CEA Comité Español de Automática
XLVII JORNADAS DE AUTOMÁTICA CÓRDOBA 2026
UNIVERSIDADE DA CORUÑA

Jornadas de Automática

Sistema de entrenamiento virtual inmersivo para prótesis robóticas mioeléctricas

Orgiler, Pablo[a], Romero, Cristina[a], Martínez-Pérez, Raquel[a], Úbeda, Andrés[a,*], García, Gabriel J.[a]

[a]*Human Robotics Group, Universidad de Alicante, Crta. San Vicente del Raspeig S/N, 03690, Alicante, España.*

To cite this article: Orgiler, Pablo, Romero, Cristina, Martínez-Pérez, Raquel,Úbeda, Andrés, García, Gabriel J. 2026. Immersive virtual training system for myoelectric robotic prostheses. Jornadas de Automática, 47. https://doi.org/10.17979/ja-cea.2026.47.13840

Resumen

El desarrollo de prótesis robóticas mioeléctricas ha mejorado la autonomía de las personas con amputación, aunque su elevado coste y la complejidad del control muscular generan altas tasas de abandono. Para abordar este problema, se ha creado un sistema de entrenamiento en realidad virtual para usuarios con amputación transradial, permitiendo practicar el control de la prótesis antes de adquirir el dispositivo físico. El sistema integra la captación EMG mediante Noraxon miniDTS, el procesamiento y clasificación en MATLAB, el control en ROS 2 y un entorno virtual en Unity que incorpora el modelo de la mano robótica RH8D, visualizado con Oculus Quest. La validación con sujetos sanos muestra mejoras en los tiempos de ejecución, buena usabilidad y un alto potencial como alternativa eficiente y de bajo coste para la rehabilitación preprotésica convencional.

Palabras clave: Tecnología asistiva e ingeniería de rehabilitación, Análisis e interpretación de bioseñales, Trabajo en entornos reales y virtuales, Modelado, simulación y visualización de sistemas biomédicos, Tecnología robótica

Immersive virtual training system for myoelectric control of robotic prostheses

Abstract

The development of myoelectric robotic prostheses has improved the autonomy of people with limb loss, although their high cost and the complexity of muscle control lead to high abandonment rates. In order to solve this problem, a virtual reality training system has been created for users with transradial amputation, allowing them to train their prosthetic control before acquiring the physical device. This system integrates EMG acquisition using the Noraxon miniDTS device, signal processing and classification in MATLAB, control through ROS 2 and a virtual environment in Unity that incorporates the RH8D hand model, visualized with an Oculus Quest. Validation with healthy subjects has shown improvements in execution time, good usability and a high potential as an efficient, low-cost alternative to conventional pre-prosthetic rehabilitation.

Keywords: Assistive technology and rehabilitation engineering, Bio-signals analysis and interpretation, Work in real and virtual environments, Biomedical system modeling, simulation and visualization, Robotics technology

1. Introducción

Las prótesis han evolucionado desde dispositivos rudimentarios hasta sistemas complejos con accionamientos mecánicos, eléctricos o mioeléctricos. Este progreso ha permitido la aparición de dispositivos capaces de reproducir movimientos más naturales mediante la interpretación de señales electromiográficas (EMG) procedentes del sistema neuromuscular.

Las prótesis mioeléctricas modernas se basan en la captación de señales musculares mediante electrodos, que pueden ser superficiales o intramusculares. Estas señales reflejan la intención motora del usuario y permiten accionar el dispositivo de forma proporcional o mediante estrategias avanzadas como máquinas de estados, control directo o reconocimiento de patrones (Farina et al. (2014)). Por otro lado, el desarrollo de manos robóticas antropomórficas, como la RH8D de Seed

*Autor para correspondencia: andres.ubeda@ua.es

Robotics, ha permitido replicar con mayor fidelidad el funcionamiento y la capacidad manipulativa de la mano humana. A pesar de estos avances, el proceso de aprendizaje para utilizar una prótesis mioeléctrica es complejo. La falta de correspondencia entre los músculos residuales y los movimientos naturales de la mano, junto con la variabilidad de la señal EMG, dificulta la adaptación del usuario y puede conducir al abandono del dispositivo (Espinosa and Nathan-Roberts (2019)).

