Date: 2025-09-08 (Mon) 📅

Prepared by: Tiong-Sik Ng 📝

English 🇺🇸

1. Summary of Accomplishments ✅

- Conducted **extensive debugging of OpenVoice audio quality**. Identified that muffling largely comes from spectrum shaping before telephony conversion.
- Developed and tested **FFmpeg filter chains** (high-pass, presence boost, compand, limiter) to improve clarity. Final “tame” profile yielded noticeably better results without clipping.
- Integrated these improvements directly into `app.py` with a **`tel_profile` toggle** (`flat | tame | bright`), allowing runtime selection of telephony EQ profiles.
- Built and validated a **CLI client (`ov_cli.py`)** for quickly generating test samples from the running Uvicorn+OpenVoice server. Confirmed end-to-end functionality for both hi-fi (24 kHz PCM) and telephony (8 kHz μ-law).
- Explored **alternative TTS models** (Marvis, Kokoro, CLEAR, StreamMel, Kyutai, SyncSpeech, Chatterbox) to benchmark against OpenVoice and FishSpeech for latency vs. naturalness.
- Compared OpenVoice vs. FishSpeech under identical telephony conditions: confirmed FishSpeech vocoder preserves 2–4 kHz band better, but OpenVoice can be enhanced with EQ/compand.

2. Issues & Risks 🔄

- Despite improvements, **OpenVoice still lacks true emotional prosody**; DSP presets only provide subtle differences.
- **FishSpeech remains superior** in naturalness, but unusable for low-latency telephony applications.
- Fine-tuning efforts with KSponSpeech may improve tone clarity, but not emotional expression.
- Audio quality tuning is delicate: too much EQ or gain causes clipping/harshness.

3. Next Steps 📌

- Fine-tune OpenVoice’s tone converter with **KSponSpeech dataset** to enhance Korean timbre clarity.
- Continue refining the **tame/bright profiles** based on real telephony handset testing.
- Explore integration of **pre-generated expressive FishSpeech samples** for common phrases while using OpenVoice for live speech.

- Evaluate feasibility of newer models (Marvis/Kokoro/CLEAR) in the pipeline for a balance of latency and expressivity.

한국어 🇰🇷

1. 금일 수행 사항 ✅

- **OpenVoice 음성 품질 디버깅** 진행. 텔레포니 변환 전 스펙트럼 셰이핑 단계에서의 뭉개짐 원인을 확인.
- **FFmpeg 필터 체인**(하이패스, 프레즌스 부스트, 컴팬드, 리미터)을 적용하여 음질 개선. 최종 "tame" 프로필은 클리핑 없이 명료도를 확보.
- `app.py` 에 **`tel_profile` 토글**(`flat | tame | bright`)을 통합하여, 요청 시 텔레포니 EQ 프로필을 선택 가능하도록 구현.
- 실행 중인 Uvicorn+OpenVoice 서버와 연동되는 **CLI 클라이언트(`ov_cli.py`)** 제작 및 검증 완료. 24 kHz PCM(고음질) 및 8 kHz μ-law(텔레포니) 모두 정상 동작 확인.
- **대체 TTS 모델**(Marvis, Kokoro, CLEAR, StreamMel, Kyutai, SyncSpeech, Chatterbox)을 조사하여 OpenVoice, FishSpeech 와 지연시간·자연스러움 비교.
- OpenVoice 와 FishSpeech 를 동일한 텔레포니 환경에서 비교: FishSpeech 보코더가 2~4 kHz 대역을 더 잘 보존함을 확인했으나, OpenVoice 는 EQ/컴팬드로 보완 가능함.

2. 이슈 & 리스크 🔄

- 개선에도 불구하고 **OpenVoice 는 실제 감정 억양(prosody)을 지원하지 않음**. DSP 프리셋은 미묘한 차이만 제공.
- **FishSpeech 는 자연스러움이 우수**하지만, 텔레포니 실시간 적용에는 적합하지 않음.
- KSponSpeech 기반 파인튜닝은 한국어 톤 명료도 개선에는 유효하나, 감정 표현은 한계가 있음.
- 음질 튜닝은 민감하여, EQ 나 게인이 과하면 클리핑·거친 소리 발생.

3. 다음 단계 📌

- **KSponSpeech 데이터셋**으로 OpenVoice 톤 컨버터 파인튜닝 진행, 한국어 톤 컬러 명료도 개선.
- 실제 전화 단말기 테스트를 통해 **tame/bright 프로필**을 지속적으로 보정.
- **FishSpeech 로 사전 생성된 감정 표현 문구**(자주 쓰이는 멘트)를 캐싱하여 사용, OpenVoice 는 실시간 문장 전용으로 활용.
- **Marvis/Kokoro/CLEAR** 등 신형 모델의 파이프라인 적용 가능성을 검토하여, 지연시간·자연스러움 균형 추구.