

Control de una Plataforma Stewart Simplificada de 3 GDL mediante Proximal Policy Optimization

Joel Ibaceta

Universidad Nacional de Ingeniería

joel.ibaceta.c@uni.pe

Marco Antonio Barrera Ninamango

Universidad Nacional de Ingeniería

marco.barrera.n@uni.pe

Jesus Gianpierre Campos Cardenas

Universidad Nacional de Ingeniería

j.campos.c@uni.pe

Resumen—Este trabajo presenta un enfoque de control basado en aprendizaje por refuerzo profundo para una plataforma Stewart simplificada con 3 grados de libertad (GDL). Se entrena una política mediante el algoritmo Proximal Policy Optimization (PPO) para lograr el control preciso de una pelota sobre la superficie móvil de la plataforma. Se describe el diseño mecánico, la generación del entorno simulado en PyBullet y la arquitectura de red basada en Actor-Critic. Los resultados muestran que el agente aprende a mantener la pelota dentro de la región objetivo a través de una combinación de recompensas modeladas cuidadosamente.

Index Terms—Plataforma Stewart, Aprendizaje por Refuerzo, PPO, PyBullet, Control Inteligente, Simulación Física.

I. Introducción

Las plataformas tipo Stewart, también conocidas como plataformas paralelas, han sido ampliamente empleadas en aplicaciones que requieren movimientos precisos en múltiples grados de libertad, como simuladores de vuelo, robótica de precisión y sistemas de estabilización. Su estructura cinemática cerrada proporciona una elevada rigidez y capacidad de respuesta, lo que las convierte en candidatas ideales para tareas de control fino.

Sin embargo, el control de plataformas con múltiples grados de libertad en espacios de acción continuos plantea desafíos significativos para los métodos de aprendizaje por refuerzo. A diferencia de los entornos discretos, donde las políticas óptimas pueden explorarse exhaustivamente, los entornos continuos requieren aproximaciones funcionales eficientes y una cuidadosa exploración para evitar comportamientos inestables o subóptimos.

En este trabajo se aborda el problema de controlar una plataforma Stewart simplificada con tres grados de libertad, cuyo objetivo es mantener una esfera en equilibrio sobre su superficie. Para ello, se emplea el algoritmo Proximal Policy Optimization (PPO), un método de optimización confiable dentro del marco Actor-Critic, conocido por su estabilidad y eficiencia en entornos continuos. Además, se presenta el diseño mecánico y la implementación del entorno de simulación en PyBullet, así como la arquitectura de red utilizada. El propósito es demostrar que es posible entrenar una política efectiva que genere comportamientos complejos de control sin supervisión directa ni modelado analítico detallado.

II. Antecedentes

Las plataformas paralelas tipo Stewart han sido estudiadas extensamente en control y robótica por su elevada rigidez y precisión, con resultados clásicos que abarcan modelado cinemático, análisis de singularidades y estrategias de control en lazo cerrado [1], [2]. En configuraciones con múltiples GDL, los métodos basados en modelo (p. ej., control computado del par, linealización, control robusto) han demostrado eficacia, pero requieren parametrizaciones precisas y pueden degradarse ante no linealidades de contacto o incertidumbres dinámicas.

El aprendizaje por refuerzo profundo (DRL) ha emergido como una alternativa promisoria para control continuo, al optimizar políticas directamente a partir de señales de recompensa. En particular, Proximal Policy Optimization (PPO) se ha consolidado como un método actor-crítico estable gracias a su objetivo con clipping, mostrando buen desempeño en tareas continuas con observaciones de alta dimensión [3]. Otros algoritmos relevantes incluyen DDPG [4] y SAC [5], que explotan políticas deterministas o máximos de entropía para mejorar exploración y estabilidad.

Como banco de pruebas análogo, el equilibrio de una esfera sobre una superficie (familia ball-on-plate/beam) ilustra los retos de control con contacto, fricción y acciones continuas gobernadas por inclinación. En visión por refuerzo, se ha mostrado que políticas parametrizadas por CNN pueden estabilizar y seguir objetivos en escenarios con dinámica parcialmente observada [?]. Para simulación física, PyBullet se ha convertido en una herramienta ampliamente adoptada en DRL por su soporte de contactos, restricciones, integración rápida y API en Python [?].

