Evaluation of Multi-Agent Reinforcement Learning for cooperative transport tasks in Flexible Manufacturing Systems
Manuel Vázquez, Carolina Saavedra Sueldo, Luis O. Avila, Gerardo Acosta and Mariano De Paula
SADIO Electronic Journal of Informatics and Operations Research (EJS) Vol. 25 No. 1 (2026) e-ISSN 1514-6774
https://revistas.unlp.edu.ar/ejs | https://doi.org/10.24215/15146774e089

Evaluation of Multi-Agent Reinforcement Learning for cooperative transport tasks in Flexible Manufacturing Systems

Manuel Ezequías Vázquez[1,2], Carolina Saavedra Sueldo[1,2], Luis O. Ávila[3], Gerardo G. Acosta[1,2], and Mariano De Paula[1,2]

[1] Centro de Investigaciones en Física e Ingeniería del Centro, UNCPBA-CICpBA-CONICET, B7400JWI, Olavarría, Buenos Aires, Argentina
{manuel.vazquez,ggacosta,mariano.depaula,carolina.saavedra}@fio.unicen.edu.ar
[2] Universidad Nacional del Centro de la Provincia de Buenos Aires (UNCPBA), Facultad de Ingeniería, INTELYMEC, B7400JWI, Olavarría, Buenos Aires, Argentina
[3] Laboratorio de Sistemas Inteligentes, CONICET-UNSL, D5730EKQ, San Luis, Argentina
loavila@unsl.edu.ar

Abstract. Advances in artificial intelligence and Multi-Agent Systems enable coordinated agents to achieve multiple, often conflicting, objectives—making them ideal for "flexible factories." These factories, driven by technologies merging physical, digital, and biological domains, are evolving into "smart factories." Modeling production processes as multi-agent systems allows simultaneous optimization of efficiency, waste reduction, sustainability (economic, social, and environmental), cost savings, and downtime reduction. However, the flexibility needed in reconfigurable environments increases the complexity of decentralized control. Small and medium-sized enterprises (SMEs) are a key example, as they often produce small batches or customized goods, requiring constant adaptation. Multi-agent reinforcement learning provides a viable solution, avoiding impractical centralized control in dynamic settings. This work explores multi-agent reinforcement learning for collaborative manufacturing tasks, such as material handling (a non-value-adding operation where efficiency is critical). A preliminary case study is presented, using virtual environments to train multiple agents in coordinated material manipulation across varying complexity scenarios.

Keywords: Process optimization, Intelligent control, Collaborative manufacturing, Smart factories

Received August 2025; Accepted November 2025; Published February 2026

Evaluación del Aprendizaje por Refuerzo Multiagente para tareas de transporte cooperativo en Sistemas de Fabricación Flexible

Resumen Los avances en inteligencia artificial y Sistemas Multi-Agente permiten coordinar agentes para cumplir múltiples objetivos, incluso contrapuestos, aplicables en "fábricas flexibles". Estas, impulsadas por tecnologías que integran lo físico, digital y biológico, evolucionan hacia "fábricas inteligentes". Modelar un proceso productivo como un sistema multi-agente permite optimizar simultáneamente la eficiencia, reducción de desperdicios, sustentabilidad (económica, social y ambiental), ahorro de costos y reducción de tiempos de inactividad. Sin embargo, la flexibilidad requerida en entornos reconfigurables incrementa la complejidad del control descentralizado. Las pequeñas y medianas empresas (PyMEs) son un caso emblemático, ya que suelen producir lotes pequeños o bienes personalizados, lo que exige una adaptación constante. El aprendizaje por refuerzo multi-agente surge como una solución viable, evitando esquemas centralizados poco prácticos ante entornos cambiantes. Este trabajo analiza dicho enfoque para tareas colaborativas en manufactura, como la manipulación de materiales (una operación sin valor agregado donde la eficiencia es clave). Se presenta un caso de estudio preliminar que utiliza entornos virtuales para entrenar múltiples agentes en tareas de manipulación coordinada en escenarios de diversa complejidad.

Palabras clave: Optimización de procesos, Control inteligente, Manufactura colaborativa, Fábricas inteligentes

1 Introducción

La creciente demanda de productos personalizados y la reducción de sus ciclos de vida están impulsando la transformación de los sistemas de producción tradicionales hacia modelos de fabricación más flexibles y adaptativos. Este escenario plantea un desafío significativo para las pequeñas y medianas empresas (PyMEs), que deben adaptarse rápidamente a cambios en los procesos productivos para responder a las necesidades del mercado. En este contexto, las denominadas fábricas flexibles buscan incorporar tecnologías que les permitan reconfigurar sus procesos de manera ágil y eficiente, manteniendo al mismo tiempo altos estándares de productividad y sustentabilidad (Chen. et al., 2018).

Uno de los principales desafíos en estos entornos es la gestión coordinada de los recursos físicos, como robots móviles y manipuladores, que deben colaborar en la ejecución de tareas como el transporte y la manipulación de materiales (Saavedra Sueldo et al., 2024). El auge de plataformas robóticas cada vez más asequibles —desde brazos montados sobre bases móviles hasta drones manipuladores— habilita la automatización de estas operaciones que, aunque no agregan valor directo al producto, resultan esenciales para la eficiencia global de la planta.

El paradigma de los Sistemas Multi-Agente (SMA), junto con los avances en Inteligencia Artificial (IA) y Aprendizaje por Refuerzo (RL), ofrece una alternativa prometedora para abordar la complejidad de estas tareas de manera descentralizada (Albrecht et al., 2024). Los enfoques basados en Aprendizaje por Refuerzo Multiagente (MARL) permiten que cada agente tome decisiones de forma autónoma, adaptándose dinámicamente a entornos cambiantes y colaborando para lograr objetivos comunes.

