Conversación sobre Piano Modal en Tiempo Real

Fecha de exportación: 2025-07-05 00:38:52

## Usuario:

La duración es fija, como se debería implementar para que sea variable o que factores se deben considerar? primero coméntame tu análisis

## Asistente:

Buena pregunta. Usar una \*\*duración fija\*\* (por ejemplo `duration=2.0`) en tu motor actual tiene sentido para prototipos, pero \*\*no se alinea con el comportamiento real de un piano acústico\*\*, donde el sonido de cada nota:

- Varía su duración en función del \*\*registro\*\* (notas graves duran más).

- Depende de la \*\*fuerza (velocidad) del golpe\*\*.

- Y está influenciada por el \*\*pedal de sustain\*\*, \*\*resonancias simpáticas\*\* y el \*\*apagador (damping)\*\*.

---

## 🎼 Análisis de una duración realista y variable

### 1. 🎵 \*\*Naturaleza de la cuerda real\*\*

En un piano acústico:

- Las notas \*\*graves\*\* pueden durar hasta \*\*10 segundos o más\*\*, incluso sin pedal.

- Las notas \*\*agudas\*\* se extinguen en \*\*menos de 1 segundo\*\*.

- Las cuerdas más gruesas vibran más tiempo porque su energía se disipa más lentamente.

Este comportamiento \*\*no es lineal\*\*: depende de masa, tensión, punto de contacto, calidad de martillo, y más.

---

### 2. 🔊 \*\*Factores que influyen en la duración sintética\*\*

| Factor | Efecto esperado en duración |

|---------------------------|-----------------------------|

| `nota\_midi` | Graves duran más, agudos menos |

| `velocity` | A mayor velocidad, más duración (energía) |

| `sustain pedal` | Detiene o prolonga liberación de señal |

| `modo` (número armónico) | Los modos más altos se apagan más rápido |

| `resonancias` | Aumentan duración global o agregan "cola" post-apagador |

---

### 3. 🧠 En tu modelo: ¿dónde se impone la duración?

En tu motor, la duración se fija al generar la señal:

```python

generate\_realistic\_modal\_signal(midi\_note, duration, fs, ...)

```

Pero en la vida real, cada voz debería tener su \*\*envelope natural\*\*, como una curva de decaimiento que llega a 0 en forma progresiva, sin necesidad de "cortar" la señal a los `2.0` segundos.

---

## ✅ Propuestas de mejora

### ✅ A. \*\*Usar `duration` como duración máxima, no recorte abrupto\*\*

Deja que cada modo se apague naturalmente con su propio `alpha`, y considera un umbral de silencio (por ejemplo -60 dB) para terminar el buffer.

### ✅ B. \*\*Ajustar `alpha` (decay) por nota y velocidad\*\*

Haz que el `alpha` por modo se calcule así:

```python

alpha = base\_decay[n] \* (1 / velocity) \* nota\_factor

```

Donde:

- `base\_decay[n]` es el coeficiente de amortiguación del n-ésimo modo.

- `nota\_factor` varía por registro (más bajo para graves).

### ✅ C. \*\*Implementar un modelo de apagado (note\_off)\*\*

Cuando se recibe `note\_off`, empieza una fase de \*\*apagado más rápido\*\*, o directamente \*\*elimina la voz\*\*, dependiendo si `sustain` está activo o no.

---

### 🧪 ¿Y cómo lo harías en la práctica?

- Define un `VoiceEnvelope` que vigila el decaimiento en dB.

- Establece una \*\*longitud máxima estimada\*\* por nota (`max\_duration`), pero permite que cada `Voice` termine cuando su amplitud es suficientemente baja.

---

¿Quieres que prepare un modelo de amortiguación realista por nota, o pasamos a rediseñar `NoteVoice` para permitir duración variable basada en condiciones naturales y no arbitrarias?