Simposio CEA de Robótica, Bioingeniería, Visión por Computador y Automática Marina 2026

Control supervisor seguro de robots cuadrúpedos mediante planificadores semánticos LLM

Díaz-Labrador, A.[a,c,*], Arcano-Bea, P.[a], Delgado, A.[c], Pérez-Iglesias, H.[c], Fontenla-Romero, O.[b], Calvo-Rolle, J.[a]

[a] *Universidade da Coruña, CTC, CITIC, Departamento de Ingeniería Industrial, Ferrol, 15471, A Coruña, España.*
[b] *Universidade da Coruña, LIDIA, CITIC, Facultade de Informática, 15471, A Coruña, España.*
[c] *Fundación Instituto Tecnológico de Galicia, A Coruña, España.*

Resumen

El uso de Grandes Modelos de Lenguaje (Large Language Models, LLMs) en robótica facilita la interacción humano-robot, pero otorgarles control de bajo nivel presenta riesgos críticos de seguridad por sus alucinaciones. Este trabajo preliminar presenta una arquitectura de control desacoplada para el robot cuadrúpedo Unitree Go2, donde un LLM (Llama 3) actúa exclusivamente como proponente de planes simbólicos, no como controlador directo. Mediante una interfaz de voz, el sistema procesa instrucciones naturales y genera comandos estructurados en JSON bajo restricciones semánticas estrictas. Estas propuestas son interpretadas y validadas por un filtro de software antes de su ejecución física vía Comunicación Web en Tiempo Real (Web Real-Time Communication, WebRTC). Tras evaluar 1000 inferencias, los resultados muestran una exactitud semántica global del 74,8 % y una robustez estructural del 98,3 %, garantizando una traducción segura de intenciones a comandos cinemáticos predefinidos. Este enfoque sienta las bases para un planificador simbólico seguro y futuras validaciones formales a priori.

Palabras clave: Inteligencia artificial, Robots móviles autónomos, Interfaz hombre-máquina, Arquitecturas de control, Control supervisor, Toma de decisiones.

Safe supervisory control in quadrupedal robots using LLMs as semantic planners

Abstract

The use of Large Language Models (LLMs) in robotics facilitates human-robot interaction, but granting them direct low-level control presents critical safety risks due to hallucinations. This preliminary work presents a decoupled control architecture for the Unitree Go2 quadrupedal robot, where an LLM (Llama 3) acts exclusively as a symbolic plan proposer, not a direct controller. Through a voice interface, the system processes natural instructions and generates JSON-structured commands under strict semantic constraints. These proposals are interpreted and validated by a software filter before physical execution via Web Real-Time Communication (WebRTC). After evaluating 1000 inferences, results show a global semantic accuracy of 74.8 % and a structural robustness of 98.3 %, ensuring the safe translation of human intentions into predefined kinematic commands. This approach lays the groundwork for a secure symbolic planner and future formal a priori validations.

Keywords: Artificial intelligence, Autonomous mobile robots, Human-machine interface, Control architectures, Supervisory control, Decision making.

1. Introducción

Tradicionalmente, la robótica se limitó a entornos industriales estructurados con mínima interacción humana (Engelberger and Devol, 1961). Sin embargo, la llegada de la robótica colaborativa y las fábricas 4.0 inició un desplazamiento hacia una convivencia estrecha en almacenes y líneas de producción (Urrea and Kern, 2025). A pesar de este avance, la interacción entre humanos y robots, también conocida como Interacción Humano-Robot (Human-Robot Interaction, HRI) seguía restringida a personal técnico capaz de operar interfaces complejas (Wang et al., 2026).

Este paradigma está transformándose gracias a dos factores:

*Autor para correspondencia: anabel.dlabrador@udc.es

la democratización de robots móviles profesionales de desarrollo y el auge de los Grandes Modelos de Lenguaje (Large Language Models, LLMs). En particular, la morfología cuadrúpeda ha surgido como una solución muy versátil para entornos no estructurados por su biomecánica inspirada en animales y a los grandes avances en su sistema de control que ha habido en los últimos años. Lo que antes era únicamente tecnología militar o de investigación de muy alto coste, hoy es accesible. Uno de los ejemplos es el Unitree Go2, que es capaz de realizar movimientos dinámicos complejos como acrobacias, mapeado y navegación en diferentes entornos utilizando su kit de desarrollo de software (Software Development Kit, SDK) disponible (Hamrani et al., 2025). El uso de LLMs para controlar este tipo de robots ayuda a mejorar la HRI, que es tan importante. Así, la comunicación y el uso de estos sistemas complejos resultan más naturales, lo que contribuye a democratizar aún más su uso (Wang et al., 2026).

