



Universitat d'Alacant
Universidad de Alicante

Grado en Ingeniería Robótica
Robótica de servicios

Proyecto Robótica de servicios
NurseBot: Guía de Usuario

Autor:
Miguel Ferrer Castellá

Índice

1	Introducción	2
2	Instalación	2
2.1	RobotStudio	2
2.2	Base de Datos	2
2.3	Control por Visión	2
2.4	Agente conversacional	3
2.4.1	Configuración para usar la Tarjeta Gráfica	3
2.4.2	Detección de Habla	3
2.4.3	STT (Speech-to-Text)	3
2.4.4	LLM (Modelos de Lenguaje)	4
2.4.5	Traducción	5
2.4.6	TTS (Text-to-Speech)	5
2.5	Interfaz	6
3	Ejecución	6

1 Introducción

En este documento se mencionan las instalaciones de las distintas librerías, modelos y herramientas necesarias para llevar a cabo la puesta en marcha del proyecto NurseBot.

Posteriormente, se incluye una guía paso a paso para poder ejecutar los distintos programas que se integran dentro del proyecto y realizar las configuraciones pertinentes para el correcto funcionamiento de la aplicación.

2 Instalación

2.1 RobotStudio

Es necesario contar con el software RobotStudio de ABB e instalar la librería de OPC UA con:

```
pip install opcua
```

2.2 Base de Datos

Sólo es necesaria instalar la librería psycopg2, para trabajar una base de datos PostgreSQL.

```
pip install psycopg2
```

2.3 Control por Visión

Librerías de Python necesarias para implementar el control por visión en el proyecto:

```
pip install opencv
pip install face_recognition
pip install dlib
```

Además, es necesario contar con los siguientes elementos:

1. Un archivo llamado `persona.txt`.
2. Una carpeta llamada `'known_faces'`. Dentro de esta carpeta deben crearse subcarpetas con el nombre de cada persona. En cada subcarpeta deben incluirse las fotos correspondientes a esa persona para su reconocimiento.

Por ejemplo, la estructura de archivos debe ser:

```
known_faces/
  Juan/
    foto1.jpg
    foto2.jpg
  Maria/
    foto1.jpg
    foto2.jpg
persona.txt
```

2.4 Agente conversacional

Configuraciones y pasos necesarios para implementar el control por voz en el proyecto:

2.4.1 Configuración para usar la Tarjeta Gráfica

Python versión 3.9
Instalación de CUDA 12.1
Versión Pytorch: 2.5.1
Tensorflow: 2.18.0
cudnn-cu12: 9.1.0.70

2.4.2 Detección de Habla

Silero-VAD

```
pip install numpy>=1.24.0
pip install torch>=1.12.0
pip install torchaudio>=0.12.0
```

```
apt install python3-pyaudio (linux) or pip install pyaudio (windows)
```

2.4.3 STT (Speech-to-Text)

Whisper

```
pip install whisper
```

WhisperS2T

```
pip install -U whisper-s2t
```

Instalar ffmpeg dependiendo del entorno de ejecución:

- **Ubuntu:**

```
apt-get install -y libsndfile1 ffmpeg
```

- **Mac:**

```
brew install ffmpeg
```

- **Ubuntu/MAC/Windows/Cualquier Conda para Python:**

```
conda install conda-forge::ffmpeg
```

2.4.4 LLM (Modelos de Lenguaje)

Docker Ollama

1. Instalar el contenedor de herramientas de NVIDIA:

```
https://docs.nvidia.com/datacenter/cloud-native/\ncontainer-toolkit/latest/install-guide.html#installation
```

2. Configurar el repositorio:

```
curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey \n| sudo gpg --dearmor -o /usr/share/keyrings/\n  nvidia-container-toolkit-keyring.gpg\n\ncurl -s -L https://nvidia.github.io/libnvidia-container/stable/\n  deb/nvidia-container-toolkit.list \n| sed 's#deb https://#deb [signed-by=/usr/share/keyrings/\n  nvidia-container-toolkit-keyring.gpg] https://#g' \n| sudo tee /etc/apt/sources.list.d/\n  nvidia-container-toolkit.list\n\nsudo apt-get update
```

3. Instalar los paquetes del contenedor de herramientas de NVIDIA:

```
sudo apt-get install -y nvidia-container-toolkit
```

4. Configurar Docker para usar el controlador Nvidia:

```
sudo nvidia-ctk runtime configure --runtime=docker\nsudo systemctl restart docker
```

5. Inicar el contenedor:

```
docker run -d --gpus=all -v ollama:/root/.ollama -p 11434:11434\n--name ollama ollama/ollama
```

6. Descargar los modelos usados:

- **Embedding:**

```
ollama run nomic-embed-text:latest
```

- **Model:**

```
ollama run llama3.1:8b
```

Docker Flowise

1. Construir la imagen:

```
docker build -t flowise .
```

2. Ejecutar imagen:

```
docker run -d --name flowise -p 3000:3000 flowise
```

2.4.5 Traducción

```
pip install googletrans==4.0.0-rc1
pip install requests
```

2.4.6 TTS (Text-to-Speech)

```
pip install flask
```

```
sudo apt-get -y install libegl1
sudo apt-get -y install libopengl0
sudo apt-get -y install libxcb-cursor
pip install gradio==4.44.1
pip install fastapi==0.103.1
pip install pydantic==2.3.0
pip install ctranslate2==4.4.0
git clone https://huggingface.co/spaces/medallo/xtts-webui
cd xtts-webui
pip install -r requirements.txt

wget https://huggingface.co/medallo/xtts-model/resolve/main/\
model.zip?download=true -O /tmp/model.zip

unzip -d /content/xtts-webui/model /tmp/model.zip > /dev/null
```

```
wget https://huggingface.co/datasets/medallo/VZ/resolve/main/\
vc.zip?download=true -O /tmp/vc.zip > /dev/null 2>&1
```

```
unzip -d /tmp/Voice /tmp/vc.zip > /dev/null
```

2.5 Interfaz

```
pip install tkinter
pip install Pillow
```

3 Ejecución

Este será el orden de ejecución de los distintos programas que constituyen nuestro proyecto.

1. Inicio de los contenedores de Flowise y de Ollama.
2. Se deberá ejecutar el programa `deteccion_facial.py` para que vaya procesando las imágenes de todas las carpetas.
3. Se ejecutará el programa de `servidor_TTS.py` para habilitar las futuras peticiones para pasar de texto a voz.
4. Se abrirá el archivo de RobotStudio y se configurará la ip del equipo en que se pruebe y se dará al botón de iniciar la simulación.
5. Se ejecutará el programa `interfaz.py` que desplegará el menú interactivo de edición de los distintos datos presentes en la base de datos.
6. Se ejecutará el programa `main.py` y se cambiará la ip a la que corresponda en el equipo.
7. Se ejecutará el programa de `NurseBot_voice` que llevará el proceso de la interacción por voz con el sistema.