Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática Industrial 21 (2024) 180-191

www.revista-riai.org

Identificación y control de robots paralelos en el espacio de estados con un laboratorio remoto

Adrián Peidró[a, *], Luis Payá[a], Mónica Ballesta[a], Arturo Gil[a], Óscar Reinoso[a, b]

[a] *Instituto Universitario de Investigación en Ingeniería I3E, Universidad Miguel Hernández de Elche, Avda. Universidad s/n, 03202 Elche, España.*
[b] *Valencian Graduate School and Research Network for Artificial Intelligence, Valencian Community, Spain.*

To cite this article: Peidró, A., Payá, L., Ballesta, M., Gil, A., Reinoso, O., 2024. Identification and control of parallel robots in the state space with a remote laboratory. Revista Iberoamericana de Automática e Informática Industrial 21, 180-191. https://doi.org/10.4995/riai.2024.20065

Resumen

En este artículo se presenta un laboratorio remoto para realizar prácticas de identificación y control de dos robots paralelos reales mediante Internet. El laboratorio remoto tiene una arquitectura cliente-servidor. El cliente es una interfaz de control Java que permite comandar experimentos y visualizar los resultados en forma de gráficas y de imágenes de video en tiempo real de los robots. Por su parte, el servidor es un computador industrial que controla a los robots mediante una tarjeta de control dSPACE 1103 programada con Matlab/Simulink, y además ejecuta Jimserver y otros programas de gestión que permiten reservar y utilizar los robots de forma segura. El artículo describe e ilustra varias prácticas y experimentos que pueden realizarse con el laboratorio remoto presentado, como la identificación del modelo de estado linealizado de los robots, el diseño de reguladores integrales con realimentación del estado, o estudiar cómo las singularidades de los robots paralelos disminuyen su controlabilidad.

Palabras clave: robot paralelo, laboratorio remoto, control en espacio de estados, identificación, controlabilidad, singularidades.

Identification and control of parallel robots in the state space with a remote laboratory

Abstract

This article presents a remote laboratory to carry out identification and control experiments of two real parallel robots over the Internet. The remote laboratory has a client-server architecture. The client is a Java control interface that allows the user to command experiments and visualize the results in the form of graphs and live feed of the robots. The server is an industrial computer that controls both robots through a dSPACE 1103 control board programmed with Matlab/Simulink, and also runs Jimserver and other programs that allow the users to reserve and safely use the robots. The article describes and illustrates various laboratory practices and experiments that can be carried out with the presented remote laboratory, such as the identification of the linearized state-space model of the robots, the design of integral regulators with state feedback, or study how singularities of parallel robots decrease their controllability.

Keywords: parallel robot, remote laboratory, state-space control, identification, controllability, singularities.

1. Introducción

Los laboratorios virtuales y remotos son plataformas software y ciber-físicas que permiten al estudiantado realizar prácticas y experimentos a través de Internet, evitando las clásicas limitaciones de los laboratorios presenciales tradicionales, que únicamente permiten al estudiantado realizar prácticas en horarios específicos y desplazándose en persona a dichos laboratorios, lo que sin duda alguna supone una menor flexibilidad. Los laboratorios virtuales permiten al estudiantado experimentar con un sistema completamente simulado, mientras que los laboratorios remotos le permiten experimentar con equipamiento físico real que se encuentra en las dependencias de la universidad. Cuando se desea que el estudiantado reciba una experiencia de aprendizaje más próxima a la que recibiría al realizar prácticas presenciales con

*Autor para correspondencia: apeidro@umh.es

equipos reales, se opta por los laboratorios remotos ya que éstos incorporan el manejo de dichos equipos a distancia, y de este modo incorporan al experimento factores reales importantes que los laboratorios virtuales suelen omitir, como ruido en sensores, rozamientos difíciles de modelar, etc. En el ámbito de la automática son numerosos los ejemplos de estos laboratorios donde se evidencia la importancia de los resultados alcanzados en su utilización (Muñoz de la Peña et al., 2022).

Los laboratorios remotos pueden incorporarse a la enseñanza y aprendizaje a distancia de prácticamente cualquier disciplina científico-técnica, pero la ingeniería de control y la robótica son algunas de las disciplinas en las que resulta más apropiado su uso, dadas sus características diferenciadoras. En particular, y para el caso de los robots manipuladores industriales de tipo serie, se han implementado laboratorios remotos que permiten al estudiantado programar robots reales para tareas de pick-and-place (Marín et al., 2005), programación mediante teach pendant virtual (Tzafestas et al., 2005), o planificación de trayectorias y visual servoing (Torres et al., 2006). Incorporando la ingeniería de control, también pueden encontrarse laboratorios remotos para experimentar a distancia con el control de robots de tipo serie, ya sea controladores de tipo Proporcional-Integral-Derivativo (PID) desacoplados para las distintas articulaciones (Wu et al., 2008; Temeltas et al., 2006; Castellanos et al., 2006), o controladores más generales, definidos con mayor libertad por el estudiantado (Temeltas et al., 2006; Castellanos et al., 2006).

Sin embargo, los laboratorios remotos de robots paralelos son más escasos que los de tipo serie. En este caso, encontramos realmente muy pocas referencias en la utilización de este tipo de laboratorios para la realización de prácticas remotas con robots paralelos. Los robots paralelos son manipuladores en los que la pinza (o efector final) está controlada por dos o más cadenas cinemáticas conectadas en paralelo. Estos robots poseen mayor rigidez y capacidad de carga que los de tipo serie, pero tienen un espacio de trabajo más limitado, y pueden encontrar singularidades en las que el robot no es controlable. Santana et al. (2013) presentan un laboratorio remoto para experimentar con el control desacoplado de cada uno de los pistones de un robot paralelo. En (Korayem et al., 2014) se describe otro laboratorio remoto para experimentar con el control de un robot paralelo de tipo cable, usando controladores PD, de pre-alimentación de su dinámica inversa, o combinaciones entre éstos. Por último, en (Li et al., 2018) se presenta un laboratorio virtual que, pese a no incorporar equipos reales controlados remotamente, permite estudiar la cinemática, dinámica y control de varios robots paralelos de tipo delta dispuestos en una línea de producción, con un elevado grado de realismo y similitud con una planta real, gracias a Unity 3D.

Los dos laboratorios remotos citados en el anterior párrafo son paradigmáticos de las dos estrategias de control más extensamente utilizadas a la hora de controlar robots paralelos. Dichas estrategias son el control lineal desacoplado, y el control por par computado (Paccot et al., 2009). La primera estrategia consiste en controlar cada articulación actuada del robot mediante un controlador PID lineal, de forma totalmente independiente del resto de articulaciones, como si el movimiento de una articulación no se viera afectado por el del resto. Tal estrategia de control es sencilla y funciona bien cuando no existe demasiado acoplamiento dinámico entre distintas articulaciones, lo cual ocurre, por ejemplo, en robots movidos por motorreductores con relaciones de reducción elevadas. Esto se debe a que, en ese caso, los pares de perturbación que una articulación sufre debido al movimiento del resto de articulaciones del robot, se ven divididos por grandes relaciones de reducción, y el efecto de tales perturbaciones se atenúa (Hsia et al., 1991). La segunda estrategia, el control por par computado (en inglés: Computed Torque Control), es un caso especial de la técnica más general de control no-lineal de linealización por realimentación, y consiste en computar una acción de control en base al modelo dinámico no-lineal del robot, de modo tal que, al aplicar dicha acción de control al robot, su modelo dinámico no-lineal se ve cancelado y, en bucle cerrado, el robot se comporta como un sistema lineal de integradores dobles, cuyo comportamiento puede regularse con precisión mediante técnicas lineales de control. Evidentemente, este segundo tipo de control permite lograr mayores prestaciones de control cuando el acoplamiento entre articulaciones es importante, pero es más difícil de implementar ya que requiere disponer de un modelo no-lineal que reproduzca con suficiente precisión la dinámica del robot real, de modo que dicha dinámica sea efectivamente cancelada por el modelo no-lineal, según se ha explicado anteriormente.

Existe una solución de control intermedia entre estos dos extremos, es decir, entre el control lineal desacoplado mono-articular y el control no-lineal acoplado multi-articular, y se trata del control lineal en el espacio de estados. La teoría de control moderno lineal en el espacio de estados permite abordar con facilidad el control preciso de sistemas multivariable que, como los robots paralelos, poseen varias entradas de control, varias salidas a controlar, y varios estados internos entre los que existe acoplamiento dinámico. Sin embargo, son muy pocos los trabajos que han explorado este tipo de control en los robots paralelos. Belda et al. (2003) linealizan y discretizan el modelo de estado de un robot paralelo con actuadores redundantes, y diseñan un controlador predictivo en base a dicho modelo. Zhang (2020) utiliza el software ADAMS para modelar un robot paralelo háptico de 6 grados de libertad (GDL) y generar automáticamente las matrices de su modelo de estado linealizado, y luego diseña un regulador cuadrático-lineal (LQR) discreto para dicho modelo. De forma similar, Alashqar (2007) utiliza Matlab/Simulink para generar y linealizar automáticamente el modelo de estado de un robot paralelo tipo Delta de 3 GDL, y posteriormente compara un controlador PID con otro predictivo, para dicho modelo. Por último, Bordalba et al. (2018) linealizan la dinámica de un robot paralelo a lo largo de la trayectoria deseada, obteniendo un modelo de estado lineal variante en el tiempo, y diseñan un LQR robusto ante singularidades.

Dado que los robots paralelos son sistemas multivariables con un elevado grado de acoplamiento entre distintas articulaciones (dada su estructura de cadenas cinemáticas cerradas), su control clásico desacoplado no siempre proporciona el mejor comportamiento dinámico. Por el contrario, sin llegar a la complejidad exigida por un controlador no-lineal de par computado, el control lineal por realimentación del estado permite considerar el acoplamiento existente entre varias entradas, salidas y estados internos, así

como la controlabilidad del sistema, que en un robot paralelo se ve degradada a medida que el robot se aproxima a singularidades. En este artículo se proponen e ilustran prácticas para experimentar con la identificación y control lineal de robots paralelos reales en el espacio de estados, describiendo la arquitectura del laboratorio remoto que permite llevar a cabo dichas prácticas a través de Internet.

