

Planeación de acciones usando un sistema basado en reglas para robots de servicio doméstico

Luis Sergio Cano Olguin

Tutor: Dr. Jesús Savage Carmona

Agradecimientos

Resumen

Índice general

1. Introducción	1
1.1. Contexto y Motivación	2
1.2. Problemática: Sistemas Basados en Reglas en Entornos Dinámicos . . .	3
1.3. Hipótesis	3
1.4. Objetivos	4
1.5. Estructura del documento	5
2. Antecedentes y Estado del Arte	7
2.1. Sistemas Basados en Reglas: Historia y Aplicaciones	7
2.2. El Lenguaje CLIPS: Características, Ventajas y Limitaciones	8
2.2.1. Características Fundamentales de CLIPS	9
2.2.2. Ventajas de CLIPS en Robótica	10
2.2.3. Limitaciones de CLIPS en Entornos Dinámicos	10
2.3. Enfoques Modernos: Aprendizaje Automático (ML) y Redes Neuronales en Robótica	11
2.3.1. Principales Paradigmas de Aprendizaje Automático en Robótica	11
2.3.2. Aplicaciones Específicas en Robótica de Servicio	12
2.3.3. Limitaciones de los Enfoques Puramente Basados en ML	13
2.4. Sistemas Híbridos: Combinando la Predictibilidad de las Reglas con la Flexibilidad del ML	14
2.4.1. Definición y Fundamentos de los Sistemas Híbridos	14
2.4.2. Tipologías de Integración Híbrida	14
2.4.3. Ejemplos de Sistemas Híbridos en Robótica	15
2.4.4. Ventajas y Desafíos de los Sistemas Híbridos	16
2.4.5. Conexión con la Propuesta de Tesis	17

3. Marco Teórico y Conceptual	18
3.1. Arquitectura de un Sistema de Planeación de Acciones Jerárquico . . .	19
3.2. Fundamentos de los Sistemas de Producción (Rule-Based Systems) . .	20
3.3. Introducción a los Modelos de Lenguaje Grande (LLMs) y ChatGPT/Qwen	20
3.4. Integración de Traductores de Lenguaje Natural con Sistemas Basados en Reglas	21
3.4.1. Utilidad y Función de la Integración	21
3.4.2. Funcionamiento de la Integración CLIPS-Lenguaje Natural . . .	21
3.4.3. Descripción de Tecnologías: CLIPS, ChatGPT y Qwen2.5-0.5B .	21
3.5. ChatGPT como Agente de Planificación Autónomo	21
3.5.1. Capacidades de Planificación de LLMs vs. Sistemas Basados en Reglas	21
3.5.2. Arquitectura de Planificación Dual: CLIPS y ChatGPT como Mo- tores Complementarios	21
3.5.3. Mecanismos de Selección y Validación de Planes	21
3.5.4. Ventajas y Riesgos de la Planificación con LLMs en Robótica . .	21
4. Metodología: Diseño del Sistema Híbrido CLIPS-ChatGPT/Qwen	22
4.1. Diseño del Sistema Basado en Reglas con CLIPS	22
4.1.1. Definición del Conjunto de Reglas Jerárquicas	22
4.1.2. Priorización de Acciones Críticas y Gestión de Emergencias . . .	22
4.1.3. Representación del Conocimiento y Hechos en la Base de Datos de CLIPS	22
4.2. Integración de ChatGPT/Qwen como Sistema Complementario	23
4.2.1. Funciones Asignadas al Módulo de Lenguaje Natural	23
4.2.2. Protocolo de Comunicación entre CLIPS y los Modelos de Lenguaje	23
4.2.3. Mecanismos de Seguridad y Validación para las Respuestas del LLM	23
4.3. Implementación de Planificación Dual	23
4.3.1. Diseño del Módulo de Planificación con ChatGPT	23
4.3.2. Protocolos de Prompting para Generación de Planes Robóticos .	23
4.3.3. Mecanismo de Selección entre Planes CLIPS vs. ChatGPT . . .	23
4.3.4. Validación y Simulación de Planes Antes de Ejecución	23

5. Implementación	25
5.1. Entornos de Desarrollo: Simulación y Plataforma Física	25
5.2. Implementación del Motor de Reglas en CLIPS	25
5.3. Desarrollo del Módulo de Integración	25
5.4. Configuración de la Interfaz con API de OpenAI para ChatGPT o Modelo Qwen Local	25
5.5. Casos de Prueba para Validar la Interacción entre los Módulos	25
6. Escenarios de Validación y Experimentación	26
6.1. Diseño de Experimentos en Entornos Controlados	26
6.1.1. Escenario 1: Ejecución de Tareas Predefinidas (solo CLIPS)	26
6.1.2. Escenario 2: Gestión de Órdenes Imprecisas o Novedosas (CLIPS + ChatGPT/Qwen)	26
6.1.3. Escenario 3: Respuesta a Eventos Inesperados o Fallos	26
6.1.4. Escenario 4: Planificación Dual para Tareas Complejas (CLIPS vs. ChatGPT)	26
6.2. Métricas de Evaluación	27
6.2.1. Métricas Cuantitativas: Tiempo de Ejecución, Tasa de Éxito, Uso de Recursos Computacionales	27
6.2.2. Métricas Cualitativas: Robustez, Interpretabilidad y Fluidez en la Interacción Humano-Robot	27
7. Análisis de Resultados	28
7.1. Comparativa del Rendimiento del Sistema Solo-CLIPS vs. el Sistema Híbrido	28
7.2. Evaluación de la Efectividad de ChatGPT/Qwen en las Diferentes Funciones Asignadas	28
7.3. Discusión de Limitaciones y Errores	28
7.4. Análisis de la Escalabilidad y el Coste Computacional de la Integración	28
8. Discusión	29
8.1. Interpretación de los Resultados en el Contexto de los Objetivos	29
8.2. Ventajas y Desventajas del Enfoque Híbrido Propuesto	29
8.3. Implicaciones para la Seguridad y Certificación en Entornos Regulados	29
8.4. Límites Éticos y Prácticos del Uso de ChatGPT/Qwen en Robótica	29

9. Contribuciones y Relevancia	30
9.1. Contribución Técnica: Marco Híbrido Escalable y Documentado	30
9.2. Relevancia Práctica: Aplicaciones en Salud, Industria y Logística	30
9.3. Relevancia Académica: Benchmark para Sistemas Rule-Based vs. Híbridos	30
10. Conclusiones y Trabajo Futuro	31
10.1. Conclusiones Principales	31
10.2. Propuestas de Trabajo Futuro	31

Índice de figuras

3.1. Arquitectura ViRoot.	19
-----------------------------------	----

Capítulo 1

Introducción

Los robots de servicio doméstico representan un campo de proyección para la inteligencia artificial y la automatización, con aplicaciones que van desde la asistencia a personas mayores hasta la gestión autónoma de entornos residenciales (1). En este contexto, la planificación de acciones, es decir, la capacidad de un robot para descomponer objetivos de alto nivel en secuencias ejecutables, es un desafío en entornos dinámicos y no estructurados donde interactúan humanos, objetos móviles y tareas imprevistas (2).