No obstante, la integración de LLMs en la robótica física introduce vulnerabilidades serias. Aunque estos modelos permiten que operadores no expertos utilicen lenguaje natural para comandar sistemas complejos, su naturaleza probabilística y estocástica genera "alucinaciones" o secuencias de comandos impredecibles (Han et al., 2026). Aquí es donde surge la "brecha de encarnación" (del inglés *embodiment gap*), entendida como la disparidad entre la capacidad de procesamiento de un modelo basado en texto y las consecuencias físicas tangibles de sus acciones en el mundo real: a diferencia de un *chatbot* convencional, un error semántico en robótica no resulta en un texto incorrecto, sino en una colisión o daño físico irreversible (Han et al., 2026). La literatura reciente sugiere que aproximaciones monolíticas, como la generación directa de código, en inglés "Code as Policies" (Liang et al., 2023), son vulnerables a inyecciones de prompts y fallos de contexto (Robey et al., 2025).

Para mitigar estos riesgos, es necesario pivotar hacia arquitecturas jerárquicas donde el LLM actúe exclusivamente como un planificador de alto nivel, delegando la ejecución motriz a controladores deterministas de bajo nivel (Ahn et al., 2022).

Investigaciones recientes han comenzado a materializar diversas estrategias de contención y validación. Sistemas como el marco SAFER (Khan et al., 2025) proponen el uso de múltiples LLMs para desacoplar explícitamente la planificación de la tarea de la auditoría de riesgos, actuando como supervisores paralelos que imponen reglas de seguridad generalizadas antes de permitir la actuación. De manera complementaria, enfoques defensivos como SafeEmbodAI (Zhang et al., 2024) han evidenciado que forzar a los modelos a emitir salidas en esquemas estructurados y procesarlos a través de validadores lógicos es una medida necesaria para mitigar comportamientos inseguros y evitar vulnerabilidades de inyección de comandos en robots móviles. A pesar de estos avances, existe un consenso en la literatura reciente sobre la inmadurez de estos sistemas para aplicaciones críticas. Como señalan (Han et al., 2026), una precisión del 99 % es insuficiente en robótica, donde un solo error es catastrófico. Asimismo, marcos como SafeEmbodAI reconocen que la investigación en seguridad física está en sus fases iniciales y limitada mayoritariamente a simulaciones. Por lo tanto, persiste una carencia de arquitecturas que logren un equilibrio entre la seguridad estrictamente determinista y la ejecución en tiempo real en plataformas comerciales como el Unitree Go2.

Bajo este enfoque, el presente artículo propone una arquitectura de control LLM jerárquica con supervisión por software desacoplada aplicada al robot Unitree Go2. La solución propuesta utiliza un sistema de reconocimiento de voz (Speech-to-Text, STT) que genera el prompt de entrada a un LLM como planificador semántico que tiene como salida un prompt estructurado para traducir instrucciones del usuario a formato JSON validado bajo restricciones explícitas. Este flujo garantiza que un controlador intermedio filtre las propuestas semánticas antes de transmitirlas al robot vía WebRTC, asegurando la integridad física del sistema en todo momento.

El resto del artículo se estructura de la siguiente manera: la Sección 2 presenta el caso de estudio basado en la plataforma robótica. La Sección 3 detalla la metodología y el diseño de la arquitectura del planificador. La Sección 4 describe la configuración experimental. La Sección 5 expone los resultados preliminares obtenidos y, finalmente, la Sección 6 presenta las conclusiones y las líneas de trabajo futuro.

2. Caso de estudio

Para el presente trabajo, se ha seleccionado como plataforma base al robot cuadrúpedo Unitree Go2. Esta elección se ha fundamentado sobre todo en su agilidad morfológica, y su SDK de fábrica permite decenas de maniobras complejas.