Este artículo está organizado como sigue. La Sección 2 describe los dos robots paralelos implementados en el laboratorio remoto presentado. La Sección 3 explica en detalle la arquitectura cliente-servidor de dicho laboratorio. A continuación, la Sección 4 presenta diversas prácticas y experimentos de identificación y control que pueden realizarse a distancia con los robots implementados. Finalmente, la Sección 5 resume las conclusiones y propone trabajos futuros.

2. Robots implementados

En esta sección se presentan los dos robots paralelos implementados en el laboratorio remoto desarrollado.

2.1. El robot 5R

El robot 5R se muestra en la Figura 1a. Se trata de una cadena cinemática cerrada formada por cinco barras (una de ellas fija) unidas mediante cinco articulaciones de rotación (de ahí el nombre "5R", donde "R" denota una articulación de rotación tipo bisagra). También puede interpretarse como dos brazos planares de tipo RRR unidos en su extremo final, en cuyo caso podríamos denotar este robot como "2RRR". Las dos articulaciones conectadas con la barra fija están actuadas por motores de corriente continua. En este artículo, las longitudes de las barras son $l_1 = l_2 = l_3 = l_4 = 0.2$ m, y $l_5 = 0.48$ m. Los ángulos (θ_1 y θ_2) que forman las barras l_1 y l_4 con la horizontal son las coordenadas articulares actuadas. Dichos ángulos están directamente controlados mediante los mencionados motores.

Este robot se utiliza cuando se necesita controlar la posición de una pinza en el plano, sin importar su orientación. Se emplea, por ejemplo, en líneas de producción para tareas de envasado o empaquetado de piezas o productos con silueta aproximadamente circular, cuya orientación es irrelevante para ser introducidos en su envase (por ejemplo, discos o tortitas).

La cinemática directa de este robot consiste en determinar su configuración cuando se conocen los ángulos actuados θ_1 y θ_2. Si se conocen dichos ángulos, el robot puede adoptar dos posibles soluciones que sitúan la pinza en posiciones simétricas respecto a la línea que pasa por los codos de ambos brazos (véase la Figura 1b). Si únicamente se miden los ángulos θ_1 y θ_2, no se puede conocer cuál de las dos soluciones adopta el robot sin usar algún sensor redundante.

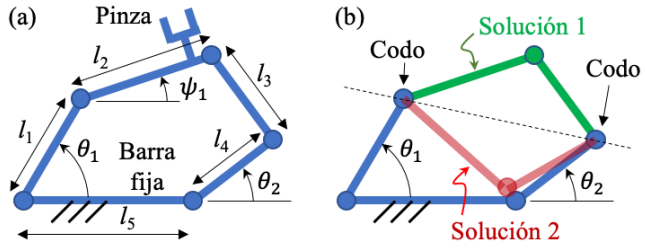

Figura 1: (a) Robot paralelo 5R. (b) Sus 2 soluciones.

2.2. El robot 3RRR

El robot 3RRR se muestra en la Figura 2. Este robot consta de tres brazos de tipo RRR. Los extremos ABC de cada brazo están conectados a una base fija, mientras que los otros están conectados a una plataforma móvil DEF que transporta una herramienta o pinza para realizar tareas o transportar productos. En este artículo, todas las barras tienen una longitud de 0.2m, los puntos ABC que unen los brazos con la base fija forman un triángulo isósceles de base $\overline{AB} = 0.645$ m y altura de 0.57 m, y los puntos DEF que unen los brazos con la plataforma móvil forman un triángulo equilátero de lado 0.2 m. Las articulaciones (A, B y C) unidas a la base fija están actuadas por motores de corriente continua, y dichos motores se encargan de controlar los ángulos θ_1, θ_2 y θ_3.

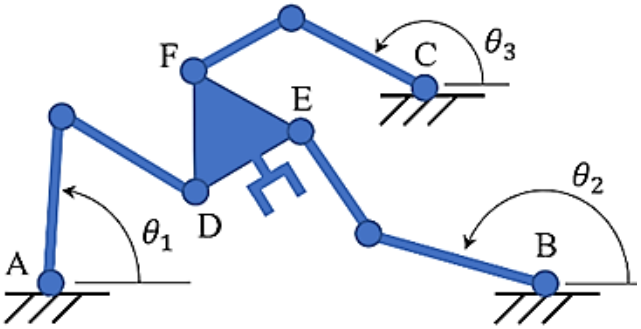

Figura 2: Robot paralelo 3RRR.

Al igual que el anterior robot, éste también puede emplearse en tareas de pick-and-place y envasado o empaquetado de productos en el plano, pero el robot 3RRR puede controlar tanto la posición de la pinza como su orientación, de modo que le permite ajustar la orientación para insertar el producto transportado en su alojamiento en el envase, pudiendo así envasar productos con cualquier geometría (no solo circular).

De forma análoga al anterior robot, la cinemática directa del robot 3RRR consiste en determinar la configuración del robot cuando se conocen los ángulos actuados θ_1, θ_2 y θ_3. No obstante, la cinemática directa del robot 3RRR es considerablemente más complicada que la del robot 5R. Esto se debe a que resolver la cinemática directa del robot 5R se reduce a resolver una ecuación cuadrática (de la que surgen las dos soluciones simétricas ilustradas en la Figura 1b), mientras que la cinemática directa del robot 3RRR requiere resolver un polinomio de grado 6 que, dependiendo del valor que tengan los ángulos θ_1, θ_2 y θ_3, puede admitir 2, 4 o 6 soluciones reales distintas. Por tanto, midiendo únicamente dichos ángulos no es posible saber cuál de esas soluciones es la adoptada por el robot, siendo necesario utilizar algún sensor redundante, como se explicará posteriormente en la Sección 3.3.

3. Arquitectura del laboratorio remoto

La Figura 3 muestra la arquitectura de este laboratorio, que va desde la aplicación cliente usada por el estudiantado para visualizar el robot y comandar sus movimientos, hasta el propio robot real situado en las instalaciones de la universidad. Entre ambos extremos (estudiante y robot remoto) hay varios elementos que se describirán en las siguientes subsecciones.

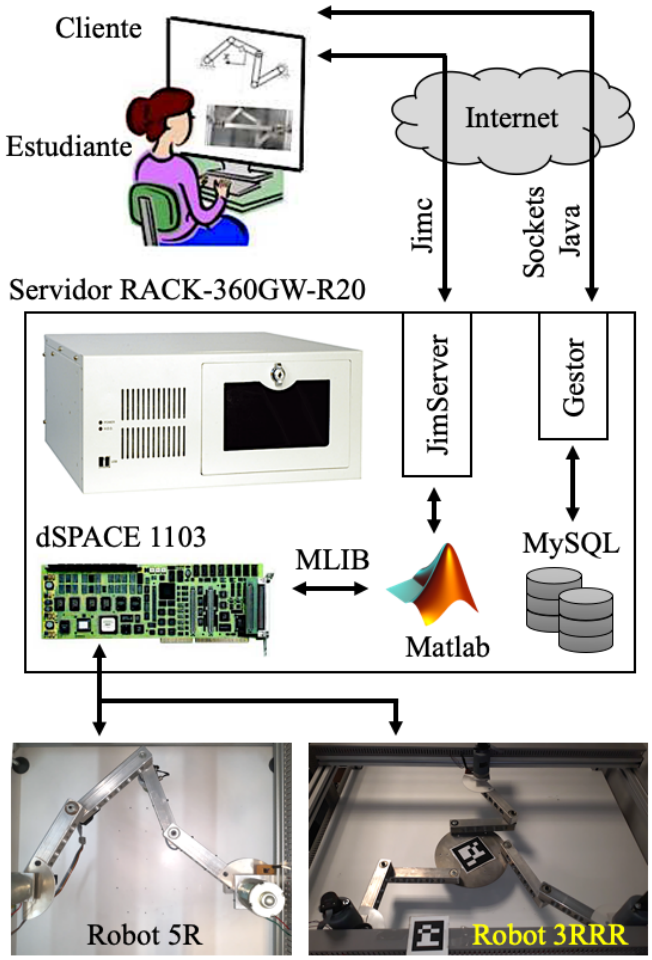

Figura 3: Arquitectura del laboratorio remoto.

3.1. Aplicación del cliente

3.1.1. Autenticación y reserva de franja horaria

La aplicación del cliente consiste en una interfaz en Java, desarrollada mediante Easy Java Simulations o EJS (Esquembre, 2015), que consta de los siguientes elementos. La primera pantalla mostrada al estudiante es una ventana de autenticación, donde cada estudiante debe ingresar su usuario y contraseña antes de poder utilizar el robot (Figura 4). Esta autenticación es necesaria por motivos de seguridad y exclusividad de uso, ya que el usuario manejará equipos reales que no deben ser controlados de forma simultánea por más de un usuario, o de otro modo el robot podría recibir órdenes contradictorias a la vez. Cuando el usuario ingresa sus credenciales, éstas son enviadas al servidor, que las contrasta con una base de datos en MySQL.

Cabe destacar que, como muestra la Figura 4 a la izquierda de la pantalla de inicio de sesión, se dispone de una imagen en tiempo real de cada robot ubicado en las instalaciones de la universidad, captada por una webcam instalada sobre cada robot. Esta imagen de webcam puede visualizarse en el navegador web, incluso aunque no se disponga de credenciales de acceso o, incluso teniéndolas, no se tenga una reserva de uso exclusivo del robot en la franja horaria actual. De este modo, hasta un máximo de 10 estudiantes conectados de forma simultánea pueden ver el robot en movimiento si otro estudiante está utilizándolo, aunque solo podrán ver la imagen de la webcam del robot, y no las gráficas de resultados de los experimentos, que solo podrá ver el estudiante que tenga una reserva activa (estas gráficas se explicarán posteriormente).

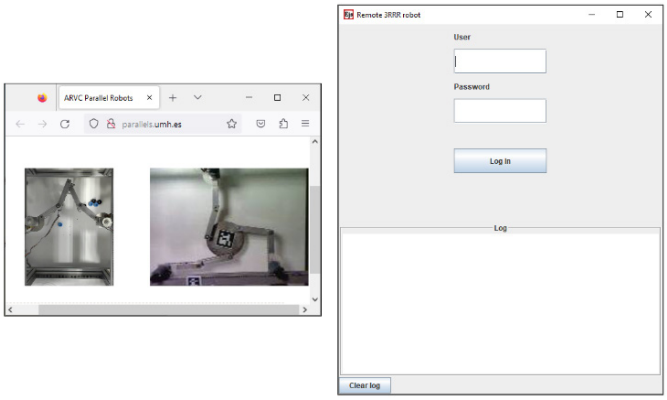

Figura 4: Pantalla de autenticación del estudiante.