Diversos estudios han demostrado la utilidad de sistemas de realidad virtual que permiten realizar un entrenamiento previo a la obtención del dispositivo. Lambrecht et al. (2011) desarrollaron un entorno VR que integra simulación física, seguimiento cinemático y control mioeléctrico para entrenar tareas funcionales con una prótesis virtual. Phelan et al. (2021) evaluaron un sistema basado en HTC VIVE y Myo Armband, observando mejoras en la aceptación del dispositivo y en la sensación de propiedad corporal. Otros trabajos, como los de Hashim et al. (2021), Rozevink et al. (2025) o Dhawan et al. (2019), han explorado enfoques basados en videojuegos, simuladores interactivos o entornos inmersivos para mejorar la coordinación muscular, la motivación y la capacidad de control del usuario. En conjunto, estas investigaciones demuestran que los sistemas basados en realidad virtual son una alternativa a la rehabilitación tradicional y facilitan el aprendizaje del control mioeléctrico antes de la adquisición de una prótesis física.

El sistema desarrollado para este trabajo se concibe como una herramienta de entrenamiento basada en realidad virtual dirigida a personas con amputación transradial, integrando todos los componentes necesarios para reproducir de forma realista el control de una prótesis robótica. Las señales EMG se adquieren mediante el sistema Noraxon miniDTS, con una frecuencia de muestreo de 1500Hz, y se procesan en Noraxon MR 3.10, desde donde se envían a MATLAB para su clasificación. Para el control de la prótesis virtual, se ha diseñado una arquitectura en ROS 2 Jazzy, que recibe en tiempo real la clasificación generada en MATLAB y ejecuta las acciones de control correspondientes utilizando una máquina de estados. La escena virtual se ha diseñado en Unity y se visualiza a través de las Oculus Quest, proporcionando un entorno inmersivo en primera persona que facilita la interacción natural del usuario con la prótesis virtual.

2. Mano robótica RH8D

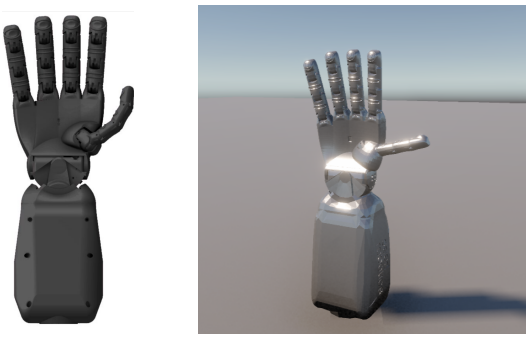
La RH8D Adult Size Dexterous Robot Hand de Seed Robotics, figura 1(a), es una mano robótica de tamaño adulto que está inspirada en la mano derecha humana.

El dispositivo incorpora 19 grados de libertad incluyendo un pulgar oponible y una muñeca esférica completa. Los dedos están formados por tres segmentos que son accionados por 8 actuadores inteligentes alojados en el interior de la propia mano. Estos actuadores proporcionan control preciso de posición, velocidad y corriente, permitiendo inferir la fuerza aplicada en tiempo real.

En cuanto a los sensores, la RH8D integra un conjunto de medidas internas (posición, velocidad, corriente, temperatura y estado de carga) junto con un sensor ToF (*Time of Flight*) en la palma y la opción de añadir almohadillas capacitivas o sensores táctiles de alta resolución (hasta 1 mN / 0,1 g). Estas capacidades amplían la percepción del entorno y mejoran la interacción con objetos de distinta geometría.

El modelo tiene una capacidad de carga de 2,5 kg en tracción vertical y 1 kg en manipulación en el espacio tridimensional. No obstante, el modelo posee un diseño compacto, de únicamente 620 g, que permite su integración en brazos robóticos, pudiendo incluso desactivar el grado de libertad asignado a la rotación de la muñeca para usar el propio del robot.

Por último, la RH8D cuenta con un paquete de ROS de código abierto, el modelo URDF (*Unified Robot Description Format*) para su integración en simuladores como Unity y compatibilidad con el protocolo de comunicación serie de bajo nivel (UART).