A pesar de estos avances, la literatura sobre plataformas Stewart simplificadas de 3 GDL gobernadas por DRL desde observaciones visuales directas sigue siendo limitada. Este trabajo contribuye en esa dirección mediante: (i) un entorno PyBullet con cierres cinemáticos impuestos en tiempo de ejecución, (ii) observaciones RGB apiladas, y (iii) entrenamiento con PPO orientado a mantener la esfera en una región céntrica de la plataforma.

III. Diseño del sistema

Se emplea una plataforma paralela tipo Stewart simplificada con tres grados de libertad (3 GDL), concebida específicamente para este estudio. El modelo geométrico

se desarrolló en CAD y se exportó como mallas STL; la cinemática y las propiedades iniciales se describieron en URDF. La plataforma superior, de geometría triangular, es accionada por tres conjuntos biela-mañivela distribuidos a 120° sobre la base. Cada actuador se modela mediante articulaciones revolutas con límites $\pm 1,57$ rad y acotaciones de velocidad y esfuerzo, reproduciendo el comportamiento de servomotores. Debido a la imposibilidad del URDF de representar lazos cinemáticos cerrados, la estructura se mantiene como árbol cinemático en la descripción y el cierre del paralelogramo se realiza en simulación mediante restricciones (véase Sec. ??).

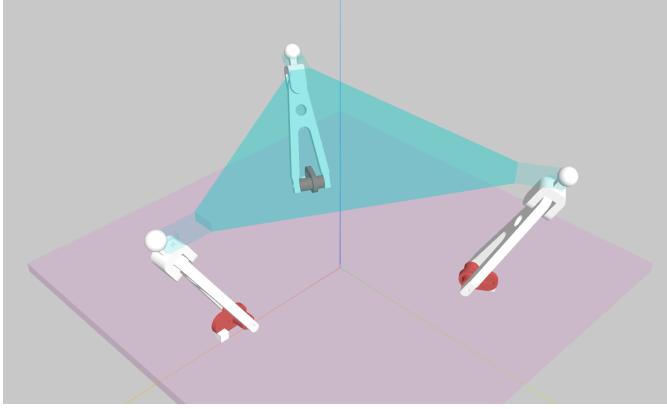


Figura 1. Modelo URDF de la plataforma Stewart simplificada (3 GDL) con mallas STL y articulaciones revolutas.

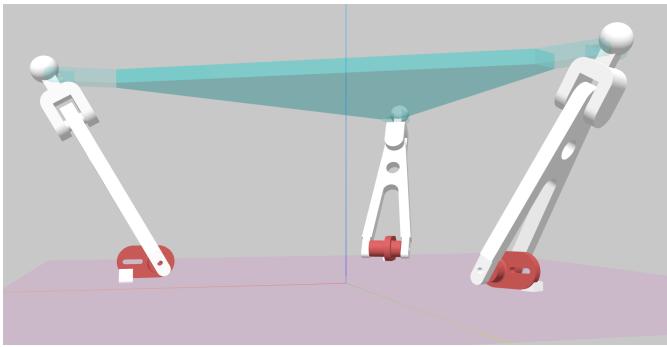


Figura 2. Vistas complementaria del modelo URDF de la plataforma.

IV. Simulación en PyBullet

Se desarrolló un entorno de simulación física utilizando PyBullet, replicando las características dinámicas del sistema físico. Se incorporaron restricciones de movimiento y detección precisa de colisiones, así como una pelota simulada que responde a la inclinación de la plataforma.

IV-A. Configuración del motor físico

Se utilizó un paso de integración de $\Delta t = 1$ ms (1 kHz). Por cada acción del agente se ejecutan cuatro subpasos de integración, obteniéndose una frecuencia de control efectiva de ≈ 250 Hz. El solucionador se configuró

con 300 iteraciones y términos de estabilización $ERP = contactERP = frictionERP = 0,85$ y $CFM = 10^{-7}$, con gravedad $g = -9.81$ m/s 2 . En la plataforma superior se fijó fricción lateral 0,12 y fricciones de rodadura y giro nulas, con el objeto de preservar sensibilidad a pequeñas inclinaciones sin introducir deriva numérica.

IV-B. Cierre cinemático en tiempo de ejecución

Dado que URDF no admite lazos cinemáticos cerrados, el cierre entre las puntas de los eslabones y la plataforma superior se impone en tiempo de ejecución mediante dos restricciones por eslabón: (i) JOINT_POINT2POINT, para co-localizar puntos de anclaje en marcos locales, y (ii) una restricción JOINT_GEAR de razón unitaria alrededor de la normal de la plataforma, que atenúa el giro relativo (twist). Este esquema mantiene la descripción cinemática como árbol en URDF, a la vez que recupera en simulación el comportamiento de un mecanismo con eslabones pasivos.