Con el fin de blindar la integridad del sistema, se ha realizado una revisión de las capacidades del SDK, discriminando el catálogo de movimientos según su riesgo cinemático. El objetivo es establecer un perímetro de seguridad donde el LLM pueda planificar sin comprometer la estabilidad del hardware, aunque ello conlleve ciertas restricciones. Esta clasificación se detalla en la Tabla 1.

Tabla 1: Clasificación de seguridad de las primitivas del Unitree Go2 y selección para el diccionario de acciones.

Nivel de Riesgo	Primitivas (Selección en negrita)
Bajo (Estático y social)	**DAMP**, **BALANCE_STAND**, **STOP_MOVE**, **STAND_UP**, **STAND_DOWN**, **RECOVERY_STAND**, **SIT**, **HELLO**, **STRETCH**, **HEART**, **SCRAPE**, Euler, RiseSit, Pose, Content
Medio (Locomoción y baile)	**DANCE1**, **DANCE2**, **TROT_RUN**, **FREE_WALK**, Move, SpeedLevel, StaticWalk, EconomicGait, ClassicWalk, WalkUpright, CrossStep
Alto (Aéreo y acrobático)	**FRONT_JUMP**, FrontFlip, FrontPounce, LeftFlip, BackFlip, Handstand, FreeBound, FreeJump

En la práctica, nuestro diccionario de acciones prioriza movimientos que fomentan una interacción empática y una navegación segura. No obstante, se ha incluido de forma deliberada la primitiva *FRONT_JUMP* (salto frontal) para cuantificar empíricamente la capacidad del sistema de asimilar, aislar y gestionar comandos potencialmente peligrosos. En definitiva, buscamos transformar el lenguaje natural abstracto en un flujo de ejecución 100 % predecible para la plataforma.

3. Metodología

La arquitectura del sistema se basa en un flujo jerárquico de procesamiento dividido en cuatro fases principales (ver Figura 1): (1) captura de comandos por voz mediante periféricos Bluetooth, (2) transcripción mediante un algoritmo de STT, (3) planificación semántica a través de un LLM y (4) filtrado lógico para el envío de comandos seguros al robot Unitree Go2 vía WebRTC.

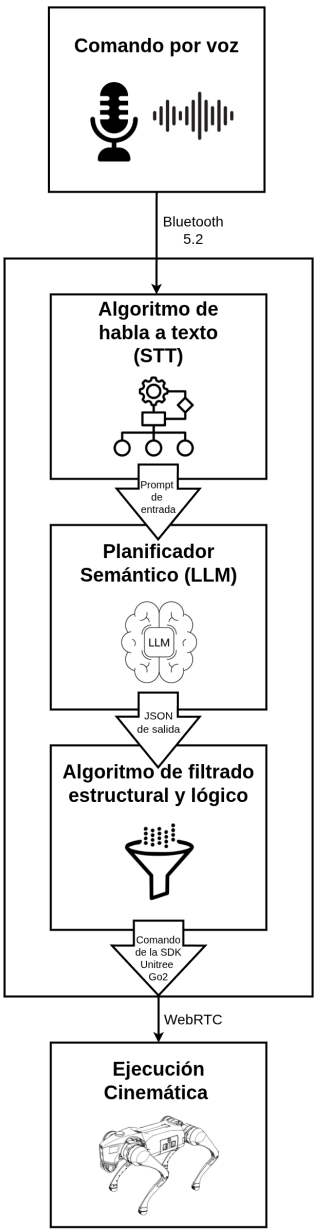

Figura 1: Diagrama de bloques de la arquitectura del sistema.

Para la captura de señales acústicas, se ha implementado un módulo basado en la librería *SpeechRecognition*. El sistema opera en un modo de escucha activa no bloqueante mediante multihilos (*threading*), lo que permite gestionar la interfaz de usuario y la captura de señal simultáneamente. Con el objetivo de aumentar la resiliencia ante entornos ruidosos, el sistema realiza una estimación del ruido ambiental durante 2 segundos antes del inicio de la operación. Para optimizar los recursos computacionales y evitar activaciones espurias, se utiliza un listado de fonemas clave (p. ej., "Jonay"). El orquestador semántico solo se activa cuando el STT identifica uno de estos términos en la cadena de texto inicial.