(a) RH8D Adult Size Dexterous Robot Hand de Seed Robotics.

(b) RH8D importada en la escena de Unity.

Figura 1: Modelo de mano robótica antropomórfica RH8D.

3. Desarrollo de la plataforma

3.1. Integración del modelo de la mano en Unity

El primer paso en el diseño del sistema es la importación del modelo URDF de la mano en Unity. Para ello, se ha utilizado la biblioteca ROS# de Siemens, que permite comunicar ROS con Unity. A través de esta biblioteca, se genera el objeto `RosConnectorObject`, que es el encargado de establecer la comunicación entre Unity y el servidor de rosbridge.

Por otro lado, se ha diseñado un archivo de lanzamiento en ROS (`.launch`) para transferir la descripción URDF de la mano robótica a Unity. En este *script* se definen los parámetros necesarios para establecer la conexión con Unity, es decir, la dirección IP y el puerto, y la dirección del modelo URDF.

De este modo, al lanzar el archivo `.launch` mientras la ventana de edición del `RosConnectorObject` está abierta en Unity, el modelo URDF de la RH8D se importa correctamente en la escena, tal como se muestra en la figura 1(b).

3.2. Diseño de la arquitectura de control en ROS

Una vez importado el modelo en Unity, se ha creado un paquete en ROS 2 Jazzy que implementa la arquitectura de control. Este paquete consta de dos archivos: el que define la máquina de estados para el control de la mano y el archivo `.launch` que permite inicializar el sistema y comunicar la parte de control con la escena en Unity.

Para realizar el control del modelo de la mano robótica, se ha utilizado una máquina de estados debido a que permite

estructurar de forma clara las diferentes acciones que puede ejecutar el sistema. Esta máquina de estados está formada por cuatro estados, cada uno asociado a una acción específica sobre el modelo virtual de la mano dentro de la escena de Unity, como se puede ver en la figura 2.

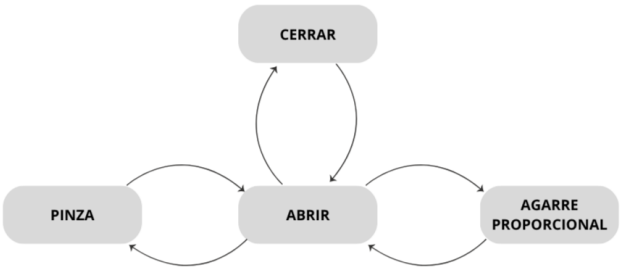


Figura 2: Máquina de estados implementada.

- **Abrir**: Este estado tiene la finalidad de extender todas las articulaciones del modelo virtual de la mano. Actúa como el reposo del sistema, ya que, una vez implementado el control mioeléctrico, el usuario accederá a esta configuración cuando relaje los músculos. Además, se han incorporado restricciones inspiradas en la biomecánica de la mano humana: únicamente es posible transitar entre estados de cierre si previamente se ha pasado por el estado abrir, garantizando una secuencia de movimientos más natural y evitando transiciones bruscas y saltos en las articulaciones entre configuraciones incompatibles.
- **Cerrar**: El estado *cerrar* tiene como objetivo llevar la mano a una configuración de cierre completo, adoptando una forma similar a un puño. El proceso se ejecuta en dos fases diferenciadas: en primer lugar, se cierran de manera simultánea las articulaciones correspondientes a los dedos índice, corazón, anular, meñique y la articulación de la base del pulgar; posteriormente, se ajustan el resto de las articulaciones del pulgar. Esta secuencia reproduce el patrón natural de cierre de la mano humana, evitando que los dedos se entrecrucen o interfieran entre sí durante el movimiento.
- **Pinza**: Al igual que en el anterior estado, hay dos fases en el cierre: en la primera fase, se cierran completamente todas las articulaciones de los dedos corazón, anular, meñique y la articulación de la base del pulgar; después, se cierran $-15°$ el resto de articulaciones. Este estado se diseñó inicialmente para permitir el agarre de objetos pequeños, aunque posteriormente se ha utilizado en todo tipo de objetos.
- **Agarre proporcional**: El estado *agarre proporcional* es similar a *cerrar*, pero incorpora un control proporcional que modula el cierre según la amplitud de la señal EMG normalizada respecto al valor de contracción máxima (MVC) medido previamente. Esto permite un control más natural e intuitivo, de modo que el usuario regula la amplitud del cierre variando su contracción muscular. Al igual que en el resto de estados, la postura alcanzada se mantiene hasta recibir la señal de transición.

