UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
FACULTAD DE INGENIERÍA ELECTRÓNICA

Proyecto Final

Manejo de Pepper

Presentado a: Ing. Diego Alejandro Barragán Vargas
Valentina Perez Fernandez , Cód. 2341132

Resumen*— *En el contexto actual de la robótica social y la inteligencia artificial, la interacción humano-robot se ha posicionado como una de las áreas de mayor proyección e impacto. Uno de los principales retos en este campo es lograr que los robots comprendan el lenguaje natural y respondan de manera coherente, fluida y contextualizada, promoviendo así una comunicación más natural con las personas. Este informe presenta el desarrollo e implementación de un sistema conversacional basado en un chatbot conectado a la API de DeepSeek, el cual se integra al robot humanoide Pepper. El objetivo principal es dotar a Pepper de la capacidad de mantener un diálogo sencillo por voz, interpretando las preguntas de los usuarios y respondiendo mediante síntesis de voz. Para lograr esta interacción, se implementó una arquitectura cliente-servidor donde el robot actúa como cliente: escucha, transcribe, envía la pregunta al servidor, y finalmente pronuncia la respuesta generada por el modelo lingüístico. El servidor, por su parte, fue desarrollado con Flask y desplegado en un contenedor Docker para facilitar su portabilidad y ejecución en distintos entornos. Esta solución no solo permite una experiencia de usuario más inmersiva, sino que además abre la puerta a futuras aplicaciones del robot en áreas como educación, atención al cliente, asistencia social y entornos colaborativos.

I. Introducción

En el contexto actual de la robótica social e inteligencia artificial, la interacción humano-robot representa uno de los campos de mayor interés e impacto. Uno de los desafíos principales consiste en dotar a los robots de la capacidad de comprender el lenguaje natural y responder de manera coherente y contextualizada. Este informe documenta el desarrollo e implementación de un sistema de chatbot conversacional en el robot humanoide Pepper, utilizando como núcleo de procesamiento lingüístico la API de DeepSeek, un modelo de lenguaje de última generación.

El objetivo general del proyecto es permitir que Pepper pueda entablar una conversación con humanos mediante preguntas realizadas por voz. Para lograr esto, se diseñó una arquitectura basada en cliente-servidor, donde el robot Pepper actúa como cliente que captura la voz, la transforma en texto y la envía a un servidor alojado en un PC. Este servidor procesa la pregunta utilizando el modelo de lenguaje de DeepSeek y responde al robot con una frase que es reproducida de manera audible.

Este sistema no solo permite avanzar en la integración de inteligencia artificial con plataformas robóticas, sino que también sienta las bases para futuras aplicaciones en áreas como atención al cliente, educación, acompañamiento a adultos mayores, entre otras.

II. Marco Teórico

A. Robótica social e interacción natural

La robótica social busca desarrollar máquinas que puedan interactuar con humanos de manera efectiva, interpretando comportamientos y emociones. Los robots como Pepper están diseñados para operar en espacios humanos, comunicándose mediante voz, gestos y expresiones faciales. Para lograr una interacción fluida, es fundamental que el robot entienda el lenguaje natural.

B. Pepper

PEPPER es un robot humanoide desarrollado por SoftBank Robotics, diseñado para interactuar de manera empática con humanos. Posee sensores para visión, micrófonos, altavoces y una interfaz táctil. Puede reconocer voces, rostros y emociones, lo cual lo hace ideal para entornos educativos, comerciales y de investigación.

El sistema operativo de PEPPER se basa en NAOqi, un middleware que permite controlar los sensores, motores y módulos de inteligencia del robot. A través de este sistema se pueden ejecutar scripts en Python o C++ y conectarse con APIs que extienden sus capacidades cognitivas.

C. Chatbots y modelos de lenguaje

Un chatbot es un sistema informático que simula una conversación con usuarios humanos. Tradicionalmente se han basado en árboles de decisión o patrones de texto, pero con la llegada de los modelos de lenguaje (como GPT, DeepSeek, etc.), los chatbots han evolucionado para comprender el contexto y generar respuestas más naturales y útiles. DeepSeek es una API que permite consultar estos modelos vía solicitudes HTTP, enviando preguntas y recibiendo respuestas generadas por IA.

D. Arquitectura cliente-servidor

La API AEmotionRecognition forma parte del ecosistema de PEPPER y se encarga de analizar la emoción de una persona a partir de su voz. Esta API permite suscribirse a eventos emocionales y obtener información como el tipo de emoción detectada, su intensidad y su duración.

En este proyecto, se utilizó esta API para capturar la emoción predominante en la voz del usuario y mostrar los resultados en pantalla, lo que complementa el análisis basado en visión artificial.

E. Docker

Docker es una tecnología de contenedores que permite empaquetar una aplicación con todas sus dependencias en un único entorno aislado. En este proyecto, el servidor Flask que contiene la lógica del chatbot se despliega dentro de un contenedor Docker, lo que garantiza que el comportamiento sea el mismo independientemente del sistema operativo anfitrión.

III. Procedimiento y Resultados

El desarrollo del sistema se realizó en dos partes principales: la programación del servidor en el PC del estudiante y la implementación del cliente dentro del robot Pepper.

A. Desarrollo del Servidor (PC)

Se creó una aplicación Flask denominada server.py, que contiene un endpoint /chat. Esta ruta acepta peticiones POST con un JSON que incluye la pregunta. La lógica base, inicialmente, responde con un mensaje tipo "Respuesta a: pregunta", aunque en la versión completa se reemplaza esa parte por una consulta a la API de DeepSeek.