Tras producirse el ingreso correcto al servidor, aparece una pantalla como la mostrada en la Figura 5, que permite al estudiante realizar la reserva de una franja horaria para la utilización exclusiva del robot (botón “Book”), o proceder a utilizar el robot si se dispone ya de una reserva para la franja actual de tiempo (botón “Use the robot”). Al clicar el botón de reserva, se muestra una tabla horaria que permite seleccionar un mes, día y hora para el uso exclusivo del robot, pudiéndose reservar una hora por día y usuario. Estas reservas se guardan y consultan en la misma base de datos MySQL utilizada para validar las credenciales de los usuarios.

Tras reservar una franja horaria, y llegado el momento de la reserva, el usuario puede clicar el botón “Use the robot” para inicializar el robot, llevándolo a una postura de referencia para calibrar sus encoders incrementales, y podrá comenzar a realizar experimentos remotos mediante la interfaz de control mostrada en la Figura 6 y descrita en la siguiente subsección.

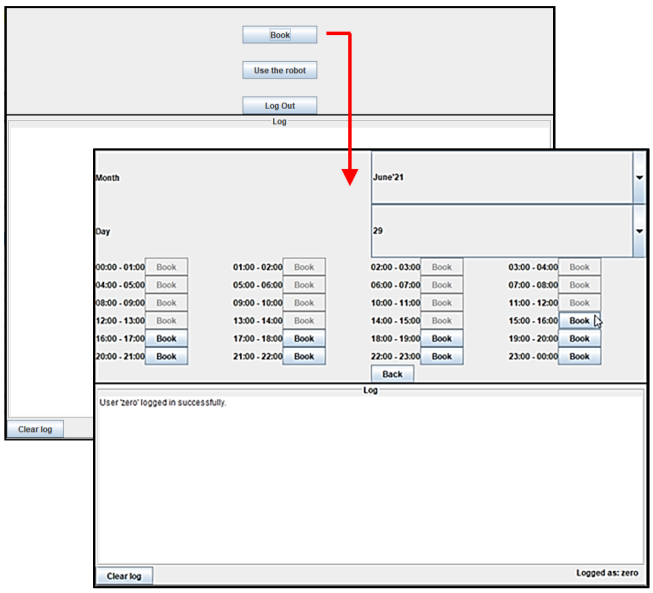

Figura 5: Reserva de una franja horaria.

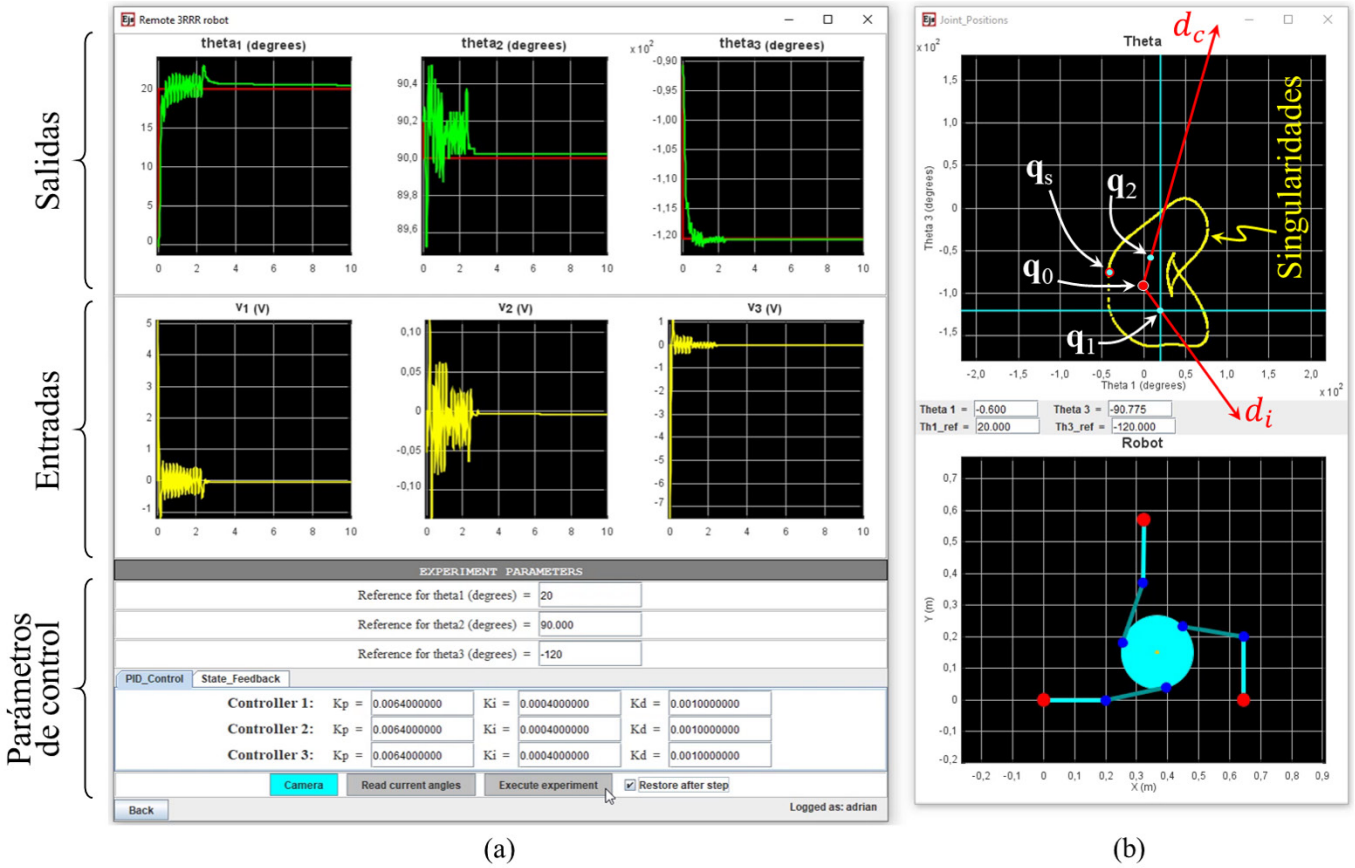

Figura 6: Interfaz del cliente para experimentos de identificación y control.

3.1.2. Interfaz de control del usuario

Una vez el usuario ha ganado control sobre el robot remoto, puede realizar experimentos mediante una interfaz gráfica de control como la mostrada en la Figura 6a. Esta interfaz consta de tres secciones. En la sección superior se muestran las gráficas de las salidas del sistema. Para ambos robots, considerados éstos como sistemas dinámicos multivariable o MIMO (Multiple Input - Multiple Output), las salidas son los ángulos girados por las articulaciones actuadas (θ_i en las Figuras 1 y 2), ya que éstas son las variables medidas mediante encoders. La sección intermedia de la interfaz muestra las correspondientes acciones de control, que son los voltajes v_i aplicados a los motores de corriente continua encargados de mover las respectivas articulaciones actuadas. Haciendo clic derecho sobre estas gráficas, es posible desplegar una nueva ventana (la herramienta de datos de EJS), donde estas gráficas aparecen en formato de una tabla que puede exportarse a Excel, Matlab o cualquier otro programa para procesar los datos con el objetivo de realizar una identificación paramétrica u otros fines, según se explicará en la Sección 4 de este artículo.

La sección inferior dispone de varios campos numéricos en los que el estudiante puede introducir los parámetros de control para realizar el experimento. En primer lugar, puede introducir el setpoint o valor deseado para cada articulación actuada θ_i, donde cada setpoint se trata como una referencia en escalón a seguir por el lazo de control. A continuación, el usuario debe especificar las ganancias de control, pudiendo elegir dos opciones. La primera es un control desacoplado PID clásico de cada articulación actuada. La segunda opción es un control multivariable por realimentación del estado (con ganancia matricial $\mathbf{K}_C$) con regulador integral $\mathbf{K}_I$ para ajuste robusto del régimen permanente, que ya no asume que las articulaciones están desacopladas (como el control PID), sino que considera el acoplamiento dinámico entre ellas. La Figura 7 muestra el esquema de control multivariable por estado realimentado que se implementa en el laboratorio remoto, para cada robot.

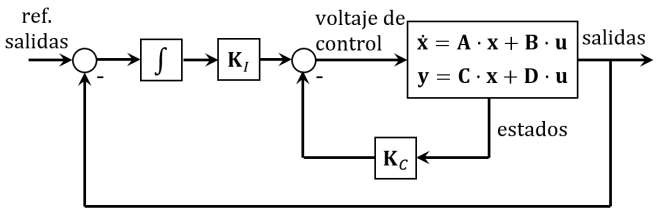

Figura 7: Control integral con estado realimentado.

En ambos esquemas de control (PID desacoplados o realimentación del estado MIMO), todas las variables son incrementales respecto a la configuración adoptada por el robot justo antes de realizar el experimento de control. Trabajar con variables incrementales permite que el estudiantado utilice técnicas de control lineal, una vez tenga identificado el modelo dinámico linealizado del robot. Evidentemente, usar técnicas lineales logrará que el robot se comporte según se desee solo si los setpoints para cada ángulo θ_i están razonablemente cerca de los valores que dichos ángulos tienen al principio de cada experimento de control. Si el estudiante introduce setpoints

demasiado alejados, estará solicitando al controlador que aleje al robot demasiado de su punto de equilibrio inicial (en torno al que se diseña el regulador lineal), por lo que el regulador diseñado podría no funcionar como se espera.

Además, como muestra la Figura 6b, la interfaz consta también de una ventana adicional con otras dos gráficas: una gráfica superior, y otra inferior.