Los sistemas basados en reglas, como los implementados en motores de inferencia del tipo CLIPS, han demostrado ser robustos y predecibles en escenarios estructurados, como líneas de producción o laboratorios controlados (3). Su fortaleza radica en la transparencia del razonamiento simbólico y la capacidad de priorizar acciones críticas, como la evasión de obstáculos o la gestión de emergencias. Sin embargo, su rigidez los hace poco adaptables ante variaciones no previstas en el entorno o ante comandos expresados en lenguaje natural con alto grado de ambigüedad o complejidad (4).

Recientemente, los modelos de lenguaje grande (LLMs), como ChatGPT, han emergido como herramientas capaces de interpretar y generar lenguaje natural, e incluso de actuar como agentes de planificación autónomos (5). Su flexibilidad contextual permite traducir instrucciones verbales en secuencias de acciones, complementando así la solidez de los sistemas basados en reglas. No obstante, su naturaleza probabilística y la falta de garantías formales de seguridad limitan su uso directo en aplicaciones robóticas donde la integridad física y la predictibilidad son prioritarias (6).

Esta tesis propone un **sistema híbrido de planificación de tareas** que integra un motor de reglas CLIPS con modelos de lenguaje natural del tipo ChatGPT, aprovechando las ventajas de ambos enfoques: la previsibilidad y seguridad de CLIPS, en

conjunto con la adaptabilidad y capacidad de comprensión lingüística de los LLMs. El sistema se implementó y validó en el robot de servicio doméstico *Justina*, desarrollado en el Laboratorio de Bio-Robótica de la UNAM, utilizando ROS 2 como marco de integración.

1.1. Contexto y Motivación

El área de la robótica orientada al servicio doméstico ha evolucionado significativamente en la última década, impulsada por avances en percepción, planificación y interacción humano-robot (1). Estos sistemas se han diseñado para operar en entornos no estructurados como hogares, hospitales o centros de cuidado, donde deben realizar tareas que van desde la entrega de objetos hasta la asistencia en actividades de la vida diaria. En este contexto, la capacidad de planificar acciones de manera autónoma, segura y adaptativa se convierte en un requisito fundamental.

La planificación de acciones en robótica ha sido tradicionalmente abordada mediante sistemas basados en reglas (*Rule-Based Systems, RBS*), que ofrecen un marco predecible y verificable para la toma de decisiones (2). Sin embargo, la creciente complejidad de los entornos domésticos—caracterizados por la presencia de humanos, objetos dinámicos y situaciones imprevistas—ha expuesto las limitaciones de los enfoques puramente simbólicos. Paralelamente, el reciente surgimiento de modelos de lenguaje grande (**LLMs**) ha abierto nuevas posibilidades para la interpretación de comandos en lenguaje natural y la generación de planes flexibles, aunque a menudo carentes de garantías formales de seguridad (6).

Esta tesis se desarrolla en el marco del Laboratorio de Bio-Robótica de la UNAM, utilizando como plataforma el robot de servicio *Justina*, un sistema modular basado en ROS 2 con capacidades avanzadas de navegación, manipulación y percepción. La motivación central del trabajo radica en la necesidad de desarrollar arquitecturas de planificación que combinen lo mejor de ambos mundos: la robustez y transparencia de los sistemas basados en reglas, y la adaptabilidad y capacidad de diálogo de los modelos de lenguaje modernos.

1.2. Problemática: Sistemas Basados en Reglas en Entornos Dinámicos

Los sistemas basados en reglas, implementados en motores de inferencia como CLIPS, se fundamentan en la lógica simbólica y el ciclo *reconocer-actuar* (3). Su principal ventaja es la previsibilidad: ante un conjunto de hechos y reglas bien definidos, la respuesta del sistema es determinista y explicable. Esto los hace idóneos para aplicaciones donde la seguridad y la certificación son críticas, como en entornos industriales o médicos.

No obstante, en contextos domésticos dinámicos, estos sistemas enfrentan desafíos significativos (4):

1. **Rigidez interpretativa:** No pueden procesar comandos en lenguaje natural complejo o ambiguo, como "trae el libro rojo que está cerca de la ventana".
2. **Falta de adaptabilidad:** Las reglas deben ser definidas a priori; cualquier situación no prevista en la base de conocimiento puede llevar al fracaso o a un comportamiento no deseado.
3. **Dificultad para gestionar la incertidumbre:** No manejan bien la información incompleta o cambiante del entorno en tiempo real.
4. **Escalabilidad limitada:** Mantener y extender un conjunto grande de reglas para cubrir todos los escenarios posibles resulta complejo y laborioso.

Por otro lado, los enfoques puramente basados en aprendizaje automático — especialmente los LLMs—ofrecen flexibilidad y capacidad de generalización, pero introducen riesgos como (6):

- Comportamientos impredecibles o no verificables.
- Falta de garantías de seguridad en la ejecución física.
- Dependencia de grandes volúmenes de datos y recursos computacionales.

La problemática central, por tanto, es cómo diseñar un sistema de planificación que mantenga la seguridad y explicabilidad de los RBS, pero que sea lo suficientemente flexible para operar en entornos domésticos dinámicos e interactuar de manera natural con humanos.

1.3. Hipótesis

La hipótesis central de esta investigación propone la integración sinérgica de un sistema basado en reglas (CLIPS) con un modelo de lenguaje grande (ChatGPT o

Qwen) puede superar las limitaciones de cada enfoque por separado, resultando en un sistema híbrido de planificación que es a la vez seguro, explicable, adaptable y capaz de entender lenguaje natural.

Esta complementariedad se articula en tres niveles:

1. **Nivel de interpretación:** Los LLMs actúan como traductores de lenguaje natural a hechos estructurados, permitiendo que comandos complejos y ambiguos sean convertidos en representaciones simbólicas que CLIPS puede procesar.
2. **Nivel de planificación:** Se implementa una arquitectura de planificación dual, donde CLIPS genera planes predecibles y seguros para tareas conocidas, mientras que los LLMs proponen soluciones flexibles para situaciones novedosas o complejas.
3. **Nivel de validación y selección:** Un mecanismo de supervisión evalúa los planes generados por ambos motores con base en criterios de seguridad, eficiencia y contexto, seleccionando la mejor opción o combinándolas de manera segura.

