

タブ 1

VOICEVOX server用意

目標は、デイサービスで働くロボット。今回は、VICEVOXのserverを立ち上げる。

構成

・W8yellow\$	Client PC CF-W8	Debian12
・Ryzen9\$	Server PC Minis Forum HX90	Debian12

目次

[概要](#)

[生成AIへの質問整理](#)

[VOICEVOXのServer立上](#)

[Docker imageの取得](#)

```
yo@Ryzen9:~\$ docker pull  
voicevox/voicevox engine:cpu-latest
```

```
yo@Ryzen9:~\$ docker images
```

[コンテナの起動](#)

```
yo@Ryzen9:~/command/13docker\_vocevox\$ cat  
02docker\_run\_voicevox.sh
```

```
yo@Ryzen9:~/command/13docker\_vocevox\$  
./02docker\_run\_voicevox.sh
```

```
yo@Ryzen9:~/command/13docker\_vocevox\$ docker ps
```

[動作確認](#)

```
noetic yo@Ryzen9:~/testPG\$ cat 03test\_voicevox.py
```

```
noetic yo@Ryzen9:~/testPG\$ python3
```

```
03test\_voicevox.py
```

```
noetic yo@Ryzen9:~/testPG\$ ls
```

[VOICEVOX ドキュメントの表示](#)

[所感](#)

[3つの回答](#)

[エラーの内容](#)

[今後の予定](#)

[参考](#)

[voicevox/voicevox engine - Docker Image](#)

[20250527テキスト音声変換の準備](#)

[質問:VOICEVOX Server立ち上げ](#)

[回答:Gemini](#)

[回答:ChatGPT\(採用\)](#)

概要

再度、VOICEVOXのサーバを立ち上げる。今回は、GPUを利用しない。Ryzen9だけで、どこまで出来るのか、知りたいところでもある。可能であれば、Hitabta単体で全てのプロセスを実行できるのか試したい。流石にそれは無理か？

生成AIへの質問整理

最近の作業の始まりは、ここからである。

- ・VOICEVOX公式のDockerを使って、サーバを立ち上げる。
- ・GPUは、利用しない。

以上かな。

これなら、わざわざAIに確認せずに、公式のHPで確認したほうが良いかもしれない。のだが。慣れとは、怖い。AIは、学習した結果である。なので、最新のことは、公式HPのほうが信頼できるのだが。

VOICEVOXのServer立上

Docker imageの取得

```
yo@Ryzen9:~$ docker pull voicevox/voicevox_engine:cpu-latest
```

```
cpu-latest: Pulling from voicevox/voicevox_engine
```

```
b08e2ff4391e: Pull complete
```

```
c6519a6f86ce: Pull complete
```

```
42a6f81776d2: Pull complete
```

```
ee79d07f45c6: Pull complete
```

```
280dc93a584a: Pull complete
```

```
2a336caa7ebf: Pull complete
```

```
Digest:
```

```
sha256:8730a7dd6d8e682dba524e2541a5e2b9ca0fb1fec18b5af70
```

```
9e9e4481ac40d5
```

Status: Downloaded newer image for
voicevox/voicevox_engine:cpu-latest
docker.io/voicevox/voicevox_engine:cpu-latest

yo@Ryzen9:~\$ docker images

REPOSITORY	TAG	IMAGE ID	
voicevox/voicevox_engine	cpu-latest	11ec2cc1083a	4
days ago	2.09GB		
ollama/ollama	latest	33719ca5f6af	
10 days ago	2.27GB		
ros-noetic-image	latest	e181ab052982	6
months ago	1.04GB		
melodic2_saved_image02	latest	9f3988d297c1	
12 months ago	4.04GB		

コンテナの起動

yo@Ryzen9:~/command/13docker_vocevox\$ cat
02docker_run_voicevox.sh

```
#!/bin/bash
docker run -d \
  --name voicevox_cpu \
  -p 50021:50021 \
  voicevox/voicevox_engine:cpu-latest
```

yo@Ryzen9:~/command/13docker_vocevox\$
./02docker_run_voicevox.sh

yo@Ryzen9:~/command/13docker_vocevox\$ docker ps

CONTAINER ID	PORTS	NAMES
436c8c779020	0.0.0.0:50021->50021/tcp,	
:::50021->50021/tcp	voicevox_cpu	
9f83047f6ed8		
ros_noetic		

