสมกับอัลกอริธึมที่พัฒนาขึ้น

- กลุ่มที่ 2 ประกอบด้วยคำพยางค์เดียวที่มีพยัญชนะต้น สระ และพยัญชนะสะกดที่แตกต่างกันจำนวนทั้งสิ้น 95 คำ เพื่อใช้เป็นคำต้นแบบในการสร้างแบบจำลองอ้างอิงการรู้จำระดับเสียงวรรณยุกต์
- กลุ่มที่ 3 ประกอบด้วยคำพยางค์เดียวที่มีพยัญชนะต้น สระ และพยัญชนะสะกดที่แตกต่างกับกลุ่มคำที่ 2 จำนวนทั้งสิ้น 50 คำ เพื่อใช้เป็นคำทดสอบแบบจำลองการรู้จำระดับเสียงวรรณยุกต์ที่สร้างขึ้น

1.4 โครงประกอบของวิทยานิพนธ์

แบ่งออกเป็น 7 บทดังนี้

บทที่ 1 ดังได้กล่าวมาแล้วข้างต้น

บทที่ 2 กล่าวถึง ระบบการพูดของมนุษย์ในด้านองค์ประกอบของสรีระ และหน้าที่ของอวัยวะต่างในการเปล่งเสียงพูด และหน่วยเสียงที่ประกอบกันขึ้นเป็นพยางค์ของคำในภาษาไทย

บทที่ 3 กล่าวถึง การดึงเอาลักษณะพารามิเตอร์ที่ต้องการออกมาจากไฟล์เสียง ซึ่งก็คือค่าความถี่มูลฐาน(F_0) โดยสามารถคำนวณหาได้จากคาบของสัญญาณเสียงที่อยู่ในรูปของค่าพิทช์โดยใช้วิธีออโตโครีเลชั่น

บทที่ 4 กล่าวถึง ขั้นตอนการเตรียมข้อมูลเพื่อใช้เป็นส่วนข้อมูลฝึกสอน (training data) ของ HMM เพื่อสร้างแบบจำลองอ้างอิงในการรู้จำของหน่วยเสียงวรรณยุกต์ทั้ง 5 เสียง

บทที่ 5 กล่าวถึงทฤษฎีการสร้างแบบจำลองการรู้จำด้วย Hidden Markov Model

บทที่ 6 เป็นขั้นตอนในการทดลอง

บทที่ 7 เป็นบทสรุปเกี่ยวกับการทดลองทั้งหมดที่ทำมา พร้อมทั้งข้อสังเกต ปัญหาที่พบในการทดลอง และข้อเสนอแนะสำหรับผู้ที่ทำการวิจัย และพัฒนาระบบการรู้จำเสียงพูดต่อไป

ภาคผนวก

ภาคผนวก ก ตัวอย่างการตัดคำที่มีพยัญชนะต้น และพยัญชนะสะกด

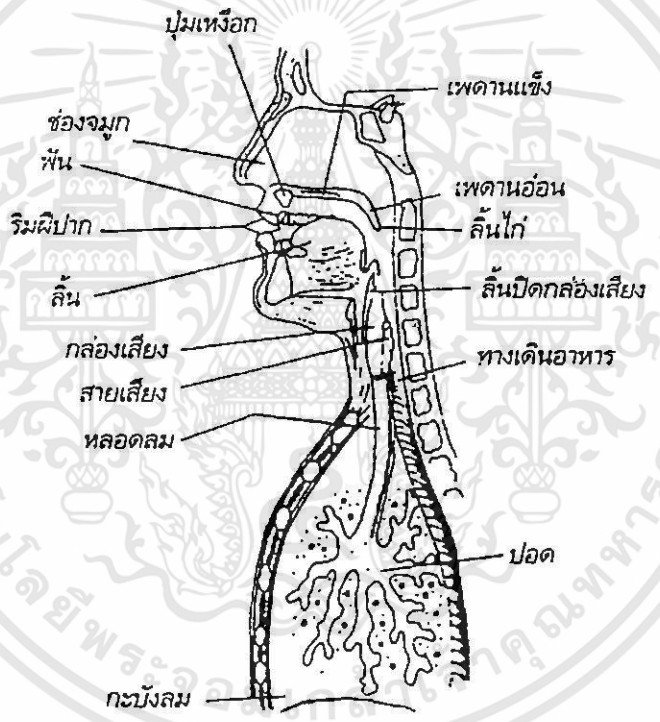
ภาคผนวก ข โปรแกรมที่พัฒนาขึ้นในการวิจัย

ภาคผนวก ค ผลงานวิจัยที่ได้รับการตีพิมพ์

บทที่ 2
ระบบเสียงในภาษาไทย

2.1 ทฤษฎีการสร้างเสียงพูด

การพูดของมนุษย์มิใช่อาการที่เกิดเฉพาะที่ปากเท่านั้น หากเริ่มจากลมหายใจเข้าของมนุษย์เองที่นำลมเข้าสู่ปอด จากนั้นจะใช้ลมจากปอดซึ่งก็คือลมหายใจออก มาทำให้เกิดเสียงพูด โดยลมจะถูกบังคับให้ผ่านอวัยวะต่างๆที่สำคัญ คือ เส้นเสียงซึ่งอยู่ในช่องของหลอดลม หรือบริเวณที่เรียกว่าลูกกระเดือก แล้วลมนั้นจะผ่านต่อออกมาทางช่องปากหรือช่องจมูก อวัยวะในช่องปากก็จะดัดแปลงให้เป็นเสียงต่างๆตามที่ผู้พูดต้องการ [6]



รูปที่ 2.1 ภาพตัดขวางแสดงอวัยวะในระบบการพูดของมนุษย์

2.1.1 อวัยวะที่ใช้ในการออกเสียงพูด

อวัยวะส่วนที่มีหน้าที่โดยตรงในการออกเสียงพูด ดังแสดงในรูปที่ 2.1 มีดังนี้คือ

1. ริมฝีปาก เป็นอวัยวะส่วนที่เคลื่อนไหวได้มาก และทำให้เสียงแตกต่างกันได้มาก เราอาจจะบังคับริมฝีปากให้อยู่ชิดกัน ห่างกัน ยื่นออก หรือห่อกลม ฯลฯ ลักษณะริมฝีปากต่างๆนี้ล้วนแต่มีอิทธิพลต่อการออกเสียง และการทำให้เสียงแตกต่างกันไปทั้งสิ้น

2. ฟัน เป็นอวัยวะที่เกิดของเสียงหลายชนิด เช่นเมื่อฟันบนกดลงบนริมฝีปากล่าง หรือฟันล่าง ลมที่ผ่านออกมาโดยแรงจะลอดช่องที่พอผ่านได้ออกมา ทำให้เกิดเป็นเสียงชนิดที่เรียกว่าเสียงเสียดแทรก เป็นต้น

3. ปุ่มเหงือก เป็นส่วนนูนออกมาอยู่หลังฟันด้านบน ถ้าเอาลิ้นแตะดูจะรู้สึกว่ามีลักษณะเป็นคลื่น

4. เพดานแข็ง หรือ เพดานปาก คือ ส่วนเฉพาะที่โค้งเป็นกระดูกแข็ง

5. เพดานอ่อน คือ ส่วนของเพดานที่อยู่ต่อจากเพดานแข็งไปข้างในมีลักษณะเป็นกระดูกอ่อนที่ขยับขึ้น-ลงได้ เวลาหายใจเพดานอ่อนและลิ้นไก่ซึ่งอยู่ตอนปลายจะลดระดับลงมาเปิดช่องให้ลมออกไปทางจมูก เวลาพูดส่วนใหญ่ปลายเพดานอ่อนและลิ้นไก่จะถูกยกขึ้นไปจรดกับหลังคอ นอกจากเวลาออกเสียงนาสิกเท่านั้นที่เพดานอ่อนจะลดระดับลงมา เพื่อให้ลมออกทางช่องจมูก

6. ลิ้นไก่ เป็นก้อนเนื้อเล็กๆอยู่ต่อปลายเพดานตรงกลางปาก อวัยวะส่วนนี้สั่นรัวได้

7. ลิ้น เป็นส่วนที่เคลื่อนไหวมากที่สุดในการออกเสียงพูด จึงต้องแบ่งออกเป็น 3 ส่วน ตามหน้าที่ในการออกเสียง คือ

7.1) ปลายลิ้น คือ ส่วนปลายลิ้นซึ่งสามารถยกขึ้นไปแตะอวัยวะส่วนต่างๆในปากตอนบนได้โดยง่าย

7.2) หน้าลิ้น คือ ลิ้นที่อยู่ตรงข้ามกับเพดานแข็ง

7.3) หลังลิ้น คือ ส่วนของลิ้นที่อยู่ตรงข้ามกับเพดานอ่อน

8. แผ่นเนื้อปากหลอดลม เป็นก้อนเนื้อเล็กๆคล้ายลิ้นไก่ อยู่ต่อโคนลิ้นลงไปในคอ มีหน้าที่ปิดช่องลมเมื่อรับประทานอาหาร และเปิดช่องลมเมื่อพูด

9. กรวยคอ หมายถึง โพรงคอที่อยู่ถัดจากปากลงไปจนถึงเส้นเสียง

10. เส้นเสียง หรือ สายเสียง เป็นอวัยวะสำคัญที่เกิดของเสียง เส้นเสียงมีลักษณะเป็นกล้ามเนื้อ 2 แผ่นปิดขวาง อยู่บริเวณปากช่องหลอดลมจากด้านหลังมาด้านหน้า ระหว่างเส้นเสียงจะมีช่องว่าง ซึ่งเป็นทางผ่านให้ลมเข้าถึงปอดและออกมาจากปอดได้ ช่องว่างนี้เรียกว่า ช่องระหว่างเส้นเสียง (Glottis) เส้นเสียงทั้งสองสามารถดึงออกให้ห่างจากกันหรือดึงเข้าหากันได้ ซึ่งเส้นเสียงนี้เป็นส่วนสำคัญที่ทำให้เกิดเสียงพูดขึ้นในภาษา

11. ช่องจมูก หมายถึง โพรงในช่องจมูก ซึ่งอยู่เหนือลิ้นไก่ขึ้นไป เป็นช่องที่ลมซึ่งผ่านเส้นเสียงขึ้นมาจะผ่านออกไปทางจมูกได้เมื่อเวลาหายใจและเวลาออกเสียงนาสิก ในเวลาที่พูดเสียงอื่นลิ้นไก่จะถูกยกขึ้นไปปิดช่องจมูก เพื่อให้ลมออกมาทางปาก

12. เส้นเสียงปลอม เป็นอวัยวะที่มีลักษณะเหมือนเส้นเสียงแต่อยู่เหนือเส้นเสียงขึ้นไป เส้นเสียงปลอมนี้เข้าใจกันว่าจะดึงเข้าหากันเมื่อเวลาพูดเสียงกระซิบ

2.1.2 ลักษณะร่วมของเสียงพูด

เสียงที่ใช้ในภาษาพูดนั้นจะมีลักษณะที่สำคัญบางประการร่วมกัน ซึ่งเรียกได้ว่าเป็นลักษณะร่วมของเสียงพูด ลักษณะที่กล่าวถึงนี้มีอยู่หลายประการ [7] คือ

1. ความก้อง หรือ ไม่ก้องของเสียง

เสียงก้อง หรือ เสียงโฆษะ (Voice)

คือเสียงที่เกิดในขณะที่เส้นเสียงเกิดการตึงตัวหรือเรียกว่าเส้นเสียงปิด เมื่อมีแรงดันให้อากาศไหลผ่านกล่องเสียงในขณะที่เส้นเสียงปิดจะเกิดการสั่นสะบัดของเส้นเสียง เป็นผลให้สัญญาณเสียงที่ได้ (speech waveform) มีลักษณะเป็นคาบ (quasi-periodic) ซึ่งสามารถเรียกความถี่ในการปิด-เปิดของเส้นเสียงนี้ว่า "ความถี่มูลฐาน" (Fundamental Frequency: F_0) ตัวอย่างของเสียงก้องได้แก่ เสียงสระต่างๆ และเสียงพยัญชนะเช่น บ ด ที่เกิดจากการเปล่งเสียงออกทางปาก หรือเสียงพยัญชนะ ม น ง ที่เกิดจากการเปล่งเสียงออกทางจมูก

เสียงไม่ก้อง หรือเสียงอโฆษะ (Unvoice หรือ voiceless)

คือเสียงที่เกิดในขณะที่เส้นเสียงคลายจากการตึงหรือเรียกว่าเส้นเสียงเปิด เมื่อมีแรงดันให้อากาศไหลผ่านกล่องเสียงในขณะที่เส้นเสียงเปิด อากาศที่ไหลผ่านอย่างรวดเร็วจะเกิดการไหลวนและปั่นป่วนทำให้เกิดเสียงที่มีลักษณะเป็นเสียงของสัญญาณรบกวน (Noise) ซึ่งไม่เป็นคาบ ตัวอย่างของเสียงไม่ก้องได้แก่ เสียงพยัญชนะ ฟ ซ ส ฯลฯ หรือเกิดจากการสร้างแรงดันอากาศหลังตำแหน่งปิดกั้นของช่องทางเดินเสียง และเมื่อการปิดกั้นนี้ถูกเปิดออก อากาศจะถูกปล่อยออกมาอย่างทันทีทันใดเกิดเป็นเสียงที่เรียกว่าเสียงระเบิด (Plosive Sound) เช่น การเปล่งเสียงเริ่มแรกของพยัญชนะต้นของคำต่างๆ

2. ความยาวของเสียง (Length)

หมายถึง การที่เสียงใดเสียงหนึ่งเปล่งออกมาได้นานเท่าใด เสียงพูดบางเสียงอาจจะเปล่งออกมาได้ติดต่อกันได้นาน เช่น เสียงสระ เสียงพยัญชนะนาสิก หรือ เสียงพยัญชนะเสียดแทรก

ในภาษาไทย เสียงพูดที่มีความยาว-สั้น ก็มีเพียงเสียงสระเท่านั้น เช่น อะ อิ อุ เป็นเสียงสั้น อา อี อู เป็นเสียงยาวเป็นต้น

3. ระดับเสียงสูง-ต่ำ (Pitch)

เสียงพูดจะมีระดับ สูง หรือ ต่ำ อยู่ที่ความถี่ของเสียง (Fundamental frequency) ถ้าความถี่ต่ำเสียงก็จะต่ำ อวัยวะส่วนที่ทำให้เสียงมีระดับ สูง-ต่ำ คือเส้นเสียง ดังนั้นระดับเสียงสูง-ต่ำก็คืออัตราการสั่นสะบัดของเส้นเสียงนั่นเอง

ในการพูดเสียงที่มีระดับสูง-ต่ำได้คือเสียงก้องเท่านั้นเพราะมีการสั่นสะเทือนของเส้นเสียงที่ทำให้เกิดมีความถี่ระดับต่างๆได้ ในภาษาไทยระดับเสียง สูง-ต่ำ ของคำเราเรียกว่า "วรรณยุกต์"

4. ความดัง (Loundness)

ความดังขึ้นอยู่กับปริมาณของลม ที่ผู้พูดเปล่งเสียงออกมาในช่วงเวลาหนึ่งๆ

5. การลงน้ำหนัก (Stress)

หมายถึง การออกเสียงพยางค์ใดพยางค์หนึ่งให้ดังเน้นมากหรือน้อยกว่าพยางค์อื่นที่อยู่ข้างเคียง (เพื่อต้องการเรียกร้องความสนใจเป็นพิเศษ หรือแสดงอารมณ์อย่างใดอย่างหนึ่ง)

6. ช่วงต่อของเสียง (Juncture)

หมายถึงช่วงระยะที่ผู้พูดเปล่งเสียงหนึ่งแล้วต่อไปเปล่งอีกเสียงหนึ่งซึ่งเรียงกันมาเป็นลำดับ เสียงที่ประกอบกันเข้าเป็นพยางค์จะมีช่วงต่อของเสียงแนบสนิทจนไม่เห็นร่องรอย (close juncture) แต่ถ้าเสียงปรากฏอยู่คนละพยางค์หรือคนละคำ จะมีช่วงต่อ "ห่าง" จนสังเกตได้ชัด (open juncture) ดังนั้นช่วงต่อของเสียง โดยเพาะช่วงต่อห่างจะมีความสำคัญมากในการแบ่งคำในภาษา

2.2 หน่วยเสียงสำคัญในภาษาไทย

"หน่วยเสียง" (phoneme) เป็นหน่วยเล็กที่สุดของภาษา หน่วยดังกล่าวได้แก่เสียงสำคัญๆ ในภาษาใดภาษาหนึ่ง ซึ่งทำหน้าที่ให้ความหมายของคำที่ใช้ในภาษานั้น และทำให้ความหมายของคำนั้นๆมีความหมายแตกต่างจากคำอื่นๆ หน่วยเสียงสำคัญในภาษาไทยมี 3 ประเภทใหญ่ๆคือ เสียงพยัญชนะ เสียงสระ และเสียงวรรณยุกต์ หน่วยเสียงทั้ง 3 นี่เองที่ประกอบกันเข้าเป็นคำที่ใช้ในภาษาไทย

เสียงพูดของมนุษย์ซึ่งมีความแตกต่างกันมากมายนั้นถ้าเราพิจารณาอย่างกว้างๆจะพบว่าสามารถแบ่งออกเป็น 2 ประเภทใหญ่ คือ

1. **เสียงเรียง** (segmental sound) เป็นหน่วยเสียงที่สามารถแยกออกจากเสียงอื่นได้โดยเด็ดขาด เพราะมีลักษณะเด่นเฉพาะตัว ในภาษาไทยได้แก่เสียงสระ และเสียงพยัญชนะ
2. **เสียงซ้อน** (supra-segmental feature) เป็นเสียงที่ทำหน้าที่เป็นส่วนประกอบของเสียงอื่นเพราะไม่สามารถแยกเปล่งเสียงได้ตามลำพัง ในภาษาไทยได้แก่เสียงวรรณยุกต์และทำนองเสียง เป็นต้น

2.3 หน่วยเสียงสระ

2.3.1 ลักษณะของเสียงสระ

ลักษณะสำคัญของเสียงสระก็คือ "เป็นเสียงก้องที่เปล่งเสียงออกมาโดยให้ลมออกทางช่องปากโดยไม่ถูกลิ้นกักหรือขัดขวาง" ดังนั้นเวลาเราออกเสียงสระจะออกเสียงได้สะดวกและออกเสียงได้นาน ทั้งนี้เพราะคุณสมบัติของเสียงสระมีความดังเด่นกว่าเสียงอื่นๆที่เรียงอยู่ข้างเสมอ อวัยวะที่เกี่ยวข้องกับการออกเสียงสระได้แก่ ลิ้น กับริมฝีปาก ถ้าลิ้นส่วนใดทำหน้าที่เพียงส่วนเดียว เสียงที่เกิดขึ้นก็จะมีเพียงเสียงเดียว เสียงเช่นนี้เรียกว่า "สระเดี่ยว" แต่ถ้าลิ้นส่วนอื่นทำหน้าที่ร่วมด้วยเสียงสระนั้นเรียงว่า "สระประสม"

สำหรับภาษาไทยมีหน่วยเสียงสระทั้งหมด 24 หน่วยเสียง แยกออกเป็นสระเดี่ยว 18 หน่วยเสียง และสระประสม 6 หน่วยเสียง [8]

สระเดี่ยว

เสียงสระเดี่ยว 18 หน่วยเสียง พิจารณาการเกิดเสียงได้เป็น 2 กรณีใหญ่ๆ คือ

1. พิจารณาการเกิดจากส่วนต่างๆของลิ้น หมายถึง ลมผ่านส่วนหน้า ส่วนกลาง หรือส่วนหลังของลิ้น
2. พิจารณาการเกิดจากลมผ่านลิ้นในขณะที่ลิ้นอยู่ในระดับ สูง กลาง หรือ ต่ำ

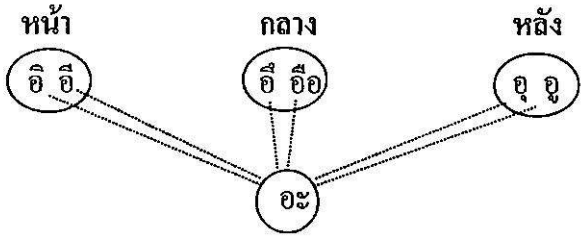
สระ	หน้า	กลาง	หลัง
สูง	อิ อี	อึ ออ	อุ อู
กลาง	เอะ เอ	เออะ เออ	โอะ โอ
ต่ำ	แอะ แอ	อะ อา	เอาะ ออ

นอกจากนี้ หน่วยเสียงสระเดี่ยว 18 หน่วย สามารถแบ่งตามความสั้น-ยาวของการออกเสียงได้เป็น

- สระเดี่ยวเสียงสั้น 9 หน่วย ได้แก่ อะ อิ อึ อุ เอะ แอะ โอะ เอาะ เออะ
- สระเดี่ยวเสียงยาว 9 หน่วย ได้แก่ อา อี ออ อู เอ แอ โอ ออ เออ

สระประสม

เสียงสระประสม 6 หน่วยเสียง เกิดจากลมผ่านกระทบลิ้น 2 ส่วนคือส่วนบนและส่วนล่าง ซึ่งในขณะออกเสียงลิ้นจะอยู่ในระดับสูงแล้วลดลงต่ำ โดยเสียงหลังเป็นเสียงสระ อะ เสมอ ดังแผนผังดังนี้



เสียงสระประสม 6 หน่วยเสียงได้แก่ เอียะ (อิ+อะ) เอีย (อี+อะ) เอือะ (อึ+อะ) เอือ (ออ+อะ) อัวะ (อุ+อะ) อัว (อู+อะ)

2.3.2 หน้าที่ของหน่วยเสียงสระในภาษาไทย

หน่วยเสียงสระในภาษาไทยทั้ง 24 หน่วยเสียงนี้ ทำหน้าที่เป็นแกนกลางของพยางค์หรือคำ กล่าวคือ คำ ทุกคำในภาษาไทยจะต้องมีเสียงสระอยู่ด้วย และเสียงสระในภาษาไทยจะสามารถเกิดกับเสียงพยัญชนะต้นได้ทุกเสียงและสามารถเกิดกับหน่วยเสียงวรรณยุกต์ได้ทุกหน่วย แต่ไม่สามารถเกิดกับหน่วยเสียงพยัญชนะสะกดได้ทุกหน่วย หน่วยเสียงสระที่ทำให้เกิดคำหรือพยางค์ใช้ได้มากที่สุดในภาษามักเป็นหน่วยเสียงสระยาว

2.4 หน่วยเสียงพยัญชนะ

เสียงพยัญชนะในภาษาไทยมีทั้งหมด 21 หน่วยเสียง(44 รูป) ดังแสดงในตารางที่ 2.1 หน่วยเสียงพยัญชนะออกเสียงได้ไม่สะดวกเท่าหน่วยเสียงสระ เพราะเวลาออกเสียงลมหายใจที่พุ่งออกมาจากหลอดลมจะถูกขัดขวางตามส่วนต่างๆของปาก เสียงพยัญชนะจึงออกเสียงให้ยาวนานอย่างเสียงสระไม่ได้ และเสียงพยัญชนะก็ไม่ใช่เสียงก้องเสมอไป

2.4.1 ลักษณะของเสียงพยัญชนะ

หน่วยเสียงพยัญชนะ 21 หน่วยเสียงนี้จำแนกเป็น เสียงก้อง เสียงไม่ก้อง เสียงหนัก เสียงเบา และลักษณะการเกิดเสียง ดังนี้

เสียงก้อง (โฆษะ)มี 9 หน่วยเสียง คือ /ง/ /ย/ /บ/ /ด/ /ม/ /น/ /ร/ /ล/ /ว/

เสียงไม่ก้อง (อโฆษะ) มี 12 หน่วยเสียง คือ /ก/ /ค/ /จ/ /ช/ /ซ/ /ท/ /ต/ /ป/ /พ/ /ฟ/ /อ/ /ฮ/

เสียงหนัก (ธนิต) มี 4 หน่วยเสียง คือ /ค/ /ช/ /ท/ /พ/

เสียงเบา (สถิล) มี 4 หน่วยเสียง คือ /ก/ /จ/ /ต/ /ป/

ตารางที่ 2.1 เสียงพยัญชนะในภาษาไทย

ลำดับที่	อักษรไทยใช้แทนหน่วยเสียง	แทนสัญลักษณ์หน่วยเสียง	
		แบบสากล	แบบไทย
1.	ก	k	ก
2.	ข ฃ ค ฅ ฆ	kh	ค
3.	ง	ŋ	ง
4.	จ (จร_*)	c	จ
5.	ฉ ช ฌ	ch	ช
6.	ญ ย (หย_*) (หญ_*)	j	ย
7.	ซ ศ ษ ส (ทร_*)	s	ซ
8.	ฐ ฑ ฒ ถ ท ธ (ทร_*)	th	ท
9.	บ	b	บ
10.	ฎ ด (ฑ_*)	d	ด
11.	ฏ ต	t	ต
12.	ป	p	ป
13.	ผ พ ภ	ph	พ
14.	ฝ ฟ	f	ฟ
15.	ม (หม_*)	m	ม
16.	น ณ (หน_*)	n	น
17.	ร	r	ร
18.	ล ฬ (หล_*)	l	ล
19.	ว (หว_*)	w	ว
20.	อ	?	อ
21.	ฮ ห	h	ฮ
อักษร 44 รูป		21 หน่วยเสียง	

หมายเหตุ (*) หมายถึง พยัญชนะที่อยู่ในตำแหน่งพยัญชนะต้นแล้วออกเสียงเช่นเดียวกับเสียงพยัญชนะที่อยู่ในลำดับนั้นๆ

ลักษณะของเสียงพยัญชนะ จำแนกตามลักษณะรูปเสียง (classification by form)

1. เสียงกัก หรือ เสียงหยุด (Stop)

เป็นเสียงที่เมื่อผ่านกล่องเสียงเข้ามาถึงช่องปากแล้ว ในปากจะมีฐานกรณ์แห่งใดแห่งหนึ่งกั้นเสียงนี้ไว้ไม่ให้ออกจากปากแต่การกั้นเป็นเพียงชั่วระยะเวลาอันสั้นเท่านั้น แล้วฐานกรณ์ที่กั้นนั้นจะเปิดออก อากาศที่ถูกกักไว้จะถูกปล่อยออกมา เนื่องจากอากาศถูกกั้นไว้เมื่อถูกปล่อยออกมาจึงออกมาในลักษณะระเบิด บางทีจึงเรียกเสียงประเภทนี้ว่าเป็นเสียงระเบิด (plosive sound) มี 9 หน่วยเสียง คือ /ป/ /พ/ /บ/ /ต/ /ท/ /ด/ /ก/ /ค/ /อ/ เสียงพยัญชนะสะกดทุกเสียงในภาษาจะมีลักษณะเป็นเสียงระเบิด หรือเสียงกัก

2. เสียงเสียดแทรก (Fricative)

เป็นเสียงที่เมื่ออากาศผ่านขึ้นมาจากปอดผ่านกล่องเสียงเข้ามาถึงช่องปากแล้วในปากจะมีฐานกรณ์แห่งใดแห่งหนึ่งกั้นอากาศนี้ไว้ แต่การกั้นนี้ไม่สนิทมิดชิดเหมือนเสียงหยุด ยังมีช่องให้อากาศเล็ดลอดแทรกออกมาได้ทำให้เกิดเสียงขณะแทรกออกมา มี 3 หน่วยเสียงคือ /ซ/ /ฟ/ /ฮ/

3. เสียงกึ่งเสียดแทรก (Affricate)

เป็นเสียงในช่องปากที่มีคุณสมบัติเหมือนกับเริ่มต้นด้วยเสียงหยุดและตามด้วยเสียงแทรก มี 2 หน่วยเสียง คือ /จ/ และ /ช/

4. เสียงนาสิก (Nasal)

เป็นเสียงที่เมื่ออากาศผ่านกล่องเสียง ผ่านช่องคอแล้วก็เข้าสู่ช่องจมูก โดยที่ช่องปากมีฐานกรณ์กั้นไว้สนิทไม่ให้อากาศออกทางช่องปาก เสียงที่อากาศผ่านออกมาทางช่องจมูก มี 3 หน่วยเสียง คือ /ม/ /น/ /ง/

5. เสียงข้าง (Lateral)

เป็นเสียงที่อากาศในช่องปากออกสู่ภายนอกปากโดยผ่านทางข้างๆลิ้น มีหน่วยเสียงเดียว คือ /ล/

6. เสียงรัว (Trill)

คือเสียงที่เมื่ออากาศเข้ามาอยู่ในช่องปากแล้วมีการกระดกปลายลิ้นรัวเพดานหลายๆครั้ง มีหน่วยเสียงเดียว คือ /ร/

7. เสียงครึ่งสระ (Semi-Vowel)

คำอธิบายทั่วๆไปของเสียงประเภทนี้ไม่ค่อยชัดแจ้งนัก มักกล่าวว่าตำแหน่งลิ้นเมื่อเริ่มต้นเสียงต่างไปจากตำแหน่งในตอนท้ายๆเสียง เสียงประเภทนี้บางครั้งก็เรียกว่าเสียงครึ่งสระ เพราะมีตำแหน่งลิ้นและปากคล้ายเสียงสระ มี 2 หน่วยเสียง คือ เสียง /ว/ และ /ย/

2.4.2 หน้าที่ของหน่วยเสียงพยัญชนะในภาษาไทย

เสียงพยัญชนะในภาษาไทย 21 หน่วยเสียงนี้สามารถทำหน้าที่ได้ดังนี้

1. เป็นพยัญชนะต้นของพยางค์ คือสามารถนำหน้าเสียงสระในพยางค์หนึ่งๆได้ ในตำแหน่งนี้เสียงพยัญชนะสามารถเกิดได้หน่วยเดียว หรือ สองหน่วยดังนี้
   - เกิดได้ หน่วยเดียว คือ ทำหน้าที่เป็นพยัญชนะต้นเดี่ยว หน่วยเสียงทั้ง 21 หน่วยเสียงนี้สามารถทำหน้าที่เป็นพยัญชนะต้นเดี่ยวได้ทั้งสิ้น
   - เกิดได้ สองหน่วย คือ ทำหน้าที่เป็นพยัญชนะต้นควบ โดยหน่วยเสียงแรกเป็น /ก/ /ค/ /ต/ /ป/ และ /พ/ กับหน่วยเสียงที่สองเป็น /ร/ /ล/ หรือ/ว/
2. เป็นพยัญชนะสะกดของพยางค์ ในตำแหน่งนี้เสียงพยัญชนะในภาษาไทยสามารถเกิดได้ 9 หน่วยเสียง คือ /ป/ (แม่กบ) /ต/ (แม่กด) /ก/ (แม่กก) /ม/ (แม่กม) /ง/ (แม่กง) /น/ (แม่กน) /ย/ (แม่เกย) /ว/ (แม่เกอว) และ ไม่มีเสียงพยัญชนะสะกด (แม่กา)

2.5 หน่วยเสียงวรรณยุกต์

เสียงวรรณยุกต์ คือ ระดับเสียงสูง-ต่ำ ของคำในภาษาไทย เช่นเดียวกับภาษาจีน และภาษาอื่นๆ ที่เป็นภาษาคำโดดซึ่งมีการกำหนดเสียงสูงต่ำไว้ตายตัวในคำแต่ละคำ ถ้าออกเสียงสูง-ต่ำผิดไปความหมายย่อมผิดตามไปด้วย

ในภาษาไทยหน่วยเสียงวรรณยุกต์เป็นหน่วยเสียงสำคัญ ที่ทำให้คำที่มีส่วนประกอบแวดล้อมอื่นๆเหมือนกัน คือมี เสียงพยัญชนะต้น สระ และพยัญชนะสะกดอย่างเดียวกันมีความหมายต่างกัน ดังนั้นอาจกล่าวได้ว่าหน้าที่ของหน่วยเสียงวรรณยุกต์ก็คือ การทำให้เกิดคำขึ้นใช้ในภาษามากขึ้นและเป็นวิธีการสร้างคำขึ้นใช้เพิ่มขึ้นในภาษาเป็นวิธีแรก ทั้งนี้เพราะถ้าเราเปลี่ยนเสียงวรรณยุกต์ก็จะทำให้คำเกิดความหมายเพิ่มขึ้นใหม่นั่นเอง

เสียง สูง-ต่ำ ในภาษาพูด เกิดจากการสั่นสะเทือนของเส้นเสียงในอัตราต่างๆกัน โดยเสียงที่เปล่งออกมาในขณะที่เส้นเสียงสั่นนั้นจะต้องเป็นเสียงก้อง ดังนั้นหน่วยเสียงวรรณยุกต์ในภาษาไทยจึงจัดเป็นหน่วยเสียงซ้อน สัทอักษรที่ใช้จึงเป็นรูปเครื่องหมายเขียนซ้อนข้างบนหน่วยเสียงสระ(ซึ่งเป็นเสียงก้อง) ซึ่งมีรูปวรรณยุกต์อยู่ 4 รูป แทนเสียงวรรณยุกต์ทั้งหมด 5 หน่วยเสียง โดยเสียงสามัญไม่มีรูปวรรณยุกต์

2.5.1 ลักษณะของเสียงวรรณยุกต์

สามารถแบ่งออกตามลักษณะระดับเสียงได้เป็น 2 กลุ่มใหญ่ๆ คือ

1. กลุ่มวรรณยุกต์ระดับ (Level tone) มี 3 หน่วยเสียง คือ

1.1 หน่วยเสียงวรรณยุกต์ระดับต่ำ (Low tone) แทนด้วยสัญลักษณ์ /˧/

คือ เสียงวรรณยุกต์เอก หน่วยเสียงนี้จะปรากฏในพยางค์ของภาษาไทยได้ทุกแบบ

1.2 หน่วยเสียงวรรณยุกต์ระดับกลาง (Mid tone) ไม่มีสัญญลักษณ์

คือ เสียงวรรณยุกต์สามัญ หน่วยเสียงนี้จะไม่ปรากฏในพยางค์ที่มีตัวสะกดเป็นพยัญชนะกัก (พยางค์คำตาย)

1.3 หน่วยเสียงวรรณยุกต์ระดับสูง (High tone) แทนด้วยสัญลักษณ์ /ˊ/

คือ เสียงวรรณยุกต์ตรี หน่วยเสียงนี้จะไม่ปรากฏในพยางค์ที่ประสมด้วยสระเสียงยาว ซึ่งมีตัวสะกดเป็นเสียงกัก

2. กลุ่มวรรณยุกต์เปลี่ยนระดับ (Contour tone) มี 2 หน่วยเสียง คือ

2.1 หน่วยเสียงวรรณยุกต์เปลี่ยนตก (Falling tone) แทนด้วยสัญลักษณ์ /ˆ/

คือ เสียงวรรณยุกต์โท หน่วยเสียงนี้จะไม่ปรากฏในพยางค์ที่มีสระเสียงสั้น และมีเสียงพยัญชนะสะกดเป็นพยัญชนะกัก

2.2 หน่วยเสียงวรรณยุกต์เปลี่ยนขึ้น (Rising tone) แทนด้วยสัญลักษณ์ /ˇ/

คือ เสียงวรรณยุกต์จัตวา หน่วยเสียงนี้จะไม่ปรากฏในพยางค์ที่มีเสียงพยัญชนะสะกดเป็นเสียงกักเลย

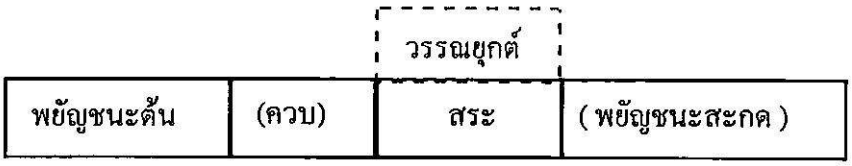
2.6 ลักษณะพยางค์ของคำไทย

2.6.1 คำจำกัดความของพยางค์และคำในภาษาไทย

พยางค์ ในระบบเสียงภาษาไทย หมายถึง "จำนวนเสียงที่ดังเด่นซึ่งปรากฏในกลุ่มเสียงที่เรียงกันเป็นคำพูด ส่วนเสียงอื่นๆที่อยู่ข้างเคียงก็จะประกอบกันเข้าเป็นส่วนของพยางค์" เสียงที่ดังเด่นในกลุ่มเสียงก็คือเสียงสระ ซึ่งมีลักษณะประจำตัวก็คือเป็นเสียงก้องซึ่งดังเด่นกว่าเสียงอื่นๆ ดังนั้นเสียงสระจึงมักเป็นเสียงที่ทำให้เกิดพยางค์ ถ้ามีเสียงสระเด่นอยู่กี่เสียง พยางค์ก็จะมีจำนวนเท่านั้นด้วย

พยางค์ที่เปล่งออกมาครั้งหนึ่งๆ อาจมีความหมายหรือไม่ก็ได้ แต่เมื่อใดพยางค์ที่ประกอบขึ้นจาก เสียงสระ พยัญชนะ และวรรณยุกต์ เป็นอย่างน้อยที่สุด และกลุ่มเสียงเหล่านี้มีความหมายและสามารถปรากฏได้โดยลำพัง พยางค์นั้นๆก็จะกลายเป็นคำในภาษา

คำในภาษาไทยส่วนใหญ่จะเป็นคำพยางค์เดียว ซึ่งเป็นคำพื้นฐาน (Base words) ของภาษา ภาษาไทยจึงจัดอยู่ในตระกูลภาษาคำโดด หรือ คำพยางค์เดียว (Monosyllabic language) หน่วยเสียงที่ประกอบกันเข้าเป็นพยางค์จะต้องมีอย่างน้อย 3 หน่วย คือ หน่วยเสียงพยัญชนะต้น 1 หน่วย หน่วยเสียงสระ 1 หน่วย และ หน่วยเสียงวรรณยุกต์ 1 หน่วย และมีหน่วยเสียงอย่างมากไม่เกิน 5 หน่วย คือเพิ่มหน่วยเสียงพยัญชนะต้นที่เป็นเสียงควบกล้ำอีก 1 หน่วย และหน่วยเสียงพยัญชนะสะกดอีก 1 หน่วย โดยมีองค์ประกอบของหน่วยเสียงต่างๆในพยางค์ แสดงได้ดังรูปที่ 2.2



รูปที่ 2.2 องค์ประกอบของพยางค์ในภาษาไทย

2.6.2 ลักษณะโครงสร้างของคำพยางค์เดียวต่อการผันของเสียงวรรณยุกต์

เราต้องตระหนักเสมอว่ารูปวรรณยุกต์ในคำภาษาไทยบางครั้งไม่แสดงเสียงให้เห็นในการเขียนเสมอไป ทั้งนี้การกำหนดเสียงวรรณยุกต์ขึ้นอยู่กับลักษณะของพยางค์ว่าเป็นคำเป็น หรือ คำตาย

ลักษณะโครงสร้างของคำพยางค์เดียวในภาษาไทยมี 5 แบบ ซึ่งลักษณะโครงสร้างที่ต่างกันของพยางค์จะมีผลต่อการผันของเสียงวรรณยุกต์ ดังแสดงในตารางที่ 2.2

ตารางที่ 2.2 แสดงลักษณะของคำพยางค์เดียวในภาษาไทย

เสียงวรรณยุกต์ / โครงสร้างพยางค์	สามัญ	เอก	โท	ตรี	จัตวา
1. พ (พ) ส ส $^{0-4}$	+	+	+	+	+
2. พ (พ) ส น $^{0-4}$	+	+	+	+	+
3. พ (พ) ส ส น $^{0-4}$	+	+	+	+	+
4. พ (พ) ส ก ^{1,3}	−	+	−	+	−
5. พ (พ) ส ส ก ^{1,2}	−	+	+	−	−