Como núcleo de decisión se utiliza el modelo Llama 3 (8B), seleccionado por su capacidad de ejecución en dispositivos *edge*. El modelo está configurado mediante un *System Prompt* que define su rol, las funciones disponibles en el SDK del Unitree Go2 y reglas de seguridad inquebrantables.

Se ha implementado una estrategia de razonamiento tipo *Chain-of-Thought* obligando al modelo a generar una estructura JSON con dos campos: *description* y *actions*. El campo *description* fuerza al LLM a justificar lingüísticamente la acción, lo que incrementa la precisión y facilita la detección de alucinaciones. En la Tabla 2 se detallan ejemplos de esta estructura de salida.

Tabla 2: Ejemplos de salidas estructuradas JSON generadas por el planificador semántico.

Acciones directas (comandos predefinidos):

```
{
  "actions": [{"type": "action",
        "value": "HEART"}],
  "description": "El usuario expresa
        afecto, respondiendo con HEART."
}
```

Movimiento cinemático (parámetros de traslación, rotación y tiempo):

```
{
  "actions": [{"type": "move",
        "params": {"x": 0.3, "y": 0,
        "yaw": 0, "duration": 2}}],
  "description": "El usuario pide
        avanzar; se traduce a x=0.3."
}
```

Se valida el JSON recibido en Python utilizando Pydantic, descartando cualquier instrucción que no coincida con el diccionario de acciones permitidas. Asimismo, se aplican saturaciones a los parámetros cinemáticos para asegurar que las velocidades y duraciones no excedan los límites operativos del robot.

Se evaluó el sistema con 100 instrucciones naturales divididas en cinco categorías: Directas (ej. "Siéntate"), Afectivas (ej. "Te quiero"), Indirectas (ej. "Me vendría bien que te sentaras"), Movimiento (ej. "Avanza 3 segundos") e Irrelevantes (ej. "¿Cuál es el sentido de la vida?"). Para esta validación preliminar, cada frase se iteró 10 veces, generando 1000 inferencias. Este es un volumen estadísticamente representativo para identificar modos de fallo sin incurrir en costes computacionales excesivos.

Para cuantificar el rendimiento del sistema propuesto y evaluar la viabilidad del planificador semántico, se han formalizado matemáticamente las cuatro dimensiones de evaluación. Estas métricas están alineadas con la necesidad de garantizar una ejecución determinista y segura frente a la brecha de encar-

nación en entornos físicos (Han et al., 2026; Ahn et al., 2022).

1. **Exactitud Semántica (*Acc*):** Evalúa la capacidad del modelo para traducir correctamente la intención humana al comando esperado, tanto a nivel global como desglosado por categoría. Se define como:

$$Acc(\%) = \left(\frac{1}{N} \sum_{i=1}^{N} A_i \right) \times 100 \tag{1}$$

   donde N es el número total de inferencias evaluadas y $A_i \in \{0, 1\}$ es una variable indicadora que toma el valor 1 si la traducción semántica a cinemática del i-ésimo comando coincide plenamente con la acción esperada, y 0 en caso de error o alucinación.
2. **Eficiencia Temporal:** Se analiza a través de la latencia media ($\bar{L}$) y su dispersión (σ_L), lo que permite entender la estabilidad y predictibilidad del tiempo de procesamiento interno requerido por el modelo:

$$\bar{L} = \frac{1}{N} \sum_{i=1}^{N} l_i \quad ; \quad \sigma_L = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (l_i - \bar{L})^2} \tag{2}$$

   donde l_i es el tiempo de inferencia, expresado en segundos, para cada iteración individual.
3. **Robustez Estructural (E_{est}):** Cuantifica la consistencia sintáctica del modelo al generar las estructuras JSON. Forzar este determinismo en la salida es una medida de seguridad fundamental (Zhang et al., 2024). Esta métrica se calcula como la tasa de errores de validación:

$$E_{est}(\%) = \left(\frac{N_{err}}{N} \right) \times 100 \tag{3}$$

   donde N_{err} es la cantidad de inferencias descartadas por la capa de software intermedia al no superar la validación del esquema lógico o presentar comandos malformados.
4. **Perfil de Fallos (E_{motriz}):** Clasifica y cuantifica la tasa de error en función de la intención motriz de salida k generada en el JSON (clasificadas estructuralmente en tres tipos: *NONE*, *MOVE* y *ACTION*), aislando en qué tipo de razonamiento espacial o abstracto falla el modelo:

$$E_{motriz}(k)(\%) = \left(\frac{F_k}{T_k} \right) \times 100 \tag{4}$$

   donde F_k representa el número de fallos de traducción para la intención específica k, y T_k es el número total de intentos evaluados para dicha intención.