Se espera que este sistema híbrido:

- Mejore la tasa de éxito en la ejecución de tareas en entornos domésticos.
- Reduzca el tiempo de respuesta ante comandos complejos.
- Mantenga un comportamiento seguro y predecible en situaciones críticas.
- Permita una interacción más natural e intuitiva con usuarios no expertos.

La validación de esta hipótesis se realizará mediante experimentos en simulación y con el robot físico Justina, comparando el desempeño del sistema híbrido frente a enfoques puramente basados en reglas o puramente basados en LLMs.

1.4. Objetivos

- **Diseño del Sistema Híbrido:**
 - Definir un conjunto jerárquico de reglas en CLIPS para priorizar acciones críticas.
 - Especificar la arquitectura de integración entre CLIPS y modelos de lenguaje natural.
 - Diseñar un marco de planificación dual que permita tanto a CLIPS como a ChatGPT generar y ejecutar planes.
 - Establecer protocolos de comunicación y validación entre los módulos.
- **Implementación Técnica:**

- Integrar el motor de reglas CLIPS con ROS (Robot Operating System).
- Desarrollar el módulo de interfaz con APIs de ChatGPT o implementación local de Qwen2.5-0.5B.
- Implementar capacidades de planificación autónoma en ChatGPT mediante prompting estructurado.
- Desarrollar un mecanismo de selección y validación entre planes generados por diferentes motores.
- **Capacidades de Interacción Natural:**
 - Implementar traducción de comandos en lenguaje natural a hechos estructurados de CLIPS.
 - Desarrollar mecanismos de diálogo para resolución de ambigüedades.
 - Crear protocolos de retroalimentación para aprendizaje incremental.
- **Validación Experimental:**
 - Validar el sistema en escenarios realistas con el robot físico y simulaciones.
 - Evaluar el rendimiento mediante métricas comparativas:
 - Tiempo de ejecución de tareas y tasa de éxito.
 - Precisión en interpretación de comandos complejos.
 - Robustez ante instrucciones ambiguas o novedosas.
 - Comparar el sistema híbrido vs. abordajes puramente basados en reglas.
 - Evaluar la calidad de planes generados por CLIPS vs. ChatGPT en diferentes escenarios.

1.5. Estructura del documento

Este documento está organizado en los siguientes capítulos que describen el sistema propuesto para el desarrollo de esta tesis, abarcando el planteamiento, desarrollo, implementación y validación del sistema híbrido de planificación. La estructura es la siguiente:

1. **Capítulo 1: Introducción.** Presenta el contexto, motivación, problemática, hipótesis y objetivos de la investigación, así como la justificación del enfoque híbrido en robótica de servicio doméstico.
2. **Capítulo 2: Antecedentes y estado del arte.** Revisa la evolución de los sistemas basados en reglas en robótica, las características de CLIPS, los enfoques modernos de aprendizaje automático y los sistemas híbridos existentes.

3. **Capítulo 3: Marco teórico y conceptual.** Describe los fundamentos de la planificación jerárquica, los sistemas de producción, los modelos de lenguaje grande (LLMs) y la arquitectura de integración propuesta.
4. **Capítulo 4: Metodología.** Detalla el diseño del sistema híbrido, incluyendo la definición de reglas en CLIPS, la integración con ChatGPT/Qwen, y los mecanismos de planificación dual y validación.
5. **Capítulo 5: Implementación.** Explica los entornos de desarrollo, la configuración técnica, la interfaz con APIs y las pruebas utilizados para validar la integración de los módulos.
6. **Capítulo 6: Escenarios de validación y experimentación.** Presenta el diseño experimental, los escenarios de prueba y las métricas cuantitativas y cualitativas utilizadas para evaluar el sistema.
7. **Capítulo 7: Análisis de resultados.** Compara el rendimiento del sistema híbrido frente al sistema basado solo en reglas, evalúa la efectividad de los LLMs y discute limitaciones y costos computacionales.
8. **Capítulo 8: Discusión.** Interpreta los resultados en relación con los objetivos, analiza ventajas y desventajas del enfoque híbrido, y explora implicaciones éticas y de seguridad.
9. **Capítulo 9: Contribuciones y relevancia.** Destaca la aportación técnica del marco híbrido, su aplicabilidad práctica en distintos sectores y su relevancia académica como benchmark.
10. **Capítulo 10: Conclusiones y trabajo futuro.** Resume los hallazgos principales y propone líneas de investigación futura, como el fine-tuning de LLMs y la mejora de mecanismos de seguridad.

Capítulo 2

Antecedentes y Estado del Arte

Se realiza una revisión comprensiva de la evolución histórica de los sistemas basados en reglas en el área de la robótica, analizando aplicaciones tradicionales y limitaciones actuales. También se examinan las características técnicas de CLIPS y se contrastan con enfoques modernos de aprendizaje automático, culminando con el estudio de sistemas híbridos que combinan la predictibilidad de las reglas con la flexibilidad del (*Machine Learning, ML*) para entornos dinámicos.

2.1. Sistemas Basados en Reglas: Historia y Aplicaciones

Los sistemas basados en reglas (*Rule-Based Systems, RBS*), constituyen uno de los paradigmas más antiguos y consolidados en inteligencia artificial (3). Su origen se remonta a los sistemas expertos de la década de 1970, donde se utilizaban para representar el conocimiento de especialistas humanos en dominios bien delimitados, como el diagnóstico médico o la configuración de sistemas complejos (2). En robótica, la adopción de estos sistemas se popularizó en los años 80 y 90, particularmente en entornos industriales estructurados, donde la previsibilidad y seguridad eran prioritarias.

Históricamente, los RBS han sido fundamentales en arquitecturas de control robótico jerárquico, como la propuesta por Brooks (4), donde capas de comportamientos reactivos podían ser coordinadas mediante reglas de supresión. Más adelante, motores de inferencia como CLIPS (*C Language Integrated Production System*), desarrollado por la NASA en 1985, se convirtieron en herramientas estándar para la implementación

de sistemas de planificación y toma de decisiones en robots autónomos (11). CLIPS permitió la representación simbólica del conocimiento a través de hechos, reglas, en conjunto con un ciclo de inferencia *reconocer-actuar* que ofreció un marco eficiente para el razonamiento en tiempo real.

