

More than 1 year has passed since last update.

@keisuke-okb (Keisuke Okubo) in DeepAI.org NPO法人AI開発推進協会

ython】GPT-4o+Whisper+VOICEVOXでアバターAI対話アプリを爆速

ion whisper チャットボット VOICEVOX GPT-4o

updated at 2024-09-01 Posted at 2024-09-01

はじめに

なアバターを使って理想の対話ができるアプリを作りたい…

かし、3DCGアプリケーションで制作したアバターを使ったAI対話アプリの構築は、アプリを動かすPCのリソース要件が高かったり(?)、BlenderやUnityといった3DCGを扱うソフトのスキルが必須で、私にとってハードルが高い。。

うことで、今回は以下のコンセプトで対話システムを構築してみました。

できる限り実装はPythonだけで完結させたい

・アップシンク、発話スピードなど対話システムとしての質はいったん無視

GPUを搭載していないPC環境でもアプリを動かしたい

・元のデスクトップにはGPUがあるので、VTuber向けの3Dアバター制作ソフトは一応動く

コンセプトでサンプルアプリを作ったところ、こんな感じになりました。

・アバター: VRoid Studio → 3tene → 動画キャプチャー

・対話生成: GPT-4o

・音声認識: whisper-mic

・音声合成: VOICEVOX ローカルAPI

・リアルタイムの音声認識チャット

・チャット履歴の画面表示

・対話に応じた画像表示

・最初の挨拶、会話の終了検知

アシスタントが発話中です

ご来店ありがとうございます。

本日はどのような商品をお探しでしょうか。

パソコンを買いたいと思ってて。

ありがとうございます。

パソコンでしたら、最新の「エアロブック・エックスワン(AeroBook X1)」はいかがでしょうか。



EZ-Video-Chatbot

はGitHubにありますので、ぜひ改造したうえでご活用ください！

変したアプリの概要

「ジービデオチャットボット」アプリ

アバターをキャプチャした動画や、人間が話している様子を撮影した動画を使ったチャットボットのサンプルアプリです。待機中（口を閉じている）」「発話中（口を開いている）」「対話開始（お辞儀、挨拶など）」の動画を用意するだけで、簡単にそのアバターがAI対話アシスタントになります。動画を用意する際は、ループ再生してもできるだけ最初と最後のつながりが自然になるように考慮してください。アシスタントの待機または発話中の時間が長かった場合、その分動画がループ再生されるためです。

概要

会話状況に応じて再生する動画を切り替えることで、疑似的に動画のアバター／人間がリアルタイムに話しているように見せることができます、「なんちゃって」対話アプリです。

フリーの音声認識ライブラリ、フリーの音声合成ソフトを使い、リアルタイムに対話をすることができます。

対話状況に応じて、画像を表示することができます。（カスタマイズには `main.py` を変更する必要があります）

高性能なGPUを必要とする3DCGエンジンは不要です。より快適な動作のためにGeForce RTXシリーズ等のディスクリートGPUを使うより良いですが、CPUのみのPCでも最低限動作するように設計しています。

Tkinterを利用し、GUI部品の画像を高速に再描画することで動画再生を実現しているため、CPUのスペックにより再生の滑らかさが異なります。

音声認識、音声合成にはローカルのモデルを利用しますので、アプリを実行するPCのスペックによって反応が遅くなることがあります。対話画面は1920x1080で固定しています。画面のスケーリング設定によって対話画面が拡大され、ディスプレイからはみ出します。その場合、OSの表示倍率の設定を100%に変更してください。