IV-C. Modelo de contacto y dinámica de la esfera

La esfera se modeló con radio $r_b = 8$ mm y masa efectiva $m_b \approx 18$ g. Se empleó restitución 10^{-3} , fricción lateral 0,15, fricción de rodadura y de giro 10^{-4} , y amortiguamientos lineal y angular de 10^{-3} . Estos valores reducen el rebote y el jitter numérico sin sacrificar responsividad a perturbaciones pequeñas.

Cuadro I
Parámetros físicos y numéricos de la simulación

Paso de simulación	1 ms; 4 subpasos por acción
Iteraciones del solver	300
ERP / contactERP / frictionERP	0,85/0,85/0,85
CFM global	10^{-7}
Gravedad	-9.81 m/s 2
Esfera: radio / masa	0,008 m / 0,018 kg
Esfera: restitución	0,001
Esfera: fricción lat./rod./giro	0,15/ 10^{-4} / 10^{-4}
Esfera: amortiguamiento lin./ang.	0,001/0,001
Top: fricción lat./rod./giro	0,12/0/0

V. Modelamiento del entorno y del agente

V-A. Espacios de observación y acción

El entorno (StewartBalanceEnv) expone observaciones visuales RGB de tamaño 84×84 píxeles. Para dotar de inercia visual, se apilan 4 cuadros consecutivos mediante FrameStackObservation, obteniéndose un tensor $\mathbf{o}_t \in [0, 255]^{84 \times 84 \times 12}$ que se normaliza a $[0, 1]$ y se reordena a formato CHW (ToCHW).

El espacio de acción es continuo, $\mathcal{A} = [-1, 1]^3$, y se interpreta como comandos normalizados para tres articulaciones. Cada acción se escala por un factor $\alpha=0,3$ rad y se aplica en position control de PyBullet con fuerza máxima 80 N·m; por paso del entorno se integran 4 subpasos de simulación.

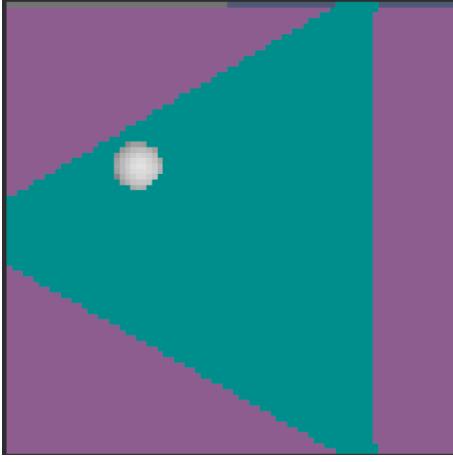


Figura 3. Observación visual usada por el agente: imagen RGB de 84×84 px capturada por la cámara cenital. En entrenamiento se apilan 4 cuadros consecutivos (stack en canal) y se reordenan a CHW para la CNN.

V-B. Función de recompensa

Sea $\mathbf{x}_t \in \mathbb{R}^2$ la posición local de la bola sobre la plataforma superior en el paso t , y c el centro del triángulo (circuncentro). Denotamos por r_{\max} el radio máximo del triángulo en coordenadas locales (distancia del centro al vértice más lejano). La distancia normalizada al centro es

$$d_t = \frac{\|\mathbf{x}_t - c\|_2}{r_{\max} + 10^{-8}} \in [0, \infty), \quad \dot{\mathbf{x}}_t \approx \frac{\mathbf{x}_t - \mathbf{x}_{t-1}}{\Delta t}. \quad (1)$$

La recompensa total es la suma de seis términos:

$$r_t = r_t^{\text{prog}} + r_t^{\text{pot}} + r_t^{\text{in}} + r_t^{\text{vel}} + r_t^{\Delta} + r_t^{\text{alive}}. \quad (2)$$

V-B0a. (a) Progreso radial: Recompensa instantánea por acercamiento al centro (densa y con signo):

$$r_t^{\text{prog}} = w_{\text{prog}}(d_{t-1} - d_t). \quad (3)$$

V-B0b. (b) Potencial cuadrático suave: Pozo que estabiliza cerca de c :

$$r_t^{\text{pot}} = -w_{\text{pot}} d_t^2. \quad (4)$$

V-B0c. (c) “Inside” suave con margen: Sea (u, v, w) la coordenada baricéntrica de \mathbf{x}_t respecto a (A, B, C) y $x_{\min} = \min\{u, v, w\}$. Con margen métrico $m = \frac{1}{2}r_b$ (siendo r_b el radio de la bola), usamos un smoothstep cúbico:

$$s(a, b, x) = t^2(3 - 2t), \quad t = \text{clip}\left(\frac{x - a}{b - a}, 0, 1\right).$$

La contribución queda

$$r_t^{\text{in}} = w_{\text{in}}\left(2s(-m, m, x_{\min}) - 1\right). \quad (5)$$

Este término evita discontinuidades en el borde del triángulo y aporta gradiente útil hacia el interior.

