Científica, vol. 27, núm. 2, pp. 01-14, julio-diciembre 2023, ISSN 2594-2921, Instituto Politécnico Nacional MÉXICO
e270208 | DOI: 0.46842/ipn.cien.v27n2a08

ARTICLE

Implementación de una interfaz háptica en Unreal Engine y la estimación de velocidades para reducir vibraciones

Implementation of a haptic interface in Unreal Engine and the estimation of speeds to reduce vibrations

Erick Manuel **López-Ortiz**[1], Antonio Maximiliano **Hernández Salazar**[2]
Irandi **Gutierrez-Carmona**[3] Juan Gabino **Díaz-Martínez**[4]

Instituto Tecnológico y de Estudios Superiores de Monterrey, MÉXICO
[1] ORCID: 0000-0002-3297-0381 | a01552622@tec.mx
[2] ORCID: 0009-0001-9037-019X | a01412441@tec.mx
[3] ORCID: 0000-0002-0869-7418 | irandi_gutierrez@tec.mx
[4] ORCID: 0009-0007-8693-0784 | juan.diaz@tec.mx

Recibido 05-02-2023, aceptado 24-05-2023.

Resumen

Este trabajo aborda la implementación de una interfaz háptica en Unreal Engine en conjunto con un robot de un grado de libertad. Unreal Engine es un potente motor gratuito de renderizado y simulación en el que se desarrolló un gemelo digital capaz de interactuar con una plataforma física mediante una comunicación bidireccional de posición y torque para seguimiento de trayectoria y reflejo de fuerzas. Se condujeron diferentes experimentos para validar la integración tecnológica ante distintas condiciones de operación. El resultado fue una plataforma de bajo costo, con una sensación háptica aceptable, y un entorno digital para visualizar, analizar y comprender la tecnología de gemelos digitales y los desafíos que se enfrentan. Se validaron además distintas estrategias para estimar la velocidad y reducir significativamente las vibraciones en el sistema producto de retardos en la comunicación, pues estás afectan negativamente la precisión y la calidad de la interacción robótica.

Palabras clave: interfaz háptica, derivadores, tiempos de muestreo, Unreal Engine, plataforma experimental.

Abstract

This work addresses the implementation of a haptic interface in Unreal Engine in conjunction with a one-degree-of-freedom robot. Unreal Engine is a powerful free rendering and simulation engine in which a digital twin was developed capable of interacting with a physical platform through bidirectional communication of position and torque for trajectory tracking and force reflection. Different experiments were conducted to validate the technological integration under different operating conditions. The result was a low-cost platform with an acceptable haptic feel and a digital environment to visualize, analyze, and understand digital twin technology and the challenges faced. Several different methods were also tested to figure out how fast and significantly to lower the system's vibrations caused by communication delays, which are bad for the accuracy and quality of the robotic interaction.

Index terms: haptic interface, derivators, sampling times, Unreal Engine, experimental platform.

 Creative Commons License: Attribution-NonCommercial-ShareAlike 4.0 International CC BY-NC-SA 4.0

I. INTRODUCCIÓN

En el campo de la ingeniería y la simulación, los gemelos digitales se han convertido en una herramienta fundamental para comprender, analizar y optimizar el rendimiento de sistemas complejos [1]. Un gemelo digital es una réplica precisa de un objeto, proceso o sistema físico, que permite simular su comportamiento en tiempo real. Estos modelos digitales son cada vez más utilizados en diversas industrias, como la automotriz [2], aeroespacial [3], energética [4], manufacturera [5], interacción en el metaverso [6], entender el funcionamiento de ciudades inteligentes [7], así como para mejorar la toma de decisiones, predecir el rendimiento y reducir costos y tiempos de desarrollo.

Para desarrollar gemelos digitales efectivos y realistas, se requiere una plataforma de software que permita la creación, visualización y simulación de entornos digitales complejos [8]. En este sentido, Unreal Engine, desarrollado por Epic Games, ha emergido, siendo una plataforma gratuita, como una de las opciones más destacadas y poderosas para la construcción de gemelos digitales [9], [10]. Además, cuando se busca incorporar un componente físico real a la simulación, como un robot, Unreal Engine también ofrece soluciones integradas para una mayor interacción y precisión, pudiendo incluso importar modelos de programas de dibujo especializado como lo es SolidWorks. Además, es importante recordar que existen diferentes propuestas de arquitecturas de hardware y software de implementaciones de gemelos digitales [11], [12], [13]; pero con el uso de Unreal Engine pretendemos mantener la arquitectura de hardware y software de este proyecto con un bajo nivel de complejidad, a un costo económico asequible para laboratorios poco equipados, pero con un alto potencial de escalabilidad. Finalmente, es necesario precisar que esta propuesta se puede clasificar como una plataforma de monitoreo y el primero de 3 niveles de desarrollo para gemelos digitales [14], siendo Unreal Engine suficiente para el desarrollo tecnológico propuesto.

Unreal Engine es un motor de creación de videojuegos que ha evolucionado para convertirse en una solución completa y versátil para la creación de entornos digitales interactivos de alta calidad [15], [16] . Su amplia gama de características y herramientas lo hace ideal para la construcción de gemelos digitales inmersivos y precisos, con una capacidad de replicar entornos físicos reales que ha sido incluso considerada para el futuro del entrenamiento de cirugías en medicina [17]. Entre las características más destacadas se encuentran: renderización fotorrealista, física avanzada, simulación de materiales y efectos visuales de última generación.

Además de sus capacidades gráficas, Unreal Engine ofrece una interfaz de programación (API, *Application Programming Interface* por sus siglas en inglés) flexible y extensible que permite la integración de modelos matemáticos y algoritmos de simulación complejos. Esto proporciona a los desarrolladores la capacidad de crear modelos digitales altamente precisos y adaptados a las necesidades específicas de cada aplicación. Al mismo tiempo, si se desea combinar el gemelo digital con un robot físico, Unreal Engine ofrece soporte para controladores de hardware y protocolos de comunicación, lo que facilita la conexión y el control del robot en tiempo real.