หมายเหตุ + หมายถึงโครงสร้างพยางค์สามารถผันระดับเสียงวรรณยุกต์นั้นได้

− หมายถึงโครงสร้างพยางค์ไม่สามารถผันระดับเสียงวรรณยุกต์นั้นได้

เมื่อกำหนดให้ พ แทนหน่วยเสียงพยัญชนะต้น 1 หน่วย

พพ แทนหน่วยเสียงพยัญชนะต้น 2 หน่วยควบกัน หรือพยัญชนะต้นควบ โดยหน่วยเสียงที่ 2 คือ ร /r/ ล /l/ หรือ ว /w/

ส แทนหน่วยเสียงสระเดี่ยวสั้น

สส แทนหน่วยเสียงสระเดี่ยวยาว และหน่วยเสียงสระประสม

น แทนหน่วยเสียงพยัญชนะสะกดที่เป็นพยัญชนะนาสิก / m, n, ŋ / และครึ่งสระ / j, w /

ก แทนหน่วยเสียงสะกดที่เป็นพยัญชนะกัก / p, t, k, ? /

0 แทนหน่วยเสียงวรรณยุกต์ สามัญ

1 แทนหน่วยเสียงวรรณยุกต์ เอก

2 แทนหน่วยเสียงวรรณยุกต์ โท

3 แทนหน่วยเสียงวรรณยุกต์ ตรี

4 แทนหน่วยเสียงวรรณยุกต์ จัตวา

และจากจำนวนอักษร 44 รูป ในภาษาไทยได้แบ่งเพื่อสะดวกต่อการผันเสียงวรรณยุกต์ เป็นอักษรไตรยางค์ ดังได้แสดงไว้ในตารางที่ 2.3

ตารางที่ 2.3 อักษรไตรยางค์

อักษรไตรยางค์		รูปวรรณยุกต์			
		เอก	โท	ตรี	จัตวา
อักษรสูง	ข ฃ ฉ ถ ฐ ผ ฝ ศ ษ ส ห	+	+	–	–
อักษรกลาง	ก จ ด ฎ ต ฏ บ ป อ	+	+	+	+
อักษรต่ำ-คู่	ค ฆ ฅ ช ฌ ฑ ท ธ ฒ พ ภ ฟ ซ ฮ	+	+	–	–
ต่ำ-เดี่ยว	ม น ง ณ ย ญ ร ล ฬ ว	+	+	–	–

อักษรสูง 11 ตัว ผันวรรณยุกต์ได้ 3 เสียง เช่น ขา ข่า ข้า

อักษรกลาง 9 ตัว ผันวรรณยุกต์ได้ครบทั้ง 5 เสียง เช่น จา จ่า จ้า จ๊า จ๋า

อักษรต่ำ 24 ตัว ผันวรรณยุกต์ได้ 3 เสียง เช่น ทา ท่า ท้า

การผันอักษรต่ำนี้มีข้อสังเกต คือถ้ามีรูปวรรณยุกต์เอกจะเป็นเสียงวรรณยุกต์โท ถ้ารูปวรรณยุกต์โทจะเป็นเสียงวรรณยุกต์ตรี นอกจากนี้อักษรต่ำยังแบ่งออกเป็นอักษรต่ำคู่ 14 ตัวและอักษรต่ำเดี่ยวอีก 10 ตัว เพื่อประโยชน์ในการผันเสียงวรรณยุกต์คือ เมื่อนำคำที่เป็นอักษรต่ำคู่มาผันร่วมกับคำที่เป็นอักษรสูงจะเกื้อกูลกันทำให้การผันเสียงวรรณยุกต์ทำได้ครบทั้ง 5 เสียง ดังตัวอย่างในตารางที่ 2.4

ตารางที่ 2.4 ตัวอย่างการผันเสียงอักษรต่ำคู่ กับอักษรสูง

เสียงวรรณยุกต์				
สามัญ	เอก	โท	ตรี	จัตวา
คา	ข่า	ค่า ข้า	ค้า	ขา

อักษรต่ำที่ใช้คู่กับอักษรสูง แล้วทำให้ผันเสียงวรรณยุกต์ได้ครบทั้ง 5 เสียง มี 7 คู่ ดังตารางที่ 2.5

ตารางที่ 2.5 การจับคู่ในการผันเสียงวรรณยุกต์

คู่ที่	อักษรสูง	อักษรต่ำ
1	ข ฃ	ค ฆ ฅ
2	ฉ	ช ฌ
3	ถ ฐ	ฑ ท ธ ฒ
4	ผ	พ ภ
5	ฝ	ฟ
6	ศ ษ ส	ซ
7	ห	ฮ
	11 ตัว	14 ตัว

ส่วนอักษรต่ำเดี่ยวอีก 10 ตัวนั้นการที่จะทำให้ผันเสียงได้ครบนั้นจะนำตัว "ห" มาช่วยก็จะทำให้ผันเสียงวรรณยุกต์ได้ครบ เช่น นา หน่า น่า น้า หนา

บทที่ 3
การหาค่าความถี่มูลฐานของสัญญาณเสียงพูด

3.1 กล่าวนำ

ระดับเสียงสูงต่ำในภาษา หรือในภาษาไทยเรียกว่าเสียงวรรณยุกต์นั้น เกิดจากการสั่นสะบัดเป็นจังหวะของเส้นเสียงในการออกเสียงก้อง ซึ่งคุณสมบัติที่สำคัญของเสียงก้องก็คือมีความเป็นคาบ และระดับเสียงจะสูงหรือต่ำนั้นสามารถสังเกตได้จากค่าความถี่ในการเกิดคาบที่เรียกว่าพิทช์นั่นเอง ซึ่งความถี่ในการเกิดพิทช์นี้เรียกว่าความถี่มูลฐาน โดยถ้ามีการสั่นสะบัดของเส้นเสียงอย่างรวดเร็วความถี่มูลฐานจะมีค่ามากเสียงที่เกิดขึ้นจะเป็นเสียงสูง ในทำนองเดียวกันถ้าความถี่มูลฐานมีค่าน้อยระดับเสียงที่เกิดขึ้นก็จะเป็นเสียงต่ำ ในภาษาไทยระดับเสียงสูง-ต่ำที่แตกต่างกันนี้มีผลต่อความหมายของคำในภาษา ดังนั้นพิทช์หรือค่าความถี่มูลฐานนี้จึงเป็นสิ่งสำคัญในการแยกแยะคำในภาษาไทย

3.2 การวิเคราะห์ในโดเมนเวลา

สัญญาณเสียงพูดเป็นสัญญาณที่เปลี่ยนแปลงไปตามเวลา โดยเกิดในลักษณะแบบสุ่ม (random) แต่ก็ขึ้นกับการควบคุมเสียงของผู้พูดด้วยเพราะเสียงที่เปล่งออกมาในระยะเวลาหนึ่งนั้นจะขึ้นอยู่กับรูปทรงของช่องทางเดินเสียง(vocal tract) และลักษณะการสั่นของเส้นเสียง(vocal cord) เสียงพูดจึงเป็นสัญญาณที่มีคาบเวลาชั่วขณะ(quisi-periodic) คือมีความเป็นคาบคงที่ในเวลาอันสั้นและมีการเปลี่ยนแปลงในช่วงระหว่างเวลานั้น ดังนั้นในการวิเคราะห์จึงต้องทำการแบ่งเสียงพูดออกเป็นช่วงๆ(Frame) โดยมีช่วงเวลาอยู่ระหว่าง 10-30 มิลลิวินาที ในช่วงเวลาดังกล่าวถือว่าเสียงจะมีการเปลี่ยนแปลงคุณสมบัติน้อยมาก ดังนั้นในแต่ละเฟรมจึงสมมติให้เสียงเป็นสัญญาณที่มีคุณลักษณะคงที่ ซึ่งทำให้การวิเคราะห์ทำได้ง่ายขึ้น

3.3 ทฤษฎีการประมาณค่าพิทช์โดยใช้ออโตคอร์รีเลชั่นฟังก์ชั่น

การวิเคราะห์โดยใช้ออโตคอร์รีเลชั่น[9]-[10] เป็นวิธีหนึ่งที่เป็นที่ยอมรับในการใช้ตรวจหาคาบพิทช์ โดยออโตคอร์รีเลชั่นฟังก์ชั่นจะทำหน้าที่ในการแสดงยอดกราฟหลัก (prominent peak) ที่เป็นคาบพิทช์ในแต่ละส่วนของเสียง(section) ซึ่งสามารถหาได้จากรายละเอียดของโครงสร้างของสัญญาณนั้นๆ

3.3.1 การจัดแบ่งการวิเคราะห์สัญญาณออกเป็นช่วงสั้นๆ (Short–Time Autocorrelation Analysis)

ถ้ากำหนดให้ discrete time signal แทนด้วย x(m) ออโตคอร์รีเลชั่นฟังก์ชั่น ของ discrete-time deterministic signalโดยทั่วๆไปเขียนได้เป็น

$$\phi(k) = \sum_{m=-\infty}^{\infty} x(m)\,x(m+k) \qquad (3.1)$$

ซึ่งออโตคอร์รีเลชั่น ฟังก์ชั่น ของสัญญาณ โดยพื้นฐานก็คือการแปลงสัญญาณ (transformation) ดังนั้นการตรวจวัดค่าคาบพิทช์ สามารถทำได้โดย

ถ้าสัญญาณ x(m) มีความเป็นคาบที่แน่นอนด้วยระยะ P นั่นคือ

$$x(m) = x(m+P) \qquad ; \text{สำหรับทุก } m$$

ดังนั้นสามารถเขียนได้ว่า

$$\phi(k) = \phi(k+p) \qquad (3.2)$$

นั่นคือ ออโตคอร์รีเลชั่นฟังก์ชั่นก็มีความเป็นคาบด้วยระยะคาบเดียวกัน หรือในทางกลับกันก็คือ "ความเป็นคาบในออโตคอร์รีเลชั่นฟังก์ชั่น เป็นตัวบ่งชี้ให้เห็นถึงความเป็นคาบในสัญญาณ"

คุณสมบัติของออโตโครีเลชั่นฟังก์ชั่นที่สำคัญ คือ

1. เป็นฟังก์ชั่นคู่ นั่นคือ $\phi(k) = \phi(-k)$
2. มีค่ามากที่สุดที่ $k = 0$ นั่นคือ $|\phi(k)| \leq \phi(0)$; สำหรับทุกค่า k

ถ้าพิจารณาสมการที่ 3.2 ควบคู่ไปกับคุณสมบัติในข้อ 1 และ 2 จะพบความเป็นคาบของสัญญาณ โดยตัวอย่างสัญญาณข้อมูลของออโตโครีเลชั่นจะมีค่ามากที่สุดที่ $0, \pm p, \pm 2p, \ldots$ โดยไม่ต้องคำนึงถึงเวลาเริ่มต้น (time origin) ของสัญญาณ การคำนวณหาคาบของสัญญาณสามารถประมาณได้จากตำแหน่งแรกที่มีค่ามากที่สุดในออโตโครีเลชั่น ฟังก์ชั่น ซึ่งจากคุณสมบัติเหล่านี้ทำให้ออโตโครีเลชั่นฟังก์ชั่นเป็นหลักการพื้นฐานที่น่าสนใจในการใช้ประมาณค่าความเป็นคาบในสัญญาณทุกชนิด

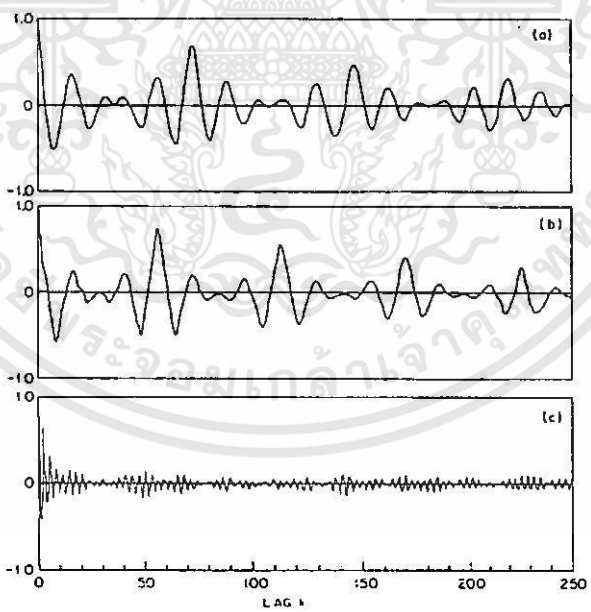
สำหรับสัญญาณที่มีลักษณะเปลี่ยนแปลงอยู่ตลอดเวลา เช่นสัญญาณเสียงพูดจะต้องทำการแบ่งสัญญาณออกเป็นช่วงสั้นๆเพื่อหาสารสนเทศ (information)ที่ต้องการ โดย short-time auto-correlation function สามารถนิยามได้เป็น

$$R(k) = \sum_{m=0}^{S-1-k} x(m)\,x(m+k) \qquad (3.3)$$

เมื่อ S คือ จำนวนตัวอย่างสัญญาณ (sample)ต่อเฟรม

k คือ จำนวนจุดที่ใช้ในการคำนวณ ออโตอร์รีเลชั่น

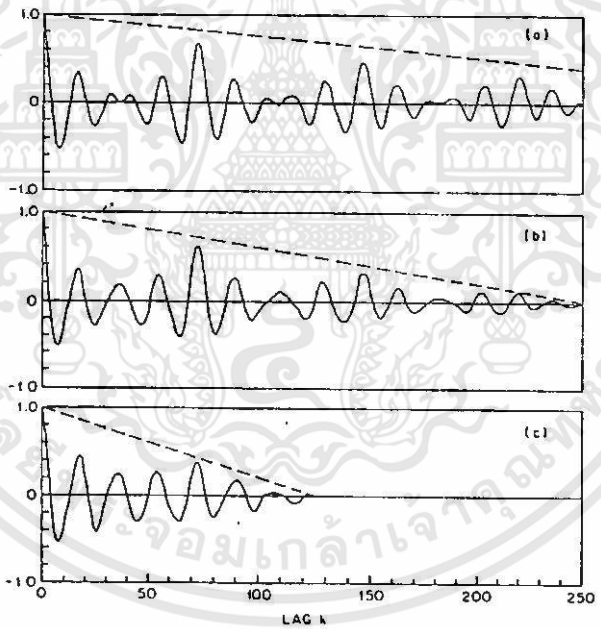
โดยในการกำหนดช่วงของค่า k จะกำหนดจากค่า k ที่น้อยที่สุดคือช่วงดาบเวลาพิทช์ของเสียงผู้หญิงซึ่งมีค่าเท่ากับ 3 มิลลิวินาที จนถึงค่า k สูงสุดเท่าที่เป็นไปได้คือช่วงดาบเวลาพิทช์ของเสียงผู้ชายซึ่งมีค่าเท่ากับ 20 มิลลิวินาที ดังนั้นถ้าใช้อัตราการชักตัวอย่างที่ 10 KHz ก็จะใช้ k อยู่ในช่วง 30 ถึง 200 และจากการแบ่งสัญญาณออกเป็นเฟรม จำนวนตัวอย่างสัญญาณในเฟรมซึ่งก็คือค่า S จะต้องมีค่ามากกว่า k มากๆ (S>>k) ซึ่งจะกล่าวถึงต่อไป



รูปที่ 3.1 ออโตโครีเลชั่น ฟังก์ชั่น (a),(b) สัญญาณเสียงก้อง และ (c) สัญญาณเสียงไม่ก้อง โดยทั้ง 3 กรณีใช้ S = 401

พิจารณารูป 3.1 แสดงตัวอย่างการคำนวณออโตโครีเลชั่น ฟังก์ชั่นของสัญญาณเสียงพูดที่มีอัตราการชักตัวอย่างด้วยความถี่ 10 KHz โดยใช้สมการคำนวณที่ 3.3 ด้วย S = 401 และค่าการเลื่อนของเวลา (lag) เป็น $0 \le k \le 250$ รูป 3.1(a-b) เป็นส่วนของสัญญาณเสียงที่มีความเป็นคาบ และรูป 3.1(c) คือส่วนของสัญญาณที่ไม่มีความเป็นคาบ จากรูปบน (a) ตำแหน่งสูงสุด (peak) เกิดที่ตำแหน่ง 72 นั่นคือสัญญาณมีคาบที่ระยะ 7.2 msec หรือมีค่าความถี่มูลฐานประมาณ 140 Hz (10 KHz / 72) ในรูป (b) ค่าสูงสุดของออโตโครีเลชั่นเกิดในตำแหน่งที่ 58 แสดงให้เห็นว่ามีค่าเฉลี่ยของคาบในช่วง 5.8 msec ส่วนรูป (c) เป็นส่วนของสัญญาณที่ไม่มีความเป็นคาบ ออโตโครีเลชั่นฟังก์ชั่นจะประกอบด้วยองค์ประกอบของความถี่สูงคล้ายๆรูปคลื่นของสัญญาณรบกวน (Noise-like waveform)

ในการเลือกค่าของจำนวนตัวอย่าง (S) ที่ใช้ในแต่ละเฟรม รูปคลื่นสัญญาณจะต้องมีความเป็นคาบที่สมบูรณ์ (complete period) อย่างน้อย 2-3 คาบ ซึ่งในความเป็นจริงแล้วความยาวของสัญญาณเสียงพูดมีผลต่อการคำนวณของ R(k) เนื่องจากค่าของ R(k) จะลดลงเรื่อยๆเมื่อ k มีค่าเพิ่มมากขึ้น ซึ่งมีผลโดยตรงต่อแอมปลิจูดสูงสุด (peak) ของโครีเลชั่นเนื่องจากจะมีค่าลดลงเช่นกัน



รูปที่ 3.2 ออโตโครีเลชั่น ฟังก์ชั่น สำหรับเสียงก้องโดยใช้ค่า S ที่แตกต่างกันคือ

(a) S = 401; (b) S = 251 และ (c) S = 125

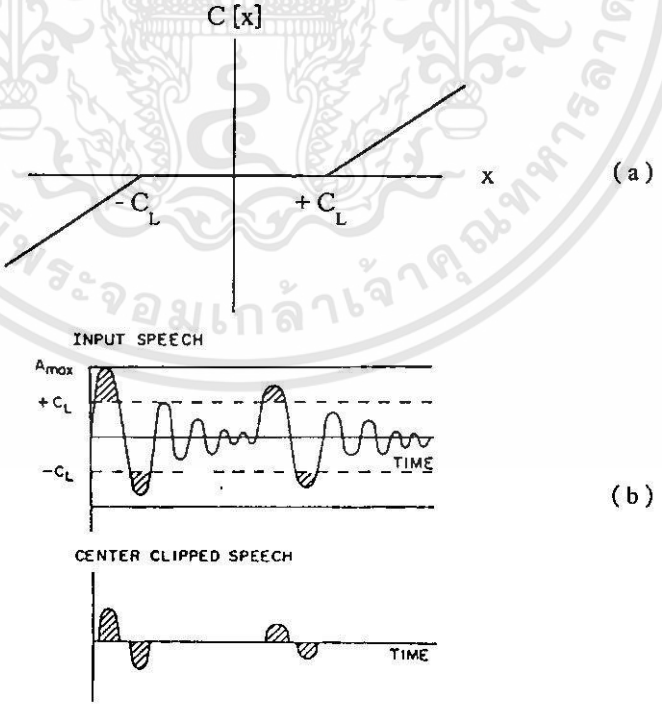
จากรูป 3.2(a) และ3.2(b) จากการคำนวณพบว่าตำแหน่งคาบจริงอยู่ในตำแหน่งที่ 72 ส่วนในรูป 3.2(c) จะพบว่าค่าสูงสุดของโครีเลชั่นอยู่ในตำแหน่งที่ 15 ทั้งนี้เนื่องจากวินโดว์ที่ใช้มีขนาดสั้นเกินไปเมื่อเทียบกับขนาดของคาบพิทช์ (pitch period) จึงทำให้ตำแหน่งสูงสุดของโครีเลชั่นที่คำนวณได้ผิดพลาดไปจากตำแหน่งจริง

3.3.2 การกำจัดผลของโครงสร้างฟอร์แมนท์ด้วยวิธีเซนเตอร์คลิปปิ้ง

จากตัวอย่างรูป 3.2 จะเห็นว่าออโตโครีเลชั่น ฟังก์ชั่นมียอดของกราฟจำนวนมาก ซึ่งยอดของกราฟเหล่านี้เป็นผลมาจากผลตอบสนองทางความถี่ที่เกิดในช่องทางเดินเสียง (vocal tract response) ซึ่งมีผลต่อรูปทรงในแต่ละคาบของสัญญาณเสียงพูด โดยในรูป 3.2(c) ตำแหน่งสูงสุดของออโตโครีเลชั่นผิดไปจากตำแหน่งคาบจริงเนื่องจากวินโดว์มีขนาดสั้นไปเมื่อเทียบกับคาบพิทช์ แต่ในขณะเดียวกันการเปลี่ยนแปลงอย่างรวดเร็วของความถี่ฟอร์แมนท์ก็มีผลให้เกิดปรากฏการณ์ในลักษณะเดียวกันนี้ด้วยเช่นกัน ซึ่งในกรณีนี้ยอดสูงสุดของออโตโครีเลชั่นที่เกิดเนื่องจากผลตอบสนองทางความถี่ในช่องทางเดินเสียงจะมีขนาดใหญ่กว่ายอดกราฟที่เกิดจากความเป็นคาบของแหล่งกำเนิดเสียง (vocal excitation) ซึ่งเหตุการณ์ลักษณะเช่นนี้จะทำให้การเลือกตำแหน่งยอดกราฟที่สูงที่สุดของออโตโครีเลชั่น ฟังก์ชั่น เกิดการผิดพลาดด้วย

ดังนั้นเพื่อที่จะหลีกเลี่ยงปัญหานี้ จึงได้มีการเสนอกรรมวิธีเพื่อที่จะจัดการสัญญาณให้ความเป็นคาบของสัญญาณนี้เด่นชัดขึ้นโดยการขจัดลักษณะของสัญญาณที่จะทำให้เกิดความไข้วเขว (distracting) ออกไป เทคนิคนี้เรียกว่า การทำสเปกตรัมราบเรียบ (spectrum flatteners) ซึ่งมีอยู่หลายวิธี [11] เซนเตอร์คลิปปิ้งก็เป็นวิธีหนึ่งที่สะดวกและสามารถคำนวณได้จากสัญญาณโดยตรง ซึ่งถูกเสนอขึ้นโดยใช้การแปลงสัญญาณแบบไม่เป็นเชิงเส้น (nonlinear)

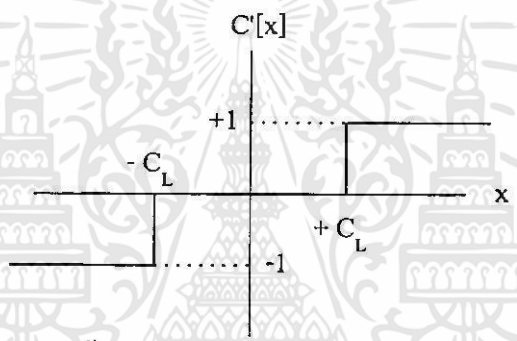
$$y(n) = C[x(m)] \tag{3.4}$$



รูปที่ 3.3 (a) ฟังก์ชั่น เซนเตอร์ คลิปปิ้ง (b) ตัวอย่างแสดงการคลิปสัญญาณเสียงพูด

เมื่อ C[x] แสดงได้ดังรูป 3.3(a) วิธีนี้อาศัยหลักการคือ สัญญาณเสียงพูดจะถูกนำมาหาค่าแอมปลิจูดสูงสุด A_{max} เพื่อนำมากำหนดระดับในการคลิปสัญญาณ (clipping level : C_L) จากนั้นค่าของสัญญาณที่มีระดับต่ำกว่าระดับคลิปปิ้งจะถูกกำหนดให้มีค่าเป็นศูนย์ ส่วนสัญญาณที่มีระดับสูงกว่าระดับคลิปปิ้งจะถูกลบออกด้วยระดับคลิปปิ้ง ดังรูป 3.3(b) จะพบว่าสัญญาณยังคงความเป็นคาบของสัญญาณเดิม แต่ส่วนของสัญญาณที่เกิดจากอิทธิพลของโครงสร้างฟอร์แมนท์(อันเนื่องมาจากการตอบสนองทางความถี่ภายในช่องทางเดินเสียง)จะถูกกำจัดออกไป แต่ในการกำหนดระดับการคลิปสัญญาณจะต้องระมัดระวังว่าระดับที่กำหนดจะต้องไม่สูงเกินไปจนทำให้สารสนเทศสูญหาย

จากวิธีดังกล่าวจะเห็นว่า วิธีเซนเตอร์คลิปปิ้งเป็นวิธีที่สะดวกในการทำให้สเปคตรัมราบเรียบ ซึ่งต่อมาได้มีการพัฒนาวิธีการนี้บนอุปกรณ์ดิจิตอล โดยทำการปรับปรุงฟังก์ชั่นของเซนเตอร์คลิปปิ้งให้ง่ายต่อการคำนวณ แสดงได้ดังรูป 3.4

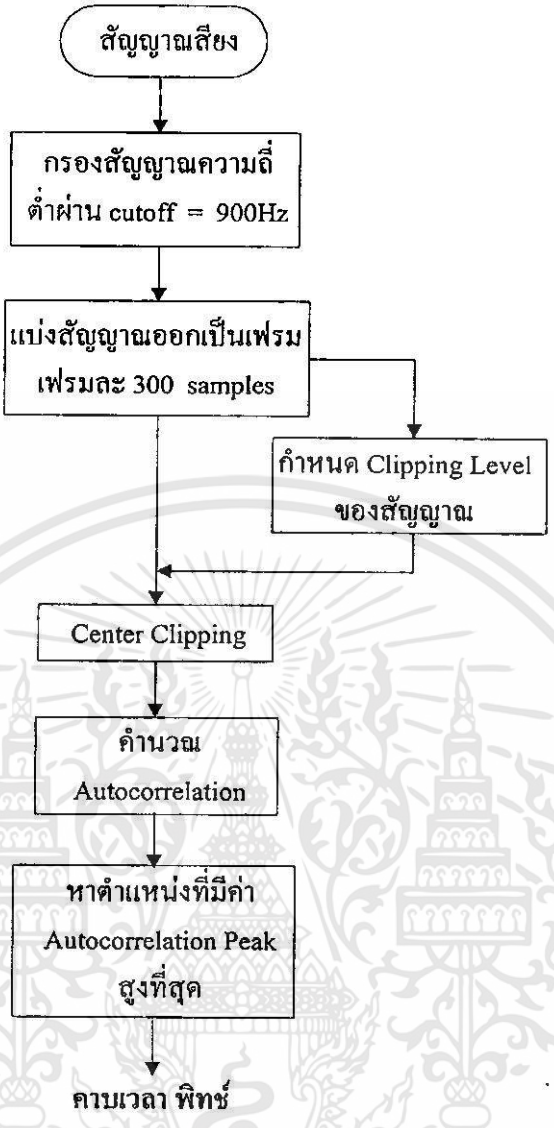


รูปที่ 3.4 ฟังก์ชั่น เซนเตอร์คลิปปิ้ง แบบ 3ระดับ

โดยสัญญาณที่ผ่านการคลิปจะมีค่าเป็น +1 ถ้า $x(m) > C_L$, –1 ถ้า $x(m) < -C_L$ และ มีค่าเป็น 0 ถ้า $-C_L \le x(m) \le C_L$ ฟังก์ชั่นนี้เรียกว่า เซนเตอร์คลิปปิ้งแบบ 3 ระดับ (3-level center clipping) การกำหนดค่าในลักษณะนี้จะช่วยลดความซับซ้อนในการคำนวณออโตโครีเลชั่นฟังก์ชั่นลง เนื่องจากแต่ละพจน์ในสมการ (3.3) อยู่ในรูปของ $x(m)x(m+k)$ และ ค่า $x(m)$ จะมีค่าได้เพียง 3 ค่า คือ +1, 0, -1 เท่านั้น ดังนั้นผลคูณในสมการ(3.3) จึงสามารถมีค่าได้เป็น

$$\begin{aligned} x(m)x(m+k) &= 0 \quad \text{ถ้า} \quad x(m) = 0 \ \text{หรือ} \ x(m+k) = 0 \\ &= +1 \quad \text{ถ้า} \quad x(m) = x(m+k) \\ &= -1 \quad \text{ถ้า} \quad x(m) \neq x(m+k) \end{aligned} \tag{3.5}$$

โดยอัลกอริธึมในการหาค่าพิทช์ที่ถูกพัฒนาขึ้น สามารถแสดงได้ดังรูปที่ 3.5



รูปที่ 3.5 อัลกอริธึม การหาค่าพิทช์

รายละเอียดของขั้นตอนมีดังนี้ คือ

1. นำสัญญาณเสียงพูดที่ได้จากการชักตัวอย่าง มาผ่านตัวกรองความถี่ต่ำผ่านที่มีความถี่คัทออฟประมาณ 900 Hz เพื่อทำการกำจัดอิทธิพลของโครงสร้างความถี่ฟอร์แมนท์ ที่จะเกิดบนออโตโครีเลชั่นฟังก์ชั่น
2. แบ่งสัญญาณออกเป็นเฟรม มีขนาดเฟรมละ 300 ตัวอย่างเพื่อทำการวิเคราะห์ โดยในการเลื่อนเฟรมกำหนดให้มีส่วนของเฟรมซ้อนทับกัน 2 ใน 3 ส่วน
3. ทำการแบ่งข้อมูลในเฟรมออกเป็นส่วนๆ ส่วนละ 100 ตัวอย่าง โดยในส่วนแรกและส่วนที่สามจะถูกนำมาหาค่าแอมปลิจูดสมบูรณ์ที่มีค่าสูงที่สุดของแต่ละส่วน เพื่อนำมากำหนดระดับคลิปปิ้ง โดยเลือกจากค่าแอมปลิจูดที่น้อยกว่าคูณกับเปอร์เซนต์ที่กำหนดขึ้น

สามารถคำนวณหาระดับคลิปปิ้งได้จากสมการต่อไปนี้

$$C_L = (68\%) \times \min(K_1, K_2) \tag{3.6}$$

โดยที่ C_L = Cliping Level

K_1 = Absolute Amplitude Peak ของ 100 samples แรก ของเฟรม

K_2 = Absolute Amplitude Peak ของ 100 samples ท้าย ของเฟรม

68% = เปอร์เซนต์ที่กำหนดขึ้น (อยู่ภายในช่วง 30-80%)

4. เมื่อกำหนดระดับคลิปปิ้งแล้ว ค่าของสัญญาณอินพุทจะถูกกำหนดใหม่โดยใช้วิธีเซนเตอร์ คลิปปิ้งแบบ 3 ระดับ โดยสัญญาณที่มีค่าอยู่ในช่วง $\pm C_L$ จะถูกกำหนดให้มีค่าเป็นไปตามความสัมพันธ์ดังนี้

$$y(m) = \text{sgn}[x(m)] = \begin{cases} 1 \ , \ x(m) \geq C_L \\ 0 \ , |x(m)| < C_L \\ -1 \ , \ x(m) \leq -C_L \end{cases} \tag{3.7}$$

เมื่อ sgn[x(m)] คือ สัญญาณที่ผ่านการคลิป จากนั้นนำค่าที่กำหนดใหม่นี้ไปทำการคำนวณออโตโครีเลชั่นฟังก์ชั่นเพื่อทำการหาคาบพิทช์ของสัญญาณเสียง

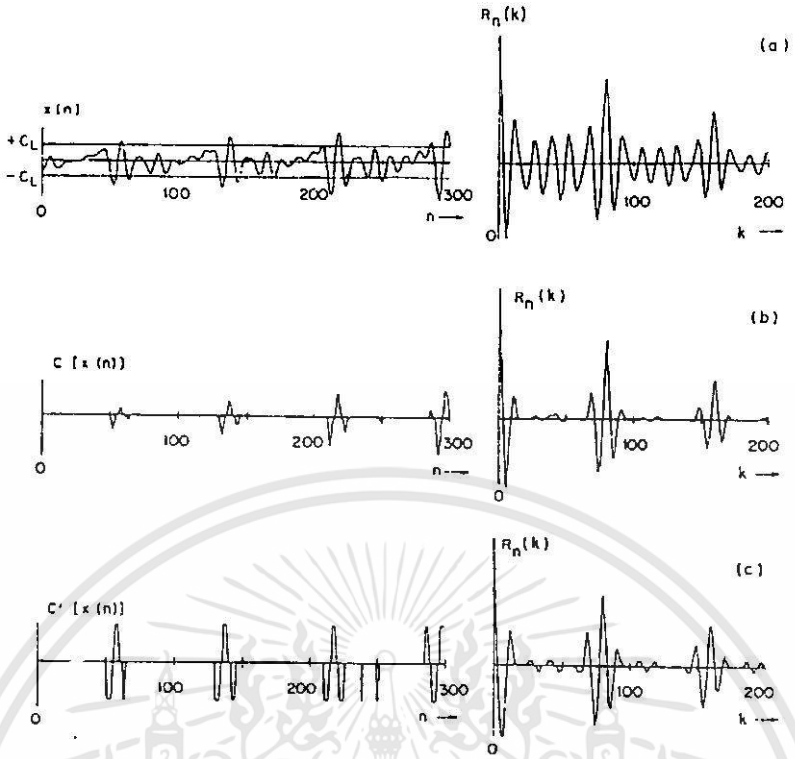
5. จากค่าคาบเวลาพิทช์ที่ได้นี้สามารถนำมาหาค่าความถี่มูลฐาน F_0 ได้จากความสัมพันธ์ คือ

$$F_0 = \frac{F_s}{P} \tag{3.8}$$

เมื่อ F_0 = ความถี่มูลฐาน (Hz)

F_s = ความถี่ที่ใช้ในการชักตัวอย่างสัญญาณ

P = คาบเวลา พิทช์



รูปที่ 3.6 ตัวอย่างสัญญาณเสียงและฟังก์ชั่น โครีเลชั่น

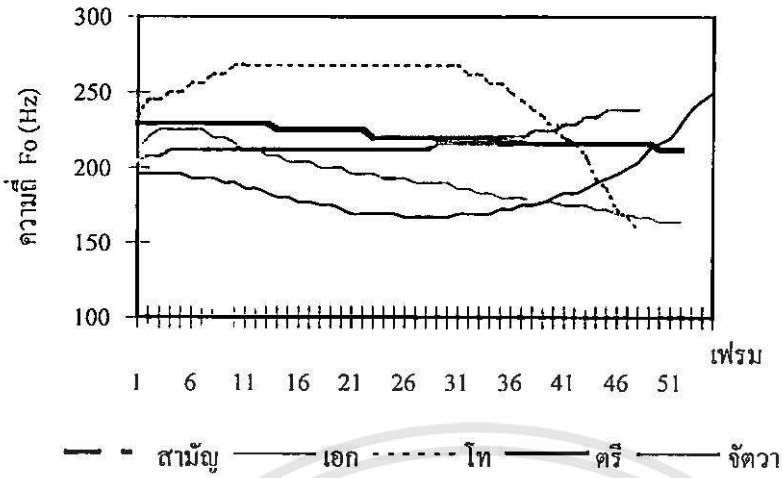
(a) ไม่มีการคลิปสัญญาณ

(b) คลิปสัญญาณโดยใช้ เซนเตอร์ คลิปปิ้ง

(c) คลิปสัญญาณโดยใช้ เซนเตอร์ คลิปปิ้ง แบบ 3 ระดับ

จากรูป3.6(a) แสดงผลของการคำนวณออโตโครีเลชั่นจากสัญญาณที่ไม่ผ่านเซนเตอร์คลิปปิ้ง จะเห็นว่ารูปกราฟประกอบด้วยจุดยอดจำนวนมากอันเกิดเนื่องจากผลตอบสนองทางความถี่ของช่องทางเดินเสียง ส่วนรูป3.6(b,c) จะสังเกตห็นว่าสัญญาณที่ผ่านเซนเตอร์คลิปปิ้งจะเหลืออยู่แต่สัญญาณที่มีค่าคาบพิทช์ และผลที่ได้จากการคำนวณออโตโครีเลชั่นจะมีจุดยอดที่จะทำให้เกิดความสับสนเหลืออยู่น้อยมาก(ในตัวอย่างนี้ใช้ระดับการคลิปที่ 68% ของแอมปลิจูดที่สูงสุดในช่วง 100 ตัวอย่างแรก)

ระดับเสียงของคำในพยางค์หนึ่งๆ จะมีระดับสูงหรือต่ำนั้นสามารถสังเกตได้จากค่าพิทช์หรือค่าความถี่มูลฐาน โดยระดับเสียงของคำในภาษาไทยจะมีระดับเสียงวรรณยุกต์ใดนั้นสามารถสังเกตได้จากแนวทางการเปลี่ยนแปลงของค่าความถี่มูลฐานเมื่อเทียบกับเวลา ดังแสดงในรูปที่ 3.7



รูปที่ 3.7 ค่าความถี่มูลฐานของคำ อา อ่า อ้า อ๊า อ๋า จากผู้ออกเสียงเพศหญิง

จากรูป3.7 แสดงการเปลี่ยนแปลงค่าความถี่มูลฐานที่ได้จากการคำนวณออโตโครีเลชั่นที่ผ่านกระบวนการเซนเตอร์คลิปปิ้ง จะเห็นว่าในแต่ละระดับเสียงวรรณยุกต์จะมีแนวทางการเปลี่ยนแปลงของค่าความถี่มูลฐานที่มีลักษณะเฉพาะตัวแตกต่างกัน

3.4 สรุป

เนื้อหาในบทนี้กล่าวถึงการหาค่าความถี่มูลฐาน ซึ่งเป็นตัวบ่งบอกถึงระดับเสียงสูง-ต่ำของคำในภาษา โดยสัญญาณข้อมูลจะถูกนำมาผ่านการทำสเปกตรัมราบเรียบด้วยวิธี เซนเตอร์คลิปปิ้ง แล้วทำการประมาณคาบพิทช์ด้วยวิธีออโตโครีเลชั่น จากนั้นค่าคาบพิทช์จะถูกแปลงให้อยู่ในรูปของค่าความถี่มูลฐาน โดยรูปแบบการเปลี่ยนแปลงของค่าความถี่มูลฐานเมื่อเทียบกับเวลา จะเป็นตัวบ่งบอกถึงระดับเสียงวรรณยุกต์ที่แตกต่างกันของคำหรือพยางค์ในภาษาไทย ซึ่งลำดับของค่าความถี่มูลฐานที่แตกต่างกันนี้จะถูกนำไปเข้าสู่กระบวนการเตรียมข้อมูล เพื่อใช้เป็นข้อมูลฝึกสอนในการสร้างแบบจำลองอ้างอิงการรู้จำระดับเสียงวรรณยุกต์ทั้ง 5 ระดับ ดังจะกล่าวถึงในบทต่อไป

บทที่ 4
การเตรียมข้อมูลเพื่อสร้างแบบจำลอง

4.1 กล่าวนำ

การจดจำเสียงพูด เป็นลักษณะหนึ่งของการจดจำรูปแบบ (Pattern Recognition) คือจะเป็นการเปรียบเทียบระหว่างแบบทดสอบ(Test Pattern) กับแบบอ้างอิง(Reference Pattern) ซึ่งเป็นรูปแบบที่ทราบและเก็บไว้ล่วงหน้า

ขั้นตอนในการจดจำแบ่งเป็น 2 ขั้นตอน ดังนี้

1. ขั้นตอนการเรียนรู้ (Learning)

จะเป็นการสร้างกลุ่มของแบบอ้างอิงในการจดจำเสียงพูด ในขั้นตอนนี้จะทำการวิเคราะห์เสียงพูดก่อน โดยดึงลักษณะของพารามิเตอร์ที่ต้องการออกมา ซึ่งในวิทยานิพนธ์นี้ก็คือแนวทางการเปลี่ยนแปลงของค่าความถี่มูลฐานของเสียง จากนั้นทำการจัดกลุ่มพารามิเตอร์โดยใช้การควอนไตซ์ข้อมูล เพื่อนำไปสร้างแบบจำลองอ้างอิงในการรู้จำต่อไป

