PAI 最終課題 レポート

omnicampus アカウント名: Kazuyuki

プロジェクトタイトル: 音声認識 AI(Whisper)を用いた ROS 2 ロボット(turtlesim)の遠隔操作

1. 課題の概要と目的

本プロジェクトでは、AI による音声認識技術を活用し、人間の声でロボットを直感的に操作するタスクの実現を目指した。具体的には、OpenAI の音声認識モデル「Whisper」と、ROS 2 の標準シミュレータ「turtlesim」を連携させ、音声ファイルの内容に応じて亀（ロボット）を前後左右に動かすシステムを構築した。これにより、AI の推論結果をロボットの物理的なアクションに繋げるという、AI×ロボティクスの基本的なワークフローを実装した。

2. システム構成と使用技術

- **OS:** Ubuntu 22.04 (Docker コンテナ上)
- **ロボット制御:** ROS 2 Humble
- **シミュレータ:** turtlesim
- **AI モデル:** OpenAI Whisper (base モデル)
- **主要言語:** Python
- **主要ライブラリ:** rclpy, openai-whisper

処理フロー:

1. 事前に「前」「右」などと録音した音声ファイル(m4a)を用意。
2. Python スクリプトが Whisper モデルをロード。
3. 音声ファイルを Whisper が文字に変換（"みぎ" -> "右"）。
4. プログラムがテキストの内容を解釈し、対応する geometry_msgs/msg/Twist 型の ROS 2 メッセージを生成。
5. /turtle1/cmd_vel トピックに対してメッセージを送信。
6. turtlesim がメッセージを受け取り、亀が動く。

3. 実装で工夫した点・困難だった点

- **工夫した点 1（問題解決能力）：** 当初、より現実に近い物理シミュレータである Gazebo とロボットアーム(CRANE+ V2)を用いて開発を進めた。しかし、シミュレーションの物理的な不安定性（モデルの振動）により、コマンドを送ってもロボットが正常に動作しない問題に直面した。また、PC のスペック不足により Gazebo を起動するとすぐに落ちてしまうなどの問題も発生した。この問題の解決に時間を費やすよりも、本課題の核心である「AI と ROS の連携」を確実に見せることを優先し、軽量で安定した turtlesim にプラットフォームを切り替えるという戦略的な判断を行った。

- **工夫した点 2（AI の特性への対応）：** Whisper での文字起こしを試す中で、「みぎ」という発音が「ネギ」と誤認識されるケースがあることを発見した。これに対し、プログラムの条件分岐で if "右" in text or "ネギ" in text:のように両方の単語を許容するロジックを加え、AI の不確実性に対応する実用的な工夫を施した。
- **困難だった点：** Docker コンテナ環境からホスト PC のハードウェア（マイク）にアクセスする部分で問題が発生し、リアルタイムでの音声認識を断念した。今回はファイルベースでの実行としたが、今後の課題としたい。

4. 結果と考察

作成した Python ノードを実行することで、音声ファイルの内容に応じて turtlesim の亀を意図通りに操作することに成功した。これにより、AI の推論結果を ROS 2 のメッセージに変換し、ロボットを制御するという一連の流れを自力で構築できた。Gazebo での物理シミュレーションは断念したが、その過程で問題の切り分けや代替案の模索など、開発における重要なスキルを学ぶことができた。

5. 今後の展望

今後は、Docker のハードウェアアクセスの問題を解決し、リアルタイムの音声入力でロボットを操作するシステムに拡張したい。