LiveKit Voice AI Quickstart (Realtime Model)

Overview

This guide walks you through building a real-time voice assistant using LiveKit Agents with a focus on OpenAI for natural language processing, Silero for voice activity detection (VAD), and noise cancellation for clear audio. This streamlined setup relying OpenAI for both transcription and generation capabilities.

Requirements

- Python 3.9 or later
- LiveKit server (Cloud or self-hosted)
- API key for OpenAI

Install Required Packages

Run the following command to install required packages:

```
pip install \
  "livekit-agents[openai,silero,turn-detector]~=1.0" \
  "livekit-plugins-noise-cancellation~=0.2" \
  "python-dotenv"
```

Environment Variables

Create a `.env` file with the following content:

```
OPENAI_API_KEY=<your_openai_api_key>
LIVEKIT_API_KEY=<your_livekit_api_key>
LIVEKIT_API_SECRET=<your_livekit_api_secret>
LIVEKIT_URL=<your_livekit_server_url>
```

Using OpenAI for Real-Time Mode

OpenAI powers both transcription and conversation generation in this streamlined setup. The `gpt-4o` model is capable of understanding streaming input and generating natural, context-aware responses in real-time. You don't need separate STT or TTS providers when using OpenAI in this role.

Silero VAD and Noise Cancellation

Silero VAD (Voice Activity Detection) is used to determine when the user is speaking and when they've finished. This helps with managing queue in conversations, avoiding interruptions or premature replies.

Noise Cancellation

Noise Cancellation is enabled via the `livekit-plugins-noise-cancellation` plugin and improves input quality by filtering out background noise. This ensures clearer audio capture, especially in noisy environments.

Example Agent Code

Below is a basic example of setting up the assistant:

```
from dotenv import load_dotenv
from livekit import agents
from livekit.agents import AgentSession, Agent, RoomInputOptions
from livekit.plugins import openai, silero, noise_cancellation
from livekit.plugins.turn_detector.multilingual import MultilingualModel

load_dotenv()

class Assistant(Agent):
    def __init__(self) -> None:
        super().__init__(instructions="You are a helpful voice assistant.")

async def entrypoint(ctx: agents.JobContext):
    await ctx.connect()
    session = AgentSession(
        stt=openai.STT(),  # OpenAI handling transcription
        llm=openai.LLM(model="gpt-4o-mini"),  # Conversational model
        tts=openai.TTS(),  # OpenAI for text-to-speech
        vad=silero.VAD.load(),
        turn_detection=MultilingualModel(),
    )
    await session.start(
        room=ctx.room,
        agent=Assistant(),
        room_input_options=RoomInputOptions(
            noise_cancellation=noise_cancellation.BVC()
        )
    )
    await session.generate_reply("Hello! How can I help you today?")

if __name__ == "__main__":
    agents.cli.run_app(agents.WorkerOptions(entrypoint_fnc=entrypoint))
```

Run the Agent

To download model files:

```
python main.py download-files
```

To start voice interaction via terminal:

```
python main.py console
```

To connect to an online room for testing:

```
python main.py dev
```