

cantometria

A rust library & TUI Application to grade human singing accuracy.

ที่มาและความสำคัญ

- โครงการนี้ออกแบบระบบให้คะแนนการร้องเพลงโดยพิจารณาทฤษฎีดนตรีและความถี่เสียง เพื่อให้การให้คะแนนยุติธรรมทั้งสำหรับนักร้องมืออาชีพและผู้ที่ไม่ได้ฝึกฝน
- ระบบถูกพัฒนาเพื่อแก้ไขข้อจำกัดของระบบเดิมที่เข้มงวดเกินไป และให้ผลลัพธ์ที่สอดคล้องกับการรับฟังของมนุษย์มากขึ้น

จุดประสงค์

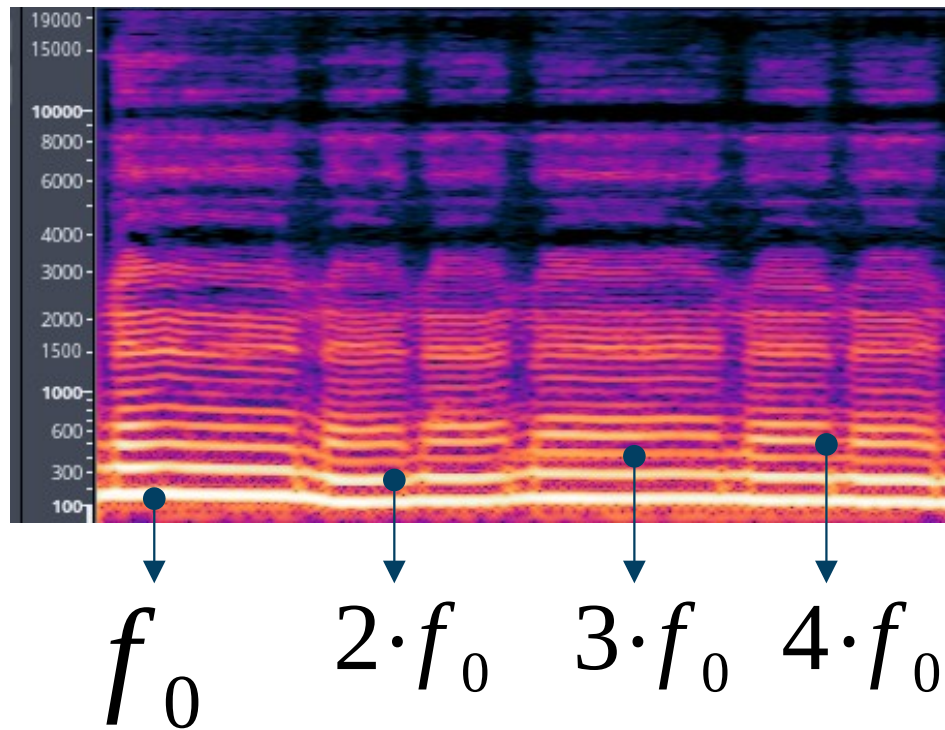
- พัฒนาอัลกอริธึมใหม่สำหรับการให้คะแนนการร้องเพลงของมนุษย์ โดยให้ความเป็นธรรม รองรับข้อผิดพลาดของมนุษย์ และสะท้อนความสุนทรีของดนตรี

ขอบเขต

- ประมวลผลเฉพาะเสียงร้องเพลง
ของมนุษย์
(**ไม่รวมเสียงเครื่องดนตรี**)
- รongรับเสียงสูง (หญิง) และเสียง
ต่ำ (ชาย) รวมถึงเสียงฮัม
- ใช้อัลกอริธึม เช่น FFT, YIN ปรับ
ใช้กับทฤษฎีดนตรี
- ระบบเริ่มต้นที่ความแม่นยำสูง
และปรับค่าความยืดหยุ่นได้
- ต้องเริ่มร้องตรงจังหวะแรกของ
ทำนองอ้างอิง
- โค้ดเปิดเผยเพื่อให้สามารถปรับ
แต่งค่าความยุติธรรมได้