4. Configuración de los experimentos

El objetivo de esta sección es maximizar la replicabilidad del artículo. En él se detalla todo lo necesario para poder realizar los experimentos.

Para garantizar una baja latencia, todos los experimentos se ejecutaron de manera local. El procesamiento *off-board* se llevó a cabo en un equipo externo conectado a la misma red inalámbrica (Wi-Fi) que el robot. Las especificaciones del hardware, software y periféricos utilizados se detallan en la Tabla 3.

Como ya se ha mencionado, la transmisión de las acciones y movimientos filtrados al robot se realiza vía WebRTC a través de la red local. A diferencia de la versión *Edu*, el modelo Unitree Go2 Pro no dispone de forma nativa la SDK abierta para el control de bajo nivel. Para solventar esta limitación técnica y establecer una comunicación bidireccional directa que permita la experimentación, se integró una pasarela de comunicación desarrollada por la comunidad (Legion1581, 2024). Esta implementación habilita el envío de comandos de control y la monitorización de la telemetría del cuadrúpedo de forma independiente a la aplicación comercial propietaria, posibilitando así su integración en arquitecturas de investigación.

Tabla 3: Hardware, Software y Periféricos del Sistema

Componente	**Descripción**
Equipo	ASUS TUF Gaming A15 (FA507NV)
CPU	AMD Ryzen 7 7735HS (8 núcleos/16 hilos, 4,8 GHz)
GPU	NVIDIA RTX 4060 (Dedicada, 8 GB VRAM)
SO	Ubuntu 24.04.4 LTS (Kernel 6.14)
Bluetooth	MediaTek MT7921 (Soporte Bluetooth 5.2)
Audio/Mic.	Razer Barracuda X (Modo inalámbrico)
RAM	16 GB DDR5 @ 5600 MHz
Almacenamiento	512 GB NVMe SSD (Micron)

El planificador semántico se implementó utilizando el modelo Llama 3 (8B), ejecutado localmente mediante la API de Ollama. Se optó por este modelo frente a alternativas más recientes (ej. familia Llama 4) porque su menor tamaño lo hace ideal para despliegues edge locales.

Para alinearse con las estrategias de contención de riesgos descritas previamente, se ajustó la temperatura empíricamente a 0.2, porque valores menores volvían al modelo demasiado rígido ante lenguaje ambiguo, mientras que valores superiores incrementaban la aleatoriedad y el riesgo de alucinaciones inseguras.

Durante las pruebas preliminares de *Prompt Engineering*, se observó que el modelo presentaba ambigüedades al distinguir entre estados posturales similares (por ejemplo, entre *BALANCE_STAND* y *STAND_UP*). Para subsanar esto e imponer el formato JSON estricto requerido por el filtro de software, se endureció el *System Prompt*. Se configuró el sistema íntegramente en español para evitar la degradación del rendimiento por saltos contextuales de idioma.

Para la evaluación cuantitativa, se empleó el conjunto de datos detallado en la Sección 3. Con el objetivo de evaluar la consistencia estadística, la robustez estructural y la dispersión temporal del LLM bajo el enfoque propuesto, cada uno de los 100 comandos se iteró 10 veces. Esto generó un volumen total de 1000 pruebas de inferencia independientes, las cuales fueron analizadas bajo las cuatro dimensiones métricas previamente establecidas.

