APLICACIÓN DE ALGORITMOS INTERACTIVOS A SISTEMAS DE CONTROL

AUTOR
JHON NICOLÁS SOLORZA DOMÍNGUEZ

Universidad Distrital Francisco José de Caldas
Tecnología en Mecánica Industrial
Bogotá, Colombia
2023

APLICACIÓN DE ALGORITMOS INTERACTIVOS A SISTEMAS DE CONTROL

AUTOR
JHON NICOLÁS SOLORZA DOMÍNGUEZ

TRABAJO DE GRADO PARA OPTAR AL TITULO
TECNÓLOGO EN MECÁNICA INDUSTRIAL

DOCENTE DIRECTOR
LUINI LEONARDO CORTES HURTADO

Universidad Distrital Francisco José de Caldas
Tecnología en Mecánica Industrial
Bogotá, Colombia
2023

PAGINA DE ACEPTACIÓN

NOTA DE ACEPTACIÓN

_____

_____

_____

_____

_____

PRESIDENTE DEL JURADO

_____

JURADO

_____

JURADO

_____

JURADO

_____

AGRADECIMIENTOS

Es un gran logro y honor para mí dedicar y agradecer a todas aquellas personas que de alguna u otra manera hicieron parte de este trabajo de grado y la culminación del mismo.

En primer lugar agradezco a mi familia, en especial a mi madre quien siempre me brindo su más sincero apoyo, amor y aliento a cada día superarme y demostrar de lo que soy capaz en los momentos más difíciles y de incertidumbre. El respaldo de mi familia ha sido fundamental para mi desarrollo académico, laboral y personal

De igual manera agradezco a mis profesores y compañeros de carrera, quienes me brindaron su apoyo, sus conocimientos y experiencias que fueron importantes en el desarrollo del trabajo, este es el resultado de lo que se empezó a formar algún tiempo atrás.

Finalmente, quiero agradecer a mi tutor de tesis Luini Cortes por su guía e instrucción a lo largo de todo este proceso, su sabiduría me permitió conocer nuevas formas de hacer las cosas y una visión más amplia dentro del tema desarrollado en este proyecto. Su apoyo fue importante para alcanzar este logro.

A todos en general, gracias por la confianza depositada en mí, este trabajo de grado muestra un gran esfuerzo y dedicación, es tan mío como de todos que estuvieron física y moralmente, es un esfuerzo que me hace sentir orgulloso.

Atentamente

JHON NICOLÁS SOLORZA DOMÍNGUEZ

Contenido

1. INTRODUCCIÓN---------- 10
2. ESTADO DEL ARTE: ALGORITMOS, HISTORIA Y APLICACIONES ---------- 11
2.1 ALGORITMOS ILUSTRATIVOS Y EDUCATIVOS ---------- 15
3. METODOLOGÍA---------- 19
4. DISEÑO DE SISTEMAS DE CONTROL UTILIZANDO ALGORITMOS INTERACTIVOS CON LIVE EDITOR ---------- 23
4.1 CONTROL DE POSICIÓN DEL CHASIS DE UNA MOTOCICLETA ---------- 23
4.1.1 ECUACIONES DE MOVIMIENTO DEL SISTEMA ---------- 26
4.1.2 MODELOS DE FUNCIÓN TRANSFERENCIA ---------- 26
4.1.3 PARÁMETROS DEL SISTEMA E INTRODUCCIÓN DE FÓRMULAS A ENTORNO MATLAB ---------- 27
4.1.4 REQUERIMIENTOS DE DISEÑO ---------- 28
4.1.5 RESPUESTA EN LAZO ABIERTO ---------- 28
4.1.6 IMPLEMENTACIÓN DE UN CONTROLADOR PID ---------- 32
4.1.7 RESPUESTA EN LAZO CERRADO ---------- 33
4.1.8 ANÁLISIS DEL SISTEMA EN ENTORNO DE SIMULINK---------- 34
4.1.9 EXTRACCIÓN DEL MODELO SIMULINK A MATLAB---------- 35
4.1.10 IMPLEMENTACIÓN DE CONTROL EN ESPACIO DE SIMULINK-------- 37
4.1.11 MODELO DEL SISTEMA EN ENTORNO SIMSCAPE---------- 40
4.1.12 DEFINIR LA ENTRADA DE CONTROL Y MEDICIÓN DE ESTADOS ADICIONALES---------- 43
4.1.13 CONTROL EN CIRCUITO CERRADO---------- 45
4.2 CONTROL DE MOVIMIENTO DE UN AVIÓN COMERCIAL ---------- 49
4.2.1 ECUACIONES DE MOVIMIENTO DEL SISTEMA ---------- 50
4.2.2 MODELOS DE FUNCIÓN DE TRANSFERENCIA Y ESPACIO DE ESTADO---------- 51
4.2.3 PARÁMETROS DEL SISTEMA E INTRODUCCIÓN DE FÓRMULAS A ENTORNO MATLAB ---------- 52
4.2.4 REQUERIMIENTOS DE DISEÑO ---------- 53
4.2.5 RESPUESTA EN LAZO ABIERTO ---------- 54
4.2.6 RESPUESTA EN LAZO CERRADO ---------- 55
4.2.7 IMPLEMENTACIÓN DE UN CONTROLADOR PID ---------- 62
4.2.8 CONTROL PROPORCIONAL (Kp)---------- 62
4.2.9 CONTROL PI---------- 72
4.2.10 CONTROL PID ---------- 73
4.2.11 ANÁLISIS DEL SISTEMA EN ENTORNO DE SIMULINK---------- 78
4.2.12 AJUSTE CON PID AUTOMATIZADO EN SIMULINK ---------- 87
4.3 CONTROL DE EQUILIBRIO A UN PÉNDULO INVERTIDO---------- 95
4.3.1 ANÁLISIS DE FUERZAS Y ECUACIONES DEL SISTEMA ---------- 97
4.3.2 REPRESENTACIÓN POR FUNCIONES DE TRANSFERENCIA---------- 99
4.3.3 REPRESENTACIÓN EN MATLAB DE LAS FUNCIONES DE TRANSFERENCIA----------100
4.3.4 REPRESENTACIÓN EN EL ESPACIO DE ESTADOS----------101
4.3.5 MODELADO EN SIMULINK/SIMSCAPE MULTIBODY----------104
4.3.6 ANÁLISIS DEL COMPORTAMIENTO----------105
4.3.7 DISEÑO DE CONTROLADOR PID ----------109
4.3.8 CONTROL PID EN SIMULINK/SIMSCAPE MULTIBODY ----------114

4.3.9 CONTROL EN EL ESPACIO DE ESTADOS ---------- 115
5. ANÁLISIS DE RESULTADOS ---------- 121
6. CONCLUSIONES ---------- 124
7. REFERENCIAS ---------- 124

Contenido de ilustraciones

Ilustración 1 Diagrama de flujo de la metodología de programación de algoritmos interactivos para solución de problemas matemáticos [los autores]. 20
Ilustración 2 Planteamiento de la metodología de desarrollo de los ejercicios mediante un esquema grafico 22
Ilustración 3 Modelo de moto, imagen tomada y adaptada de https://www.canariasenmoto.com/motocicletas/benelli-bx-enduro.html 24
Ilustración 4 DCL para suspensión de motocicleta 25
Ilustración 5 Respuesta al escalón unitario de G1 29
Ilustración 6 Respuesta al escalón unitario de G2 30
Ilustración 7 Esquema del sistema en lazo cerrado 31
Ilustración 8 Esquema del sistema donde interactúa F(s) y G1(s)=G2(s) 31
Ilustración 9 Respuesta al escalón unitario bajo controlador PID 33
Ilustración 10 Diagrama de bloques en Simulink 34
Ilustración 11 Respuesta de diagrama de bloques de Simulink 35
Ilustración 12 Diagrama de bloques en Simulink hacia Matlab 36
Ilustración 13 Respuesta de la entrada simulada en entorno de Matlab 37
Ilustración 14 Diagrama de bloques en Simulink hacia Matlab de varias salidas 38
Ilustración 15 Diagrama de bloques con retroalimentación en Simulink 39
Ilustración 16 Respuesta al diagrama de bloques con retroalimentación 40
Ilustración 17 Modelo del sistema en entorno Simscape 42
Ilustración 18 Respuesta del modelo de Simscape 43
Ilustración 19 Diagrama de bloques en entorno de Simscape 44
Ilustración 20 Respuesta al diagrama de bloque en entorno Simscape 45
Ilustración 21 Diagrama de bloque en Simscape con varias salidas 46
Ilustración 22 Diagrama de bloques con retroalimentación en Simscape 47
Ilustración 23 Respuesta al Diagrama de bloques con retroalimentación en Simscape 48
Ilustración 24 Modelo avión y diagrama de cuerpo, Autoría propia 50

Ilustración 25 Respuesta al escalón unitario en lazo abierto 54
Ilustración 26 Esquema del sistema en lazo cerrado 56
Ilustración 27 Respuesta del sistema de lazo cerrado 57
Ilustración 28 Análisis de la gráfica respuesta del sistema de lazo cerrado . 57
Ilustración 29 Respuesta en lazo cerrado 61
Ilustración 30 Esquema del sistema con retroalimentación 62
Ilustración 31 Interfaz gráfica apartado Control System Designer 63
Ilustración 32 Configuración compensador F 64
Ilustración 33 Configuración compensador C 65
Ilustración 34 Creación de nueva gráfica en entorno del control system 66
Ilustración 35 Configuración de la gráfica 67
Ilustración 36 Resultado de la gráfica resultante 68
Ilustración 37 Menú tuning methods 69
Ilustración 38 Configuración compensador C 70
Ilustración 39 Comportamiento del sistema con ganancia Kp 71
Ilustración 40 Fórmula compensador C y controlador PI 72
Ilustración 41 Comportamiento del sistema con ganancia Kp y Ki..... 73
Ilustración 42 Fórmula compensador C y controlador PID..... 74
Ilustración 43 Comportamiento del sistema con ganancia Kp, Ki y Kdi 75
Ilustración 44 Ajuste manual controlador PID 76
Ilustración 45 Ajuste manual del controlador final PID 77
Ilustración 46 Comportamiento final con controlador PID 78
Ilustración 47 Esquema del sistema de la planta en lazo abierto..... 79
Ilustración 48 Parámetros de bloque State-Space 80
Ilustración 49 Diagrama de bloques con bloque State-Space 80
Ilustración 50 Respuesta del sistema de diagrama de bloques con State-Space..... 81
Ilustración 51 Parámetros bloque ganancia 82
Ilustración 52 Diagrama de bloques con retroalimentación 82
Ilustración 53 Respuesta diagrama de bloques con retroalimentación 83
Ilustración 54 Diagrama de bloques con retroalimentación y bloque N 84
Ilustración 55 Respuesta diagrama de bloques con retroalimentación y bloque N 85
Ilustración 56 Diagrama de bloques agregando una perturbación..... 86
Ilustración 57 Respuesta diagrama de bloques agregando una perturbación 87
Ilustración 58 Diagrama de bloques con perturbación y bloque PID..... 88
Ilustración 59 Respuesta diagrama de bloques con perturbación y bloque PID 89
Ilustración 60 Respuesta generada por controlador PID Tuning..... 90