フルHD未満のディスプレイには非対応です。

実装方法

onにデフォルトで搭載されているGUIモジュール「Tkinter」を活用して対話ウィンドウを実装しました。

な処理のシーケンスは以下のようにしました。



舌開始時の挙動

初期に行いたい部分は `threading.Thread()` を多用しています。。

初期メッセージを追加: `self.messages` にアシスタントの初期メッセージを追加

メッセージを分割: 初期メッセージをセグメントに分割

音声生成スレッドの開始: 音声生成のためのスレッドを開始

チャットバブルの作成: 各セグメントに対してチャットバブルを作成

よくあるチャットの吹き出しを「チャットバブル」と定義します。

ご来店ありがとうございます。

本日はどのような商品をお探しでしょうか。

ラベルの変更: アシスタントの状態を示すラベルを変更

音声ファイルの存在確認: 音声ファイルが存在するか確認

ビデオの再生: 必要に応じてビデオを再生

チャット表示の更新: チャットの表示を更新

音声の再生: 音声ファイルを再生

```
s EZVideoChatBotApp:  
# ..(中略)..  
def start_chat(self):  
    self.messages.append(  
        {"role": "assistant", "content": self.init_message}  
    )  
  
    assistant_response_segments = tools.split_text(self.init_message)  
    gen_thread = threading.Thread(target=tts.generate_voice, args=(assistant_response_segments,))  
    gen_thread.daemon = True  
    gen_thread.start()  
  
    for i, seg in enumerate(assistant_response_segments):  
        if len(assistant_response_segments) == 1:  
            mode = "single"  
        elif i == 0:  
            mode = "start"  
        elif i == len(assistant_response_segments) - 1:  
            mode = "end"  
        else:  
            mode = "middle"  
        create_chat_bubble(seg, role="assistant", mode=mode)  
  
        self.change_label("assistant", Constants.GENERATING_VOICE_TEXT)  
        while True:  
            if os.path.exists(f"./audio/{i}.wav"):  
                break  
  
        if i == 0:  
            self.play_video(self.starting_video)  
            sleep(2.5)
```

```
self.change_label("assistant", Constants.ASSISTANT_SPEAKING_TEXT)
self.update_chat_display()
self.play_video(self.speaking_video)
tts.play_sound(f"./audio/{i}.wav")
self.play_video(self.waiting_video)
```

舌ループ

- ./chats と ./audio フォルダ内のファイルを削除
チャット画面を更新し、チャットを開始
無限ループ内で以下の処理を実行：
- ./audio フォルダ内のファイルを削除
 - ユーザーの音声入力を取得し、テキストに変換
 - ユーザーの入力をチャットバブルに分割して表示
 - チャット画面を更新
 - アシスタントの応答を生成し、チャットバブルに分割して表示
 - 特定のキーワードに応じて画像をオーバーレイ表示
 - アシスタントの応答を音声に変換し、再生
 - 必要に応じてチャット履歴と音声ファイルをリセット

```
def main_chat(self):
    tools.delete_files_in_folder("./chats")
    tools.delete_files_in_folder("./audio")
    self.update_chat_display()
    self.start_chat()

    while True:
        try:
            tools.delete_files_in_folder("./audio")
            self.change_label("user", Constants.LISTENING_TEXT)
            user_input = self.stt_model.listen()

            self.change_label("assistant", Constants.GENERATING_CHAT_TEXT)
            self.messages.append(
                {"role": "user", "content": user_input}
            )
            user_input_segments = tools.split_text(user_input)
            for i, seg in enumerate(user_input_segments):
                if len(user_input_segments) == 1:
                    mode = "single"
                elif i == 0:
                    mode = "start"
                elif i == len(user_input_segments) - 1:
                    mode = "end"
                else:
                    mode = "middle"
                create_chat_bubble(seg, role="user", mode=mode)

            self.update_chat_display()
```

```

assistant_response, reset = chat.run_completion(self.messages)
self.messages.append(
    {"role": "assistant", "content": assistant_response}
)

# Customize image display
if "クリーンマスター" in assistant_response:
    overlay_image = "./images/cleanmaster.png"
elif "エアロブック" in assistant_response:
    overlay_image = "./images/aerobook.png"
else:
    overlay_image = None

assistant_response_segments = tools.split_text(assistant_response)
gen_thread = threading.Thread(target=tts.generate_voice, args=(assistant_response_segments))
gen_thread.daemon = True
gen_thread.start()

for i, seg in enumerate(assistant_response_segments):
    if i == 0:
        mode = "start"
    elif i == len(assistant_response_segments) - 1:
        mode = "end"
    else:
        mode = "middle"
    create_chat_bubble(seg, role="assistant", mode=mode)

    self.change_label("assistant", Constants.GENERATING_VOICE_TEXT)
    while True:
        if os.path.exists(f"./audio/{i}.wav"):
            break

    self.change_label("assistant", Constants.ASSISTANT_SPEAKING_TEXT)
    self.update_chat_display(overlay_image=overlay_image)
    self.play_video(self.speaking_video)
    tts.play_sound(f"./audio/{i}.wav")
    self.play_video(self.waiting_video)

if reset:
    tools.delete_files_in_folder("./chats")
    tools.delete_files_in_folder("./audio")
    self.messages = [
        {"role": "system", "content": self.system_prompt}
    ]
    self.update_chat_display()
    self.start_chat()

except Exception as e:
    print(e)