Dado que el objetivo central de este estudio es analizar la brecha de encarnación y la seguridad en la traducción semántica a cinemática, la evaluación del rendimiento acústico del módulo STT queda fuera del alcance de estos experimentos cuantitativos. Para aislar el rendimiento del LLM y evitar propagar errores de transcripción, se asumió una entrada de texto ideal. No obstante, la viabilidad del flujo completo ininterrumpido (interacción verbal → STT → LLM → filtro → WebRTC → res-

puesta física) ha sido validada cualitativamente. Dicha integración, junto con el material audiovisual demostrativo y el código fuente necesario para su replicación, se encuentra documentada y disponible de forma abierta en el repositorio del proyecto (Díaz-Labrador, 2026).

5. Resultados

Tras 1000 inferencias, Llama 3 (8B) demostró su viabilidad como planificador semántico, obteniendo una exactitud global del 74,8 % (Figura 2). El modelo destaca en comandos directos (92,5 %), pero se degrada ante instrucciones irrelevantes (66,5 %), donde la tasa de fallos por falsos positivos alcanza el 33,5 %. Adicionalmente, configurar el *System Prompt* y las acciones en español frente al inglés elevó la exactitud global en casi un 13 %. Esto confirma que la falta de coherencia lingüística entre las definiciones internas y el idioma de uso genera malinterpretaciones semánticas.

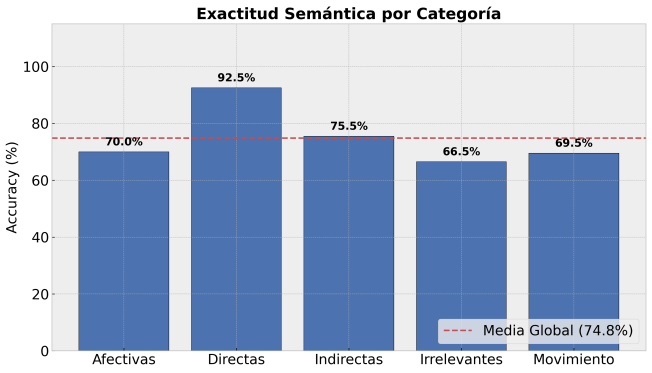

Figura 2: Exactitud semántica desglosada por categoría de comando.

La restricción estructural impuesta en el *System Prompt* fue altamente efectiva (Tabla 4). Con solo 17 errores de validación JSON/Pydantic en 1000 iteraciones (éxito global del 98,3 %), la mayoría de los fallos se limitaron a instrucciones irrelevantes por su ambigüedad intrínseca. Ningún comando malformado superó el filtro hacia WebRTC.

Tabla 4: Frecuencia de errores estructurales (formato JSON) por categoría semántica.

Categoría	Errores (n)	Tasa de Error
Irrelevantes	11	5,5 %
Afectivas	5	2,5 %
Indirectas	1	0,5 %
Directas	0	0,0 %
Movimiento	0	0,0 %
Total Global	**17**	**1,7 %**

El perfil de fallos (Tabla 5) refleja cómo las cinco categorías de voz convergen en tres intenciones motrices: `ACTION` (gestos estáticos), `MOVE` (desplazamientos) y `NONE` (inacción). Contraintuitivamente, la mayor vulnerabilidad (33,5 % de error) reside en gestionar la inacción (`NONE`) ante frases irrelevantes, generando alucinaciones motrices. Esta vulnerabilidad se explica porque existe un sesgo de acción inherente al entrenamiento de los LLMs. Al estar optimizados para ser resolutivos, el modelo tiende matemáticamente a forzar una acción ante estímulos conversacionales vacíos, por lo que al modelo le resulta poco natural optar por no hacer nada. Los comandos `MOVE` presentan un 29,0 % de error al intentar mapear razonamiento espacial abstracto a variables estrictas (ej., $x = 0$, $y = 1$, $yaw = 0$, $dur = 2$). Al examinar el origen de estos fallos se puede encontrar que el modelo confunde sistemáticamente la polaridad de los ejes, invirtiendo las rotaciones y desplazamientos laterales al carecer de una comprensión del marco de referencia absoluto del robot. Finalmente, las primitivas estáticas (`ACTION`) resultaron ser las más robustas, fallando puntualmente por ambigüedad postural (ej., confundir sentarse con agacharse).

Tabla 5: Perfil de fallos agrupado según el tipo de intención motriz de salida generada en el formato JSON.