Ilustración 61 Ajuste manual PID Tuning 91
Ilustración 62 Respuesta del ajuste manual PID Tuning 92
Ilustración 63 Parámetros bloque PID saturación 93
Ilustración 64 Respuesta del sistema con bloquue PID limitado en la saturación 94
Ilustración 65 Modelo péndulo invertido, imagen tomada y adaptada de https://ctms.engin.umich.edu/CTMS/Content/InvertedPendulum/System/Modeling/figures/pendulum.png 95
Ilustración 66 DCL para péndulo invertido, imagen tomada y adaptada de https://ctms.engin.umich.edu/CTMS/Content/InvertedPendulum/System/Modeling/figures/pendulum2.png 97
Ilustración 67 Modelado del sistema péndulo invertido en entorno simscape-multibody 104
Ilustración 68 Representación por diagrama de bloques del sistema 104
Ilustración 69 Respuesta en lazo abierto del sistema 106
Ilustración 70 Respuesta del sistema en lazo abierto del sistema con dos entradas 108
Ilustración 71 Esquema por diagrama de bloques del sistema sin control de un péndulo invertido 109
Ilustración 72 Reorganización del esquema por diagrama de bloques del sistema sin control de un péndulo invertido 110
Ilustración 73 Respuesta de la posición del péndulo con parámetros iguales a 1 111
Ilustración 74 Respuesta de la posición del péndulo variando la variable proporcional 112
Ilustración 75 Respuesta de la posición del péndulo variando la variable proporcional y la derivativa 113
Ilustración 76 Diagrama de bloques del sistema con controlador PID 115
Ilustración 77 Respuesta del sistema con un controlador LQR 118
Ilustración 78 Respuesta del sistema con varias entradas con controlador LQR 120

RESUMEN

Este trabajo presenta la metodología para aplicarla por medio de algoritmos interactivos a ejemplos de sistemas de control utilizando una herramienta de software. Inicialmente, se hace una búsqueda de la información relacionada con el concepto de algoritmo, su historia, evaluación y aplicación, hasta su estrecha conexión con la educación. Posteriormente, se hace uso de la herramienta de Live Editor de Matlab para la solución de los ejercicios de sistemas de control aquí propuestos. Finalmente, se presentan los resultados y las conclusiones de este trabajo, que evidencian una mejor manera de desarrollar y entender este tipo de problemas matemáticos en ingeniería.

El fin de este trabajo es ayudar a crear una estrategia de aprendizaje ahora que nos hemos visto involucrados de manera casi directa con la educación remota. El desarrollo de cursos en el que se involucran conceptos, algoritmos, simulaciones y demás con lo que también, paralelamente, se realiza desde las aulas. Será un paso importante en la educación en tecnología e ingeniería incluyendo herramientas basadas en la nube que permiten a los docentes y alumnos acceder en cualquier momento y lugar al contenido del curso.

PALABRAS CLAVES: Algoritmo, educación, interacción, script, simulación.

1. INTRODUCCIÓN

Las metodologías de enseñanza se han venido reinventado a través de los años y de los sucesos que marcan el diario vivir, podemos decir entonces, que la enseñanza y el contenido temático varía con el tiempo, de igual forma sucede con los sistemas de control y su aplicación en la vida misma. Anteriormente la solución de estos sistemas sin importar su naturaleza era resuelto con algoritmos complejos de entender y solo desarrollados por profesionales que entendían el concepto [Orlando Regalón Anias et al., 2012], sin embargo, la propuesta que se hace y explica es para generar todo lo contrario involucrando de manera directa textos que pueden servir como guía de instrucción, imágenes o hipervínculos que complementen mejor el contenido, entre muchas otras herramientas que serán lo que irá nutriendo el algoritmo, dando una buena solución, con mejor capacidad receptiva para los lectores y sin tantos tecnicismos.

Un algoritmo es un conjunto ordenado de instrucciones o pasos diseñados para resolver un problema o realizar una tarea específica, en otras palabras son saberes direccionados e instruidos cuya característica principal es la resolución de problemas por medio de secuencias que están dictaminadas de manera congruente. Los algoritmos se utilizan comúnmente en la informática y va muy relacionado con los ingenieros de software y por lo general cualquier persona que no fuese instruida de tal forma, el uso de algoritmos se le dificultaba para la realización de una amplia variedad de aplicaciones, sin embargo, se podrá dar cuenta que la nueva incorporación de algoritmos interactivos está diseñado para todo público sin necesidad de ser ingeniero o técnico de software o de programación, y hará uso de los algoritmos como más le convenga y con facilidad.

Ahora bien, el manejo de un algoritmo interactivo será una nueva forma de ver los algoritmos y de ejecutarlos. Con el algoritmo interactivo se busca, por ejemplo, que con cualquier modificación en algunos de los parámetros se ocasionen cambios significativos dentro del programa esto con el fin de ir facilitando de una u otra manera el funcionamiento del sistema y que, como consecuencia, se busca que éste sea más sencillo de ejecutar y de recordar.

Es decir, se tiene como reto combinar la solución de los sistemas propuestos y la incorporación directa de texto, imágenes, hipervínculos, fórmulas, entre muchos otros. Para ello, es necesario definir la correlación de la solución con

la explicación. A juicio personal, cada vez que se define un procedimiento para la ejecución de un sistema o en cualquier proceso se reduce el tiempo invertido en el desarrollo y en efecto, es algo que se consiguió, es probable que el definir unos pasos a realizar antes de llevar a cabo algún proceso sea el futuro para la ejecución de problemas.

2. ESTADO DEL ARTE: ALGORITMOS, HISTORIA Y APLICACIONES

El estado del arte de este proyecto de grado investigativo está realizado de la manera más rigurosa y confiable posible, con la ayuda del software VosViewer®. En su página web (https://www.vosviewer.com/) indica que es útil para la construcción y visualización de redes bibliométricas, a partir de autoría o coautoría de documentos científicos como medición cuantificable de la colaboración o referenciación de investigadores, instituciones y hasta países. Además, este software nos entrega la información por medio de un mapa de densidad, de superposición de colores y tamaño con los nombres de los autores con más conexiones.

Para el caso particular del tema de investigación, se realizó una búsqueda de los principales aportes a la algoritmia, a continuación se relacionan los trabajos más sobresalientes:

El trabajo presentado por Kowalkiewicz (Kowalkiewicz, M., Octubre 2019), menciona aspectos importantes de la historia de los algoritmos. Hasta hace poco, los algoritmos eran un dominio de los informáticos. Pero ahora, han entrado en nuestras vidas y se están generalizando. Algoritmo ya no es una palabra extraña. El comercio algorítmico o comúnmente conocido en inglés como Algorithmic trading, el sesgo algorítmico también mejor conocido en inglés como Algorithmic bias, el algoritmo de Facebook, incluso la guerra algorítmica: todos estos términos se han convertido en parte de nuestro vocabulario en los últimos años. Algunas personas llegan a afirmar que vivimos en una nueva era: la era del algoritmo. Pero los algoritmos no son tan nuevos y recientes como se piensan que son. Los hemos usado, sabiendo si existían o no, durante cientos y miles de años. Los algoritmos son simplemente descripciones específicas de acciones de un paso a paso que se deben hacer para lograr un resultado en particular. Son uno de los instrumentos más comunes de intercambio de conocimientos.

No fue hasta la década de 1930 cuando vimos las primeras menciones y connotaciones de algoritmos en computadoras electrónicas (no mecánicas). Alan Turing fue uno de los primeros científicos que capturó formalmente cómo los individuos realizaban cálculos. El objetivo de Turing era capturar un proceso general, en lugar de uno específico para una tarea particular, como identificar números primos o calcular el mayor divisor común. El proceso general podría usarse para realizar tareas específicas. El trabajo conceptual de Turing condujo al desarrollo de lo que ahora se conoce como la Máquina de Turing. Ésta, a su vez, condujo a la aparición de computadoras de propósito general que son las que hoy en día la mayoría del mundo tiene. El prefijo de propósito general es esencial aquí. Contrariamente a las máquinas anteriores, las nuevas computadoras ejecutarían conjuntos arbitrarios de instrucciones. Podrían ser utilizados para fines que ni siquiera predijeron sus creadores. En otras palabras: el trabajo de Turing condujo al desarrollo de computadoras en las que podemos instalar y ejecutar aplicaciones como lo hacemos hoy en día de manera muy natural.

Igualmente, Turkel (Ahmet Fatih Turkel, A. F., 2023), presenta una historia un poco más resumida, lo que quiere decir que se resaltan los datos importantes y destacados por el autor para así explicar de una manera breve el tema. El algoritmo es uno de los descubrimientos más importantes de la historia de la humanidad. Es un conjunto ordenado de procesos que son instrucciones y operaciones. Estos procesos incluyen pasos para la solución de los problemas o realizar un cálculo. Según Britannica, el algoritmo es un procedimiento sistemático que produce, en un número finito de pasos, la respuesta a una pregunta o la solución de un problema. El origen del término es la traducción latina del tratado aritmético de al-Khwarizmi "Al-Khwarizmi Concerning the Hindu Art of Reckoning"; que es "Algoritmi de numero Indorum". El nombre "Algoritmi" significaba "el sistema numérico decimal". Ese fue el significado real durante siglos hasta que se convirtió en la definición que usamos hoy. Al-Khwārizmī era un erudito musulmán nacido en Khwarazm durante los siglos 8-9. Hoy, la humanidad le debe todo a esta palabra extraordinaria llamada Algoritmo. Son utilizados por casi todos los dispositivos electrónicos o computadoras. Todo lo que la gente usa desde sus dispositivos son las ejecuciones de algoritmos. Por ejemplo, aplicaciones móviles, videojuegos, sitios web, etc. En este artículo, hablamos y definimos sobre lo que era un algoritmo. Hablamos sobre cómo cambió con el tiempo.

De forma parecida, (codecruks, 2021), habla sobre la evolución que han tenido los algoritmos y además explica una de las ramas de estudio. Es una noción

antigua. Utilizamos algoritmos todo el tiempo, nos demos cuenta o no. Puede pensar en ellos como un elemento inseparable de cada actividad que tomamos en nuestra vida diaria. Los algoritmos no solo son fascinantes de estudiar, sino que también son críticos para el funcionamiento de nuestro mundo digital moderno.