V-B0d. (d) Velocidad penalizada cerca del centro: Sólo se penaliza la velocidad cuando d_t es pequeño, usando una ventana gaussiana $\text{near}(d_t) = \exp(-(d_t/\sigma)^2)$:

$$r_t^{\text{vel}} = -w_{\text{vel}} \text{near}(d_t) \frac{\|\dot{\mathbf{x}}_t\|_2}{v_{\text{ref}} + 10^{-8}}. \quad (6)$$

V-B0e. (e) Suavidad de acción y bono de vida: Con $\Delta \mathbf{a}_t$ el incremento de acción entre pasos:

$$r_t^{\Delta} = -\kappa_{\Delta} \|\Delta \mathbf{a}_t\|_2, \quad r_t^{\text{alive}} = 10^{-3}. \quad (7)$$

V-B0f. Valores por defecto: A menos que se indique lo contrario, se usa

$$w_{\text{prog}} = 0,50, \quad w_{\text{pot}} = 0,10, \quad w_{\text{in}} = 0,20, \quad w_{\text{vel}} = 0,05, \\ \sigma = 0,25, \quad v_{\text{ref}} = 0,10 \text{ m/s}, \quad \kappa_{\Delta} = 0,01, \quad r_t^{\text{alive}} = 10^{-3}.$$

Esta descomposición provee una señal densa, diferenciable y bien condicionada: r_t^{prog} guía el acercamiento, r_t^{pot} sostiene la estabilización, r_t^{in} reemplaza el indicador duro por una transición suave en el borde, r_t^{vel} amortigua oscilaciones en la vecindad del centro, r_t^{Δ} favorece acciones suaves y r_t^{alive} desincentiva terminaciones prematuras.

V-C. Criterios de terminación

Un episodio termina si la bola permanece fuera del triángulo durante ≥ 8 pasos consecutivos (out_patience). Se trunca al alcanzar 2000 pasos.

V-D. Arquitectura perceptual y actor-crítico

Las observaciones apiladas alimentan una CNN ligera (VisionCNN) con tres bloques Conv-ReLU de kernel/stride (8, 4), (4, 2) y (3, 1), seguidos de Global Average Pooling y una cabeza densa hasta un embedding de 256 dimensiones.

Sobre dicho embedding se montan dos cabezas:

- Actor: MLP 256 → 128 → 3 (ReLU intermedia) que produce la media $\mu(\mathbf{o}_t)$ de una Gaussiana diagonal. La desviación estándar es global y aprendible, con parámetro $\log \sigma \in [-5, 2]$.
- Crítico: MLP 256 → 64 → 1 (Tanh intermedia) que estima $V(\mathbf{o}_t)$.

La política se implementa como una Gaussiana “aplastada” por tanh (distribución Tanh-squashed) para respetar $\mathcal{A} = [-1, 1]^3$. Durante el muestreo se utiliza rsample (reparametrización), y la probabilidad corregida incorpora el jacobiano de tanh; para la bonificación de entropía se emplea la entropía aproximada de la Gaussiana base. Todas las capas lineales y convolucionales utilizan inicialización ortogonal.

V-E. Entrenamiento

Se utiliza PPO [3] con estimación de ventajas GAE($\gamma=0,99$, $\lambda=0,98$) y objetivo con clipping $\epsilon=0,2$. Las ventajas se normalizan por minibatch. Las observaciones uint8 se escalan a $[0, 1]$. El entrenamiento se organiza en rollouts de $n=1024$ pasos seguidos de $E=4$ épocas sobre minibatches de $B=64$. Se aplica clip_grad_norm con umbral 0,5. La pérdida total es

$$\mathcal{L} = \mathcal{L}_{\text{PPO}} + c_v \mathcal{L}_{\text{value}} - c_{\text{ent}}(t) \mathcal{H},$$

con $c_v=0,5$ y un coeficiente de entropía $c_{\text{ent}}(t)$ que se reduce linealmente (annealing) desde 10^{-3} hasta 0 durante el 10 % inicial de los pasos de entrenamiento, favoreciendo exploración temprana y explotación posterior.