El funcionamiento de la máquina de estados requiere un conjunto de *topics* para intercambiar información entre el nodo de control, la escena de Unity y el *script* de MATLAB. Cada *topic* cumple una función específica dentro del sistema, asegurando que la mano virtual reciba las órdenes adecuadas. A continuación, se van a mostrar las transiciones utilizadas:

- `/transition`: Este *topic* transmite las órdenes de cambio de estado. El nodo interpreta los comandos enviados a través del *topic* y los traduce a las transiciones internas de la máquina de estados. En el caso de que el mensaje recibido no coincida con ninguno de los comandos asignados a los estados definidos, la máquina de estados ignorará dicha entrada y permanecerá en el estado actual hasta recibir una orden válida.
- `/pos`: Proporciona la posición actual de las articulaciones de la mano en Unity. El nodo almacena los valores articulares para utilizarlos como referencia en el estado *abrir*, donde es necesario conocer la postura actual para incrementar progresivamente los ángulos hasta alcanzar la apertura completa.
- `/proportional`: En este *topic* se publica el valor normalizado de la contracción muscular, utilizado exclusivamente por el estado *agarre proporcional*. Este valor, comprendido entre 0 y 1, se utiliza para modular dinámicamente el cierre de la mano, permitiendo que el usuario controle el nivel de cierre a voluntad. Este mecanismo permite un control continuo e intuitivo: cuanto mayor es la activación muscular, mayor es el cierre de la mano, manteniendo siempre una relación proporcional entre la señal EMG y la postura de la mano.
- `/fingers`: Es el *topic* encargado de enviar los valores objetivo de las articulaciones hacia Unity. Cada estado calcula los ángulos deseados para cada articulación y los publica en este *topic*, que es recibido por el modelo de la mano en la escena de Unity y así actualizar su postura en tiempo real. Además, para evitar que la mano adopte de forma instantánea el ángulo final, lo que produciría movimientos bruscos y poco naturales, se realiza una interpolación en la que, cada 0,01 segundos, se ajusta el ángulo de cada articulación en incrementos o decrementos unitarios, según el estado, hacia su valor objetivo, enviando en cada iteración la nueva posición a Unity. Este procedimiento garantiza un movimiento continuo y suave.

3.3. Simulación en Unity

De forma simultánea al diseño de la arquitectura de control en ROS, se diseñó el comportamiento que debe tener el modelo virtual de la mano en Unity y sus interacciones con los objetos de la escena. Este trabajo se estructuró en tres *scripts*, cada uno de ellos orientado a una funcionalidad específica dentro del sistema.

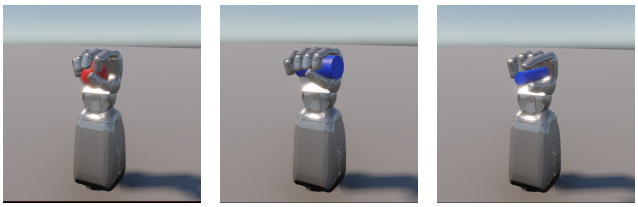
- `HandController`: Se encarga de reproducir en Unity las posiciones articulares recibidas desde ROS mediante un proceso que incluye inicialización, recepción de mensajes y actualización del modelo. En la fase inicial

se crean los vectores que almacenan los ángulos objetivo y el estado de bloqueo de cada dedo, además de configurar los `HingeJoint` y `Rigidbodies` para garantizar la estabilidad y colisiones fiables. Al llegar un mensaje, se identifica cada articulación por su etiqueta y se actualiza su ángulo objetivo. Finalmente, el movimiento se aplica mediante `WritePhysicsPosition`, ajustando los parámetros del resorte del `HingeJoint` para lograr un movimiento suave y realista.