La rama del estudio de problemas tratables y no tratables se conoce como teoría de la complejidad. Los algoritmos están siendo analizados en varios límites. El enfoque de los investigadores fue minimizar el tiempo de ejecución del límite inferior de los algoritmos. Más tarde, desarrollaron métodos formales de especificación, que describen la naturaleza precisa de la entrada para verificar la corrección del algoritmo. Por lo general, los algoritmos están diseñados para resolver problemas específicos solamente, por lo que están obligados a operar solo con entradas específicas. Ningún algoritmo funciona en todas las entradas. Por lo tanto, con el desarrollo de algoritmos, ha sido esencial especificar los límites de entrada y salida esperada del algoritmo. El estudio y desarrollo de algoritmos eficientes condujo a otro aspecto llamado equilibrio espacio-tiempo. Esto abre el estudio de problemas que son manejables en el tiempo y manejables en el espacio. Los problemas manejables son aquellos que pueden resolverse en un tiempo razonablemente aceptable. Los algoritmos de aproximación entraron en escena para proporcionar una solución cerca a problemas no tratables en un tiempo razonablemente aceptable. Los algoritmos aleatorios han dado una nueva dirección para resolver muchos problemas que consumen mucho tiempo. La aleatoriedad puede considerarse como una parte integral de muchas estrategias algorítmicas, como el recocido simulado y los algoritmos genéticos.

Además de la historia y evolución de los algoritmos, también es importante conocer las formas o tipos de aprendizaje automático dado por los desarrollos, avances y la evolución de los algoritmos. (Janiesch et al., 2021) mencionan eb su artículo que hoy en día, los sistemas inteligentes que ofrecen capacidades de inteligencia artificial a menudo se basan en el aprendizaje automático. El aprendizaje automático describe la capacidad de los sistemas para aprender de los datos de entrenamiento específicos del problema para automatizar el proceso de creación de modelos analíticos y resolver las tareas asociadas. El aprendizaje profundo es un concepto de aprendizaje automático basado en redes neuronales artificiales. Para muchas aplicaciones, los modelos de aprendizaje profundo superan a los modelos de aprendizaje automático poco profundos y los enfoques tradicionales de análisis de datos. En el artículo, se resumen los fundamentos del aprendizaje automático y el aprendizaje profundo para generar una comprensión más amplia de la base metódica de

los sistemas inteligentes actuales. En particular, ellos proporcionan una distinción conceptual entre términos y conceptos relevantes, explicar el proceso de construcción de modelos analíticos automatizados a través del aprendizaje automático y el aprendizaje profundo, y discutir los desafíos que surgen al implementar dichos sistemas inteligentes en el campo de los mercados electrónicos y los negocios en red. Estos, naturalmente, van más allá de los aspectos tecnológicos y resaltan problemas en la interacción hombre-máquina y el servicio de la inteligencia artificial.

Por otro lado, Losing, Hammer y Wersing (Losing y otros, 2018), comentan que recientemente el aprendizaje incremental y en línea ganó más atención, especialmente en el contexto de big data y aprendizaje de flujos de datos, en conflicto con la suposición tradicional de disponibilidad completa de datos. A pesar de que hay una variedad de métodos diferentes disponibles, a menudo no está claro cuál de ellos es adecuado para una tarea específica y cómo se desempeñan en comparación entre sí. Se analizan las propiedades clave de ocho métodos incrementales populares que representan diferentes clases de algoritmos. De este modo, los evaluamos con respecto a su error de clasificación en línea, así como a su comportamiento en el límite. Además, se discute el tema a menudo descuidado de la optimización de hiperparámetros, específicamente para cada método y se prueba qué tan robustamente se puede hacer en función de un pequeño conjunto de ejemplos. La extensa evaluación de conjuntos de datos con diferentes características ofrece una visión general del rendimiento con respecto a la precisión, la velocidad de convergencia y la complejidad del modelo, lo que facilita la elección del mejor método para una aplicación determinada.

En el artículo de Burke (Burke, A., 2019), se revisan dos definiciones de algoritmo, sus usos y sus modelos implícitos de computación en la sociedad. La primera, denominada definición de programación estructural, se alinea más con el uso en informática, y como su nombre indica, el proyecto intelectual de la programación estructurada. La segunda, denominada definición sistemática, es más informal y surge de observaciones etnográficas de discusiones sobre software tanto en entornos profesionales como cotidianos. Se comparten ejemplos específicos de localización de algoritmos dentro de bases de código modernas, así como código que afecta directamente las preocupaciones sociales y éticas. La distinción estructural entre algoritmos y preocupaciones sociales se explica como un reflejo de la construcción de ingeniería de algoritmos y estructuras de datos. Se propone que, en lugar de que esta separación sea un intento de imponer un límite

profesional y evadir la responsabilidad social, sea una distinción técnica crucial dentro del código que lo haga más claro y transparente. Se reconsideran las estructuras de poder reforzadas por las interpretaciones culturales más amplias del algoritmo, junto con lo que significaría para el software tener una cultura de diseño inclusiva.

2.1 ALGORITMOS ILUSTRATIVOS Y EDUCATIVOS

A partir de acá el enfoque de los temas será distinto, sin embargo, manteniendo el mismo hilo de algoritmos y todos los subtemas que lo rodean, pues bien, se va a hablar de estos subtemas, como se relacionan entre sí, cuáles han sido esos avances significativos para que hoy en día rijan de la manera en que lo hacen. Nwankpa (Nwankpa, C. E., 2020), presenta los avances, optimización y las técnicas de un aprendizaje profundo basado con algoritmos.

En la última década, el aprendizaje profundo (DL, siglas en inglés de Deep Learning) ha sido testigo de excelentes rendimientos en una variedad de problemas, incluido el reconocimiento de voz, el reconocimiento de objetos, la detección y el procesamiento del lenguaje natural (NLP, siglas en inglés de Natural Lenguaje Process), entre muchos otros. De estas aplicaciones, un desafío común es obtener parámetros ideales durante el entrenamiento de las redes neuronales profundas (DNN, siglas en inglés de Deep neural Networks). Estos parámetros típicos se obtienen mediante algunas técnicas de optimización que han sido ampliamente estudiadas. Estas investigaciones han producido resultados de vanguardia (SOTA) sobre mejoras de velocidad y memoria para arquitecturas de redes neuronales profundas (NN, siglas en inglés de Neuronal Networks). Sin embargo, los optimizadores SOTA han seguido siendo un área de investigación activa sin compilaciones de los optimizadores existentes reportados en la literatura. Este artículo proporciona una visión general de los avances recientes en algoritmos de optimización y técnicas utilizadas en DNN, destacando los optimizadores SOTA actuales, las mejoras realizadas en estos algoritmos y técnicas de optimización, junto con las tendencias en el desarrollo de optimizadores utilizados en el entrenamiento de modelos basados en DL. Los resultados de la búsqueda en la base de datos Scopus de los optimizadores en DL proporcionan los artículos reportados como el resumen de los optimizadores DL. Por lo que se puede decir, no existe una compilación exhaustiva de los algoritmos y técnicas de optimización desarrollados y utilizados hasta ahora en la investigación y aplicaciones de DL, y este documento resume estos hechos.

Adicionalmente, Sabato & Hess (Sabato, S., & Hess, T., 2018). Consideran algoritmos interactivos en la configuración basada en grupos y en la configuración basada en flujos. Los algoritmos interactivos observan los elementos sugeridos (que representan acciones o consultas), seleccionan interactivamente algunos de ellos y reciben respuestas. Los algoritmos basados en grupos pueden seleccionar elementos en cualquier orden, mientras que los algoritmos basados en flujos observan elementos en secuencia y solo pueden seleccionar elementos inmediatamente después de observarlos. Además, consideramos un ajuste intermedio, que denominamos flujo pre cognitivo, en el que el algoritmo conoce de antemano la identidad de todos los elementos de la secuencia, pero puede seleccionarlos solo en el orden de su aparición. Para todas las configuraciones, asumimos que los elementos sugeridos se generan independientemente de alguna distribución de origen, y preguntamos cuál es el tamaño de flujo requerido para emular un algoritmo de grupo con un tamaño de grupo determinado, en la configuración basada en flujo y en la configuración de flujo pre cognitivo. Proporcionamos algoritmos y límites inferiores coincidentes para algoritmos de grupos generales y para algoritmos de grupos basados en utilidades. Además, derivamos límites superiores e inferiores casi coincidentes en la brecha entre los dos entornos para el caso especial de aprendizaje activo para la clasificación binaria.

Los algoritmos se han encargado en los últimos años de llevar la educación a otro nivel, es por ello, que es necesario hablar de cómo se han visto involucrado estos algoritmos y que tipo de algoritmos son usados con este propósito de enseñanza, así es como (Chen & Zhao, 2018) hablan y hacen una investigación bastante interesante para mejorar el plan de enseñanza de la escuela, optimizar el sistema de aprendizaje en línea y ayudar a los estudiantes a lograr mejores resultados de aprendizaje, se estableció un modelo educativo de minería de datos para la supervisión del proceso de aprendizaje electrónico. Se aplicaron técnicas de análisis estadístico y visualización en minería de datos, algoritmos de reglas de asociación y algoritmos de agrupamiento. Se analizaron sistemáticamente los datos de enseñanza de una plataforma de gestión de enseñanza de inglés universitario. Se llegó a una conclusión relacionada sobre la relación entre los efectos del aprendizaje de inglés de los estudiantes y los hábitos de aprendizaje en línea. Los resultados mostraron que este método podría ayudar efectivamente a los maestros a juzgar los resultados de aprendizaje en línea de los estudiantes, comprender su estado de aprendizaje en línea y mejorar

su proceso de aprendizaje en línea. Por lo tanto, el modelo puede mejorar la eficacia de los estudiantes

Y esta reproducción de conocimientos con el fin de enseñar es importante si se quiere hablar de una evolución cognitiva así como mencionan (Zhou, Y., Zhang, S., Luo, Q., & Abdel-Baset, M. 2019), que el comportamiento cognitivo es un factor indispensable en la historia de la evolución humana; a través del comportamiento cognitivo, el desarrollo humano ha evolucionado desde la Edad de Piedra hasta la actual era de la información de alta tecnología. Al simular el proceso de evolución cultural, un algoritmo cultural puede reflejar el proceso evolutivo de la sociedad con precisión, lo que puede acelerar el proceso de resolución de problemas. En este artículo, la evolución del comportamiento cognitivo y un algoritmo cultural se combinan para formar el algoritmo de optimización de la evolución cognitiva cultural en sus siglas en inglés (Culture Cognitive Evolution Optimization, CCEO). Un algoritmo cultural con la estructura de doble capa del espacio de creencias y el espacio de población se transforma en un mecanismo de evolución de tres capas en un CCEO para que el espacio de creencias mejore la siguiente capa de orientación cuidadosa del proceso evolutivo. Se utilizaron veintitrés funciones de prueba de referencia y dos problemas de ingeniería para probar el algoritmo CCEO, y los resultados demuestran que el algoritmo de evolución cognitiva de la cultura tiene una alta velocidad de convergencia y precisión de cálculo.