```

舌生成

ia-indexでGPT-4oのAPIを利用して、RAGデータを用いた会話生成を実装しました。

```
.py
rt openai

| llama_index.llms.openai import OpenAI
| llama_index.core import VectorStoreIndex, SimpleDirectoryReader
| llama_index.core.llms import ChatMessage

| constants import Constants

ai.api_key = Constants.OPENAI_API_KEY

= OpenAI(
model="gpt-4o",
temperature=0.5,
max_tokens=512,
streaming=True

ments = SimpleDirectoryReader("./rag").load_data()
x = VectorStoreIndex.from_documents(documents)
iever = index.as_retriever()

check_reset_strings(input_string, keywords):
return int(any(substring in input_string for substring in keywords))

run_completion(messages):
user_query = messages[-1]["content"]
if check_reset_strings(user_query, Constants.RESET_KEYWORDS_USER):
    return Constants.RESET_MESSAGE, 1
_messages = [
    ChatMessage(
        role=item['role'],
        content=item['content']
    )
    for item in messages
]

nodes = retriever.retrieve(user_query)
context = "\n".join([node.node.text for node in nodes])
_messages[0].content = Constants.SYSTEM_PROMPT.replace("{context}", context)

response_text = ""
response_stream = llm.stream_chat(_messages)
for response in response_stream:
    response_text += response.delta

return response_text, check_reset_strings(response_text, Constants.RESET_KEYWORDS_ASSISTANT)

run_completion_simple(messages):
user_query = messages[-1]["content"]
if check_reset_strings(user_query, Constants.RESET_KEYWORDS_USER):
    return Constants.RESET_MESSAGE, 1
```

```
response = openai.chat.completions.create(
    model="gpt-4o",
    messages=messages,
)
assistant_text = response.choices[0].message.content
return assistant_text, check_reset_strings(assistant_text, Constants.RESET_KEYWORDS_ASSISTANT)
```

音認識(STT)

whisperのラッパーモジュール「whisper-mic」を使うことで、特別な実装をしなくてもマイクからの入力を自動的に受け取りSTTができます。

トバブルの生成のため、文章の分割を行っています。

```
py
`whisper_mic import WhisperMic

s STTModel:
def __init__(self):
    self.model = WhisperMic(model="small")

def listen(self):
    result = self.model.listen()
    result = result.replace(" ", ".").replace("[", "").replace("]", "")
    if not result.endswith(".!?", "!", "?"):
        result += "."
    return result
```

音合成(TTS)

-の高機能音声合成ソフト「VOICEVOX」のローカルホストに起動するAPIを活用して、Pythonからも簡単に音声合成を投げました。

章の解析、②音声合成の2つの処理を投げる必要があります。今回はインストネーションなどのパラメータ調整はせずに、①で受け取ったデータをそのまま②に渡して生成を行いました。

```
rt requests
`pydub import AudioSegment
pydub.playback import play

constants import Constants
```

```

play_sound(sound_path):
    audio = AudioSegment.from_wav(sound_path)
    play(audio)

generate_voice(texts):
    for i, text in enumerate(texts):
        url = Constants.VOICEVOX_HOST + "/audio_query"
        params = {
            'text': text,
            'speaker': Constants.VOICEVOX_SPEAKER_ID
        }
        response = requests.post(url, params=params)
        query = response.text
        # query = query.replace('"pitchScale":0.0', '"pitchScale":0.02') # Customize parameters
        print(query)
        query = query.encode('utf-8')

    url = Constants.VOICEVOX_HOST + "/synthesis"
    headers = {
        'Content-Type': 'application/json'
    }
    params = {
        'speaker': Constants.VOICEVOX_SPEAKER_ID
    }

    response = requests.post(url, headers=headers, params=params, data=query)

    if response.status_code == 200:
        with open(f'./audio/{i}.wav', 'wb') as file:
            file.write(response.content)
        print("Command executed successfully.")
    else:
        print(f"Error executing command: {response.status_code}")

generate_voice("こんにちは。")