La gráfica inferior es una representación simulada del robot, que se realiza midiendo primero el valor actual de los ángulos θ_i en el robot real (a través del botón "Read current angles" de la ventana mostrada en la Figura 6a), y resolviendo luego la cinemática directa del robot, para esos ángulos (como se explicará en la Sección 3.3, además es necesario usar sensores redundantes para determinar sin ambigüedad cuál de todas las soluciones posibles de la cinemática directa es la adoptada por el robot real). Esta representación simulada del robot puede compararse con la que adopta el robot real, mostrado en la webcam (parte izquierda de la Figura 4).

En cuanto a la gráfica superior de la Figura 6b, ésta representa el espacio de articulaciones actuadas, con las singularidades del robot representadas en color amarillo (en tales singularidades, el robot no es controlable). Para el robot 5R, dicho espacio es el plano θ_1-θ_2, y las singularidades se representan como curvas en dicho plano. Análogamente, para el robot 3RRR se debería representar el espacio tridimensional θ_1-θ_2-θ_3, donde las singularidades se representarían como superficies, pero la geometría de tales superficies es bastante intrincada y resultaría difícil visualizarlas en tres dimensiones. En su lugar, para facilitar la comprensión del mapa de singularidades, para el robot 3RRR se ha preferido representar el plano θ_1-θ_3, donde en color amarillo se representan las curvas de singularidades que resultan de intersecar las mencionadas superficies de singularidades con el plano $\theta_2 = K$, donde K es el valor actual que adopta el ángulo θ_2 del robot real (para el ejemplo mostrado en la Figura 6, $K = 90º$). Esta representación gráfica planar resulta más simple y didáctica para el estudiantado que utiliza la interfaz del robot 3RRR.

Por último, el punto rojo grueso $\mathbf{q}_0$ mostrado en la gráfica superior de la Figura 6b representa el valor actual de los ángulos actuados del robot (θ_i), mientras que el centro $\mathbf{q}_1$ de la cruz de color cian representa el setpoint, target o referencia a la que se desea mover el robot mediante el experimento de control. Tal setpoint puede seleccionarse clicando directamente el punto deseado en la gráfica, o tecleando su valor en los campos numéricos disponibles en la interfaz.

3.2. Servidor

El servidor es un ordenador industrial IEI Technology Corp., modelo RACK-360GW-R20, que incorpora una tarjeta de control dSPACE 1103 que se conecta directamente a los robots remotos que se describirán en la Sección 3.3. Dicha tarjeta dispone de numerosas entradas y salidas analógicas y digitales, con las que leer encoders y otras señales digitales o analógicas, ejecutando el algoritmo de control y enviando a los drivers de los motores las tensiones de control necesarias. Además, el servidor ejecuta un conjunto de programas que permiten gestionar el acceso del estudiantado, sus reservas, y los flujos de información entre estudiante y robot. Estos programas y flujos de información se resumen en la Figura 3, y a continuación se describen con mayor detalle.

3.2.1. Gestor de credenciales y reservas + MySQL

Un programa en Java, de elaboración propia, se ejecuta continuamente en el servidor a modo de gestor de credenciales y reservas, escuchando peticiones del cliente para validar sus credenciales (introducidas por el estudiante en la pantalla de autenticación mostrada en la Figura 4), o para realizar la reserva de una franja horaria en la que hacer un uso exclusivo del robot. La comunicación entre este gestor y la interfaz del cliente se realiza mediante sockets de Java. El gestor de credenciales y reservas accede a dos bases de datos MySQL almacenadas en el propio servidor: una base de datos contiene las credenciales de los usuarios para validarse en el sistema, y la otra contiene todas las reservas realizadas por los usuarios.

3.2.2. JimServer y Jimc

JimServer es un programa escrito en Java (Farias et al., 2010) que se ejecuta en el servidor. JimServer permite comunicar una sesión de Matlab corriendo en el servidor con la interfaz de usuario de la Figura 6 que se ejecuta en el ordenador del estudiante, de modo que dicha interfaz, que emplea la librería Jimc (Java Internet Matlab Client; Farias, 2010), puede enviar comandos de Matlab para que se ejecuten en la sesión de Matlab del servidor, pudiendo intercambiar, además de dichos comandos, variables y datos de forma bidireccional. Los comandos de Matlab que se envían desde la interfaz de usuario del cliente para ser ejecutados en la sesión de Matlab del servidor, se usan para modificar parámetros del esquema de control de los robots remotos, como por ejemplo las ganancias de control o los setpoints, y también para registrar los resultados de cualquier experimento de control (gráficas de las salidas del sistema y de las acciones de control) y devolverlas al usuario, para que éste pueda visualizarlos en forma de gráficas, como se ha ilustrado en la Figura 6a.

3.2.3. Matlab y Simulink

Finalmente, el último programa necesario para completar la comunicación entre los robots remotos y el estudiante es Matlab/Simulink. El esquema de control encargado de controlar ambos robots se encuentra programado en forma de un diagrama de Simulink que alberga, entre muchos otros bloques, las ganancias de control y los setpoints a seguir por el robot. Este esquema de control se carga y ejecuta directamente sobre la tarjeta dSPACE 1103 antes mencionada. Cuando el usuario desea introducir un nuevo setpoint o referencia para las articulaciones del robot, o cuando se desean modificar las ganancias de control de acuerdo con lo descrito en la Sección 3.1.2, esos cambios se producen mediante la ejecución de comandos de la librería MLIB de dSPACE, que comunica Matlab con la tarjeta dSPACE. Dicha librería contiene comandos que permiten modificar cualquier parámetro del esquema de control Simulink, o extraer datos de la tarjeta. Estos comandos se envían desde la interfaz de control del cliente a través de la pasarela Jimc-JimServer, y se ejecutan en la sesión de Matlab que corre en el servidor.

Como ejemplo para ilustrar todo lo anterior, considérese la siguiente línea extraída del código fuente del cliente:

externalApp.eval(*"mlib('Write',mlib('GetTrcVar', 'Model Root/PID 1 POSICION-3RRR/KP/Gain'),'Data',kp1);"*);

Esta línea se ejecuta en la interfaz del cliente, y emplea una instancia "externalApp" de la clase "RMatlabExternalApp" perteneciente a la librería Jimc, que permite comunicar un cliente Java con una sesión remota de Matlab. El método "eval" ejecuta en Matlab la sentencia destacada arriba en *cursiva* y entre comillas dobles: *"mlib(...);"*. Esta sentencia emplea la librería MLIB para escribir (*'Write'*), en el esquema de control Simulink que corre en la dSPACE 1103, el valor de la ganancia Proporcional (*KP*) del PID que controla al motor 1 del robot 3RRR, asignándole el valor *kp1*. Para ello, antes será necesario ejecutar el método "setValue" para enviar a la sesión remota de Matlab el valor de dicha variable *kp1* desde la interfaz cliente, lo que puede hacerse como sigue:

externalApp.setValue(*"kp1",kp1_java*);

donde *kp1_java* es la variable del cliente Java que almacena el valor introducido por el estudiante en el campo numérico Kp del Controller 1 que puede observarse en la Figura 6a, mientras que *kp1* es el análogo de esta variable en Matlab.

3.3. Robots Remotos

Como se ha comentado en la Sección 2, los robots paralelos implementados en el laboratorio remoto son los robots 5R y 3RRR, que se muestran en el inferior de la Figura 3. Las dimensiones de los robots son las que se han indicado en la Sección 2, y cada barra pesa 0.5 kg. Los robots están movidos por motores de corriente continua de 24 V, que van alimentados por servocontroladoras ESCON 50/5 del fabricante Maxon. Los pines digitales y analógicos de tales servocontroladoras van directamente conectados a la tarjeta dSPACE 1103 instalada en el servidor, y reciben de dicha tarjeta la tensión analógica de control que debe ser aplicada a los bornes de cada motor, según determine el esquema de control programado en dicha tarjeta a través de Simulink.

Como también se ha comentado en la Sección 2, el robot 5R dispone de dos articulaciones actuadas, que son las conectadas directamente a la base fija del robot, mientras que el robot 3RRR dispone de tres articulaciones actuadas, que también son las situadas sobre la base fija. En robots paralelos es habitual actuar las articulaciones conectadas directamente a la base fija del robot, para que los motores encargados de mover dichas articulaciones descansen sobre dicha base, y así minimizar las inercias, que serían mucho mayores si los motores se colocasen sobre otras articulaciones más alejadas de la base. En cualquiera de los dos robots, cada articulación actuada consta de un motor de corriente continua de 24 V y un encoder óptico modelo HEDR-55L2-BY09, que tiene una resolución de 3600 pulsos por vuelta para medir cada ángulo θ_i.

Se ha prescindido de reductoras entre los motores y las articulaciones actuadas para evitar la holgura que suelen introducir los engranajes de las reductoras, que pueden llegar a empeorar notablemente la precisión del robot y generar un comportamiento demasiado vibratorio. El prescindir de reductoras, además de suprimir holguras, también permite al robot alcanzar mayores velocidades, a costa de reducir el par que los motores pueden proporcionar a las articulaciones, aunque esto no supone un problema ya que ambos robots se mueven en planos horizontales, de modo que los motores no han de vencer el peso propio de las barras del robot. Nótese que, al prescindir de reductoras, el acoplamiento dinámico entre distintas articulaciones gana importancia (Hsia et al., 1991), de modo que, si se desea diseñar un regulador que cumpla con las especificaciones de control, será necesario utilizar técnicas de control MIMO en lugar de un controlador PID desacoplado independiente para cada articulación.

En cada uno de estos dos robots, como ya se ha comentado, cada articulación actuada está sensorizada por un encoder óptico que mide la posición angular θ_i de dicha articulación, con el fin de realimentar dicha posición. Los encoders ópticos empleados son incrementales, por lo que es necesario establecer un protocolo de calibración durante el arranque del robot, para encontrar la posición de referencia cero de cada articulación y así resetear los contadores encargados de contar pulsos de los encoders incrementales, para conocer así el ángulo absoluto de cada articulación actuada (ángulo absoluto θ_i respecto a la horizontal, según se ha representado en las Figuras 1 y 2). Dicho protocolo de calibración inicial se basa en incorporar un sector circular solidario a cada articulación actuada, en combinación con sensores ópticos modelo CNY70. En el arranque del robot, si el sensor óptico está tapado por dicho sector circular, se controla la articulación en velocidad, haciéndola girar a baja velocidad en un sentido hasta que el sensor óptico quede destapado. O si se encuentra destapado, se hace girar en sentido contrario hasta que el sensor óptico queda tapado. Cuando se produce el cambio entre tapado y destapado, se resetea el contador del encoder y se fija así la posición cero de cada articulación, finalizando la calibración.