En este artículo, exploraremos el uso de Unreal Engine en la construcción de gemelos digitales, incluyendo la incorporación de un robot físico. Analizaremos sus capacidades, la forma de estimar parámetros como velocidad y aceleración en el robot físico [18], enfocándonos en mostrar la forma de implementación. Cabe recalcar que en la literatura se han desarrollado algunos observadores para estimar parámetros internos y externos de los sistemas mecánicos [19] o limitaciones en la comunicación por el ancho de banda [20], pero en este artículo no discutimos estas estrategias. Asimismo, discutiremos los desafíos, como las oscilaciones que pueden ocurrir durante la operación del sistema [21], y consideraciones importantes al utilizar esta poderosa plataforma de software en combinación con un robot físico.

CC BY-NC-SA 4.0

En resumen, el uso de Unreal Engine en gemelos digitales, junto con la incorporación de un robot físico, ofrece una solución sólida y flexible para la simulación y visualización de sistemas complejos. Su combinación de capacidades gráficas avanzadas, física realista, una API extensible y soporte para controladores de hardware, lo convierte en una elección viable y eficiente para desarrollar gemelos digitales precisos y altamente funcionales en entornos que involucran la interacción entre lo digital y lo físico.

El artículo se compone de la siguiente manera: en la sección II se aborda la construcción de la plataforma experimental, en la sección III se presentan las ecuaciones de movimiento, en la sección IV la estrategia de control empleada, en la sección V la forma en que fueron estimados los parámetros del sistema, en la sección VI se enuncian los resultados experimentales y se brinda un enlace a los archivos generados, y en las secciones VII y VIII se escriben la discusión y conclusiones respectivamente.

II. PLATAFORMA EXPERIMENTAL

En esta sección se describe la plataforma física operada mediante un motor de corriente directa, y un brazo mecánico (véase Fig. 1), así como una interfaz desarrollada en Unreal Engine (véase Fig. 2). Las características de hardware y software del equipo empleado en este experimento son:

- Computadora de escritorio con
  - Sistema operativo Windows 10
  - Unreal Engine 5
  - 8 GB de memoria RAM.
  - Tarjeta de video Nvidia de 8 GB.
  - Disco duro de estado sólido de 1TB
- Motor de engranaje sin escobillas GIM4305 DC, velocidad de engranaje de 53mm, 10:1, 300rpm.
- Módulo Mcp2515 para la interfaz entre un Bus CAN (*Controller Area Network* por sus siglas en inglés) y SPI (*Serial Peripheral Interface* por sus siglas en inglés) para comunicación Arduino-Motor.
- Arduino NANO.
- Un brazo robótico impreso en plástico PLA (*Poliácido Láctico*) con una longitud de 20 cm y un peso aproximado de 80 g, se puede observar en la Fig. 1 los detalles del diseño mecánico.

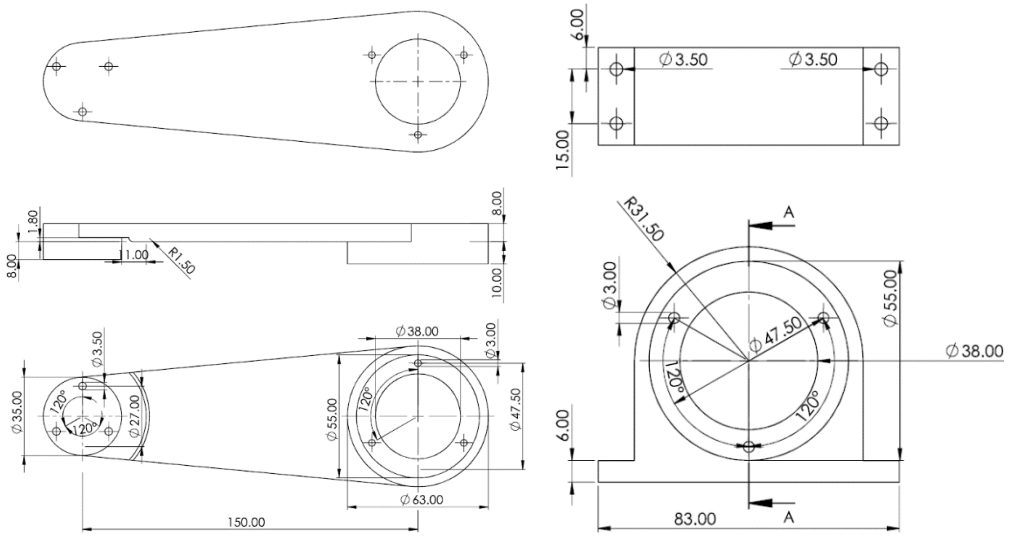


Fig. 1. Diseño mecánico del brazo robótico

CC BY-NC-SA 4.0

III. DINÁMICA DEL SISTEMA

Ahora bien, la dinámica del sistema real es dictada por las leyes físicas y actuada mediante la corriente eléctrica en el motor, en el otro sistema, el digital, se programó la ecuación dinámica de movimiento (1) para lograr un comportamiento similar al de un péndulo simple, en esta ecuación de movimiento:

$$J\ddot{q_d}(t) + mgl \sin q_d(t) + c\dot{q_d}(t) = u_d(t) \tag{1}$$

la variable $J \in \mathbb{R}$ es la inercia de la barra, $m \in \mathbb{R}$ la masa total, $g \in \mathbb{R}$ la constante de gravedad, $l \in \mathbb{R}$ la distancia del pivote al centro de masa de la barra, $c \in \mathbb{R}$ el coeficiente de fricción, $u \in \mathbb{R}$ la acción de control o torque aplicado al pivote, y $q_d \in \mathbb{R}$ la posición angular del sistema digital de la barra medida con respecto al punto de equilibrio inferior (siendo $\dot{q}_d$ y $\ddot{q}_d$ la primera y segunda derivada de la posición del sistema digital).

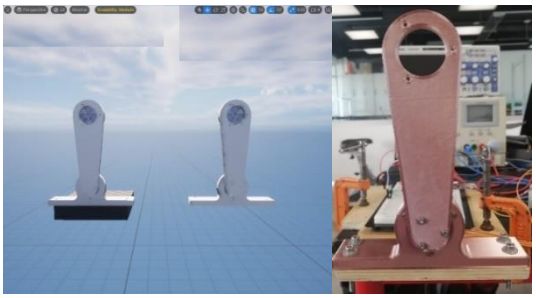


Fig. 2. a) Prototipo digital en Unreal Engine; b) Plataforma física experimental.