```

バブルの生成

ユアル的にも、この後連續して吹き出しが出るかどうかを分かりやすくするために、吹き出しが連結したような見た目になりました。

分けなどでコードが冗長な見た目になってしまったので、詳しくはGitHubのリポジトリをご覧ください。

吹き出し

-辺を角張るようにして、下に続くことを伝えます。

ご来店ありがとうございます。

の吹き出し

パソコンでしたら、最新の「エアロブック・エックスワン(AeroBook X1)」はいかがでしょうか。

の吹き出し

側は角を丸くして、会話の終了を示します。

どのような用途でお使いになる予定ですか？

-の吹き出し

部の角を丸くして、単体の吹き出しを表現します。

パソコンを買いたいと思ってて。

バター動画の用意

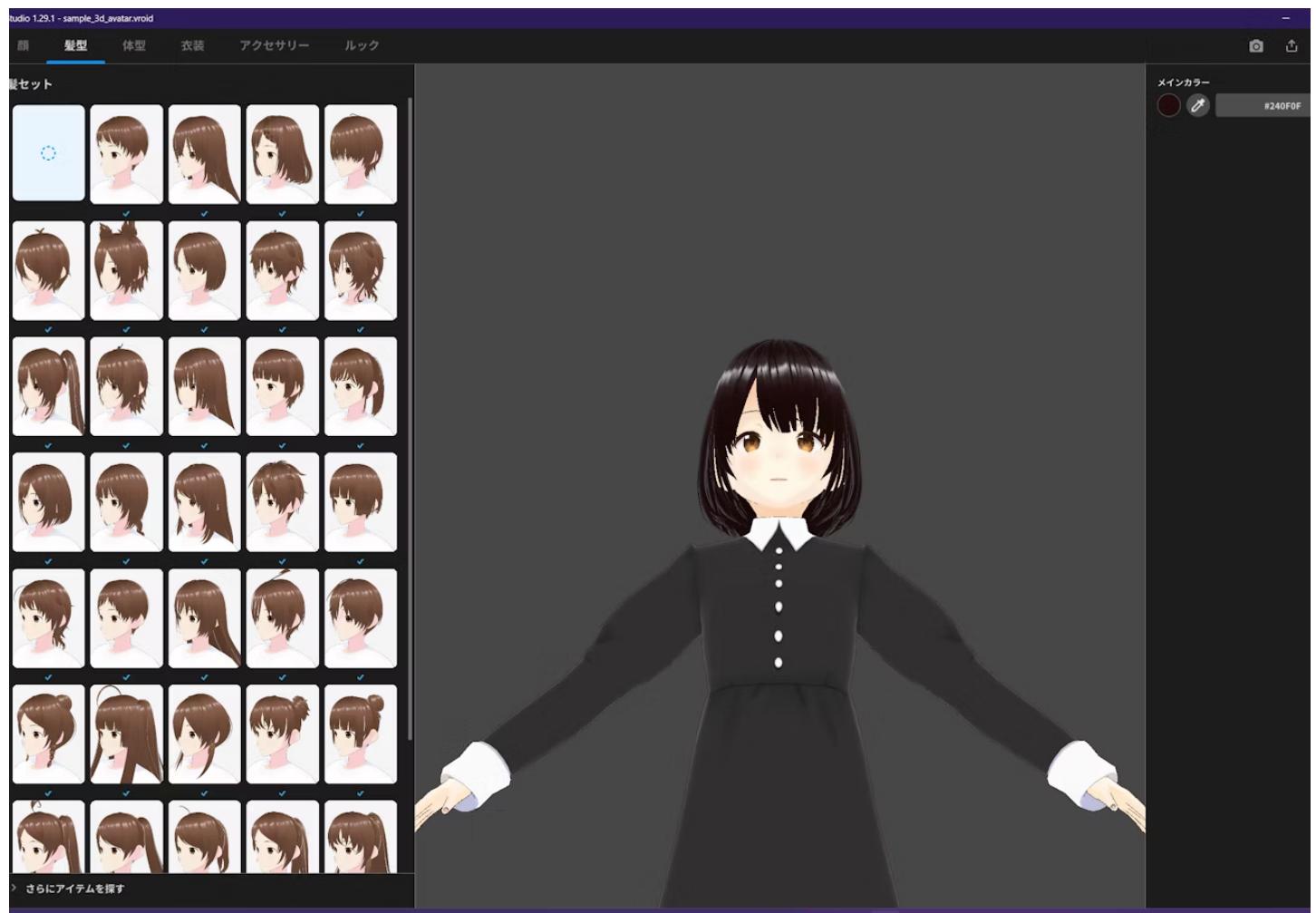
はVRoid Studioで作ったアバターを3teneで動かし、それをキャプチャーしてAviUtlで簡単に整えました。