- `JointStatePublisher`: Está incluido por defecto en ROS# y publica en ROS la posición actual de cada articulación del modelo de la mano virtual. Se ha empleado para conocer la configuración articular previa antes de ejecutar el estado *abrir*, garantizando una transición suave y evitando saltos bruscos entre posiciones. Las posiciones se envían mediante el *topic* `/pos`.

- `HandGrabManager`: Su función principal es detectar cuándo la mano se encuentra lo suficientemente cerrada como para iniciar un agarre, identificar objetos cercanos susceptibles de ser manipulados y gestionar su vinculación temporal a la mano durante la interacción. De este modo, actúa como un módulo intermedio entre el sistema de control de ROS y el comportamiento físico de los objetos en la escena de Unity.

En la figura 3 se puede observar la adaptación del modelo virtual de la RH8D a diferentes objetos utilizando los distintos estados de cierre especificados previamente. Cada estado se encuentra manipulando un objeto diferente: el estado *cerrar* está manipulando una esfera mientras que los otros dos están interactuando con cilindros de diferentes tamaños.



(a) Estado *cerrar*. (b) Estado *agarre proporcional*. (c) Estado *pinza*.

Figura 3: Ejemplos de agarre de diferentes objetos utilizando los tres estados de la mano virtual.

3.4. Control mioeléctrico

El sistema de control mioeléctrico permite transformar la actividad muscular del usuario en comandos para la mano virtual. Para ello, se desarrolló un código en MATLAB que recibe en tiempo real las señales EMG transmitidas desde Noraxon MR 3.10, establece comunicación con ROS 2 y publica tanto los cambios de estado como el nivel proporcional de activación en los *topics* correspondientes.

Tras iniciar la conexión con el servidor de *streaming* de Noraxon MR 3.10, el *script* adquiere continuamente las señales EMG, calcula su amplitud, las normaliza respecto a MVC y determina el estado muscular comparando cada canal con umbrales predefinidos. Para medir el MVC del usuario, antes de comenzar la sesión de entrenamiento se le solicita que realice la máxima contracción posible de los músculos evaluados. A partir de esta contracción, se selecciona el pico de la señal EMG como el nivel máximo de activación que el usuario puede generar en cada músculo. Por otro lado, para ajustar los umbrales de activación empleados en las transiciones entre estados, se siguió un procedimiento análogo al utilizado para el MVC: se registró el nivel de la señal EMG cuando el usuario se encontraba en reposo y durante la realización de movimientos que no deberían generar un cambio de estado, como desplazar el brazo o cambios en la postura del usuario. Estos registros permitieron identificar el ruido basal y las activaciones involuntarias asociadas al movimiento del miembro, estableciendo así un umbral suficientemente alto como para evitar transiciones no deseadas, pero lo bastante sensible para detectar activaciones reales del usuario.

Los valores procesados se publican en ROS, donde son utilizados por la máquina de estados que controla la mano virtual. Con el fin de simplificar la interacción, el sistema emplea únicamente dos sensores EMG: uno para *pinza*, otro para *agarre proporcional*, y la activación simultánea de ambos para *cerrar*, manteniéndose en *abrir* mientras no haya actividad muscular. Esta configuración basada en dos sensores se seleccionó con el objetivo de reducir la complejidad de uso del sistema, minimizar el tiempo de calibración por usuario y evitar una carga cognitiva innecesaria durante la interacción. Por estos motivos, se descartó la incorporación de un mayor número de canales en la arquitectura desarrollada, priorizando así un diseño más accesible, rápido de ajustar y adecuado para usuarios sin experiencia previa en control mioeléctrico.

Para garantizar el correcto funcionamiento del sistema durante las pruebas con participantes sanos, se estableció la colocación de los sensores EMG en los siguientes músculos: el flexor superficial de los dedos y la cabeza lateral del tríceps braquial, ambos en el brazo derecho, como se puede ver en la figura 4. Además, los electrodos de cada sensor se situaron a menos de 2 cm entre sí, asegurando una correcta captación de la señal y minimizando el ruido asociado a separaciones excesivas. En el caso de los usuarios finales con amputación transradial, convendría evaluar distintos músculos para determinar cuáles ofrecen un control más sencillo e intuitivo. Aun así, el sistema seguirá funcionando independientemente de los músculos seleccionados siempre que se obtengan señales EMG válidas, manteniéndose idénticos los procedimientos para calcular el MVC y los umbrales.