Por último, cada robot dispone de un sensor redundante para conocer qué solución de la cinemática directa adopta, y así poder representarlo correctamente en la Figura 6b. Como se ha comentado en la Sección 2, cada robot posee varias soluciones a su cinemática directa, de modo que, conociendo únicamente los ángulos actuados θ_i, no puede saberse cuál de esas soluciones es la adoptada por el robot real. En el robot 5R, conociendo los ángulos θ_i, el robot puede adoptar dos soluciones simétricas (Figura 1b). Para determinar su solución, se ha colocado un tercer encoder en el codo izquierdo, que permite medir el ángulo ψ_1 (Figura 1a) y así conocer sin ambigüedad la solución adoptada. En el caso del robot 3RRR esto es más complicado, ya que éste puede adoptar hasta 6 soluciones distintas para un valor dado de los ángulos θ_i. En este caso, como se observa en la Figura 3, se han colocado marcas ArUco en la base fija del robot y en su efector final, para estimar la pose relativa entre ambos, y se compara dicha pose con las seis posibles soluciones teóricas que se obtendrían al resolver la cinemática directa a partir de los ángulos θ_i medidos por los encoders, eligiendo la solución más parecida.

4. Prácticas propuestas

El laboratorio remoto presentado está pensado para realizar prácticas de control con robots paralelos, especialmente prácticas en el contexto de asignaturas de robótica o control multivariable en el espacio de estados. A continuación, se describen tres experimentos o actividades prácticas que pueden realizarse con este laboratorio en dichas asignaturas.

4.1. Práctica 1: identificación

4.1.1. Parte 1: modelado y registro de datos

Como se ha explicado en la Sección 3.1.2, la interfaz de usuario desarrollada permite al estudiantado registrar las entradas y salidas de cada robot paralelo remoto, considerando éste como un sistema dinámico multivariable MIMO, donde las salidas son los ángulos θ_i de las articulaciones actuadas, y las entradas son los voltajes v_i aplicados a los motores de corriente continua encargados de mover cada articulación actuada respectiva θ_i. Si se realiza un experimento moviendo el robot, y se registran sus señales de entradas y salidas a lo largo del experimento, se pueden utilizar dichas señales para realizar la identificación del modelo dinámico del robot en el espacio de estados, ya sea un modelo no-lineal más preciso, o bien un modelo linealizado en torno a la configuración inicial del experimento. En este artículo, nos centraremos en la identificación de modelos de estado linealizados.

Para un robot paralelo de dos GDL, como el robot 5R, el modelo de estado linealizado en torno a una configuración de equilibrio tiene la siguiente forma (Peidró et al., 2022):

$$\frac{d}{dt}\begin{bmatrix}x_1\\x_2\\x_3\\x_4\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 0 & 1 & 0\\0 & 0 & 0 & 1\\a_{31} & a_{32} & a_{33} & a_{34}\\a_{41} & a_{42} & a_{43} & a_{44}\end{bmatrix}}_{\mathbf{A}}\underbrace{\begin{bmatrix}x_1\\x_2\\x_3\\x_4\end{bmatrix}}_{\mathbf{x}} + \underbrace{\begin{bmatrix}0 & 0\\0 & 0\\b_{31} & b_{32}\\b_{41} & b_{42}\end{bmatrix}}_{\mathbf{B}}\begin{bmatrix}u_1\\u_2\end{bmatrix} \quad (1)$$

$$\underbrace{\begin{bmatrix}y_1\\y_2\end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\end{bmatrix}}_{\mathbf{C}}\begin{bmatrix}x_1\\x_2\\x_3\\x_4\end{bmatrix} + \underbrace{\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}}_{\mathbf{D}}\underbrace{\begin{bmatrix}u_1\\u_2\end{bmatrix}}_{\mathbf{u}} \quad (2)$$

donde los coeficientes a_{ij} y b_{ij} dependen en general de las masas e inercias del sistema, de los coeficientes de rozamiento, de la gravedad, y de la configuración de equilibrio en torno a la que el modelo del robot está linealizado.

Para el robot 5R, las entradas al sistema (u_1 y u_2) son los voltajes aplicados a sus motores de corriente continua. Las salidas (y_1 y y_2) son los ángulos actuados (θ_1 y θ_2), puesto que éstas son las variables medidas mediante sendos encoders. Resolviendo la cinemática directa del robot para valores conocidos de θ_1 y θ_2, se podría estimar la posición (x, y) de su efector final si ésta fuera necesaria. Por último, en cuanto a los estados, x_1 y x_2 coinciden también con los ángulos θ_1 y θ_2, mientras que los estados x_3 y x_4 son las derivadas de los primeros dos estados, es decir, coinciden con las velocidades angulares de las barras l_1 y l_4 de este robot ($\dot{\theta}_1$ y $\dot{\theta}_2$).

Para el robot 3RRR, dado que éste tiene tres grados de libertad, el modelo de estado sería similar al dado por (1) y (2), pero con tres entradas (un voltaje por cada motor), tres salidas (los ángulos θ_1, θ_2 y θ_3), y seis estados (tres coordenadas generalizadas de posición, y tres coordenadas generalizadas de velocidad). No obstante, por simplicidad y por los motivos didácticos indicados al final de la Sección 3.1.2, podemos asumir que una de las articulaciones del robot 3RRR se mantiene fija. Por ejemplo, podemos suponer que existe un controlador que se encarga de mantener fijo (o casi fijo) el ángulo θ_2 de este robot en cualquier valor deseado. En ese caso, supuesto θ_2 fijo, el robot 3RRR pasaría a tener dos grados de libertad, y su modelo de estado linealizado también vendría dado por (1) y (2), donde las salidas, entradas y estados quedarían asignadas de la siguiente manera:

- salidas = $[y_1, y_2] = [\theta_1, \theta_3]$;
- entradas = $[u_1, u_2]$ = voltajes $[v_1, v_3]$ aplicados a los motores encargados de controlar los ángulos θ_1 y θ_3;
- estados = $[x_1, x_2, x_3, x_4] = [\theta_1, \theta_3, \dot{\theta}_1, \dot{\theta}_3]$.

En los experimentos que se mostrarán en lo que sigue, se demostrará la viabilidad de identificar y controlar el robot 3RRR como si solo tuviera dos grados de libertad, con su articulación θ_2 fijada mediante un controlador externo.

Sea como fuere, si asumimos que tenemos un robot paralelo cuyo comportamiento dinámico en torno a una configuración de equilibrio se aproxima al modelo de estado linealizado dado por (1) y (2), entonces podemos identificar dicho modelo (es decir, estimar los coeficientes a_{ij} y b_{ij}) mediante los experimentos y cálculos que se describen en esta sección. A modo de ejemplo para ilustrar lo que sigue, se utilizará el robot 3RRR, identificando el modelo de estado linealizado (1), asumiendo que θ_2 varía muy poco en torno a 90º.

Para realizar el experimento de identificación, en primer lugar, se introduce en la interfaz de usuario un regulador PID independiente para cada articulación. El principal requisito de dicho regulador PID es que logre estabilizar al robot con suficiente precisión en la configuración de equilibrio en torno a la que se desea identificar un modelo de estado linealizado. El comportamiento dinámico, es decir, la respuesta transitoria lograda por dicho regulador PID, no es tan relevante como el hecho de que el error en régimen permanente sea pequeño, por lo que se admite cualquier comportamiento transitorio siempre que la sobreoscilación y el tiempo de establecimiento no alcancen valores excesivos. Pueden lograrse estos objetivos, por ejemplo, si se asignan las siguientes ganancias al regulador PID encargado de regular cada una de las tres articulaciones del robot 3RRR (mismas ganancias para los tres reguladores, como muestra la Figura 6a): $K_p = 6.4 \cdot 10^{-3}$, $K_i = 4 \cdot 10^{-4}$, $K_d = 10^{-3}$. Estos valores se ajustaron de forma experimental.

A continuación, utilizando este control PID desacoplado, se controla el robot para llevarlo hasta una configuración inicial $\mathbf{q}_0$ de equilibrio, en torno a la que se identificará un modelo de estado linealizado. Esto puede hacerse clicando en el punto $\mathbf{q}_0$ deseado en el plano de articulaciones actuadas del robot 3RRR (plano θ_1-θ_3, mostrado en la Figura 6b), o tecleando las coordenadas precisas de la configuración deseada. Tras llevar el robot a dicha configuración inicial de reposo y estabilizarlo en ella, puede realizarse el experimento de identificación.

Para realizar la identificación, se mantiene el mismo PID indicado dos párrafos atrás, que ha sido utilizado para llevar el robot hasta $\mathbf{q}_0$. A continuación, se marca la casilla “Restore after step” en la interfaz de control del cliente. Esto hará que, tras el experimento de identificación, el robot sea devuelto a la configuración inicial $\mathbf{q}_0$. El objetivo de volver a $\mathbf{q}_0$ tras el experimento de identificación es que, cuando se hayan identificado las matrices **A** y **B** de (1) como resultado de tal experimento, y tras diseñar un regulador por estado realimentado en base a dichas matrices (véase el experimento descrito en la Sección 4.2), el usuario pueda probar dicho regulador partiendo precisamente de $\mathbf{q}_0$, que es la configuración en torno a la que dichas matrices identificadas (y, por extensión, el regulador diseñado) son válidas.

Tras esto, se selecciona un target o setpoint $\mathbf{q}_1$ al que mover las articulaciones actuadas del robot, en el plano θ_1-θ_3. Dicho target debería ser un punto cercano a la configuración inicial $\mathbf{q}_0$, o, de otro modo, se estaría solicitando al robot que se aleje demasiado del punto de equilibrio $\mathbf{q}_0$ en torno al que se desea identificar un modelo de estado linealizado, y el modelo que resulte de la identificación podría no ser bueno si éste se identifica recabando datos del robot mientras éste se mueve en un rango de movimiento demasiado amplio y alejado del punto de equilibrio. Por ejemplo, en la Figura 6b se ha seleccionado el punto $\mathbf{q}_1$, que está cerca de $\mathbf{q}_0$. Nótese que, para este ejemplo, $\mathbf{q}_0$ y $\mathbf{q}_1$ están lejos de singularidades, que son las curvas de color amarillo en la Figura 6b. En la Sección 4.3 se abordará el caso en el que $\mathbf{q}_0$ está cerca de una singularidad.