2. ขั้นตอนการจดจำ (Recognition)

จะเป็นการทดสอบการจดจำระหว่างแบบอ้างอิงกับแบบทดสอบ โดยจะทำการเปรียบเทียบพารามิเตอร์ของแบบทดสอบกับแบบอ้างอิงทั้งหมด แบบอ้างอิงที่เลือกคือ แบบอ้างอิงที่มีพารามิเตอร์ใกล้เคียงกับแบบทดสอบที่สุด

สัญญาณเสียงพูดที่ผ่านขั้นตอนในการหาค่าความถี่มูลฐานแล้ว ก่อนที่จะถูกนำมาเป็นเสียงต้นแบบเพื่อใช้ในกระบวนการสร้างแบบจำลองอ้างอิง หรือใช้เป็นแบบทดสอบ จะต้องนำมาผ่านกระบวนการในการเตรียมข้อมูลเสียก่อน เพื่อที่จะขจัดข้อจำกัดอันเนื่องมาจากความถี่มูลฐานที่แตกต่างกันระหว่างผู้ออกเสียงที่เป็นชายและหญิง โดยมีวัตถุประสงค์เพื่อให้แบบจำลองอ้างอิงที่สร้างขึ้นนี้สามารถใช้งานร่วมกันได้ไม่ว่าผู้ออกเสียงจะเป็นชายหรือหญิง ซึ่งกระบวนการเตรียมข้อมูลมี 2 ขั้นตอน คือ ขั้นตอนการปรับปรุงความต่อเนื่องของข้อมูลด้วยวิธีการกรองมัธยฐาน และขั้นตอนการควอนไตซ์ทิศทางการเปลี่ยนแปลงของค่าความถี่มูลฐาน โดยรายละเอียดในแต่ละขั้นตอนมีดังนี้

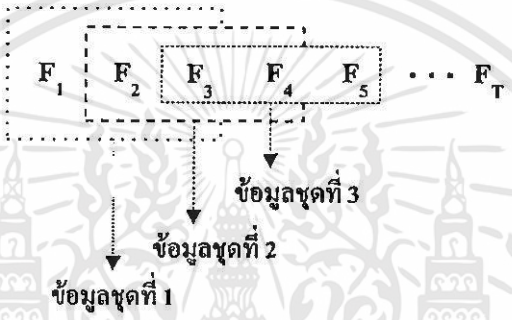
4.2 การปรับปรุงความต่อเนื่องของข้อมูลด้วยวิธีการกรองมัธยฐาน (Median Filtering)

สัญญาณเสียงที่ผ่านขั้นตอนในการหาค่าความถี่มูลฐานแล้ว อาจมีความไม่ต่อเนื่องของลำดับความถี่เกิดขึ้น เนื่องจากความไม่ต่อเนื่องของสัญญาณเสียงในช่วงต้นของการออกเสียงพูด

และจากการปัดเศษในการคำนวณ ดังนั้น ขั้นตอนแรกของการเตรียมข้อมูลก็คือ นำสัญญาณเสียงที่ผ่านขั้นตอนในการหาค่าความถี่มูลฐานแล้วมาผ่านตัวกรองมัธยฐาน เพื่อปรับปรุงให้ข้อมูลมีความต่อเนื่องเพิ่มขึ้น [12] โดยลำดับของความถี่มูลฐานซึ่งเป็นข้อมูลอินพุทจะอยู่ในรูปของข้อมูล 1 มิติ ขนาด [1xT] เมื่อ T คือจำนวนเฟรมของสัญญาณเสียง

ขั้นตอนการทำงานของการกรองมัธยฐาน

ทำการจัดเรียงค่าความถี่มูลฐานออกเป็นชุดข้อมูล โดยในแต่ละชุดข้อมูลประกอบด้วยค่าความถี่ 3 ค่า โดยกำหนดให้มีการเลื่อนของชุดข้อมูลแสดงได้ดังรูป 4.1



รูปที่ 4.1 การจัดแบ่งความถี่มูลฐานออกเป็นชุดข้อมูล

เมื่อ F_1 = ค่าความถี่มูลฐานของเฟรมที่ 1

F_2 = ค่าความถี่มูลฐานของเฟรมที่ 2

F_T = ค่าความถี่มูลฐานของเฟรมสุดท้าย

จากนั้นนำค่าความถี่ทั้ง 3 ค่า ในแต่ละชุดข้อมูลมาจัดเรียงใหม่ตามความสัมพันธ์

$$a \leq b \leq c \tag{4.1}$$

โดยที่

a = ความถี่ F_0 ที่มีค่าน้อยที่สุดของแต่ละชุดข้อมูล

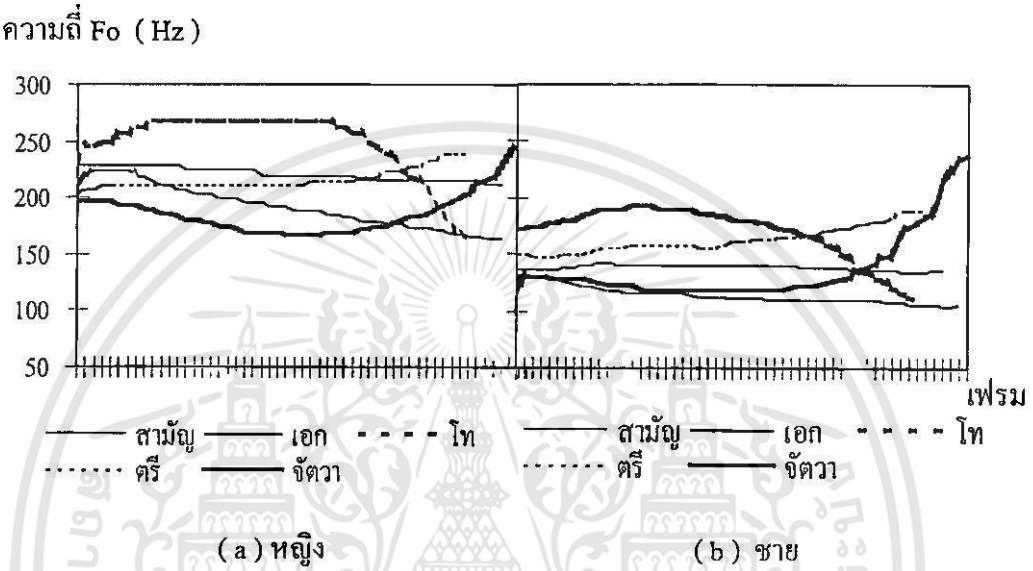
b = ความถี่ F_0 ที่มีค่าอยู่ระหว่างกลาง

c = ความถี่ F_0 ที่มีค่ามากที่สุดของแต่ละชุดข้อมูล

จากนั้นนำความถี่ค่ากลาง (b) ที่ได้จากชุดข้อมูลแต่ละชุดมาจัดเรียงตามลำดับ ก็จะได้ความถี่มูลฐานชุดใหม่ที่ผ่านกระบวนการกรองมัธยฐานแล้ว

4.3 การควอนไตซ์ทิศทางการเปลี่ยนแปลงของค่าความถี่มูลฐาน

ขั้นตอนสุดท้ายของการเตรียมข้อมูลก็คือ การควอนไตซ์ทิศทางการเปลี่ยนแปลงของค่าความถี่มูลฐาน จากข้อเท็จจริงที่ว่าระดับความถี่มูลฐานของเสียงชายและหญิงมีความแตกต่างกัน ซึ่งโดยเฉลี่ยแล้ว ในผู้ชายความถี่มูลฐานจะมีค่าอยู่ในช่วง 80-160 Hz และ 160-400 Hz ในผู้ออกเสียงที่เป็นหญิง [13] ดังตัวอย่างในรูป 4.2



รูปที่ 4.2 แสดงระดับความถี่มูลฐานที่แตกต่างกันระหว่าง (a) หญิง และ (b) ชาย

จากรูปจะสังเกตเห็นว่าลักษณะการเปลี่ยนแปลงของค่าความถี่มูลฐานในแต่ละระดับเสียงวรรณยุกต์จะมีรูปแบบการเปลี่ยนแปลงที่มีลักษณะเฉพาะโดยไม่ขึ้นกับผู้ออกเสียงว่าเป็นเพศใด

ดังนั้นในวิทยานิพนธ์นี้จึงได้ดึงเอาลักษณะเด่นนี้มาใช้ในการสร้างแบบจำลองการรู้จำระดับเสียงวรรณยุกต์ โดยทำการจัดกลุ่มค่าความถี่มูลฐานออกเป็น 3 ระดับตามแนวทางการเปลี่ยนแปลงของความถี่ (ΔF) ที่เพิ่มขึ้นหรือลดลงเมื่อเวลาเปลี่ยนไป

โดย

$$\Delta F_t = F_{t+1} - F_t \qquad (4.2)$$

เมื่อ $t = 1, 2, \ldots, (T-1)$ โดย T คือ จำนวนเฟรม

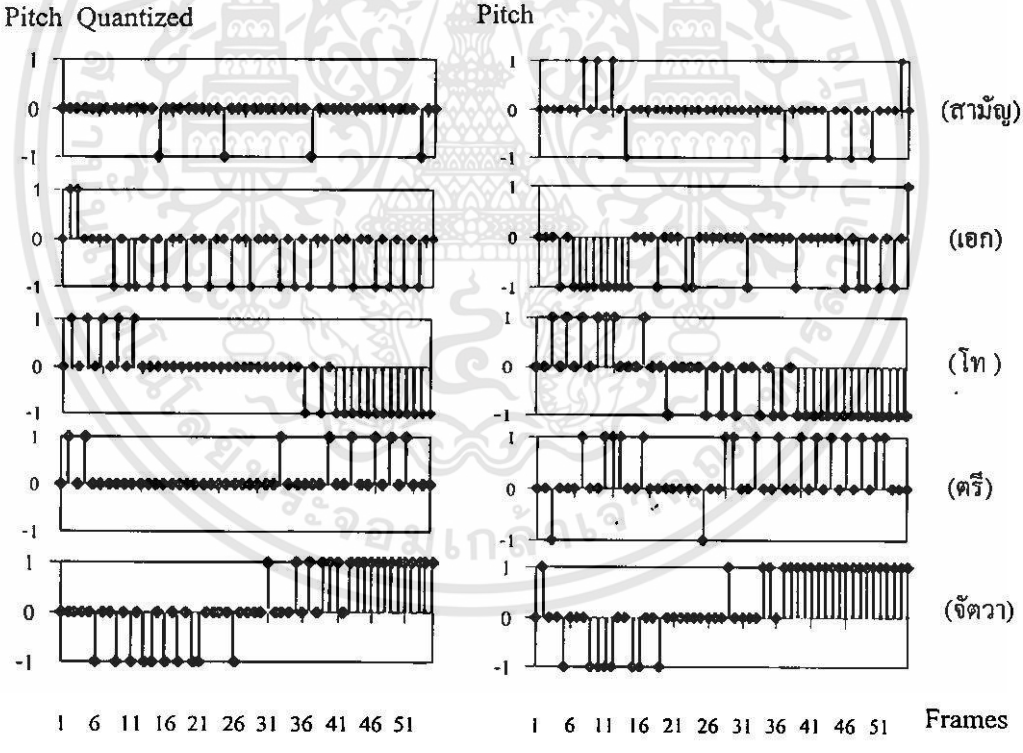
F_t = ความถี่ F_0 ที่เวลา t

F_{t+1} = ความถี่ F_0 ที่เวลา t+1

จากนั้นทำการควอนไตซ์ ΔF โดยแบ่งออกเป็น 3 ระดับตามทิศทางการเปลี่ยนแปลงของความถี่มูลฐาน โดยกำหนดให้

$$V_t = \begin{cases} 1 \; ; \; \Delta F_t > 0 \\ 0 \; ; \; \Delta F_t = 0 \\ -1 \; ; \; \Delta F_t < 0 \end{cases} \qquad (4.3)$$

จากนั้นค่า $V_t = \{-1, 0, 1\}$ จะถูกนำไปใช้เป็นข้อมูลฝึกสอน (trainning) เพื่อใช้ในการสร้างแบบจำลองอ้างอิงของเสียงวรรณยุกต์ต่อไป ซึ่งจะเห็นว่าการควอนไตซ์ความถี่ออกเป็น 3 ระดับนี้ นอกจากจะขจัดข้อจำกัดของความถี่มูลฐานที่แตกต่างกันระหว่าง ชาย, หญิงแล้ว ยังช่วยลดเนื้อที่ของหน่วยความจำในการจัดเก็บข้อมูล และทำให้การคำนวณทำได้เร็วขึ้นเมื่อเทียบกับการใช้ช่วงความถี่มูลฐานทั้งหมดมาสร้างแบบจำลอง



รูปที่ 4.3 แสดงการจัดแบ่งค่าความถี่มูลฐานออกเป็น 3 ระดับตามทิศทางการเปลี่ยนแปลงต่อเวลาที่เพิ่มขึ้น จากผู้ออกเสียงที่เป็น (a) หญิง และ (b) ชาย

เมื่อนำค่าความถี่มูลฐานในรูป 4.2 มาทำการจัดระดับค่าการเปลี่ยนแปลงความถี่ออกเป็น 3 ระดับจะแสดงได้ดังรูปที่ 4.3 และเมื่อพิจารณาแนวทางการเปลี่ยนแปลงในระดับเสียงวรรณยุกต์ทั้ง 5 ระดับ จะพบว่า

1. เสียงสามัญ ตลอดทั้งเสียง ความถี่มูลฐานมีการเปลี่ยนแปลงลดลงเล็กน้อย
2. เสียงเอก ความถี่มูลฐานของเสียงจะมีค่าลดลงอย่างต่อเนื่อง
3. เสียงโท ความถี่มูลฐานมีค่าเพิ่มขึ้นในช่วงแรก และลดลงอย่างต่อเนื่องในช่วงท้ายของเสียง
4. เสียงตรี ความถี่มูลฐานมีแนวโน้มเพิ่มขึ้น
5. เสียงจัตวา ความถี่มูลฐานมีค่าลดลงในช่วงแรก และเพิ่มขึ้นอย่างต่อเนื่องในช่วงท้ายของเสียง

4.4 สรุป

สัญญาณเสียงพูดที่ผ่านขั้นตอนในการหาค่าความถี่มูลฐานแล้ว ก่อนที่จะถูกนำมาเป็นเสียงต้นแบบเพื่อใช้ในกระบวนการสร้างแบบจำลองอ้างอิง หรือใช้เป็นแบบทดสอบ จะต้องนำมาผ่านกระบวนการในการเตรียมข้อมูลเสียก่อน โดยขั้นแรก ลำดับข้อมูลจะต้องนำมาผ่านการกรองมัธยฐาน เพื่อปรับปรุงให้ข้อมูลมีความต่อเนื่องเพิ่มขึ้น จากนั้นจะทำการควอนไตซ์ข้อมูลออกเป็น 3 ระดับตามทิศทางการเปลี่ยนแปลงที่เพิ่มขึ้น คงที่ หรือลดลงของค่าความถี่มูลฐาน เพื่อที่จะขจัดข้อจำกัดอันเนื่องมาจากความถี่มูลฐานที่แตกต่างกัน ทำให้แบบจำลองอ้างอิงที่ถูกสร้างขึ้นนี้สามารถใช้ร่วมกันได้กับผู้ออกเสียงที่เป็นทั้งชายและหญิง อีกทั้งยังเป็นการลดเนื้อที่หน่วยความจำในการจัดเก็บข้อมูล และลดเวลาที่ใช้ในการคำนวณ โดยข้อมูลเอาท์พุทที่ได้จากการควอนไตซ์ออกเป็น 3 ระดับนี้ จะถูกใช้เป็นข้อมูลฝึกสอน หรือข้อมูลทดสอบ ของการสร้างแบบจำลองการรู้จำระดับเสียงวรรณยุกต์ทั้ง 5 ระดับด้วย ฮิดเดน มาร์คอฟ โมเดล ซึ่งจะกล่าวถึงในบทต่อไป

บทที่ 5
การสร้างแบบจำลองการรู้จำด้วยวิธี Hidden Markov Model

5.1 กล่าวนำ

แบบจำลองมาร์คอฟเป็นแบบจำลองทางสถิติซึ่งพัฒนามาเพื่อแบ่งกลุ่มของอนุกรมทางเวลา หรือสัญญาณที่ไม่คงที่ นั่นคือใช้สำหรับจัดกลุ่มของสัญญาณที่ไม่รู้จัก (Unknown signal) ให้ไปอยู่ในกลุ่มใดกลุ่มหนึ่งของสัญญาณ ซึ่งแบบจำลองมาร์คอฟได้ถูกนำมาประยุกต์ใช้ในการรู้จำเสียงพูด [14] และเป็นวิธีการที่วิทยานิพนธ์นี้เลือกใช้

แบบจำลองมาร์คอฟ แบ่งออกเป็น 2 ประเภท คือ แบบต่อเนื่อง(Continuous) และแบบไม่ต่อเนื่อง(Discrete-time) ในวิทยานิพนธ์นี้ได้เลือกใช้แบบไม่ต่อเนื่อง เพราะคุณลักษณะของข้อมูลที่ผ่านการควอนไตซ์ ซึ่งใช้เป็นข้อมูลอินพุท มีลักษณะเป็นชนิดไม่ต่อเนื่อง โดยเนื้อหาในบทนี้จะกล่าวถึงทฤษฎีที่ใช้ในการสร้างแบบจำลองการรู้จำจากเสียงต้นแบบ และขั้นตอนในการทดสอบการรู้จำ

5.2 ส่วนประกอบของแบบจำลองมาร์คอฟ

พารามิเตอร์สำคัญที่เกี่ยวข้องในการสร้างแบบจำลองอ้างอิง ที่ต้องรู้จักได้แก่

1. T คือ ความยาวของลำดับข้อมูลที่ได้จากการควอนไตซ์ค่าความถี่มูลฐาน ซึ่งมีขนาดความยาวของลำดับเท่ากับจำนวนเฟรมทั้งหมดในเสียงแต่ละเสียง ซึ่งจะใช้เป็นข้อมูลอินพุทในส่วนของ HMM โดยต่อไปจะเรียกแทนว่า "ลำดับของค่าปรากฏ"(Observation sequence)
2. N คือ จำนวนสเตทในแบบจำลอง ถ้ากำหนดให้เซ็ตของสเตทเป็น $\{ 1 , 2 , \ldots , N \}$ จะสามารถแทนสเตทที่เปลี่ยนไปตามเวลา t ด้วย เซ็ตของ $Q = \{ q_1 , q_2 , \cdots , q_N \}$
3. M คือจำนวนของค่าปรากฏที่สามารถเป็นไปได้ต่อหนึ่งสเตท แทนสัญลักษณ์ ด้วย $V = \{ v_1 , v_2 , \ldots v_M \}$ ซึ่งจากการจัดระดับของการเปลี่ยนแปลงของความถี่ (ΔF_t) ออกเป็น 3 ระดับ จะได้เซ็ตของค่าปรากฏที่สามารถเป็นไปได้ในแต่ละสเตทมีค่าเป็น $V = \{-1 , 0 , 1\}$
4. ค่าความน่าจะเป็นในการย้ายสเตท : $A = \{ a_{ij} \}$

   โดย a_{ij} แทนการย้ายสเตทจาก i ไป j

   เมื่อ

$$a_{ij} = P [q_t = j | q_{t-1} = i] \quad ; 1 \le i , j \le N \tag{5.1}$$

5. การกระจายความน่าจะเป็น ของค่าปรากฏที่สามารถเป็นไปได้ภายในสเตท : $B = \{b_j(k)\}$

โดยที่ $b_j(k) = P[v_k \text{ ที่เวลา } t \mid q_j \text{ ที่เวลา } t] \quad ; 1 \le k \le M$ (5.2)

เป็นนิยามการกระจายสัญลักษณ์ในสเตท j เมื่อ j = 1, 2, ..., N

6. ค่าความน่าจะเป็นของการเป็นสเตทเริ่มต้น : $\pi = \{ \pi_i \}$

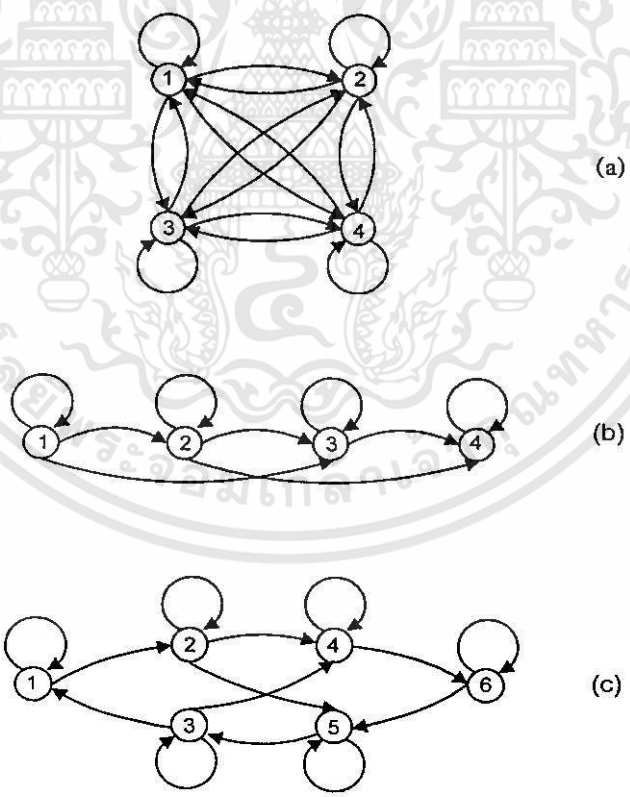
เมื่อ $\pi_i = P[q_i \text{ ที่เวลา } t = 1] \quad ; 1 \le i \le N$ (5.3)

จะเห็นว่า Hidden Markov Model ต้องการพารามิเตอร์ของแบบจำลองคือ N, M และ กลุ่มของความน่าจะเป็น A, B, π ดังนั้นในการแสดงเซ็ตของพารามิเตอร์ที่สมบูรณ์ของแบบจำลองอ้างอิง จะแทนด้วยสัญลักษณ์

$$\lambda = (A, B, \pi) \quad (5.4)$$

5.3 ชนิดของ HMM

แบ่งชนิดตามการย้ายสเตทของเมตริกซ์ A



รูปที่ 5.1 แบบจำลองชนิดต่างๆของ HMM

1. HMM แบบ Egordic Model หรือ Fully Connected Model

การย้ายสเตทสามารถย้ายไปยังทุกๆสเตทของแบบจำลอง ดังรูปที่ 5.1(a) เป็นตัวอย่างของแบบจำลองที่มี N = 4 ซึ่งจากรูปนี้มีค่าของเมตริกซ์ A เป็น

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

2. HMM แบบ Left-Right Model หรือ Bakis Model

การย้ายสเตทจะย้ายจากซ้ายไปขวาซึ่งจะมีคุณสมบัติของสัมประสิทธิ์ในการย้ายสเตทดังนี้

$$a_{ij} = 0 \qquad , j < i$$

คือจะไม่มีการย้ายสเตทไปยังสเตทที่ต่ำกว่าสเตทปัจจุบัน และนอกจากนี้ก็ยังมีความน่าจะเป็นของสเตทเริ่มต้นดังนี้

$$\pi_i = \begin{cases} 0 & , i \neq 1 \\ 1 & , i = 1 \end{cases}$$

คือลำดับของสเตทจะต้องเริ่มที่สเตทที่ 1 เสมอ และ Left-Right Model นี้มักมีกฎบังคับการย้ายสเตท เพื่อไม่ให้มีการเปลี่ยนแปลงดัชนีของสเตทมากนัก กล่าวคือ

$$a_{ij} = 0 \qquad , j > i + \Delta i$$

ดังรูปที่ 5.1(b) ค่าของ $\Delta i = 2$ คือจะไม่มีการย้ายข้ามสเตทไปเกิน 2 สเตท และมีเมตริกซ์ในการย้ายสเตทเป็น

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

จะเห็นว่าสเตทสุดท้าย สัมประสิทธิ์การย้ายสเตทจะเป็น

$$a_{NN} = 1$$

$$a_{Ni} = 0 \qquad , i < N$$

แบบจำลองแบบนี้จะเหมาะกับสัญญาณที่มีลักษณะเปลี่ยนแปลงตามเวลาอย่างต่อเนื่อง เช่น เสียงพูด

3. HMM แบบ parallel Left-Right Model

เป็นแบบจำลองที่มีความยืดหยุ่นมากกว่าแบบที่ 2 แสดงได้ดังรูปที่ 5.1 (c)

5.4 ปัญหาพื้นฐานของแบบจำลอง มาร์คอฟ

ปัญหาของ HMM มี 3 ข้อ ซึ่งต้องใช้อัลกอริธึมวิธีต่างๆในการคำนวณเพื่อแก้ปัญหานี้

ปัญหาที่ 1 เมื่อมีลำดับของค่าปรากฏ $O = \{O_1 O_2 O_3 \ldots O_T\}$ และมีแบบจำลอง $\lambda = (A,B,\pi)$ จะคำนวณหาค่าความน่าจะเป็น $P(O|\lambda)$ ของลำดับค่าปรากฏนั้นได้อย่างไร

ปัญหาที่ 2 เมื่อมีลำดับของค่าปรากฏ $O = \{O_1 O_2 O_3 \ldots O_T\}$ และแบบจำลอง $\lambda = (A,B,\pi)$ จะคำนวณหาลำดับสเตท $q = \{q_1 q_2 q_3 \ldots q_T\}$ ที่เหมาะสมกับลำดับค่าปรากฏนั้นได้อย่างไร

ปัญหาที่ 3 เราจะปรับพารามิเตอร์ของแบบจำลอง $\lambda = (A,B,\pi)$ เพื่อให้ได้ค่า $P(O|\lambda)$ สูงสุดได้อย่างไร

การคำนวณเพื่อแก้ปัญหาของ HMM

การแก้ปัญหาที่ 1 เป็นการคำนวณหาว่าแบบจำลอง λ ใดๆ มีโอกาสจะให้ค่าลำดับเป็นไปตามลำดับของค่าปรากฏนั้น ด้วยค่าของความน่าจะเป็นมาก-น้อยเท่าใด

การแก้ปัญหาสามารถทำได้โดยระบุสเตทให้กับลำดับของค่าปรากฏซึ่งยาว T (โดยที่ค่าปรากฏหนึ่งตัวมีความเป็นไปได้ที่จะอยู่ในสเตทได้ N สเตท) ซึ่งสามารถเป็นไปได้ถึง N^T แบบ ให้สเตทต่างๆแทนด้วย

$$q = q_1 q_2 q_3 \cdots q_T \tag{5.5}$$

เมื่อ q_1 เป็นสเตทเริ่มต้นที่เวลา $t = 1$ ความน่าจะเป็นของลำดับของค่าปรากฏ O ที่กำหนดคือ

$$P(O|q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda) \tag{5.6a}$$

ความน่าจะเป็นในการเกิดค่าปรากฏคือ

$$P(O|q,\lambda) = b_{q_1} O_1 \cdot b_{q_2} O_2 \cdot \ldots \cdot b_{q_T} O_T \tag{5.6b}$$

และ ความน่าจะเป็นในการย้ายข้ามสเตท q จะเป็น

$$P(q|\lambda) = \pi_{q_1} \bullet a_{q_1 q_2} \bullet a_{q_2 q_3} \bullet \dots \bullet a_{q_{T-1} q_T} \tag{5.7}$$

ดังนั้นเมื่อนำความน่าจะเป็นของการเกิดค่าปรากฏ O และค่าความน่าจะเป็นในการย้ายสเตท q มารวมกัน ซึ่งนั่นก็คือความน่าจะเป็นที่ O และ q จะเกิดขึ้นพร้อมกัน จะได้

$$P(O, q|\lambda) = P(O|q, \lambda) P(q|\lambda) \tag{5.8}$$

$$= (b_{q_1} O_1 \bullet b_{q_2} O_2 \bullet \dots \bullet b_{q_T} O_T)(\pi_{q_1} \bullet a_{q_1 q_2} \bullet a_{q_2 q_3} \bullet \dots \bullet a_{q_{T-1} q_T})$$

โดยที่ความน่าจะเป็นของ O ได้มาจากผลรวมของความน่าจะป็นที่ O และ q เกิดขึ้นพร้อมกันโดยคิดจากทุกสเตท q ที่จะเป็นไปได้ ดังนี้

$$P(O|\lambda) = \sum_{\text{all } q} P(O|q, \lambda) P(q|\lambda) \tag{5.9}$$

$$= \sum_{q_1 q_2 \dots q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \dots a_{q_{T-1} q_T} b_T O_T \tag{5.10}$$

ที่เวลาเริ่มต้น (t = 1) เราจะอยู่ที่สเตท q_1 ด้วยค่าความน่าจะเป็น π_{q_1} และแทนค่าความน่าจะเป็นในการเกิดค่าปรากฏ O_1 ที่สเตทนี้ด้วย $b_{q_1} O_1$

ที่เวลา เพิ่มขึ้นจาก $t \rightarrow t+1$ (t=2) เราแทนการย้ายสเตทจากสเตท q_1 ไปยัง q_2 ด้วยค่าความน่าจะเป็น $a_{q_1 q_2}$ และแทนค่าความน่าจะเป็นในการเกิดค่าปรากฏเป็น O_2 ด้วยค่าความน่าจะเป็น $b_{q_2} O_2$

จนกระทั่ง ที่เวลา T เราแทนการย้ายสเตทจากสเตท q_{T-1} ไปยัง q_T ด้วยค่าความน่าจะเป็น $a_{q_{T-1} q_T}$ และแทนค่าความน่าจะเป็นในการเกิดค่าปรากฏเป็น O_T ด้วยค่าความน่าจะเป็น $b_{q_T}(O_T)$

จะเห็นว่าสมการนี้มีการคำนวณที่ยุ่งยากเนื่องจากมีการคูณกันเป็นจำนวนมากในรูปของลำดับ $2T*N^T$ ดังนั้นจึงมีการคิดหาวิธีมาช่วย ซึ่งแบ่งออกเป็น

1. กระบวนการไปข้างหน้า (Forward Procedure) ; $\alpha_t(i)$ = Forward variable

นิยาม

$$\alpha_t(i) = P(O_1 O_2 \dots O_T, q_t = i|\lambda) \tag{5.11}$$

คือ ความน่าจะเป็นของการเกิดลำดับของค่าปรากฏ $O_1 O_2 \ldots O_T$ และอยู่ที่สเตท q_i ณ เวลา t โดยมีแบบจำลองเป็น λ เราสามารถหา $\alpha_t(i)$ ได้ดังนี้

1. <u>การเริ่มต้น (Initialization)</u>

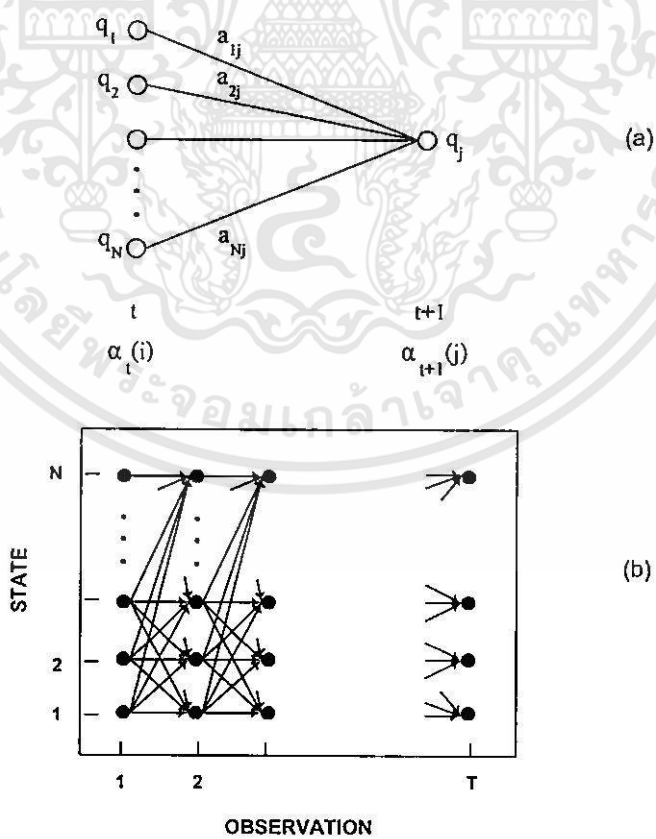
$$\alpha_t(i) = \pi_i b_i o_1 ; \quad 1 \leq i \leq N \tag{5.12}$$

เริ่มด้วยการกำหนดความน่าจะเป็นไปข้างหน้าซึ่งเป็นความน่าจะเป็นร่วมของสเตท i และมีเหตุการณ์เริ่มต้นเป็น O_1

2. <u>การเหนี่ยวนำ (Induction)</u>

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i)\, a_{ij} \right] b_j(O_{t+1}) \quad ; \begin{matrix} 1 \leq t \leq T-1 \\ 1 \leq j \leq N \end{matrix} \tag{5.13}$$

หมายความว่า สเตท j ที่เวลา t+1 สามารถมาได้จากสเตทก่อนหน้านี้ซึ่งเป็นไปได้ถึง N สเตท (สเตท i ณ เวลา t โดยที่ $1 \leq i \leq N$) ดังรูป 5.2 (a)



รูปที่ 5.2 กระบวนการไปข้างหน้า

จากรูป 5.2 (b) แสดงให้เห็นว่าการคำนวณค่าความน่าจะเป็นแบบไปข้างหน้า (Forward probability) มีโครงสร้างการคำนวณคล้ายๆลักษณะของโครงผลึก และเนื่องจากมีจำนวนสเตทเพียง N สเตท (แทนด้วยจำนวนโหนดในแต่ละช่วงเวลา t ใดๆในโครงผลึก) จำนวนลำดับสเตทจะถูกจัดเรียงลงในโหนดเหล่านี้ โดยในเวลา $t = 1$ จะทำการคำนวณค่าของ α_t (i) ในทุกๆสเตท, $1 \leq i \leq N$ และที่เวลา $t = 2, 3, ..., T$ จะทำการคำนวณค่าของ α_t (j) ในทุกๆสเตท, $1 \leq j \leq N$ โดยในแต่ละค่าจะทำการคำนวณมาจาก α_{t-1} (i) จำนวน N ค่าก่อนหน้านี้

3. <u>การสิ้นสุด (Termination)</u>

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_t(i) \quad ; \ 1 \leq i \leq N \tag{5.14}$$

เราสามารถหา $P(O|\lambda)$ ได้จากผลรวมของ α_t (i) จากทุกๆสเตท

2. กระบวนการย้อนกลับ (Backward Procedure) ; β_t (i) = Backward variable

นิยาม

$$\beta_t(i) = P(O_{t+1} O_{t+2} \ldots O_T | i_t = q_i, \lambda) \tag{5.15}$$

คือ ความน่าจะเป็นของลำดับค่าปรากฏส่วนหลังจากเวลา t+1 ไปจนจบโดยกำหนดว่าต้องอยู่ที่สเตท i ที่เวลา t และมีแบบจำลองเป็น λ เราจะคำนวณหา β_t (i) ได้ดังนี้

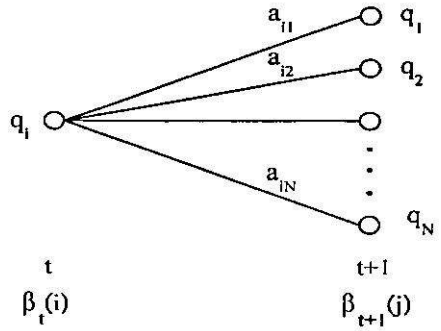
1. <u>การเริ่มต้น (Initialization)</u>

$$\beta_t(i) = 1 \quad ; \quad 1 \leq i \leq N \tag{5.16}$$

2. <u>การเหนี่ยวนำ (Induction)</u>

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \tag{5.17}$$

เมื่อ $t = T-1, T-2, ..., 1$, $1 \leq i \leq N$



รูปที่ 5.3 กระบวนการย้อนกลับ

จากรูป 5.3 เพื่อที่จะให้ค่าปรากฏอยู่ที่สเตท i ณ เวลา t โดยคาดคะเนจากลำดับค่าปรากฏจากเวลา t+1 ซึ่งเราจะต้องพิจารณาจากสเตท j ที่เป็นไปได้ทั้งหมด โดยจะขึ้นอยู่กับค่า a_{ij} และ $b_j(O_{t+1})$

การแก้ปัญหาที่ 2 ใช้ วิเทอร์บีอัลกอริธึม (Viterbi Algorithm) เพื่อที่จะหาลำดับสเตทที่ดีที่สุด, $q = (q_1\ q_2\ q_3 \dots q_T)$ ให้กับลำดับของค่าปรากฏ $O = \{O_1 O_2 O_3 \dots O_T\}$ ที่มีอยู่

โดยนิยามให้

$$\delta_t(i) = \max_{q_1, q_2, \dots, q_{t-1}} P[q_1\ q_2 \dots q_{t-1}, q_t = i, O_1\ O_2 \dots O_t \mid \lambda] \qquad (5.18)$$

เมื่อ $\delta_t(i)$ คือ ความน่าจะเป็นสูงสุด (hightest probability) ของเส้นทาง (path) ซึ่งจะหาได้จากค่าความน่าจะเป็นสูงสุด เมื่อเทียบกับสเตททุกสเตทในการให้ค่าปรากฏเป็นไปตามค่าปรากฏที่กำหนดให้ ที่ขณะเวลา t ใดๆ และจากการอาศัยคุณสมบัติของการเหนี่ยวนำจะได้

$$\delta_{t+1}(j) = [\max_i \delta_t(i)\, a_{ij}] \cdot b_j(O_{t+1}) \qquad (5.19)$$

โดยกำหนดให้ $\psi_t(j)$ เป็นอาเรย์ที่เก็บตำแหน่งของสเตท ที่ให้ค่าความน่าจะเป็นสูงที่สุดที่คำนวณได้ในแต่ละเวลา t และแต่ละลำดับ j ซึ่งจะสามารถหาลำดับสเตทที่ดีที่สุดได้โดยใช้กระบวนการต่อไปนี้

1. <u>การเริ่มต้น (Initialization)</u>

$$\delta_1(i) = \pi_i b_i(O_1) \qquad ; 1 \le i \le N \qquad (5.20a)$$

$$\psi_1(i) = 0 \tag{5.20b}$$

2. <u>การย้อนกลับ (Recursion)</u>

$$\delta_t(j) = [\max_{1 \le i \le N} \delta_{t-1}(i)\, a_{ij}] \cdot b_j(O_t) \quad ; \begin{matrix} 2 \le t \le T \\ 1 \le j \le N \end{matrix} \tag{5.21a}$$

$$\psi_t(j) = \arg\max_{1 \le i \le N} [\delta_{t-1}(i)\, a_{ij}] \quad ; \begin{matrix} 2 \le t \le T \\ 1 \le j \le N \end{matrix} \tag{5.21b}$$

3. <u>การสิ้นสุด (Termination)</u>

$$P^* = \max_{1 \le i \le N} [\delta_T(i)] \tag{5.22a}$$

$$q_T = \arg\max_{1 \le i \le N} [\delta_T(i)] \tag{5.22b}$$

4. <u>เส้นทางเดินย้อนกลับ (Backtracking)</u>

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \quad ; t = T-1, T-2, \ldots, 1 \tag{5.23}$$

การแก้ปัญหาที่ 3 จากที่กล่าวมาแล้วข้างต้นว่าแบบจำลองของเสียงจะแทนด้วยค่าพารามิเตอร์ $\lambda = (A,B,\pi)$ ดังนั้นเมื่อมีลำดับของค่าปรากฏจำนวนหนึ่ง เพื่อที่จะนำมาสร้างแบบจำลองอ้างอิงจะต้องทำการคำนวณหาค่าพารามิเตอร์ A, B, π ของแบบจำลองซึ่งจะอยู่ในรูปของค่าความน่าจะเป็น โดยวิธีที่เลือกใช้ก็คือ วิธีของ บาม-เวลส์ (Baum-Welch method) หรือเรียกอีกชื่อหนึ่งว่า EM (Expectation-Maximization method) โดยมี

