

LiveKit Intelligent Interruption Handling Challenge

Repository to Use

Fork and work on this repository:

<https://github.com/Dark-Sys-Jenkins/agents-assignment>

DO NOT RAISE PR IN ORIGINAL LIVEKIT REPO

Overview

This challenge tests your ability to refine the conversational flow of a real-time AI agent.

The Problem:

Currently, when the AI agent is explaining something important, LiveKit's default Voice Activity Detection (VAD) is too sensitive to user feedback. If the user says "yeah," "ok," "aha," or "hmm" (known as backchanneling) to indicate they are listening, the agent interprets this as an interruption and abruptly stops speaking.

The Goal:

You must implement a logic layer that is **context-aware**. The agent must distinguish between a "passive acknowledgement" and an "active interruption" based on whether the agent is currently speaking or silent.

Strict Requirement:

If the agent is speaking and the user says a filler word, **the agent must NOT stop**. It must continue its sentence seamlessly. **Partial solutions where the agent pauses and then resumes, or stutters, will not be accepted.**

Core Logic & Objectives

Your system must handle the following logic matrix:

User Input	Agent State	Desired Behavior
------------	-------------	------------------

"Yeah / Ok / Hmm"	Agent is Speaking	IGNORE: The agent continues speaking without pausing or stopping.
"Wait / Stop / No"	Agent is Speaking	INTERRUPT: The agent stops immediately and listens to the new command.
"Yeah / Ok / Hmm"	Agent is Silent	RESPOND: The agent treats this as valid input (e.g., User: "Yeah." -> Agent: "Great, let's continue.").
"Start / Hello"	Agent is Silent	RESPOND: Normal conversational behavior.

Key Features to Implement:

1. **Configurable Ignore List:** Define a list of words (e.g., ['yeah', 'ok', 'hmm', 'right', 'uh-huh']) that act as "soft" inputs.
 2. **State-Based Filtering:** The filter must *only* apply when the agent is actively generating or playing audio.
 3. **Semantic Interruption:** If the user says a mixed sentence like "Yeah wait a second," this contains a command ("wait"). The agent **must** interrupt in this case.
 4. **No VAD Modification:** Do not rewrite the low-level VAD kernel. Implement this as a logic handling layer within the agent's event loop.
-

Technical Expectations

- **Integration:** Work within the existing LiveKit Agent framework in the provided repo.
 - **Transcription Logic:** You will likely need to utilize the STT (Speech-to-Text) stream. Since VAD is faster than STT, you need a strategy to handle the "false start" interruption (where VAD triggers a stop before the code realizes the user only said "yeah").
 - *Hint:* You may need to manage how the agent queues interruptions or validates text before cutting off the audio stream.
 - **Latency:** The solution must remain real-time. The delay to determine if a word is "valid" or "ignored" must be imperceptible.
-

Example Scenarios (Test Cases)

Scenario 1: The Long Explanation

- **Context:** Agent is reading a long paragraph about history.
- **User Action:** User says "Okay... yeah... uh-huh" while Agent is talking.

- **Result:** Agent audio **does not break**. It ignores the user input completely.

Scenario 2: The Passive Affirmation

- **Context:** Agent asks "Are you ready?" and goes silent.
- **User Action:** User says "Yeah."
- **Result:** Agent processes "Yeah" as an answer and proceeds (e.g., "Okay, starting now").

Scenario 3: The Correction

- **Context:** Agent is counting "One, two, three..."
- **User Action:** User says "No stop."
- **Result:** Agent cuts off immediately.

Scenario 4: The Mixed Input

- **Context:** Agent is speaking.
 - **User Action:** User says "Yeah okay but wait."
 - **Result:** Agent stops (because "but wait" is not in the ignore list).
-

Evaluation Criteria

1. **Strict Functionality (70%):**
 - Does the agent continue speaking over "yeah/ok"?
 - **Fail Condition:** If the agent stops, pauses, or hiccups on "yeah" while speaking, the submission is rejected.
 2. **State Awareness (10%):**
 - Does the agent correctly respond to "yeah" when it is *not* speaking? (It should not ignore valid short answers).
 3. **Code Quality (10%):**
 - Is the logic modular?
 - Can the list of ignored words be changed easily (e.g., environment variable or config array)?
 4. **Documentation (10%):**
 - Clear README.md explaining how to run the agent and how your logic works.
-

Submission Instructions

1. **Branch:** Create a new branch feature/interrupt-handler-<yourname> on your forked repo.
2. **Code:** Commit your changes. Ensure requirements.txt is updated if you added libraries.

3. **Proof:** Include a short video recording or a log transcript in your submission demonstrating:
 - The agent ignoring "yeah" while talking.
 - The agent responding to "yeah" when silent.
 - The agent stopping for "stop."
4. **Pull Request:** Submit the PR link to
<https://github.com/Dark-Sys-Jenkins/agents-assignment>

DO NOT RAISE PR IN ORIGINAL LIVEKIT REPO

⚠️ Reminder: Do not submit partial working code. The core challenge is the difference between *ignoring* a word while speaking vs. *hearing* the same word while silent.