Tras seleccionar el target $\mathbf{q}_1$, pulsar el botón "Execute experiment" de la interfaz ejecuta el control PID desacoplado para llevar el robot desde $\mathbf{q}_0$ hasta $\mathbf{q}_1$, durante un tiempo de 10 segundos. Es decir, se da un tiempo de 10 segundos para que el PID desacoplado introducido tenga tiempo de mover el robot desde $\mathbf{q}_0$ hasta $\mathbf{q}_1$, estabilizándolo en $\mathbf{q}_1$ con un error de posición pequeño o nulo. Durante estos 10 segundos, la tarjeta de control dSPACE registra los datos de entrada y salida del robot, es decir, los ángulos θ_1 y θ_3 leídos mediante encoders y los voltajes v_1 y v_3 aplicados a los respectivos motores, y los devuelve como gráficas que se muestran en la interfaz de usuario (Figura 6a). Tras transcurrir los 10 segundos del experimento, un controlador PID predefinido se encarga de devolver el robot a su configuración inicial $\mathbf{q}_0$.

Para ilustrar lo comentado en los párrafos anteriores, en la Figura 6a se muestra la interfaz de control del robot 3RRR, donde se ha producido un movimiento desde $\mathbf{q}_0 = (0, -90)^{\text{o}}$ hasta $\mathbf{q}_1 = (20, -120)^{\text{o}}$ en el plano θ_1-θ_3, manteniendo θ_2 aproximadamente constante en un valor de 90º, y se han devuelto las gráficas de los tres ángulos (θ_1, θ_2, θ_3) y de los voltajes de control aplicados a los respectivos motores (v_1, v_2, v_3) tras realizar dicho movimiento. Dado que, como muestran las gráficas, en este experimento el ángulo θ_2 se mantiene aproximadamente constante, a continuación describiremos cómo identificar un modelo de estado de una versión de 2 GDL del robot 3RRR, que obedezca a (1) y (2), donde el vector de salidas es $\mathbf{y} = [\theta_1, \theta_3]^T$, el de entradas es $\mathbf{u} = [v_1, v_3]^T$, y el de estados es: $\mathbf{x} = \left[\theta_1, \theta_3, \dot{\theta}_1, \dot{\theta}_3\right]^T$. Realizando un experimento de identificación similar se podría estimar un modelo de estado para el robot 5R, o incluso para el robot 3RRR con tres GDL.

Los cálculos de identificación sugeridos en esta sección para estimar los coeficientes a_{ij} y b_{ij} de las matrices **A** y **B** de (1) no harán uso de medidas de los estados. La razón es que dos de los estados (las velocidades $\dot{\theta}_1$ y $\dot{\theta}_3$) no son medidos en el robot real, solo se miden los ángulos θ_1 y θ_3 mediante encoders. No se puede diseñar un observador del estado para estimar $\dot{\theta}_1$ y $\dot{\theta}_3$ porque ello requiere el conocimiento de las matrices **A** y **B**, que es precisamente lo que se pretende estimar mediante el experimento descrito en esta sección. Otra opción sería estimar $\dot{\theta}_1$ y $\dot{\theta}_3$ diferenciando numéricamente las señales de θ_1 y θ_3, pero esto amplificaría el ruido de cuantización de dichas señales, y las estimaciones de $\dot{\theta}_1$ y $\dot{\theta}_3$ serían muy ruidosas, requiriéndose algún filtro que atenúe el ruido a expensas de distorsionar las señales en cierta medida. Por tanto, para evitar todos los problemas anteriores, la identificación se realizará sin necesidad de estimar los estados, utilizando solo las señales medibles de entrada **u** y salida **y**.

Dichas señales pueden exportarse desde la interfaz mostrada en la Figura 6a, haciendo clic derecho sobre las gráficas de las señales deseadas y abriendo la herramienta de análisis de datos de EJS, donde se muestra cada señal en formato de tabla. Estas tablas pueden exportarse a Matlab para operar con los datos. Como resultado, se dispone de las salidas $\mathbf{y}(t_k)$ y entradas $\mathbf{u}(t_k)$ muestreadas en instantes de tiempo $t_k = 0, 0.01, 0.02, \ldots, 10$ s (periodo de muestreo de 0.01 segundos). Tras convertir estas señales muestreadas en señales incrementales (ya que se va a identificar un modelo linealizado del robot), la identificación puede realizarse como sigue.

4.1.2. Parte 2: estimación paramétrica

En primer lugar, a partir de las matrices **A**, **B**, **C** y **D** del modelo de estado dado por (1) y (2), se puede obtener la matriz de funciones de transferencia de la siguiente forma:

$$\mathbf{G} = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} \tag{3}$$

De modo que la ecuación de entrada-salida es:

$$\mathbf{y} = \mathbf{G} \cdot \mathbf{u} = \frac{\mathbf{N}}{d} \cdot \mathbf{u} \tag{4}$$

donde el numerador **N** es una matriz 2x2 de polinomios de grado 2, y el denominador d es el polinomio característico del sistema en lazo abierto, con las siguientes expresiones:

$$\mathbf{N} = \begin{bmatrix} p_1 & p_2 \\ p_3 & p_4 \end{bmatrix} s^2 + \begin{bmatrix} p_5 & p_6 \\ p_7 & p_8 \end{bmatrix} s + \begin{bmatrix} p_9 & p_{10} \\ p_{11} & p_{12} \end{bmatrix} \tag{5}$$

$$d = s^4 + p_{16}s^3 + p_{15}s^2 + p_{14}s + p_{13} \tag{6}$$

y donde los parámetros auxiliares p_i dependen de los coeficientes a_{ij} y b_{ij}, de la siguiente manera:

$$\left.\begin{aligned}
&p_1 = b_{31}, \quad p_2 = b_{32}, \quad p_3 = b_{41}, \quad p_4 = b_{42} \\
&p_5 = a_{34}b_{41} - a_{44}b_{31}, \quad p_6 = a_{34}b_{42} - a_{44}b_{32} \\
&p_7 = a_{43}b_{31} - a_{33}b_{41}, \quad p_8 = a_{43}b_{32} - a_{33}b_{42} \\
&p_9 = a_{32}b_{41} - a_{42}b_{31}, \quad p_{10} = a_{32}b_{42} - a_{42}b_{32} \\
&p_{11} = a_{41}b_{31} - a_{31}b_{41}, \quad p_{12} = a_{41}b_{32} - a_{31}b_{42} \\
&p_{13} = a_{31}a_{42} - a_{32}a_{41} \\
&p_{14} = a_{31}a_{44} - a_{32}a_{43} + a_{33}a_{42} - a_{34}a_{41} \\
&p_{15} = -a_{31} + a_{33}a_{44} - a_{34}a_{43} - a_{42} \\
&p_{16} = -a_{33} - a_{44}
\end{aligned}\right\} \tag{7}$$

El problema de identificación se reduce, por tanto, a estimar el vector de 16 parámetros $\mathbf{p} = [p_1, \ldots, p_{16}]^T$, ya que si se conoce **p**, se podrá calcular a_{ij} y b_{ij} resolviendo (7).

A continuación, multiplicamos (4) por d, dividimos entre s^4 para que ningún término de la ecuación contenga potencias positivas de s (i.e., derivadas), y despejamos $\mathbf{y}(t)$ en función de integrales de la entrada $\mathbf{u}$ y de la salida $\mathbf{y}$. El resultado, tras organizarlo de forma matricial, queda en la siguiente forma:

$$\mathbf{y}(t) = \mathbf{R}(t) \cdot \mathbf{p} \tag{8}$$

donde $\mathbf{R}(\tau)$ es la matriz de regresión en el instante τ, formada por las integrales de las señales de entradas y salidas entre $t = 0$ y $t = \tau$ (recuérdese que dividir una señal entre s^k equivale a integrarla k veces), según la siguiente expresión:

$$\mathbf{R}(t) = \left[\frac{\mathbf{U}}{s^2} \quad \frac{\mathbf{U}}{s^3} \quad \frac{\mathbf{U}}{s^4} \quad -\mathbf{y}\left[\frac{1}{s^4} \quad \frac{1}{s^3} \quad \frac{1}{s^2} \quad \frac{1}{s}\right]\right] \tag{9}$$

donde $\mathbf{U} = \begin{bmatrix} u_1 & u_2 & 0 & 0 \\ 0 & 0 & u_1 & u_2 \end{bmatrix}$.

La matriz de regresión $\mathbf{R}(t)$ en el instante t puede obtenerse integrando numéricamente las señales de entrada y salida hasta dicho instante, a partir de las señales muestreadas $\mathbf{u}(t_k)$ y $\mathbf{y}(t_k)$ exportadas desde la herramienta de análisis de datos de EJS. La integración numérica es una operación preferible a la diferenciación numérica, que como ya se ha indicado antes, se ha decidido evitar en la identificación para no amplificar el ruido de las señales. En nuestro caso, $\mathbf{R}(t)$ se calcula por integración numérica en Simulink, con un paso temporal fijo de 0.01 segundos y el integrador ode3 (Bogacki-Shampine).