En cuanto a aplicaciones, los sistemas basados en reglas han demostrado su utilidad en diversos dominios robóticos:

1. **Robótica industrial:** En cadenas de montaje automatizadas, donde las tareas son repetitivas y el entorno está controlado. Los RBS se utilizan para secuenciar operaciones, gestionar excepciones y garantizar la seguridad de los operarios humanos (1).
2. **Robótica de servicio y doméstica:** En tareas simples como la entrega de objetos, navegación en interiores y asistencia a personas con movilidad reducida.
3. **Robótica espacial y de exploración:** Sistemas como Remote Agent de la NASA emplearon arquitecturas basadas en reglas para la planificación y ejecución autónoma de experimentos en misiones no tripuladas (12).
4. **Sistemas híbridos contemporáneos:** Recientemente, los RBS se han integrado con técnicas de aprendizaje automático para compensar sus limitaciones en entornos dinámicos. Por ejemplo, en *Robotic Pick-and-Place with Natural Language Commands* (9) se combina un RBS con un modelo de lenguaje natural para interpretar comandos verbales en tareas de manipulación.

A pesar de la relevancia de enfoques basados en aprendizaje profundo, los sistemas basados en reglas mantienen su relevancia en aplicaciones donde la explicabilidad, verificabilidad y seguridad son críticas (6). Su capacidad para representar conocimiento simbólico de forma transparente los hace ideales para entornos regulados o donde se requiere auditoría de las decisiones del robot.

2.2. El Lenguaje CLIPS: Características, Ventajas y Limitaciones

CLIPS es un lenguaje de programación basado en reglas, diseñado específicamente para la construcción de sistemas expertos y aplicaciones basadas en conocimiento (11). Concebido originalmente como una alternativa a los costosos sistemas implementados en LISP y en estaciones de trabajo especializadas, CLIPS destacó por su portabilidad, eficiencia y su capacidad para ejecutarse en hardware de bajo costo (3). Su adopción se

extendió rápidamente en la comunidad de robótica e inteligencia artificial, convirtiéndose en una herramienta de referencia para la implementación de sistemas de producción (2).

2.2.1. Características Fundamentales de CLIPS

CLIPS se fundamenta en el paradigma de los sistemas de producción, operando mediante su ciclo de inferencia que permite razonar sobre una base de hechos utilizando reglas del tipo if-then (3). Entre sus características más distintivas se encuentran:

1. **Encadenamiento hacia adelante (*forward chaining*):** CLIPS utiliza predominantemente encadenamiento hacia adelante, lo que significa que, partiendo de un conjunto de hechos iniciales, el motor de inferencia aplica reglas para derivar nuevos hechos de manera iterativa (11). Este enfoque es especialmente adecuado para aplicaciones en tiempo real donde los cambios en el entorno deben reflejarse inmediatamente en la base de conocimiento (1).
2. **Algoritmo RETE para emparejamiento de patrones:** CLIPS implementa una versión optimizada del algoritmo RETE, que permite un emparejamiento eficiente de patrones entre reglas y hechos, incluso cuando la base de conocimiento contiene miles de elementos (3). Esta característica es crítica en robótica, donde la velocidad de respuesta es determinante (4).
3. **Representación del conocimiento multifacética:** Además de reglas, CLIPS permite representar conocimiento mediante hechos ordenados y plantillas (*def-template*), así como mediante funciones definidas por el usuario en C (11). Esto facilita la integración con sistemas de percepción y control de bajo nivel (8).
4. **Portabilidad y código abierto:** Escrito en C, CLIPS puede ejecutarse en prácticamente cualquier plataforma, desde microcontroladores hasta sistemas de alto rendimiento (3). Su naturaleza open source ha permitido su adaptación en múltiples contextos robóticos, como el control de robots de exploración y sistemas de inspección (11).
5. **Integración con otros lenguajes:** CLIPS ofrece una API en C y una librería en python, lo que le permite ser empleado en aplicaciones más grandes, posibilitado su integración con frameworks robóticos como ROS y entornos de simulación (1).

2.2.2. Ventajas de CLIPS en Robótica

La adopción de CLIPS en robótica de servicio y sistemas autónomos se justifica por varias ventajas clave:

- **Explicabilidad y transparencia:** Las decisiones tomadas por un sistema CLIPS pueden ser rastreadas y explicadas mediante la inspección de las reglas activadas y los hechos utilizados (3). Esto es fundamental en aplicaciones donde la seguridad y la auditoría son prioritarias, como en robótica médica o espacial (6).
- **Previsibilidad del comportamiento:** Dado que el conjunto de reglas es definido explícitamente, el comportamiento del sistema es determinista y verificable formalmente (2). Esta característica contrasta con los enfoques de deep learning, donde las decisiones pueden ser opacas y difíciles de validar (6).
- **Bajo consumo de recursos computacionales:** CLIPS puede ejecutarse en hardware con capacidades limitadas, como robots móviles embarcados, sin necesidad de GPUs o grandes cantidades de memoria (1). Esto lo hace adecuado para plataformas robóticas con restricciones energéticas (8).
- **Facilidad de mantenimiento y actualización:** La base de conocimiento puede ser modificada o extendida sin necesidad de recompilar el sistema completo, lo que facilita la adaptación a nuevos entornos o requisitos (3).

2.2.3. Limitaciones de CLIPS en Entornos Dinámicos

A pesar de sus fortalezas, CLIPS presenta limitaciones significativas cuando se aplica en entornos domésticos dinámicos y no estructurados:

1. **Rigidez semántica:** CLIPS no puede interpretar directamente comandos en lenguaje natural complejo o ambiguo; requiere que la entrada sea previamente estructurada en hechos formales (9). Esto limita su capacidad para interactuar con usuarios no expertos de manera intuitiva (6).
2. **Falta de aprendizaje automático:** CLIPS es un sistema simbólico puro; no posee mecanismos intrínsecos para aprender de la experiencia o adaptar sus reglas automáticamente ante situaciones novedosas (2). Cualquier modificación debe ser realizada manualmente por un programador.
3. **Escalabilidad del conocimiento:** Mantener una base de reglas que cubra todas las posibles situaciones en un hogar resulta complejo y propenso a inconsistencias (4). A medida que crece el número de reglas, la gestión y depuración se vuelven

más difíciles (3).

4. **Manejo limitado de la incertidumbre:** CLIPS no incorpora mecanismos nativos para representar información incierta o probabilística, lo que dificulta su uso en tareas de percepción donde los sensores proporcionan datos ruidosos (1).
5. **Integración compleja con subsistemas modernos:** Aunque existen iniciativas como el **ROS 2 CLIPS-Executive** (8), la integración de CLIPS con pipelines de percepción basados en aprendizaje profundo requiere desarrollos ad hoc que no siempre son triviales (9).