นิยาม 1. คือ

$$\gamma_t(i) = P(q_t = i | O, \lambda) \tag{5.24}$$

เมื่อ $\gamma_t(i)$ คือ ค่าความน่าจะเป็นที่จะอยู่ที่สเตท i ที่ขณะเวลา t โดยให้ลำดับของค่าปรากฏด้วยโมเดล λ โดยที่กำหนดลำดับของค่าปรากฏให้ สามารถแสดงค่า $\gamma_t(i)$ ได้ดังนี้

$$\gamma_t(i) = P(q_t = i|O,\lambda)$$

$$= \frac{P(O, q_t = i|\lambda)}{P(O|\lambda)}$$

$$= \frac{P(O, q_t = i|\lambda)}{\sum_{i=1}^{N} P(O, q_t = i|\lambda)} \qquad (5.25)$$

เนื่องจาก $P(O, q_t = i|\lambda)$ มีค่าเท่ากับ $\alpha_t(i)\beta_t(i)$ ดังนั้นสามารถเขียน $\gamma_t(i)$ ได้เป็น

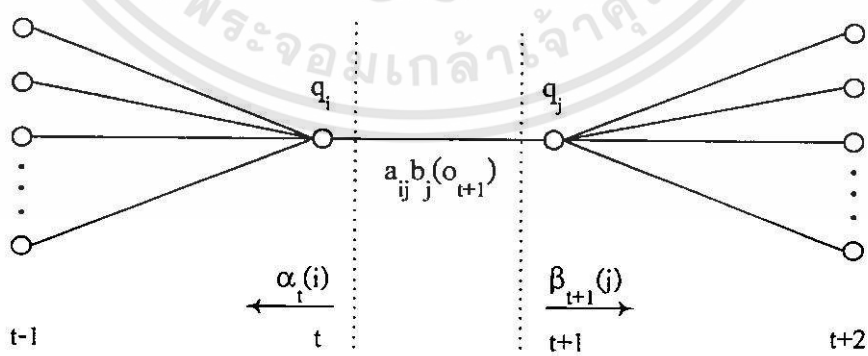
$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \qquad (5.26)$$

โดย $\alpha_t(i)$ เริ่มจาก $O_1, O_2, \ldots, O_t$ จนถึงสเตท i ที่เวลา t

โดย $\beta_t(i)$ เริ่มจาก $O_{t+1}, O_{t+2}, \ldots, O_T$ จนถึงสเตท $q_t = i$ ที่เวลา t

นิยาม 2. $\varepsilon_t(i,j) = P(q_t = i, q_{t+1} = j|O, \lambda)$ (5.27)

เมื่อ $\varepsilon_t(i,j)$ คือความน่าจะเป็นที่จะอยู่ที่สเตท i ที่เวลา t และสเตท j ที่เวลา t+1 เมื่อกำหนดแบบจำลองและลำดับค่าปรากฏให้



รูปที่ 5.4 ลำดับการคำนวณการเกิดค่าปรากฏร่วมซึ่งจะอยู่ที่สเตท i ที่เวลา t และอยู่ที่ สเตท j ที่เวลา t+1

จากรูปแสดง ลำดับการคำนวณการเกิดค่าปรากฏร่วม ซึ่งระบบจะอยู่ในสเตท i ที่เวลา t และอยู่ที่ สเตท j ที่เวลา t+1 โดย $\alpha_t(i)$ เริ่มจากเวลา t = 1 ที่ค่าปรากฏแรก จนถึงสเตท q_i ที่เวลา t และ $a_{ij}b_jO_{t+1}$ เป็นการเปลี่ยนสเตทที่เวลา t ไปเป็น q_j ที่เวลา t+1 และให้ค่าปรากฏเป็น O_{t+1}

ซึ่งจากนิยามของตัวแปรไปข้างหน้า $\alpha_t(i)$ และตัวแปรย้อนกลับ $\beta_t(i)$ สามารถนำมาสัมพันธ์กับ $\varepsilon_t(i,j)$ ได้เป็น

$$\varepsilon_t(i,j) = \frac{P(q_t = i, q_{t+1} = j, O|\lambda)}{P(O|\lambda)}$$

$$= \frac{\alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\beta_{t+1}(j)}{P(O|\lambda)}$$

$$= \frac{\alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\beta_{t+1}(j)} \qquad (5.28)$$

จากที่ได้นิยาม $\gamma_t(i)$ แล้ว นำมาสัมพันธ์กับ $\varepsilon_t(i,j)$ ได้เป็น

$$\gamma_t(i) = \sum_{j=1}^{N} \varepsilon_t(i,j) \qquad (5.29)$$

เมื่อ $\sum_{t=1}^{T-1} \gamma_t(i)$ = จำนวนของการย้ายสเตทจากสเตท i ในลำดับค่าปรากฏ O (5.30a)

$\sum_{t=1}^{T-1} \varepsilon_t(i,j)$ = จำนวนของการย้ายสเตทจากสเตท i ไป j ในลำดับค่าปรากฏ O (5.30b)

ดังนั้น สามารถคำนวณหาค่าของพารามิเตอร์ได้ดังนี้

π_i' = จำนวนครั้งในการอยู่ที่สเตท i ที่เวลา t = 1

$$\pi_i' = \gamma_1(i) \qquad ; 1 \le i \le N \qquad (5.31a)$$

$$a_{ij}' = \frac{\text{จำนวนครั้งที่คาดไว้ของการย้ายสเตทจาก i ไป j}}{\text{จำนวนครั้งที่คาดว่าจะย้ายจากสเตท i}}$$

$$a'_{ij} = \frac{\sum_{t=1}^{T-1} \varepsilon_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \qquad (5.31b)$$

$$b'_j(k) = \frac{\text{จำนวนครั้งที่คาดว่าจะอยู่ในสเตท j และเกิดค่าปรากฏเป็น } V_K}{\text{จำนวนครั้งที่คาดว่าจะอยู่ที่สเตท j}}$$

$$b'_j(k) = \frac{\sum_{t=1, O_t = V_K}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \qquad (5.31c)$$

จากกระบวนการข้างต้นถ้าให้ $\lambda = (A,B,\pi)$ เป็นแบบจำลองปัจจุบัน และใช้ λ นี้คำนวณในด้านขวาของสมการที่(5.31a-c)และให้แบบจำลองที่ได้จากการคำนวณซ้ำเป็น $\lambda' = (A',B',\pi')$ เป็นแบบจำลองที่ได้จากด้านซ้ายของสมการที่(5.31a-c) ซึ่งจะได้จุดวิกฤตของฟังก์ชั่นความน่าจะเป็นในกรณีที่ $\lambda' = \lambda$ หรือถ้า λ' มีความน่าจะเป็นมากกว่าแบบจำลอง λ [$P(O|\lambda') > P(O|\lambda)$] นั่นคือจะได้แบบจำลอง λ' ใหม่ ที่น่าจะทำให้เกิดลำดับของค่าปรากฏ O ที่ดีกว่า

5.5 การปรับปรุงค่าพารามิเตอร์ของ HMM

5.5.1 การสเกลลิง (Scaling)

พิจารณาคำจำกัดความของ $\alpha_t(i)$ ในสมการที่ 5.11 จะเห็นว่า $\alpha_t(i)$ ประกอบไปด้วยผลรวมเทอมขนาดใหญ่ที่อยู่ในรูป

$$\prod_{s=1}^{t-1} a_{q_s q_{s+1}} \prod_{s=1}^{t} b_{q_s}(O_s)$$

เนื่องจากค่า a และ b เป็นค่าความน่าจะเป็น ซึ่งโดยทั่วไปแล้วมีค่าน้อยกว่า 1 ด้วยเหตุนี้เมื่อ t มากขึ้นค่าแต่ละเทอมของ $\alpha_t(i)$ จะเข้าสู่ศูนย์ ทำให้ช่วงไดนามิก (Dinamic Range) ของการคำนวณ $\alpha_t(i)$ มีค่าสูงเกินขอบเขตการทำงานของเครื่องคำนวณทำให้ค่าที่ได้ไม่ถูกต้อง ซึ่งเราสามารถแก้ไขปัญหานี้ได้โดยใช้กระบวนการสเกลลิง (Scaling Procedure)

การสเกลลิ่งทำได้โดยการคูณ $\alpha_t(i)$ ด้วยสัมประสิทธิ์การสเกลลิง ซึ่งไม่ขึ้นกับ i (นั่นคือขึ้นอยู่กับค่าของเวลา t เท่านั้น) เพื่อให้ $\alpha_t(i)$ ที่ผ่านการสเกลลิงแล้วมีค่าอยู่ภายในช่วง Dynamic Range ของเครื่องคำนวณในทุกๆค่าเวลาภายใต้ $1 \le t \le T$ และในทำนองเดียวกันจะต้องทำการคำนวณค่าสัมประสิทธิ์การสเกลลิงของค่า $\beta_t(i)$ ด้วย ซึ่งในขั้นตอนสุดท้ายของการคำนวณ ค่าสัมประสิทธิ์ของการสเกลลิงจะตัดกันหมดไป

เพื่อให้เข้าใจการทำงานของกระบวนการสเกลลิงดีขึ้น เราจะพิจารณาสมการของการย้ายสเตท(a_{ij}) ที่อยู่ในเทอมของตัวแปรไปข้างหน้า และตัวแปรย้อนกลับ

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{t=1}^{T}\sum_{j=1}^{N} \alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\beta_{t+1}(j)} \qquad (5.32)$$

พิจารณาสัญลักษณ์ในการคำนวณ $\alpha_t(i)$ เมื่อกำหนดให้

$\alpha_t(i)$ แทน α ที่ยังไม่ผ่านการสเกล

$\hat{\alpha}_t(i)$ แทน α ที่สเกลแล้ว

$\hat{\hat{\alpha}}_t(i)$ แทน α แทนเวอร์ชั่นของ α ก่อนการสเกล

เมื่อเวลาเริ่มต้น $t = 1$

คำนวณ $\alpha_t(i)$ ตามสมการที่ 5.12 และกำหนดให้ $\hat{\hat{\alpha}}_1(i) = \alpha_1(i)$

เมื่อ $$c_1 = \frac{1}{\sum_{i=1}^{N} \alpha_1(i)}$$

และ $$\hat{\alpha}_1(i) = c_1 \alpha_1(i)$$

เมื่อเวลา $2 \le t \le T$

เริ่มแรกทำการคำนวนหา $\hat{\hat{\alpha}}_t(i)$ ตามสมการการเหนี่ยวนำ สมการที่ 5.13 โดยใช้เทอมของค่าที่ผ่านการสเกลแล้ว $\hat{\alpha}_t(i)$ จะได้ดังนี้

$$\hat{\hat{\alpha}}_t(i) = \sum_{j=1}^{N} \alpha_{t-1}(j)\, a_{ji}\, b_i(O_t) \qquad (5.33a)$$

เมื่อกำหนดค่าสัมประสิทธ์การสเกลลิง ; c_t เป็น

$$c_t = \frac{1}{\sum_{i=1}^{N} \hat{\alpha}_t(i)} \tag{5.33b}$$

เมื่อให้

$$\hat{\alpha}_t(i) = c_t \hat{\alpha}_t(i) \tag{5.33c}$$

จากสมการที่ 5.33 a-c สามารถเขียนสมการได้เป็น

$$\hat{\alpha}_t(i) = \frac{\sum_{j=1}^{N} \hat{\alpha}_{t-1}(j)\, a_{ji}\, b_i(O_t)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \hat{\alpha}_{t-1}(j)\, a_{ji}\, b_i(O_t)} \tag{5.34}$$

และโดยการเหนี่ยวนำสามารถเขียน $\hat{\alpha}_{t-1}(j)$ ได้เป็น

$$\hat{\alpha}_{t-1}(j) = \left(\prod_{T=1}^{t-1} c_T \right) \alpha_{t-1}(j) \tag{5.35a}$$

ดังนั้นสามารถเขียน $\hat{\alpha}_t(i)$ ได้เป็น

$$\hat{\alpha}_t(i) = \frac{\sum_{j=1}^{N} \alpha_{t-1}(j) \left(\prod_{T=1}^{t-1} c_T \right) a_{ji}\, b_i(O_t)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_{t-1}(j) \left(\prod_{T=1}^{t-1} c_T \right) a_{ji}\, b_i(O_t)} = \frac{\alpha_t(i)}{\sum_{i=1}^{N} \alpha_t(i)} \tag{5.35b}$$

นั่นคือการสเกลลิงจะทำได้โดยนำ $\alpha_t(i)$ แต่ละค่า มาหารด้วยผลรวมของ $\alpha_t(i)$ ทุกสเตท จากนั้นทำการคำนวณลักษณะเดียวกันนี้กับเทอมของตัวแปรย้อนกลับ $\beta_t(i)$ โดยใช้สเกลเฟคเตอร์เดียวกัน ในรูปของ

$$\hat{\beta}_t(i) = c_t \beta_t(i) \tag{5.36}$$

พิจารณาสมการ 5.32 ในเทอมของตัวแปรที่ผ่านการสเกล จะได้เป็น

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \hat{\alpha}_t(i)\, a_{ij}\, b_j(O_{t+1})\hat{\beta}_{t+1}(j)}{\sum_{t=1}^{T-1}\sum_{j=1}^{N} \hat{\alpha}_t(i)\, a_{ij}\, b_j(O_{t+1})\hat{\beta}_{t+1}(j)} \tag{5.37}$$

โดยแต่ละ $\hat{\alpha}_t(i)$, $\hat{\beta}_{t+1}(j)$ สามารถเขียนให้อยู่ในรูปของ

$$\hat{\alpha}_t(i) = \left[\prod_{s=1}^{T} c_s\right] \alpha_t(i) = C_t \alpha_t(i) \tag{5.38}$$

$$\hat{\beta}_{t+1}(j) = \left[\prod_{s=t+1}^{T} c_s\right] \beta_{t+1}(j) = D_{t+1} \beta_{t+1}(j) \tag{5.39}$$

ดังนั้นสมการ 5.37 สามารถเขียนได้เป็น

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} C_t \alpha_t(i)\, a_{ij}\, b_j(O_{t+1}) D_{t+1} \beta_{t+1}(j)}{\sum_{t=1}^{T-1}\sum_{j=1}^{N} C_t \alpha_t(i)\, a_{ij}\, b_j(O_{t+1}) D_{t+1} \beta_{t+1}(j)} \tag{5.40}$$

โดยเทอม $C_t D_{t+1}$ สามารถเขียนให้อยู่ในรูปของ

$$C_t D_{t+1} = \prod_{s=1}^{t} c_s \prod_{s=t+1}^{T} c_s = \prod_{s=1}^{T} c_s = C_T \tag{5.41}$$

จะเห็นว่าเทอม $C_t D_{t+1}$ เป็นค่าที่ไม่ขึ้นกับเวลา ดังนั้นสามารถตัดออกจากทั้งเศษและส่วนของสมการ 5.40 ได้ ซึ่งจะทำให้ได้สูตรของการคำนวณซ้ำ กระบวนการสเกลลิงดังกล่าวนี้จะถูกนำไปใช้กับสัมประสิทธิ์ π และ β การสเกลลิงนี้จะทำให้การคำนวณค่า $P(O|\lambda)$ เปลี่ยนไป เราจะไม่สามารถหาได้จากการรวมเทอมของ $\hat{\alpha}_T(i)$ เนื่องจากเป็นค่าที่ถูกสเกลแล้ว แต่เราสามารถคำนวณได้จากคุณสมบัติ

$$\prod_{t=1}^{T} c_t \prod_{i=1}^{N} \alpha_T(i) = C_T \sum_{i=1}^{N} \alpha_T(i) = 1 \qquad (5.42)$$

ดังนั้นจะได้

$$\prod_{t=1}^{T} c_t \cdot P(O|\lambda) = 1 \qquad (5.43)$$

หรือ

$$P(O|\lambda) = \frac{1}{\prod_{t=1}^{T} c_t} \qquad (5.44)$$

หรือ

$$\log [P(O|\lambda)] = \sum_{t=1}^{T} \log c_t \qquad (5.45)$$

นั่นคือ การคำนวณค่า P จะอยู่ในรูป log ของ Pเพื่อไม่ให้เกินช่วงไดนามิก (Dinamic Range) ของเครื่องคำนวณ

5.5.2 ลำดับของค่าปรากฏหลายลำดับ (Multiple Observation Sequences)

ในการสร้างแบบจำลองด้วย Left-Right Model จำเป็นจะต้องใช้จำนวนลำดับของเหตุการณ์หลายๆลำดับเข้ามาเทรนเพื่อให้การประมาณค่าพารามิเตอร์ของแบบจำลองที่ได้มีความน่าเชื่อถือที่สุด ถ้ากำหนดให้ k แทน เซ็ตของลำดับค่าปรากฏ ดังนี้

$$O = [O^{(1)}, O^{(2)}, ..., O^{(k)}] \qquad (5.46)$$

เมื่อ $O^{(k)} = (O_1^{(k)} \; O_2^{(k)} \; ... \; O_{T_k}^{(k)})$ คือ ลำดับค่าปรากฏอันดับที่ k โดยสมมติให้แต่ละอันดับของค่าปรากฏเป็นอิสระต่อกัน โดยมีจุดประสงค์ เพื่อที่จะปรับค่าพารามิเตอร์ของแบบจำลอง λ ให้มีค่ามากที่สุด

$$P(O|\lambda) = \prod_{k=1}^{K} P(O^{(k)}|\lambda) \qquad (5.47)$$

$$= \prod_{k=1}^{K} P_k \qquad (5.48)$$

ดังนั้นจะได้สมการของการคำนวณซ้ำที่ใช้ในการปรับค่า $\bar{a}_{ij}$ และ $\bar{b}_j(l)$ เป็น

$$\bar{a}_{ij} = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i)\, a_{ij}\, b_j(O_{t+1}^{(k)}) \beta_{t+1}^k(j)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)} \tag{5.49}$$

และ

$$\bar{b}_j(l) = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1, O_t=v_l}^{T_k-1} \alpha_t^k(i)\, \beta_t^k(i)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)} \tag{5.50}$$

ส่วนค่า π_i ไม่ต้องมีการคำนวณซ้ำเนื่องจาก $\pi_1 = 1, \pi_i = 0, i \neq 1$

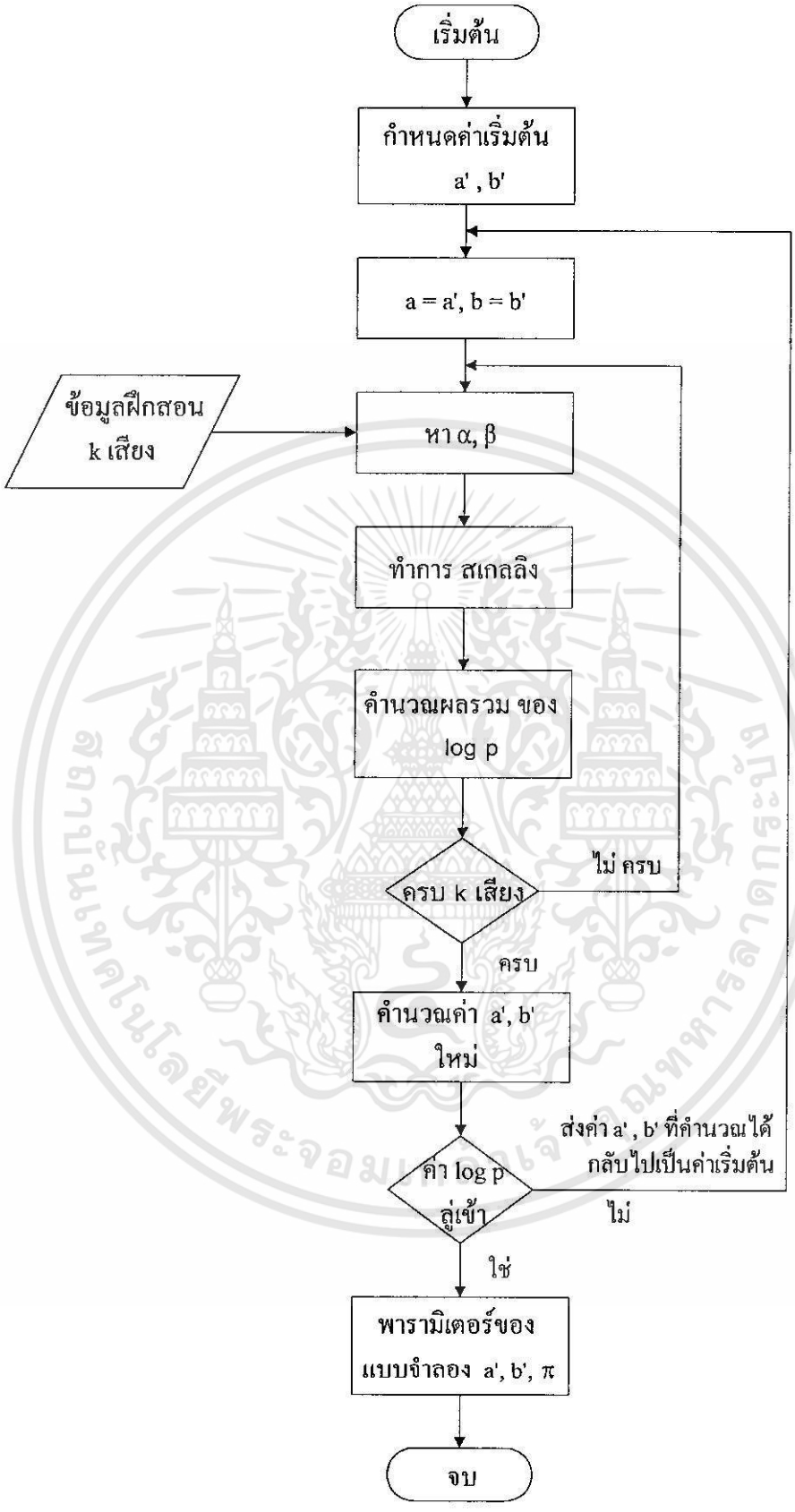
จากสมการของการสเกลลิงสมการที่ 5.49-5.50 เราสามารถเขียนสมการที่อยู่ในเทอมของตัวแปรที่สเกลแล้วได้เป็น

$$\bar{a}_{ij} = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i)\, a_{ij}\, b_j(O_{t+1}^{(k)}) \hat{\beta}_{t+1}^k(j)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)} \tag{5.51}$$

$$\bar{b}_j(l) = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1, O_t=v_l}^{T_k-1} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)} \tag{5.52}$$

5.6 การสร้างแบบจำลองอ้างอิง

จากหัวข้อ 5.4 ได้กล่าวถึงการแก้ปัญหาทั้ง 3 ข้อของ HMM ซึ่งจะถูกนำมาใช้ในการคำนวณหาพารามิเตอร์ของแบบจำลองอ้างอิงในการรู้จำ โดยขั้นตอนในการคำนวณหาพารามิเตอร์ของแบบจำลองอ้างอิง สามารถแสดงได้ดัง รูปที่ 5.5

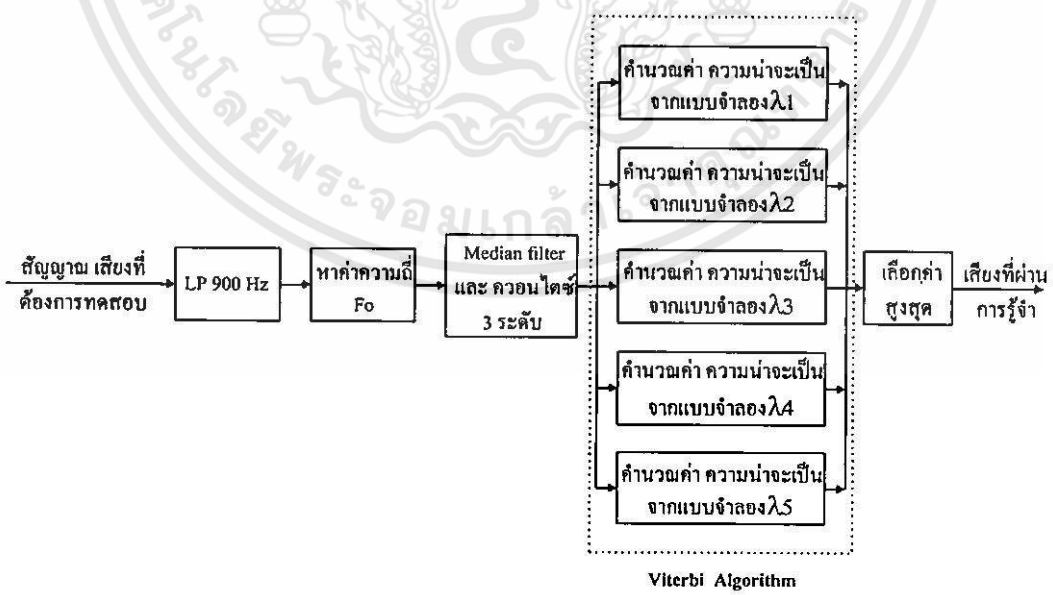


รูปที่ 5.5 โฟลว์ชาร์ต การคำนวณหาค่าพารามิเตอร์ของแบบจำลองอ้างอิง

เมื่อต้องการสร้างแบบจำลองอ้างอิงเสียงวรรณยุกต์ทั้ง 5 ระดับเสียง สิ่งที่จะต้องมีก็คือ กลุ่มเสียงต้นแบบ หรือลำดับของค่าปรากฏทั้ง 5 กลุ่ม เพื่อใช้เป็นข้อมูลฝึกสอน (Training data) จากรูปที่ 5.5 แสดงขั้นตอนของการคำนวณการสร้างแบบจำลองอ้างอิง โดยในขั้นแรกจะต้องทำการกำหนดค่า A, B เริ่มต้น จากนั้น ทำการคำนวณหาค่า α, β โดยใช้การแก้ปัญหาที่ 1 ของ HMM แล้วทำการสเกลลิง เพื่อไม่ให้ค่าที่ได้จากการคำนวณมีค่าเกินช่วงไดนามิกของเครื่องคำนวณ (Dynamic Range) จากนั้นนำค่าสัมประสิทธิ์ของการสเกลลิงมาคำนวณหาค่า ความน่าจะเป็น $P(O|\lambda)$ ซึ่งจะอยู่ในรูปของค่า log P และเนื่องจากในการสร้างแบบจำลองอ้างอิง จำเป็นจะต้องใช้ข้อมูลฝึกสอนจำนวนมาก เพื่อให้แบบจำลองอ้างอิงที่สร้างขึ้น คลอบคลุมความแปรปรวนของลักษณะเสียงให้ได้มากที่สุด ดังนั้นจึงจะต้องมีการคำนวณซ้ำเกิดขึ้น ตามจำนวนของเสียงที่นำมาฝึกสอน จากนั้นทำการหาค่าผลรวมของค่าความน่าจะเป็น (ผลรวมของ log P จากจำนวนเสียงทั้งหมด) เพื่อมาใช้ในการคำนวณหาค่าพารามิเตอร์ของแบบจำลอง $\lambda=(A',B',\pi)$ โดยใช้การแก้ปัญหาที่ 3 ของ HMM จากนั้นทำการคำนวณซ้ำจนกว่าค่าผลรวมของ log P ที่ได้ในแต่ละรอบมีค่าลู่เข้า หรือไม่เปลี่ยนแปลง พารามิเตอร์ของแบบจำลอง $\lambda'=(A',B',\pi)$ ค่าสุดท้าย จะเป็นแบบจำลองที่น่าจะทำให้เกิดลำดับของค่าปรากฏ O ที่ดีกว่า โดยรายละเอียดของขั้นตอนต่างๆ ได้กล่าวมาแล้วในหัวข้อก่อนหน้านี้

5.7 แบบจำลอง ฮิดเดนมาร์คอฟ ในการรู้จำเสียงวรรณยุกต์ภาษาไทย

การรู้จำระดับเสียงวรรณยุกต์ทั้ง 5 ระดับ ด้วย HMM แสดงได้ดังรูปที่ 5.6



รูปที่ 5.6 บล็อคไดอะแกรม ของการรู้จำระดับเสียงวรรณยุกต์ด้วยแบบจำลองมาร์คอฟ

รูปที่ 5.6 แสดงขั้นตอนในการทดสอบการรู้จำระดับเสียงวรรณยุกต์ทั้ง 5 ระดับ โดยเสียงที่ต้องการทดสอบจะถูกนำมาผ่านขั้นตอนในการหาค่าความถี่มูลฐาน แล้วควอนไตซ์เป็น 3 ระดับ ดังกล่าวมาแล้วในบทที่ 3-4 ซึ่งข้อมูลที่ได้จากการควอนไตซ์ จะถูกนำมาเทียบกับแบบจำลองอ้างอิงเสียงวรรณยุกต์ทั้ง 5 แบบ (λ_1-λ_5) โดยแบบจำลองอ้างอิงใดที่ให้ค่าความน่าจะเป็น(ในการเกิดเหตุการณ์)สูงสุด จะถือว่าคำศัพท์ที่นำมาทดสอบ จะมีระดับเสียงเดียวกันกับแบบจำลองนั้น นั่นเอง โดยขั้นตอนการคำนวณหาค่าความน่าจะเป็นจะใช้การแก้ปัญหาที่ 2 หรือวิทเทอร์บี อัลกอริธึม ดังได้กล่าวถึงรายละเอียดมาแล้วข้างต้น

บทที่ 6

การทดลอง และผลการทดลอง

6.1 กล่าวนำ

เนื้อหาในบทนี้จะกล่าวถึงวิธีการทดลองและผลการทดลองที่ได้ในขั้นตอนต่างๆของการสร้างแบบจำลองการรู้จำระดับเสียงวรรณยุกต์ของคำพยางค์เดียวในภาษาไทย โดยการทดลองจะแบ่งออกเป็น 3 ส่วนใหญ่ๆดังนี้

ส่วนที่ 1 เป็นขั้นตอนในการวิเคราะห์ และพัฒนาอัลกอริธึมในการสร้างแบบจำลองการรู้จำระดับเสียงวรรณยุกต์

ส่วนที่ 2 เป็นขั้นตอนในการสร้างแบบจำลองอ้างอิง

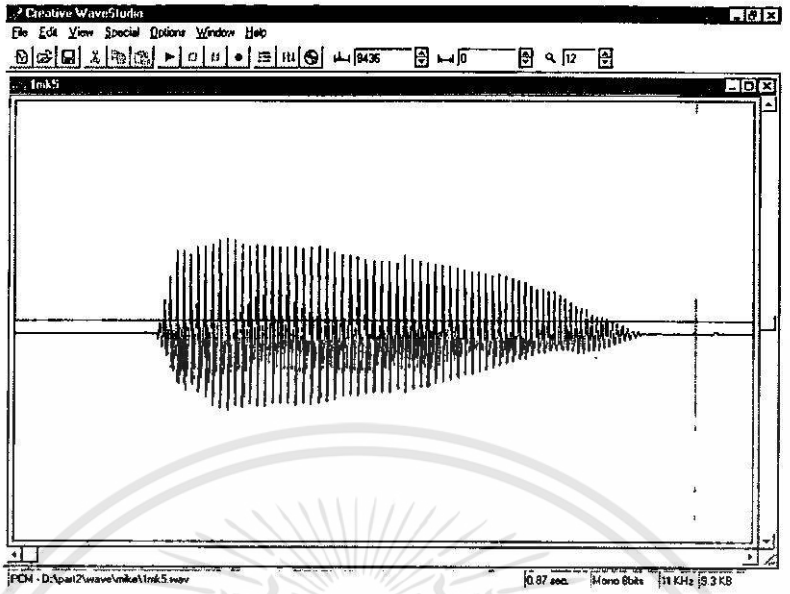
ส่วนที่ 3 เป็นขั้นตอนในการทดสอบแบบจำลองอ้างอิงที่สร้างขึ้น

6.2 การกำหนดขอบเขตของพยางค์ หรือคำ

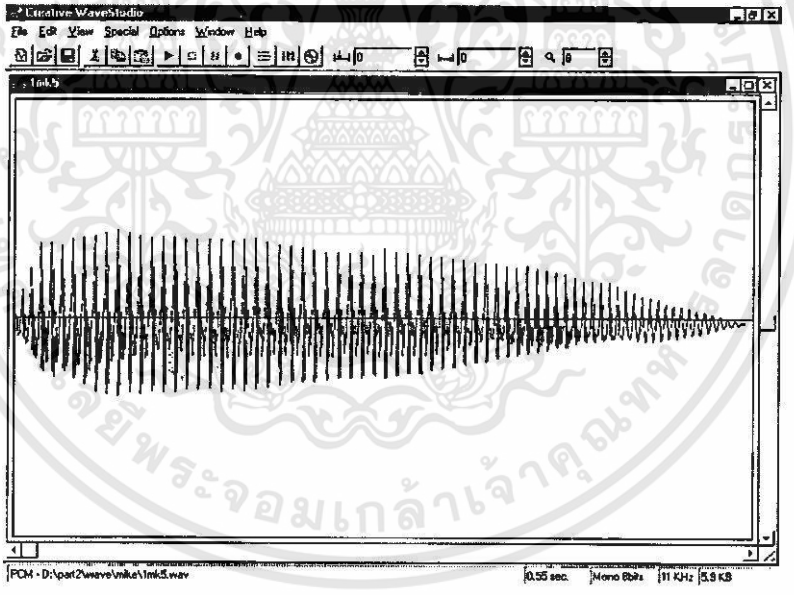
สัญญาณเสียงพูดที่ใช้ในวิทยานิพนธ์นี้ เป็นคำพยางค์เดียวที่ได้จากการเก็บตัวอย่างข้อมูลเสียง โดยใช้เครื่องคอมพิวเตอร์ส่วนบุคคล และการ์ดเสียง (Sound Blaster AWE64) ซึ่งข้อมูลจะถูกเก็บอยู่ในรูปของไฟล์ '.wav' ข้อมูล 1 ตัวอย่างของเสียงจะถูกแทนด้วยข้อมูลขนาด 8 บิท โดยใช้ความถี่ในการชักตัวอย่างเท่ากับ 11.025 KHz และไฟล์ข้อมูล '.wav' นี้จะถูกใช้เป็นข้อมูลอินพุทสำหรับการคำนวณของโปรแกรมที่เขียนขึ้น โดยในวิทยานิพนธ์นี้เลือกใช้โปรแกรม Borland C++ ในการทดลองและพัฒนาวิธีการต่างที่ใช้ในการรู้จำเสียงพูด

จากที่กล่าวมาแล้วว่า หน่วยเสียงวรรณยุกต์เป็นหน่วยเสียงซ้อน วางตัวอยู่เหนือเสียงก้อง ซึ่งคุณสมบัติของเสียงก้องที่สังเกตได้เด่นชัดในโดเมนของเวลา คือ ความเป็นคาบ ดังนั้นเมื่อพิจารณาการเลื่อนไปของเสียงวรรณยุกต์ หรือเรียกว่าทางเดินเสียงวรรณยุกต์ในแกนของเวลา การกำหนดขอบเขตของคำเพื่อที่จะวิเคราะห์หาหน่วยเสียงวรรณยุกต์ จะกำหนดขอบเขตภายในช่วงที่มีความเป็นคาบทั้งหมดของคำ หรือพยางค์นั้นๆ

ตัวอย่างสัญญาณเสียงพูดก่อนตัดคำ และหลังตัดคำแสดงได้ดังรูปที่ 6.1 และภาคผนวก ก



(a) ก่อนตัดคำ



(b) หลังตัดคำ

รูปที่ 6.1 ตัวอย่างสัญญาณเสียงพูดของคำว่า "อ๋อ" จากผู้ออกเสียงเพศชาย

(a) ก่อนตัดคำ

(b) หลังตัดคำ

6.3 ขั้นตอนในการวิเคราะห์ และพัฒนาอัลกอริธึมในการสร้างแบบจำลองการรู้จำระดับเสียงวรรณยุกต์

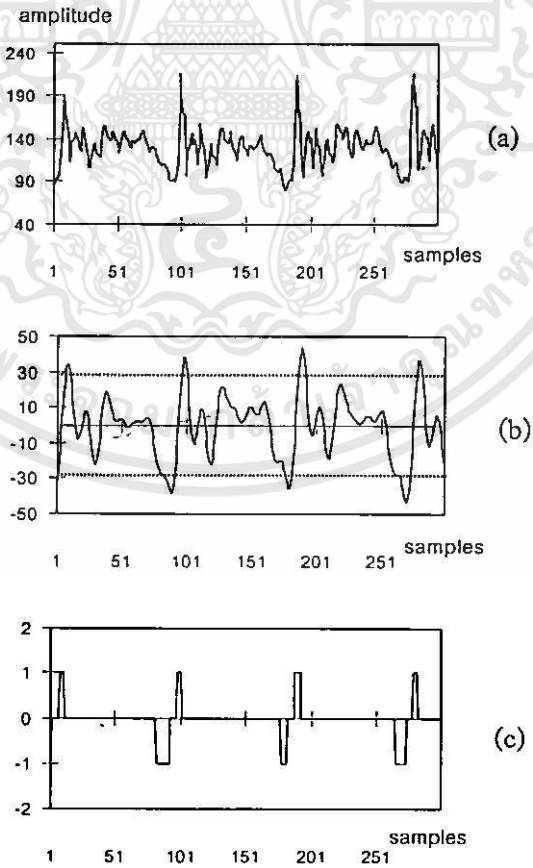
แบ่งออกเป็น 3 ขั้นตอน แสดงได้ดังรูปที่ 6.2

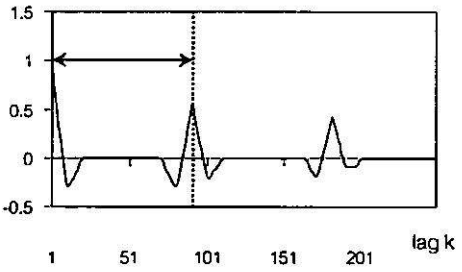
1. การหาค่าพิทช์
2. การเตรียมข้อมูล เพื่อใช้เป็นข้อมูลฝึกสอนในการสร้างแบบจำลองอ้างอิง
3. การสร้างแบบจำลองอ้างอิงด้วย ฮิดเดนมาร์คอฟโมเดล

รูปที่ 6.2 ขั้นตอนในการวิเคราะห์

6.3.1 การหาค่าพิทช์

การหาค่าพิทช์ทำได้โดยใช้วิธี ออโตโครีเลชั่น ที่มีการคลิปสัญญาณ ซึ่งมีขั้นตอนในการวิเคราะห์ดังกล่าวมาแล้วในบทที่3 หัวข้อที่ 3.3 โดยรูปที่ 6.3(a-d) แสดงตัวอย่างสัญญาณที่ผ่านขั้นตอนต่างๆตามลำดับ