La información de los algoritmos siempre tiende hacer extensa y más cuando se trata de aplicaciones, sin embargo acá no se va a centrar tanto en aplicaciones, sino en la evolución de los mismos en los diferentes ámbitos, así entonces en el ámbito de la educación los algoritmos han sido un tema importante, y es precisamente porque en la academia es donde surgen la mayoría de los avances cuando de algoritmos se trata, también una cuestión de que el conocimiento se transmita y que sea debatido, esto también genera avance. Baker & Hawn, (2021) menciona en un artículo, que si revisamos el sesgo algorítmico en la educación, discutiendo las causas de ese sesgo y revisando la literatura empírica sobre las formas específicas en que se sabe que el sesgo algorítmico se ha manifestado en la educación. Mientras que otros trabajos recientes han revisado las definiciones matemáticas de equidad y los enfoques algorítmicos ampliados para reducir el sesgo, la revisión se centra en solidificar la comprensión actual de los impactos concretos del sesgo algorítmico en la educación: qué grupos se sabe que se ven afectados y qué etapas y agentes en el desarrollo y despliegue de algoritmos educativos están implicados. Discutimos perspectivas teóricas y formales sobre el sesgo

algorítmico, conectamos esas perspectivas con la canalización de aprendizaje automático y revisamos las métricas para evaluar el sesgo. A continuación, revisamos la evidencia en torno al sesgo algorítmico en la educación, comenzando con las categorías más estudiadas de raza / etnia, género y nacionalidad, y pasando a la evidencia disponible de sesgo para categorías menos estudiadas, como el estado socioeconómico, la discapacidad y el estado relacionado con el ejército. Reconociendo las brechas en lo que se ha estudiado, proponemos un marco para pasar del sesgo desconocido al sesgo conocido y de la justicia a la equidad. Discutimos los obstáculos para abordar estos desafíos y proponemos cuatro áreas de esfuerzo para mitigar y resolver los problemas de sesgo algorítmico en los sistemas AIED y otras tecnologías educativas.

Para concluir esta sección se dará a conocer información relevante de los algoritmos y su estrecha relación con la educación, pues se sabe que todos estos avances son ampliamente usados en el ámbito educacional, pues bien (Yamashita, K., Fujioka, R., Kogure, S., Noguchi, Y., Konishi, T., & Itoh, Y. 2016) describen tres ejercicios prácticos relacionados con la educación de algoritmos. Los ejercicios se basan en un sistema de apoyo al aprendizaje que ofrece visualización del comportamiento del programa. Los sistemas con la capacidad de visualizar el comportamiento del programa son efectivos para promover la comprensión del comportamiento del algoritmo. Se espera que la introducción de estos sistemas en un curso de algoritmos permita a los estudiantes cultivar una comprensión más profunda. Sin embargo, casi todos los sistemas existentes no pueden incorporar la intención de instrucción del maestro que puede ser necesaria para acomodar a los estudiantes con diferentes habilidades mediante el uso de un enfoque de instrucción diferente. En base a estas consideraciones, realizan sesiones de práctica en el aula como parte de un curso de algoritmos incorporando el sistema de visualización que desarrollan en un trabajo anterior. El sistema visualiza el mundo de dominio objetivo de acuerdo con la política de visualización definida por el profesor. El objetivo con las clases prácticas es permitir a los estudiantes comprender las propiedades de los algoritmos, como el número de comparaciones e intercambios de datos. Los contenidos del curso están estructurados de tal manera que las propiedades de un algoritmo pueden ser entendidas por descubrimiento de aprendizaje en el trabajo práctico. En este documento, se proporciona una visión general de las prácticas educativas y las respuestas de los alumnos y muestran que el marco que utilizan las prácticas se puede establecer en clases de algoritmos. Además, resumen los

requisitos para la inclusión del aprendizaje por descubrimiento en las clases de algoritmos como el conocimiento obtenido de las prácticas.

3. METODOLOGÍA

Los problemas de sistemas de control tienden hacer muy similares entre sí y pueden ser extrapolables, esto significa que, se pueden aplicar en distintas áreas dentro de cualquier conocimiento, no es necesario que esté dentro de la ingeniería de control específicamente. Los ejemplos escogidos se desarrollarán con el algoritmo interactivo descrito a continuación y con la herramienta de [MathWorks®, 2022] Live Script de Matlab, esta nueva herramienta se ha venido actualizando de manera constante siempre con buenas mejoras y aciertos, en su interfaz nos mostrará 2 ventanas de forma vertical, una mostrará la parte del código, el texto, la inserción de imágenes y fórmulas, hipervínculos, etc. y la otra ventana nos mostrará el compilado del código (esto es ajustable a conveniencia, lo descrito aquí es la forma que está como predeterminado). ¿La novedad? Este apartado de Matlab (Live Script o Live Editor) garantizará un aprendizaje moderno y sencillo, la reducción en tiempos de desarrollo, la interacción de herramientas como inclusión de imágenes, controles de variables, hipervínculos entre muchas otras.

Los problemas escogidos para la representación y puesta a prueba de estos algoritmos interactivos son aquellos encontrados en un apartado de la universidad de Michigan para el control de ejercicios. Se hizo la realización de algunos de estos ejercicios pero dentro del entorno de Live Script de Matlab, no en un Script como se encuentran, esto con el fin de tener el desarrollo del ejercicio de manera más clara e interactiva y así mismo poner a prueba la metodología descrita a continuación. A su vez, el programa nos da la facilidad de replicar el problema de forma gráfica con mapas, diagramas de bloques y la representación misma del componente a utilizar, este último se podrá ubicar de manera individual lo que será un plus a la hora de mostrar y enseñar el algoritmo. Finalmente, existe la posibilidad de generar un documento extraíble de la solución del problema para una posible distribución de fácil y libre acceso, los formatos en los que se puede realizar la descarga para distribución son: PDF, .DOCX, .HTML, Latex.

El Diagrama de flujo a continuación explica de manera general el procedimiento que se debe llevar a cabo para la programación de algoritmos interactivos.

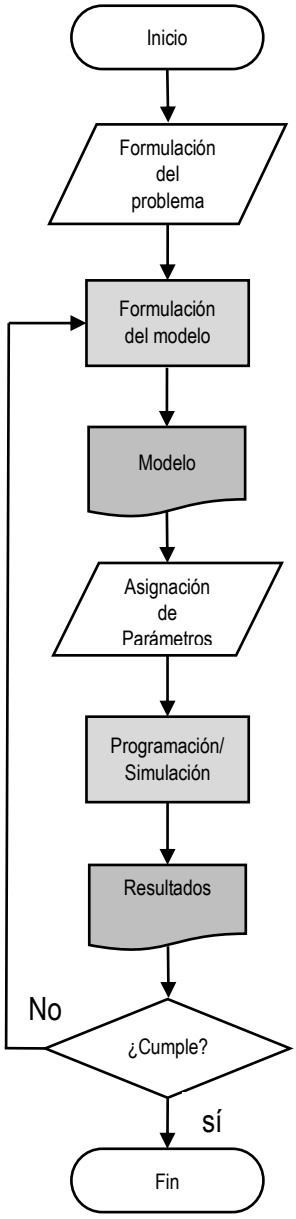


Ilustración 1 Diagrama de flujo de la metodología de programación de algoritmos interactivos para solución de problemas matemáticos [los autores].

La metodología usada para la solución de ejercicios de sistema de control se compone por:

1. Escribir el enunciado del ejercicio que se vaya a elaborar.
2. Dibujar el sistema caso de estudio con alguna herramienta CAD o imagen que lo represente.
3. Puede ir acompañado con el esquema en Simscape del sistema que se desea controlar. En algunos casos no es necesario (cuando los bloques no están adscritos a la biblioteca de Simscape).
4. Plantear la solución mediante código.
5. Desarrollar la solución en código de Matlab y generar las simulaciones requeridas para mostrar los resultados de manera estables.

La metodología se ejemplifica y explican otros apartados con el siguiente diagrama.

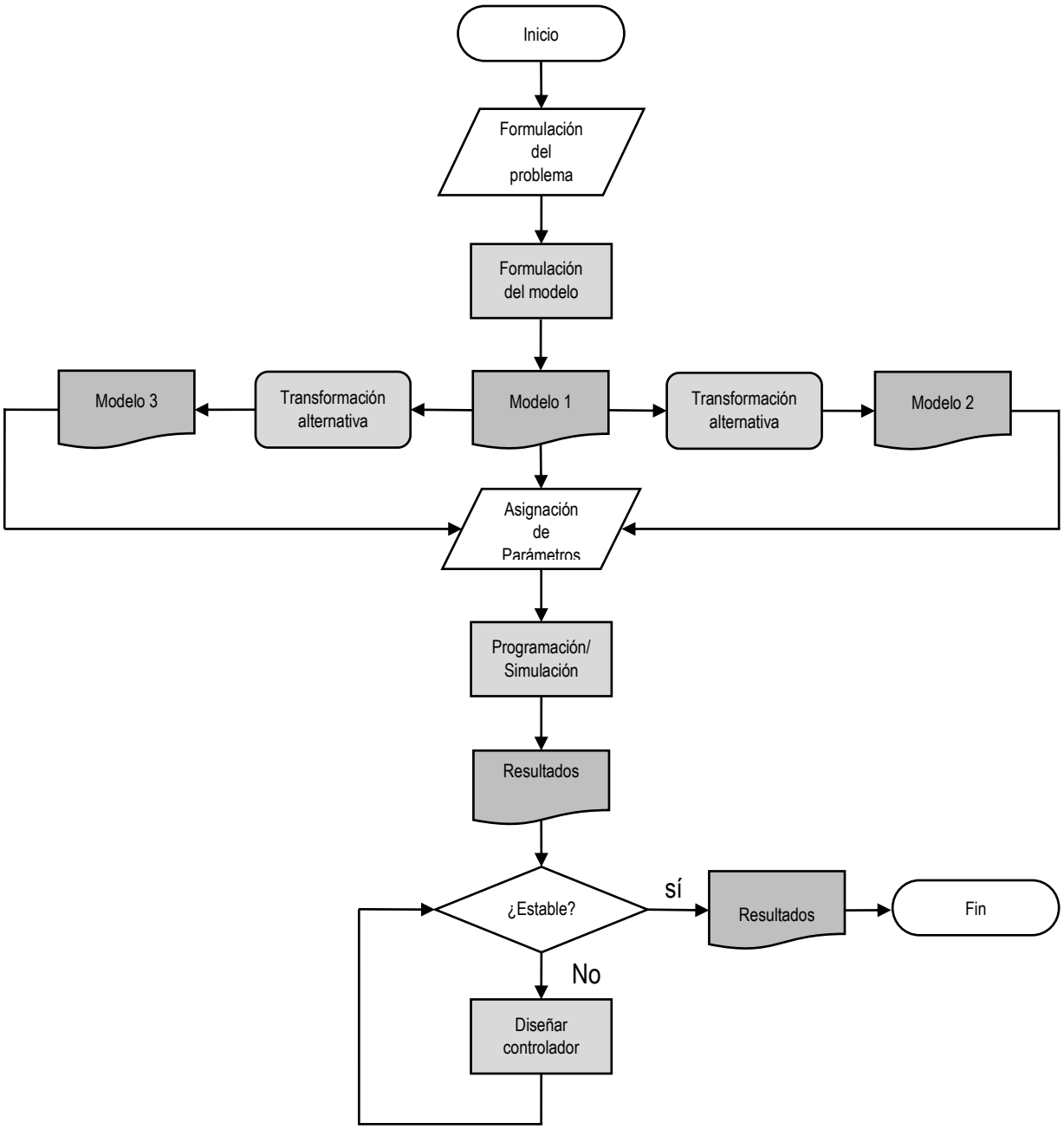


Ilustración 2 Diagrama de flujo de la metodología de programación de algoritmos interactivos para solución de problemas de modelado, simulación y control de sistemas físicos con Matlab® live editor