แนวทางและทฤษฎีต่าง ๆ ที่ใช้

- **ความถี่มูลฐาน**
(Fundamental Frequency, f_0)
คือความถี่ต่ำสุดของเสียง มีค่าส่วน
กลับเป็นคาบมูลฐาน (T)
- **ฮาร์โมนิกส์ (Harmonics)**
เป็นพหุคูณของ f_0 ที่เกิดจากการสั่น
สะท้อนของเสียง
- **เสียงมนุษย์**
ประกอบด้วย f_0 และโอเวอร์โทน ซึ่ง
สามารถมองเห็นได้ในสเปกโตรแกรม



แนวทางและทฤษฎีต่าง ๆ ที่ใช้

- **ขั้นตอนวิธี YIN** ใช้ประมาณ f_0 ของเสียงพูด ลดผลกระทบจากฮาร์โมนิกส์และเสียงรบกวน

1) คำนวณฟังก์ชันความแตกต่างแทนอัตราสหสัมพันธ์

$$d(\tau) = \sum_{t=1}^{N-\tau} (x(t) - x(t+\tau))^2$$

2) คำนวณฟังก์ชันผลต่างปกติเฉลี่ย

$$\tilde{d}(\tau) = \frac{d(\tau)}{\frac{1}{\tau} \sum_{j=1}^{\tau} d(j)}$$

3) หา τ_0 ที่ให้ค่าต่ำสุดและใช้การสอดคล้องแรกกำลังสอง

4) คำนวณ f_0 จาก

$$f_0 = \frac{f_s}{\tau_0}$$

แนวทางและทฤษฎีต่าง ๆ ที่ใช้

- **ขั้นคู่เสียง:** ความแตกต่างของระดับเสียงระหว่างสองเสียง เช่น
 - คู่แปดสมบูรณ์ (2:1), คู่ห้าสมบูรณ์ (3:2), คู่สี่สมบูรณ์ (4:3)
 - วัดความแตกต่างเป็น **เซนต์ (cent)** โดย 1 ครึ่งเสียง = 100 เซนต์
 - หมายเลขโน้ต 60 = **โดกลาง (Middle C)**
- **การรับรู้เสียง:** หูมนุษย์รับรู้ระดับเสียงแบบ **ลอการิทึม**
 - ความถี่ของโน้ตคำนวณโดย:
$$n = 69 + 12 \cdot \log_2 \frac{f}{440}$$
 - โน้ตที่เป็นพหุคูณง่ายของความถี่มูลฐาน (เช่น คู่ห้า) ให้เสียงประสานกลมกลืน

แนวทางและทฤษฎีต่าง ๆ ที่ใช้

- **MIDI:** เป็นมาตรฐานสำหรับการสื่อสารดนตรี ไม่ได้เก็บเสียง แต่เป็นคำสั่งให้เล่นโน้ตต่าง ๆ
- ขนาดไฟล์เล็กและง่ายต่อการส่งต่อ
- เวลาใน MIDI ถูกเก็บเป็น **ติก (Tick)** และสามารถแปลงเป็นวินาทีได้
- **WAV:** เป็นไฟล์เสียงมาตรฐานที่เก็บเสียงแบบไม่บีบอัด
- ข้อมูลเสียงเก็บเป็นบิตสตรีมและไล่ลำดับตัวอย่างเสียงตามช่องเสียง (ซ้าย-ขวา)
- ขนาดของกลุ่มตัวอย่างสามารถคำนวณได้จากอัตราการชักตัวอย่าง