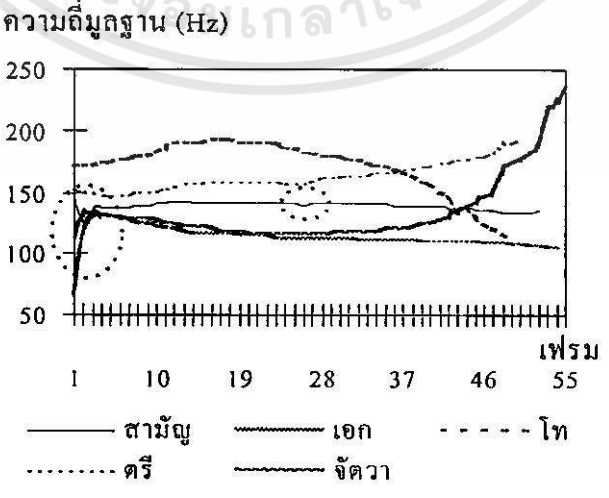




รูปที่ 6.3 (a-d) ตัวอย่างสัญญาณในขั้นตอนต่างๆของการหาพิทช์

จากรูปที่ 6.3 (a) เป็นสัญญาณที่ได้จากการชักตัวอย่างข้อมูล (sampling) ซึ่งจะสังเกตุได้ว่าสัญญาณข้อมูลจะประกอบด้วยองค์ประกอบของความถี่จำนวนมาก อันเป็นผลมาจากการตอบสนองทางความถี่ภายในช่องทางเดินเสียง ซึ่งจากที่กล่าวมาแล้วในบทที่ 3 ว่าความถี่เหล่านี้อาจมีผลทำให้การกำหนดตำแหน่งพิทช์ในการคำนวณออโตโครีเลชั่นคลาดเคลื่อนไปจากตำแหน่งจริงได้ ดังนั้นเพื่อเป็นการกำจัดผลของความถี่เหล่านี้ออกไป จึงนำสัญญาณเสียงมาผ่านการกรองความถี่ต่ำผ่าน 900 Hz ซึ่งจะได้สัญญาณที่มีความราบเรียบมากขึ้นดังรูปที่ 6.3 (b) โดยเส้นประในรูปแสดงระดับในการคลิปสัญญาณ ซึ่งในวิทยานิพนธ์นี้ใช้ระดับการคลิปที่ 65 เปอร์เซนต์ (ได้จากการทดลอง) และสัญญาณที่ผ่านการคลิปแสดงได้ดังรูปที่ 6.3 (c) จากนั้นนำสัญญาณที่ผ่านการคลิปมาทำการคำนวณออโตโครีเลชั่นเพื่อหาพิทช์ จะได้สัญญาณที่มีลักษณะดังรูปที่ 6.3 (d) โดยระยะห่างระหว่าง R(0) กับจุดยอดที่สูงที่สุดถัดไปก็คือคาบพิทช์ ซึ่งจากรูปได้ตำแหน่งที่ 91 และสามารถหาค่าความถี่มูลฐานได้เท่ากับ 121 Hz

เมื่อเสร็จสิ้นกระบวนการออโตโครีเลชั่นจะได้ว่าสัญญาณข้อมูลเสียง 1 เฟรม จะถูกแทนด้วยค่าความถี่มูลฐาน 1 ค่า นั่นคือข้อมูลเสียง 1 เสียงจะถูกแทนด้วยลำดับของความถี่มูลฐานที่มีขนาดเป็น 1xN เมื่อ N คือจำนวนเฟรมทั้งหมดของเสียง แสดงได้ดังรูปที่ 6.4



รูปที่ 6.4 ลักษณะการเปลี่ยนแปลงความถี่มูลฐานในวรรณยุกต์ทั้ง 5 เสียง

จากรูปที่ 6.4 เป็นการออกเสียงคำว่า อา อ่า อ้า อ๊า อ๋า ในผู้ออกเสียงเพศชาย ซึ่งจะสังเกตุได้ว่าในระดับเสียงวรรณยุกต์แต่ละระดับจะมีทิศทางการเปลี่ยนแปลงของความถี่มูลฐานที่แตกต่างกัน โดยกราฟที่อยู่ในวงกลมเส้นประจะถูกปรับแต่งให้เรียบขึ้น ซึ่งจะกล่าวถึงในขั้นตอนต่อไป

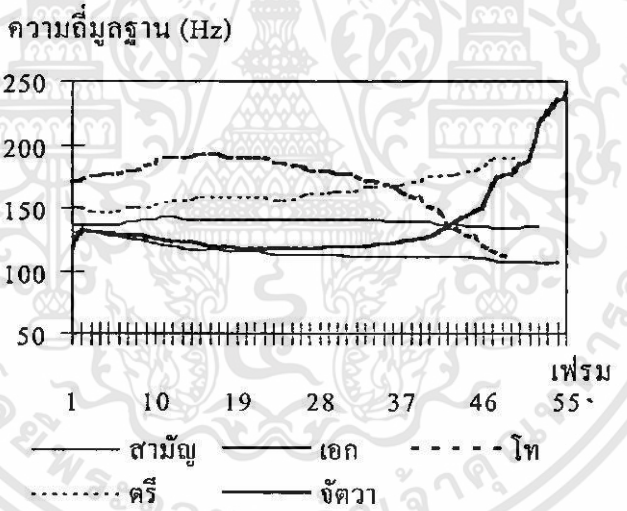
6.3.2 การเตรียมข้อมูล (Pre-process)

เป็นส่วนของการจัดการข้อมูลซึ่งมีอยู่ 2 ขั้นตอน ดังกล่าวไว้ในบทที่ 4 คือ

1. การปรับปรุงความต่อเนื่องของข้อมูลด้วยวิธีการกรองมัธยฐาน (Median Filtering)
2. การควอนไตซ์ทิศทางการเปลี่ยนแปลงของค่าความถี่มูลฐาน

โดยแต่ละขั้นตอนสามารถอธิบายได้ดังนี้ คือ

ขั้นตอนที่ 1 การกรองมัธยฐาน : เป็นส่วนของการปรับปรุงข้อมูลให้มีความต่อเนื่องมากขึ้น ซึ่งจากรูปที่ 6.4 เมื่อนำมาผ่านขั้นตอนนี้จะได้กราฟที่มีลักษณะดังรูปที่ 6.5

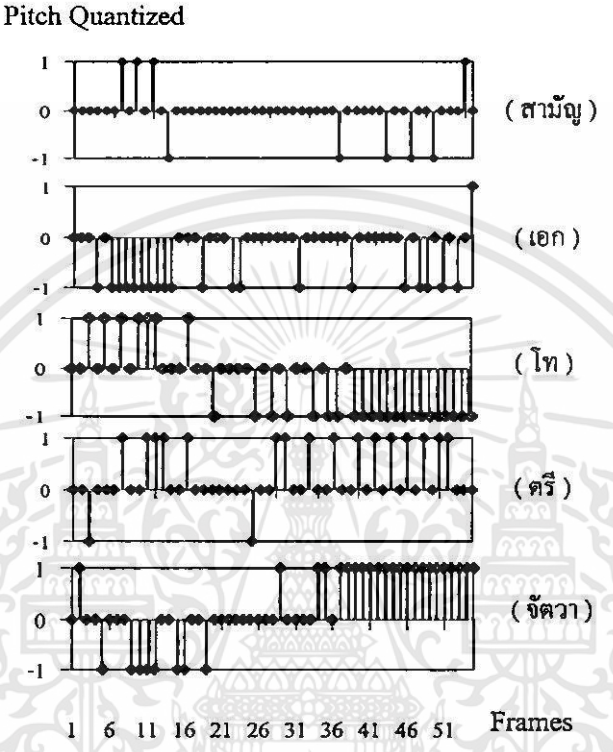


รูปที่ 6.5 ตัวอย่างข้อมูลที่นำมาผ่านตัวกรองมัธยฐาน

จากรูปจะพบว่าข้อมูลในช่วงต้นเสียงจะถูกปรับให้มีความต่อเนื่องมากขึ้นเมื่อเทียบกับรูปที่ 6.4 จากนั้นข้อมูลที่ได้นี้จะถูกนำไปทำการควอนไตซ์ค่าการเปลี่ยนแปลงของความถี่ในขั้นตอนต่อไป

ขั้นตอนที่ 2 การควอนไตซ์ : เป็นการเตรียมข้อมูลเพื่อใช้เป็นข้อมูลฝึกสอนในกระบวนการสร้างแบบจำลองด้วย ฮิดเดนมาร์คอฟโมเดล ซึ่งข้อมูลที่ผ่านการกรองมัธยฐานจะถูกนำมาทำการจัดระดับออกเป็น 3 ระดับตามทิศทางการเปลี่ยนแปลงของความถี่มูลฐานต่อเวลา โดยแทนค่าเป็น 1

เมื่อความถี่เพิ่มขึ้น เป็น 0 เมื่อความถี่คงที่ และ –1 เมื่อความถี่ลดลง ตัวอย่างสัญญาณข้อมูลในรูปที่ 6.5 เมื่อนำมาผ่านการควอนไตซ์ออกเป็น 3 ระดับสามารถแสดงได้ดังรูปที่ 6.6 เมื่อผ่านขั้นตอนนี้แล้วข้อมูลจะอยู่ในรูปของ "ลำดับการเปลี่ยนแปลงของความถี่" ซึ่งมีสมาชิกของลำดับเป็น {-1, 0, 1}



รูปที่ 6.6 การควอนไตซ์ข้อมูลออกเป็น 3 ระดับตามทิศทางการเปลี่ยนแปลงของความถี่มูลฐาน

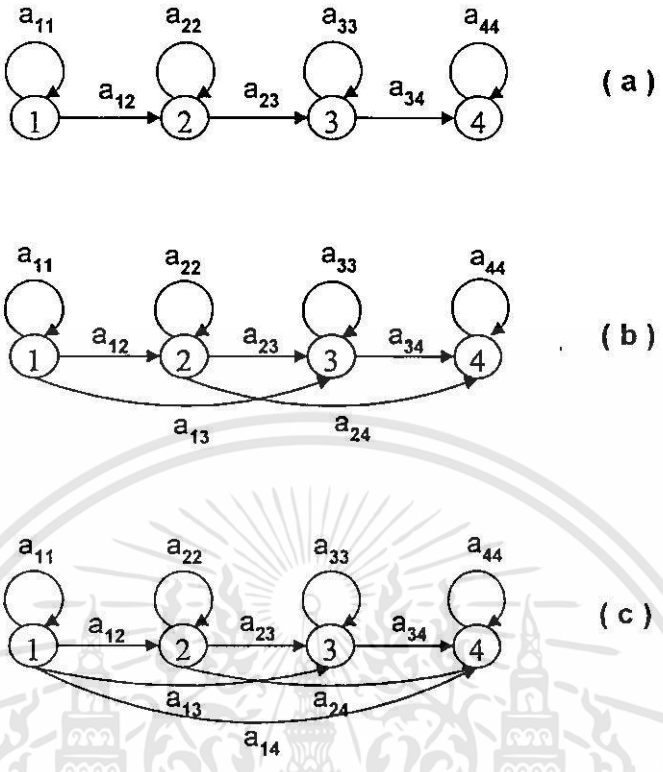
6.3.3 การสร้างแบบจำลองการรู้จำด้วย ฮิดเดนมาร์คอฟโมเดล (HMM)

แบบจำลองที่ใช้ในวิทยานิพนธ์นี้เป็นแบบ Left-Right Model เพราะเป็นแบบจำลองที่เหมาะสำหรับการรู้จำรูปแบบของคำ ประเภทคำศัพท์เดี่ยว (Isolated word) เนื่องจากสามารถนำเวลาเข้ามาเกี่ยวข้องกับสเตทของแบบจำลองได้โดยตรง และอาจตีความหมายทางกายภาพของสเตทเป็นเสียงที่แตกต่างกันของคำได้ [14]

การทดสอบหารูปแบบ HMM ที่เหมาะสม

ในวิทยานิพนธ์นี้ได้ทำการทดสอบหารูปแบบของ Left-Right Model ที่เหมาะสมสำหรับอัลกอริธึมที่ได้พัฒนาขึ้นดังกล่าวมาก่อนหน้านี้ โดยพารามิเตอร์ที่มีความสัมพันธ์เกี่ยวข้อง ที่ต้องคำนึงถึงมีอยู่ 2 ค่า คือ

1. จำนวนสเตท
2. การย้ายข้ามสเตท



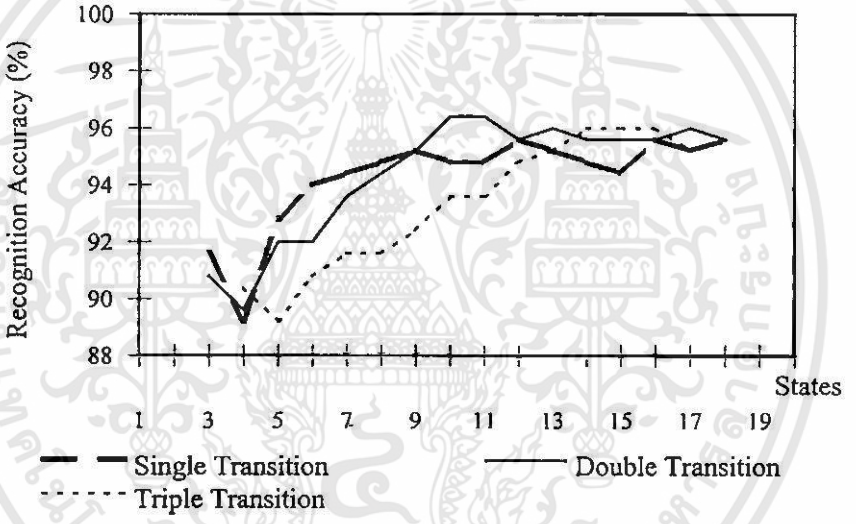
รูปที่ 6.7 Left-Right Model 4 สเตท
(a) การย้ายข้ามสเตทสูงสุดได้ไม่เกิน 1 สเตท (Single Transition)
(b) การย้ายข้ามสเตทสูงสุดได้ไม่เกิน 2 สเตท (Double Transition)
(c) การย้ายข้ามสเตทสูงสุดได้ไม่เกิน 3 สเตท (Triple Transition)

รูปที่ 6.7 เป็นตัวอย่างของแบบจำลอง HMM ที่มีจำนวนสเตท 4 สเตท และมีค่าการย้ายข้ามสเตทที่เป็นไปได้ 3 แบบ ซึ่งจากลักษณะเดียวกันนี้ วิทยานิพนธ์นี้ได้ทำการศึกษาทดลองหาจำนวนขนาดสเตท และการย้ายข้ามสเตทที่เหมาะสม เพื่อให้แบบจำลองการรู้จำระดับเสียงที่สร้างขึ้นมีความถูกต้องแม่นยำสูงสุด โดยทำการศึกษา HMM ตั้งแต่ 3 สเตท จนถึง 18 สเตท และเพิ่มขึ้นทีละ 1 สเตท ในขณะเดียวกันก็กำหนดให้ HMM แต่ละขนาดมีการย้ายข้ามสเตทสูงสุดได้ทั้ง 3 แบบ คือมีการย้ายข้ามสเตทสูงสุดได้ 1สเตท, 2 สเตท และ 3 สเตท ซึ่งในการทดลองนี้ได้ทำการสร้างแบบจำลองขึ้นมาทั้งหมด 47 แบบจำลอง (แบบจำลองที่มี 3 สเตทมี 2 แบบจำลอง)

โดยในทุกๆแบบจำลองจะถูกสร้างขึ้นจากคำต้นแบบจำนวน 25 คำ ดังแสดงไว้ในตารางที่ 6.1 จากผู้ออกเสียง 10 คน เป็นชาย 5 คนและหญิง 5 คนออกเสียงคนละ 1 ครั้ง รวมเป็นเสียงต้นแบบจำนวนทั้งสิ้น 250 เสียง (ระดับเสียงละ 50 เสียง) และในการทดสอบการรู้จำเพื่อหารูปแบบ HMM ที่เหมาะสมนี้ จะวัดผลโดยนำเสียงต้นแบบเดิมทั้ง 250 เสียง เข้าไปทำการทดสอบการรู้จำอีกครั้ง ในทุกๆแบบจำลองที่สร้างขึ้นทั้ง 47 แบบจำลอง โดยผลการทดสอบแสดงได้ดังรูปที่ 6.8

ตารางที่ 6.1 กลุ่มคำที่ 1 ใช้ในการทดสอบหาแบบจำลอง HMM ที่เหมาะสม

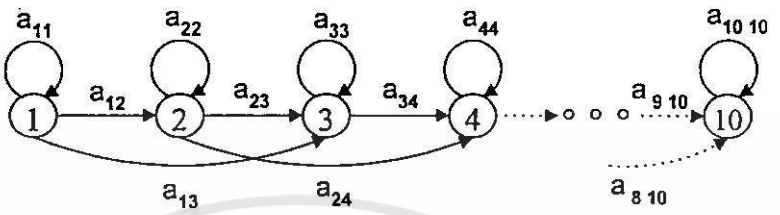
สามัญ	เอก	โท	ตรี	จัตวา
อา	อ่า	อ้า	อ๊า	อ๋า
อี	อี่	อี้	อี๊	อี๋
อู	อู่	อู้	อู๊	อู๋
เอ	เอ่	เอ้	เอ๊	เอ๋
โอ	โอ่	โอ้	โอ๊	โอ๋



รูปที่ 6.8 เปอร์เซนต์ความถูกต้องเมื่อมีการเปลี่ยนแปลงสเตท และการย้ายข้ามสเตทของ HMM

จากรูปที่ 6.8 แสดงให้เห็นว่าแบบจำลองที่มีจำนวนสเตทเพิ่มมากขึ้นจะให้ผลการรู้จำที่แม่นยำขึ้น และมีค่าค่อนข้างคงที่ตั้งแต่สเตทที่ 12 ขึ้นไป โดยแบบจำลองที่สร้างจาก HMM 4 สเตทให้ผลการรู้จำแม่นยำน้อยที่สุด ทั้งนี้เนื่องจากจำนวนสเตทมีผลโดยตรงต่อค่าพารามิเตอร์ B (ค่าความน่าจะเป็นของค่าปรากฏที่สามารถเป็นไปได้ภายในสเตท {-1, 0, 1}) ซึ่งจะเป็นตัวกำหนดรายละเอียดของข้อมูล โดยอยู่ในรูปของเมตริกซ์ขนาดเท่ากับ "จำนวนสเตทxระดับการควอนไตซ์" ดังนั้นถ้าสเตทมีจำนวนน้อยเมตริกซ์ B จะมีขนาดเล็ก ซึ่งนั่นหมายถึงรายละเอียดของข้อมูลในแบบจำลองจะน้อยตามลงไปด้วยเป็นผลให้การรู้จำมีความแม่นยำลดลง ในขณะเดียวกันแบบจำลอง HMM ที่สร้างจากสเตทจำนวนมาก จากรูปจะสังเกตุได้ว่าตั้งแต่สเตทที่ 12 ถึง 18 ให้ค่าการรู้จำสูงและค่อนข้างคงที่ ทั้งนี้เป็นผลเนื่องมาจากข้อมูลอินพุท คือค่าการควอนไตซ์ความถี่มีระดับเพียงแค่ 3 ระดับซึ่งให้ค่ารายละเอียดของข้อมูลน้อยเกินไปเมื่อเทียบกับจำนวนสเตท ทำให้ไม่ว่าจะเพิ่ม

จำนวนสเตทขึ้นเท่าใดก็ไม่ทำให้ผลการรู้จำแม่นยำเพิ่มขึ้น ซึ่งจากการทดลอง แบบจำลองที่ให้ผลการรู้จำดีที่สุดอยู่ที่ 10 สเตท โดยมีรูปแบบของการย้ายข้ามสเตทได้สูงสุด 2 สเตท แสดงได้ดังรูปที่ 6.9 โดยให้ผลการแม่นยำสูงสุดถึง 96.4 % เมื่อทำการทดสอบกับเสียงต้นแบบเดิม



รูปที่ 6.9 แบบจำลอง HMM ที่เหมาะสมกับอัลกอริธึมที่พัฒนาขึ้น

6.4 การสร้างแบบจำลองอ้างอิงเพื่อใช้ในการรู้จำระดับเสียงวรรณยุกต์

จากการทดลองในส่วนแรก เราจะได้ขั้นตอนในการดึงพารามิเตอร์ที่ต้องการซึ่งก็คือรูปแบบการเปลี่ยนแปลงของค่าความถี่มูลฐาน (เป็นตัวกำหนดระดับเสียงวรรณยุกต์ในคำหรือพยางค์ในภาษาไทย) และรูปแบบของ HMM ที่เหมาะสมกับอัลกอริธึมที่พัฒนาขึ้น

การทดลองในส่วนนี้จึงเป็นการสร้างแบบจำลองอ้างอิงที่ใช้ในการรู้จำระดับเสียง โดยใช้เสียงต้นแบบจำนวนทั้งสิ้น 925 เสียง จาก 5 แบบจำลอง (185x5) แบ่งออกเป็นแบบจำลองระดับเสียงสามัญ เสียงเอก เสียงโท เสียงตรี และเสียงจัตวา ระดับเสียงละ 185 เสียง โดยรูปแบบของคำที่ใช้เป็นคำพยางค์เดียว ดังแสดงไว้ในตารางที่ 6.2 ซึ่งเสียงต้นแบบที่ใช้ ได้จากผู้ออกเสียงจำนวนทั้งสิ้น 10 คน แบ่งเป็นชาย 5 คน และหญิง 5 คน ออกเสียงคำในตารางที่ 6.2 คนละ 1ครั้ง

เนื่องจากหน่วยเสียงวรรณยุกต์เป็นหน่วยเสียงซ้อนที่วางตัวอยู่เหนือหน่วยเสียงก้อง ดังนั้นรูปแบบของคำที่ใช้ในวิทยานิพนธ์นี้จึงพยายามใช้คำที่มี หน่วยเสียงพยัญชนะต้น หน่วยเสียงสระ และพยัญชนะสะกดต่างๆกัน [15] เพื่อให้ได้แบบจำลองอ้างอิงที่ครอบคลุมความหลากหลายมากที่สุด

ตารางที่ 6.2 กลุ่มคำที่ 2 ใช้ในการสร้างแบบจำลองอ้างอิง

คำที่	ระดับเสียง วรรณยุกต์				
	สามัญ	เอก	โท	ตรี	จัตวา
1	กิน	บอก	ป่า	น้ำ	อ๋อ
2	แก	ไก่	ลูก	ค้า	แจ๋ว
3	เกิน	ตี่	ย่า	ล้อ	ติ๋ม
4	ลอ	อ่าน	น่า	น้อง	โอ๋
5	ปี	เต่า	แก้	มด	กู๋
6	กาน	แต่	โก้	น้ำ	หุง
7	ออม	เตะ	กู้	คู้	ขาย
8	เกา	หนึ่ง	ยุ่ง	พื้น	หา
9	ดัง	ปู่	เก้า	ล้าน	หมอง
10	ปู	ต่อ	บี้	โย้	หมา
11	กาง	ดิบ	เอ้	นี้	หงอ
12	ลือ	แกะ	อ้อ	ไม้	เก๋
13	ไอ	กก	เก้อ	นก	หนี
14	ตา	ปาก	หนี้	วัด	หมู
15	เปีย	กีบ	เลข	นึก	ไข
16	ตึง	แตก	ลอก	นัด	หวาย
17	เป็น	ปัก	ทาก	มิด	เขียว
18	อำ	เอก	เชื่อ	แม้น	สอง
19	ตอ	ป่า	ชื่อ	ซื้อ	เก๋า

ผลที่ได้จากการทดลองในส่วนนี้ คือ แบบจำลองอ้างอิงจำนวน 5 แบบจำลอง ได้แก่

1. แบบจำลองอ้างอิง เสียงสามัญ แทนด้วยพารามิเตอร์ $\lambda_1 = (A_1, B_1, \pi)$
2. แบบจำลองอ้างอิง เสียงเอก แทนด้วยพารามิเตอร์ $\lambda_2 = (A_2, B_2, \pi)$
3. แบบจำลองอ้างอิง เสียงโท แทนด้วยพารามิเตอร์ $\lambda_3 = (A_3, B_3, \pi)$
4. แบบจำลองอ้างอิง เสียงตรี แทนด้วยพารามิเตอร์ $\lambda_4 = (A_4, B_4, \pi)$.
5. แบบจำลองอ้างอิง เสียงจัตวา แทนด้วยพารามิเตอร์ $\lambda_5 = (A_5, B_5, \pi)$

โดยรายละเอียดของค่าพารามิเตอร์ของแต่ละแบบจำลองแสดงได้ดังนี้

1. แบบจำลองการรู้จำเสียง สามัญ แทนด้วยพารามิเตอร์ $\lambda_1 = (A_1, B_1, \pi)$

ค่า ความน่าจะเป็นในการย้ายข้ามสเตท : A_1

$\mathbf{A}_{10x10}$

$$\begin{bmatrix}
0.2457 & 0.3879 & 0.3664 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.3505 & 0.3298 & 0.3197 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.3395 & 0.3317 & 0.3288 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.3396 & 0.3324 & 0.3281 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.3406 & 0.3329 & 0.3265 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.3400 & 0.3339 & 0.3261 & 0.0 & 0 0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.3415 & 0.3308 & 0.3277 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.3413 & 0.3397 & 0.3189 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5256 & 0.4744 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0000
\end{bmatrix}$$

ค่า ความน่าจะเป็นในการเกิดค่าปรากฏในแต่ละสเตท : B_1

$\mathbf{B}^{T}_{3x10} =$

$$\begin{bmatrix}
0.0057 & 0.3041 & 0.1868 & 0.1373 & 0.0820 & 0.0560 & 0.0580 & 0.0579 & 0.0610 & 0.0328 \\
0.9871 & 0.4429 & 0.5504 & 0.6961 & 0.7465 & 0.8013 & 0.8291 & 0.8495 & 0.8821 & 0.8532 \\
0.0070 & 0.2528 & 0.2626 & 0.1665 & 0.1714 & 0.1426 & 0.1128 & 0.0925 & 0.0562 & 0.1138
\end{bmatrix}$$

ค่า ความน่าจะเป็นในการเป็นสเตทเริ่มต้น (π_{1xN})

$\pi_{1x10} = [\,1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\,]$

2. แบบจำลองการรู้จำเสียง เอก แทนด้วยพารามิเตอร์ $\lambda_2 = (A_2, B_2, \pi)$

ค่า ความน่าจะเป็นในการย้ายข้ามสเตท : A_2

$\mathbf{A}_{10x10}$ =

$$\begin{bmatrix}
0.2138 & 0.4066 & 0.3796 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.4512 & 0.2826 & 0.2661 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.3483 & 0.3409 & 0.3107 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.3565 & 0.3370 & 0.3064 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.3729 & 0.3351 & 0.2920 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.3964 & 0.3305 & 0.2731 & 0.0 & 0 0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.4064 & 0.3114 & 0.2822 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.4026 & 0.3595 & 0.2378 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.6914 & 0.3086 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0000
\end{bmatrix}$$

ค่า ความน่าจะเป็นในการเกิดค่าปรากฏในแต่ละสเตท : B_2

$\mathbf{B}^{T}_{3x10}$ =

$$\begin{bmatrix}
0.0090 & 0.3297 & 0.1001 & 0.0266 & 0.0099 & 0.0030 & 0.0013 & 0.0009 & 0.0006 & 0.0252 \\
0.9815 & 0.4053 & 0.5416 & 0.6628 & 0.5387 & 0.4724 & 0.4750 & 0.4845 & 0.3824 & 0.6734 \\
0.0094 & 0.2649 & 0.3581 & 0.3104 & 0.4513 & 0.5245 & 0.5237 & 0.5145 & 0.6169 & 0.3015
\end{bmatrix}$$

ค่า ความน่าจะเป็นในการเป็นสเตทเริ่มต้น ($\pi_{1 x N}$)

$\pi_{1 x10}$ = [1 0 0 0 0 0 0 0 0 0]

3. แบบจำลองการรู้จำเสียง โท แทนด้วยพารามิเตอร์ $\lambda_3 = (A_3, B_3, \pi)$

ค่า ความน่าจะเป็นในการย้ายข้ามสเตท : A_3

$\mathbf{A}_{10\times10} =$

$$\begin{bmatrix}
0.3272 & 0.4357 & 0.2371 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.4885 & 0.3279 & 0.1835 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.5026 & 0.3257 & 0.1717 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.5065 & 0.3235 & 0.1699 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.5164 & 0.3203 & 0.1633 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5404 & 0.3206 & 0.1390 & 0.0 & 0 0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5778 & 0.2680 & 0.1542 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5950 & 0.3892 & 0.0158 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.9149 & 0.0851 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0000
\end{bmatrix}$$

ค่า ความน่าจะเป็นในการเกิดค่าปรากฏในแต่ละสเตท : B_3

$\mathbf{B}^{T}_{3\times10} =$

$$\begin{bmatrix}
0.0220 & 0.5147 & 0.5422 & 0.4991 & 0.4367 & 0.3288 & 0.2635 & 0.2238 & 0.0287 & 0.0049 \\
0.9610 & 0.3848 & 0.4365 & 0.4939 & 0.5611 & 0.6705 & 0.7359 & 0.7751 & 0.8983 & 0.2505 \\
0.0166 & 0.1004 & 0.0213 & 0.0070 & 0.0022 & 0.0006 & 0.0004 & 0.0010 & 0.0728 & 0.7445
\end{bmatrix}$$

ค่า ความน่าจะเป็นในการเป็นสเตทเริ่มต้น ($\pi_{1\times N}$)

$\pi_{1\times10} = [\,1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\,]$

4. แบบจำลองการรู้จำเสียง ตรี แทนด้วยพารามิเตอร์ $\lambda_4 = (A_4, B_4, \pi)$

ค่า ความน่าจะเป็นในการย้ายข้ามสเตท : A_4

A_{10x10} =

$$\begin{bmatrix}
0.2013 & 0.4186 & 0.3801 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.3801 & 0.3154 & 0.3045 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.3555 & 0.3307 & 0.3138 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.3672 & 0.3259 & 0.3069 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.3659 & 0.3290 & 0.3051 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.3730 & 0.3250 & 0.3020 & 0.0 & 0 0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.3787 & 0.3186 & 0.3026 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.3637 & 0.3408 & 0.2955 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.6217 & 0.3783 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0000
\end{bmatrix}$$

ค่า ความน่าจะเป็นในการเกิดค่าปรากฏในแต่ละสเตท : B_4

B^T_{3x10} =

$$\begin{bmatrix}
0.0105 & 0.4145 & 0.3252 & 0.1812 & 0.1710 & 0.1526 & 0.1386 & 0.1746 & 0.1087 & 0.3230 \\
0.9832 & 0.3584 & 0.4902 & 0.7012 & 0.7335 & 0.7680 & 0.7763 & 0.7493 & 0.8367 & 0.6611 \\
0.0062 & 0.2270 & 0.1845 & 0.1176 & 0.0954 & 0.0792 & 0.0850 & 0.0761 & 0.0545 & 0.0151
\end{bmatrix}$$

ค่า ความน่าจะเป็นในการเป็นสเตทเริ่มต้น (π_{1xN})

π_{1x10} = [1 0 0 0 0 0 0 0 0 0]

5. แบบจำลองการรู้จำเสียง จัตวา แทนด้วยพารามิเตอร์ $\lambda_5 = (A_5, B_5, \pi)$

ค่า ความน่าจะเป็นในการย้ายข้ามสเตท : A_5

A_{10x10} =

$$\begin{bmatrix}
0.3699 & 0.3934 & 0.2366 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.4070 & 0.3642 & 0.2288 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.4555 & 0.3552 & 0.1893 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.5634 & 0.2962 & 0.1404 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.6072 & 0.2669 & 0.1259 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.6116 & 0.2745 & 0.1139 & 0.0 & 0 0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.6050 & 0.2510 & 0.1440 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.6051 & 0.3789 & 0.0159 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.9002 & 0.0998 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0000
\end{bmatrix}$$

ค่า ความน่าจะเป็นในการเกิดค่าปรากฏในแต่ละสเตท : B_5

B^T_{3x10} =

$$\begin{bmatrix}
0.0232 & 0.1396 & 0.0239 & 0.0014 & 0.0004 & 0.0001 & 0.0006 & 0.0034 & 0.0408 & 0.6778 \\
0.9642 & 0.4651 & 0.4645 & 0.3154 & 0.2718 & 0.3316 & 0.4495 & 0.5250 & 0.8181 & 0.3127 \\
0.0125 & 0.3952 & 0.5115 & 0.6831 & 0.7277 & 0.6682 & 0.5498 & 0.4716 & 0.1409 & 0.0095
\end{bmatrix}$$

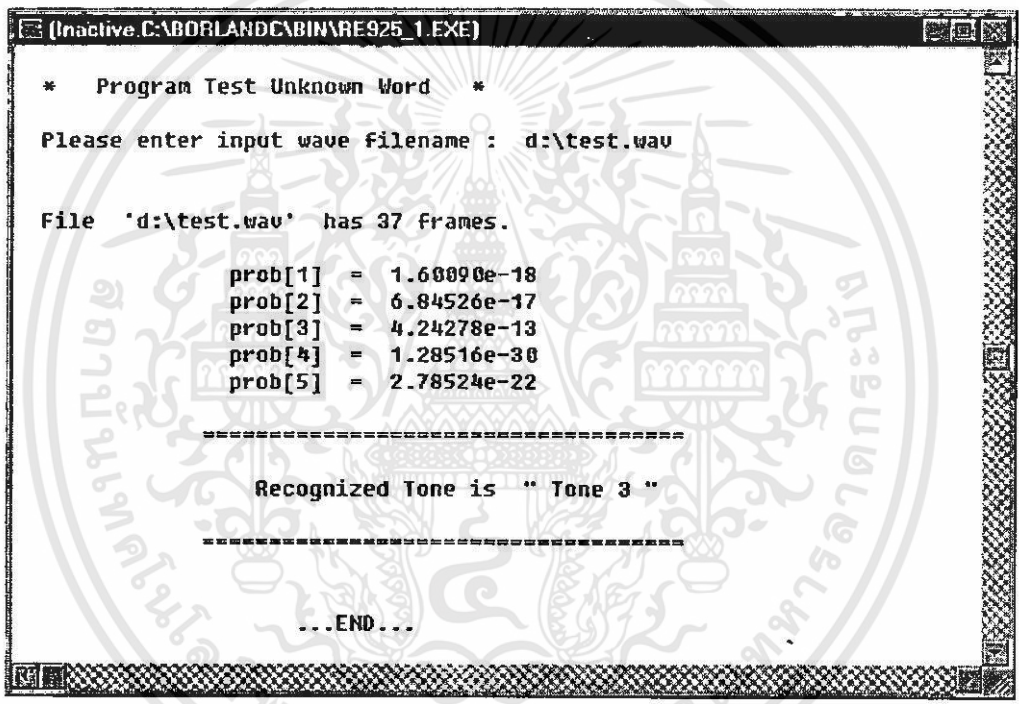
ค่า ความน่าจะเป็นในการเป็นสเตทเริ่มต้น (π_{1xN})

$$\pi_{1x10} = [\,1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\,]$$

ตัวอย่าง การแสดงผลของโปรแกรมในการทดสอบการรู้จำ

ในตัวอย่าง เป็นการทดสอบคำพยางค์เดียวที่มีเสียงวรรณยุกต์ โท โดยใช้คำว่า "เทือก" (test.wav) เมื่อนำเสียงที่ต้องการทดสอบไปคำนวณโดยใช้ค่าพารามิเตอร์ของแบบจำลองอ้างอิงของเสียงวรรณยุกต์ทั้ง 5 แบบจำลอง จะได้ค่าความน่าจะเป็นจำนวน 5 ค่า โดยแบบจำลองอ้างอิงใดให้ค่าความน่าจะเป็นสูงสุด จะถือว่าเสียงที่นำมาทดสอบมีระดับเสียงวรรณยุกต์เดียวกันกับแบบจำลองนั้น

จากตัวอย่าง จะเห็นว่าค่าความน่าจะเป็น (prob[ ]) ของโทนเสียงที่ 3 มีค่าสูงสุด คือ 4.24278×10^{-13} นั่นคือ คำที่นำมาทดสอบ (test.wav)จะถือว่ามีระดับเสียงวรรณยุกต์ "โท"



6.5 การทดสอบแบบจำลองอ้างอิง

ในการทดสอบแบบจำลองอ้างอิงที่สร้างขึ้นนี้ ได้แบ่งการทดสอบออกเป็น 2 กลุ่ม คือ

1. ทดสอบกับกลุ่มคำต้นแบบ
   - โดยใช้ เสียงต้นแบบเดิม
2. ทดสอบกับกลุ่มคำใหม่
   - โดยใช้ เสียงจากผู้ออกเสียงต้นแบบ
   - โดยใช้ เสียงจากผู้ออกเสียงกลุ่มใหม่

6.5.1 การทดสอบแบบจำลองอ้างอิงโดยใช้กลุ่มคำต้นแบบ

แบบจำลองอ้างอิงเสียงวรรณยุกต์ λ_1 - λ_5 ที่ถูกสร้างขึ้น จะถูกนำมาทดสอบการทำงานเพื่อพิสูจน์ว่าสามารถรู้จำระดับเสียงได้จริง โดยนำมาทดสอบกับกลุ่มคำต้นแบบและใช้เสียงต้นแบบเดิม ซึ่งแต่ละระดับเสียงใช้จำนวนเสียงในการทดสอบเท่ากันคือ 185 เสียงจากผู้ออกเสียง 10 คน และจากการทดสอบ ให้ผลการรู้จำถูกต้องเฉลี่ย 94.72 เปอร์เซนต์ ดังแสดงไว้ในตารางที่ 6.3

ตารางที่ 6.3 ผลการทดสอบแบบจำลองอ้างอิง โดยใช้กลุ่มคำและเสียงต้นแบบ จำนวน 925 เสียง

ผู้ออกเสียง	การรู้จำระดับเสียงถูกต้อง (%)					(%) ถูกต้องเฉลี่ย / คน
	สามัญ	เอก	โท	ตรี	จัตวา	
M1	77.78	94.44	100.00	77.78	100.00	**90.00**
M2	77.78	100.00	100.00	88.89	100.00	**93.33**
M3	100.00	94.44	100.00	77.78	100.00	**94.44**
M4	88.89	88.89	100.00	94.44	100.00	**94.44**
M5	72.22	77.78	100.00	72.22	100.00	**84.44**
W1	100.00	100.00	100.00	100.00	100.00	**100.00**
W2	94.74	89.47	100.00	100.00	100.00	**96.84**
W3	89.47	100.00	100.00	100.00	100.00	**97.89**
W4	100.00	94.74	100.00	100.00	100.00	**98.95**
W5	89.47	100.00	100.00	94.74	100.00	**96.84**
เฉลี่ย	**89.04**	**93.98**	**100.00**	**90.58**	**100.00**	**94.72**

หมายเหตุ M1- M5 แทนผู้ออกเสียงต้นแบบที่เป็นชาย คนที่ 1-5

W1- W5 แทนผู้ออกเสียงต้นแบบที่เป็นหญิง คนที่ 1-5

6.5.2 การทดสอบแบบจำลองอ้างอิงโดยใช้กลุ่มคำใหม่

กลุ่มคำใหม่ เป็นกลุ่มคำพยางค์เดียวจำนวนทั้งสิ้น 50 คำ ที่ไม่ซ้ำกับกลุ่มคำต้นแบบ ดังแสดงในตารางที่ 6.4 โดยถูกนำมาใช้ในการทดสอบ เพื่อพิสูจน์ว่าแบบจำลองอ้างอิงเสียงวรรณยุกต์ λ_1 - λ_5 ที่สร้างขึ้นนี้สามารถใช้ได้กับคำพยางค์เดียวทั่วไป และไม่จำกัดตัวบุคคล

ตารางที่ 6.4 กลุ่มคำที่ 3 ใช้ในการทดสอบแบบจำลองอ้างอิง

คำที่	ระดับเสียงวรรณยุกต์				
	สามัญ	เอก	โท	ตรี	จัตวา
1	ตอน	ถ่อ	เนิบ	เย้ย	ไถ
2	งา	สื่อ	ตุ้ย	ง้อ	แหวน
3	เอน	จอก	เทือก	แนะ	สาม
4	ดาว	เก่า	ลาบ	ท้อง	ผิว
5	ทอ	จุ่ม	อึ้ง	โม้	หนอ
6	แบ	ส่ง	ล่อ	เว้น	เขย
7	ดาย	เกิด	ยืด	มื้อ	หยี
8	นาน	แก่	เท่า	ยุ้ง	เหงา
9	ดู	จ่า	เลิก	แท้	ขุย
10	แนว	ตืด	โง่	ม้า	โขง

