**Tên dự án : Quản gia Mini**

- HUFLIT Robotics -

Thông tin nhóm :

A. Thông tin Robot:

* Tên gốc : UGOT
* Model : Tranforming Car

A close up of text

Description automatically generated

**Đặc điểm:**

* Vượt chướng ngại vật
* Leo cầu thang
* Đi trên các địa hình chiều cao khác nhau
* Vận động linh hoạt

B. Chức năng chính

1. Di chuyển xung quanh nhà :

a. Tổng quan và thuật toán chức năng:

- Tạo bản đồ và xác định các điểm chỉ định bằng cách xây dựng một ma trận 2D hoặc một lưới tọa độ và xác định điểm chỉ định trên bản đồ đó.

- Lập trình thuật toán điều hướng bằng thuật toán A\* để tìm đường đi ngắn nhất từ vị trí hiện tại tới điểm chỉ định trên bản đồ.

- Lập trình các câu lệnh điều khiển robot tới vị trí chỉ định theo thuật toán đã định nghĩa từ trước

**b. Code mẫu ý tưởng tương ứng:**

Lập trình ra một bản đồ ảo bằng code dựa trên bản đồ thật:

**A screenshot of a computer

Description automatically generated**

Thuật toán A\*

**A screen shot of a computer program

Description automatically generated**

Điều khiển robot di chuyển theo đường đi

A screenshot of a computer program

Description automatically generated

2.

**Bước 1: Tích hợp mô hình nhận dạng giọng nói tiếng Việt**

1. **Cài đặt thư viện cần thiết**:
   * Cài đặt transformers, librosa, và các thư viện khác cần thiết cho mô hình Wav2Vec2.

bash

Copy code

pip install transformers librosa torch

1. **Tải mô hình nhận dạng giọng nói**:
   * Sử dụng mô hình khanhld/wav2vec2-base-vietnamese-160h từ Hugging Face.

python

Copy code

from transformers import Wav2Vec2Processor, Wav2Vec2ForCTC

import librosa

import torch

device = torch.device("cuda" if torch.cuda.is\_available() else "cpu")

processor = Wav2Vec2Processor.from\_pretrained("khanhld/wav2vec2-base-vietnamese-160h")

model = Wav2Vec2ForCTC.from\_pretrained("khanhld/wav2vec2-base-vietnamese-160h")

model.to(device)

1. **Hàm chuyển đổi giọng nói thành văn bản**:

python

Copy code

def transcribe(wav\_path):

wav, \_ = librosa.load(wav\_path, sr=16000)

input\_values = processor(wav, sampling\_rate=16000, return\_tensors="pt").input\_values

logits = model(input\_values.to(device)).logits

pred\_ids = torch.argmax(logits, dim=-1)

pred\_transcript = processor.batch\_decode(pred\_ids)[0]

return pred\_transcript

**Bước 2: Gọi API ChatGPT**

1. **Cài đặt thư viện yêu cầu**:
   * Sử dụng thư viện requests để gọi API ChatGPT.

bash

Copy code

pip install requests

1. **Gọi API ChatGPT**:

python

Copy code

import requests

def call\_chatgpt(api\_key, prompt):

url = "https://api.openai.com/v1/chat/completions"

headers = {

"Authorization": f"Bearer {api\_key}",

"Content-Type": "application/json"

}

data = {

"model": "gpt-4",

"messages": [{"role": "user", "content": prompt}]

}

response = requests.post(url, headers=headers, json=data)

return response.json()["choices"][0]["message"]["content"]

**Bước 3: Tích hợp với robot**

1. **Điều khiển robot đọc dữ liệu từ ChatGPT**:

python

Copy code

def robot\_speak(text):

# Đây là hàm giả lập, bạn cần thay thế bằng hàm thực sự điều khiển robot phát âm

print(f"Robot đọc: {text}")

def main(api\_key, wav\_path):

# Chuyển đổi giọng nói thành văn bản

transcript = transcribe(wav\_path)

print(f"Nhận dạng giọng nói: {transcript}")

# Gọi API ChatGPT với văn bản đã chuyển đổi

response = call\_chatgpt(api\_key, transcript)

print(f"ChatGPT trả lời: {response}")

# Yêu cầu robot đọc dữ liệu

robot\_speak(response)

# API key cho ChatGPT

api\_key = "your\_openai\_api\_key"

# Đường dẫn đến file âm thanh

wav\_path = "path/to/your/audio/file.wav"

main(api\_key, wav\_path)

**Bước 4: Thực hiện và thử nghiệm**

1. **Thử nghiệm mã nguồn**:
   * Chạy mã nguồn và kiểm tra xem robot có thể nhận dạng giọng nói, gọi API ChatGPT và đọc dữ liệu từ ChatGPT hay không.

**Bước 5: Tối ưu hóa và triển khai**

1. **Tối ưu hóa mã nguồn**:
   * Cải thiện hàm robot\_speak để robot phát âm chính xác.
   * Xử lý các lỗi có thể xảy ra khi gọi API hoặc nhận dạng giọng nói.
2. **Triển khai**:
   * Triển khai mã nguồn lên hệ thống của bạn và đảm bảo rằng robot hoạt động ổn định.
3. C. Nguồn tài liệu tham khảo:
4. Website thông tin về robot và API điều khiển : [Document (ubtrobot.com)](https://docs.ubtrobot.com/ugot/#/en-us/README.md)  
    **VietASR**: Đây là một dự án mã nguồn mở trên GitHub dành cho nhận dạng tiếng nói tiếng Việt. Dự án này sử dụng mô hình QuartzNet, được huấn luyện trên khoảng 100 giờ dữ liệu tiếng Việt từ nhiều nguồn khác nhau như YouTube, radio, và các bộ dữ liệu công khai như VLSP và VIVOS. Bạn có thể tìm hiểu thêm và sử dụng mã nguồn tại đây: [VietASR](https://github.com/dangvansam/viet-asr)​ ([GitHub](https://github.com/dangvansam/viet-asr))​.
5.  **Wav2Vec2-Base-Vietnamese-160h**: Đây là mô hình nhận dạng tiếng Việt dựa trên Wav2Vec2, được huấn luyện trên khoảng 160 giờ dữ liệu tiếng Việt từ các nguồn như VIOS, COMMON VOICE, và VLSP. Mô hình này cho phép bạn sử dụng mã nguồn mở để nhận dạng tiếng nói và bạn có thể tải và sử dụng mô hình này trên Hugging Face: [Wav2Vec2-Base-Vietnamese-160h](https://huggingface.co/khanhld/wav2vec2-base-vietnamese-160h)​ ([Hugging Face](https://huggingface.co/khanhld/wav2vec2-base-vietnamese-160h))​.
6.  **VietMed**: Đây là một bộ dữ liệu và mô hình nhận dạng tiếng nói tiếng Việt trong lĩnh vực y tế. VietMed bao gồm 16 giờ dữ liệu tiếng nói y tế đã được gán nhãn và 1200 giờ dữ liệu tiếng nói tổng hợp chưa gán nhãn. Các mô hình như w2v2-Viet và XLSR-53-Viet đã được huấn luyện và tinh chỉnh trên bộ dữ liệu này, có thể được sử dụng cho các ứng dụng liên quan đến y tế: VietMed Dataset​ ([ACL Anthology](https://aclanthology.org/2024.lrec-main.1509/))​.