แนวทางและทฤษฎีต่าง ๆ ที่ใช้

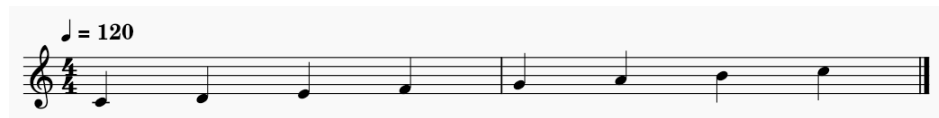
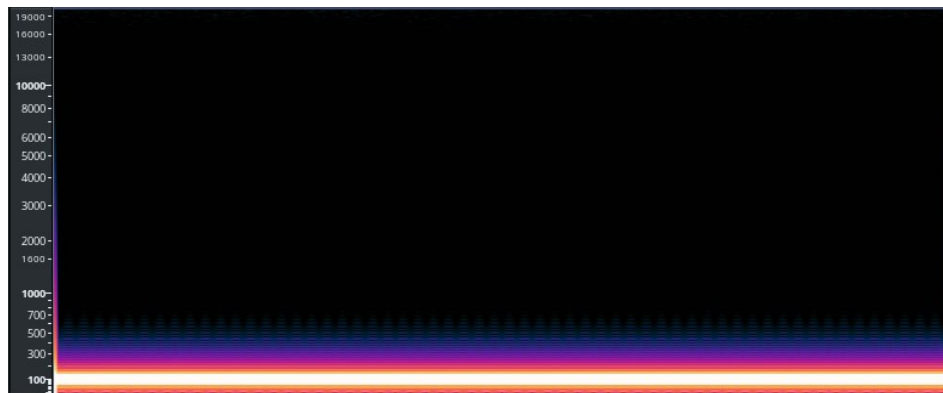
- **การถือค่าอันดับที่ศูนย์ (Zero-order Hold):** ใช้ในการแปลงสัญญาณดิจิทัลเป็นอนาล็อก โดยถือค่าตัวอย่างเดิมจนกว่าตัวอย่างใหม่จะมา
- **การเติมศูนย์ (Zero Padding):** เติมค่า 0 ในสัญญาณเวลาเพื่อให้ FFT มีประสิทธิภาพมากขึ้น
- ใช้เพื่อให้สัญญาณสองตัวมีขนาดเท่ากันเพื่อเปรียบเทียบได้ง่าย
- **FFT:** คำนวณ **DFT** แต่ลดความซับซ้อนจาก $O(n^2)$ เป็น $O(n \log n)$
- **สหสัมพันธ์ไขว้ (Cross-correlation):** ใช้เพื่อคำนวณ**เวลาเลื่อนที่เหมาะสม**
- **ค่าเฉลี่ยเลขคณิตของผลต่างเชิงคู่:** ใช้เพื่อการ**จัดตำแหน่งช่วงกว้าง**ของสัญญาณ

วิธีการแก้ปัญหา

- 1) เปิดไฟล์ MIDI และบันทึกแต่ละคำสั่งพร้อมเวลาปัจจุบัน
- 2) เปิดไฟล์ WAV และคำนวณและบันทึก f_0 ในแต่ละกรอบเวลาย่อย
- 3) นำสัญญาณจากข้อ 1) มาหาค่าอันดับศูนย์
- 4) เติมศูนย์ให้สัญญาณที่สั้นกว่าระหว่าง 2) และ 3) ให้ยาวเท่ากัน
- 5) เลื่อนสัญญาณ 2) ให้ตรงกับ 3) ทางเวลามากที่สุด
- 6) เลื่อนสัญญาณ 2) ให้ตรงกับ 3) ทางช่วงกว้างมากที่สุด
- 7) คำนวณคะแนนความแม่นยำ มี 4 มิติได้แก่ ความตรงต่อเวลา กุญแจเสียง ระดับเสียง (ใช้แนวคิดขั้นคู่เสียงเพื่อความยุติธรรม) และความครอบคลุมทำนอง
- 8) คำนวณความแม่นยำสุดท้าย
- 9) แสดงผลความแม่นยำแต่ละมิติแก่ผู้ใช้

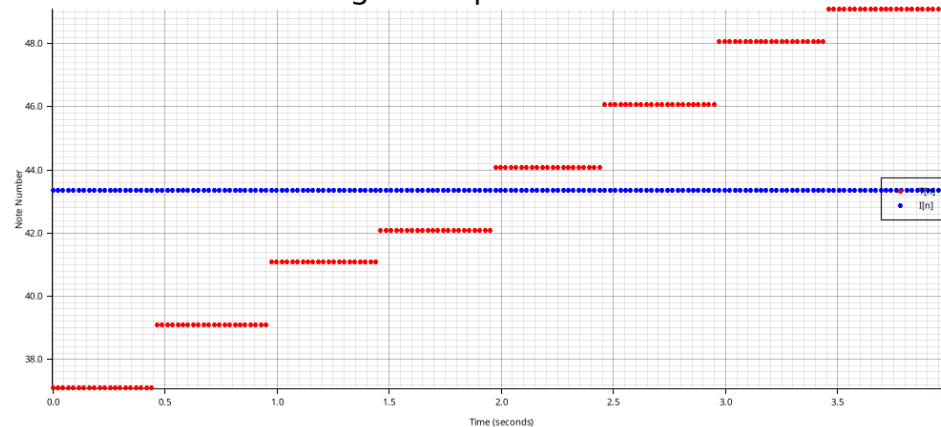
ผลการทดลอง

- การร้องทำนองบันไดกฤษแจเสียงโดเมเจอร์
ด้วยคลื่นไซน์สังเคราะห์ (การทดลองควบคุม)