โดยในการทดสอบ ได้แบ่งการทดสอบออกเป็น 2 กรณี คือ

1. **ทดสอบโดยใช้เสียงจากผู้ออกเสียงต้นแบบ** จำนวน 400 เสียง จากผู้ออกเสียง 8 คน เป็นชาย 4 คน และหญิง 4 คน ออกเสียงกลุ่มคำใหม่ในตารางที่ 6.4 จำนวน 50 คำ คนละ 1 ครั้ง โดยในการทดสอบ ให้ผลการรู้จำถูกต้องเฉลี่ย 92.75 เปอร์เซนต์ ดังแสดงไว้ในตารางที่ 6.5
2. **ทดสอบโดยใช้เสียงจากผู้ออกเสียงกลุ่มใหม่** จำนวน 400 เสียง จากผู้ออกเสียง 8 คน เป็นชาย 4 คน และหญิง 4 คน ออกเสียงกลุ่มคำใหม่ในตารางที่ 6.4 จำนวน 50 คำ คนละ 1 ครั้ง โดยในการทดสอบ ให้ผลการรู้จำถูกต้องเฉลี่ย 91.75 เปอร์เซนต์ ดังแสดงไว้ในตารางที่ 6.6

ตัวอย่าง ทางเดินเสียงวรรณยุกต์ หรือการปลี่ยนแปลงของค่าความถี่มูลฐาน ของคำในตารางที่ 6.4 จากผู้ออกเสียง 16 คน ที่เป็นหญิง 8 คน และชาย 8 คน ได้แสดงไว้ในรูปที่ 6.10

ตารางที่ 6.5 ผลการรู้จำระดับเสียงวรรณยุกต์ จากการทดสอบโดยใช้เสียงจากผู้ออกเสียงต้นแบบ

ผู้ออกเสียง	การรู้จำระดับเสียงถูกต้อง (%)					(%) ถูกต้องเฉลี่ย / คน
	สามัญ	เอก	โท	ตรี	จัตวา	
M1	100.00	100.00	100.00	100.00	100.00	**100.00**
M2	90.00	90.00	90.00	100.00	100.00	**94.00**
M3	90.00	60.00	100.00	90.00	100.00	**88.00**
M4	100.00	90.00	100.00	70.00	100.00	**92.00**
W1	100.00	70.00	100.00	90.00	100.00	**92.00**
W2	100.00	90.00	100.00	90.00	100.00	**96.00**
W3	100.00	50.00	100.00	100.00	100.00	**90.00**
W4	90.00	90.00	100.00	70.00	100.00	**90.00**
เฉลี่ย	**96.25**	**80.00**	**98.75**	**88.75**	**100.00**	**92.75**

หมายเหตุ M1- M4 แทนเสียง จากกลุ่มผู้ออกเสียงต้นแบบที่เป็นชาย คนที่ 1-4

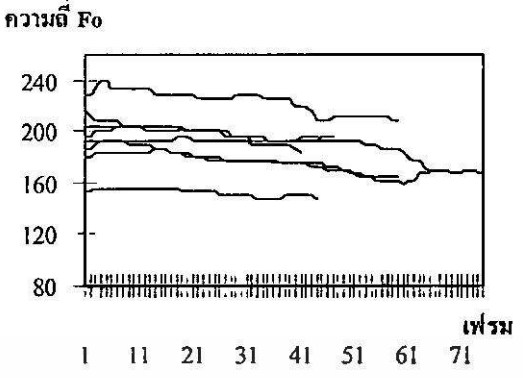
W1- W4 แทนเสียง จากกลุ่มผู้ออกเสียงต้นแบบที่เป็นหญิง คนที่ 1-4

ตารางที่ 6.6 ผลการรู้จำระดับเสียงวรรณยุกต์ จากการทดสอบโดยใช้เสียงจากผู้ออกเสียงกลุ่มใหม่

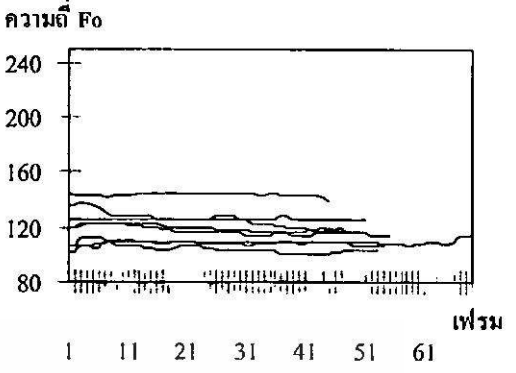
ผู้ออกเสียง	การรู้จำระดับเสียงถูกต้อง (%)					(%) ถูกต้องเฉลี่ย / คน
	สามัญ	เอก	โท	ตรี	จัตวา	
NM1	100.00	100.00	100.00	80.00	100.00	**96.00**
NM2	80.00	100.00	100.00	80.00	100.00	**92.00**
NM3	100.00	80.00	90.00	60.00	100.00	**86.00**
NM4	80.00	100.00	100.00	100.00	100.00	**96.00**
NW1	90.00	90.00	100.00	80.00	100.00	**92.00**
NW2	90.00	100.00	100.00	100.00	100.00	**98.00**
NW3	90.00	100.00	100.00	50.00	100.00	**88.00**
NW4	100.00	60.00	100.00	70.00	100.00	**86.00**
เฉลี่ย	**91.25**	**91.25**	**98.75**	**77.50**	**100.00**	**91.75**

หมายเหตุ NM1- NM4 แทนเสียง จากกลุ่มผู้ออกเสียงกลุ่มใหม่ที่เป็นชาย คนที่ 1-4

NW1- NW4 แทนเสียง จากกลุ่มผู้ออกเสียงกลุ่มใหม่ที่เป็นหญิง คนที่ 1-4

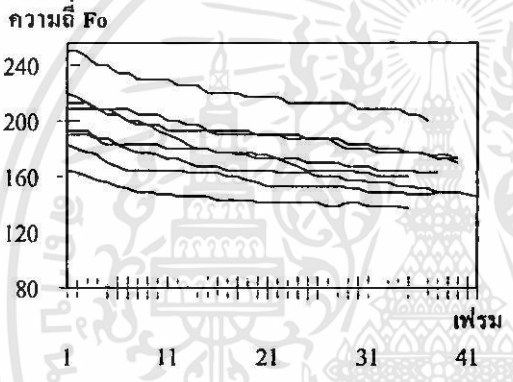


(a) หญิง

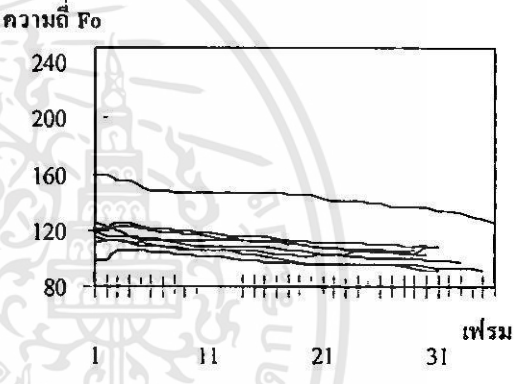


(f) ชาย

เสียง สามัญ "นาน"

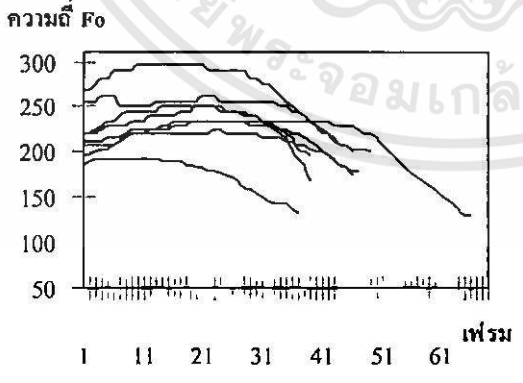


(b) หญิง

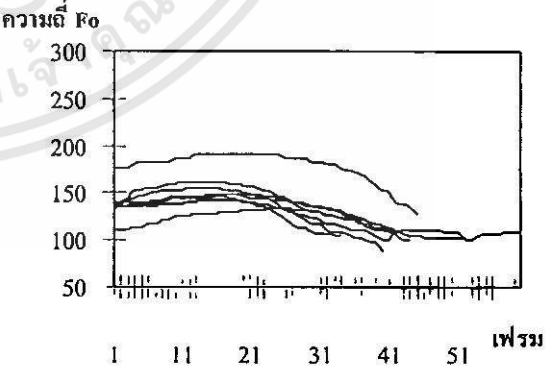


(g) ชาย

เสียง เอก "จอก"

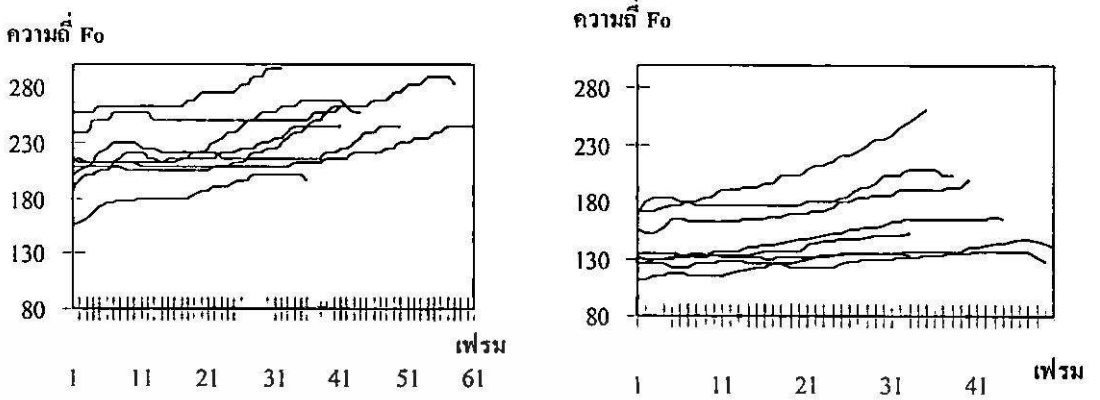


(c) หญิง

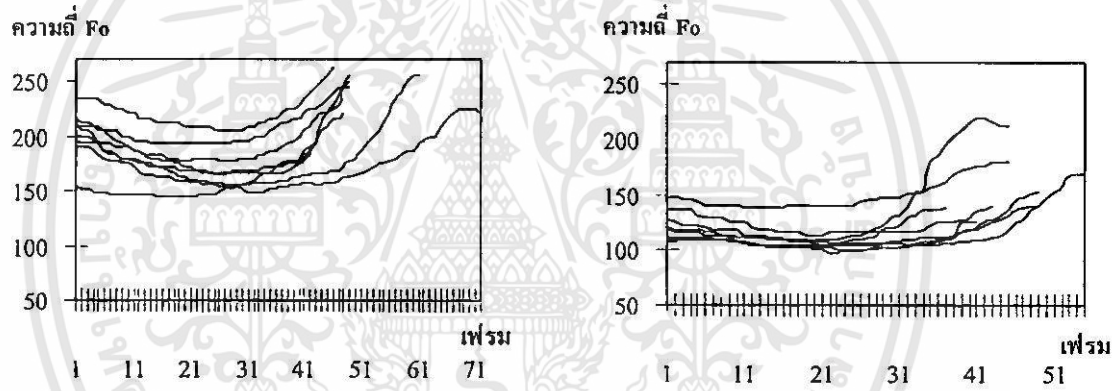


(h) ชาย

เสียง โท "ล่อ"



เสียง ตรี "ขุ้ง"



เสียง จัตวา "แหวน"

รูปที่ 6.10 ตัวอย่างทางเดินเสียงวรรณยุกต์ของคำที่เป็นเสียง สามัญ เอก โท ตรี และ จัตวา จากผู้ออกเสียง 16 คน โดย

(a-e) เป็นทางเดินเสียงวรรณยุกต์ของผู้ออกเสียงเพศหญิง 8 คน และ

(f-j) เป็นทางเดินเสียงวรรณยุกต์จากผู้ออกเสียงเพศชาย 8 คน

บทที่ 7
สรุปผล และข้อเสนอแนะ

วิทยานิพนธ์นี้เป็นการเสนอการสร้างแบบจำลองการรู้จำระดับเสียงวรรณยุกต์ของคำพยางค์เดียวแบบไม่จำกัดบุคคล โดยทำการวิเคราะห์ในโดเมนของเวลา และเนื่องจากหน่วยเสียงวรรณยุกต์เป็นหน่วยเสียงซ้อนวางตัวอยู่เหนือหน่วยเสียงก้อง ซึ่งคุณสมบัติเด่นที่สำคัญของเสียงก้องก็คือความเป็นคาบ ดังนั้นการวิเคราะห์ระดับเสียงในโดเมนของเวลาก็คือการหาลักษณะการเปลี่ยนแปลงของคาบพิทช์ หรืออีกนัยหนึ่งคือหาแนวทางการเปลี่ยนแปลงของค่าความถี่มูลฐานในเสียงนั้นๆนั่นเอง การนำเสนอในงานวิจัยนี้จะทำให้ทราบตั้งแต่ขั้นตอนการหาค่าความถี่มูลฐานจากสัญญาณเสียงพูด จนถึงการสร้างแบบจำลองอ้างอิงในการรู้จำระดับเสียงด้วยฮิดเดนมาร์คอฟโมเดล และทำการทดสอบการรู้จำโดยใช้ทั้งกลุ่มคำต้นแบบที่ใช้ในการสร้างแบบจำลอง และกลุ่มคำใหม่

7.1 การทดลอง

การทดลองแบ่งออกเป็น 3 ตอน คือ

1. การพัฒนาอัลกอริธึมที่ใช้ในการสร้างแบบจำลองการรู้จำระดับเสียงวรรณยุกต์

การหาค่าคาบพิทช์ของสัญญาณเสียง จะหาได้โดยใช้วิธีออโตโครีเลชั่นที่มีการคลิปสัญญาณ จากนั้นค่าพิทช์จะถูกเปลี่ยนให้อยู่ในรูปของค่าความถี่มูลฐาน รูปแบบการเปลี่ยนแปลงค่าความถี่มูลฐาน (หรือเรียกว่าเส้นทางเดินเสียงวรรณยุกต์) ที่แตกต่างกัน จะเป็นตัวบ่งบอกถึงระดับเสียงของวรรณยุกต์ที่แตกต่างกันในแต่ละคำหรือพยางค์ โดยแบบจำลองการรู้จำที่สร้างขึ้นมุ่งเน้นให้สามารถใช้ได้แบบไม่จำกัดเพศและตัวบุคคล ดังนั้นจึงใช้วิธีการคอวนไตซ์โดยทำการจัดกลุ่มค่าความถี่มูลฐานออกเป็น 3 ระดับตามแนวทางการเปลี่ยนแปลงของความถี่มูลฐานที่ เพิ่มขึ้น คงที่ หรือลดลงเมื่อเวลาเปลี่ยนไป เพื่อที่จะขจัดข้อจำกัดอันเนื่องมาจากความถี่มูลฐานที่แตกต่างกันระหว่างผู้ออกเสียงที่เป็นชายและหญิง อีกทั้งยังช่วยลดเนื้อที่ของหน่วยความจำในการจัดเก็บข้อมูลและทำให้การคำนวณทำได้เร็วขึ้นเมื่อเทียบกับการใช้ช่วงความถี่มูลฐานทั้งหมดมาสร้างแบบจำลอง

ทำการศึกษารูปแบบของ HMM ที่เหมาะสม ซึ่งจากการทดลองพบว่า HMM ขนาด 10 สเตทและมีการย้ายข้ามสเตทได้สูงสุดไม่เกิน 2 สเตท เป็นรูปแบบที่เหมาะสมกับอัลกอริธึมที่พัฒนาขึ้นมากที่สุด

2. การสร้างแบบจำลองอ้างอิง

การเลือกคำต้นแบบจะพยายามใช้คำที่มี หน่วยเสียงพยัญชนะต้น ความสั้น-ยาวของหน่วยเสียงสระ และหน่วยเสียงพยัญชนะสะกด ที่แตกต่างกัน เพื่อให้ได้แบบจำลองอ้างอิงที่ครอบคลุมความหลากหลายมากที่สุด โดยใช้คำจำนวนทั้งหมด 95 คำ จากผู้ออกเสียง 10 คน เป็น ชาย 5 คน และหญิง 5 คน ออกเสียงเพื่อใช้เป็นข้อมูลฝึกสอนในการสร้างแบบจำลองอ้างอิงจำนวนทั้งสิ้น 925 เสียง

3. การทดสอบแบบจำลองอ้างอิงที่สร้างขึ้น

แบ่งการทดสอบออกเป็น 3 กรณี คือ

3.1 ใช้กลุ่มคำต้นแบบ และเสียงต้นแบบเดิมที่นำมาสร้างแบบจำลอง

- ให้ผลการรู้จำถูกต้องเฉลี่ยคิดเป็นร้อยละ 94.72
- ให้ผลการรู้จำในแต่ละระดับเสียง สามัญ เอก โท ตรี จัตวา คิดเป็นร้อยละ 89.04, 93.98, 100.00, 90.58, 100.00 ตามลำดับ

3.2 ใช้กลุ่มคำใหม่ ที่ออกเสียงโดยผู้ที่ออกเสียงต้นแบบ 8 คน เป็นชาย 4คน หญิง 4 คน

- ให้ผลการรู้จำถูกต้องเฉลี่ยคิดเป็นร้อยละ 92.75
- ให้ผลการรู้จำในแต่ละระดับเสียง สามัญ เอก โท ตรี จัตวา คิดเป็นร้อยละ 96.25, 80.00, 98.75, 88.75, 100.00 ตามลำดับ

3.3 ใช้กลุ่มคำใหม่ ที่ออกเสียงโดยผู้ที่ออกเสียงกลุ่มใหม่ 8 คน เป็นชาย 4คน หญิง 4 คน

- ให้ผลการรู้จำถูกต้องเฉลี่ยคิดเป็นร้อยละ 91.75
- ให้ผลการรู้จำในแต่ละระดับเสียง สามัญ เอก โท ตรี จัตวา คิดเป็นร้อยละ 91.25, 91.25, 98.75, 77.50, 100.00 ตามลำดับ

7.2 ข้อสังเกต ปัญหาที่พบในการทดลอง และข้อแสนอแนะ

การใช้วิธีการควอนไตซ์การเปลี่ยนแปลงค่าความถี่มูลฐานออกเป็น 3 ระดับ พบว่าให้ผลดีในการรู้จำระดับเสียงวรรณยุกต์แบบไม่จำกัดเพศและบุคคล โดยให้ผลการรู้จำเฉลี่ยมากกว่า 90 % ในทุกกรณี และจากที่กล่าวมาแล้วว่าหน่วยเสียงวรรณยุกต์แบ่งออกเป็น 2 ประเภท คือ

หน่วยเสียงวรรณยุกต์เปลี่ยนระดับ ได้แก่ระดับเสียงโท และเสียงจัตวา ซึ่งจากการทดสอบพบว่าหน่วยเสียงวรรณยุกต์เปลี่ยนระดับให้การรู้จำถูกต้องสูงสุดในทุกกรณี เนื่องจากมีเส้นทางเดินเสียงวรรณยุกต์ที่มีลักษณะเฉพาะตัวและไม่ใกล้เคียงกับระดับเสียงใด ทำให้การรู้จำถูกต้องถึง 99 เปอร์เซนต์ในเสียงโท และ 100 เปอร์เซนต์ในเสียงจัตวา

หน่วยเสียงวรรณยุกต์คงระดับ ได้แก่ระดับเสียง สามัญ เอก และตรี ซึ่งจากการทดลองพบว่า สำเนียง หรือท่วงทำนองในการออกเสียงมีส่วนสำคัญกับการรู้จำหน่วยเสียงทั้ง 3 นี้เป็นอย่างมาก โดยผลการรู้จำที่ผิดพลาด สาเหตุหนึ่งมาจากการออกเสียงไม่ชัดเจน เมื่อออกเสียงเพี้ยนไปหรือออกเสียงเอื้อนในส่วนท้ายพยางค์มากไปจะทำให้การรู้จำผิดพลาด ซึ่งในการพัฒนาให้มีความแม่นยำการรู้จำสูงขึ้น อาจใช้ความแตกต่างของค่าความถี่มูลฐานในช่วงต้นพยางค์และช่วงท้ายของพยางค์เป็นพารามิเตอร์ร่วมในการรู้จำ ตัวอย่างเช่นในกรณีของเสียงสามัญและเสียงเอก ที่มีการเปลี่ยนแปลงของค่าความถี่มูลฐานลดลงเหมือนกัน แต่ในระดับเสียงเอกจะมีการลดลงในช่วงท้ายพยางค์ต่ำกว่าเสียงสามัญมาก และในเสียงตรี ช่วงท้ายพยางค์จะมีค่าความถี่มูลฐานสูงกว่าช่วงต้นพยางค์ ดังนั้นการนำค่าความแตกต่างของความถี่มูลฐานในช่วงต้นและท้ายพยางค์มาเป็นพารามิเตอร์ร่วมในการรู้จำ คาดว่าอาจทำให้การรู้จำระดับเสียง สามัญ เอก และตรี มีความแม่นยำสูงขึ้น

จากงานวิจัยทั้งหมดที่ได้กล่าวมาแล้ว สามารถนำไปพัฒนาใช้ในการรู้จำระดับเสียงวรรณยุกต์หรือระดับเสียงสูง-ต่ำในภาษาไทยที่มีสำเนียงในภาคต่างๆได้ โดยต้องทำการเก็บตัวอย่างเสียงต้นแบบที่เป็นสำเนียงในภาคนั้นๆ เพื่อทำการสร้างแบบจำลองอ้างอิงใหม่โดยใช้กระบวนการต่างๆที่ได้อธิบายไว้ในตอนต้น

เอกสารอ้างอิง

1. ชวดี อิศรปรีดา และคณะ. "การรู้จำเสียงพูด." ปริญญานิพนธ์วิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง. 2538.
2. กรุณา แก้วสมศรี และคณะ. "การต่อหมายเลขโทรศัพท์โดยใช้เสียง" ปริญญานิพนธ์วิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง. 2539.
3. ธันวา ศรีประโมง. "การวิเคราะห์เสียงพูดภาษาไทยในแกนความถี่ฮาร์โมนิค" วิทยานิพนธ์วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า บัณฑิตวิทยาลัย, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง. 2537.
4. ณัฐกร ทับทอง. "การรู้จำคำพูดภาษาไทย โดยใช้ลักษณะบ่งความต่างของหน่วยเสียง" วิทยานิพนธ์วิทยาศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ บัณฑิตวิทยาลัย, จุฬาลงกรณ์มหาวิทยาลัย. 2538.
5. ทัศเวท วีระวัฒน์. "การรู้จำเสียงคำไทยเฉพาะบุคคล" วิทยานิพนธ์วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า บัณฑิตวิทยาลัย, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง. 2541.
6. วิไลวรรณ ขนิษฐานันท์. **ภาษาและภาษาศาสตร์.** พิมพ์ครั้งที่ 5. กรุงเทพฯ : สำนักพิมพ์มหาวิทยาลัยธรรมศาสตร์. 2533.
7. ราตรี ธันวารชร. **การศึกษาภาษาไทยตามแนวภาษาศาสตร์ เล่ม๑ เสียงและระบบเสียงในภาษาไทย.** กรุงเทพฯ : คณะศิลปศาสตร์ มหาวิทยาลัยธรรมศาสตร์. 2537.
8. ชาคริต อนันทราวัน. **หลักภาษาไทย.** กรุงเทพฯ : โอเดียนสโตร์. 2524.
9. อภิชาต ตั้งทางธรรม. "การเปลี่ยนความเร็วของเสียงพูด." การประชุมทางวิศวกรรมไฟฟ้า ครั้งที่ 17, 2537. หน้า 329-332.
10. Rabiner L.R., Schafer R.W. **Digital Processing of Speech Signals.** New Jersey : Prentice-Hall, Inc. 1978.
11. Rabiner L.R., Cheng M.J. et. al. "A Comparative Performance Study of Several Pitch Detection Algorithms.", IEEE Trans. Acoust., Speech, Signal Processing, vol.ASSP-24, no.5, Oct. 1976. pp. 399-418.
12. Rosenfeld A., Kak A.C. **Digital Picture Processing.** Orlando Florida : Academic Press, Inc. 1982.
13. Thomas W. **Voice and Speech Processing.** Newyork : McGraw-Hill, Inc. 1987.

14. Rabiner L.R., Juang B.H. **Fundamentals of Speech Recognition.** New Jersey : Prentice Hall, Inc. 1993.

15. เรืองเดช ปันเขื่อนขัติย์. **แบบทดสอบ ระบบเสียงวรรณยุกต์ภาษาถิ่น.** สถาบันวิจัยภาษา และวัฒนธรรมเพื่อพัฒนาชนบท มหาวิทยาลัยมหิดล. 2532.

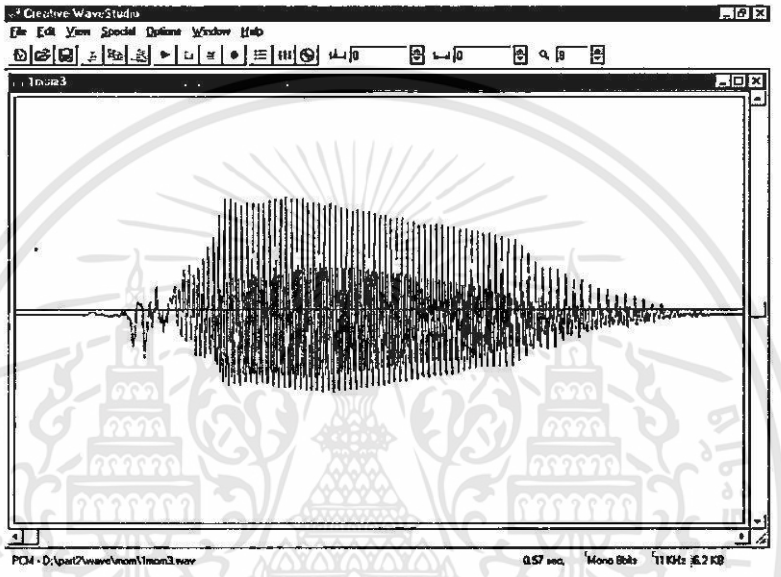
16. Yang W.J., Lee J.C., Chang Y.C.and Wang H.C., "Hidden Markov Model for Mandarin Lexical Tone Recognition," IEEE Trans. Acoust., Speech, and Signal Processing, vol. 36, July 1988, pp.988-992.

ภาคผนวก

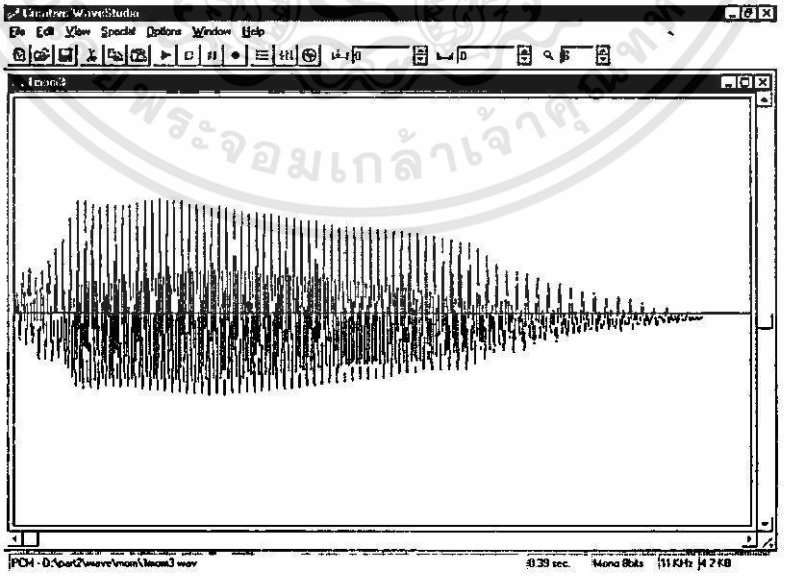
ภาคผนวก ก
ตัวอย่างการตัดคำที่มีพยัญชนะต้น และพยัญชนะสะกด

สัญญาณเสียงคำว่า **"ชื่อ"**

ก่อนตัดคำ

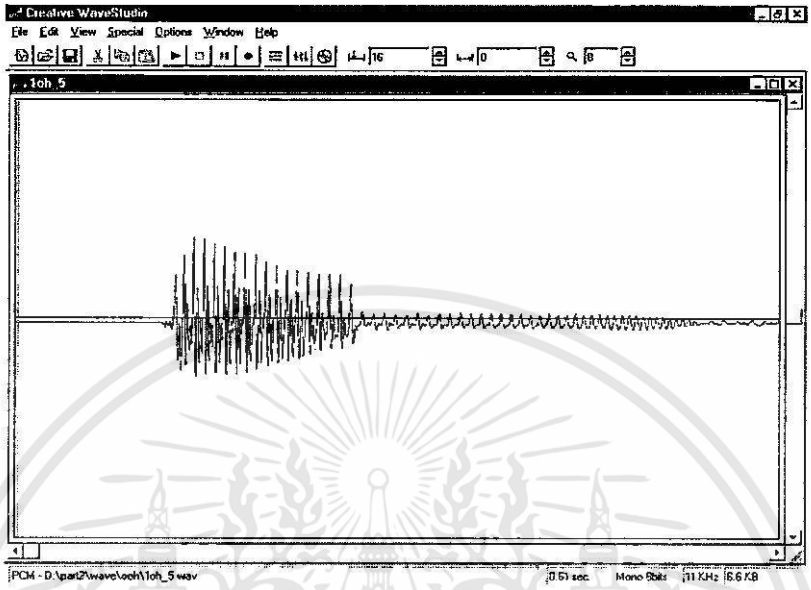


หลังตัดคำ

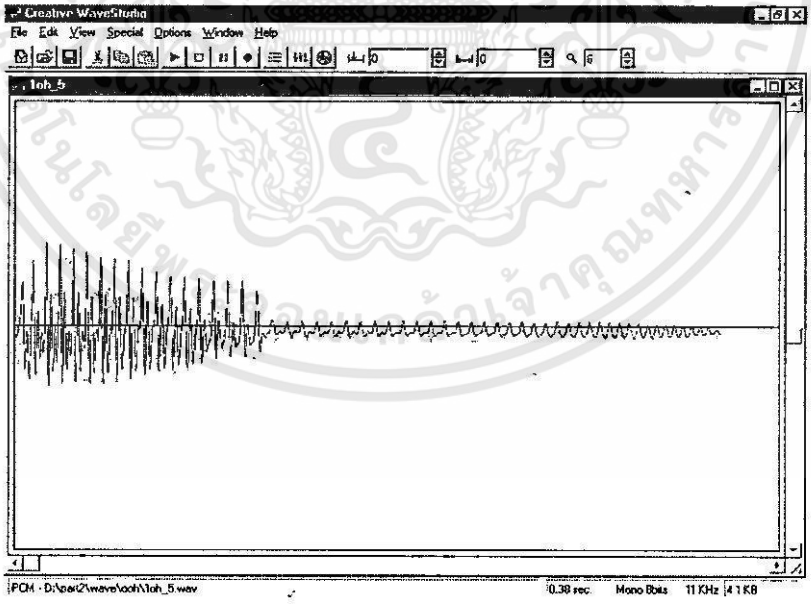


สัญญาณเสียงคำว่า “ติ๋ม”

ก่อนตัดคำ

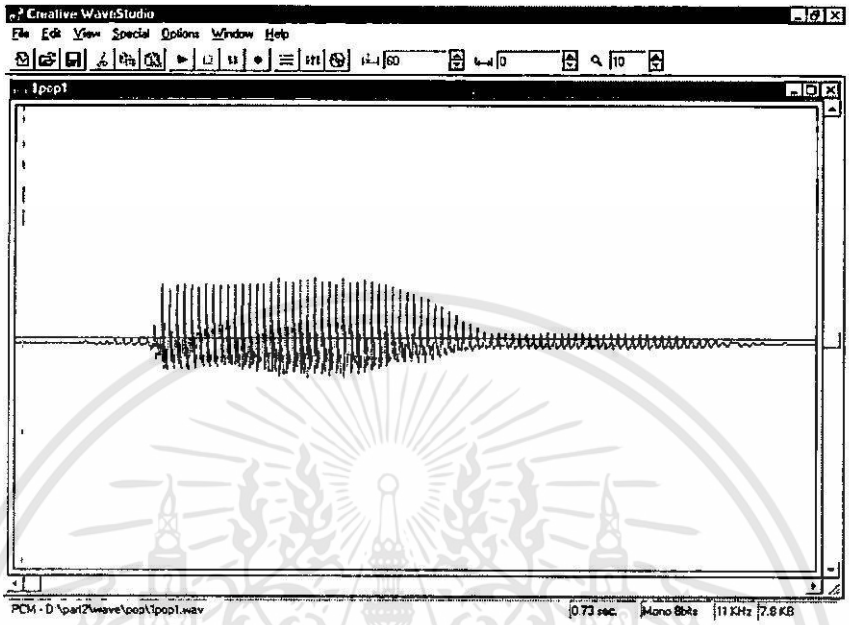


หลังตัดคำ

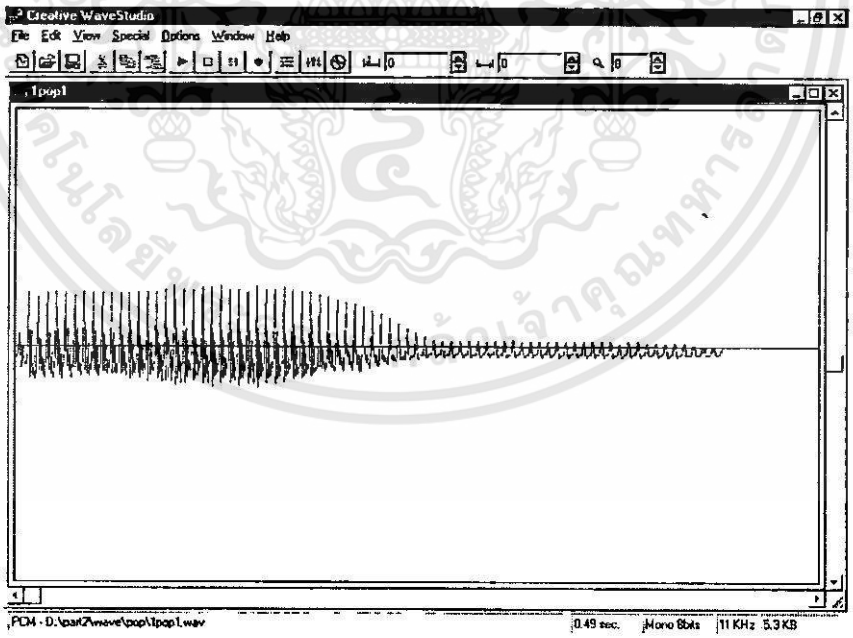


ตัวอย่างสัญญาณเสียงคำว่า “ดัง”