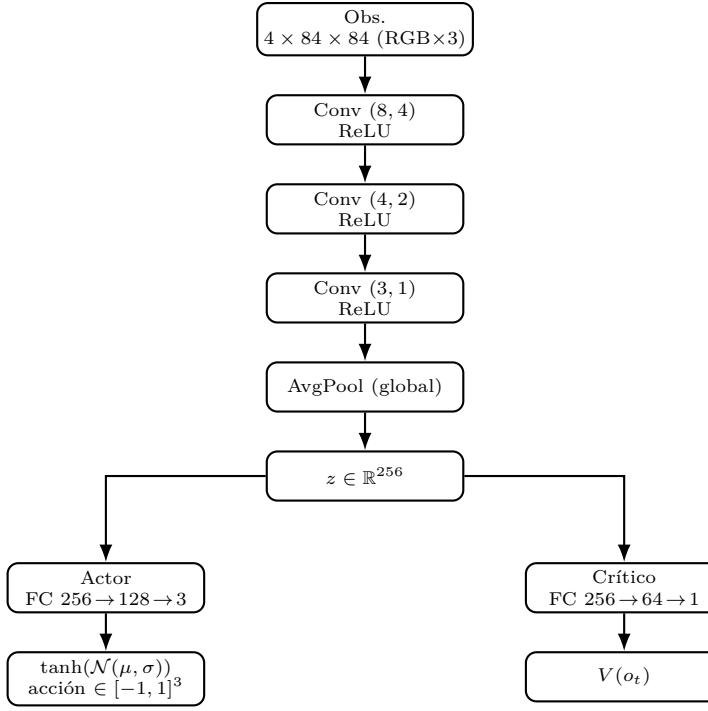


Figura 4. Arquitectura resumida. Backbone Conv–Conv–Conv–AvgPool $\rightarrow z \in \mathbb{R}^{256}$; la rama actor (FC 256 \rightarrow 128 \rightarrow 3) produce $\tanh(\mathcal{N}(\mu, \sigma))$; la rama crítico (FC 256 \rightarrow 64 \rightarrow 1) estima $V(o_t)$.

Cuadro II
Hiperparámetros principales del entorno

Parámetro	Valor
Resolución / stack	84x84 / 4 frames
Acción α (rad)	0,3 (escala)
Subpasos por paso	4 a 1 kHz
z_{drop} (m)	-0,01
out_patience	8
$k_{\text{in}}, k_{\text{out}}$	0,1, 0,1
k_c, k_{Δ}	1,0, 0,01
k_{stay}, τ	0,2, 0,1 $r_{\text{máx}}$

VI. Entrenamiento con Proximal Policy Optimization

El agente se entrena con PPO en régimen on-policy a partir de rollouts de $n=1024$ pasos. Cada actualización itera $E=6$ épocas sobre minibatches de tamaño $B=256$, con optimizador Adam ($\text{lr } 3 \times 10^{-4}$, $\varepsilon=10^{-5}$) y recorte de gradiente $\|\nabla\|_2 \leq 0,5$. Las imágenes uint8 se escalan a $[0, 1]$; las ventajas se normalizan por minibatch.

VI-0a. Estimación de ventajas (GAE).: Con $\gamma=0,99$ y $\lambda=0,98$,

$$\delta_t = r_t + \gamma V_{\theta}(o_{t+1}) - V_{\theta}(o_t), \quad (8)$$

$$\hat{A}_t = \sum_{l=0}^{T-t-1} (\gamma\lambda)^l \delta_{t+l}, \quad R_t = \hat{A}_t + V_{\theta}(o_t). \quad (9)$$

VI-0b. Objetivo de PPO.: Sea $r_t(\theta) = \exp(\log \pi_{\theta}(a_t|o_t) - \log \pi_{\theta_{\text{old}}}(a_t|o_t))$ el cociente de

probabilidades. La pérdida por política con clipping ($\varepsilon=0,2$) es

$$\mathcal{L}_{\text{PPO}} = -\mathbb{E}[\min(r_t(\theta) \hat{A}_t, \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon) \hat{A}_t)]. \quad (10)$$