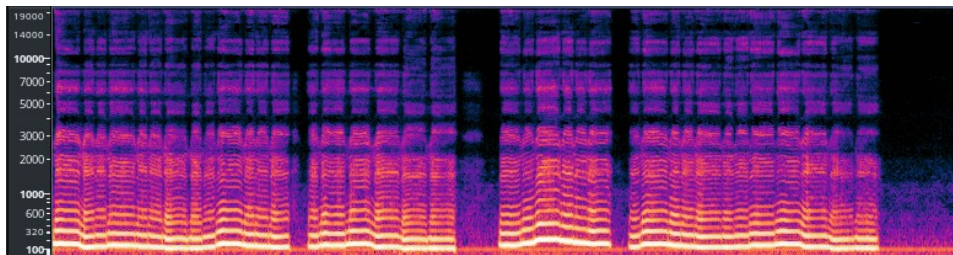
- ค่าความแม่นยำสุทธิ 16.87
- ครอบคลุม: 99.81 เวลา: 100
ระดับเสียง: 8.32 ภัยเงียบเสียง: 42.67

Target vs Input Melodies



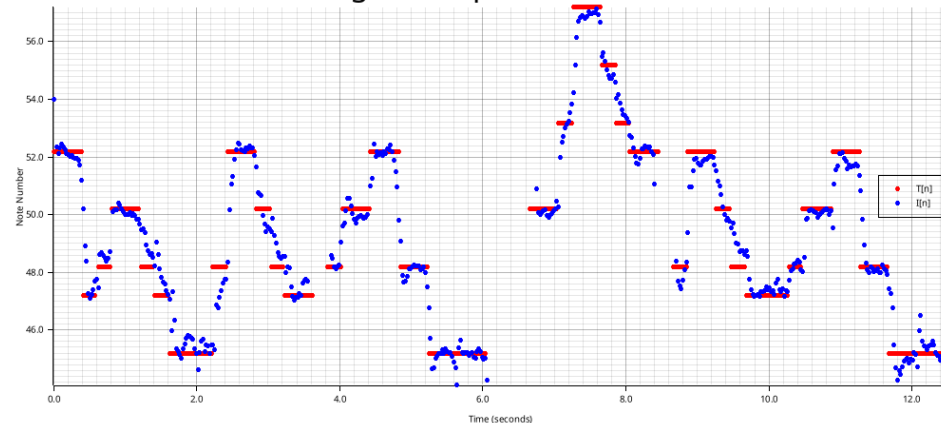
ผลการทดลอง

- การร้องเพลง *Tetris – Theme A*



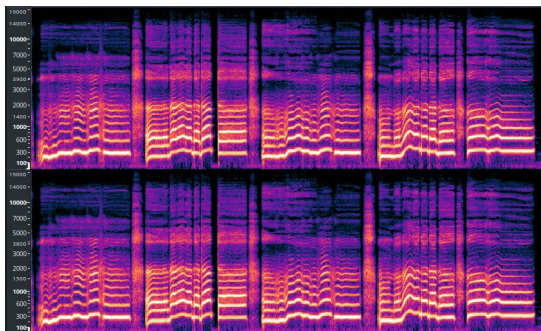
- ค่าความแม่นยำสุทธิ 98.42
- ครอบคลุม: 99.03 เวลา: 100
- ระดับเสียง: 99.18 ฤญแจเสียง: 100