El diagrama de conexiones eléctricas y el flujo de información se muestran en la Fig.3. La lógica de operación es la siguiente:

1. En la plataforma de Unreal Engine se simula el comportamiento de un péndulo simple. Se recibe una señal de posición de referencia del Arduino, proveniente del motor físico, con la cual se calcula el torque implementado al sistema digital.
2. El Arduino recibe la posición de referencia de la plataforma Unreal Engine, y la lectura de posición del motor, lo que le permite calcular y enviar el torque necesario al sistema físico.

Es importante observar que mientras que las variables de posición, velocidad y aceleración del sistema digital son medibles en todo instante, la velocidad y aceleración del sistema físico solo pueden ser estimadas a partir de las mediciones del codificador rotacional del motor (encoder).

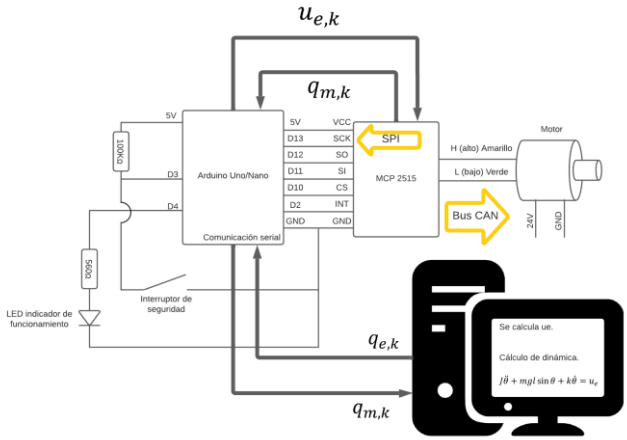


Fig. 3. Diagrama de conexión y comunicación entre los distintos componentes.

CC BY-NC-SA 4.0

En la Fig. 4 se muestra la plataforma experimental cableada y los componentes empleados en la instrumentación. El motor fue seleccionado por tener un alto torque, y un controlador que integrado facilitó su instrumentación. El Arduino Nano cuenta con el suficiente poder de cómputo, así como puertos de comunicación $i2C$, así como SPI, para comunicación con el motor y con Unreal Engine. Finalmente, el módulo MCP2515 es un recurso necesario para poder comunicar el controlador del motor con el Arduino.

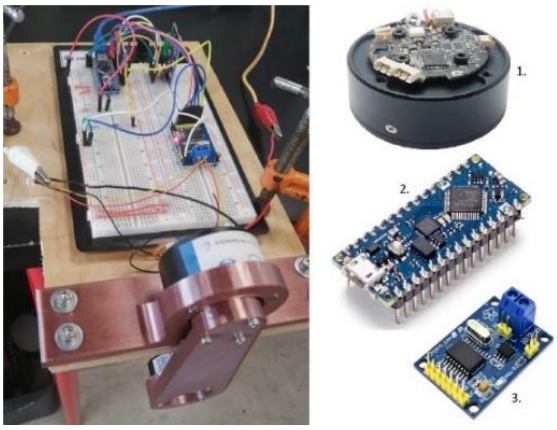


Fig. 4. a) Conexión física de componentes; b) Módulos: 1. Motor GIM4305, 2. Arduino NANO, 3. Módulo CAN BUS-SPI Mcp251.

Nota: La plataforma digital consta de dos modelos 3D del mismo péndulo (véase Fig. 2a), uno para verificar la integridad de datos recibidos (movimiento cinemático) y el otro para responder a sus propias ecuaciones diferenciales y a la estrategia de control para una sensación háptica (movimiento dinámico).

IV. ESTRATEGIA DE CONTROL

Es usual pensar en los sistemas hápticos en términos de dos tareas, seguimiento de posición y transparencia [19]. Mientras que la primera tarea es relativamente simple y requiere de una correcta sintonización de las ganancias de los controladores [20], la segunda tarea es un poco más compleja y requiere de la estimación de fuerzas externas para establecer estrategias de compensación y anular el efecto de fuerzas [21].

En esta sección presentaremos la estrategia de control empleada para el seguimiento de posición, la cual consiste en colocar un control proporcional (P) a la diferencia entre la plataforma física y digital, que emula un resorte entre ambos sistemas cuya rigidez depende de la ganancia de la acción de control, además de considerar un término de fricción viscosa para eliminar las oscilaciones del sistema mecánico el cual puede ocurrir por varias razones, entre las que destacan: retardos en la comunicación, pérdida de datos, errores numéricos en la simulación, así como fuerzas de restitución en el sistema mecánico. Entonces, se hace necesario presentar una estrategia para la correcta estimación de velocidad a partir de la medición de la posición de los motores con el uso de codificadores rotacional.

Además, la acción de control no busca solo un seguimiento de trayectorias bidireccional, sino que busca reproducir las fuerzas que el gemelo está experimentado. Una acción de control proporcional, aunque simple tiene un significado mecánico claro, esto es, un resorte con cierta rigidez que conecta ambos sistemas. Implementar alguna otra estrategia de control como una acción proporcional-integral (PI) o una acción proporcional-integral-derivativo (PID) podría causar perdida en la sensación háptica. Mientras que la acción derivativa se puede interpretar como la conexión de un amortiguador generando una sensación viscosa entre ambos sistemas, la acción integral pudiera provocar, que al aumentar o disminuir la integral numérica del error,

CC BY-NC-SA 4.0

se perciba un efecto de cambio de rigidez en ambos mecanismos. Experimentalmente los mejores resultados se observaron con la estrategia de control más simple, esto es la acción proporcional.