VI-0c. Pérdidas de valor y entropía.: La función de valor se ajusta por MSE y se añade una bonificación entrópica para favorecer la exploración:

$$\mathcal{L} = \mathcal{L}_{\text{PPO}} + c_v \mathbb{E}[(V_{\theta}(o_t) - R_t)^2] - c_{\text{ent}}(t) \mathbb{E}[\mathcal{H}(\pi_{\theta}(\cdot|o_t))]. \quad (11)$$

Se emplea $c_v=0,5$ y un annealing lineal del coeficiente entrópico:

$$c_{\text{ent}}(t) = \max(c_{\text{mín}}, c_0 \left(1 - \frac{t}{T_a}\right)), \quad c_0=0,01, \quad c_{\text{mín}}=10^{-4}, \quad T_a=0,1 T,$$

donde T es el número total de pasos de entrenamiento. La política es gaussiana diagonal “aplastada” con tanh; las acciones se muestran con reparametrización y la log-prob se corrige con el jacobiano de tanh.

VI-0d. Protocolo de actualización.:

1. Recolectar n transiciones $\{o_t, a_t, r_t\}$ con $\pi_{\theta_{\text{old}}}$.
2. Bootstrap con $V_{\theta}(o_T)$ y computar \hat{A}_t , R_t (GAE).
3. Barajar el buffer y optimizar durante E épocas sobre minibatches:
 - Normalizar \hat{A}_t por minibatch.
 - Calcular \mathcal{L} y hacer backprop con clip_grad_norm.
4. Actualizar $\theta_{\text{old}} \leftarrow \theta$; registrar métricas (pérdidas, entropía, norma del gradiente, retorno/longitud de episodio) en TensorBoard.

VII. Resultados

Se entrenó el agente durante $\sim 10^5$ pasos de interacción (equivalentes a varios centenares de episodios). La Fig. 5 resume el desempeño: el return por episodio muestra una tendencia monótonamente creciente y se estabiliza en torno a $2,4\text{--}2,6 \times 10^2$, mientras que la longitud de episodio aumenta de forma sostenida (incremento aproximado de 3–4 \times), evidenciando que la política aprende a mantener la esfera dentro del área objetivo durante intervalos cada vez más prolongados.

La Fig. 6 caracteriza la optimización. La pérdida de política (PPO clipped) desciende rápidamente y se approxima a cero, indicando convergencia de la política bajo el criterio proximal. La pérdida del crítico decrece de forma marcada, reflejando estimaciones de valor más consistentes a medida que progresó el entrenamiento. En paralelo, el coeficiente de entropía $c_{\text{ent}}(t)$ se reduce (programa de annealing), y la norma del gradiente cae desde valores altos ($\sim 10^2$) hasta un régimen estable en el que la dinámica de actualización permanece bien condicionada. En conjunto, estas curvas sugieren una transición ordenada de exploración a explotación sin inestabilidades numéricas.

Para evaluar la calidad del control, la Fig. 7 presenta la evolución temporal de los ángulos en las tres articulaciones. Se observa una disminución progresiva de la varianza y

de la amplitud pico–pico, con rangos típicos finales por debajo de unas pocas centésimas de radián. Este patrón es consistente con un comportamiento de servoing más suave y con la penalización de variaciones de acción incluida en la señal de recompensa, lo que reduce oscilaciones y artefactos de saturación.

En términos cualitativos, las trayectorias evidencian que la política aprendida inclina la plataforma en direcciones mínimas pero oportunas para recentrar la esfera sin exceder los límites articulares. Asimismo, la mejora simultánea de retorno y longitud, junto con la caída del coeficiente de entropía, sugiere que el esquema de annealing resulta clave: con una entropía inicial suficiente para explorar la dinámica de contacto, y una reducción gradual que evita que el ruido estocástico “ensucie” la política en etapas avanzadas.

Limitaciones observadas.: Aun cuando el agente logra un control estable y suave, se identificaron dos retos: (i) la estimación implícita de velocidad a partir de observaciones exclusivamente visuales es no trivial; la política aprende un correlato suficientemente útil, pero persisten fallos esporádicos en transitorios rápidos cerca del borde del triángulo; (ii) el shaping geométrico es eficaz en simulación, aunque su traslación directa al sistema físico (real2sim) exige una calibración dinámica cuidadosa para preservar el efecto de fricción y micro–rebotes. Estas observaciones motivan, como trabajo futuro, incorporar señales de estado mínimas adicionales (p. ej., velocidad proyectada estimada) o técnicas de domain randomization orientadas a robustez sim–real.