Estas limitaciones han motivado la exploración de arquitecturas híbridas que combinan la robustez y transparencia de CLIPS con la flexibilidad y capacidad de adaptación de los modelos de lenguaje grande, como se discutirá en secciones posteriores de esta tesis (6).

2.3. Enfoques Modernos: Aprendizaje Automático (ML) y Redes Neuronales en Robótica

El desarrollo del aprendizaje automático, y en particular del aprendizaje profundo, ha transformado radicalmente la robótica contemporánea, habilitando capacidades que eran difíciles de lograr con enfoques puramente simbólicos o geométricos. A diferencia de los sistemas basados en reglas, que requieren la especificación explícita de todo el conocimiento del dominio, los enfoques de ML permiten a los robots aprender directamente de los datos, extrayendo patrones complejos y adaptándose a entornos variables sin necesidad de programación explícita.

2.3.1. Principales Paradigmas de Aprendizaje Automático en Robótica

Aprendizaje Supervisado y Visión por Computadora: Las redes neuronales convolucionales (CNNs) se han convertido en el estándar de facto para tareas de percepción robótica, incluyendo detección y reconocimiento de objetos, segmentación semántica y estimación de poses . Arquitecturas como YOLO (You Only Look Once) y sus variantes permiten a robots como Justina identificar objetos en tiempo real con alta precisión, incluso en entornos domésticos desordenados . Las CNNs operan mediante capas de convolución que extraen jerárquicamente características visuales, desde bordes

simples hasta formas complejas, permitiendo una representación robusta del entorno .

Aprendizaje por Refuerzo para Control y Navegación: El aprendizaje por refuerzo profundo (Deep RL) ha emergido como una herramienta poderosa para desarrollar políticas de control en tareas secuenciales. Mediante la interacción con el entorno, los agentes aprenden a maximizar recompensas acumulativas, descubriendo estrategias óptimas para navegación, manipulación y evitación de obstáculos . En navegación robótica, se han desarrollado políticas híbridas que combinan representaciones aprendidas con modelos geométricos tradicionales para operar robustamente en entornos reales .

Redes Recurrentes y Modelado Secuencial: Las redes neuronales recurrentes (RNNs) y sus variantes (LSTM, GRU) son fundamentales para tareas que requieren memoria temporal, como el seguimiento de objetos en movimiento, la predicción de trayectorias humanas o la interpretación de comandos verbales en contexto . Estas arquitecturas mantienen un estado interno que evoluciona con la secuencia de entradas, permitiendo modelar dependencias temporales de largo alcance.

Modelos de Lenguaje Grande (LLMs) en Robótica: La irrupción de modelos transformadores, como los descritos por Vaswani et al. , ha dado lugar a los Large Language Models (LLMs), capaces de procesar y generar lenguaje natural con una fluidez sin precedentes. En robótica, estos modelos se utilizan para interpretar comandos complejos, traducir lenguaje natural a representaciones simbólicas, e incluso generar planes completos a partir de descripciones de alto nivel (6). Su capacidad para captar matices semánticos y contextuales los hace especialmente valiosos para la interacción humano-robot en entornos domésticos.

2.3.2. Aplicaciones Específicas en Robótica de Servicio

- **Percepción y Reconocimiento:** La integración de CNNs en pipelines de percepción permite a los robots identificar objetos, personas y gestos con alta precisión. Sistemas como YOLO y FastSAM (Segment Anything Model) se utilizan para segmentación en tiempo real y localización de objetos de interés .
- **Navegación Autónoma:** Los enfoques basados en aprendizaje profundo han demostrado superioridad frente a métodos geométricos tradicionales en entornos desconocidos, utilizando entradas como mapas locales de obstáculos (LOM) para generar comandos de velocidad directamente . La combinación de SLAM visual con redes neuronales permite una localización robusta incluso en escenarios

2.3. ENFOQUES MODERNOS: APRENDIZAJE AUTOMÁTICO (ML) Y REDES NEURONALES EN

dinámicos .

- **Manipulación y Control:** En robótica de manipulación, las redes neuronales se emplean para estimar propiedades físicas de objetos (rigidez, forma) a partir de señales propioceptivas, permitiendo agarres adaptativos y seguros . En robots paralelos y cables, el ML se utiliza para modelar cinemática compleja y compensar errores geométricos .
- **Planificación y Toma de Decisiones:** Los algoritmos de planificación de rutas y decisiones en vehículos autónomos combinan enfoques basados en conocimiento (reglas, teoría de juegos) con métodos basados en datos (imitación, RL) para lograr sistemas robustos y adaptativos .

2.3.3. Limitaciones de los Enfoques Puramente Basados en ML

A pesar de sus ventajas, los enfoques puramente basados en ML presentan limitaciones significativas:

1. **Falta de garantías de seguridad:** Las redes neuronales pueden producir comportamientos impredecibles ante entradas fuera de la distribución de entrenamiento, lo que es crítico en aplicaciones donde la seguridad física es prioritaria (6).
2. **Opacidad y falta de explicabilidad:** La naturaleza de “caja negra” de las redes profundas dificulta la interpretación de sus decisiones, limitando su adopción en entornos regulados donde se requiere auditoría [citation:Russell2020].
3. **Dependencia de datos:** Los modelos de ML requieren grandes volúmenes de datos etiquetados o experiencias de interacción, cuyo costo de adquisición en robótica puede ser prohibitivo .
4. **Dificultad para incorporar conocimiento previo:** A diferencia de los sistemas basados en reglas, las redes neuronales no pueden aprovechar fácilmente el conocimiento simbólico existente sobre el dominio [citation:Brooks1986].

Estas limitaciones han motivado el desarrollo de arquitecturas híbridas que combinen lo mejor de ambos paradigmas.

2.4. Sistemas Híbridos: Combinando la Predictibilidad de las Reglas con la Flexibilidad del ML

Los sistemas híbridos en robótica representan una tercera vía que busca integrar sinérgicamente enfoques simbólicos (basados en reglas) y subsimbólicos (basados en aprendizaje), aprovechando las fortalezas de cada paradigma mientras se compensan sus debilidades respectivas .

2.4.1. Definición y Fundamentos de los Sistemas Híbridos

Un sistema híbrido de planificación y control se define como aquella arquitectura que integra explícitamente componentes basados en reglas (deterministas, explicables, verificables) con componentes basados en aprendizaje (adaptativos, flexibles, basados en datos) . El objetivo no es sustituir un enfoque por otro, sino establecer mecanismos de coordinación que permitan a cada módulo operar en el ámbito donde es más competente.