4. DISEÑO DE SISTEMAS DE CONTROL UTILIZANDO ALGORITMOS INTERACTIVOS CON LIVE EDITOR

A partir de esta sección se dará una explicación de lo que es el apartado de Live Editor en Matlab. Al comienzo del artículo investigativo, se hizo una corta pero clara información de lo que es el software de Matlab, sin embargo, este software ingenieril tiene dentro de sus múltiples funciones un apartado para realizar Scripts (se hará énfasis en los Live Scripts), que para hacer la correlación, este es uno de los múltiples avances que ha desarrollado el progreso de los algoritmos descritos en el apartado de estado del arte de este trabajo de grado. Así bien, la opción de generar Live Scripts dentro de la interfaz de Matlab permite la combinación de código, entrada y salida de texto, imágenes, tablas, hipervínculos, ecuaciones, entre muchas opciones más. Live Script o Live Editor (se encuentra de las dos maneras) dará las herramientas suficientes para que pueda combinar código y texto. La idea es que se desenvuelva mostrando el contenido que quiera dar a conocer de la manera más interactiva posible.

A continuación se desarrollan unos ejercicios para así ver y explicar la metodología propuesta y claramente los puntos del mismo. Esta metodología la puede evidenciar en lista y en diagrama en el capítulo 3.

4.1 CONTROL DE POSICIÓN DEL CHASIS DE UNA MOTOCICLETA

El desarrollo tecnológico cada día avanza más buscando contrarrestar de manera directa los obstáculos y problemas que puedan surgir a partir de un sistema en específico. Así bien, el sistema de suspensión en motos ha sido una de las puestas a punto de muchos ingenieros a través del tiempo y es que este sistema es uno de los más importantes en cualquier vehículo. Es el que nos da el confort de manejo y la seguridad del mismo, su función es también reducir de forma eficaz, las vibraciones y/o oscilaciones fuertes que se puedan generar motivadas por el estado mismo del terreno por donde se transite. A lo largo del tiempo, las tecnologías nos ha mostrado el uso de diferentes

mecanismos que por medio de fluidos que ayudan a la amortiguación del sistema, ya sea neumática, hidráulica, mecánica o combinadas.

En este caso, se realizará el estudio cinemático y matemático de la suspensión trasera, es decir, un medio (1/2) de la suspensión total de la motocicleta tipo enduro. El tipo de suspensión fue escogido debido a las buenas prestaciones que ésta posee en terrenos no pavimentados e irregulares. La moto a analizar es una Benelli bx 125cc que se muestra a continuación, se presenta como ejemplo ilustrativo.



Ilustración 3 Modelo de moto, imagen tomada y adaptada de https://www.canariasenmoto.com/motocicletas/benelli-bx-enduro.html

La simplificación del sistema de suspensión de una motocicleta (se ignora la resistencia de la fricción del caucho de la llanta, los efectos de la resistencia del viento, entre otros) da como resultado como un sistema mecánico traslacional el cual consta de 2 masas: la masa 1, considera el peso de los brazos de la suspensión, la rueda trasera y el sistema de freno trasero. La masa 2, hace referencia a la masa combinada de la motocicleta y el piloto.
Acoplados en serie bajo la masa A se encuentra el resorte (k1) y el amortiguador (B1), estos representan respectivamente el resorte y el amortiguador que está presente en un sistema convencional de suspensión de una motocicleta.

Analizando el comportamiento del neumático, se observa un comportamiento similar al de una suspensión de corto recorrido, ya que, gracias a las propiedades elásticas del caucho, este retoma su forma original luego de sufrir una deformación elástica. Teniendo en cuenta lo anterior, se decide simular este comportamiento remplazando el caucho por un sistema compuesto por un resorte (k2) y un amortiguador (B2).

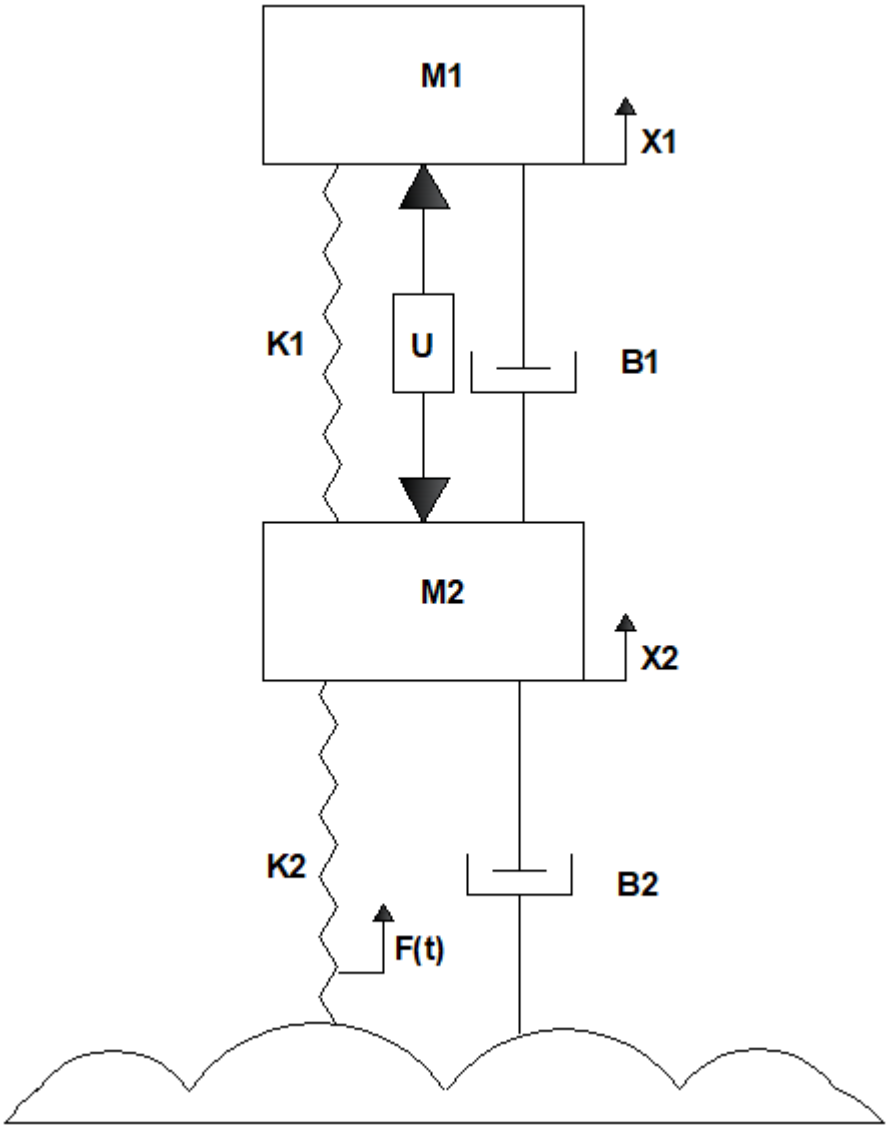


Ilustración 4 DCL para suspensión de motocicleta

4.1.1 ECUACIONES DE MOVIMIENTO DEL SISTEMA

Analizando el diagrama de cuerpo libre presentado anteriormente, se obtienen las siguientes ecuaciones:

Sistema 1

$$M_1X_1'' = -B_1\left(x_1' - x_2'\right) - K_1(x_1 - x_2) + U \qquad (1)$$

Sistema 2

$$M_2X_2'' = B_1\left(x_1' - x_2'\right) + K_1(x_1 - x_2) + B_2\left(F'(t) - X_2'\right) + K2(F(t) - X_2) - U \qquad (2)$$

4.1.2 MODELOS DE FUNCIÓN TRANSFERENCIA

El análisis se va a realizar suponiendo que todas las condiciones iniciales son cero, esto para simular que el sistema de suspensión de la motocicleta se encuentra en reposo y será afectado por un sobresalto o imperfección de la carretera que transita. Las ecuaciones de movimiento presentadas anteriormente también se pueden expresar en forma de función transferencia tomando claramente la transformada de Laplace. El cambio específico de cada una de las ecuaciones a función transferencia como G1(s) y G2(s) se muestra a continuación; donde cada función tiene una salida de X1 y X2 y entradas de U y F(t), respectivamente:

$$\left(M_1s^2 + B_1s + K_1\right)X_1(s) - (B_1s + K_1)X_2(s) = U(s) \qquad (3)$$

$$-(B_1s + K_1)X_1(s) + \left(M_2s^2 + (B_1 + B_2)s + (K_1 + K_2)\right)X_2(s) = (B_2s + K_2)W(s) - U(s) \qquad (4)$$

Realizando una matriz para poder visualizar mejor los términos y así seguir con el procedimiento, se obtiene:

$$\begin{bmatrix} \left(M_1s^2 + B_1s + K_1\right) & -(B_1s + K_1) \\ -(B_1s + K_1) & \left(M_2s^2 + (B_1 + B_2)s + (K_1 + K_2)\right) \end{bmatrix} \begin{bmatrix} X_1(s) \\ X_2(s) \end{bmatrix} = \begin{bmatrix} U(s) \\ (B_2s + K_2)W(s) - U(s) \end{bmatrix} \qquad (5)$$

$$A = \begin{bmatrix} \left(M_1s^2 + B_1s + K_1\right) & -(B_1s + K_1) \\ -(B_1s + K_1) & \left(M_2s^2 + (B_1 + B_2)s + (K_1 + K_2)\right) \end{bmatrix} \qquad (6)$$

$$\triangle = det\begin{bmatrix} \left(M_1s^2 + B_1s + K_1\right) & -(B_1s + K_1) \\ -(B_1s + K_1) & \left(M_2s^2 + (B_1 + B_2)s + (K_1 + K_2)\right) \end{bmatrix} \qquad (7)$$

Donde:

$$\triangle = \left(M_1 s^2 + B_1 s + K_1\right) \cdot \left(M_2 s^2 + (B_1 + B_2)s + (K_1 + K_2)\right) - (B_1 s + K_1) \cdot (B_1 s + K_1) \quad (8)$$