En este trabajo, proponemos una contribución mediante el diseño y evaluación de un escenario de manufactura colaborativa utilizando simuladores físicos de alta fidelidad. Específicamente, analizamos la emergencia de comportamientos colaborativos en tareas donde múltiples agentes deben coordinar la manipulación conjunta de objetos bajo distintos niveles de complejidad, una capacidad fundamental para los procesos adaptativos requeridos por las PyMEs modernas.

2 Estado del arte

El desarrollo de propuestas basadas en el paradigma de sistemas multiagentes en entornos industriales es un área que ha retomado el interés de los investigadores para explorar diversas metodologías y herramientas para mejorar la eficiencia y dotar con capacidades de auto-adaptabilidad a los procesos productivos. En particular, el RL es una de las metodologías más difundidas, del campo de la inteligencia artificial, para el desarrollo de propuestas de sistemas autónomos de toma de decisiones en escenarios dinámicos y complejos.

El RL ha sido explorado como una aproximación promisoria para optimizar procesos en sistemas de manufactura flexible. (Li et al., 2022) presentan un sistema basado en RL para la planificación automática de movimientos en una línea de ensamblaje, utilizando un gemelo digital para el entrenamiento y monitoreo de la producción. Los resultados demuestran que los agentes entrenados con RL pueden mejorar la tasa de éxito en ensamblajes sin colisiones.

Otro enfoque es el presentado por (Schwung et al., 2018), donde se propone una arquitectura de RL para unidades de manufactura altamente flexibles. En este caso, los agentes no aprenden directamente el control de los procesos, sino que ajustan parámetros específicos del sistema de control, permitiendo una rápida adaptación a distintos escenarios de producción. Este tipo de enfoques refuerzan la idea de que la combinación de RL con estructuras de control tradicionales puede mejorar la eficiencia y adaptabilidad de sistemas industriales.

Adicionalmente, se ha explorado la integración de algoritmos de aprendizaje automático en sistemas robóticos multiagente para la planificación y control de operaciones productivas y logísticas. Estudios recientes han analizado diversas aplicaciones industriales, identificando que aproximadamente un 60% de los casos han sido experimentales, mientras que el resto se distribuye entre celdas de producción, simulación y modelos teóricos (Velastegui et al., 2023). Esto demuestra el creciente interés en el uso de RL y técnicas de IA en manufactura avanzada.

Desde una perspectiva de control inteligente, se han desarrollado modelos basados en paradigmas holónicos y multiagente, los cuales buscan proporcionar

sistemas con mayor autonomía, cooperación y robustez ante cambios en el entorno (Quintero Henao, 2009). En particular, los sistemas holónicos de fabricación han sido modelados con metodologías multiagente que permiten representar de manera estructurada la jerarquía de producción y mejorar la gestión de recursos (Boggino, 2005). Estas metodologías han demostrado ser una herramienta eficaz para modelar la estructura recursiva de holones y facilitar el análisis de sistemas dinámicos en manufactura.

Asimismo, el concepto de sistemas de ejecución de manufactura (MES) con capacidades de inteligencia artificial ha sido estudiado como una solución para mejorar la interoperabilidad en fábricas inteligentes (Rolón & Martínez, 2012). La integración de aprendizaje automático en MES permite la detección de anomalías y la optimización de procesos mediante mecanismos de toma de decisiones autónomos, reduciendo la necesidad de intervención humana en la gestión de producción (Durão et al., 2022; Mantravadi et al., 2019; Meyer-Hentschel et al., 2020).

El uso de motores de juego como plataformas de simulación ha ganado popularidad en el desarrollo y evaluación de agentes de inteligencia artificial. Unity ML-Agents (Juliani et al., 2018) se ha consolidado como una de las principales herramientas en este ámbito, proporcionando un entorno flexible para la implementación y entrenamiento de agentes. A diferencia de otras plataformas, Unity permite la integración de gráficos realistas, simulación física avanzada y una configuración adaptable para distintos escenarios de aprendizaje (Hanski & Baris, 2021).

Estudios previos han utilizado Unity ML-Agents para evaluar el desempeño de agentes en entornos con diferentes estructuras de recompensa, analizando el impacto de recompensas densas y escasas en el proceso de aprendizaje. Asimismo, se han comparado distintos algoritmos de RL, como Proximal Policy Optimization (PPO) y Soft Actor-Critic (SAC), en tareas de navegación y resolución de problemas en entornos tridimensionales (Ilosvay & Iaccarino, 2024). Estos trabajos destacan la versatilidad de Unity para la experimentación con algoritmos de RL en tareas complejas.

Si bien los estudios mencionados han demostrado la aplicabilidad del RL en entornos industriales y simulaciones avanzadas, aún existen desafíos en la implementación de estrategias colaborativas en manufactura flexible. En este trabajo, proponemos un enfoque basado en múltiples agentes que trabajan de manera cooperativa para completar tareas en un entorno simulado utilizando Unity ML-Agents. A diferencia de estudios previos, nuestra investigación se centra en la interacción entre agentes y en la capacidad de adaptación a escenarios dinámicos, buscando evaluar la emergencia de comportamientos colaborativos en entornos de manufactura flexible.

3 Antecedentes metodológicos

Esta sección presenta los fundamentos teóricos del trabajo, organizados en tres ejes: 1) los SMA y su aplicación industrial mediante el paradigma holónico; 2) el marco del RL y su extensión MARL, incluyendo modelos como los MDPs (Markov Decision Processes,) y los Dec-POMDPs (Decentralized Partially

Observable Markov Decision Processes,); y 3) el rol de la simulación en MARL, comparando plataformas como Unity y Gazebo para sistemas robóticos colaborativos.

3.1 Sistemas Multiagente

Los SMA son un paradigma de la IA distribuida donde múltiples agentes autónomos interactúan para resolver problemas complejos. En manufactura flexible, los SMA permiten gestionar sistemas descentralizados con adaptabilidad, escalabilidad y tolerancia a fallos.