Tipo de Intención	Intentos	Tasa Error	Foco de Fallo Principal
NONE	200	33,5 %	Falsos positivos
MOVE	210	29,0 %	Variables espaciales
ACTION	590	21,0 %	Ambigüedad postural

En cuanto a eficiencia temporal (Figura 3), los gestos directos mantienen una latencia de inferencia estable (media de 1,66 s), mientras que las secuencias de movimiento paramétricas exigen mayor carga de procesamiento interno (media de 2,13 s). Si bien estas latencias introducen un ligero retraso en la fluidez de la comunicación humano-robot, el desacoplamiento de la arquitectura garantiza la seguridad. Durante este tiempo de cómputo semántico, el controlador de bajo nivel mantiene la estabilidad autónoma del sistema, por lo que la latencia afecta a la experiencia de usuario, pero no a la integridad física.

Finalmente, la validación cualitativa en tiempo real (Figura 4) confirma que la arquitectura propuesta logra capturar, filtrar lógicamente y materializar de forma segura las intenciones verbales sobre la cinemática del robot.

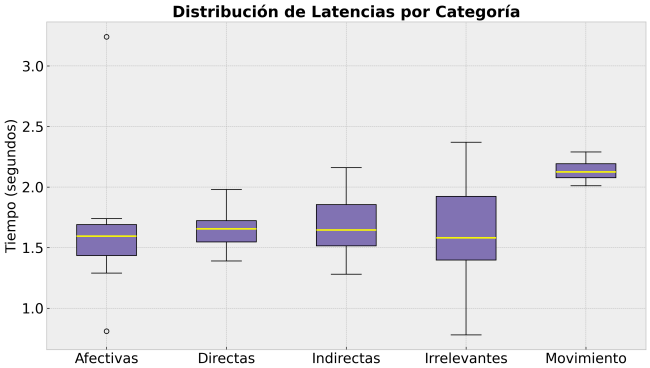

Figura 3: Distribución de latencias temporal por categoría.

6. Conclusiones

Este trabajo preliminar valida una arquitectura robusta para mitigar la brecha de encarnación en la HRI. El uso de Llama 3 (8B) exclusivamente como proponente de acciones simbólicas, con un desacoplamiento estricto del control de bajo nivel, garantiza una traducción segura de intenciones a cinemática. Bajo

el paradigma “el LLM propone, el sistema valida, el robot ejecuta”, el trabajo futuro expandirá esta base hacia un planificador autónomo.

Figura 4: Demostración en vivo del sistema (Díaz-Labrador, 2026).

El primer paso será generar secuencias completas mediante una capa de supervisión que simule los planes antes de su ejecución física, vetando trayectorias inseguras. Paralelamente, se implementará un protocolo de metacognición en lazo cerrado (*closed-loop*) donde el modelo razone sobre sus propios errores de validación y regenere planes corregidos de forma autónoma.

Asimismo, se dotará al planificador de conciencia situacional mediante fusión sensorial. El LLM recibirá retroalimentación bidireccional (postura, equilibrio, telemetría) del Unitree Go2, un requisito indispensable para autorizar acciones dinámicas de riesgo bajo verificación múltiple.

Finalmente, la arquitectura evolucionará desde el cómputo *off-board* actual hacia un despliegue *edge* embebido. Para viabilizar el sistema computacionalmente, el trabajo futuro evaluará la ejecución de modelos más compactos (3B-4B parámetros) y arquitecturas de cuantización extrema a 1,58 bits (Ma et al., 2024). Estos avances consolidarán un marco donde los LLMs operen como supervisores de alto nivel, maximizando la eficiencia de inferencia sin comprometer la seguridad física.

Agradecimientos

La investigación de Anabel Díaz Labrador ha sido financiada por la Xunta de Galicia, a través de las ayudas para doctorados industriales (http://gain.xunta.gal/), bajo la ayuda “Doutoramentos Industriais 2024” con referencia: 02_IN606D_2024_3100897.

Paula Arcano-Bea’s research was supported by the Xunta de Galicia (Regional Government of Galicia) through grants to Ph.D. (http://gain.xunta.gal), under the “Axudas á etapa predoutoral” grant with reference: ED481A-2025-088.