A. Ecuación controladores

Tanto el control $u_d \in \mathbb{R}$ para el sistema digital, como el control $u_f \in \mathbb{R}$ para el sistema físico tienen la siguiente estructura:

$$u_f(t) = K_1\left(q_d(t) - q_f(t)\right) - K_2\,\dot{q}_f(t) \tag{2}$$

$$u_d(t) = K_3\left(q_f(t) - q_d(t)\right) - K_4\,\dot{q}_d(t) \tag{3}$$

donde $q_f \in \mathbb{R}$ es la posición del sistema físico. Las ganancias $\{\square_1, \square_3\}$, representan la rigidez de los resortes que conectan a ambos motores, y aunque es usual que $K_1 = K_3$, es posible ajustar estos valores de forma heurística para obtener un mejor desempeño; entre más se incrementa el valor de estas ganancias más rígido se vuelve la conexión entre ambos sistemas. Las ganancias $\{\square_2, \square_4\}$ representan la capacidad del sistema a disipar energía y reducir oscilaciones, entre mayor sean estas ganancias más rápido se reducirán oscilaciones no deseadas, pero valores muy altos provocarán un sistema muy rígido y poco ágil; además, la eficacia de este mecanismo de disipación depende de la precisión en la que podemos medir o estimar el valor de la velocidad del sistema, pues estimaciones incorrectas o ruidosas provocarán oscilaciones en el sistema en vez de reducirlas llegando a producir inestabilidad en el comportamiento del sistema.

B. Implementación de señales de control en la plataforma experimental

En la plataforma Arduino, fue programada la ecuación (4):

$$u_f[k] = K_1\left(q_d[k] - q_f[k]\right) - K_2\,\hat{\dot{q}}_f[k] \tag{4}$$

mientras que la ecuación (5) se programó en Unreal Engine,

$$u_d[k] = K_3\left(q_f[k] - q_d[k]\right) - K_4\,\dot{q}_d[k] \tag{5}$$

Observe que hemos sustituido la variable continua tiempo (t) para dar paso a la $[k]-$ésima iteración para expresar la discretización que ocurre en los sistemas digitales; también, se sustituyó la velocidad $\dot{q}_f$ por una aproximación numérica $\hat{\dot{q}}_f$ que resulta de solamente poder medir la posición de los motores por medio de un codificador rotacional, debiendo reconstruir de alguna forma la velocidad. Las estrategias y códigos para estimar la velocidad se presentan en la siguiente sección; además se presentarán las estimaciones de la aceleración que, pese a no utilizarse en este trabajo, serán útiles para aquellos que quieran estimar las fuerzas externas que actúan sobre los motores.

V. ESTRATEGIAS DE ESTIMACIÓN DE VELOCIDADES Y ACELERACIONES EN PLATAFORMA EXPERIMENTAL

Existen muchas estrategias para obtener la estimación de la velocidad, aquí presentamos tres estrategias que nos permiten obtener de forma simple la derivada de una señal, además de entender algunos aspectos sobre la sintonización de las ganancias.

CC BY-NC-SA 4.0
Científica, vol. 27, núm. 2, pp. 01-14, julio-diciembre 2023, ISSN 2594-2921, Instituto Politécnico Nacional MÉXICO
e270208 | DOI: 10.46842/ipn.cien.v27n2a08

A. Método de Euler

El método de Euler es ampliamente conocido, en este trabajo solo se presenta con fines comparativos. Recordemos que, en este método se establece que para una función diferenciable $f(t)$ su derivada está definida como:

$$\dot{f}(t) = \lim_{T \to 0} \frac{f(t) - f(t-T)}{T} \tag{6}$$

Y para fines de implementación en la plataforma Arduino, la constante T se considera como el tiempo de muestreo y el algoritmo numérico se puede escribir como:

$$\dot{f}[k] = \frac{f[k] - f[k-1]}{T} \tag{7}$$

B. Derivador con filtro pasa baja

Consideremos ahora $x(t) \in \mathbb{R}$ una señal diferenciable, y sea $y(t) = \dot{x}(t)$ la derivada que deseamos estimar. En el dominio de la frecuencia se tiene que $Y(s) = L\{\dot{x}\} = sX(s)$, la cual tiene la función de transferencia no realizable:

$$G(s) = \frac{Y(s)}{X(s)} = s \tag{8}$$

que, al ser multiplicada por una función propia de primer orden asociada a un filtro pasa-baja permite obtener una función realizable de la forma:

$$Y(s) = \frac{a}{s+a} sX(s) = \frac{a}{1 + \frac{a}{s}} X(s) \tag{9}$$

Ahora, para obtener un diferenciador de segundo orden, basta multiplicar dos veces la ecuación (9), para obtener la ecuación:

$$Y(s) = \frac{a^2}{1 + 2\frac{a}{s} + \frac{a^2}{s^2}} X(s) \tag{10}$$

Empleando el cambio de variable $\ddot{\hat{x}} = y(t)$ en el dominio del tiempo, de la ecuación (10) se obtiene la ecuación diferencial de segundo orden:

$$\ddot{\hat{x}}(t) = a^2(x(t) - \hat{x}(t)) - 2a\dot{\hat{x}}(t) \tag{11}$$

La cuál es estable para valores positivos de a, y se observa que $\hat{x} \to x$ exponencialmente por lo que las derivadas convergen exponencialmente $\dot{\hat{x}} \to \dot{x}$ y $\ddot{\hat{x}} \to \ddot{x}$ en una vecindad. Para verificar esta propiedad, se puede definir la variable de error $\xi(t) \in \mathbb{R}$ como como $\xi(t) = |x - \hat{x}|$, y verificar que se cumple que :

$$\xi(t) \to 0 \Rightarrow \dot{\xi}(t) \leq \epsilon_1 \text{ así como } \ddot{\xi}(t) \leq \epsilon_2 \tag{12}$$

CC BY-NC-SA 4.0

1) Implementación numérica en nuestra plataforma experimental

Para implementar el derivador de segundo orden, recordemos que la transformada Z de una integral definida, por ejemplo, $\omega(t) = \int_0^t \dot{\omega}(\tau)d\tau$, con condiciones iniciales $\omega(0) = 0$, asociada al método trapezoidal, se puede escribir como:

$$\Omega(z) = \frac{T}{2}\frac{1+z^{-1}}{1-z^{-1}}\dot{\Omega}(z), \Rightarrow \Omega[k] = \Omega[k-1] + \frac{T}{2}\left(\dot{\Omega}[k] + \dot{\Omega}[k-1]\right), \tag{13}$$