ก่อนตัดคำ



หลังตัดคำ



ภาคผนวก ข
โปรแกรมที่พัฒนาขึ้นในการวิจัย

1. โปรแกรมการหาค่า พิทช์ และควอนไตซ์ค่าความถี่มูลฐาน

```
**************************************************************************
        Program PITCH :    Input: file.wav   ; mono 8 bits, sampling at 11.025 KHz
                           Output: Quantized 3 Level of Fundamental Frequency
**************************************************************************
#include <stdio.h>
#include <stdlib.h>
#include <alloc.h>
#include <conio.h>
#include <io.h>
#include <math.h>
#include <dir.h>
#include <string.h>
#define Fs          11025  /* sampling frequency */
#define FRAME     300
#define SHIFT     100
#define FIRST_CALL 1
#define NEXT_CALL 2
#define LAG       250
#define N         120  /* approximate number of frame */
FILE *input_wav_file;
FILE *output_diff_file;
FILE *output_bi_file;

char input_wav_filename[50],output_bi_filename[50];
long length;
int  number_of_frame;
int  nf,fl,j,k,l,c,m,n,position;
```

```
int  signal[FRAME];
int  buffer;
int  freq[N],median[N];
float lp[FRAME],s[FRAME],x[FRAME],R[LAG];
float y1[100],y1_max,y2[100],y2_max;
float cl;
/*************************************************************************/
void open_all_file(void)
{
 clrscr();
 printf("     Program find fundamental frequency or pitch period      \n");
 printf("\n\n  Please enter input wave filename (and directory): ");

 /* get signal input : wave file */
 gets(input_wav_filename);
  if((input_wav_file = fopen(input_wav_filename,"rb"))==NULL)
  {
          printf(" \7  can't open input wave file ! \n");
          exit(-1);
  }
/* open output binary file */
 printf("\n  please enter output filename (filename.bi):  ");
 gets(output_bi_filename);
    if((output_bi_file = fopen(output_bi_filename,"wb"))==NULL)
  {
          printf("  \7 can't open output parameter file ! \n");
          exit(-1);
  }
}
/*************************************************************************/
void prepare_data(void)
{
```

```
  /* find length of data */
 length = filelength(open(input_wav_filename,0));
 length -= 44L;  /* bypass header of wav file; 44L means 44 items in long */

 printf("\n  File '%s' has %d samples.\n\n",input_wav_filename,length);
 fseek(input_wav_file,44L,SEEK_CUR);   /* set pointer to first data */

/* find number of frame (is round of loop) */
 number_of_frame =(int)(length-FRAME+SHIFT)/SHIFT;
 printf("  File '%s' has %d frames.\n",input_wav_filename,number_of_frame);

/* set initial data */
 signal[0] = (int)fgetc(input_wav_file);
}
/*****************************************************************************/
void find_lowpass(void)    /* lowpass filtered cutoff 900 Hz */
{
 static int called = FIRST_CALL;
  if(called==FIRST_CALL)
   {
    for(fl=0;fl<FRAME;fl++)
    {
      signal[fl] = (int)fgetc(input_wav_file);
            if(fl-1<0) {signal[fl-1] = 0; lp[fl-1] = 0.0;}
            if(fl-2<0) {signal[fl-2] = 0; lp[fl-2] = 0.0;}
            if(fl-3<0) {signal[fl-3] = 0; lp[fl-3] = 0.0;}
            if(fl-4<0) {signal[fl-4] = 0; lp[fl-4] = 0.0;}
           if(fl-5<0) {signal[fl-5] = 0; lp[fl-5] = 0.0;}
            if(fl-6<0) {signal[fl-6] = 0; lp[fl-6] = 0.0;}
```

```
    lp[fl] = (0.01369*signal[fl])-(0.05112*signal[fl-1])+(0.09864*signal[fl-2])-(0.11959*signal
             [fl-3])+(0.09864*signal[fl-4])-(0.05112*signal[fl-5])+(0.01369*signal[fl-6])+
             (5.06706*lp[fl-1])-(11.14066*lp[fl-2])+(13.53499*lp[fl-3])-(9.56224*lp[fl-4])+
             (3.72104*lp[fl-5])-(0.62337*lp[fl-6]);
}
  for(fl=0;fl<(FRAME-10);fl++)
  {
    lp[fl] = lp[fl+10];
  }
  called = NEXT_CALL;
}
 else  /* Second and later called,Use some previous data again */
 {
  for(fl=0;fl<(FRAME-SHIFT);fl++)
  {
     signal[fl] = signal[fl+SHIFT];
          lp[fl] = lp[fl+SHIFT];
  }
  for(fl=(FRAME-SHIFT);fl<FRAME;fl++)
  {
     signal[fl] = (int)fgetc(input_wav_file);
          if(fl-1<0) {signal[fl-1] = 0; lp[fl-1] = 0.0;}
          if(fl-2<0) {signal[fl-2] = 0; lp[fl-2] = 0.0;}
          if(fl-3<0) {signal[fl-3] = 0; lp[fl-3] = 0.0;}
          if(fl-4<0) {signal[fl-4] = 0; lp[fl-4] = 0.0;}
          if(fl-5<0) {signal[fl-5] = 0; lp[fl-5] = 0.0;}
          if(fl-6<0) {signal[fl-6] = 0; lp[fl-6] = 0.0;}
```

```
        lp[fl]=(0.01369*signal[fl])-(0.05112*signal[fl-1])+(0.09864*signal[fl-2])-(0.11959*signal
              [fl-3])+ (0.09864*signal[fl-4])-(0.05112*signal[fl-5])+(0.01369*signal[fl-6])+
              (5.06706*lp[fl-1])-(11.14066*lp[fl-2])+(13.53499*lp[fl-3])-(9.56224*lp[fl-4])+
              (3.72104*lp[fl-5])-(0.62337*lp[fl-6]);
    }
  }
}
/*****************************************************************************/
void find_origin_line(void)
{
float  x_max,x_min,x_origin;
  x_max = lp[0];
  x_min = lp[0];
  for(fl=0;fl<FRAME;fl++)
  {
    if(x_max<lp[fl]) {x_max = lp[fl];}
    if(x_min>lp[fl]) {x_min = lp[fl];}
  }
  x_origin = (x_max + x_min)/2;
  for(fl=0;fl<FRAME;fl++)
  {
    s[fl] = lp[fl] - x_origin;
   }
}
/*****************************************************************************/
void abs_first_data(void)  /*find absolute peak of 100 data first and last */
{
 /* 100 data fisrt of frame */
  for(fl=0;fl<100;fl++)
  {
   y1[fl] = fabs(s[fl]);
  }
```

```
  y1_max = y1[0];

 for(n=0;n<100;n++)
  {
    if(y1_max<y1[n]) {y1_max = y1[n];}
  }
}
/*****************************************************************************/
void abs_last_data(void)
{
     /* 100 data last of frame */
  for(fl=200;fl<300;fl++)
  {
    y2[fl] = fabs(s[fl]);
   }
  y2_max = y2[200];

  for(n=200;n<300;n++)
  {
    if(y2_max<y2[n]) {y2_max = y2[n];}
  }
}
/*****************************************************************************/
void find_clipping_level(void)
{
/* select minimum peak */
 float  abs_peak;
  if(y1_max <= y2_max) { abs_peak = y1_max;}
  if(y2_max <  y1_max) { abs_peak = y2_max;}

  /* use clipping level 65% of minimum peak (abs_peak) */
```

```
  cl = (abs_peak*0.65);
}
/*****************************************************************************/
void clipping_signal(void)
{
    for(fl=0;fl<FRAME;fl++)
  {
    x[fl] = s[fl];

    if(x[fl]>=0.0)
    {
      if(x[fl]<=cl)
          x[fl] = 0.0;
      else
          x[fl] =  1;  /* x[fl]-cl; */
    }
    else
    {
      if(x[fl]>=(-cl))
          x[fl] = 0.0;
      else
          x[fl] = -1;  /* x[fl]+cl; */
    }
  }
}
/*****************************************************************************/
void autocor_relation(void)
{    /* K is lag */
 for(k=0;k<LAG;k++)
 {
   R[k] = 0.0;
```

```
    for(n=0;n<=(FRAME-1-k);n++)
    {
      R[k] += x[n]*x[n+k];
    }
  }
}
/*****************************************************************************/
void normalize(void)
{
 float  R_max;
 R_max = R[0];

  for(k=0;k<LAG;k++)
  {
    if(R_max<R[k]) {R_max = R[k];}
  }
  for(k=0;k<LAG;k++)
  {
    R[k] = R[k]/R_max;
   }
}
/*****************************************************************************/
void find_pitch_period(void)
{
float  R_pitch;
 R_pitch = R[30];

  for(n=30;n<200;n++)
  {
    if(R_pitch<R[n]) {R_pitch = R[n];}
  }
   position = 0;
```

```
  for(n=30;n<200;n++)
  {
    if(R[n]==R_pitch) {position = n;}
  }
 /* fine fundamental frequency from pitch position */
   buffer  =  (int)(Fs/position);
   freq[nf] =  buffer;
}
/*****************************************************************************/
void median_filter(void)
{
int a,b,c;
//  a = 0;
   b = 0;
//  c = 0;
 median[N] = 0;
for(j=0;j<(number_of_frame-2);j++)
{
 if(freq[j] <= freq[j+1])
   {
         if(freq[j+1] <= freq[j+2])
         {
   //      a = freq[j];
           b = freq[j+1];
   //      c = freq[j+2];
         }
         else
         {
           if(freq[j] <= freq[j+2])
           {
   //        a = freq[j];
```

```
            b = freq[j+2];
//          c = freq[j+1];
        }
        else
            {
//          a = freq[j+2];
            b = freq[j];
//          c = freq[j+1];
            }
    }
}
else
{
        if(freq[j] <= freq[j+2])
        {
//      a = freq[j+1];
        b = freq[j];
//      c = freq[j+2];
        }
        else
        {
          if(freq[j+1] <= freq[j+2])
          {
//          a = freq[j+1];
            b = freq[j+2];
//          c = freq[j];
        }
            else
            {
//          a = freq[j+2];
            b = freq[j+1];
//          c = freq[j];
```

```
        }
      }
    }
/* move value b into median[N] */
  median[j] = b;
}
}
/*****************************************************************************/
void close_all_file(void)
{
  fclose(input_wav_file);
  fclose(output_diff_file);
  fclose(output_bi_file);
}
/*****************************************************************************/
void main(void)
{
 int   Z1[N],Z[N]; /* approximated number of frame as long as 120 frames */
 open_all_file();
 prepare_data();
 buffer = 0;
 freq[N] = 0;
 for(nf=0;nf<number_of_frame;nf++)
 {
    find_lowpass();
    find_origin_line();
    abs_first_data();
    abs_last_data();
    find_clipping_level();
    clipping_signal();
    autocor_relation();
    normalize();
```

```
    find_pitch_period();
  }
 median_filter();
  /* fine different value of previous frame */
  for(nf=0;nf<(number_of_frame-2);nf++)
  {
    if(nf==0)
          Z1[nf] = 0;
    else
          Z1[nf] = median[nf] - median[nf-1];
    }
  /* change value Z1 by -1,0,1 */
  for(nf=0;nf<(number_of_frame-2);nf++)
  {
    /* check value of Z1 */
    if(Z1[nf]==0) { Z[nf] = 0; }
    if(Z1[nf] >0) { Z[nf] = 1; }
    if(Z1[nf] <0) { Z[nf] = -1;}

    if(c==20) {printf("Type  Enter.\n");getch();c=1;};
    printf("Freq[%d] = %d   median[%d] = %d    Z1[%d] = %d   Z[%d] = %d\n",nf,freq
          [nf],nf,median[nf],nf,Z1[nf],nf,Z[nf]);c++;
}
 fwrite(&Z,2,(number_of_frame-2),output_bi_file);
  close_all_file();
}
/*****************************************************************************/
```

2. โปรแกรมการสร้างแบบจำลองอ้างอิง

```
/*************************************************************************/
   Hidden Markov Models for Speech Recognition :  TRAINING PART
/*************************************************************************/
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <time.h>
#define N        10              /* number of state in each model */
#define R        3               /* range of data -1,0,1  */
#define ROUND    50              /* number of reestimate round */
#define MIN_B  0.00001           /* minimum value of b[][]  */
FILE  *observation_file;
FILE  *model_a_file;
FILE  *model_b_file;
FILE  *model_text_file;
char  observation_file_name[40];
char  model_a_file_name[40];
char  model_b_file_name[40];
char  model_text_file_name[40];
int   *T;     /* number of observation for each training */
int   *O;     /* training sequence */
int   max,number_of_observation;
int   i,j,k,m,w,v,t;
int   min_k,max_k;
float  Pi[N] = {1,0,0,0,0,0,0,0,0,0};
float  a[N][N], aprime[N][N];
float  b[N][R], bprime[N][R];
float  imb[N];
double x[N][N], y[N][R], z[N];
double Aprime[N][N], AMprime[N];
```

```
double Bprime[N][R], BMprime[N], l[N];
double Plog[ROUND];   /* for check distorsion in each round */
double  *alpha,           /* alpha */
        *pre_alpha,       /* alpha pre scale */
        *alphas,          /* alpha scaled */
        *beta,            /* beta */
        *pre_beta,        /* beta pre scale  */
        *betas,           /* beta scaled*/
        *sc,
        *c;               /* scaling coefficient*/
float    *plog;           /* for check distorsion */
/*****************************************************************************/
void open_file(void)
{
 int temp;
 clrscr();
 /* open observation input file */
 printf("\n Please enter observation (input) file name : ");
 gets(observation_file_name);
 if ((observation_file = fopen(observation_file_name,"rb")) == NULL)
  {    printf("\n\7 □ Cannot open input file !"); exit(1);   };

 /* open model  a output file */
 printf("\n Please enter Model (o/p) A file name : ");
 gets(model_a_file_name);
 if ((model_a_file = fopen(model_a_file_name,"wb")) == NULL)
  {  printf("\n\7 □ Cannot open output A file !"); exit(1);   };

 /* open model b output file */
 printf("\n Please enter Model (o/p) B file name : ");
 gets(model_b_file_name);
 if ((model_b_file = fopen(model_b_file_name,"wb")) == NULL)
```

```
  {     printf("\n\7 □ Cannot open output B file !"); exit(1);   };

/* open output text file */
printf("\n Please enter Model TEXT (output) file name : ");
gets(model_text_file_name);
if ((model_text_file = fopen(model_text_file_name,"wt")) == NULL)
  {     printf("\n\7 □ Cannot open output text file !"); exit(1);   };

/* count number of observation in observation file */
number_of_observation = 0;
for (;;)
{
  fread(&temp,2,1,observation_file);
  if (temp != 0 )  number_of_observation++; else break;
  if (feof(observation_file))
   {     printf("\n\7 □ Invalid input file !\n"); exit(1);     }
}
//printf(" number_of_observation = %d ",number_of_observation);
rewind(observation_file);
/* read each observation length */
T = (int *) calloc(number_of_observation,sizeof(int));
if (T == NULL)
  {     printf("\n\7 □ Cannot allocate memory for data !\n"); exit(1);   };
fread(T,2,number_of_observation,observation_file);
fread(&temp,2,1,observation_file);  /* for bypass FLAG (-1) */
/* find max of number of observation */
max = T[0];
for (i=0;i<number_of_observation;i++)
 {   if (max <= T[i]){ max = T[i];}            }
O = (int *) calloc(number_of_observation*max,sizeof(int));
if (O == NULL)
  {     printf("\n\7 □ Cannot allocate memory for data !\n"); exit(1);   };
```

```
 /* read observation data */
 for (i=0;i<number_of_observation;i++)
  {    fread(&O[i*max],2,T[i],observation_file);  }
 /* allocate memory */
 alpha      = calloc(max*N,sizeof( double));
 pre_alpha = calloc(max*N,sizeof( double));
 alphas     = calloc(max*N,sizeof( double));
 beta       = calloc(max*N,sizeof( double));
 pre_beta   = calloc(max,  sizeof( double));
 betas      = calloc(max*N,sizeof( double));
    sc      = calloc(max,  sizeof( double));
    c       = calloc(max,  sizeof( double));
  plog      = calloc(number_of_observation,sizeof(float));
 if ((alpha||pre_alpha||alphas||beta||pre_beta||betas||sc||c||plog) == NULL)
  {    printf("\n\7 □ Cannot allocate memory for data !\n");  exit(1);  }
}
/*************************************************************************/
void random_abvalue(void)
{
 float sum_a[N],sum_b[N];
 /*-------- define some value = 0 in a matrix---------*/
 /*---------it's Left-Right Model conditions ---------*/
a[0][0]=(1.0/3.0); a[0][1]=(1.0/3.0); a[0][2]=(1.0/3.0); a[0][3]=0; a[0][4]=0; a[0][5]=0;
a[0][6]=0;   a[0][7]=0;   a[0][8]=0;   a[0][9]=0;
a[1][0]=0;   a[1][1]=(1.0/3.0); a[1][2]=(1.0/3.0); a[1][3]=(1.0/3.0); a[1][4]=0; a[1][5]=0;
a[1][6]=0;   a[1][7]=0;   a[1][8]=0;   a[1][9]=0;
a[2][0]=0;   a[2][1]=0;   a[2][2]=(1.0/3.0); a[2][3]=(1.0/3.0); a[2][4]=(1.0/3.0);  a[2][5]=0;
a[2][6]=0; a[2][7]=0;  a[2][8]=0;   a[2][9]=0;
a[3][0]=0;   a[3][1]=0;   a[3][2]=0;  a[3][3]=(1.0/3.0); a[3][4]=(1.0/3.0); a[3][5]=(1.0/3.0);
a[3][6]=0; a[3][7]=0;  a[3][8]=0;   a[3][9]=0;
a[4][0]=0;   a[4][1]=0;   a[4][2]=0;  a[4][3]=0;   a[4][4]=(1.0/3.0); a[4][5]=(1.0/3.0);
a[4][6]=(1.0/3.0); a[4][7]=0; a[4][8]=0;   a[4][9]=0;
```

```
a[5][0]=0;  a[5][1]=0;  a[5][2]=0; a[5][3]=0;  a[5][4]=0; a[5][5]=(1.0/3.0); a[5][6]=(1.0/3.0);
a[5][7]=(1.0/3.0); a[5][8]=0; a[5][9]=0;
a[6][0]=0;  a[6][1]=0;  a[6][2]=0; a[6][3]=0;  a[6][4]=0; a[6][5]=0;  a[6][6]=(1.0/3.0);
a[6][7]=(1.0/3.0); a[6][8]=(1.0/3.0); a[6][9]=0;
a[7][0]=0;  a[7][1]=0;  a[7][2]=0;  a[7][3]=0;  a[7][4]=0;a[7][5]=0;  a[7][6]=0;
a[7][7]=(1.0/3.0); a[7][8]=(1.0/3.0); a[7][9]=(1.0/3.0);
a[8][0]=0;  a[8][1]=0;  a[8][2]=0;  a[8][3]=0;  a[8][4]=0;a[8][5]=0;  a[8][6]=0;  a[8][7]=0;
a[8][8]=(1.0/2.0); a[8][9]=(1.0/2.0);
a[9][0]=0;  a[9][1]=0;  a[9][2]=0; a[9][3]=0;  a[9][4]=0; a[9][5]=0;  a[9][6]=0;  a[9][7]=0;
 a[9][8]=0;   a[9][9]=1.00;
////////////////////////////////////////////////////////////
/* random start b[][] value */
 for(i=0;i<N;i++)
 {
  for(k=0;k<R;k++)      /* fix all value in b[i][k] = 1/3 */
    {   b[i][k] = (1.0/3.0);   }
 }
 /* find a_prime and b_prime */
 for(i=0;i<N;i++)
  {
    sum_a[i] = 0;
    sum_b[i] = 0;
    for(j=0;j<N;j++) {sum_a[i] += a[i][j];}
    for(k=0;k<R;k++) {sum_b[i] += b[i][k];}
  }
 for(i=0;i<N;i++)
  {
    for(j=0;j<N;j++)
        {  aprime[i][j] = a[i][j]/sum_a[i];  }
    for(k=0;k<R;k++)
        {  bprime[i][k] = b[i][k]/sum_b[i]; }
  }
```

```
}
/*****************************************************************************/
void copy_abprime_to_ab(void)
{
 for(i=0;i<N;i++)
     for(j=0;j<N;j++)
          {  a[i][j] = aprime[i][j];  }
  for(i=0;i<N;i++)
     for(k=0;k<R;k++)
          {  b[i][k] = bprime[i][k];   }
}
/*****************************************************************************/
void find_alpha_beta_value(void)
{
 double sum_alf;
 /* Alpha Initialization */
 for(i=0;i<N;i++)
  {   alpha[i] = Pi[i] * b[i][(O[v*max+0])+1];   }
 /* Alpha Induction */
 for(t=0;t<(T[v]-1);t++)
 {
  for(j=0;j<N;j++)
  {
    sum_alf = 0;
     for(i=0;i<N;i++)
     {   sum_alf    += alpha[(t*N)+i] * a[i][j];      /* lf[t*N+i] */    }
    alpha[(t+1)*N+j] = sum_alf * b[j][(O[(v*max)+(t+1)])+1];
  }
 }
 /* Beta Initialization */
 for(i=0;i<N;i++)
 {   beta[(T[v]-1)*N+i] = 1.0;  }
```

```
 /* Beta Induction */
 for(t=T[v]-2;t>=0;t--)
 {
  for(i=0;i<N;i++)
  {
   beta[t*N+i] = 0;
   for(j=0;j<N;j++)
   {     beta[t*N+i] += a[i][j]*b[j][(O[(v*max)+(t+1)])+1]*beta[(t+1)*N+j];     }
  }
 }
}
/*************************************************************************/
void scaling_alpha_beta(void)
{
 for (i=0;i<(max*N);i++)  pre_alpha[i] = 0;
 for (i=0;i<max    ;i++)   pre_beta[i] = 0;
 /* find sum coefficient */
 for(i=0;i<T[v];i++)
 {
   sc[i] = 0;
   for(j=0;j<N;j++)
   {     sc[i] += alpha[i*N+j];     }
 }
 /*-------------- find alpha scale ----------*/
 /* at  t = 0  */
 c[0] = 1/sc[0];
 for(i=0;i<N;i++)
 {
   pre_alpha[i] = alpha[i];
   alphas[i]    = c[0] * alpha[i];
 }
 /* at t=1 to T */
```

```
  for(t=1;t<T[v];t++)
   {
     c[t] = 0;
     for(i=0;i<N;i++)
     {      pre_alpha[t*N+i] = 0;
            for(j=0;j<N;j++)
            {    pre_alpha[(t*N)+i] += alphas[(t-1)*N+j]*a[j][i]*b[i][(O[v*max+t])+1];   }
            c[t] += pre_alpha[t*N+i];
      }
     c[t] = 1/c[t];
     for(i=0;i<N;i++)
     {    alphas[t*N+i] = c[t] * pre_alpha[t*N+i];      }
   }
  /*---------------- find beta scale ----------------*/
  pre_beta[T[v]-1] = c[T[v]-1];
  for(t=T[v]-2;t>=0;t--)
  {  pre_beta[t] = pre_beta[t+1] * c[t];  }
  for(t=0;t<T[v];t++)
  {
     for(i=0;i<N;i++)
     {    betas[(t*N)+i] = pre_beta[t] * beta[(t*N)+i];    }
  }
}
/*************************************************************************/
void find_logP(void)
{
  plog[v] = 0;
  for(t=0;t<T[v];t++)
    plog[v] += log10(c[t]);
    plog[v]  = -plog[v];
    Plog[w] +=  plog[v];
}    /*****************************************************************/
```

```
void find_A_AM_B_BM_prime(void)
{
 double q,G;
 /*------------ find A_prime and AM_prime ------------*/
 for(i=0;i<N;i++)
 for(j=0;j<N;j++)
 {
   x[i][j] = 0;
   for(t=0; t<T[v]-1;t++)
   {   x[i][j] += alphas[t*N+i] * a[i][j] * b[j][(O[v*max+(t+1)])+1] * betas[(t+1)*N+j];     }
   Aprime[i][j] += (double) x[i][j];
}
 for(i=0;i<N;i++)
 {
   z[i] = 0;
   for(t=0;t<T[v]-1;t++)
   {
     q = 0;
         for(j=0;j<N;j++)
               q+= alphas[t*N+i] * a[i][j] * b[j][(O[v*max+(t+1)])+1] * betas[(t+1)*N+j];
           z[i] += q;
   }
   AMprime[i] += (double) z[i];
 }
 /*---------------- find B_prime and BM_prime ----------------*/
 for(j=0;j<N;j++)
 for(k=0;k<R;k++)
 {       y[j][k] = 0;
         l[j]   = 0;
         for(t=0;t<T[v]-1;t++)
         {
           G = 0;
```

```
            for(i=0;i<N;i++)
              G += alphas[t*N+j]*a[j][i]*b[i][(O[v*max+(t+1)])+1]*betas[(t+1)*N+i];
            if(((O[v*max+t])+1) == k)    y[j][k] += G;
            l[j] += G;
          }
         /* final value */
         if(((O[v*max+(T[v]-1)])+1) == k)  y[j][k] += alphas[(T[v]-1)*N+j];
         l[j] += alphas[(T[v]-1)*N+j];
  }
  for(i=0;i<N;i++)
   for(j=0;j<R;j++)
   {    Bprime[i][j] += (double) y[i][j];  }
  for(i=0;i<N;i++)
   {    BMprime[i] += (double) l[i];   }
}
/*****************************************************************************/
void find_new_abvalue(void)
{ double bcomplex;
  /*---------- find new_aprime ----------*/
  for(i=0;i<N;i++)
  {
   for(j=0;j<N;j++)
   {      aprime[i][j] = (double)( Aprime[i][j]/AMprime[i] );   }
  }
  /*---------- find new_bprime ----------*/
  for(j=0;j<N;j++)
  {   for(k=0;k<R;k++)
      {   /* improve b_prime value */
          if(Bprime[j][k] <= MIN_B) {Bprime[j][k] = MIN_B;}
          bprime[j][k] = Bprime[j][k]/BMprime[j];
      }
  }       /*************************************************/
```

```
/* check combination of probability of  bprime[i][j] */
 for (i=0;i<N;i++)
 {
   bcomplex = 0;
   for (j=0;j<R;j++)
   {     bcomplex += bprime[i][j];    }
 }
/*********************************************************/
 for(i=0;i<N;i++)
 { imb[i] = 0;
    for(k=0;k<R;k++)
    {   imb[i] += bprime[i][k];   }
 }
 for(i=0;i<N;i++)
 {
    for(k=0;k<R;k++)
    {   bprime[i][k] /= imb[i];   }
 }
}   /*********************************************************************/
void display_a_b_pi_parameter(void)
{ double bcomplex;
 for (i=0;i<N;i++)
 {
   for (j=0;j<N;j++)
   {   printf("A[%d][%d]=%12.10g  \n ",i,j,aprime[i][j]);    }
    printf("\n Press any key ...");
    getch();
    printf("\n");
 }
 for (i=0;i<N;i++)
 {
   for (j=0;j<R;j++)
```

```
    {     printf("B[%d][%d]=%12.10g  \n",i,j,bprime[i][j]);   }
    printf("\n Press any key ...");
    getch();
    printf("\n");
   }
  printf("Pi = ");
  for (i=0;i<N;i++)
    printf("%3.2f  ",Pi[i]);
   // fwrite(&Pi,4,N,model_file);
  printf("\n");
  for (i=0;i<N;i++)
   { bcomplex = 0;
     for (j=0;j<R;j++) bcomplex += bprime[i][j];
     printf("B complex [%d] = %12.10g\n",i+1,bcomplex);
   }
}   /*****************************************************************/
void save_model_file(void)
{
  fwrite(&aprime,4,N*N,model_a_file);
  fwrite(&bprime,4,N*R,model_b_file);
//  fwrite(&Pi,4,N,model_file);
  for (i=0;i<N;i++)
  { for (j=0;j<N;j++)
     {  fprintf(model_text_file,"A [%3d][%3d] = %.10g\n",i,j,aprime[i][j]);    }
    fprintf(model_text_file,"\n");
  }
  for (i=0;i<N;i++)
  { for (j=0;j<R;j++)
     {  fprintf(model_text_file,"B [%3d][%3d] = %.10g\n",i,j,bprime[i][j]);   }
    fprintf(model_text_file,"\n");
  }
  for (i=0;i<N;i++)
```

```
  {   fprintf(model_text_file,"Pi[%d]=%.2f  ",i+1,Pi[i]); }
  fprintf(model_text_file,"\n");
  fclose(observation_file);
  fclose(model_a_file);
  fclose(model_b_file);
  fclose(model_text_file);
  free(O);
  free(T);
  free(alpha);
  free(pre_alpha);
  free(alphas);
  free(beta);
  free(pre_beta);
  free(betas);
  free(sc);
  free(c);
  free(plog);
}  /*****************************************************************/
void main(void)
{
  open_file();
  random_abvalue();
  for(i=0;i<N;i++)
  {  for(j=0;j<N;j++)
     {  Aprime[i][j] = 0; AMprime[i] = 0;    }
  }
  for(i=0;i<N;i++)
  { for(k=0;k<R;k++)
    {  Bprime[i][k] = 0; BMprime[i] = 0;   }
  }
  for (w=0;w<50;w++)
  {  printf("## w = %d\n",w+1);
```

```
    copy_abprime_to_ab();
    for (v=0;v<number_of_observation;v++)
    {
            find_alpha_beta_value();
             scaling_alpha_beta();
             find_logP();
             find_A_AM_B_BM_prime();
    }
    find_new_abvalue();
  }
 display_a_b_pi_parameter();
 save_model_file();
}
/*****************************************************************************/
```

3. โปรแกรมทดสอบการรู้จำ คำศัพท์ทั่วไป

```
/*****************************************************************/
        Program:  Test Unknown Word,    Using  HMM " 10 " States and  2 step
/*****************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <alloc.h>
#include <conio.h>
#include <io.h>
#include <math.h>
#define  Fs       11025
#define  FRAME    300
#define  SHIFT    100
#define  FIRST_CALL 1
```

```
#define NEXT_CALL 2
#define LAG     250
#define M       120  /* maximum of frame's member */
#define N       10   /* state */
#define K       3
#define STORED_MODEL 5 /* number of store model */
FILE *input_wav_file;
FILE *parameter_file;
FILE *model_a_file;
FILE *model_b_file;
FILE *unknown_word_file;
char input_wav_file_name[40];
char parameter_file_name[]   = "para.pt";
char unknown_word_file_name[] = "para.pt";
char *model_a_file_name[] = {"d:\\model\\tone1.api","d:\\model\\tone2.api",
                             "d:\\model\\tone3.api","d:\\model\\tone4.api",
                             "d:\\model\\tone5.api"};

char *model_b_file_name[] = {"d:\\model\\tone1.bpi","d:\\model\\tone2.bpi",
                             "d:\\model\\tone3.bpi","d:\\model\\tone4.bpi",
                             "d:\\model\\tone5.bpi"};
char *name[STORED_MODEL] = {"model_1","model_2","model_3","model_4","model_5"};
int  T;
int  *O;
double *ar;
double *path;
double *dt;
int   model,max_index,buffer;
int   number_of_frame,position;
int   nf,fl,i,j,k,l,c,m,n,t;
int   signal[FRAME];
int   freq[M],median[M];
```

```
long  length;
float  Pi[N] = {1,0,0,0,0,0,0,0,0,0};
float  a[N][N],b[N][K];
float  lp[FRAME],s[FRAME],x[FRAME];
float  y1[100],y2[100],y1_max,y2_max,cl;
double d[N],dmax;
double prob[STORED_MODEL];
/*************************************************************************/
void open_all_file(void)
{
clrscr();
printf("\n\n  *   Program Test Unknown Word   *   \n");
printf("\n  Please enter input wave filename :  ");
gets(input_wav_file_name);
 if((input_wav_file = fopen(input_wav_file_name,"rb"))==NULL)
 {
  printf("\7  Can't open input file! \n");
  exit(-1);
 }
 if((parameter_file = fopen(parameter_file_name,"wb"))==NULL)
 {
  printf("\7  Can't open output parameter file! \n");
  exit(-1);
 }
}
/*************************************************************************/
void prepare_data(void)
{
 /* find length of data */
length = filelength(open(input_wav_file_name,0));
length -= 44L;  /* bypass header of wav file;44L means 44 items in long */
fseek(input_wav_file,44L,SEEK_CUR);  /* set pointer to first data */
```

```
/* find number of frame (is round of loop) */
number_of_frame =(int)(length-FRAME+SHIFT)/SHIFT;
printf("\n\n  File  '%s'  has %d frames.\n\n",input_wav_file_name,number_of_frame);
/* set initial data */
signal[0] = (int)fgetc(input_wav_file);
}
/*************************************************************************/
void find_lowpass(void)  /* lowpass filtered cutoff 900 Hz */
{
 static int called = FIRST_CALL;
  if(called==FIRST_CALL)
  {
    for(fl=0;fl<FRAME;fl++)
    {
     signal[fl] = (int)fgetc(input_wav_file);
          if(fl-1<0) {signal[fl-1]=0; lp[fl-1]=0.0;}
          if(fl-2<0) {signal[fl-2]=0; lp[fl-2]=0.0;}
          if(fl-3<0) {signal[fl-3]=0; lp[fl-3]=0.0;}
          if(fl-4<0) {signal[fl-4]=0; lp[fl-4]=0.0;}
          if(fl-5<0) {signal[fl-5]=0; lp[fl-5]=0.0;}
          if(fl-6<0) {signal[fl-6]=0; lp[fl-6]=0.0;}
          lp[fl] = (0.01369*signal[fl])-(0.05112*signal[fl-1])+(0.09864*signal[fl-2])-
           (0.11959*signal[fl-3])+(0.09864*signal[fl-4])-(0.05112*signal[fl-5])+
           (0.01369*signal[fl-6])+(5.06706*lp[fl-1])-(11.14066*lp[fl-2])+(13.53499*lp[fl-3])-
           (9.56224*lp[fl-4])+(3.72104*lp[fl-5])-(0.62337*lp[fl-6]);
    }
    for(fl=0;fl<(FRAME-10);fl++)
    {  lp[fl] = lp[fl+10];   }
   called = NEXT_CALL;
  }
   else  /* second and later called,use some previous data again */
  {
```

```
    for(fl=0;fl<(FRAME-SHIFT);fl++)
    {
     signal[fl] = signal[fl+SHIFT];
          lp[fl] = lp[fl+SHIFT];
    }
    for(fl=(FRAME-SHIFT);fl<FRAME;fl++)
    {
     signal[fl] = (int)fgetc(input_wav_file);
          if(fl-1<0) {signal[fl-1]=0; lp[fl-1]=0.0;}
          if(fl-2<0) {signal[fl-2]=0; lp[fl-2]=0.0;}
          if(fl-3<0) {signal[fl-3]=0; lp[fl-3]=0.0;}
          if(fl-4<0) {signal[fl-4]=0; lp[fl-4]=0.0;}
          if(fl-5<0) {signal[fl-5]=0; lp[fl-5]=0.0;}
          if(fl-6<0) {signal[fl-6]=0; lp[fl-6]=0.0;}

          lp[fl] =  (0.01369*signal[fl])-(0.05112*signal[fl-1])+(0.09864*signal[fl-2])-
          (0.11959*signal[fl-3])+(0.09864*signal[fl-4])-(0.05112*signal[fl-5])+
          (0.01369*signal[fl-6])+(5.06706*lp[fl-1])-(11.14066*lp[fl-2])+(13.53499*lp[fl-3])-
          (9.56224*lp[fl-4])+(3.72104*lp[fl-5])-(0.62337*lp[fl-6]);
    }
  }
}
/*****************************************************************/
void find_origin_line(void)
{
 float x_max, x_min ,x_origin;
 x_max = lp[0];
 x_min = lp[0];
 for(fl=0;fl<FRAME;fl++)
 {
   if(x_max<lp[fl]) {x_max = lp[fl];}
   if(x_min>lp[fl]) {x_min = lp[fl];}
```

```
 }
 x_origin = (x_max + x_min)/2;
 for(fl=0;fl<FRAME;fl++)
 {
  s[fl] = lp[fl] - x_origin;
 }
}
/*************************************************************************/
void abs_first_data(void)  /* find absolute peak of 100 data first and last */
{
 /* 100 data first of frame */
 for(fl=0;fl<100;fl++)
 {
   y1[fl] = fabs(s[fl]);
 }
 y1_max = y1[0];
 for(n=0;n<100;n++)
 {
   if(y1_max<y1[n]) {y1_max = y1[n];}
 }
}
/*************************************************************************/
void abs_last_data(void)
{
 /* 100 last of frame */
 for(fl=200;fl<300;fl++)
 {  y2[fl] = fabs(s[fl]); }
 y2_max = y2[200];
 for(n=200;n<300;n++)
 {    if(y2_max<y2[n]) {y2_max = y2[n];} }
}
/*************************************************************************/
```

```
void find_clipping_level(void)
{
  /* select minimun peak */
float abs_peak;
if(y1_max<=y2_max) {abs_peak = y1_max;}
if(y2_max<y1_max)  {abs_peak = y2_max;}
/* use clipping level 65% of minimum peak (abs_peak) */
cl = (abs_peak*0.65);
}
/*************************************************************************/
void clipping_signal(void)
{
 for(fl=0;fl<FRAME;fl++)
 {
  x[fl] = s[fl];
  if(x[fl]>=0.0)
   {
                if(x[fl]<=cl)
                        x[fl] = 0.0;
        else
                        x[fl] = 1;
   }
  else
   {
                if(x[fl]>=(-cl))
                        x[fl] = 0.0;
    else
                        x[fl] = -1;
   }
 }
}
/*************************************************************************/
```

```
void autocor_relation(void)
{
float  R_max,R_pitch;
float  R[LAG];
 /* k is lag */
 for(k=0;k<LAG;k++)
 {
  R[k] = 0.0;
  for(n=0;n<=(FRAME-1-k);n++)
  {
    R[k] += x[n]*x[n+k];
  }
 }
////////////////////////////////////////////////////////////
/*  normalize  */
 R_max = R[0];
 for(k=0;k<LAG;k++)
 {
  if(R_max<R[k]) {R_max = R[k];}
 }
 for(k=0;k<LAG;k++)
 {
  R[k] = R[k]/R_max;
 }
////////////////////////////////////////////////////////////
/*  find_pitch_period  */
 R_pitch = R[30];
 for(n=30;n<200;n++)
 {
  if(R_pitch<R[n]) {R_pitch = R[n];}
 }
 position = 0;
```

```
for(n=30;n<200;n++)
{
 if(R[n]==R_pitch) {position = n;}
}
/* find pitch frequency */
 buffer = (int)(Fs/position);
freq[nf] = buffer;
}
/*****************************************************************************/
void median_filter(void)
{
int a,b,c;
b = 0;
median[M] = 0;
for(j=0;j<(number_of_frame-2);j++)
{
  if(freq[j] <= freq[j+1])
  {
       if(freq[j+1] <= freq[j+2])
       {        b = freq[j+1]; }
       else
       {
         if(freq[j] <= freq[j+2])
         {        b = freq[j+2];        }
         else
         {        b = freq[j];    }
       }
}
 else
  {
            if(freq[j] <= freq[j+2])
            {        b = freq[j];            }
```

```
        else
                {
                if(freq[j+1] <= freq[j+2])
                {               b = freq[j+2];              }
          else
          {             b = freq[j+1];              }
   }
  }
    median[j] = b;
 }
}
/*****************************************************************************/
void close_all_pitch_file(void)
{
 fclose(input_wav_file);
 fclose(parameter_file);
}
/*****************************************************************************/

void pitch(void)
{
 int flag = 0;
 int Z1[M],Z[M];
 open_all_file();
 prepare_data();
 buffer  = 0;
 freq[M] = 0;
  for(nf=0;nf<number_of_frame;nf++)
  {
   find_lowpass();
   find_origin_line();
   abs_first_data();
```

```
    abs_last_data();
    find_clipping_level();
    clipping_signal();
    autocor_relation();
   }
  median_filter();
  for(nf=0;nf<(number_of_frame-2);nf++)
  {
   if(nf==0)
     Z1[nf] = 0;
   else
     Z1[nf] = median[nf] - median[nf-1];
  }
 for(nf=0;nf<(number_of_frame-2);nf++)
  {
   if(Z1[nf]==0) { Z[nf] =  0;}
   if(Z1[nf] >0) { Z[nf] =  1;}
   if(Z1[nf] <0) { Z[nf] = -1;}
  }
 /*  delete 2 value due to meadian filtering  */
 number_of_frame = (number_of_frame-2);
 fwrite(&number_of_frame,2,1,parameter_file);
 fwrite(&flag,2,1,parameter_file);
 fwrite(&Z,2,number_of_frame,parameter_file);
 close_all_pitch_file();
}
/*****************************************************************************/
void load_unknown_word_file(void)
{
 int temp;
  /* open unknown word file */
  if ((unknown_word_file = fopen(unknown_word_file_name,"rb")) == NULL)
```

```
  {
    printf("\n\7 □ Cannot open unknown_word_file !\n");
    exit(-1);
  };
 fread(&T,2,1,unknown_word_file);
 fread(&temp,2,1,unknown_word_file);  /* for bypass FLAG (0) */
  if (temp != 0)
  {
    printf("\n\7 □ Invalid unknown word file !\n");
    exit(-1);
  };
 O = (int*)calloc(T,sizeof(int));
  if (O == NULL)
  {
    printf("\n\7 □ Cannot allocate memory for data !\n");
    exit(-1);
  };
  /* read observation data */
 fread(O,2,T,unknown_word_file);
}
/*****************************************************************************/
void allocate_memory(void)
{
  dt   = (double*)calloc(T*N,sizeof(double));
  ar   = (double*)calloc(T*N,sizeof(double));
  path =  (double*)calloc(T  ,sizeof(double));
  if ( (dt||ar||path) == NULL)
  {
    printf("\n\7 □ Cannot allocate memory for data !\n");
    exit(-1);
  }
  else
```

```
  { // printf(" allocated memory complete\n");  }
}
/*****************************************************************************/
void load_model(int model_name)
{
 /* open model " aprime " file */
if ((model_a_file = fopen(model_a_file_name[model_name],"rb")) == NULL)
{
  printf("\n\7 □ Cannot open model '%s' file !\n",model_a_file_name[model_name]);
  exit(-1);
}
else
{ // printf(" Open model '%s' file  complete\n\n",model_a_file_name[model_name]); }
for(i=0;i<N;i++)
{
  for(j=0;j<N;j++)
  {
   fread(&a,4,N*N,model_a_file);
  }
}
 /*  open model " bprime " file */
if ((model_b_file = fopen(model_b_file_name[model_name],"rb")) == NULL)
{
  printf("\n\7 □ Cannot open model '%s' file !\n",model_b_file_name[model_name]);
  exit(-1);
}
else
{ // printf(" Open model '%s' file  complete\n\n",model_b_file_name[model_name]); }
for(i=0;i<N;i++)
{
  for(k=0;k<K;k++)
  {    fread(&b,4,N*K,model_b_file);   }
```

```
 }
 fclose(model_a_file);
 fclose(model_b_file);
}
/*****************************************************************************/
void viterbi(void)
{
 /*------------------------ find initial values ------------------------*/
 /*--- dts have a row and 6 columns, ars have a row and 6 columns too ---*/
 for (i=0;i<N;i++)
  {
    dt[0*N+i] = (double)(Pi[i] * b[i][O[0]+1]);
    ar[0*N+i] = 0;
  }
 /*---------------------- find recursion values ----------------------*/
 for (t=1;t<T;t++)  /* t is lessthan real_value 1 value */
  {
   for (j=0;j<N;j++)
    {
          for (i=0;i<N;i++)
           {
            d[i] = dt[(t-1)*N+i] * (double)a[i][j];
     }
          dmax     = 0;     /* double */
          max_index = 0;     /* int */
          for (k=0;k<N;k++)
           {
            if (dmax < d[k])
            {
                  dmax = d[k];
                  max_index = k;
              }
```

```
            }
            dt[t*N+j] = dmax * (double)b[j][O[t]+1];    /* double */
            ar[t*N+j] = max_index;                      /* double */
        }
    }
    /*------------------------ find terminal values ------------------------*/
    dmax     = 0;
    max_index = 0;
    for (k=0;k<N;k++)
    {
      if (dmax < dt[(T-1)*N+k])
      {
                dmax = dt[(T-1)*N+k];
                max_index = k;
      }
    }
    prob[model] = dmax;
    path[T-1]   = max_index;  /* long */
    printf("                    prob[%d]  =  %7.5e    \n",(model+1),prob[model]);c++;
    /*------------------------ path backtracking ------------------------*/
    for (t=T-2;t>=0;t--)
    {
      path[t] = ar[(t+1)*N+(path[t+1])];
    }
}
/*****************************************************************************/
void display_recognized_word(void)
{
    dmax     = 0;
    max_index = 0;
    for (k=0;k<STORED_MODEL;k++)
    {
```

```
    if (dmax < prob[k])
     {
          dmax = prob[k];
          max_index = k;
   }
   printf("      ==========================================\n");
   printf("      □ Recognized Tone is \" %s \" □ \n",name[max_index]);
   printf("      ==========================================\n");
   printf("\n\n              ...END...  \n");
}
/*************************************************************************/
void recognize(void)
{
  load_unknown_word_file();
  allocate_memory();
  for (model=0;model<STORED_MODEL;model++)
  {
    load_model(model);
    viterbi();
  }
  display_recognized_word();
}
/*************************************************************************/
void main(void)
{
   pitch();
   recognize();
}
/*************************************************************************/
```

ภาคผนวก ค
ผลงานวิจัยที่ได้รับการตีพิมพ์

ผลงานวิจัยเรื่อง "แบบจำลองเสียงวรรณยุกต์สำหรับภาษาไทย โดยใช้เทคนิคการควอนไตซ์พิทช์และ Hidden Markov Modeling" ได้นำเสนอและตีพิมพ์ใน งานประชุมวิชาการทางวิทยาการคอมพิวเตอร์และวิศวกรรมคอมพิวเตอร์แห่งชาติ 2541 ซึ่งจัดประชุมโดย คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์

แบบจำลองเสียงวรรณยุกต์สำหรับภาษาไทย โดยใช้เทคนิคการควอนไตซ์พิทช์และ Hidden Markov Modeling

Tone Recognition Model for Thai Language Using Pitch Quantization and Hidden Markov Modeling Techniques.