Si particularizamos (8) para cada instante de tiempo t_k en el que se han registrado las entradas y salidas, tendremos:

$$\underbrace{\begin{bmatrix} \mathbf{y}(t_1) \\ \mathbf{y}(t_2) \\ \vdots \\ \mathbf{y}(t_N) \end{bmatrix}}_{\bar{\mathbf{y}}} = \underbrace{\begin{bmatrix} \mathbf{R}(t_1) \\ \mathbf{R}(t_2) \\ \vdots \\ \mathbf{R}(t_N) \end{bmatrix}}_{\bar{\mathbf{R}}} \cdot \mathbf{p} \tag{10}$$

La ecuación (10) es un sistema de $2N$ ecuaciones ($N = 101$ porque se registran 10 segundos con un muestreo de 0.01 segundos) y 16 incógnitas (dimensión de $\mathbf{p}$). Como es habitual en identificación, (10) presenta muchas más ecuaciones que incógnitas, y en general no tiene solución. Lo mejor que se puede obtener en tal caso es la solución de mínimos cuadrados:

$$\mathbf{p}^* = (\bar{\mathbf{R}}^T\bar{\mathbf{R}})^{-1}\bar{\mathbf{R}}^T\bar{\mathbf{y}} \tag{11}$$

que es la que mejor ajusta los datos de entrada-salida del experimento a (10), en el sentido de que minimiza el residuo $\|\bar{\mathbf{y}} - \bar{\mathbf{R}} \cdot \mathbf{p}\|^2$. Por otro lado, una vez se tiene $\mathbf{p}$, podemos extraerle los valores de a_{ij} y b_{ij}, resolviendo el sistema no-lineal (7) mediante el método iterativo de Newton-Raphson a partir de una semilla inicial $\boldsymbol{\xi}_0$ del vector de 12 incógnitas a resolver: $\boldsymbol{\xi} = [a_{31}, \dots, a_{34}, a_{41}, \dots, a_{44}, b_{31}, b_{32}, b_{41}, b_{42}]^T$.

Algoritmo: método de Newton-Raphson para resolver $\boldsymbol{\xi}$
Repetir:
Resolver $\Delta\boldsymbol{\xi}$ de: $\mathbf{J}(\boldsymbol{\xi}_k) \cdot \Delta\boldsymbol{\xi} = -\mathbf{F}(\boldsymbol{\xi}_k)$ (12)
Actualizar: $\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k + \Delta\boldsymbol{\xi}$
Mientras: $\|\Delta\boldsymbol{\xi}\| > \varepsilon$ (e.g. $\varepsilon = 0.001$)

Donde $\mathbf{F}(\boldsymbol{\xi})$ es un vector columna de tamaño 16x1 formado por el lado izquierdo de las ecuaciones de (7), tras trasladar todos los términos al lado izquierdo de dichas ecuaciones, es decir: $\mathbf{F}(\boldsymbol{\xi}) = [p_1 - b_{31}, p_2 - b_{32}, \dots, p_{16} + a_{33} + a_{44}]^T$. $\mathbf{J}$ es la matriz Jacobiana de $\mathbf{F}$ respecto a $\boldsymbol{\xi}$, de tamaño 16x12. El problema de tener que resolver $\Delta\boldsymbol{\xi}$ de (12) es que se trata de un sistema lineal de 16 ecuaciones y tan solo 12 incógnitas, por lo que tampoco tiene solución. De nuevo, lo mejor que puede obtenerse es una solución de mínimos cuadrados. Por tanto, para realizar las iteraciones de Newton-Raphson, resolveremos (12) mediante la solución de mínimos cuadrados, que proporciona los valores de a_{ij} y b_{ij} que, al sustituirlos en (7), generarán un valor de $\mathbf{p}$ lo más parecido posible (en el sentido de mínimos cuadrados) al valor de $\mathbf{p}^*$ obtenido en (11).

4.1.3. Parte 3: resultados

Si aplicamos todo el procedimiento anterior a los datos de las gráficas mostradas en la Figura 6a, se obtienen las siguientes matrices del modelo de estado para el robot 3RRR:

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 36.885 & 25.2458 & 10.7234 & 9.8449 \\ 0.9882 & 0.7443 & -0.4639 & -3.2258 \end{bmatrix}$$
$$\mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1.4583 & 0.4799 \\ 0.0434 & 0.6395 \end{bmatrix} \cdot 10^3 , \tag{13}$$

donde, como ya se ha indicado anteriormente, el ángulo θ_2 se mantiene aproximadamente constante. Como se demostrará en la siguiente subsección 4.2, este modelo de estado (13) resulta muy apropiado a efectos de diseñar un controlador integral MIMO por realimentación del estado, a pesar de haber sido necesario concatenar dos soluciones de mínimos cuadrados durante la identificación: una para obtener el $\mathbf{p}^*$ óptimo en (11), y otra para obtener los valores de a_{ij} y b_{ij} que mejor satisfacen (7) para un valor dado de $\mathbf{p}$.

En el video adjuntado como material adicional, se muestra el experimento de identificación descrito en esta sección.

Para terminar esta sección, nótese que el experimento de identificación que se ha descrito se realiza con el sistema en bucle cerrado con un regulador PID que estabilice al robot, ya que los robots estudiados son inestables en bucle abierto: al introducirles un voltaje constante, sus articulaciones adquieren una velocidad constante en régimen permanente, por lo que las salidas (que son la integral de dichas velocidades) tendrían un comportamiento no acotado que complicaría la identificación.

4.2. Práctica 2: diseño de regulador MIMO

Partiendo del modelo dinámico lineal identificado en la anterior práctica, en esta segunda práctica el estudiantado debe diseñar un regulador por realimentación del estado con ajuste robusto del régimen permanente mediante control integral, según el esquema mostrado en la Figura 7. El objetivo es diseñar las ganancias $\mathbf{K}_C$ y $\mathbf{K}_I$ de dicho esquema para lograr que, en bucle cerrado, el sistema exhiba un determinado

comportamiento dinámico (tiempo de establecimiento y sobreoscilación) y un error nulo en régimen permanente. El método sugerido para calcular dichas ganancias de control es la extensión del método de Bass-Gura para asignación de polos en sistemas MIMO (véase el capítulo 10 de Bay, 1999). Por ejemplo, si las matrices del sistema son las de (13), y si deseamos que las salidas del sistema sigan referencias o setpoints de tipo escalón exhibiendo un comportamiento transitorio críticamente amortiguado con un tiempo de establecimiento (en una banda del $\pm 5\%$) de 1 segundo, por la aplicación del método de Bass-Gura obtendríamos las siguientes ganancias matriciales:

$$\mathbf{K}_C = \begin{bmatrix} 0.3549 & -0.23 & 0.0476 & -0.0213 \\ -0.0226 & 0.7514 & -0.004 & 0.0852 \end{bmatrix} \qquad \mathbf{K}_I = \begin{bmatrix} 0.7422 & -0.557 \\ -0.0504 & 1.6925 \end{bmatrix} \tag{14}$$

mientras que, si deseamos un comportamiento transitorio subamortiguado con una sobreoscilación del 25% y un tiempo de establecimiento de 1 segundo, obtendríamos las siguientes ganancias matriciales:

$$\mathbf{K}_C = \begin{bmatrix} 0.2063 & -0.1184 & 0.0342 & -0.0112 \\ -0.0125 & 0.4126 & -0.003 & 0.0547 \end{bmatrix} \qquad \mathbf{K}_I = \begin{bmatrix} 1.3343 & -1.0012 \\ -0.0906 & 3.0427 \end{bmatrix} \tag{15}$$

En la Figura 8 se muestran las gráficas de salidas (ángulos θ_i) para los experimentos de control realizados usando cada uno de estos dos controladores diseñados, para llevar el robot hasta el target $\mathbf{q}_2 = (8.3, -56.95)^{\text{o}}$, validando así tanto la identificación como el diseño de los controladores, ya que los requisitos de sobreoscilación y tiempo de establecimiento se satisfacen aproximadamente. Estas gráficas se han extraído directamente de experimentos reales de control, que pueden observarse también en el video adjunto a este artículo. Nótese, en la Figura 6b, que el target $\mathbf{q}_2$ seguido en este experimento de control está en una dirección d_c distinta de la dirección d_i en la que se movió el robot hacia el target $\mathbf{q}_1$ para realizar el experimento de identificación. Esto se ha hecho a propósito, para demostrar que el modelo identificado no solo es válido para controlar el robot en la dirección d_i en la que éste se movió para identificar su modelo de estado (como así se observa también en el mencionado video), sino también para controlarlo en otras direcciones, demostrando así robustez.

Por último, cabe destacar que para realimentar los estados correspondientes a las velocidades articulares (estados x_3 y x_4), dado que estas velocidades no se miden directamente con sensores, a efectos de realimentarlas en el control pueden estimarse, por ejemplo, mediante un derivador con filtro paso-bajo representado por la función de transferencia siguiente: $s/(1 + s/\omega_c)$, donde $\omega_c = 2\pi f$, siendo f la frecuencia de corte, que toma el valor de $f = 10$ Hz en los experimentos mostrados en la Figura 8. O, alternativamente, podría requerirse al estudiantado que diseñe un observador MIMO de orden reducido para estimar esos dos estados, toda vez que ya se conocen las matrices **A** y **B** tras identificar el sistema.

4.3. Práctica 3: análisis de controlabilidad

Los robots paralelos implementados en este laboratorio remoto, como la mayoría de robots manipuladores, exhiben un comportamiento no-lineal entre sus entradas (voltajes de los motores) y salidas (desplazamientos de sus articulaciones). No obstante, en asignaturas de control moderno es interesante estudiar dichos robots en torno a una configuración dada, lo que permite aproximar su comportamiento mediante un modelo lineal que puede estimarse mediante un procedimiento de identificación como el descrito en la Sección 4.1.

Estudiar la dinámica de un robot paralelo como un sistema linealizado en torno a un punto de trabajo también tiene otro interés en asignaturas de robótica y control, y es que permite estudiar cómo varía la controlabilidad del robot cuando éste adopta distintas configuraciones. Es bien sabido que los robots paralelos pueden encontrar singularidades llamadas "de tipo 2", donde no pueden controlarse todos los estados (Peidró et al., 2022). La teoría de control en el espacio de estados permite estudiar la controlabilidad calculando la siguiente matriz: $\mathbf{Q} = [\mathbf{B}, \mathbf{AB}, \mathbf{A}^2\mathbf{B}, \ldots, \mathbf{A}^{n-1}\mathbf{B}]$ (n = número de estados). Cuando $\mathbf{Q}$ tiene rango máximo, los estados pueden controlarse en cualquier dirección del espacio de estados. Cuando el robot se aproxima a una singularidad, el número de condición $\text{cond}(\mathbf{Q})$ aumenta, indicando una degradación de controlabilidad que requerirá aplicar mayores acciones de control para mover los estados en algunas direcciones. En el límite, cuando el robot alcanza exactamente una singularidad de tipo 2, $\mathbf{Q}$ pierde rango y su número de condición tiende a infinito, existiendo direcciones del espacio de estados en las que el robot no puede controlarse mediante sus entradas (Peidró et al., 2022).