動作確認

```
noetic yo@Ryzen9:~/testPG$ cat 03test_voicevox.py
```

```
import requests
```

```
text = "こんにちは、これはテストです。"
```

```
speaker = 1 # 話者ID(1 = 四国めたん ノーマル)
```

```
# 音声合成用クエリ作成
```

```
query = requests.post(  
    "http://localhost:50021/audio_query",  
    params={"text": text, "speaker": speaker}  
)  
query_json = query.json()
```

```
# 音声データ生成
```

```
response = requests.post(  
    "http://localhost:50021/synthesis",  
    params={"speaker": speaker},  
    json=query_json  
)
```

```
# 保存
```

```
with open("output.wav", "wb") as f:  
    f.write(response.content)
```

```
noetic yo@Ryzen9:~/testPG$ python3 03test_voicevox.py
```

```
noetic yo@Ryzen9:~/testPG$ ls
```

```
03test_voicevox.py output.wav pyopenjtalk_test.py
```

```
vosk_version.py
```

上記、output.wavをW8yellowにコピーaplayで確認したところ、音声再生OKでした。

VOICEVOX ドキュメントの表示

すごいですね。これは。以下にアクセスすると、有用なドキュメントを参照することができます。

所感

3つの回答

ChatGPTの回答が3つ出てきた？。

1回目の回答で、回答の最後にネットワークエラーなるものが発生。回答をコピーボタンでコピーすることができなかった。更新ボタンがあったので、更新。

2回めの回答は、回答内容に不審な点。後述。不審なのでもう一度更新。

3回目の回答も、納得できない。

それぞれの回答で、気になった点。docker pullコマンド

1回目 : docker pull voicevox/voicevox_engine:cpu-latest

2回目 : docker pull voicevox/voicevox_engine:nonnvidia-latest

3回目 : docker pull voicevox/voicevox_engine:latest

結局、公式ドキュメントを確認したところ、1回目ではないかと思うので1回目を採用した。コピーは、ChatGPTの回答にあるボタンではなく、テキストを選択してコピーを行った。1回目のエラーを下記に転機する。

エラーの内容

エラー表示内容:テキストのコピーに制限がかかっている。

しかし、エラーの内容をChatGPTに翻訳してもらっている。

質問:

以下を翻訳してください。""A network error occurred. Please check your connection and try again. If this issue persists please contact us through our help center at help.openai.com.""

回答:

以下が日本語訳です:

**「ネットワークエラーが発生しました。接続を確認して、もう一度お試しください。この問題が解決しない場合は、
help.openai.com のヘルプセンターからお問い合わせください。」**

ネットワークを確認しても、問題はない。初回の回答が一番信頼ができる。

以前にも、同様なエラーが発生したことがある。

そもそも、無料で使わせてもらっているシステムなので、不備があっても知恵で解決。

ただ、疑問にのこるだけ。

今後の予定

- ・生成AIをggmlからOllama に変更し、モデルをGemma3n:e4bとする。
- ・VOICEVOXのServerを立ち上げる。

<ここまでOK>

- ・テキスト音声出力をpyopenjtalkからVOICEVOXに変更する。
- ・再度、Ros noeticで音声テキスト変換:Wisperを利用できるか検討する。
- ・音声テキスト変換VOSKの辞書ファイルをsmallからnormalに変更する。
- ・音声テキスト変換されたテキストを評価するnodeを作成する。

「えっと」の削除、評価の後、AIに送るか「指示」に対して必要なnodeを起動するか決める。

- ・ロボットの顔を表示するシステムをqt5に変更する。
- ・利用者の顔を認識するシステムを構築する。
- ・移動装置を制作する。
- ・顔に向かって進む。
- ・認識した顔の方向に進むようにする。
- ・一定の場所に落ち着くようにする。
- ・認識した顔に、話しかけるようにする。

参考

[voicevox/voicevox_engine - Docker Image](https://voicevox.jp/voicevox_engine - Docker Image)

 20250527テキスト音声変換の準備

質問:VOICEVOX Server立ち上げ

VOICEVOX公式のDockerを使って、サーバを立ち上げる方法を教えてください。今回はGPUを利用できません。

回答: Gemini

VOICEVOXの公式Dockerイメージを使って、GPUを利用せずにサーバーを立ち上げる方法ですね。以下の手順で設定できます。

前提条件

* Dockerがインストールされていること。

手順

1. **Dockerイメージのプル**

まず、VOICEVOXの公式Dockerイメージを取得します。GPUを利用しない場合は、`cpu-only`タグの付いたイメージを使用します。

```
```bash
docker pull voicevox/voicevox_engine:cpu-only
```
```