El CITIC, como Centro de Investigación del Sistema Universitario de Galicia, está financiado por la Consellería de Educación, Universidade e Formación Profesional de la Xunta de Galicia a través del Fondo Europeo de Desarrollo Regional (FEDER) y la Secretaría Xeral de Universidades (Ref. ED431G 2019/01).

Xunta de Galicia. Ayudas para la consolidación y estructuración de unidades de investigación competitivas, Grupos de Potencial Crecimiento (GPC) (ED431B 2023/49).

Referencias

Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Ho, D., Hsu, J., Ibarz, J., Ichter, B., Irpan, A., Jang, E., Ruano, R. J., Jeffrey, K., Jesmonth, S., Joshi, N., Julian, R., Kalashnikov, D., Kuang, Y., Lee, K.-H., Levine, S., Lu, Y., Luu, L., Parada, C., Pastor, P., Quiambao, J., Rao, K., Rettinghouse, J., Reyes, D., Sermanet, P., Sievers, N., Tan, C., Toshev, A., Vanhoucke, V., Xia, F., Xiao, T., Xu, P., Xu, S., Yan, M., Zeng, A., 2022. Do as i can and not as i say: Grounding language in robotic affordances. In: arXiv preprint arXiv:2204.01691.

Díaz-Labrador, A., 2026. LLM-WebRTC-Unitree-Go2: Implementation and Experimental Demonstration. `https://github.com/anabeldilab/LLM-WebRTC-Unitree-Go2`, accedido: 2026-03-01. Incluye demostración en vídeo y código fuente.

Engelberger, J., Devol, G., 1961. Installation of the first Unimate industrial robot. General Motors Ternstedt Plant, Trenton, N.J., first industrial robot application.
URL: `https://www.thehenryford.org/collections-and-research/digital-collections/artifact/183434/`

Hamrani, A., Rayhan, M. M., Mackenson, T., McDaniel, D., Lagos, L., 2025. Smart quadruped robotics: a systematic review of design, control, sensing and perception. Advanced Robotics 39 (1), 3–29.
DOI: `10.1080/01691864.2024.2411684`

Han, J., Seo, J., Min, J., Kim, J., Oh, J., 2026. Safety not found (404): Hidden risks of llm-based robotics decision making.
DOI: `arXiv.2601.05529`

Khan, A. A., Andrev, M., Murtaza, M. A., Aguilera, S., Zhang, R., Ding, J., Hutchinson, S., Anwar, A., 2025. Safety aware task planning via large language models in robotics. In: 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 21024–21031.
DOI: `10.1109/IROS60139.2025.11246041`

Legion1581, 2024. unitree_webrtc_connect: Webrtc interface for unitree robots. `https://github.com/legion1581/unitree_webrtc_connect`, accedido: 2026-02-28.

Liang, J., Huang, W., Xia, F., Xu, P., Hausman, K., Ichter, B., Florence, P., Zeng, A., 2023. Code as policies: Language model programs for embodied control. In: 2023 IEEE International Conference on Robotics and Automation (ICRA). pp. 9493–9500.
DOI: `10.1109/ICRA48891.2023.10160591`

Ma, S., Wang, H., Ma, L., Wang, L., Wang, W., Huang, S., Dong, L., Wang, R., Xue, J., Wei, F., 2024. The era of 1-bit llms: All large language models are in 1.58 bits.

Robey, A., Ravichandran, Z., Kumar, V., Hassani, H., Pappas, G. J., 2025. Jailbreaking llm-controlled robots. In: 2025 IEEE International Conference on Robotics and Automation (ICRA). pp. 11948–11956.
DOI: `10.1109/ICRA55743.2025.11128119`

Urrea, C., Kern, J., 2025. Recent advances and challenges in industrial robotics: A systematic review of technological trends and emerging applications. Processes 13 (3).
DOI: `10.3390/pr13030832`

Wang, Y., Xu, Y., Nikolova, A., Wang, Y., Wang, J., Wang, C., Tong, X., 2026. How do we research human-robot interaction in the age of large language models? a systematic review. In: Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems. CHI ’26. Association for Computing Machinery.
DOI: `10.48550/arXiv.2602.15063`

Zhang, W., Kong, X., Braunl, T., Hong, J. B., 2024. Safeembodai: a safety framework for mobile robots in embodied ai systems.
URL: `https://arxiv.org/abs/2409.01630`