Un agente es una entidad computacional que percibe su entorno mediante sensores, procesa información y actúa mediante actuadores para cumplir tareas específicas. Según Russell & Norvig (2010), los agentes en SMA se caracterizan por cuatro propiedades clave: autonomía (operan sin intervención humana), reactividad (responden a cambios en tiempo real), proactividad (toman iniciativas para alcanzar objetivos) y habilidad social (comunicación y cooperación con otros agentes). En manufactura, los agentes pueden representar recursos físicos (robots, máquinas) o lógicos (sistemas de planificación), modelando fábricas como redes colaborativas.

El paradigma holónico (Van Brussel et al., 1998) integra los SMA con la estructura jerárquica de los sistemas de producción. Cada holón actúa como un agente que combina una entidad física (como un robot) con su correspondiente control lógico, organizándose en jerarquías recursivas que van desde el nivel de máquina hasta el de fábrica completa. La toma de decisiones se realiza de forma distribuida, donde cada holón negocia localmente con otros para optimizar objetivos globales como la minimización del tiempo de producción.

Matemáticamente, este sistema se representa mediante un grafo G=(V,E), donde V corresponde al conjunto de holones (agentes) y E define las relaciones de cooperación entre ellos, incluyendo flujos de materiales y canales de comunicación.

La coordinación se basa en negociación y teoría de juegos. Un protocolo común es el Contract Net Protocol (Smith, 1980), donde un agente manager anuncia una tarea, los contratistas envían ofertas (costo, tiempo) y el manager asigna la tarea al mejor candidato. En problemas colaborativos (ej: transporte coordinado), el equilibrio de Nash asegura que ningún agente se desvíe de la estrategia óptima colectiva (Shoham & Leyton-Brown, 2008). Por ejemplo, dos robots transportando una carga convergen a políticas de movimiento complementarias.

3.2 Aprendizaje por Refuerzo y MARL

El Aprendizaje por Refuerzo (RL) es un paradigma de aprendizaje automático donde un agente aprende a tomar decisiones óptimas mediante la interacción con un entorno dinámico, maximizando una señal de recompensa acumulada. En el contexto de la manufactura flexible, el RL permite la optimización de procesos estocásticos donde las condiciones operativas varían continuamente.

Proceso de Decisión de Markov. El marco matemático estándar para el RL es el Proceso de Decisión de Markov (MDP) (Sutton & Barto, 2018), definido por la tupla $\langle S, A, P, R, \gamma \rangle$, donde:

- S es el espacio de estados del sistema.
- A es el espacio de acciones disponibles.
- $P(s'|s, a)$ es la función de transición que define la probabilidad de moverse al estado s' desde s al ejecutar a.
- $R(s, a)$ es la función de recompensa inmediata.
- $\gamma \in [0, 1]$ es el factor de descuento para ponderar recompensas futuras.

El objetivo es hallar una política π^* que maximice el retorno esperado, definido por la ecuación de Bellman:

$$V^{\pi}(s) = R(s, \pi(s)) + \gamma \sum_{s'} P(s'|s, \pi(s)) V^{\pi}(s') \tag{1}$$

Extensión a Entornos Multiagente. Para escenarios con múltiples agentes, el modelo se generaliza a Procesos de Decisión de Markov Parcialmente Observables Descentralizados (Dec-POMDPs). Una tupla Dec-POMDP se define como $\langle I, S, \{A_i\}, P, \{O_i\}, \{R_i\}, \gamma \rangle$, donde I representa el conjunto de agentes y cada agente i percibe solo una observación parcial $o_i \in O_i$ del estado global S. En entornos cooperativos, todos los agentes comparten la misma función de recompensa ($R_i = R$), buscando maximizar el retorno conjunto a través de políticas descentralizadas $\pi_i(a_i|o_i)$.

Algoritmos en MARL Las aproximaciones actuales en MARL se dividen principalmente en tres categorías:

Métodos basados en valor: Estiman la función de valor conjunta. *QMIX* (Rashid et al., 2018) es el estado del arte en esta categoría para entornos cooperativos, utilizando una red de mezcla que asegura la monotonicidad entre las funciones de utilidad individuales y la conjunta.

Métodos Actor-Crítico: *MADDPG* (Multi-Agent Deep Deterministic Policy Gradients) (Lowe et al., 2017) extiende DDPG al dominio multiagente utilizando un esquema de "entrenamiento centralizado, ejecución descentralizada" (CTDE), donde el crítico tiene acceso al estado global durante el entrenamiento.

Gradientes de Política: *PPO* (Schulman et al., 2017), aunque nativo de un solo agente, es ampliamente utilizado en MARL (como MAPPO) debido a su estabilidad y eficiencia de muestra. Por otro lado, *COMA* (Counterfactual Multi-Agent) (Foerster et al., 2018) introduce un mecanismo de línea base contrafactual para resolver el problema de asignación de crédito en equipos cooperativos.

3.3 Entornos computacionales y MARL

Los entornos de simulación se han convertido en una pieza fundamental en el desarrollo de MARL, permitiendo la experimentación en escenarios complejos

de manufactura flexible sin los costos prohibitivos y riesgos asociados a los prototipos físicos. Estos entornos no solo facilitan el entrenamiento masivo de agentes, sino que habilitan la evaluación de estrategias colaborativas para tareas *ad-hoc* antes de su despliegue real.

En la última década, diversas plataformas se han consolidado como estándares de investigación. Mientras que *Gazebo*, integrado con ROS (Robot Operating System), domina la robótica tradicional gracias a su precisión en sensores y actuadores (Koenig & Howard, 2004), herramientas como *Unity ML-Agents* han ganado terreno por su equilibrio entre fidelidad visual y física, facilitando la creación de escenarios dinámicos personalizados (Juliani et al., 2018). Por otro lado, plataformas de nueva generación como *NVIDIA Isaac Sim* aprovechan la aceleración por GPU para simular miles de agentes en paralelo, una capacidad crítica para el aprendizaje profundo (Coumans & Bai, 2016).