Teniendo esto en cuenta, esta tercera práctica consiste en que el estudiante mueva el robot a diferentes configuraciones, tanto alejadas de singularidades como cercanas, identificando el modelo dinámico linealizado del robot en torno a cada configuración (repitiendo en cada configuración las operaciones descritas en la Sección 4.1), obteniendo la matriz de controlabilidad $\mathbf{Q}$ en cada configuración, y comprobando que su número de condición aumenta cerca de las singularidades, denotando una pérdida de controlabilidad.

Por ejemplo, calculando el número de condición para el modelo de estado (13) que se ha identificado en torno a la configuración $\mathbf{q}_0$ en la Sección 4.1.3, se obtiene: $\text{cond}(\mathbf{Q}) = 2.0696 \cdot 10^4$. Si se repite el experimento de identificación, pero identificando el robot en torno a una nueva configuración $\mathbf{q}_s = (-43.775,\ -73.325)^{\text{o}}$ mostrada en la Figura 6b (en lugar de $\mathbf{q}_0$), se identifican unas nuevas matrices de estado ($\mathbf{A}$, $\mathbf{B}$) que resultan en $\text{cond}(\mathbf{Q}) = 4.0773 \cdot 10^4$, que es mayor al que se tenía en la configuración $\mathbf{q}_0$, debido a que $\mathbf{q}_s$ está más cerca de singularidades que $\mathbf{q}_0$, como muestra la Figura 6b. Esto denota, por tanto, que el robot es más controlable cuando se mueve en torno a $\mathbf{q}_0$ que cuando se mueve en torno a $\mathbf{q}_s$.

5. Conclusiones y trabajos futuros

En este artículo hemos presentado un laboratorio remoto para realizar prácticas de control multivariable en el espacio de estados con dos robots paralelos. Se ha explicado la arquitectura del laboratorio, identificando sus componentes, y se han descrito varias prácticas para realizar remotamente con

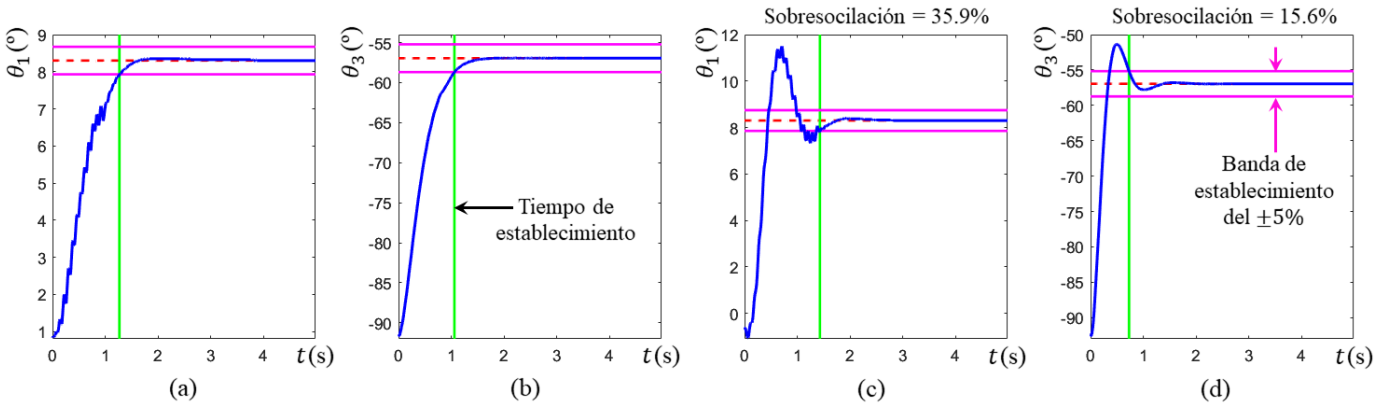

Figura 8: En azul: gráficas de las salidas del robot 3RRR real, controladas con comportamiento críticamente amortiguado (a-b), y con comportamiento subamortiguado (c-d). En magenta: banda de establecimiento del ±5%. En rojo a trazos: setpoint a alcanzar. En verde: tiempo de establecimiento.

dichos robots. Dichas prácticas permiten experimentar con la identificación de robots paralelos como sistemas MIMO, estudiar su controlabilidad y diseñar reguladores por estado realimentado. Las prácticas propuestas han sido llevadas a cabo con éxito usando este laboratorio remoto en la asignatura "1785 Control en el Espacio de Estado" del Grado en Ingeniería Electrónica y Automática Industrial de la Universidad Miguel Hernández, durante el curso 2022/23.

Como trabajos futuros, incorporaremos nuevos robots al laboratorio remoto, así como nuevas prácticas que permitan estudiar el control y la observabilidad de modelos dinámicos no-lineales de los robots, empleando otras técnicas de control más allá de las expuestas en este artículo. Para ello, habilitaremos una pestaña de texto libre en la interfaz, que permita al estudiantado escribir su propio código de control, con el fin de no ceñirlo a un esquema de control predefinido. También incorporaremos la posibilidad de definir trayectorias continuas arbitrarias (y no solo referencias en escalón) para comprobar su seguimiento con los controladores diseñados.

Agradecimientos

Trabajo financiado por el Centro de Inteligencia Digital de Alicante mediante el proyecto "Hacia la formación práctica ubicua y digital en robótica mediante laboratorios remotos".

Referencias

Alashqar, H. A. H., 2007. Modeling and high precision motion control of 3 DOF parallel delta robot manipulator. Master thesis. The Islamic University of Gaza, Palestina.

Bay, J. S., 1999. Fundamentals of linear state space systems, 1st edition. McGraw-Hill Education, Londres.

Belda, K., Böhm, J., Valášek, M., 2003. State-space generalized predictive control for redundant parallel robots. Mechanics Based Design of Structures and Machines, 31(3), 413-432. DOI: 10.1081/SME-120022857

Bordalba, R., Porta, J. M., Ros, L., 2018. A singularity-robust LQR controller for parallel robots. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp. 5048-5054. DOI: 10.1109/IROS.2018.8594084

Castellanos, A. S., Santana, L. H., Rubio, E., Ching, I. S., Santonja, R. A., 2006. Virtual and remote laboratory for robot manipulator control study. International Journal of Engineering Education, 22(4), 702-710.

Esquembre, F., 2015. Facilitating the creation of virtual and remote laboratories for science and engineering education. IFAC-PapersOnLine, 48(29), 49-58. DOI: 10.1016/j.ifacol.2015.11.212

Farias, G., 2010. Adding Interactive Human Interface to Engineering Software. PhD thesis. Universidad Nacional de Educación a Distancia, Spain.

Farias, G., De Keyser, R., Dormido, S., Esquembre, F., 2010. Developing networked control labs: a Matlab and Easy Java Simulations approach. IEEE Transactions on Industrial Electronics, 57(10), 3266-3275. DOI: 10.1109/TIE.2010.2041130

Hsia, T. S., Lasky, T. A., Guo, Z., 1991. Robust independent joint controller design for industrial robot manipulators. IEEE Transactions on Industrial Electronics, 38(1), 21-25. DOI: 10.1109/41.103479

Korayem, M., Maddah, S. M., Taherifar, M., Tourajizadeh, H., 2014. Design and programming a 3D simulator and controlling graphical user interface of ICaSbot, a cable suspended robot. Scientia Iranica, 21(3), 663-681.

Li, C., Fu, L., Wang, L., 2018. Innovate engineering education by using virtual laboratory platform based industrial robot. In: 2018 Chinese Control and Decision Conf. IEEE, pp. 3467-3472. DOI: 10.1109/CCDC.2018.8407723

Marin, R., Sanz, P. J., Nebot, P., Wirz, R., 2005. A multimodal interface to control a robot arm via the web: a case study on remote programming. IEEE Transactions on Industrial Electronics 52(6), 1506-1520. DOI: 10.1109/TIE.2005.858733

Muñoz de la Peña, D., Domínguez, M., Gomez-Estern, F., Reinoso, O., Torres, F., Dormido, S., 2022. State of the art of control education. Revista Iberoamericana de Automática e Informática Industrial 19(2), 117-131. DOI: 10.4995/riai.2022.16989

Paccot, F., Andreff, N., Martinet, P., 2009. A review on the dynamic control of parallel kinematic machines: theory and experiments. The International Journal of Robotics Research, 28(3), 395-416. DOI: 10.1177/0278364908096236

Peidró, A., Payá, L., Valiente, D., Gil, A., Reinoso, O., 2022. Design of a simulation tool to study the controllability and state-space control of a parallel robot, In: INTED2022 Proceedings. IATED, pp. 3790-3800. DOI: 10.21125/inted.2022.1051

Santana, I., Ferre, M., Izaguirre, E., Aracil, R., Hernandez, L., 2013. Remote laboratories for education and research purposes in automatic control systems. IEEE Transactions on Industrial Informatics, 9(1), 547-556. DOI: 10.1109/TII.2011.2182518

Temeltas, H., Gokasan, M., Bogosyan, S., 2006. Hardware in the loop robot simulators for on-site and remote education in robotics. International Journal of Engineering Education, 22(4), 815-828.

Torres, F., Candelas, F. A., Puente, S. T., Pomares, J., Gil, P., Ortiz, F. G., 2006. Experiences with virtual environment and remote laboratory for teaching and learning robotics at the University of Alicante. International Journal of Engineering Education, 22(4), 766-776.

Tzafestas, C. S., Palaiologou, N., Alifragis, M., 2005. Experimental evaluation and pilot assessment study of a virtual and remote laboratory on robotic manipulation. In: Proceedings of the IEEE International Symposium on Industrial Electronics. IEEE, pp. 1677-1684. DOI: 10.1109/ISIE.2005.1529184

Wu, M., She, J. H., Zeng, G. X., Ohyama, Y., 2008. Internet-based teaching and experiment system for control engineering course. IEEE Transactions on Industrial Electronics, 55(6), 2386-2396. DOI: 10.1109/TIE.2008.921229

Zhang, Y., 2020. Control implementation and co-simulation of a 6-DOF TAU haptic device. Master thesis. KTH Royal Institute of Technology, School of Industrial Engineering and Management, Sweden.