La motivación fundamental de los sistemas híbridos radica en la complementariedad de ambos paradigmas:

- Los sistemas basados en reglas aportan predictibilidad, seguridad, explicabilidad y la capacidad de incorporar conocimiento experto explícito.
- Los enfoques basados en ML aportan adaptabilidad, capacidad de generalización a situaciones novedosas, y la habilidad de procesar entradas complejas y ruidosas (como imágenes o lenguaje natural).

2.4.2. Tipologías de Integración Híbrida

En la literatura se identifican varios patrones arquitectónicos para la integración híbrida :

1. **Arquitecturas de Dominio de Reglas con Complemento ML (Rule First + ML):** En este enfoque, el sistema basado en reglas constituye el núcleo principal de toma de decisiones, y los componentes de ML se utilizan para extender sus capacidades en áreas específicas. Por ejemplo, un sistema CLIPS puede manejar la lógica de alto nivel de una tarea, mientras que una CNN se encarga de la detección de objetos necesaria para instanciar los hechos. Este es el caso de la integración de CLIPS con traductores de lenguaje natural en el robot Justina.

2. **Arquitecturas de Dominio ML con Restricciones de Reglas (Model First + Rules):** Aquí, un modelo de ML (como una política de RL o un LLM) genera las acciones principales, pero sus salidas son validadas y, si es necesario, corregidas por un conjunto de reglas de seguridad que actúan como “caja de seguridad”. Esto es común en navegación autónoma, donde el planificador basado en aprendizaje puede sugerir trayectorias, pero un módulo de reglas garantiza que se eviten obstáculos y se respeten límites de velocidad.
3. **Arquitecturas de Fusión o Scheduling Paralelo:** En este esquema, tanto el sistema basado en reglas como el modelo de ML generan acciones concurrentemente, y un módulo supervisor (scheduler) decide qué acción ejecutar basándose en el contexto, la confianza de cada módulo o criterios predefinidos. Este enfoque permite aprovechar la redundancia y seleccionar la mejor estrategia para cada situación.
4. **Arquitecturas de Aprendizaje para Adaptación de Reglas:** En sistemas más avanzados, los componentes de ML se utilizan para adaptar o ajustar dinámicamente los parámetros del sistema basado en reglas. Por ejemplo, un meta-aprendizaje basado en Soft Actor-Critic (SAC) puede ajustar en línea los factores de escala de un controlador PID difuso, mejorando el rendimiento sin perder la estabilidad del lazo determinista de alto frecuencia.

2.4.3. Ejemplos de Sistemas Híbridos en Robótica

- **Navegación Híbrida:** La tesis de Sadek propone agentes autónomos que combinan técnicas geométricas tradicionales (modelado explícito del entorno) con políticas aprendidas mediante redes neuronales, demostrando que los enfoques híbridos pueden operar robustamente tanto en simulación como en entornos físicos reales.
- **Análisis Semántico para Interacción Humano-Robot:** Se han desarrollado enfoques híbridos de análisis semántico que combinan técnicas basadas en reglas con aprendizaje automático para interpretar comandos de lenguaje natural en asistentes robóticos, logrando una comprensión más robusta y contextual.
- **Control de Robots Industriales:** La combinación de controladores PID difusos (basados en 49 reglas) con meta-optimización mediante aprendizaje por refuerzo (SAC) ha demostrado reducciones significativas en error de seguimiento (46-52
- **Planificación Híbrida (Hybrid Planning):** El paradigma de Hybrid Planning integra explícitamente sistemas de reglas (árboles de tareas, máquinas de

estado finito) con módulos de predicción basados en modelos (MPC, LLMs) para lograr comportamientos robustos en entornos complejos. Este enfoque se aplica en robótica de almacenes, interacción humano-robot y vehículos autónomos.

- **Modelado Físico con Aprendizaje:** En robótica de robots blandos y paralelos, se están explorando enfoques que combinan modelos físicos con aprendizaje automático (hybrid learning physics modeling) para mejorar la precisión del modelado cinemático y dinámico, aprovechando el conocimiento previo de la estructura del sistema mientras se aprenden compensaciones de datos.

2.4.4. Ventajas y Desafíos de los Sistemas Híbridos

Ventajas:

- **Robustez mejorada:** Al combinar múltiples fuentes de conocimiento, el sistema puede fallar de manera controlada y recurrir a estrategias alternativas.
- **Seguridad preservada:** Las reglas pueden actuar como cortafuegos que previenen comportamientos inseguros generados por modelos de ML.
- **Explicabilidad selectiva:** Las decisiones críticas pueden ser explicadas mediante las reglas que las generaron, mientras que las decisiones rutinarias pueden delegarse a modelos opacos.
- **Adaptabilidad sin sacrificar estabilidad:** Los componentes de ML permiten adaptación a contextos cambiantes, mientras que las reglas mantienen un comportamiento base estable y verificado.

Desafíos:

- **Complejidad de integración:** La coordinación entre módulos heterogéneos requiere diseños arquitectónicos cuidadosos y protocolos de comunicación eficientes.
- **Resolución de conflictos:** Es necesario definir mecanismos claros para cuando las recomendaciones de reglas y modelos entren en conflicto.
- **Validación y verificación:** Los sistemas híbridos heredan la dificultad de verificar componentes de ML, aunque la presencia de reglas puede facilitar la validación del comportamiento global.
- **Sobrecarga computacional:** Ejecutar múltiples motores de razonamiento concurrentemente puede aumentar los requisitos de cómputo a bordo.

2.4.5. Conexión con la Propuesta de Tesis

La arquitectura híbrida CLIPS-ChatGPT propuesta en esta tesis se enmarca precisamente en esta categoría de sistemas, siguiendo un patrón de dominio de reglas con complemento ML para la traducción de lenguaje natural, y extendiéndose hacia una arquitectura de planificación dual que permite tanto a CLIPS como a los LLMs generar planes concurrentemente (6). Esta aproximación busca capitalizar la predictibilidad y seguridad de CLIPS para tareas críticas y bien definidas, mientras que aprovecha la flexibilidad semántica y capacidad de generación contextual de ChatGPT para interpretar comandos complejos y adaptarse a situaciones novedosas. La validación experimental de esta hipótesis de complementariedad constituye el núcleo de la contribución de esta investigación.