La utilidad de estas herramientas en la industria es tangible. Estudios recientes reportan reducciones de hasta un 60% en tiempos de configuración para ensamblaje colaborativo (Wang et al., 2021) y mejoras de eficiencia del 30-40% en logística de AGVs (Chen et al., 2022). Esto es posible gracias a motores físicos avanzados que modelan fricción y restricciones articulares con alta fidelidad, factores cruciales en tareas como el transporte cooperativo de carga (Zhang et al., 2021).

Sin embargo, el mayor desafío reside en la brecha entre simulación y realidad (*sim-to-real gap*). Para mitigar esto, técnicas como la aleatorización de dominio (*domain randomization*) permiten variar parámetros físicos (masas, fricción) durante el entrenamiento, generando políticas robustas capaces de tolerar discrepancias superiores al 20% en el entorno real (Curşeu et al., 2020). Esta convergencia, sumada a la integración con gemelos digitales, está cerrando el ciclo de diseño, permitiendo el desarrollo de fábricas verdaderamente adaptativas (Saavedra Sueldo et al., 2023).

4 Metodología

En esta sección se presenta la metodología empleada para abordar el caso de estudio propuesto. En primer lugar, se define la problemática a resolver, vinculada a la manipulación colaborativa de materiales en espacios reducidos. Luego, se describe el desarrollo del entorno de simulación, la arquitectura del sistema, el diseño de los agentes, los mecanismos de interacción y el algoritmo utilizado. Finalmente, se detalla cómo los robots interactúan y aprenden a coordinarse para manipular y transportar una pieza dentro de un entorno físico restringido, representativo de condiciones reales en entornos de manufactura flexible.

4.1 Definición del problema

Tal como se menciona en la introducción, uno de los principales desafíos en entornos productivos modernos es la gestión coordinada de recursos físicos que

deben colaborar para ejecutar tareas logísticas tales como la manipulación y el transporte de materiales (Saavedra Sueldo et al., 2024). Estas tareas, aunque no agregan valor directo al producto final, impactan de forma directa sobre el rendimiento global del sistema, afectando la eficiencia del flujo de producción, el uso del espacio físico y la seguridad operativa.

En este contexto, la irrupción de robots móviles con patas, tanto bípedos como cuadrúpedos, equipados con mecanismos de manipulación (como grippers) representa una oportunidad para transformar los entornos productivos actuales. Estas tecnologías permiten concebir layouts dinámicos, en los que las estaciones de trabajo puedan reconfigurarse de manera continua en función de requerimientos variables, como cambios en la demanda o personalización de productos. Este nivel de flexibilidad exige el desarrollo de nuevas estrategias de colaboración y coordinación entre agentes autónomos capaces de levantar, mover, rotar y posicionar materiales en escenarios dinámicos y no estructurados.

4.2 Entorno simulado

Se diseñó un entorno tridimensional de $20 \times 20 \times 10$ unidades de longitud (ul) que representa una habitación industrial, donde dos agentes deben cooperar para transportar una pieza rígida a través de una abertura estrecha. Esta situación simula tareas reales en espacios reducidos, como pasillos o zonas de transferencia.

Los agentes, modelados como cápsulas móviles de $1 \times 1 \times 0.8$ ul, deben trasladar una tabla de 15×1 ul y 0.1 unidades de masa (um). La abertura tiene 10×10 ul, por lo que es necesario coordinar movimientos y orientación para evitar colisiones o caídas.

4.3 Arquitectura del sistema

El sistema se estructura de forma modular en dos componentes principales: `GameController`, que gestiona el estado general de la simulación, y la clase `Robot`, que implementa la lógica de cada agente. La Figura 1 muestra su diseño en UML (Unified Modeling Language).

`GameController` inicializa episodios, evalúa condiciones de éxito o fallo y administra las recompensas. Cada agente, mediante `CollectObservations`, obtiene información sobre su velocidad, la posición y orientación de la tabla, y actúa aplicando fuerzas en los ejes X y Z desde `OnActionReceived`.

El reinicio de episodio depende del sistema completo y no de cada agente por separado, ya que el éxito de la tarea requiere coordinación entre ambos.

4.4 Diseño de los agentes

Como se mencionó anteriormente, el comportamiento de cada agente se implementa en la clase `Robot`, la cual integra percepción, decisión y aprendizaje. Si bien cada agente opera de forma independiente en cuanto a su lógica de acción, el entorno y la tarea en sí los obligan a cooperar de forma implícita.

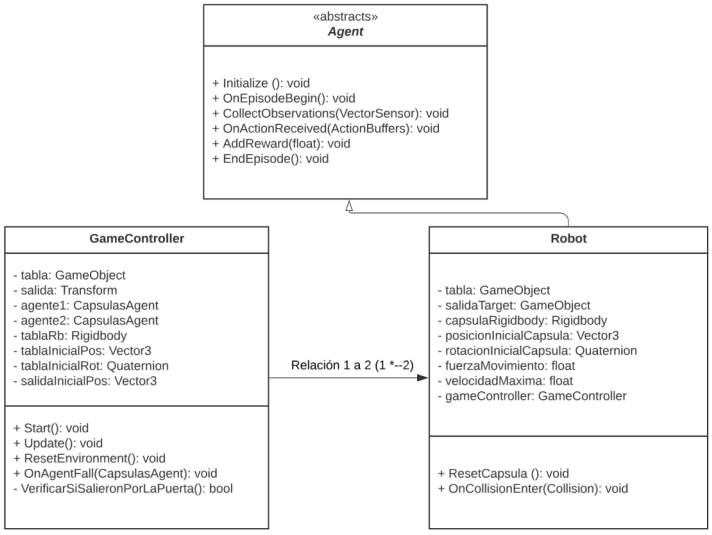


Fig. 1: Diagrama UML de la arquitectura del sistema.