2. **Dockerコンテナの実行**

次に、プルしたイメージを使ってDockerコンテナを実行します。ここでは、VOICEVOXエンジンがデフォルトで使用するポート(50021番)をホストマシンに公開します。

```
```bash
docker run -it --rm -p 50021:50021
voicevox/voicevox_engine:cpu-only
```
```

* `-it` : コンテナの標準入出力をターミナルに接続し、インタラクティブモードで実行します。

* `--rm` : コンテナが停止した際に、自動的にコンテナを削除します。

* `-p 50021:50021` : ホストマシンの50021番ポートと、コンテナの50021番ポートをマッピングします。

3. **サーバーの確認**

コンテナが正常に起動すると、VOICEVOXエンジンが起動し、指定したポートでリクエストを受け付けるようになります。

ブラウザで `http://localhost:50021/docs` にアクセスすると、VOICEVOXエンジンのAPIドキュメントが表示され、サーバーが正しく動作していることを確認できます。

補足

***永続化について**：上記のコマンドでは `--rm` オプションを使用しているため、コンテナが停止すると変更は失われます。もし設定やモデルを永続化したい場合は、ボリュームマウントを使用することを検討してください。ただし、VOICEVOXエンジンの場合、通常はAPIとして利用するため、永続化の必要性は低いかもしれません。

***バックグラウンド実行**：ターミナルを占有したくない場合は、`-d` オプションを使ってバックグラウンドで実行できます。

```
```bash
docker run -d --rm -p 50021:50021
voicevox/voicevox_engine:cpu-only
```
```

この場合、コンテナIDが出力されます。コンテナを停止するには、`docker stop <コンテナID>` を使用します。

***特定バージョンの指定**：`cpu-only` の最新版ではなく、特定のバージョンのVOICEVOXエンジンを使用したい場合は、`voicevox/voicevox\_engine:cpu-only-<バージョン番号>` のようにタグを指定してください。利用可能なタグはDocker HubのVOICEVOX公式リポジトリで確認できます。

これで、GPUを利用せずにVOICEVOXサーバーをDockerで立ち上げることができます。

回答: ChatGPT (採用)

GPUを使わずに、VOICEVOXの公式Dockerイメージを利用してサーバーを立ち上げるには、CPUモードで動作する設定が必要です。以下の手順で構築できます。

 手順概要

Docker をインストール済みであることを確認

VOICEVOX公式Dockerイメージの取得 (CPU版)

コンテナを起動 (必要なポートを開放)

1. VOICEVOX CPU版イメージの取得

VOICEVOX公式のCPU用Dockerイメージは、GitHubリリースページまたはDocker Hubで提供されています。

```
docker pull voicevox/voicevox_engine:cpu-latest
```

:cpu-latest がGPUなし環境用のタグです。

2. コンテナを起動する

以下のコマンドでVOICEVOXエンジンを起動します：

```
docker run -d \  
  --name voicevox_cpu \  
  -p 50021:50021 \  
  voicevox/voicevox_engine:cpu-latest
```

-p 50021:50021: ホストのポート50021をコンテナの50021にマッピング (APIのポート)

--name voicevox_cpu: コンテナ名を voicevox_cpu に指定

3. 起動確認

ブラウザまたは curl で以下を実行：

```
curl http://localhost:50021/version
```

正常であれば、エンジンのバージョン文字列が返ります。

 補足：Pythonからアクセスする例

以下のように、Pythonでサーバにテキストを送信し、音声を生成することができます：

```
import requests
```

```
text = "こんにちは、これはテストです。"
```

```
speaker = 1 # 話者ID(1 = 四国めたん ノーマル)
```

```
# 音声合成用クエリ作成
```

```
query = requests.post(  
    "http://localhost:50021/audio_query",  
    params={"text": text, "speaker": speaker}  
)  
query_json = query.json()
```

```
# 音声データ生成
```

```
response = requests.post(  
    "http://localhost:50021/synthesis",  
    params={"speaker": speaker},  
    json=query_json  
)
```

```
# 保存
```

```
with open("output.wav", "wb") as f:  
    f.write(response.content)
```

トラブルシューティング

症状 解決策

Cannot connect to the Docker daemon `sudo` を付ける、または
docker グループにユーザーを追加

ポートにアクセスできない `-p` オプションで 50021 をマッピングして
いるか確認

CPUが重すぎる 軽い話者(例:1 四国めたん)を選ぶ、長文を避
ける

必要であれば `systemd` や `Docker Compose` による自動起動設定もご
案内できます。ご希望があればお知らせください。