Donde, nuevamente, T es el tiempo de muestreo en segundos del sistema. Por lo que nuestro derivador de segundo orden puede ser escrito de forma discreta como:

$$\begin{aligned} \ddot{\hat{x}}[k] &= a^2(x[k-1] - \hat{x}[k-1]) - 2a\dot{\hat{x}}[k-1], \\ \dot{\hat{x}}[k] &= \dot{\hat{x}}[k-1] + \frac{T}{2}\left(\ddot{\hat{x}}[k] + \ddot{\hat{x}}[k-1]\right), \\ \hat{x}[k] &= \hat{x}[k-1] + \frac{T}{2}\left(\dot{\hat{x}}[k] + \dot{\hat{x}}[k-1]\right). \end{aligned} \tag{14}$$

De la ecuación anterior se vuelve un como más claro el papel de la constante a, usado para definir la frecuencia de corte del filtro pasaba baja en la ecuación (10), entre más alto sea el valor de a, mayor será el ajuste de la aceleración ante un error en la estimación de la posición lo que permitirá reaccionar más rápido ante variaciones de la señal y responder ante altas frecuencias; si, por el contrario, el parámetro a es pequeño observaremos ajustes pequeños de la aceleración siendo ideal para bajas frecuencias.

Al implementar este algoritmo se debe conocer en que frecuencias opera el sistema para la correcta sintonización del parámetro a.

C. Observadores de alta ganancia

Finalmente, construiremos un derivador empleando un observador de alta ganancia que es formulado a partir del conocimiento de la planta del sistema. Dado que la aceleración del sistema es proporcional al torque τ del sistema, esto es $\ddot{x}(t) = k\tau(t)$, podemos escribir el sistema dinámico, con $\boldsymbol{x} \in \mathbb{R}^2$, en forma matricial como:

$$\dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t) + B\tau(t), y(t) = C\boldsymbol{x}(t) \tag{15}$$

donde $\boldsymbol{x}(t) = [x(t) \quad \dot{x}(t)\,]^T = [x_1(t) \quad x_2(t)]^T$, $A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, y $C = [1 \quad 0]$. Como la matriz de observabilidad $O = \begin{bmatrix} C \\ CA \end{bmatrix}$ es de rango completo, el sistema es observable y podemos construir un observador con la ecuación dinámica dada como:

$$\dot{\hat{\boldsymbol{x}}}(t) = A\hat{x}(t) + H\left(y(t) - C\hat{\boldsymbol{x}}(t)\right) = A\hat{\boldsymbol{x}}(t) + HC\left(\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)\right), \tag{16}$$

donde $\hat{\boldsymbol{x}} = [\hat{x}_1 \quad \hat{x}_2]^T$, y $H = [h_1 \quad h_2]^T$. Definiendo el error $\boldsymbol{\xi}(t) = \boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)$, se puede construir la ecuación dinámica del error:

$$\dot{\boldsymbol{\xi}}(t) = A_0\boldsymbol{\xi}(t) + B\tau, \tag{17}$$

CC BY-NC-SA 4.0

Donde $A_0 = \begin{bmatrix} -h_1 & 1 \\ -h_2 & 0 \end{bmatrix}$. Con una función de transferencia de τ a $\boldsymbol{\xi}$ como:

$$G_\tau(s) = \frac{k}{s^2 + h_1 s + h_2}\begin{bmatrix} 1 \\ h_1 + s \end{bmatrix} \tag{18}$$

Que al seleccionar $h_2 \gg h_1 \gg 1$, de la forma:

$$h_1 = \frac{\alpha_1}{\varepsilon}, h_2 = \frac{\alpha_2}{\varepsilon^2} \tag{19}$$

La ecuación puede ser escrita como:

$$G_\tau(s, \varepsilon) = \frac{\varepsilon k}{(\varepsilon s)^2 + \alpha_1 \varepsilon s + \alpha_2}\begin{bmatrix} \varepsilon \\ \varepsilon s + \alpha_1 \end{bmatrix} \tag{20}$$

por lo tanto $G_\tau(s, \varepsilon) \rightarrow 0$, las altas ganancias rechazan la perturbación y

$$\boldsymbol{\xi}(t) \rightarrow 0 \Rightarrow \hat{x}_1 \rightarrow x, \hat{x}_2 \rightarrow \dot{x} \tag{21}$$

Lo que nos permite observar la velocidad directamente, mientras que la aceleración será entendida como $\dot{\hat{x}}_2$.

1) Implementación numérica en nuestra plataforma experimental

Expandiendo la ecuación del observador, se tiene:

$$\begin{bmatrix} \dot{\hat{x}}_1 \\ \dot{\hat{x}}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} + \begin{bmatrix} h_1 \\ h_2 \end{bmatrix}(x - \hat{x}_1) \tag{22}$$

Por lo que podemos escribir las ecuaciones discretas de la siguiente forma:

$$\begin{aligned} \dot{\hat{x}}_1[k] &= \hat{x}_2[k-1] + h_1(x_1[k-1] - x_1[k-1]), \\ \dot{\hat{x}}_2[k] &= h_2(x_1[k] - x_1[k-1]), \\ \hat{x}_1[k] &= \hat{x}_1[k-1] + \frac{T}{2}\left(\dot{\hat{x}}_1[k] + \dot{\hat{x}}_1[k-1]\right), \\ \hat{x}_2[k] &= \hat{x}_2[k-1] + \frac{T}{2}\left(\dot{\hat{x}}_2[k] + \dot{\hat{x}}_2[k-1]\right). \end{aligned} \tag{23}$$

VI. RESULTADOS EXPERIMENTALES

Para estudiar el comportamiento del robot háptico, se levantó el brazo del robot a una posición de $2.5\ rad$, ver Fig. 5, y se dejó caer para observar las distintas respuestas del sistema bajo condiciones iniciales similares, pero distintas ganancias en las estrategias de control (4) y (5). En todos los experimentos, como era de esperarse, las dinámicas del robot y su gemelo digital hacían un mutuo seguimiento de trayectorias y sus dinámicas se interrelacionaban. La velocidad de comunicación entre todos los dispositivos se ajustó a 115,200 baudios, en donde se observó que a menores velocidades de comunicación el rendimiento del sistema también disminuía.