Esta cooperación emerge del diseño conjunto de su espacio de observación, las acciones disponibles y el esquema de recompensas, el cual incentiva tanto el cumplimiento de metas individuales como de objetivos comunes.

La percepción del entorno se implementa a través del método `CollectObservations`, el cual recopila un vector de 12 dimensiones para cada agente. Este vector incluye información clave del estado del sistema, como la dirección relativa hacia la salida objetivo, la rotación actual de la tabla (considerando la componente Y del quaternion), la posición relativa de la tabla respecto al agente, la velocidad lineal del agente en los ejes X y Z, y la distancia euclidiana al objetivo. Estas observaciones permiten a los agentes construir una representación informativa del entorno, lo que les posibilita tomar decisiones razonadas en función de su situación actual. Para facilitar el proceso de entrenamiento y mejorar la estabilidad del modelo, todas las observaciones son normalizadas antes de ser procesadas por la red neuronal.

Espacio de acciones. El espacio de acción de cada agente consiste en valores continuos en el rango $[-1, 1]$, que representan las fuerzas aplicadas en los ejes X y Z del plano horizontal. Estas acciones se traducen en movimientos físicos mediante la aplicación de fuerzas al componente `Rigidbody` del agente, con una magnitud proporcional al atributo `fuerzaMovimiento` (15.0 N). Para garantizar estabilidad durante el aprendizaje, se impone un límite de velocidad máxima (5.0 unidades/segundo) que previene comportamientos erráticos.

Función de Recompensa. En este problema, los agentes deben transportar una pieza a través de un espacio reducido sin que esta se caiga ni se produzcan colisiones, manteniendo la estabilidad de la tabla que cargan en conjunto. Para fomentar tanto el progreso individual como la coordinación entre agentes, se

ha diseñado una función de recompensa compuesta por múltiples términos que capturan distintos aspectos del desempeño.

La recompensa total que recibe cada agente en cada paso de simulación está dada por:

$$R_{\text{total}} = \frac{1}{1+d} - 0.01 \cdot \mathbb{I}_{\text{inactivo}} - \mathbb{I}_{\text{inestable}} - 0.5 \cdot \mathbb{I}_{\text{colisión}} + 10 \cdot \mathbb{I}\text{éxito} \qquad (2)$$

donde d es la distancia euclidiana entre el agente y la salida, $\mathbb{I}_{\text{inactivo}}$ vale 1 si la velocidad del agente es menor a 0.1 unidades/segundo, $\mathbb{I}_{\text{inestable}}$ vale 1 si la tabla está inclinada más de 10° en los ejes X o Z, $\mathbb{I}_{\text{colisión}}$ vale 1 si el agente colisiona con una pared, y $\mathbb{I}_{\text{éxito}}$ vale 1 cuando se completa correctamente la tarea de transportar la tabla a través del vano de luz. Esta función logra equilibrar penalizaciones por comportamientos no deseados (como inactividad, colisiones o inestabilidad) con incentivos por el progreso hacia el objetivo y la finalización exitosa de la tarea.

4.5 Implementación del aprendizaje

Se optó por utilizar el algoritmo Proximal Policy Optimization (PPO) para resolver la tarea, por su eficacia para resolver problemas con espacios de acción continua y su estabilidad en entornos multiagente (Schulman et al., 2017). La implementación se realizó mediante RLlib, que facilitó la gestión distribuida del entrenamiento.

Dado que el entorno de simulación se desarrolló en Unity y el entrenamiento se ejecutó en Python, se utilizó Ray como puente de comunicación entre ambas plataformas, permitiendo una correcta integración durante el proceso de aprendizaje.

Arquitectura e Hiperparámetros. Cada agente utiliza una red neuronal profunda con dos capas completamente conectadas de 256 neuronas cada una, empleando la función de activación ReLU para mejorar la representación de las características del entorno (Haykin, 2009). Las observaciones se normalizan automáticamente para estabilizar el aprendizaje y mejorar la convergencia. Durante la fase de entrenamiento se utilizan los siguientes hiperparámetros (Schulman et al., 2017): una tasa de aprendizaje $\alpha = 0.0003$, un factor de descuento $\gamma = 0.99$, una longitud de episodio de $T = 3000$ pasos, un tamaño de batch de 4000 muestras, un ratio de clipping de 0.2, y un gradiente máximo también limitado a 0.2 para evitar explosiones en la magnitud de los ajustes.

Configuración del entrenamiento. Para optimizar la eficiencia del aprendizaje, el entrenamiento se llevó a cabo con cuatro procesos paralelos, lo que permitió la recolección simultánea de experiencias y redujo el tiempo total de simulación. Se estableció un esquema de checkpoints cada 5 iteraciones, lo que permitió la reanudación del entrenamiento en caso de interrupciones. El entrenamiento se da por concluido al cumplirse alguno de los siguientes criterios: i) se alcanzaron 10 millones de pasos simulados; ii) la recompensa media obtenida

por los agentes superó los 10.000 puntos; iii) se completaron 5.000 iteraciones de entrenamiento.

Estrategia Multi-agente. El comportamiento adoptado fue descentralizado, donde cada agente mantuvo su propia política de aprendizaje; no obstante, ambas políticas compartieron la misma arquitectura de red y el mismo conjunto de hiperparámetros. De esta manera, se logra que el aprendizaje de cada agente evolucione en función de su interacción con el entorno e independientemente del otro agente.

5 Experimentos

En esta sección se presenta la configuración experimental y un análisis exhaustivo del desempeño de los agentes entrenados mediante MARL en tareas colaborativas de transporte. A diferencia de trabajos previos, este estudio no solo evalúa la capacidad de aprendizaje (recompensa), sino también el impacto computacional que introduce la complejidad física del entorno (tiempos de cómputo y uso de recursos), validando la viabilidad de la simulación para escenarios de manufactura flexible.