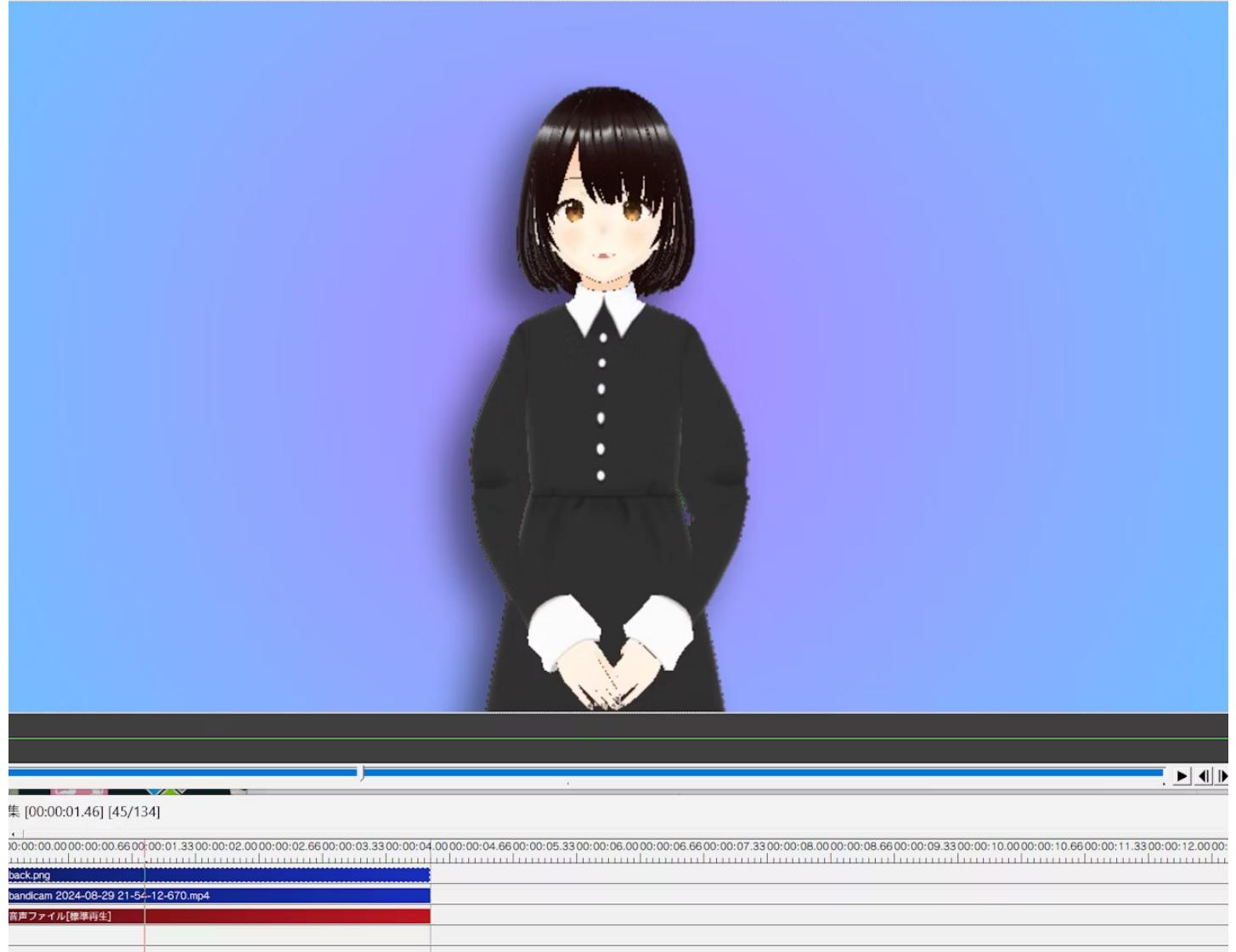
は対話を表現するため、以下の3つを動画ファイルとして書き出しました。

機中(口を閉じて待っている)

も話中(少し頭を動かしながら口を動かす)

最初の挨拶(お辞儀)





つりに

CEVOX、VRoid Studio等比較的簡単に使えるアプリを活用して、Pythonだけで対話システムを実装することができました！

全く無視したリップシンク、会話の生成速度、発話速度、感情表現、、などを考えると「対話システム」といえるかどうかはわかりません。簡単に動作する対話アプリとしてはいろいろな用途で使えそうです。