Target vs Input Melodies

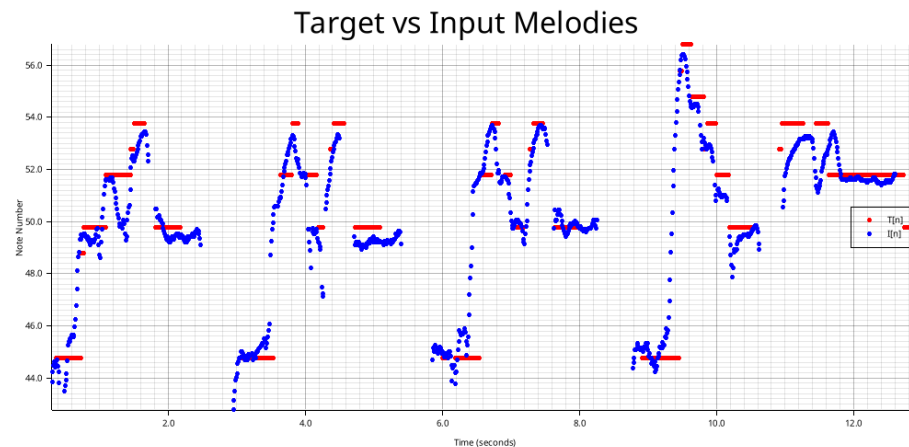


ผลการทดลอง

- การฮัมเพลง *Sān-Z – BITE!*

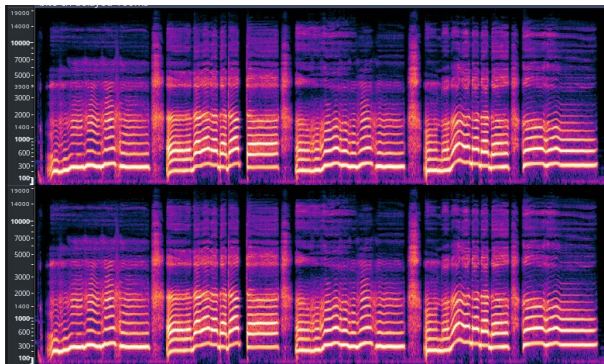


- ค่าความแม่นยำสุทริ 66.86
- ครอบคลุม: 98.46 เวลา: 100
ระดับเสียง: 71.86 ฤญแจเสียง: 56.02



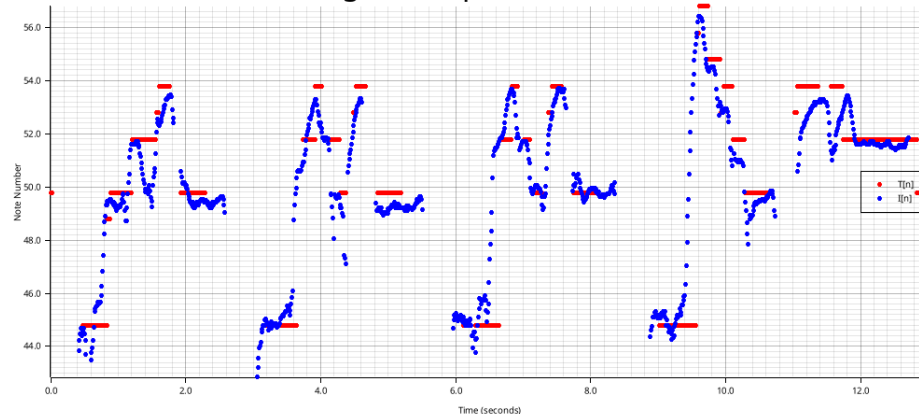
ผลการทดลอง

- การฮัมเพลง *Sān-Z – BITE!* โดยล่าช้า 100 มิลลิวินาที



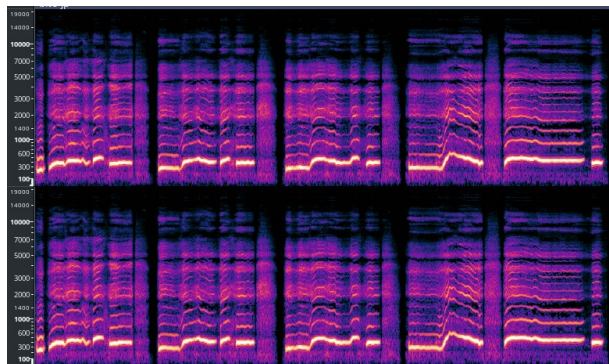
- ค่าความแม่นยำสุทริ 18.91
- ครอบคลุม: 98.47 เวลา: 28.33
- ระดับเสียง: 71.67 ฤญแจเสียง: 56.02