A continuación, se detalla la infraestructura de simulación, la definición de los escenarios y el análisis de las métricas obtenidas.

5.1 Recursos computacionales

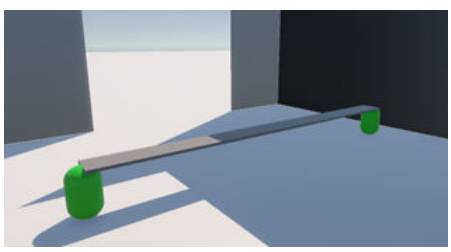
Para garantizar la reproducibilidad de los experimentos y la ejecución fluida de la física en tiempo real, se utilizó una estación de trabajo de alto rendimiento equipada con: procesador AMD Ryzen 7 3700X (8 núcleos, 16 hilos), 32 GB de RAM DDR4 a 3200 MHz, y una GPU AMD Radeon RX 6700 XT (12 GB). El entorno de software se desplegó sobre un sistema dual Windows 10 Pro / Linux con soporte ROCm. Esta configuración permitió paralelizar múltiples instancias del entorno, acelerando significativamente la recolección de experiencias (trazas) necesarias para el entrenamiento de las redes neuronales.

5.2 Configuración de los escenarios

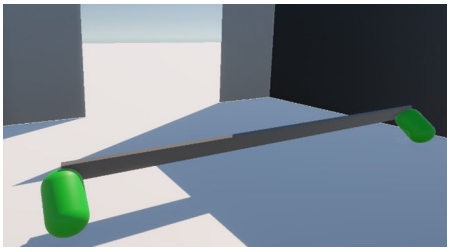
Se diseñaron dos escenarios experimentales para evaluar la robustez de la política aprendida ante distintos niveles de complejidad dinámica. En ambos casos, el objetivo es cooperativo: dos agentes deben trasladar una tabla de 15 unidades de longitud a través de una abertura de 10 unidades, lo que exige una coordinación precisa de fuerzas y movimientos para evitar colisiones o la caída del objeto.

Escenario 1: Manipulación de una pieza simple. Este escenario base (Figura 2a) evalúa la capacidad de sincronización básica. La tabla es un cuerpo rígido simple. Si los agentes no coordinan sus velocidades y direcciones, la tabla oscila y cae, finalizando el episodio con una penalización (Figura 2b).

Escenario 2: Manipulación de carga compuesta. Para incrementar la dificultad, se añade una caja de masa $m = 1$ kg sobre la tabla, sin



(a) Transporte coordinado.



(b) Caída de la tabla.

Fig. 2: Escenario 1: Transporte de una pieza simple.

sujeción mecánica (Figura 3). Esto introduce un problema de control más complejo: los agentes deben minimizar las aceleraciones bruscas para evitar que la inercia deslice la caja fuera de la tabla, añadiendo restricciones de suavidad al movimiento.

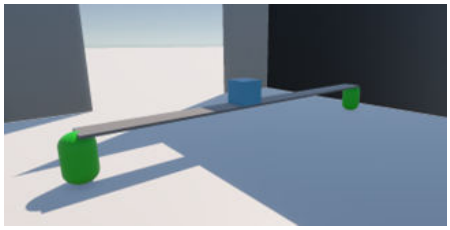


Fig. 3: Escenario 2: Transporte coordinado con carga.

5.3 Análisis de resultados

El desempeño de los agentes en ambos escenarios se evaluó a partir de tres métricas principales: la recompensa media obtenida por episodio, el tiempo de iteración y el uso de CPU. Estas métricas permiten analizar tanto la convergencia del modelo como el impacto computacional de cada entorno.

Recompensa Media La Figura 4 ilustra la evolución del aprendizaje. Se observa que el *Escenario 1* (curva celeste) alcanza una política estable y óptima (retorno $\approx$ 1000 puntos) alrededor del episodio 125. Por el contrario, el *Escenario 2* presenta una curva de aprendizaje más lenta, estabilizándose cerca del episodio 210 (retorno $\approx$ 4000 puntos). Este retraso de un 68% en la convergencia es atribuible a la necesidad de los agentes de explorar políticas más conservadoras ("suaves") para evitar la caída de la carga adicional, lo que reduce el espacio de acciones viables.

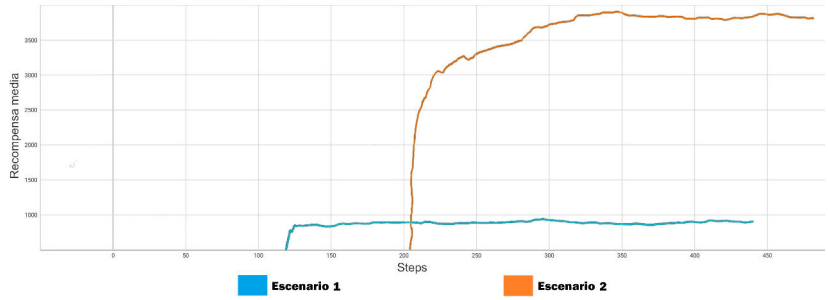


Fig. 4: Comparativa de convergencia: Recompensa media vs Episodios.

Tiempo de Iteración Un factor crítico en simulación es la estabilidad del paso de tiempo. La Figura 5 muestra el tiempo de cómputo por iteración de entrenamiento. A pesar de la diferencia de complejidad, el tiempo se mantuvo estable entre 60 y 61 segundos para ambos escenarios. Esto indica que la red neuronal utilizada es lo suficientemente ligera como para que la inferencia no se convierta en un cuello de botella, incluso cuando la tarea se vuelve más difícil.