Capítulo 3

Marco Teórico y Conceptual

Se presenta el fundamento teórico del sistema propuesto, describiendo arquitecturas de planeación jerárquica y los principios de los sistemas de producción. Se introduce el funcionamiento de los modelos de lenguaje grande y se detalla la integración de traductores de lenguaje natural con sistemas basados en reglas, estableciendo las bases tecnológicas para la implementación del sistema híbrido CLIPS-ChatGPT/Qwen y su uso en la arquitectura ViRoot descrita a continuación (ver Figura 3.1).

Arquitectura del Sistema

La arquitectura VIRBOT de Justina se organiza en cuatro capas principales:

- **Capa de Entradas**
 - Procesamiento de datos de sensores internos y externos.
 - Aplicación de técnicas de reconocimiento de patrones.
 - Generación del estado del entorno.
- **Capa de Planificación**
 - Validación mediante gestión del conocimiento.
 - Reconocimiento de situaciones y activación de objetivos.
 - Planificación de secuencias de operaciones físicas.
- **Capa de Gestión del Conocimiento**
 - Mapas del entorno creados con técnicas SLAM.
 - Sistema de localización basado en filtro de Kalman.
 - Sistema basado en reglas CLIPS para representación del conocimiento.
- **Capa de Ejecución**
 - Ejecución y verificación de planes de movimiento.

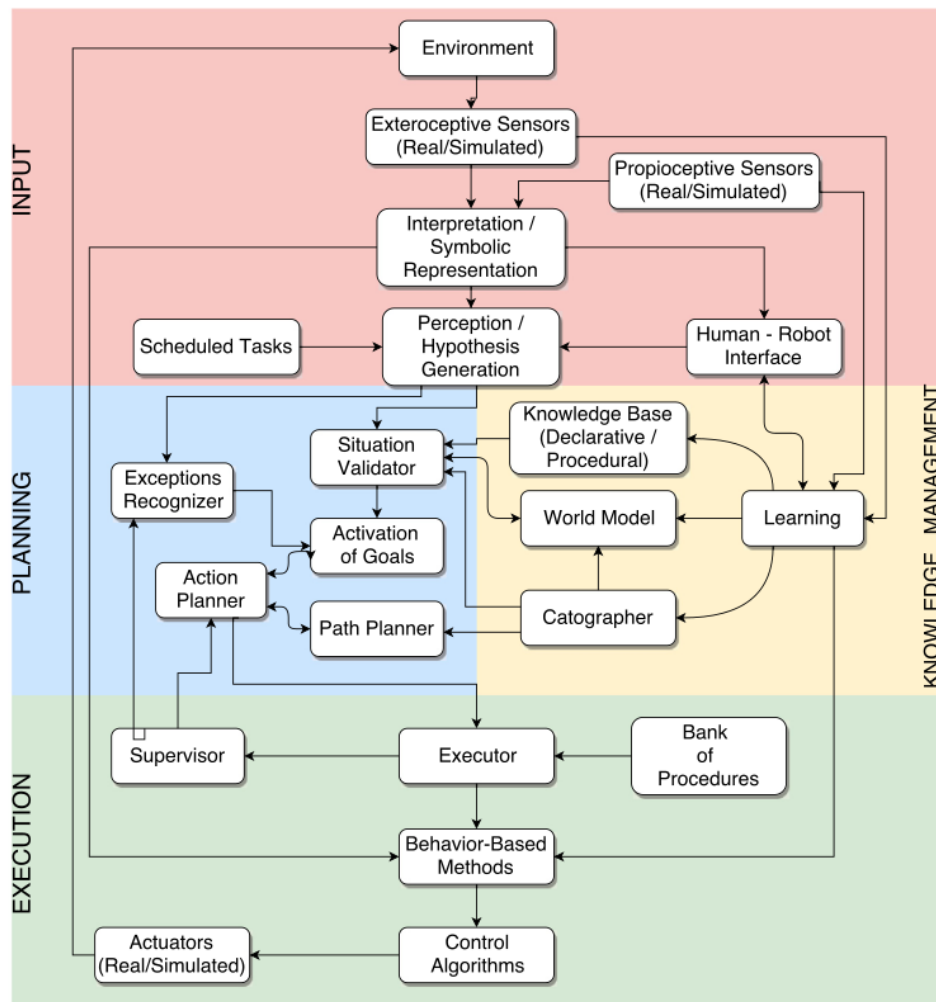


Figura 3.1: Arquitectura ViRoot.

- Procedimientos predefinidos representados como máquinas de estado.
- Integración de procedimientos para generar planes complejos.

3.1. Arquitectura de un Sistema de Planeación de Acciones Jerárquico

Descripción del diseño escalonado del sistema de planificación, donde las decisiones de alto nivel (estrategias) se descomponen en acciones ejecutables de bajo nivel, por ejemplo ("servir la cena") se descomponen en tareas intermedias ("ir a la cocina", "tomar platos", "colocar en mesa") y finalmente en acciones primitivas ejecutables por el robot.

3.2. Fundamentos de los Sistemas de Producción (Rule-Based Systems)

Análisis teórico de los sistemas basados en reglas, centrándose en el ciclo de inferencia reconocer-actuar el algoritmo RETE para emparejamiento eficiente de patrones. Se detalla cómo CLIPS implementa estos conceptos mediante su motor de inferencia y memoria de trabajo.

3.3. Introducción a los Modelos de Lenguaje Grande (LLMs) y ChatGPT/Qwen

Exposición de los principios arquitectónicos de los transformadores y el mecanismo de atención que fundamenta los LLMs. Se contrastan las capacidades de ChatGPT (modelo propietario) frente a Qwen2.5-0.5B (modelo open-source), destacando ventajas computacionales y de personalización para entornos robóticos.

3.4. Integración de Traductores de Lenguaje Natural con Sistemas Basados en Reglas

3.4.1. Utilidad y Función de la Integración

3.4.2. Funcionamiento de la Integración CLIPS-Lenguaje Natural

3.4.3. Descripción de Tecnologías: CLIPS, ChatGPT y Qwen2.5-0.5B

3.5. ChatGPT como Agente de Planificación Autónomo

3.5.1. Capacidades de Planificación de LLMs vs. Sistemas Basados en Reglas

Análisis comparativo de los enfoques de planificación, destacando la rigidez predecible de CLIPS frente a la adaptabilidad contextual de los LLMs.

3.5.2. Arquitectura de Planificación Dual: CLIPS y ChatGPT como Motores Complementarios

3.5.3. Mecanismos de Selección y Validación de Planes

3.5.4. Ventajas y Riesgos de la Planificación con LLMs en Robótica

Capítulo 4

Metodología: Diseño del Sistema Híbrido CLIPS-ChatGPT/Qwen

4.1. Diseño del Sistema Basado en Reglas con CLIPS

4.1.1. Definición del Conjunto de Reglas Jerárquicas

Especificación de las reglas organizadas por niveles de prioridad, donde las reglas de seguridad (evitación de colisiones, gestión de emergencias) tienen máxima prioridad sobre las operativas.