(a) Electrodos en el músculo flexor de los dedos. (b) Electrodos en la cabeza lateral del tríceps.

Figura 4: Colocación de los electrodos.

3.5. *Sistema de realidad virtual*

Para implementar el sistema de realidad virtual es necesario instalar Unity OpenXR Plugin y los *Building Blocks* **Camera Rig**, encargado del seguimiento de la cabeza y del cuerpo del usuario, y **Hand Tracking**, que permite el seguimiento y la representación visual de las manos dentro de la escena.

Una vez instaladas las dependencias necesarias para el funcionamiento del sistema en las Oculus Quest, se editaron los componentes encargados de mostrar la mano derecha para sustituirla por el modelo funcional de la RH8D, tal y como se puede ver en la figura 5.

Figura 5: Visualización de las manos en la ejecución.

La idea de que el modelo virtual se superponga a la mano derecha se fundamenta en que, cuando se utilice con pacientes amputados, estos dispongan de una prótesis pasiva, proporcionando una percepción realista del peso y volumen que tendrá la prótesis definitiva una vez instalada, aumentando la inmersión.

3.6. *Interacción de los elementos del sistema*

El objetivo de esta última sección es explicar el funcionamiento conjunto del sistema y detallar la comunicación entre sus elementos para lograr una simulación inmersiva y útil para los usuarios finales.

En resumen, como se ha comentado a lo largo de la sección 3, las señales EMG se captan con los sensores EMG y son enviadas al *software* Noraxon MR 3.10, donde se rectifican y suavizan. Una vez se han procesado, MATLAB recibe las señales en tiempo real, a través del protocolo de comunicación HTTP Streaming, y determina si alguna de las señales ha superado el umbral de activación. Una vez clasificadas las señales, MATLAB envía a ROS el estado actual a través del *topic* `/transition` y la proporción de la contracción de la señal captada respecto al MVC usando el *topic* `/proportional`. Asimismo, ROS interpreta estos mensajes, actualiza la máquina de estados y genera las configuraciones articulares que la mano debe adoptar. Paralelamente, Unity envía a ROS la posición articular actual, lo que permite que la arquitectura de control ejecute los movimientos partiendo de la postura real del modelo. Una vez especificadas las órdenes de control, estas se envían a Unity a través del *topic* `/fingers`. En caso de que la mano esté en contacto con un objeto y la orden de control sea un estado de cierre, se rodea el objeto hasta conseguir el agarre y se emparenta el objeto a la mano, es decir, ambos elementos quedan vinculados. En contraparte, si la mano está agarrando un objeto y le llega la orden de abrir, se comienzan a incrementar los ángulos de las articulaciones hasta superar el umbral de cierre, momento en el que se desvincula el objeto de la jerarquía, quedando libre de la mano como al inicio de la ejecución.

En la figura 6 se puede ver de forma esquemática la interacción existente entre los diferentes elementos del sistema, permitiendo el funcionamiento del mismo.

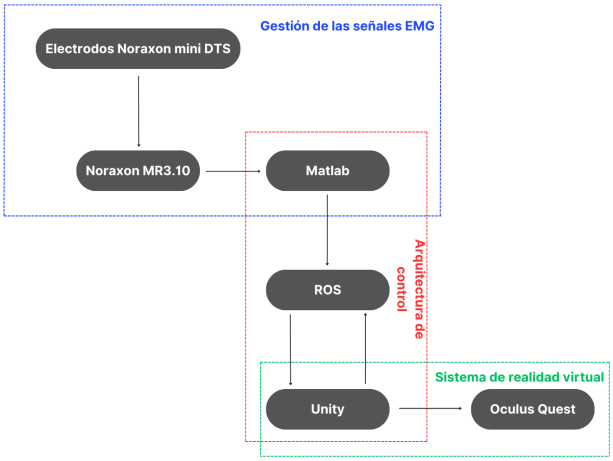


Figura 6: Esquema del funcionamiento del sistema.