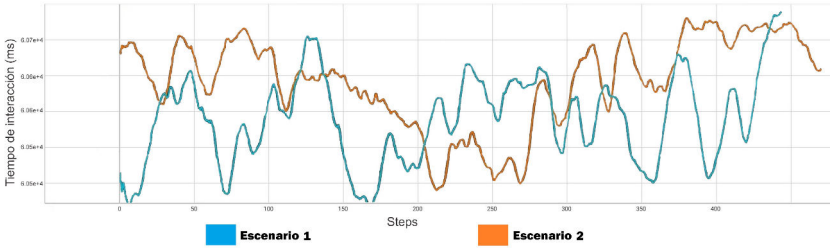


Fig. 5: Tiempo promedio de iteración.

Uso de CPU A diferencia del tiempo de iteración, el uso de CPU (Figura 6) revela el impacto directo de la física. Mientras que el escenario base oscila cerca del 25-30% de uso, el Escenario 2 muestra picos sostenidos hacia el 40%. Este incremento se justifica por la carga adicional en el motor de física (PhysX/Unity Physics): el sistema debe resolver colisiones continuas y cálculos de fricción dinámica entre la tabla y la caja en cada paso de tiempo (dt). Este dato es relevante para dimensionar el hardware necesario si se decidiera escalar el sistema a múltiples agentes y cargas.

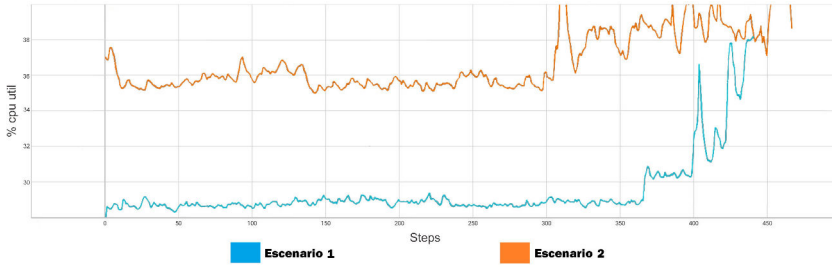


Fig. 6: Impacto de la complejidad física en el uso de CPU.

Comportamiento emergente En ambos escenarios, los agentes lograron desarrollar estrategias colaborativas sin intervención explícita . Esto evidencia la aparición de comportamiento emergente, entendido como la capacidad de los agentes para generar dinámicas colectivas complejas a partir de políticas individuales simples, entrenadas mediante refuerzo.

En el escenario sin carga, los agentes aprenden rápidamente a sincronizar sus movimientos para mantener la tabla equilibrada, evitando oscilaciones que llevarían al reinicio del episodio. Este comportamiento no fue predefinido, sino que emergió a partir de la retroalimentación del entorno y la necesidad de maximizar la recompensa.

En el escenario con carga, el comportamiento emergente es más complejo: además de mantener la sincronización lateral, los agentes ajustan la intensidad y dirección de la fuerza aplicada para evitar que la caja caiga de la tabla. Este ajuste de fuerzas, surge como resultado de la experiencia acumulada en múltiples episodios, donde errores sutiles son penalizados al provocar la caída de la carga.

En ambos casos, se observa que los agentes desarrollan una forma de "comunicación implícita" basada en la respuesta dinámica del objeto compartido. Es decir, al reaccionar a los cambios en la posición de la tabla o de la caja, cada agente modula su comportamiento como respuesta al del otro, sin intercambiar mensajes explícitos. Esta coordinación emergente sugiere que es posible alcanzar formas de cooperación efectivas en tareas físicas compartidas mediante MARL, incluso en ausencia de canales de comunicación directa. La Figura 7 y la Figura 8. ilustran una secuencia típica del comportamiento aprendido en el escenario sin carga y con carga, respectivamente.

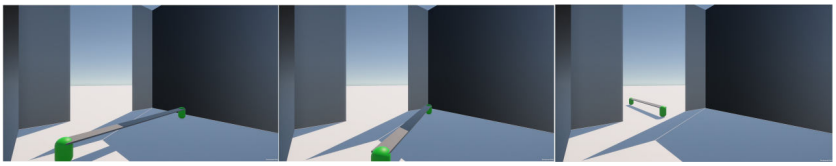


Fig. 7: Secuencia de ejecución: Transporte simple.

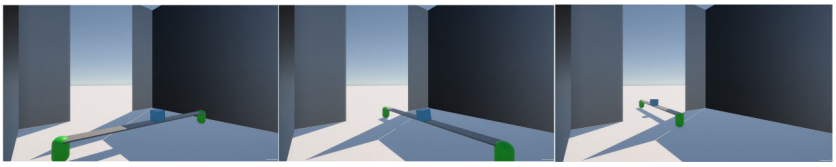


Fig. 8: Secuencia de ejecución: Transporte con carga balanceada.

6 Conclusiones

Este trabajo exploró el potencial del MARL para abordar tareas colaborativas típicas de entornos de manufactura flexible, en particular aquellas relacionadas con la manipulación y el transporte de materiales. A través del desarrollo de un entorno simulado y la implementación de dos escenarios de distinta complejidad, se demostró que es posible entrenar agentes capaces de coordinar sus acciones para alcanzar objetivos comunes, incluso bajo restricciones físicas como el transporte de una carga compartida.

Los resultados obtenidos, tanto en términos de métricas como de comportamiento emergente, evidencian que los agentes fueron capaces de aprender políticas efectivas sin necesidad de un controlador centralizado. Esto refuerza la idea de que los enfoques MARL constituyen una estrategia viable para enfrentar la creciente necesidad de flexibilidad, autonomía y adaptabilidad en sistemas de producción modernos, particularmente en el caso de PyMEs que deben reconfigurar sus procesos de forma constante.