CC BY-NC-SA 4.0

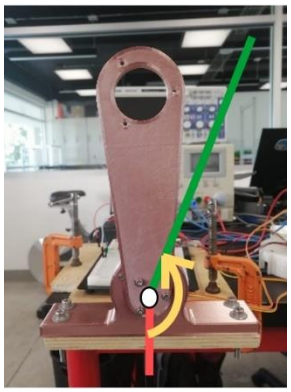
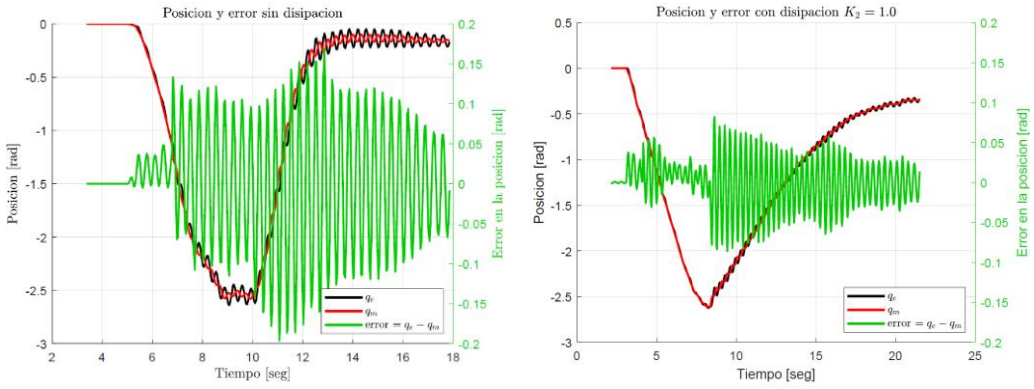


Fig. 5. El brazo se coloca a una posición inicial de 2.5 rad, se permite que el brazo se mueva por efecto de la gravedad y se observa como la dinámica del gemelo digital afecta y es afectada.

En la Fig. 6a se observa el experimento con ganancias $K_2 = K_4 = 0$, mientras que en la Fig. 6b se observa con ganancias $K_2 = 1.0$ y $K_4 = 0$; observe que el sistema digital no tiene la capacidad de disipar energía y solo es activado o desactivado este atributo en el sistema físico. En el experimento donde no se disipa energía se presentaron oscilaciones no deseadas de alta frecuencia las cuales pueden incluso provocar daños mecánicos en los mecanismos. En el sistema donde se consideró disipación en el motor físico, se observa que las oscilaciones disminuyen, además de lograr un mejor seguimiento de trayectorias, pero esta fricción adicional evita que el sistema alcance el punto de equilibrio inferior.



x

Fig. 6. a) Comportamiento del sistema sin compensación de fricción; b) Comportamiento del sistema con compensación de fricción empleando el método de filtro pasa baja con $a = 6$.

En la Fig. 7, se observa la estimación de velocidades en las dos condiciones experimentales, con y sin fricción. Mientras que el método de Euler presenta oscilaciones que hacen inviable su implementación, el uso de filtros pasa bajas nos brinda una señal con menos oscilaciones. El observador de alta ganancia, así como el derivador con filtro pasa baja con ganancia adecuadamente sintonizadas resultan ser útiles para el objetivo de este trabajo y experimentalmente no se observaron grandes diferencias.

CC BY-NC-SA 4.0

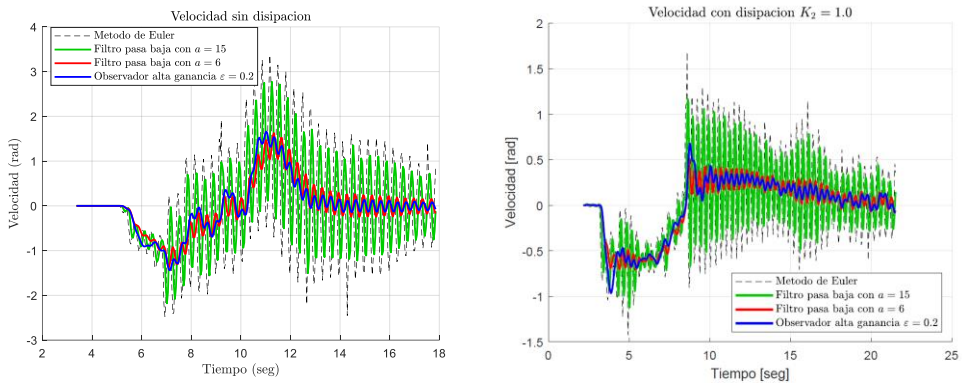


Fig. 7. a) Estimación de la velocidad sin compensación de fricción; b) Estimación de la velocidad con compensación de fricción.

En la Fig. 8, se muestra la aproximación a la segunda derivada. Mientras que el método de Euler y el filtro pasa bajas con $a = 15$ presentan oscilaciones de gran amplitud en ambos experimentos, el observador de alta ganancia, así como el filtro pasa bajas con $a = 6$ generan señales de amplitud más pequeñas.

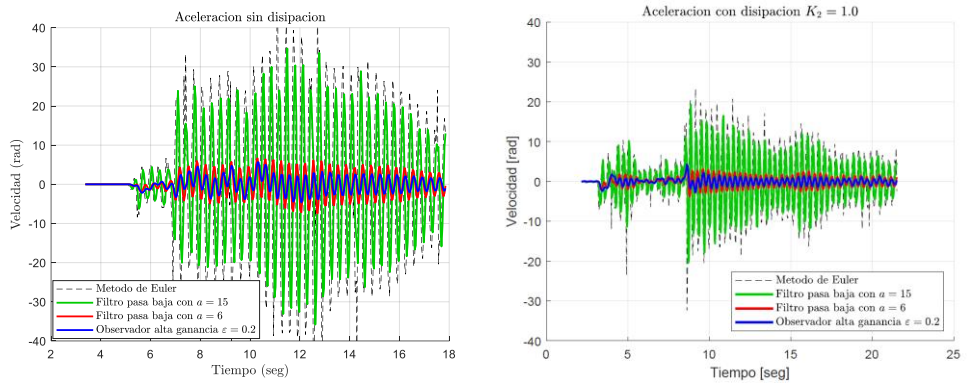


Fig. 8. a) Estimación de la aceleración sin compensación de fricción; b) Estimación de la aceleración con compensación de fricción.