4. Validación y resultados

Tras el desarrollo del sistema, se llevó a cabo una validación del mismo en la que participaron cinco sujetos sanos, cuyas características se pueden ver en la tabla 1.

Participante	Edad	Sexo	Lateralidad
P1	22	Masculino	Diestro
P2	21	Masculino	Diestro
P3	21	Masculino	Zurdo
P4	22	Masculino	Diestro
P5	18	Femenino	Diestro

Tabla 1: Características de los participantes.

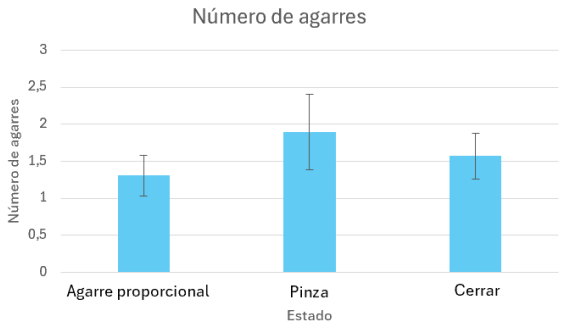
El protocolo de validación consistió en que cada participante debía trasladar los cinco objetos de la escena desde un punto inicial hasta otro punto de la mesa virtual. Durante estas tareas se registraron dos métricas principales: el tiempo de ejecución y el número de agarres necesarios para completar cada trayectoria. Al finalizar las pruebas, cada usuario completó un formulario destinado a recoger información relacionada con la fatiga percibida durante la tarea, la intuitividad del control y la facilidad de uso.

En general, el estado *agarre proporcional* fue el que mostró un menor tiempo medio de ejecución y una mayor estabilidad en el número de intentos, estableciéndose como el gesto más sencillo y útil para todos los participantes. En cambio, el estado *pinza* fue el que generó más intentos fallidos y más tiempo requirió para completar las trayectorias, especialmente en las primeras iteraciones, debido a la dificultad de accionar la cara lateral del tríceps braquial. Finalmente, en los resultados se aprecia un aprendizaje y adaptación al sistema por parte de los usuarios debido a la disminución de los tiempos de ejecución y a que los participantes requieren de menos

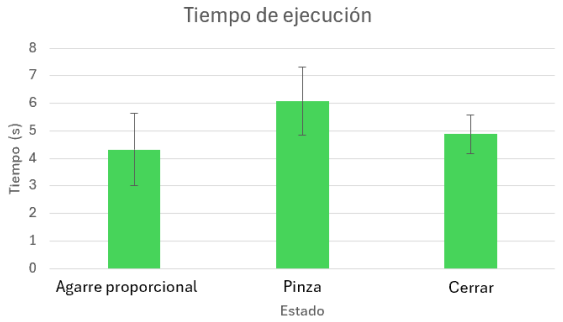
Pregunta	P1	P2	P3	P4	P5
¿Cómo de intuitivo te ha parecido el sistema?	4	4	4	4	3
¿Cómo de inmerso te has sentido dentro del entorno virtual?	5	5	5	5	5
¿Cómo de fácil te ha resultado alternar entre los diferentes tipos de agarre?	4	4	5	4	3
¿Cómo de inmediata ha sido la respuesta de la mano a la orden?	5	4	4	4	5
Tras realizar el control de la prótesis, ¿cómo de fatigado te has encontrado?	3	4	4	4	2
¿Sientes que con la práctica has mejorado la habilidad de manejar la prótesis?	5	5	5	5	5
¿Crees que el entrenamiento te hubiera servido de ayuda en caso de tener que usar ahora la prótesis real?	4	5	5	5	5

Tabla 2: Tabla con las respuestas del formulario completado por los participantes.

intentos para completar las tareas. En la figura 7 se muestran las medias y desviaciones típicas calculadas a partir de los resultados de todos los usuarios, reflejando el comportamiento observado en los participantes durante la ejecución de las pruebas.



(a) Gráfica de la media del número de agarres por estado.



(b) Gráfica tiempo medio de ejecución por estado.

Figura 7: Gráficas de los resultados de la validación.