Si bien los experimentos se realizaron en un entorno virtual y bajo ciertas simplificaciones, los aprendizajes obtenidos sientan las bases para futuras líneas de investigación orientadas a trasladar estas estrategias a sistemas físicos reales. Entre las posibles extensiones del trabajo se destacan la incorporación de más agentes, el manejo de múltiples objetos con diferentes propiedades físicas, y la inclusión de mecanismos de comunicación explícita entre agentes, con el fin de escalar la complejidad de las tareas y aproximarse a escenarios más realistas propios de la industria 4.0.

References

Albrecht, S. V., Christianos, F., & Schafer, L. (2024). *Multi-agent reinforcement learning: Foundations and modern approaches*. MIT Press.

Boggino, A. S. G. (2005). *Anémona: Una metodología multi agente para sistemas holónicos de fabricación* [Doctoral dissertation].

Chen., Cheng, C., & Li, J. (2018). Resource-constrained assembly line balancing problems with multi-manned workstations. *Journal of Manufacturing Systems*, *48*, 107–119.

Chen, X., Chen, R., & Yang, C. (2022). Research to key success factors of intelligent logistics based on iot technology. *Journal of Supercomputing*, *78*, 3905–3939.

Coumans, E., & Bai, Y. (2016). Pybullet, a python module for physics simulation for games, robotics and machine learning.

Curşeu, P. L., Rusu, A., Maricuţoiu, L. P., Vîrgă, D., & Măgurean, S. (2020). Identified and engaged: A multi-level dynamic model of identification with the group and performance in collaborative learning. *Learning and Individual Differences*, *78*.

Durão, L. F. C. S., McMullin, H., Kelly, K., & Zancul, E. (2022). Manufacturing execution system as an integration backbone for industry 4.0. *IFIP Advances in Information and Communication Technology*, *639*, 461–473.

Foerster, J. N., Farquhar, G., Afouras, T., Nardelli, N., & Whiteson, S. (2018). Counterfactual multi-agent policy gradients. *32nd AAAI Conference on Artificial Intelligence*, 2974–2982.

Hanski, J., & Baris, K. (2021). *An evaluation of the unity machine learning agents toolkit in dense and sparse reward video game environments* [Master's thesis, Faculty of Arts Department of Game Design].

Haykin, S. (2009). *Neural networks and learning machines* (Third). Pearson.

Ilosvay, V. B., & Iaccarino, E. (2024). Unity ml agents: Wall jump and soccertwos environment using reinforcement learning (rl) technique.

Juliani, A., Berges, V.-P., Teng, E., Cohen, A., Harper, J., Elion, C., Goy, C., Gao, Y., Henry, H., Mattar, M., & Lange, D. (2018). Unity: A general platform for intelligent agents.

Koenig, N., & Howard, A. (2004). Design and use paradigms for gazebo, an open-source multi-robot simulator. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, *3*, 2149–2154.

Li, J., Pang, D., Zheng, Y., Guan, X., & Le, X. (2022). A flexible manufacturing assembly system with deep reinforcement learning. *Control Engineering Practice*, *118*.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., & Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in Neural Information Processing Systems*, *2017-December*, 6380–6391.

Mantravadi, S., Li, C., & Møller, C. (2019). Multi-agent manufacturing execution system (mes): Concept, architecture & ml algorithm for a smart factory case. *ICEIS 2019 - Proceedings of the 21st International Conference on Enterprise Information Systems*, *1*, 465–470.

Meyer-Hentschel, M., Lohse, O., Rao, S., & Lepratti, R. (2020). Manufacturing operations management for smart manufacturing – a case study. *IFIP Advances in Information and Communication Technology*, *591*, 91–98.

Quintero Henao, L. F. (2009). *Un modelo de control inteligente para sistemas de manufactura basado en los paradigmas holónico y multi-agente* [Doctoral dissertation, Universidad Nacional de Colombia].

Rashid, A., Danezis, G., Chivers, H., Lupu, E., Martin, A., Lewis, M., & Peersman, C. (2018). Scoping the cyber security body of knowledge. *IEEE Security & Privacy*, *16*(4), 96–102.

Rolón, M., & Martínez, E. (2012). Agent-based modeling and simulation of an autonomic manufacturing execution system. *Computers in Industry*, *63*, 53–78.

Saavedra Sueldo, C., Perez Colo, I., De Paula, M., Villar, S. A., & Acosta, G. G. (2023). Ros-based architecture for fast digital twin development of smart manufacturing robotized systems. *Annals of Operations Research*, *322*(1), 75–99.

Saavedra Sueldo, C., Perez Colo, I., De Paula, M., Villar, S. A., & Acosta, G. G. (2024). Simulation-based metaheuristic optimization algorithm for material handling. *Journal of Intelligent Manufacturing*.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms.

Schwung, D., Reimann, J. N., Schwung, A., & Ding, S. X. (2018). Self learning in flexible manufacturing units: A reinforcement learning approach. *9th International Conference on Intelligent Systems 2018: Theory, Research and Innovation in Applications*, 31–38.

Shoham, Y., & Leyton-Brown, K. (2008). *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*.

Smith, R. G. (1980). The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, *29*(12).

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd). MIT Press.

Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., & Peeters, P. (1998). Reference architecture for holonic manufacturing systems: Prosa. *Computers in Industry*, *37*, 255–274.

Velastegui, R., Poler, R., & Díaz-Madroñero, M. (2023). Aplicación de algoritmos de aprendizaje automático a sistemas robóticos multiagente para la programación y control de operaciones productivas y logísticas: Una revisión de la literatura reciente. *Direccion y Organizacion*, *80*, 60–70.

Wang, C., Kim, Y. S., & Kim, C. Y. (2021). Causality between logistics infrastructure and economic development in china. *Transport Policy*, *100*, 49–58.

Zhang, M., Li, P., Xia, Y., Wang, K., & Jin, L. (2021). Labeling trick: A theory of using graph neural networks for multi-node representation learning. *Advances in Neural Information Processing Systems*, *11*, 9061–9073.