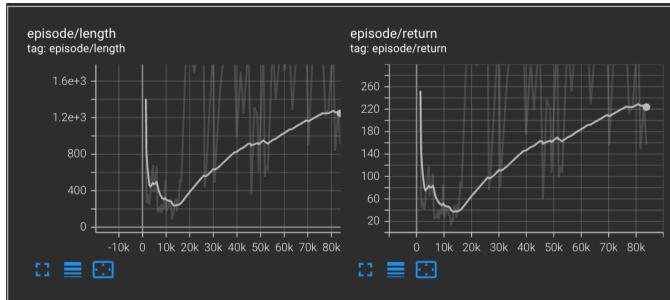


Figura 5. Desempeño del agente: retorno por episodio (línea clara = media móvil).

VIII. Conclusiones y discusión

Este trabajo mostró que una política PPO con arquitectura actor–crítico basada en visión (CNN) puede aprender a estabilizar una esfera sobre una plataforma Stewart simplificada de 3 GDL utilizando únicamente observaciones RGB apiladas. A partir de los experimentos realizados, destacamos:

1) Control continuo y entropía. Aunque PPO resulta adecuado para espacios continuos, su desempeño es altamente sensible al diseño de la señal de recompensa, al coeficiente de entropía y al annealing de dicho coeficiente.

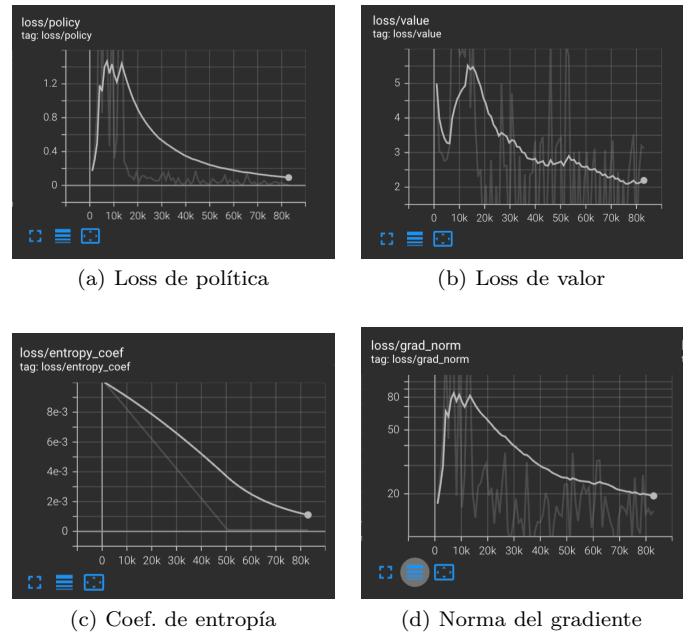


Figura 6. Curvas de optimización y estabilidad numérica.

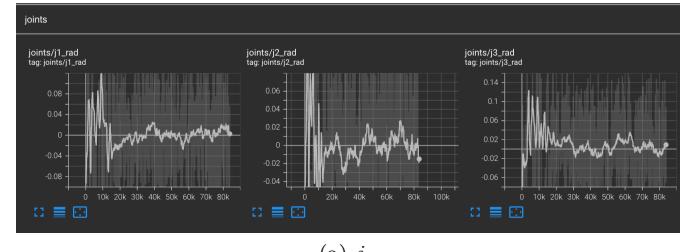


Figura 7. Evolución de los ángulos articulares: tendencia a movimientos más suaves.

Sin un decaimiento progresivo de la entropía y una penalización a variaciones bruscas de acción, la política tiende a comportamientos ruidosos que impiden la estabilización fina.

2) Importancia del modelado físico. Para que el aprendizaje sea estable, el entorno debe reflejar con fidelidad la interacción física bola–plataforma. En particular, los parámetros del solver (ERP/CFM), fricciones (lateral, de rodadura y de giro) y las restricciones que cierran el lazo cinemático influyen de forma directa en la calidad de los gradientes que recibe el agente y, por tanto, en la tasa de convergencia.

3) Observabilidad de la velocidad. Inferir velocidad exclusivamente desde imágenes es difícil. El apilado de cuadros provee una pista temporal mínima, pero no sustituye una medición explícita. Aun así, se evitó inyectar velocidad o estados del simulador para mantener el objetivo de real-to-sim basado en visión. Esta decisión, si bien conservó el realismo del flujo sensorial, hace el problema más exigente para PPO.