จิตรลดา จารุมิศร์ ไกรสิน ส่งวัฒนา อิทธิชัย อรุณศรีแสงไชย

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง
3-2 ถ.ฉลองกรุง ลาดกระบัง กรุงเทพ 10520
โทร. (662) 739-0961 E-Mail: s8061246@kmitl.ac.th

บทคัดย่อ

บทความนี้เสนอการสร้างแบบจำลองระดับเสียงวรรณยุกต์ 5 ระดับสำหรับภาษาไทย โดยขั้นแรก เสียงพูดจะถูกแบ่งให้เป็นส่วนย่อยๆ แต่ละส่วนจะถูกนำมาคำนวณหาคาบเวลาพิทช์โดยใช้วิธีออโตโคริเลชั่น จากนั้นทำการควอนไตซ์การเบี่ยงเบนของความถี่มูลฐานของพิทช์นั้นๆเพื่อเป็นข้อมูลฝึกสอนสำหรับการทำ Hidden Markov Modeling (HMM) ซึ่งจะทำให้ได้แบบจำลองเสียงวรรณยุกต์ออกมา จากนั้นทำการทดลองตรวจสอบประสิทธิภาพการรู้จำระดับเสียงวรรณยุกต์ของแบบจำลองดังกล่าว โดยการสุ่มป้อนข้อมูลเสียงที่ได้จากเพศชาย 5 คน และเพศหญิง 5 คน ผลปรากฏว่ามีความแม่นยำเฉลี่ยมากกว่า 90 เปอร์เซนต์

Abstract

This paper presents 5 tones level recognition Modeling for Thai Language. First, the speech is divided into small frames, and the autocorrelation is calculated for each frame of speech to determine the pitch period and it fundamental frequency. Quantization technique was applied to convert the difference of fundamental frequency sequence into a data training sequence for Hidden Markov Modeling (HMM). A Model for 5 tones was generated and random of tonal recognition tests were then conducted from the speech database of 5 male and 5 female to evaluate the effects of such model. The testing results shown the average accuracy of recognition above 90 percent.

1.บทนำ

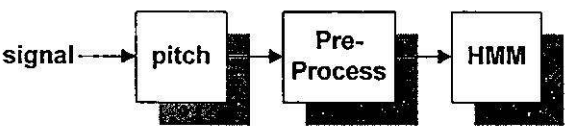
จากความต้องการให้เครื่องคอมพิวเตอร์ สามารถเรียนรู้จำคำพูดได้ จึงทำให้เกิดศาสตร์แขนงหนึ่งเรียกว่า Speech Recognition โดยใช้การสร้างแบบจำลองของคำเพื่อสอนให้ระบบรู้จำคำพูดเหล่านั้นเสียก่อน แต่การรู้จำคำพูดในภาษาไทยยังมีการพัฒนาค่อนข้างล่าช้าเพราะเป็นภาษาที่มีความยุ่งยากซับซ้อน เนื่องจากเป็นภาษาที่มีระดับเสียงหลายระดับ (Tonal language) โดยมีเสียงของวรรณยุกต์เป็นตัวบังคับระดับเสียงของคำ หรือ พยางค์ ซึ่งระดับเสียงที่แตกต่างกันนี้จะมีผลต่อความหมายของคำนั้นๆด้วย ดังนั้นในการสร้างระบบการรู้จำคำพูดจำเป็นจะต้องมีส่วนของการรู้จำระดับเสียงด้วย เพื่อทำให้การรู้จำนั้นมีความเที่ยงตรงและแม่นยำสูงขึ้น เช่นเดียวกับการศึกษาการรู้จำระดับเสียงของภาษาจีน [1]

ระดับเสียงเกิดจากการสั่นหรือการปิด-เปิดของเส้นสายเสียงในหลอดเสียง องค์ประกอบที่มีความสัมพันธ์กับอัตราการสั่นของเส้นสายเสียงก็คือพิทช์ (pitch) หรือความถี่มูลฐาน(F_0) โดยจะพบว่าถ้า F_0 มีค่าคงที่เราก็จะได้ยินเสียงพูดที่มีระดับเดียว แต่ถ้า F_0 มีค่าเพิ่มขึ้นเนื่องจากมีการสั่นของเส้นสายเสียงเร็วขึ้นเสียงที่ได้ยินก็จะเป็นเสียงสูง ดังนั้นจากความสำคัญของระดับเสียงนี้จึงได้มีผู้ทำการศึกษาการรู้จำเสียงวรรณยุกต์ในโดเมนของความถี่โดยใช้ความถี่ฮาร์โมนิกส์ และใช้การแปลงสัญญาณข้อมูลทางเวลาร่วมกับการแปลง Fast Fourier Transform ให้ได้ข้อมูลที่อยู่ในรูปฮาร์โมนิกส์ของความถี่ [2]

บทความวิจัยนี้ได้เสนอแนวทางสร้างแบบจำลองเสียงวรรณยุกต์สำหรับภาษาไทย โดยทำการคำนวณหาคาบของสัญญาณเสียงที่อยู่ในรูปของค่าพิทช์ด้วยวิธีออโตโครีเลชั่น[3] และนำค่า F_0 ที่ได้มาผ่าน median filtering เพื่อลดการผิดพลาดอันเนื่องมาจากการปัดเศษของการคำนวณดังกล่าว จากนั้นทำการหาค่าการเปลี่ยนแปลงของความถี่ F_0 เทียบกับความต่างเวลาเป็นค่า ΔF โดยกำหนดให้การเปลี่ยนแปลงค่า ΔF มี 3 ระดับตามทิศทางการเพิ่มขึ้นหรือลดลงของความถี่ F_0 และค่าการเปลี่ยนแปลงนี้จะถูกนำไปใช้เป็นส่วนข้อมูล training ของ HMM เพื่อสร้างแบบจำลองการรู้จำของหน่วยเสียงวรรณยุกต์ทั้ง 5 เสียง ส่วนที่ 2 ในบทความนี้จะกล่าวถึงขั้นตอนการวิเคราะห์เสียงวรรณยุกต์ ส่วนการทดลองและผลการทดลองจะกล่าวในส่วนที่ 3 และบทสรุปจะกล่าวในส่วนที่ 4

2. ขั้นตอนการวิเคราะห์เสียงวรรณยุกต์

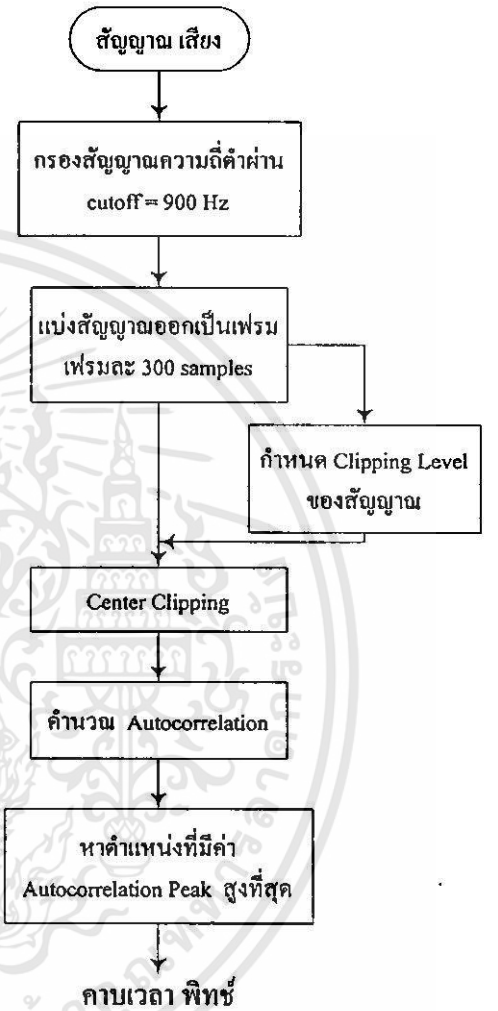
ในการวิเคราะห์และสร้างแบบจำลองเสียงวรรณยุกต์นั้น แบ่งขั้นตอนออกเป็น 3 ขั้นตอนดังแสดงในรูปที่ 1



รูปที่ 1 แสดงขั้นตอนในการวิเคราะห์

2.1 การหาค่าพิทช์

ทำการหาค่าคาบเวลาพิทช์โดยใช้วิธี Modified Autocorrelation Method using Clipping (AUTOC) [4-5] ซึ่งมีขั้นตอนในการวิเคราะห์ดังแสดงในรูปที่ 2



รูปที่ 2 ขั้นตอนในการวิเคราะห์เสียงวรรณยุกต์

เริ่มจากนำสัญญาณเสียงที่ได้จากการ sampling ที่ความถี่ 11.025 KHz มาผ่าน lowpass filter ที่มีความถี่ cutoff 900 Hz เพื่อกำจัดความถี่ฮาร์โมนิกส์ที่ไม่ต้องการออกไป จากนั้นทำการแบ่งสัญญาณออกเป็นช่วงๆหรือเฟรม (frame) เพื่อที่จะคำนวณหาพารามิเตอร์ที่อยู่ในช่วงเวลานั้นๆออกมา โดยในแต่ละเฟรมกำหนดให้มีตัวอย่างสัญญาณ 300 samples การวิเคราะห์จะทำการวิเคราะห์ทีละเฟรม โดยกำหนดให้มีช่วงของการเลื่อนเฟรมครั้งละ

100 samples นั่นคือแต่ละเฟรมจะมีช่วงของการซ้อนทับกัน 2 ใน 3 เฟรม

จากนั้นเพื่อเป็นการกำจัดสัญญาณที่มีขนาดแอมปลิจูดต่ำออกไป จึงทำการ clip สัญญาณที่อยู่ในช่วง 65 เปอร์เซนต์ของ Absolute Amplitude Peak [5] ซึ่งจะสามารถคำนวณหา Cliping Level ได้จากสมการต่อไปนี้

$$C_L = (65\%) \times \min(K_1, K_2) \qquad (1)$$

โดยที่ C_L = Cliping Level

K_1 = Absolute Amplitude Peak ของ 100 samples แรก ของเฟรม

K_2 = Absolute Amplitude Peak ของ 100 samples ท้าย ของเฟรม

เมื่อได้ Cliping Level ของสัญญาณแล้ว สัญญาณที่มีค่าอยู่ในช่วง $\pm C_L$ จะถูกกำหนดให้มีค่าเป็นไปตามความสัมพันธ์ดังต่อไปนี้

$$y(n) = \mathrm{sgn}[x(n)] = \begin{cases} 1 & , x(n) \geq C_L \\ 0 & , |x(n)| < C_L \\ -1 & , x(n) \leq -C_L \end{cases} \qquad (2)$$

โดยที่ sgn[x(n)] คือ สัญญาณที่ผ่านการ clip จากนั้นนำสัญญาณที่ผ่านการ clip คือค่า y(n) มาคำนวณออโตโครีเลชั่น ตามสมการต่อไปนี้

$$R(k) = \sum_{n=0}^{N-1-k} y(n) \times y(n+k) \qquad (3)$$

เมื่อ k = การเลื่อนไปของเวลา ในที่นี้ให้มีค่าเป็น 250

n = จำนวนข้อมูลในเฟรม (0,1,2,...,N)

ซึ่งจากคุณสมบัติของออโตโครีเลชั่นฟังก์ชั่น ถ้าสัญญาณมีความเป็นคาบที่ระยะ P จะได้ว่า R(k) ก็จะมีความเป็นคาบที่ระยะ P เช่นเดียวกัน โดยค่าที่มากที่สุดของ R(k) จะเกิดที่ตำแหน่ง $k = 0, \pm P, \pm 2P, \ldots$ จากนั้นทำการหาตำแหน่งที่มี Autocorrelation Peak สูงที่สุดเมื่อเทียบกับ R(0) ซึ่งระยะที่ได้ก็คือ คาบเวลาพิทช์นั่นเอง

จากค่าคาบเวลาพิทช์ที่ได้นี้สามารถนำมาหาค่าความถี่มูลฐาน F_o ได้จากความสัมพันธ์ คือ

$$F_o = \frac{F_s}{P} \qquad (4)$$

เมื่อ F_o = ความถี่มูลฐาน (Hz)

F_s = ความถี่ที่ใช้ในการสุ่มสัญญาณ ในบทความนี้ใช้ 11.025 KHz

P = คาบเวลา พิทช์

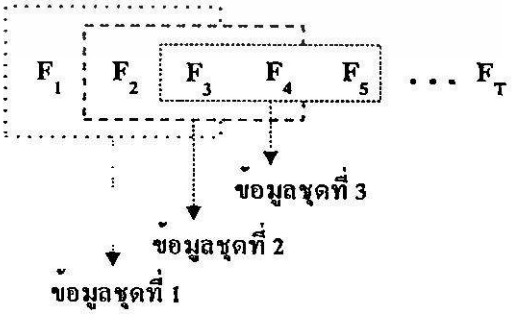
2.2 การ Pre-process ข้อมูล

ขั้นตอนนี้ประกอบด้วยขั้นตอนย่อย 3 ขั้นตอนคือ

2.2.1 median filtering

ค่า F_o ที่คำนวณได้จากการหาค่าพิทช์ในขั้นตอนแรกนั้นจะอยู่ในรูปของเลขจำนวนเต็ม ซึ่งอาจเกิดความไม่ต่อเนื่องของค่า F_o ขึ้นเนื่องจากการปัดเศษในการคำนวณ ด้วยเหตุนี้จึงได้นำ median filter [6] มาใช้เพื่อลดความไม่ต่อเนื่องที่เกิดขึ้นดังกล่าว

จากการคำนวณหาพิทช์ในขั้นตอนที่แล้วเราจะได้ค่าความถี่มูลฐาน 1 ค่า ต่อเฟรมเสียง 1 เฟรม จากนั้นทำการจัดเรียงค่าความถี่มูลฐานที่คำนวณได้ออกเป็นชุดข้อมูล โดยในแต่ละชุดข้อมูลประกอบด้วยค่าความถี่ 3 ค่า โดยการเลื่อนของชุดข้อมูล แสดงได้ดังรูปที่ 3



รูปที่ 3 การจัดแบ่งความถี่มูลฐานออกเป็นชุดข้อมูล

เมื่อ F_1 = ค่าความถี่มูลฐานของเฟรมที่ 1

F_2 = ค่าความถี่มูลฐานของเฟรมที่ 2

F_T = ค่าความถี่มูลฐานของเฟรมสุดท้าย

จากนั้นนำค่าความถี่ 3 ค่า ในแต่ละชุดข้อมูลมาจัดเรียงใหม่ตามความสัมพันธ์

$$a \leq b \leq c \qquad (5)$$

โดยที่

a = ความถี่ F_0 ที่มีค่าน้อยที่สุดของแต่ละชุดข้อมูล

b = ความถี่ F_0 ที่มีค่าอยู่ระหว่างกลาง

c = ความถี่ F_0 ที่มีค่ามากที่สุดของแต่ละชุดข้อมูล

จากนั้นนำความถี่ค่ากลาง (b) ที่ได้จากชุดข้อมูลแต่ละชุดมาจัดเรียงตามลำดับ ก็จะได้ความถี่มูลฐานชุดใหม่ที่ผ่านกระบวนการ median filter แล้ว

2.2.2 หาการเปลี่ยนแปลงเป็น ΔF ของความถี่ F_0 ต่อเวลาที่เพิ่มขึ้น

$$\Delta F_t = F_{t+1} - F_t \qquad (6)$$

เมื่อ $t = 1, 2, \ldots, (T-1)$; T = จำนวนเฟรม

F_t = ความถี่ F_0 ที่เวลา t

F_{t+1} = ความถี่ F_0 ที่เวลา t+1

2.2.3 ทำการ Quantized ΔF ให้ได้ V_t อยู่ใน 3 ระดับ ตามทิศทางการเปลี่ยนแปลงของความถี่ F_0 โดยกำหนดให้

$$V_t = \begin{cases} 1 & ; \ \Delta F_t > 0 \\ 0 & ; \ \Delta F_t = 0 \\ -1 & ; \ \Delta F_t < 0 \end{cases} \qquad (7)$$

จากนั้นค่า $V_t = \{-1, 0, 1\}$ จะถูกนำไปใช้เป็นข้อมูล Training เพื่อสร้างแบบจำลองของเสียงวรรณยุกต์ต่อไป

2.3 การสร้างแบบจำลองเสียงวรรณยุกต์โดยใช้เทคนิค HMM

ด้วยเทคนิค HMM นี้ [7] การสร้างแบบจำลองเสียงพูดสามารถทำได้โดยใช้จำนวนสเตทที่แตกต่างกันจำนวนหนึ่ง คำศัพท์แต่ละคำที่ใช้ในการรู้จำจะถูกสร้างใหม่โดยเปลี่ยนจากรูปของการเปลี่ยนแปลงความถี่ให้อยู่ในรูปของการจัดเรียงตัวของสเตท โดยมีการย้ายจากสเตทเริ่มต้นไปยังสเตทถัดไปตามการเลื่อนไปของ discrete time ด้วยเซตของความน่าจะเป็นที่เกี่ยวข้องกับสเตทนั้นๆจนกระทั่งได้ output ที่มีลักษณะเป็นการเรียงตัวกันของสเตทที่ใช้แทนรูปแบบของคำนั้นๆ

2.3.1 ส่วนประกอบของ HMM

1. N คือ จำนวนสเตทในแบบจำลอง

ถ้าเราให้เซ็ตของสเตทเป็น { 1 , 2 ,..., N } ในบทความวิจัยนี้กำหนดให้ N = 5 ตามจำนวนระดับเสียงของวรรณยุกต์ และแทนสเตทที่เปลี่ยนไปตามเวลา t ด้วยเซ็ตของ

$$Q = \{q_1, q_2, \ldots, q_N\}$$

2. M คือจำนวนของเหตุการณ์ที่สามารถเป็นไปได้ใน 1 สเตท แทนสัญลักษณ์ ด้วย

$$V = \{v_1, v_2, \ldots v_M\}$$

ซึ่งจากการจัดระดับของ ΔF_t ออกเป็น 3 ระดับจะได้เซ็ตของเหตุการณ์ที่สามารถเป็นไปได้ในแต่ละสเตทมีค่าเป็น

$$V = \{-1, 0, 1\}$$

3. ค่าความน่าจะเป็นในการย้ายสเตท ; $A = \{a_{ij}\}$

$$a_{ij} = P[q_t = j \mid q_{t-1} = i]$$

เมื่อ $1 \leq i, j \leq N$

4. ค่าความน่าจะเป็นของเหตุการณ์ที่สามารถเป็นไปได้ภายในสเตทแทนด้วยเมตริกซ์ B เมื่อ $B = \{b_j(k)\}$

โดยที่ $b_j(k) = P[\, v_k$ ที่เวลา t $\mid \, q_j$ ที่เวลา t $]$

เมื่อ $1 \leq j \leq N$ และ $1 \leq k \leq M$

5. ความน่าจะเป็นของการเป็นสเตทเริ่มต้น ; $\pi = \{\pi_i\}$

$\pi_i = P[\, q_i$ ที่เวลา $t = 1\,]$; $1 \leq i \leq N$

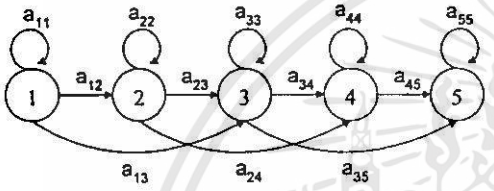
จะเห็นว่า Hidden Markov Model ต้องการพารามิเตอร์ของแบบจำลองคือ N , M และ กลุ่มของความน่าจะ

เป็น A, B, π ดังนั้นในการแสดงเซ็ตของพารามิเตอร์ที่สมบูรณ์ของแบบจำลองจะแทนด้วยสัญลักษณ์

$$\lambda = (A, B, \pi)$$

2.3.2 แบบจำลอง HMM

ในบทความนี้เลือกใช้ HMM แบบ Left-Right Model ที่ประกอบไปด้วยสเตททั้งหมด 5 สเตท เนื่องจากแบบจำลองนี้เหมาะกับสัญญาณที่มีลักษณะการเปลี่ยนแปลงตามเวลาอย่างต่อเนื่อง เช่น เสียงพูด โดยมีรูปแบบในการย้ายสเตทที่สามารถเป็นไปได้ดังรูปที่ 4



รูปที่ 4 Left-Right Model 5 state

ซึ่งแบบจำลองนี้มีคุณสมบัติในการย้ายสเตท ดังนี้

1. ไม่มีการย้ายสเตท ไปยังสเตทที่ต่ำกว่า

$$a_{ij} = 0 \quad ; \quad j < i$$

2. จะย้ายสเตทไปยังสเตทที่อยู่สูงกว่าได้ไม่เกิน Δi สเตท

$$a_{ij} = 0 \quad ; \quad i > i + \Delta i$$

ดังรูป $\Delta i = 2$ คือจะไม่มีการย้ายข้ามสเตทเกิน2 สเตท

3. มีค่าความน่าจะเป็นของสเตทเริ่มต้น คือ

$$\pi_i = \begin{cases} 1 & ; \ i = 1 \\ 0 & ; \ i \neq 1 \end{cases}$$

4. ที่ทุกๆความน่าจะเป็น จะมีค่าอยู่ในช่วง 0 ถึง 1 และผลรวมของความน่าจะเป็นของการย้ายสเตทจากสเตทใดๆจะต้องมีค่าเท่ากับ 1 เสมอ

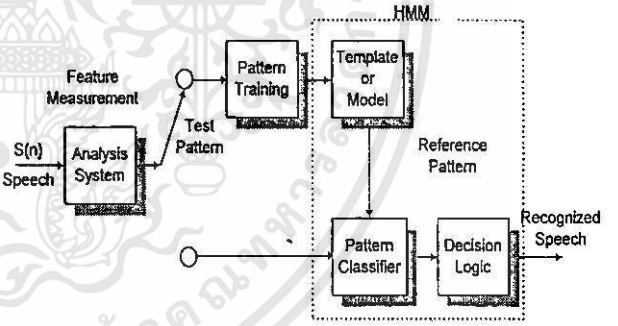
$$0 \leq a_{ij} \leq 1 \quad ; \quad \forall i, j$$

$$\sum_{j=1}^{N} a_{ij} = 1 \quad ; \quad \forall i$$

ในการสร้างและทดสอบแบบจำลองแบ่งการพิจารณาออกเป็น 3 ขั้นตอน คือ

1. การสร้างแบบจำลอง โดยใช้วิธี Forward-Backward Procedure และ Baum-Welch โดยลำดับ V_t ที่ได้จากการวิเคราะห์เสียงจะถูกนำมาเข้ากระบวนการสร้างแบบจำลอง เราเรียกลำดับความถี่นี้ว่า "ลำดับ เทรนนิ่ง" โดยลำดับเทรนนิ่งนี้จะถูกนำมาทำการคำนวณหา $[\lambda = A, B, \pi]$ ที่เหมาะสมกับเสียงนั้นๆ
2. กำหนดลำดับสเตท เป็นส่วนที่พยายามระบุสเตทให้กับแต่ละลำดับเทรนนิ่งของคำโดยใช้Viterbi Algorithm [7] เพื่อให้แบบจำลองมีความสามารถในการจำลองเสียงที่พูดเข้าไปนั่นเอง
3. เป็นขั้นตอนในการนำเสียงที่ต้องการทดสอบ มาเทียบกับแบบจำลองของเสียงที่มีอยู่ทั้งหมด ซึ่งเสียงที่นำมาทดสอบต้องผ่านการวิเคราะห์ให้อยู่ในรูปของความถี่ที่ถูกควอนไตซ์(V_t)เสียก่อน แล้วจึงนำมาเทียบกับแบบจำลองเพื่อดูความน่าจะเป็นว่าเสียงที่นำมาทดสอบเป็นเสียงใด

รูปที่5 แสดงส่วนของ HMMในระบบการรู้จำเสียงพูด

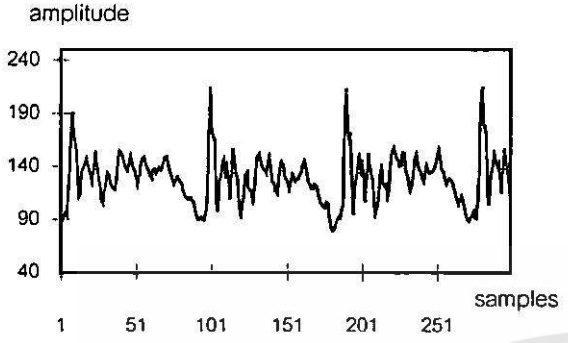


รูปที่ 5 ระบบการรู้จำเสียงพูด

3. ผลการทดลอง

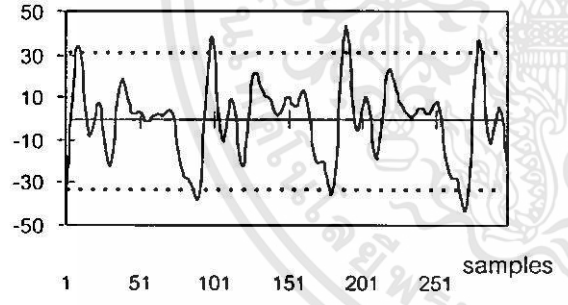
ทำการวิเคราะห์เสียงคำพยางค์เดียวจำนวน 25 คำ จากคำว่า อา อี อู เอ และ โอ ทั้ง 5 ระดับวรรณยุกต์ของเพศหญิง 5 คนและเพศชาย 5 คน แล้วนำลักษณะของการเปลี่ยนแปลงความถี่ของวรรณยุกต์ทั้ง 5 เสียงมาสร้างแบบจำลอง รูปที่ 6 แสดงลักษณะสัญญาณเสียงที่ได้จากการ sampling เสียง "อา" ของผู้ชาย โดยในบทความนี้ใช้

ความถี่ Sampling ที่ 11.025 KHz จะเห็นว่าสัญญาณที่ได้จากการ sampling จะประกอบด้วยความถี่จำนวนมาก

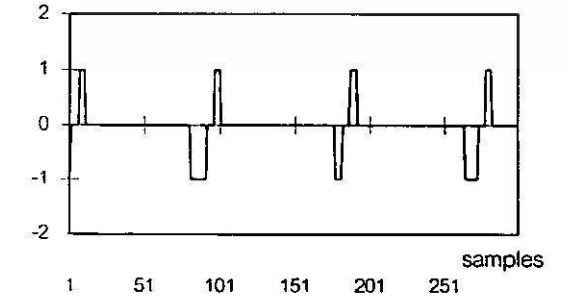


รูปที่ 6 สัญญาณที่ได้จากการ sampling

เมื่อนำสัญญาณมาผ่าน Low-pass filter จะได้สัญญาณที่มีลักษณะเรียบขึ้นดังรูปที่ 7 โดยเส้นประในรูปแสดง Cliping Level ของสัญญาณ ซึ่งในบทความวิจัยนี้ใช้ Cliping Level ของสัญญาณเป็น 65%ของ Absolute Amplitude Peak ในแต่ละเฟรมดังกล่าวมาแล้วข้างต้น สัญญาณที่ได้จากการ clip จะเป็นดังรูปที่ 8

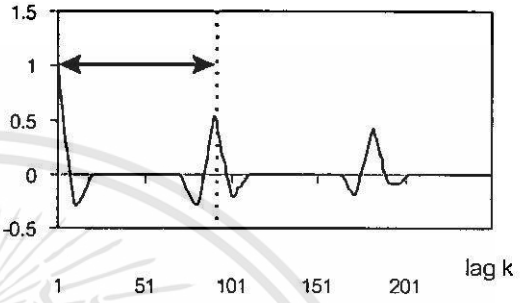


รูปที่ 7 สัญญาณที่ผ่าน lowpass filter



รูปที่ 8 สัญญาณที่ผ่านการ clip แล้ว

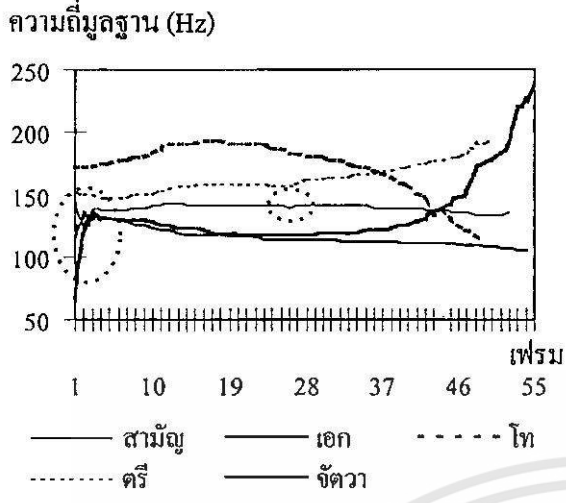
จากนั้นนำสัญญาณที่ผ่านการclipไปทำการคำนวน Autocorrelation Function ตามสมการที่ (3) จะได้สัญญาณที่มีลักษณะดังรูปที่ 9 ระยะห่างระหว่าง R(0) กับจุดยอดที่สูงที่สุดถัดไปก็คือคาบพิทช์ ซึ่งสามารถนำมาหาค่าความถี่มูลฐานได้ตามสมการที่ (4) จากรูปได้ค่าความถี่มูลฐานเท่ากับ 121 Hz (11025/91 = 121)



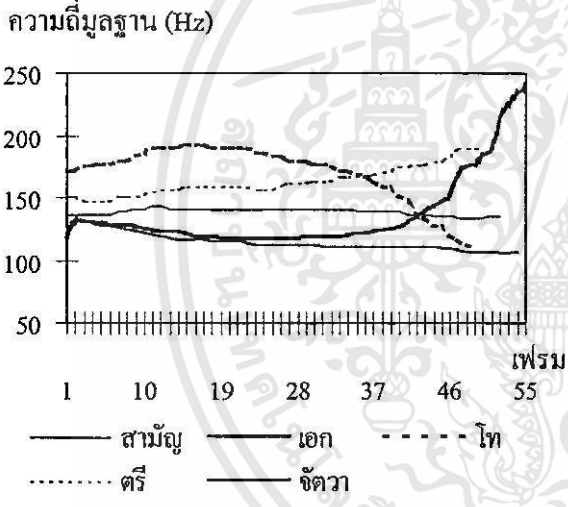
รูปที่ 9 ค่าที่ผ่านการคำนวณ Normalized Autocorrelaion Function

การหาลักษณะของการเปลี่ยนแปลงความถี่มูลฐานเทียบกับเวลาของวรรณยุกต์ทั้ง 5 เสียง จากคำทั้งหมด 25 คำ จะได้ว่าการเปลี่ยนแปลงของความถี่จะมีลักษณะเป็นรูปแบบเดียวกันในวรรณยุกต์แต่ละเสียง รูปที่10 แสดงเส้นกราฟการเปลี่ยนแปลงความถี่ F_o ของวรรณยุกต์ทั้ง 5 ระดับเสียง (สามัญ เอก โท ตรี จัตวา) ในผู้พูดเพศชาย รูปที่ 11 แสดงค่าความถี่ F_o เมื่อนำมาผ่าน median filter ซึ่งจะแสดงให้เห็นว่าภายในวงกลมเส้นประในรูปที่ 9 จะถูกปรับให้เรียบขึ้นเมื่อผ่านขั้นตอนนี้ และรูปที่ 12 แสดงตัวอย่างเสียงวรรณยุกต์ทั้ง 5 เสียงที่ผ่านการควอนไตซ์แล้ว

ทำการทดสอบผลโดยจัดกลุ่มทดสอบออกเป็น 5 กรณี โดยกรณีที่ 1 และ กรณีที่ 2 ใช้เสียงต้นแบบชาย 1 คน และหญิง 1 คนออกเสียง อา อี อู เอ โอ ทั้ง 5 ระดับซ้ำกัน 5 ครั้ง กรณีที่ 3 และ กรณีที่ 4 ใช้เสียงต้นแบบเป็นชาย 5 คนและหญิง 5 คน ออกเสียง อา อี อู เอ โอ ทั้ง 5 ระดับคนละ 1 ครั้ง และกรณีสุดท้ายใช้เสียงต้นแบบเป็นชาย 2 คน และหญิง 3 คนออกเสียงคนละ1 ครั้ง ผลการทดสอบการรู้จำเมื่อทำการทดสอบกับเสียงต้นแบบเดิม แสดงได้ดังตารางที่ 1



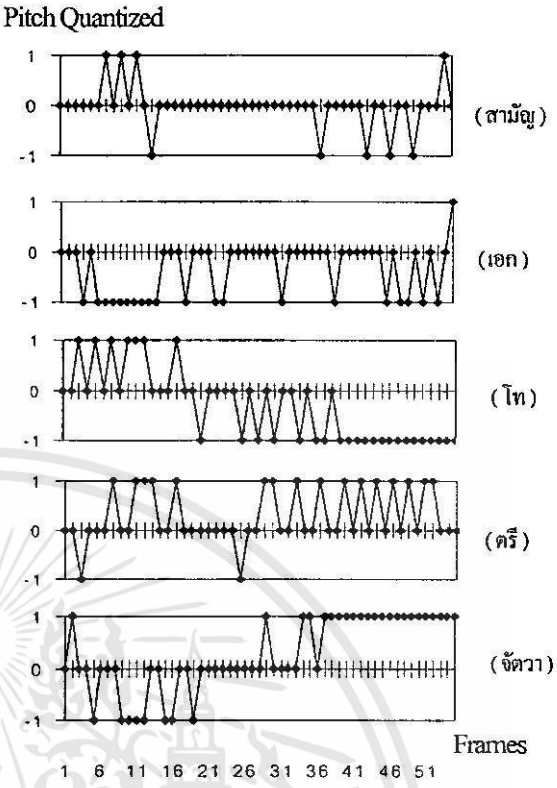
รูปที่ 10 ลักษณะของการเปลี่ยนแปลงความถี่มูลฐานในวรรณยุกต์ทั้ง 5 เสียง



รูปที่ 11 ความถี่มูลฐานที่ผ่าน median filtering

ตารางที่ 1 ผลการทดสอบแบบจำลองกับเสียงต้นแบบ

กรณี	แบบจำลอง	จำนวนเสียงต้นแบบ	จำนวนเสียงที่ถูกต้อง	ความถูกต้อง (%)
1.	ผู้หญิง 1 คน	125	118	94.40
2.	ผู้ชาย 1 คน	125	113	90.40
3.	ผู้หญิง 5 คน	125	120	96.00
4.	ผู้ชาย 5 คน	125	118	94.40
5.	ผู้ชาย 2 + ผู้หญิง 3 คน	125	120	96.00



รูปที่12 เสียงวรรณยุกต์ทั้ง 5 เสียง ที่ผ่านการควอนไตซ์

4. สรุป

จากการวิเคราะห์ลักษณะของเสียงวรรณยุกต์ พบว่าการเปลี่ยนแปลงความถี่มูลฐานมีลักษณะเฉพาะตัว ซึ่งสามารถนำมาสร้างแบบจำลองในการรู้จำเสียงวรรณยุกต์แต่ละเสียงได้ การนำ median filter มาใช้นอกจากช่วยลดความไม่ต่อเนื่องของความถี่แล้วยังช่วยปรับความเรียบของความถี่ในช่วงต้นเสียงด้วย และจากการสร้างแบบจำลองจากการจัดระดับของการเปลี่ยนแปลงความถี่(V_t) ทำให้แบบจำลองนี้สามารถใช้ได้ทั้งผู้พูดที่เป็นเพศชายและเพศหญิง

จากการทดสอบการรู้จำจะพบว่าแบบจำลองของกลุ่มทดสอบในทุกกรณีให้เปอร์เซนต์ความแม่นยำในการรู้จำมากกว่า 90 เปอร์เซนต์ โดยแบบจำลองที่สร้างจากผู้พูดหลายคนจะให้ผลการรู้จำแม่นยำกว่าแบบจำลองที่สร้างจากผู้พูดเพียงคนเดียวเนื่องจากมีความหลากหลายของข้อมูลต้นแบบมากกว่า และจะพบว่าแบบจำลองที่สร้างจากชาย 2 คนและหญิง 3 คนมีความแม่นยำในการรู้จำถึง 96% นอกจากนี้การจัดแบ่ง V_t ออกเป็น 3 ระดับยังเป็นการลด

เวลาที่ต้องใช้ในการคำนวณการสร้างแบบจำลองอีกด้วย ซึ่งจากผลการทดลองสร้างแบบจำลองที่ใช้ในการรู้จำระดับเสียงวรรณยุกต์จากจำนวนเสียงต้นแบบ 25 เสียงใช้เวลาเพียง 0.50 นาที โดยทำการประมวลผลบนเครื่องคอมพิวเตอร์PC รุ่นเพนเทียม 166

เอกสารอ้างอิง

[1] W.J. Yang, J.C. Lee, Y.C. Chang and H.C. Wang, "Hidden Markov Model for Mandarin Lexical Tone Recognition," IEEE Trans. Acoust., Speech, and Signal Processing, vol. 36, pp.988-992, July 1988.

[2] ธันวา ศรีประโมง "การวิเคราะห์เสียงพูดภาษาไทยในแกนความถี่ฮาร์โมนิค"วิทยานิพนธ์ปริญญามหาบัณฑิต คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง พ.ศ.2537

[3] อภิชาติ ตั้งทางธรรม "การเปลี่ยนความเร็วของเสียงพูด" การประชุมวิชาการทางวิศวกรรมไฟฟ้าครั้งที่ 17 พ.ศ. 2537

[4] L.R. Rabiner,"On the Use of Autocorrelation Analysis for Pitch Detection," IEEE Trans. Acoust., Speech, and Signal Processing, vol. ASSP-25, pp. 24- 33, Feb 1977

[5] L.R. Rabiner and R.W. Schafer,"Digital Process of Speech Signal," New jersey: Prentice Hall, 1978.

[6] A. Rosenfeld and A.C. Kak,"Digital picture Processing ," Orlando, Florida: Academic Press,.Inc, 1982.

[7] L.R. Rabiner and B.H. Juang,"Fundamental of Speech Recognition," New jersey: Prentice Hall, 1993.

ประวัติผู้เขียน

นางสาว จิตรลดา จารุมิศร์ เกิดเมื่อวันที่ 17 กุมภาพันธ์ พ.ศ. 2515 ที่จังหวัดนครสวรรค์ สำเร็จการศึกษาระดับปริญญาตรี วิทยาศาสตรบัณฑิต สาขาวิชาวัสดุศาสตร์ มหาวิทยาลัยเชียงใหม่ เมื่อปี พ.ศ. 2537