Cuando se encuentra la inversa de la matriz que se definió como A y posterior a ello multiplicándolas por las entradas U(s) y W(s), se obtiene:

$$\begin{bmatrix} X_1(s) \\ X_2(s) \end{bmatrix} = \frac{1}{\triangle} \begin{bmatrix} \left(M_2 s^2 + B_2 s + K2\right) & \left(B_1 B_2 s^2 + (B_1 K_2 + B_2 K_1)s + K_1 K_2\right) \\ -M_1 s^2 & \left(M_1 B_2 s^3 + (M_1 K_2 + B_1 B_2)s^2 + (B_1 K_2 + B_2 K_1)s + K_1 K_2\right) \end{bmatrix} \begin{bmatrix} U(s) \\ W(s) \end{bmatrix} \quad (9)$$

Con esto entonces, se puede definir y considerar que cuando se quiera solo la entrada de control U(s) se establece W(s)=0, de este modo podemos obtener la función transferencia G1(s) como:

$$G_1(s) = \frac{X_1(s) - X_2(s)}{U(s)} = \frac{(M_1 + M_2)s^2 + B_2 s + K_2}{\triangle} \quad (10)$$

Del mismo modo para considerar solo la entrada de la perturbación que es W(s), se establece a U(s)=0 y se obtiene la función transferencia de G2(s) como:

$$G_2(s) = \frac{X_1(s) - X_2(s)}{W(s)} = \frac{-M_1 B_2 s^3 - M_1 K_2 s^2}{\triangle} \quad (11)$$

4.1.3 PARÁMETROS DEL SISTEMA E INTRODUCCIÓN DE FÓRMULAS A ENTORNO MATLAB

Con los parámetros definidos para cada una de las masas, los coeficientes de amortiguación y las constantes de elasticidad de los resortes. Se pueden generar los modelos de funciones de transferencia en MATLAB con los siguientes comandos; Los datos escogidos para este sistema son en un entorno general, sin embargo, al estar en código se pueden cambiar dependiendo la necesidad de análisis que se necesite.

```
M1=310; %MASA MOTOCICLETA%
M2=80; %MASA PILOTO%
K1=18000; %RESORTE%
K2=38000; %RESORTE%
B1=30; %AMORTIGUADOR%
B2=1320; %AMORTIGUADOR%
s=tf('s'); %CONSTRUYE LA FUNCIÓN TRANSFERENCIA O CONVIERTE A FUNCIÓN
TRANSFERENCIA%
```

```
 G1=((M1+M2)*s^2+B2*s+K2)/((M1*s^2+B1*s+K1)*(M2*s^2+(B1+B2)*s+(K1+K2))-
(B1*s+K1)*(B1*s+K1)); %DEFINICIÓN FUNCIÓN TRANSFERENCIA PARA SISTEMA 1%
 G2=(-M1*B2*s^3-
M1*K2*s^2)/((M1*s^2+B1*s+K1)*(M2*s^2+(B1+B2)*s+(K1+K2))-
(B1*s+K1)*(B1*s+K1)); %DEFINICIÓN FUNCIÓN TRANSFERENCIA PARA SISTEMA 2%
```

4.1.4 REQUERIMIENTOS DE DISEÑO

Un buen y óptimo diseño de una suspensión tipo enduro y en cualquier suspensión en general es que entregue una buena sensación de agarre al terreno donde se transite y dar comodidad cuando se experimente cualquier perturbación en la carretera (terrenos irregulares o montañosos, trochas, baches entre otros), lo ideal es que el piloto no esté expuesto a oscilaciones de larga duración, pues afectaría el rendimiento, pero tampoco se necesita erradicarlas por completo pues el impacto sería más traumático. En conclusión, se deben tener las oscilaciones pero se deben controlar rápidamente.
La distancia entre X1-F(t) es muy difícil de medir y la deformación de la llanta X2-F(t) es insignificante. Se analizará la distancia X1-X2, en lugar de X1-F(t). Tenga en cuenta que esto es una aproximación.
La perturbación del camino F(t) en este problema será simulada por una entrada tipo Step para imitar cuando la motocicleta toma un sobresalto en la vía. Se realizará un controlador de retroalimentación para que esta salida de X1-X2 tenga un sobre impulso menor o igual al 10% y un tiempo menor a 5 segundos, es decir, cuando la moto tenga una perturbación de aproximadamente 30cm, el piloto y la moto en general oscile dentro de un rango de +/- 15mm y vuelva a establecerse en aprox. 5 segundos.

4.1.5 RESPUESTA EN LAZO ABIERTO

Se puede visualizar el comportamiento que tiene el sistema (sin controlador y sin retroalimentación) por medio de una gráfica Step, el programa realiza el análisis. Para ello es necesario haber tenido las variables con sus respectivos resultados en código para la visualización.

La respuesta al escalón unitario de la función de transferencia es el siguiente:

```
step(G1) %COMANDO PARA VISUALIZAR EL COMPORTAMIENTO DE G1%
```

```
 title('RESPUESTA AL ESCALÓN UNITARIO DE LA FUNCIÓN TRANSFERENCIA G1')
%CONFIGURA EL TITULO DE LA GRAFICA%
```

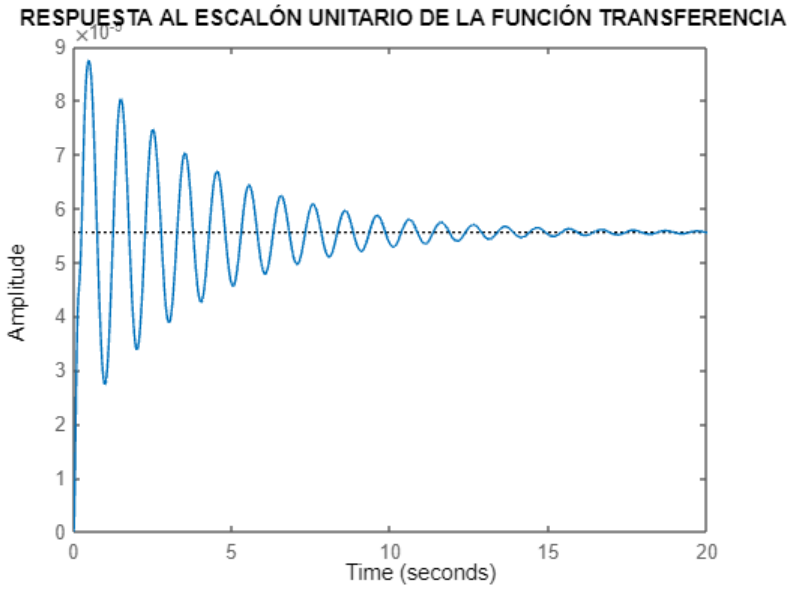


Ilustración 5 Respuesta al escalón unitario de G1

A partir del gráfico en respuesta de lazo abierto (lazo abierto u open loop, quiere decir que la salida censada del proceso no es comparada con la señal de referencia, diferente a un sistema de lazo cerrado que toma la señal de salida del proceso y la compara con la señal de referencia y así conocer en todo momento la evolución de la variable) para una fuerza accionada por el escalón unitario, se evidencia que el sistema está subamortiguado, el piloto sentirá la oscilación muy pequeña pero tarda un tiempo exageradamente largo en estabilizarse, aproximadamente 20Seg. Ingresando los datos para ver la respuesta de una entrada de una perturbación de igual manera en escalón unitario, F(s) con una magnitud de 0,3m se realiza lo siguiente:

```
 G2; %LLAMAR VARIABLE DEFINIDA COMO G2%
 step(0.3*G2) %COMANDO PARA VISUALIZAR EL COMPORTAMIENTO DE G2 CON UN
PASO DE 0.3%
 title('RESPUESTA AL ESCALÓN UNITARIO DE LA FUNCIÓN TRANSFERENCIA G2')
%CONFIGURA EL TITULO DE LA GRAFICA%
```

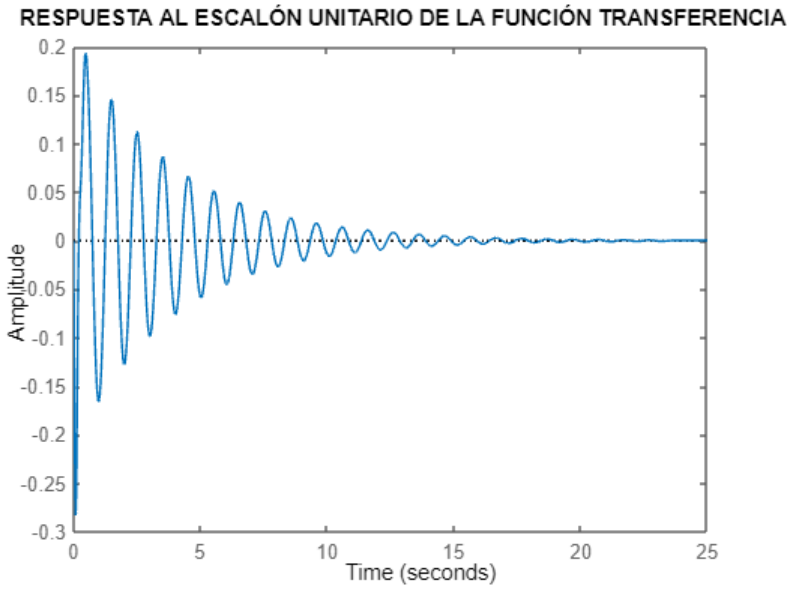


Ilustración 6 Respuesta al escalón unitario de G2

En la gráfica se puede evidenciar que ante una entrada escalón que simula la irregularidad del terreno a 30cm y se ve afectada de manera directa y muestra la perturbación. Posterior a ello, la señal se va aplacando y normalizando hasta que el sistema queda en reposo, pero es un tiempo muy largo (~25 segundos). El piloto no se sentirá cómodo con tal oscilación debido al gran exceso de tiempo.