4) Shaping geométrico y transferencia. El shaping hacia

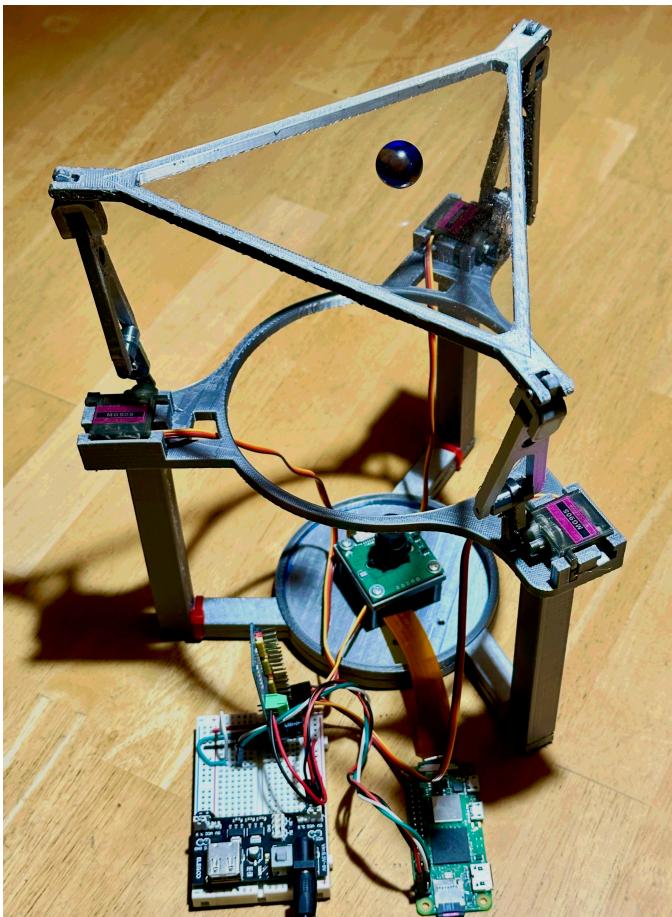


Figura 8. Prototipo físico del modelo en el mundo real: plataforma Stewart simplificada de 3 GDL.

el circuncentro facilita la exploración y acelera el aprendizaje; sin embargo, parte del sesgo geométrico puede no trasladarse con fidelidad al dispositivo físico si la superficie no es ideal o si existen pequeñas asimetrías. En consecuencia, el rendimiento observado en simulación no necesariamente se replica en hardware sin una calibración o identificación cuidadosa de parámetros.

En conjunto, los resultados confirman la viabilidad de PPO para este tipo de tareas visuales continuas, pero subrayan que la calidad del modelado físico y el tratamiento de la entropía son determinantes para evitar políticas ruidosas y para lograr estabilidad sostenida.

IX. Limitaciones y trabajo futuro

Entre las principales limitaciones se encuentran: (i) ausencia de líneas base clásicas (p.ej., control PD/LQR con observación densa) para cuantificar la ganancia relativa del enfoque basado en RL; (ii) falta de una evaluación sistemática de robustez bajo variaciones de fricción y pendientes; y (iii) brecha sim-to-real aún no verificada en prototipo físico.

Como trabajo futuro se propone: (a) incorporar estimadores visuales de flujo óptico o self-supervised para

mejorar la observabilidad de velocidad sin romper el real-to-sim; (b) combinar visión con un mínimo de propriocepción (ángulos o derivadas articulares) bajo un esquema de fusión que preserve transferibilidad; (c) utilizar domain randomization e identificación de parámetros para cerrar la brecha sim-to-real; y (d) evaluar políticas híbridas con objetivos geométricos más próximos a las restricciones físicas del prototipo.

Referencias

- [1] D. Stewart, “A platform with six degrees of freedom,” Proceedings of the Institution of Mechanical Engineers, vol. 180, no. 1, pp. 371–386, 1965.
- [2] J.-P. Merlet, Parallel Robots, 2nd ed. Berlin, Heidelberg: Springer, 2006.
- [3] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” arXiv:1707.06347, 2017.
- [4] T. P. Lillicrap, J. J. Hunt, A. Pritzel, et al., “Continuous control with deep reinforcement learning,” in Proc. ICLR, 2016.
- [5] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor,” in Proc. ICML, pp. 1861–1870, 2018.
- [6] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, “Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection,” The International Journal of Robotics Research, vol. 37, no. 4–5, pp. 421–436, 2018.
- [7] E. Coumans and Y. Bai, “PyBullet, a Python module for physics simulation for games, robotics and machine learning,” 2016–2021. [Online]. Available: <https://pybullet.org>