Finalmente, pese a que en los métodos presentados se busca estimar la velocidad y la aceleración, no se cuenta con sensores para medir directamente el valor correcto de estas señales, pero observando que de cada método se puede reconstruir la posición del motor, parámetro que es conocido, nos permite proponer una estrategia para comparar las técnicas de derivación presentadas. En la Fig. 9 se observa la posición medida y la posición estimada, así como el error en cada uno de los métodos. Cualitativamente se puede observar que la mejor aproximación se logra con el método de alta ganancia. Cuantitativamente, se empleó la norma cuadrática:

$$\|x_1 - \hat{x}_1\|_2 = \left(\sum_{k=1}^{n} x_1[k] - \hat{x}_1[k] \right)^{1/2} \tag{20}$$

CC BY-NC-SA 4.0

para estimar la desviación entre la posición estimada y la medida e identificar el mejor método de diferenciación. Los resultados obtenidos se muestran en la Tabla 1, en donde se observa que el error con la menor norma es el de alta ganancia.

TABLA 1.
TABLA COMPARATIVA ENTRE LA NORMA DEL ERROR DE LOS DISTINTOS MÉTODOS DE DIFERENCIACIÓN.

EXPERIMENTO	$\|x_1 - \hat{x}_1\|_2$
CON FILTRO PASA BAJA $a = 15$	1.9553
CON FILTRO PASA BAJA $a = 6$	3.6803
CON ALTA GANANCIA	1.4217

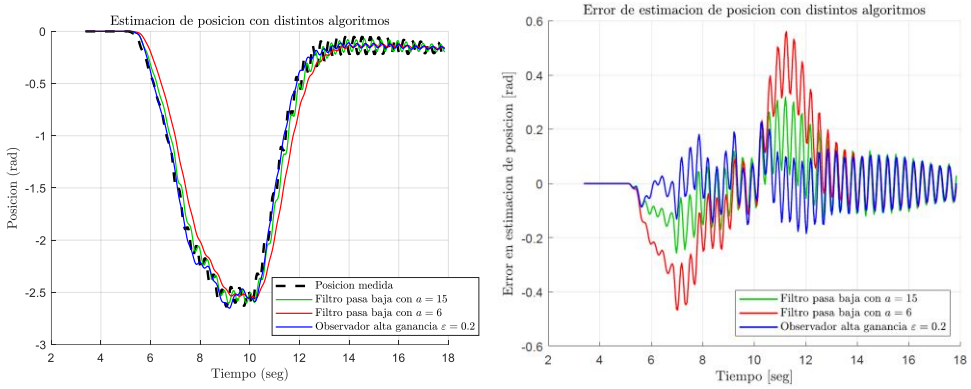


Fig. 9. a) Reconstrucción de la posición a partir de los diferentes métodos de estimación de velocidad. b) Error en la reconstrucción de la posición a partir de la estimación de velocidad.

Finalmente, los archivos que se generaron durante este experimento se pueden encontrar en el repositorio de archivos de GitHub [15].

VII. DISCUSIÓN

En los experimentos realizados, se demostró la viabilidad de la propuesta tecnológica para conseguir una sensación háptica entre un robot físico y la plataforma digital Unreal Engine. Los diagramas para la implementación electromecánica, el diseño digital y algoritmos computacionales se proveen para la reproducibilidad del experimento en el repositorio en línea. La estrategia de interconexión entre ambos sistemas se establece como un control proporcional, lo que equivale a conectar un "resorte digital" entre ambos mecanismos. En la Fig. 6 se observa la capacidad del sistema para seguir la posición de su gemelo, hay que recordar que este no es un sistema obligado a "mantener la posición", pues debe ser capaz de responder a los estímulos externos de su propio entorno, lo que hace poco práctico implementar algunas otras estrategias de control robusto convencionales.

CC BY-NC-SA 4.0

Los algoritmos presentados son simples para implementarse en plataformas de bajo rendimiento como lo es Arduino, recordemos que existen propuestas de otros algoritmos más complejos [19] [20] pero cuya implementación es más compleja, o bien estén pensadas para plataformas electromecánicas de mayor costo y sistemas de cómputo con características especiales.

Al querer mantener esta plataforma simple, la mera retroalimentación de posiciones no es suficiente pues, como ha sido ampliamente documentado en otros trabajos, retardos en la comunicación, sistemas mecánicos con mucha inercia, ganancias mal sintonizadas, poca fricción mecánica, o inclusive cambios de rigidez producidos por la interacción humana [21] puede provocar oscilaciones que pueden prevalecer una gran cantidad de tiempo y dañar al sistema. Una estimación adecuada de la velocidad (véase Fig. 7) es necesaria pues solo de esta forma podremos reducir las oscilaciones y lograr una mejor operación.

Cabe resaltar que, aunque en la Fig. 7 y Fig. 8 se muestran las estimaciones de velocidad y aceleración, en la práctica no existe una forma de compararlas con los valores reales, pues no se consideró un sensor para tal propósito; sin embargo, puesto que de estas medidas se puede reconstruir nuevamente la posición, podemos establecer una medida sobre qué tan adecuados son los métodos numéricos presentados.

VIII. Conclusión

En el presente trabajo se desarrolló un robot de un grado de libertad y su gemelo digital empleando el software Unreal Engine, donde una transmisión bidireccional de datos permitió generar una sensación háptica entre ambos dispositivos. El prototipo mantuvo una construcción simple y a un bajo costo, lo cual permite su utilización tanto para la divulgación de la ciencia y tecnología como plataforma de investigación. Cabe resaltar que el diseño fue logrado al emplear la interfaz de Unreal Engine la cual, optimizada para el desarrollo de videojuegos, cuenta con gráficos visualmente atractivos y permite la programación de las ecuaciones dinámicas de movimiento para un comportamiento adecuado del sistema mecánico de interés; además, la selección de los actuadores permitió mantener un diseño electromecánico simple que puede ser construido con cualquier sistema de impresión 3D. Se implementó un control proporcional para lograr una sensación háptica, siendo la rigidez de conexión entre los mecanismos ajustada mediante la ganancia de la ley de control, implementar estrategias de control PI y PID puede lograr un mejor seguimiento de posición, pero añadiría una sensación de fricción viscosa entre ambos mecanismos o fuerzas que cuya interpretación física no es clara.