Register as a new user and use Qiita more conveniently

1. You get articles that match your needs
2. You can efficiently read back useful information
3. You can use dark theme

[What you can do with signing up](#)

[Sign up](#)

[Login](#)

isuke-okb's pickup articles

【Stable Diffusion】DiffusersでWebUIのHires.fix(latent)を再現してみる(Jupyter/Colab/ipythonなどで実行可能)

| Python StableDiffusion diffusers StableDiffusionWebUI 2023-06-19

【SoundFont作成効率化】Pythonで楽器からのサンプリング作業を劇的に早める方法

| Python FFT SoundFont 楽器 ピアノ 2024-04-14

【Python】で爆速構築】PCから出る音も録れるGUI録音アプリを作る(ステレオミキサー非搭載PC対応)【Python】

| Python Tkinter 録音 CustomTkinter 2024-04-25



@keisuke-okb (Keisuke Okubo)

ディープラーニングを中心に、自宅環境で再現、実装できそうなものを記事にしています。趣味では、サイト制作や、DTMを利用して耳コピでを打ち込み、カバー音源を制作しカラオケとして公開したり、VTuber／ライバーの方々の配信向けに音源提供したりしています。界隈では特ですが、ほとんど楽器が弾けず(ピアノは多少)、全楽器を五線譜で打ち込みます。個人的なコンセプト:情報×音楽×美術の融合

[Follow](#)



NPO法人AI開発推進協会

elurus.
NPO法人

誰もが手軽にAI(DeepLearning)を使える社会を創りたい。私たちはこのような社会の実現に向け、AIのモデルやアプリを開発できる「deepAelurus」を創出し活動をしています。

<https://sites.google.com/deepaelurus.com/aboutus/>

[Follow](#)

↗ Today's trending articles

【AI開発の珍プレー好プレー大賞!(珍プレー多め)】

TooMe in KDDIアジャイル開発センター株式会社

2025-10-23

GitHub Copilotを使っている人は全員"copilot-instrucions.md"を作成してください

| githubcopilot copilot-instructions.md

178

由kunitomo926 in Kaoエンジニアコミュニティβ

025-10-21

Claude × MCPで社内DBに話しかけてみた

TypeScript MCP AzureSQLDatabase Claude MCPサーバー

2 41

AI開発の珍プレー好プレー大賞!(珍プレー多め)

由yushibats in 日本オラクル株式会社

025-10-23

[OCI]新サービスのAIデータ基盤「Oracle AI Data Platform (AIDP)」を試す: ADBと連携してデータ

oracle adb AI oci aidp

2 16

由__nix__

025-10-22

Mermaid図経由でCopilotから情報流出 - 間接プロンプト注入型攻撃の新たな形

Security AI copilot LLM echoleak

2 22

由kazaf91 in 株式会社クラベス

025-10-21

Webサーバーサイド言語としてのRustについてご紹介

Rust WEBサーバー axum Dioxus

2 24

See all

記事 Recommended by  LOGLY

オープンソースAI(Whisper、BERT、VOICEVOX)を用いた音声対話ロボットの作成

by kunishou

自分のPCオンリーでキャラクターと音声対話がしたい!ローカルで動くspeech-to-speechサ...

by sakasegawa

ChatGPT+3Dモデルの音声対話エージェントを15分で開発する方法⚡

by uezo

自分で作ったAI同士を会話させてみた

by KokumaruGarasu

nments

omments

Let's comment your feelings that are more than good

Login

Sign Up

Qiita Conference 2025 Autumn will be held!: 11/5(Wed) - 11/7(Fri)



Conference is Qiita's largest tech conference, dedicated to engineers in the age of AI!

→ote Speaker

re, Tsuyoshi Ushio, Esteban Suarez, Takuto Wada, seya, MinoDriven, Toshihiro Ichitani, Karaage, Yoshimasa Iwase, Matz, Minorun and ...

[View event details →](#)

ing held Article posting campaign



「Qiita Coding」で創る、インストール不要の生成AI「Skywork」での開発体験を投稿しよう!

09-26 ~ 2025-10-30

[View details](#)



生成AI開発の珍プレー好プレー大賞!(珍プレー多め)

09-30 ~ 2025-11-10

[View details](#)

A