La solución entonces es agregar un controlador al sistema para mejorar el rendimiento. El esquema del sistema en lazo cerrado es el siguiente:

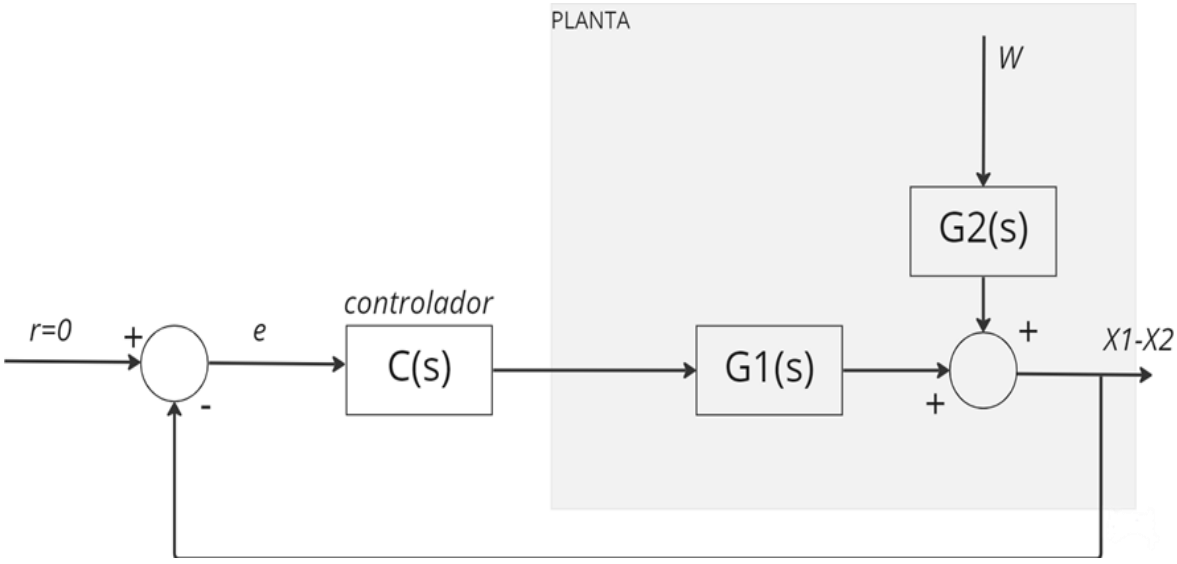


Ilustración 7 Esquema del sistema en lazo cerrado

Del problema principal, se recuerdan las ecuaciones de movimiento en función transferencia 10 y 11

El esquema del sistema es el siguiente donde F(s) G1(s)=G2(s)

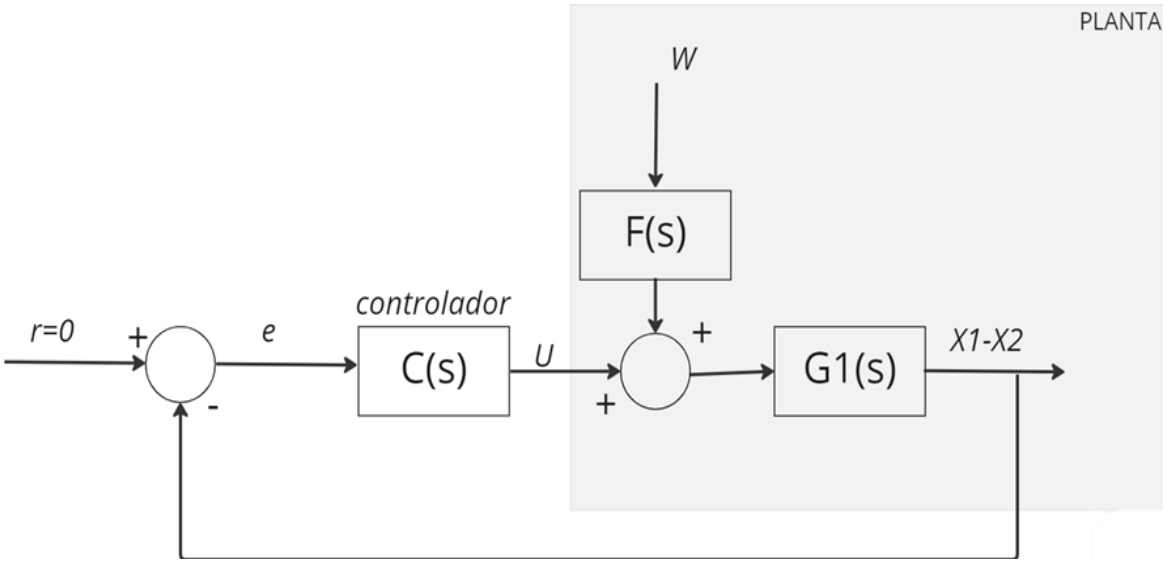


Ilustración 8 Esquema del sistema donde interactúa F(s) y G1(s)=G2(s)

El diseño del controlador de retroalimentación para que cuando la perturbación de la carretera F(t) sea simulada con la entrada de escalón unitario, la salida en X1-X2 tenga un tiempo de estabilización de aproximadamente 5 segundos

y un sobre impulso de menor o igual al 10%. Las mismas condiciones de las que se hablaron anteriormente. Así entonces, teniendo en cuenta las mismas variables que se han venido planteando se añaden las siguientes:

```
 nump=[(M1+M2) B2 K2]; %SE DEFINE NUMERADOR CON LAS VARIABLES%
 denp=[(M1*M2) (M1*(B1+B2))+(M2*B1) (M1*(K1+K2))+(M2*K1)+(B1*B2)
(B1*K2)+ (B2*K1) K1*K2]; %SE DEFINE DENOMINADOR CON LAS VARIABLES%
 G1=tf(nump,denp); %SE DEFINE FUNCIÓN TRANSFERENCIA CON NUMERADOR Y
DENOMINADOR PARA SISTEMA 1%
 num1=[-(M1*B2) -(M1*K2) 0 0]; %SE DEFINE NUMERADOR CON LAS VARIABLES%
 den1=[(M1*M2) (M1*(B1+B2))+(M2*B1) (M1*(K1+K2))+(M2*K1)+(B1*B2)
(B1*K2)+(B2*K1) K1*K2]; %SE DEFINE DENOMINADOR CON LAS VARIABLES%
 G2=tf(num1,den1); %SE DEFINE FUNCIÓN TRANSFERENCIA CON NUMERADOR Y
DENOMINADOR PARA SISTEMA 2%

 numf=num1; %CAMBIO DE NOMBRE VARIABLES%
 denf=nump; %CAMBIO DE NOMBRE VARIABLES%
 F=tf(numf,denf); %SE DEFINE FUNCIÓN TRANSFERENCIA%
```

4.1.6 IMPLEMENTACIÓN DE UN CONTROLADOR PID

Recordemos que la función para un controlador PID está dada por la función transferencia

$$C(s) = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + k_p s + K_i}{s} \qquad (12)$$

Donde Kp es la ganancia proporcional, Ki es la ganancia integral y Kd es la ganancia derivativa. En principio se supondrán los valores de las ganancias se definen en código como:

```
 Kd=2208; %GANANCIA DERIVATIVA%
 Kp=8346; %GANANCIA PROPORCIONAL%
 Ki=6240; %GANANCIA INTEGRAL%
 C=pid(Kp,Ki,Kd); %SE DEFINE CONTROL PID CON LAS GANANCIAS ANTERIORES%
```

Se simulará la respuesta del sistema, la distancia entre X1-X2, a una perturbación tipo Step en la carretera. Del esquema anterior se halla la función transferencia de la perturbación F(t) a la salida X1-X2.

```
 sys_cl=F*feedback(G1,C); %SE DEFINE FUNCIÓN TRANSFERENCIA Y HACE
RETROALIMENTACIÓN%
```

4.1.7 RESPUESTA EN LAZO CERRADO

Ya fue creada la función transferencia en lazo cerrado como código en MATLAB que representará la planta, la perturbación y el controlador. Usando un paso de 0,3m como una perturbación propia, para simular esto se ve representado en código de la siguiente manera:

```
 t=0:0.05:5; %VARIABLE QUE DEFINE TIEMPO DE SIMULACIÓN%
 step(0.3*sys_cl,t) %COMANDO PARA VISUALIZAR EL COMPORTAMIENTO DE F.
TRANSFERENCIA CON LA RETROALIMENTACIÓN%
 title('Respuesta a un Step de 0.3m bajo control PID') %CONFIGURA EL
TITULO DE LA GRAFICA%
```

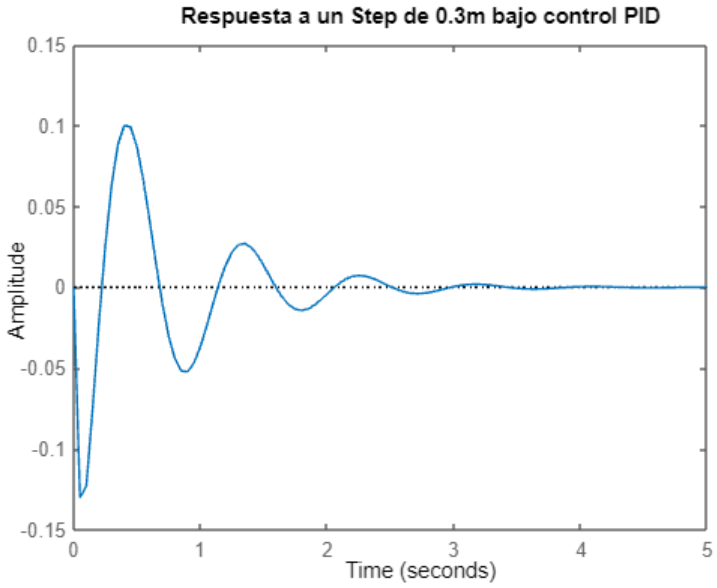


Ilustración 9 Respuesta al escalón unitario bajo controlador PID

Ahora vemos que el porcentaje de sobre impulso y el tiempo de estabilización cumplen con los requisitos que se definieron al principio. El sobre impulso menor o igual al 10% de la amplitud de la entrada y el tiempo de estabilización aproximadamente a los 4 segundos que es menor a los 5 que se requirió en el sistema. Teniendo en cuenta el tipo de motocicleta con el que se está trabajando, los resultados son óptimos.

Para este sistema resulta muy conveniente que el método de diseño PID lo controla adecuadamente, no obstante, se puede mejorar de manera óptima cambiando las ganancias Kp, Ki y Kd y obtener un mejor resultado de sobre impulso y tiempo de estabilización.

4.1.8 ANÁLISIS DEL SISTEMA EN ENTORNO DE SIMULINK

El sistema se modelará teniendo en cuenta las fuerzas que interactúan sobre ambas masas (cuerpo y suspensión) e integrando las aceleraciones de cada una de las masas dos veces para así obtener las velocidades y las posiciones, de este modo se realiza el diagrama de bloques del sistema quedando de esta manera.