Además, se presentaron algoritmos para le estimación de la velocidad y aceleración a partir de la medición de la posición. Mientras que la velocidad estimada fue empleada para añadir fricción viscosa al sistema mecánico y reducir oscilaciones comúnmente encontradas en este tipo de aplicaciones, la estimación de la aceleración se presentó para mostrar las diferencias en los distintos diferenciadores propuestos; eventualmente, la estimación de la aceleración puede servir para mejorar la sensación háptica al transformar esta aceleración en información de fuerza que actúa sobre el sistema.

Referencias

[1] C. Semararo, M. Lezoche, P. Hervé, M. Dassisti, “Digital twin paradigm: A systematic literature review”, *Computers in Industry*, vol. 130, p. 103-469, 2021.

[2] Z. Wang, X. Liao, X. Zhao, K. Han, P. Tiwari, M. J. Barth, G. Wu, “A digital twin paradigm: Vehicle-to-cloud based advanced driver assistance systems”, en *2020 IEEE 91st Vehicular Technology Conference* (VTC2020-Spring), 2020.

[3] E. Glaessgen, D. Stargel, “The digital twin paradigm for future NASA and US Air Force vehicles”, en *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 20th AIAA/ASME/AHS adaptive structures conference 14th AIAA*, 2012.

CC BY-NC-SA 4.0

[4] W. Yu, P. Panos, B. Young, E. Klinac, T. G. Walmsley, "Energy digital twin technology for industrial energy management: Classification, challenges and future", *Renewable and Sustainable Energy Reviews*, vol. 161, pp. 112-407, 2022.
[5] W. Kritzinger, M. Karner, G. Traar, J. Henjes, W. Sihn, "Digital Twin in manufacturing: A categorical literature review and classification", *Ifac-PapersOnline*, vol. 51, pp. 1016-1022, 2018.
[6] M. Faisal, F. Laamarti, A. El Saddik, "Digital Twin Haptic Robotic Arms: Towards Handshakes in the Metaverse", *Electronics*, vol. 12, 2023.
[7] A. Fuller, Z. Fan, C. Day, C. Barlow, "Digital twin: Enabling technologies, challenges and open research", *IEEE access*, vol. 8, pp. 108952-108971, 2020.
[8] C. K. Liu, D. Negrut, "The role of physics-based simulators in robotics", *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 4, pp. 35-58, 2021.
[9] J. Resch, J. Ehrentraut, M. Barnett-Cowan, et al., "Gamified automation in immersive media for education and research",» *arXiv preprint*:1901.00729, 2018.
[10] F. Jiang, Q. Hao, "Pavilion: Bridging Photo-Realism and Robotics", en *2019 International Conference on Robotics and Automation (ICRA)*, 2019.
[11] Y. Zheng, S. Yang, H. Cheng, "An application framework of digital twin and its case study", *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, pp. 1141-1153, 2019.
[12] G. Steindl, M. Stagl, L. Kasper, W. Kastner, R. Hofmann, "Generic digital twin architecture for industrial energy systems", *Applied Sciences*, vol. 24, p. 8903, 2020.
[13] Y. Lu, C. Liu, I. Kevin, K. Wang, H. Huang, X. Xu, "Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues", *Robotics and computer-integrated manufacturing*, vol. 61, p. 101837, 2020.
[14] C. Boje, A. Guerriero, S. Kubicki, Y. Rezgui, "Towards a semantic Construction Digital Twin: Directions for future research", *Automation in construction*, vol. 114, p. 103179, 2020.
[15] J. Leudet, F. Christophe, T. Mikkonen, T. Mannisto, "Ailivesim: An extensible virtual environment for training autonomous vehicles", en *2019 IEEE 43rd annual computer software and applications conference (COMPSAC)*, 2019.
[16] S. Jeršov, "Development of Digital Twin in extended Reality with Unreal Engine", Tesis, Tal Tech- School of Information Technologies, Estonia, 2020.
[17] Y. A. El-Wajeh, P. V. Hatton, N. J. Lee, "Unreal Engine 5 and immersive surgical training: translating advances in gaming technology into extended-reality surgical simulation training programmes", *British Journal of Surgery*, vol. 109, pp. 470-471, 2022.
[18] Y. Wang, G. Zheng, D. Efimovk, W. Perruquetti, "Differentiator application in altitude control for an indoor blimp robot", *International Journal of Control*, vol. 91, pp. 2121-2130, 2018.
[19] Y. Matsumoto, S. Katsura, K. Ohnishi, "An analysis and design of bilateral control based on disturbance observer", en *IEEE International Conference on Industrial Technology 2003*, vol. 2, pp. 802-807, 2003.
[20] D. Yashiro, K. Ohnishi, "Performance Analysis of Bilateral Control System With Communication Bandwidth Constraint", *IEEE Transactions on Industrial Electronics*, vol. 58, pp. 436-443, 2011.
[21] H. Z. Tan, M. A. Srinivasan, B. Eberman, B. Cheng, "Human factors for the design of force-reflecting haptic interfaces", *Dynamic Systems and Control*, vol. 55, pp. 353-359, 1994.
[22] H. Li, L. Zhang, K. Kawashima, "Operator dynamics for stability condition in haptic and teleoperation system: A survey", *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 14, p. e1881, 2018.
[23] A. Colomé, D. Pardo, G. Alenya, C. Torras, "External force estimation during compliant robot manipulation", en *2013 IEEE International Conference on Robotics and Automation*, 2013.
[24] Repositorio de GITHUB: https://github.com/erick-man11/InterfazHaptica_UE5, 2019.

CC BY-NC-SA 4.0

Científica, vol. 27, núm. 2, pp. 01-14, julio-diciembre 2023, ISSN 2594-2921, Instituto Politécnico Nacional MÉXICO
e270208 | DOI: 10.46842/ipn.cien.v27n2a08