En cuanto a las respuestas del cuestionario, los participantes respondieron positivamente al sistema, manifestando que su experiencia fue completamente inmersiva, bastante intuitiva y que el proceso fue de ayuda para practicar con el uso de la prótesis, sintiéndose seguros en caso de tener que manejar una prótesis real tras las pruebas. Respecto al nivel de fatiga, dos de los participantes expresaron haber sentido un nivel de fatiga muscular considerable al acabar el proceso, mientras que el resto experimentó únicamente una fatiga leve. Las preguntas y respuestas del cuestionario se pueden observar en la tabla 2, en la cual se ha utilizado una escala de puntuación entre 0 y 5, donde 0 es la menor puntuación.

Sin embargo, todos los participantes coincidieron en que el accionamiento asociado a la activación de la cabeza lateral del tríceps braquial resultaba especialmente complicado de ejecutar. Por esta razón, se deberían buscar alternativas más intuitivas que faciliten el manejo de la mano robótica y que permitan una mejor adaptación por parte del usuario.

5. Conclusión

El artículo presenta un sistema completo que permite a personas con amputación transradial entrenar el uso de una prótesis robótica en un entorno de realidad virtual inmersivo. El sistema integra tres módulos principales: la arquitectura de control en ROS 2 para el modelo virtual de la mano RH8D, el procesamiento y clasificación de señales EMG en MATLAB y la gestión del movimiento y del entorno virtual en Unity.

Como trabajos futuros, se propone incorporar el modelo físico de la mano RH8D para funcionar como un gemelo digital y mejorar el sistema de control para evitar las molestias detectadas durante la validación. Estas mejoras incluirían sustituir el accionamiento basado en el tríceps por otro músculo o emplearlo únicamente para las transiciones entre modos de agarre, dejando el control principal al flexor superficial de los dedos. Además, se plantea la realización de pruebas con pacientes amputados, con el fin de validar el sistema directamente con los usuarios finales y evaluar su eficacia en un contexto clínico real.

Referencias

Dhawan, D., Barlow, M., Lakshika, E., 2019. Prosthetic rehabilitation training in virtual reality. In: 2019 IEEE 7th International Conference on Serious Games and Applications for Health (SeGAH). IEEE, Kyoto, Japan, pp. 1–8.
DOI: 10.1109/SeGAH.2019.8882455

Espinosa, M., Nathan-Roberts, D., 2019. Understanding prosthetic abandonment. In: Proceedings of the Human Factors and Ergonomics Society Annual Meeting. Vol. 63. SAGE Publications, Los Angeles, CA, pp. 1644–1648.
DOI: 10.1177/1071181319631508

Farina, D., Jiang, N., Rehbaum, H., Holobar, A., Graimann, B., Dietl, H., 2014. The extraction of neural information from the surface emg for the control of upper-limb prostheses: Emerging avenues and challenges. IEEE Transactions on Neural Systems and Rehabilitation Engineering 22 (4), 797–809.
DOI: 10.1109/TNSRE.2014.2305111

Hashim, N. A., Abd Razak, N. A., Gholizadeh, H., Abu Osman, N. A., 2021. Video game–based rehabilitation approach for individuals who have undergone upper limb amputation: Case-control study. JMIR Serious Games 9 (1).
DOI: 10.2196/17017

Lambrecht, J. M., Pulliam, C. L., Kirsch, R. F., 2011. Virtual reality environment for simulating tasks with a myoelectric prosthesis: an assessment and training tool. Journal of Prosthetics and Orthotics 23 (2), 89–94.
DOI: 10.1097/JPO.0b013e318217a30c

Phelan, I., Arden, M., Matsangidou, M., Carrion-Plaza, A., Lindley, S., 2021. Designing a virtual reality myoelectric prosthesis training system for amputees. In: Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems. Association for Computing Machinery, pp. 1–7.
DOI: 10.1145/3411763.3443454

Rozevink, S. G., Maas, B., Murgia, A., Bongers, R. M., van der Sluis, C. K., 2025. Therapists' and prosthesis users' assessments of a virtual reality environment designed for upper limb prosthesis control training. Journal of Hand Therapy 38 (4), 745–752.
DOI: 10.1016/j.jht.2025.01.004