Target vs Input Melodies



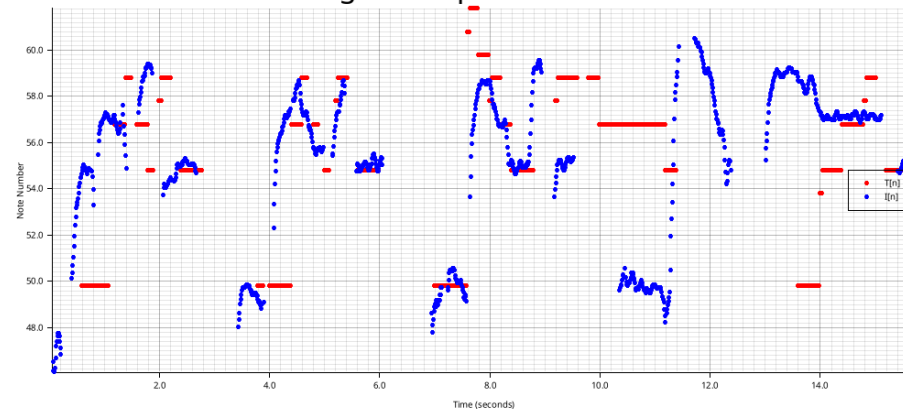
ผลการทดลอง

- การอิมเพลง *Sān-Z – BITE!* โดย **ฮัมมีสะดุด**



- ค่าความแม่นยำสุทธิ 0
- ครอบคลุม: 96.31 เวลา: 0
ระดับเสียง: 32.8 ภัยแเสียง: 100

Target vs Input Melodies




การวัดประสิทธิภาพของวิธีการนำเสนอ

- **การทดลองควบคุม:** เสียงสังเคราะห์ได้คะแนนสูงในบางมิติเนื่องจากความสม่ำเสมอของคลื่นเสียง แต่ต่ำในด้านความตรงกับทำนอง
- **การทดลองที่ตั้งใจร้อง:** เสียงร้องในสภาวะควบคุมได้คะแนนสูงในทุกมิติเนื่องจากลดปัจจัยรบกวน และระบบสามารถชดเชยระดับเสียงที่ต่างกันได้ดี
- **การทดลองที่ฮัมเพลง:** การฮัมเพลงให้คะแนนที่สูงแต่ไม่สมบูรณ์ เนื่องจากไม่มีความตั้งใจให้แม่นยำตามโน้ต
- **การทดลองที่มีการล่าช้า:** ถูกหักคะแนนตามความเหมาะสม ได้คะแนนความตรงต่อเวลาน้อย
- **การทดลองที่ฮัมเพลงสะดุด:** การฮัมที่มีสะดุดให้คะแนนต่ำสุด เนื่องจากระบบลงโทษการร้องที่ขาดช่วง
- โดยรวม ระบบสามารถให้คะแนนได้อย่างยุติธรรมและสมเหตุสมผล

สรุปผล

- โครงการนี้พัฒนาระบบให้คะแนนความแม่นยำในการร้องเพลงตามโน้ตดนตรีอย่างยุติธรรม โดยใช้ไฟล์ MIDI เป็นข้อมูลทำนองและไฟล์ WAV เป็นข้อมูลเสียงร้อง แล้วคำนวณค่าความแม่นยำของเสียงร้องแต่ละช่วงเวลาเทียบกับโน้ตเป้าหมาย ผลลัพธ์แสดงให้เห็นว่าระบบสามารถให้คะแนนได้อย่างเหมาะสม ไม่เคร่งครัดหรือผ่อนปรนเกินไป พร้อมทั้งมีอินเตอร์เฟซที่เรียบง่ายและใช้ทรัพยากรต่ำ
- แม้โครงการจะสำเร็จตามเป้าหมาย แต่ยังสามารถพัฒนาเพิ่มเติมได้ เช่น การปรับค่า Parameter ให้เหมาะสมยิ่งขึ้น และการเปรียบเทียบกับระบบอื่นเพื่อประเมินประสิทธิภาพ หากสนใจใช้งานหรือต่อยอด สามารถเข้าถึงซอร์สโค้ดได้ที่ <https://github.com/krtchnt/cantometria>



**Thank you
for your attention.
Any questions?**