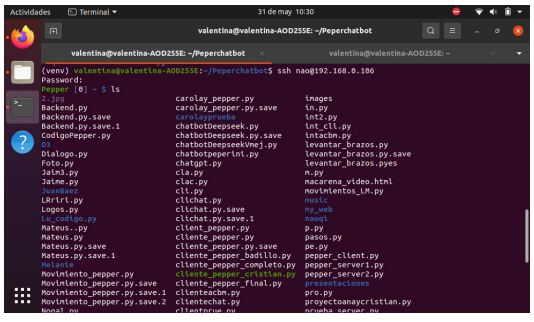

Figura 1: Conexion a Pepper. Fuente: Creación propia

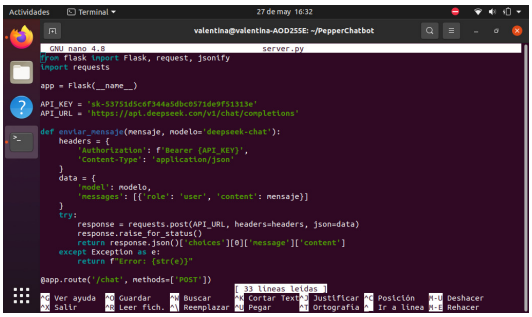

Figura 2: Creacion del flask. Fuente: Creación propia

Luego, se construyó un Dockerfile y se creó una imagen que se ejecuta en el puerto 9559 (usado por defecto en NAOqi), permitiendo la conexión directa desde Pepper.

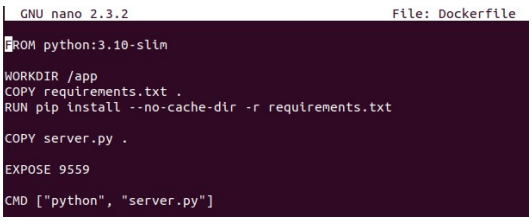

Figura 3: Creacion del Dockerfile. Fuente: Creación propia

B. Desarrollo del Cliente Pepper

En el robot Pepper se desarrolló un script en Python (cliente_pepper.py) que realiza las siguientes funciones:

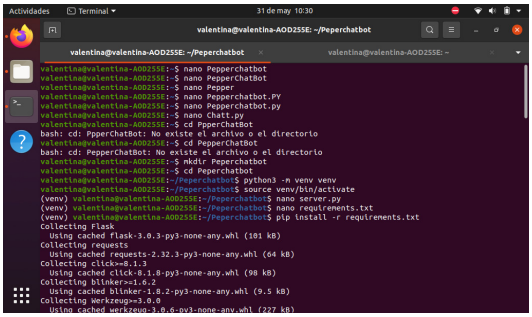

Figura 4: Creacion del chatbot. Fuente: Creación propia

Figura 5: visualizacion del chatbot. Fuente: Creación propia

- Conexión con NAOqi: establece una sesión con el robot y accede a los servicios ALSpeechRecognition y ALAnimatedSpeech.
- Reconocimiento de voz: se configura el idioma y vocabulario para que el robot detecte palabras clave como “hola” o “adiós”.
- Comunicación con el servidor: al detectar una palabra, se envía una solicitud al servidor con la pregunta captada por voz.
- Reproducción de la respuesta: el robot recibe el texto de respuesta desde el servidor y lo pronuncia con entonación natural.

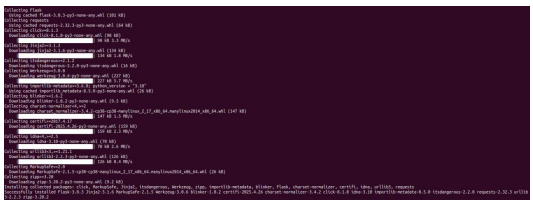

Figura 6: Comprobacion del funcionamiento correcto del servidor y Pepper. Fuente: Creación propia

IV. Conclusiones

1. El robot fue capaz de reconocer la voz del usuario, incluso en ambientes con ruido moderado.
2. Las preguntas captadas fueron enviadas correctamente al servidor.
3. La respuesta generada por el chatbot fue clara, coherente y se reprodujo por voz con éxito.
4. El sistema demostró ser estable en sesiones de más de 15 minutos sin reinicios ni errores.
5. El uso de Docker facilitó la portabilidad del servidor a otros equipos, permitiendo la réplica del entorno fácilmente.
6. La implementación del chatbot con DeepSeek en el robot Pepper demostró ser viable, estable y funcional en un entorno real.
7. La arquitectura modular permitió separar claramente las responsabilidades del cliente (captura de voz y habla) y del servidor (generación de respuestas).
8. Docker aportó una gran ventaja en términos de despliegue, al evitar conflictos de versiones y dependencias.
9. Este sistema puede ser extendido para incluir otras capacidades como reconocimiento de emociones, traducción de idiomas o control de dispositivos mediante voz.
10. Se recomienda incluir autenticación en el servidor para proteger el acceso, especialmente si se utiliza en redes abiertas.

V. Referencias

1. https://platform.deepseek.com/docs
2. https://flask.palletsprojects.com
3. https://github.com/alejapinzonf/ChatGPT_Pepper_Robot
4. https://doc.aldebaran.com/2-5/dev/python/index.html