4.1.2. Priorización de Acciones Críticas y Gestión de Emergencias

Diseño de reglas específicas para escenarios de fallo y situaciones críticas, asegurando respuestas predecibles y seguras.

4.1.3. Representación del Conocimiento y Hechos en la Base de Datos de CLIPS

Estructuración del conocimiento del entorno doméstico en hechos CLIPS iniciales que sirven como base para el razonamiento.

Descripción de la base de conocimiento inicial incluye:

- *Topología del entorno*: habitaciones, conexiones, zonas navegables.
- *Taxonomía de objetos*: categorías, subcategorías, propiedades heredables.
- *Relaciones espaciales*: contención, adyacencia, orientación.
- *Capacidades del robot*: acciones posibles, restricciones físicas.

4.2. Integración de ChatGPT/Qwen como Sistema Complementario

4.2.1. Funciones Asignadas al Módulo de Lenguaje Natural

4.2.2. Protocolo de Comunicación entre CLIPS y los Modelos de Lenguaje

4.2.3. Mecanismos de Seguridad y Validación para las Respuestas del LLM

Implementación de verificaciones para asegurar que los planes generados por LLMs cumplen con constraints de seguridad y factibilidad robótica.

4.3. Implementación de Planificación Dual

4.3.1. Diseño del Módulo de Planificación con ChatGPT

4.3.2. Protocolos de Prompting para Generación de Planes Robóticos

4.3.3. Mecanismo de Selección entre Planes CLIPS vs. ChatGPT

4.3.4. Validación y Simulación de Planes Antes de Ejecución

Integración con el Robot Justina

- *Procesamiento de Entrada de Voz*.
- *Ejecución de Planes en el Entorno Físico*.

- Mecanismos para transformar los planes publicados en los tópicos de resultados en comandos ejecutables por los actuadores del robot (navegación, manipulación de objetos, etc.)
- *Retroalimentación y Aprendizaje Incremental.*

Capítulo 5

Implementación

- 5.1. Entornos de Desarrollo: Simulación y Plataforma Física
- 5.2. Implementación del Motor de Reglas en CLIPS
- 5.3. Desarrollo del Módulo de Integración
- 5.4. Configuración de la Interfaz con API de OpenAI para ChatGPT o Modelo Qwen Local
- 5.5. Casos de Prueba para Validar la Interacción entre los Módulos

Capítulo 6

Escenarios de Validación y Experimentación

6.1. Diseño de Experimentos en Entornos Controlados

6.1.1. Escenario 1: Ejecución de Tareas Predefinidas (solo CLIPS)

6.1.2. Escenario 2: Gestión de Órdenes Imprecisas o Novedosas (CLIPS + ChatGPT/Qwen)

6.1.3. Escenario 3: Respuesta a Eventos Inesperados o Fallos

6.1.4. Escenario 4: Planificación Dual para Tareas Complejas (CLIPS vs. ChatGPT)

- Comparativa de eficiencia en generación de planes
- Evaluación de robustez ante escenarios novedosos
- Análisis de seguridad en planes generados por LLMs

6.2. Métricas de Evaluación

6.2.1. Métricas Cuantitativas: Tiempo de Ejecución, Tasa de Éxito, Uso de Recursos Computacionales

6.2.2. Métricas Cualitativas: Robustez, Interpretabilidad y Fluidez en la Interacción Humano-Robot

Capítulo 7

Análisis de Resultados

- 7.1. Comparativa del Rendimiento del Sistema Solo-CLIPS vs. el Sistema Híbrido
- 7.2. Evaluación de la Efectividad de ChatGPT/Qwen en las Diferentes Funciones Asignadas
- 7.3. Discusión de Limitaciones y Errores
- 7.4. Análisis de la Escalabilidad y el Coste Computacional de la Integración

Capítulo 8

Discusión

- 8.1. Interpretación de los Resultados en el Contexto de los Objetivos
- 8.2. Ventajas y Desventajas del Enfoque Híbrido Propuesto
- 8.3. Implicaciones para la Seguridad y Certificación en Entornos Regulados
- 8.4. Límites Éticos y Prácticos del Uso de ChatGPT/Qwen en Robótica

Capítulo 9

Contribuciones y Relevancia

- 9.1. Contribución Técnica: Marco Híbrido Escalable y Documentado
- 9.2. Relevancia Práctica: Aplicaciones en Salud, Industria y Logística
- 9.3. Relevancia Académica: Benchmark para Sistemas Rule-Based vs. Híbridos

Capítulo 10

Conclusiones y Trabajo Futuro

10.1. Conclusiones Principales

10.2. Propuestas de Trabajo Futuro

- Fine-tuning de un LLM de código abierto para dominio específico
- Mejora de los mecanismos de seguridad y verificación
- Exploración de arquitecturas de integración más profundas

Referencias

- [1] R. R. Murphy, *Introduction to AI Robotics*. MIT Press, 2nd ed., 2019.
- [2] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson, 4th ed., 2020.
- [3] J. C. Giarratano and G. D. Riley, *Expert Systems: Principles and Programming*. Thomson Course Technology, 4th ed., 2005.
- [4] R. A. Brooks, “A robust layered control system for a mobile robot,” *IEEE Journal of Robotics and Automation*, vol. 2, no. 1, pp. 14–23, 1986.
- [5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [6] L. Shi, X. Chen, Y. Hu, J. Li, H. Zhang, and H. Zhao, “Chatgpt for robotics: Design principles and model abilities,” *Microsoft Research*, 2023. Preprint.
- [7] NASA, *CLIPS Reference Manual*. NASA, 2023. Disponible en línea.
- [8] J. Savage and otros, “Virbot: A modular architecture for service robots with hybrid planning,” in *RoboCup Symposium*, 2024.
- [9] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, and J. Lee, “Robotic pick-and-place with natural language commands,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 4615–4622, 2020.
- [10] L. P. Kaelbling and T. Lozano-Pérez, “Hierarchical task and motion planning in the now,” in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1470–1477, 2011.
- [11] NASA Johnson Space Center, “Clips reference manual,” tech. rep., NASA, 1991.

- [12] B. Pell *et al.*, “An autonomous spacecraft agent prototype,” in *First International Conference on Autonomous Agents*, pp. 253–261, 1997.