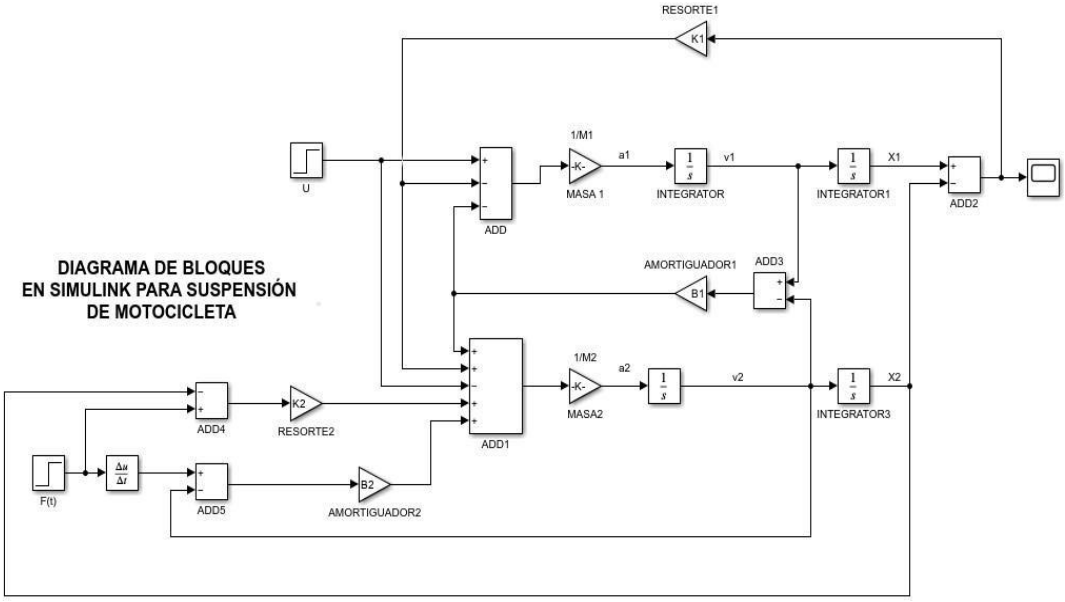


Ilustración 10 Diagrama de bloques en Simulink

De este modo se encuentra representado el sistema en cuestión en forma de diagrama de bloques, con las variables de masas, constantes de resortes y amortiguadores dados desde el principio, con ello podemos ver su comportamiento por medio de la siguiente gráfica:

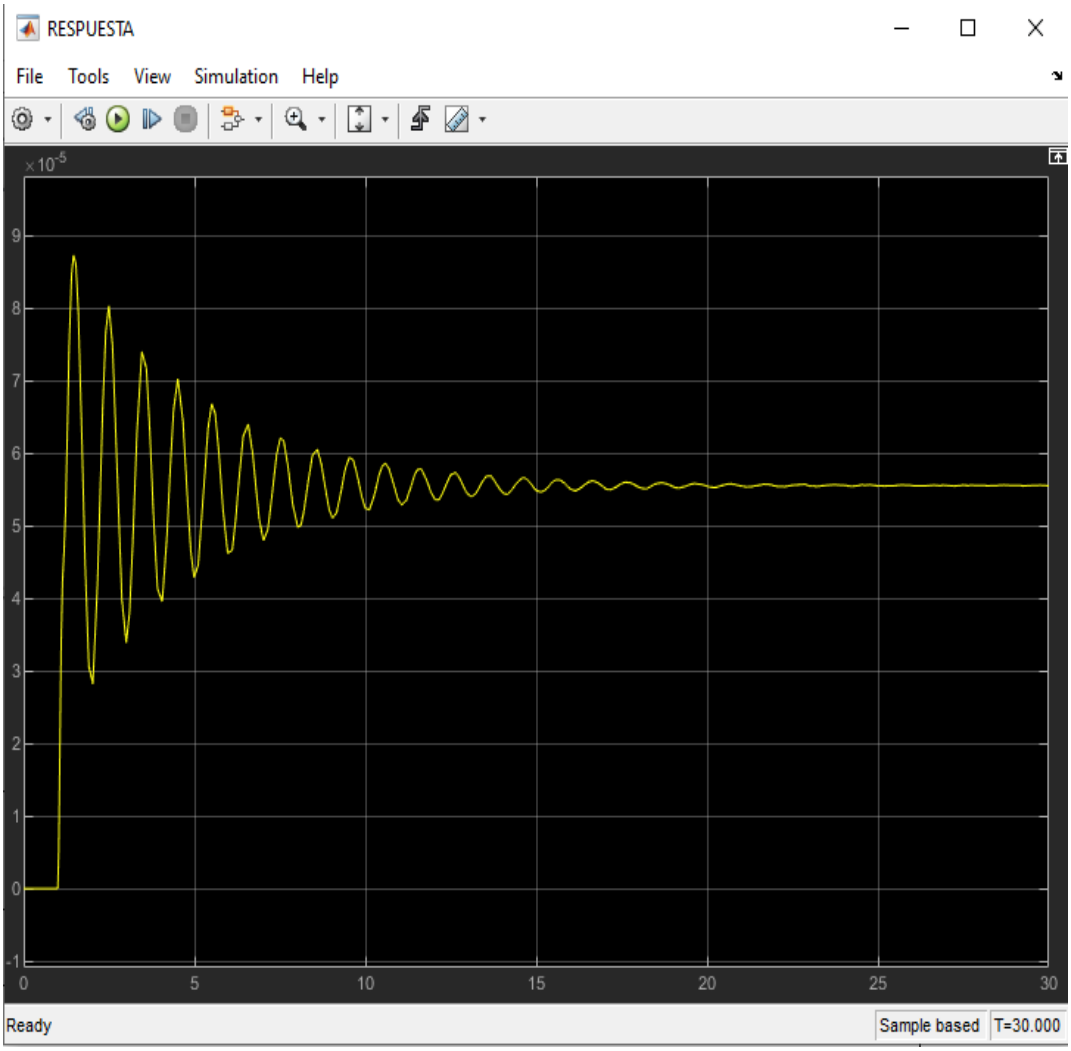


Ilustración 11 Respuesta de diagrama de bloques de Simulink

Teniendo en cuenta el propósito del ejercicio, lo conveniente es mejorar la respuesta de la suspensión por medio del controlador de la entrada U. Así entonces se empieza el procedimiento para la extracción de un modelo de Simulink a entorno propio MATLAB.

4.1.9 EXTRACCIÓN DEL MODELO SIMULINK A MATLAB

Esta extracción es posible desde el modelo de Simulink usando solamente dos bloques, en este caso solo se extraerá el modelo de entrada U a la salida X1-X2. El modelo propuesto es el siguiente:

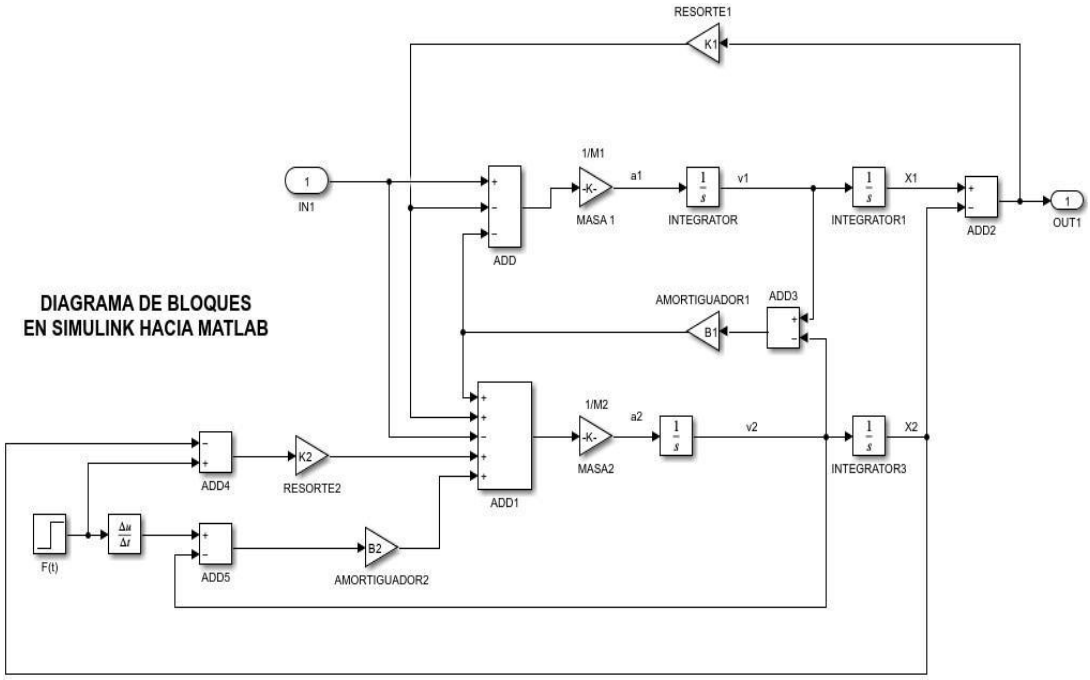


Ilustración 12 Diagrama de bloques en Simulink hacia Matlab

Fíjese en el cambio de bloque "Step U" por "In1" y "Scope" por "Out1". Guarde el documento con el nombre de su preferencia pero con extensión ".slx", ingresando los siguientes comandos para visualizar los resultados.

```
 [A,B,C,D]=linmod('SMSM'); % EXTRAE EL MODELO DE ESPACIO DE ESTADOS
ALREDEDOR DEL PUNTO DE OPERACIÓN%
 [num,den]=ss2tf(A,B,C,D); %COMANDO PARA PASAR DE ESTADO DE ESTADOS A
FUNCIÓN TRANSFERENCIA%
 step(num,den )%COMANDO PARA VISUALIZAR EL COMPORTAMIENTO DEL NUMERADOR
Y DENOMINADOR DEFINIDOS%
 title('RESPUESTA A LA ENTRADA SIMULADA') %CONFIGURA EL TITULO DE LA
GRAFICA%
```

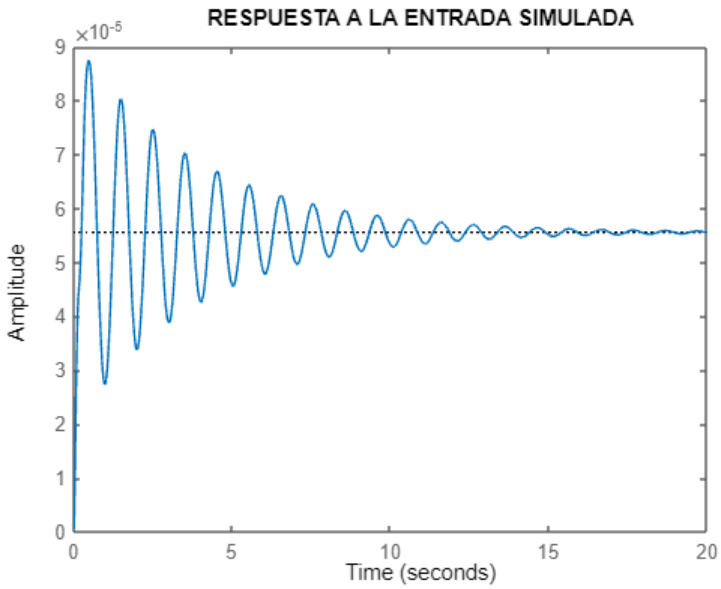


Ilustración 13 Respuesta de la entrada simulada en entorno de Matlab

4.1.10 IMPLEMENTACIÓN DE CONTROL EN ESPACIO DE SIMULINK

A partir de esta sección se realizará el diseño para controlar el sistema desde el espacio de Simulink.

El controlador usó la siguiente matriz de ganancia de retroalimentación.

```
K=[0 4.3E6 3E8 0 9E6]; %SE DEFINE UNA VARIABLE K%
```

Se define este valor de K, para realizar el diagrama de bloques y controlarlo, posteriormente, así entonces empezamos creando unas salidas convenientes a "DIAGRAMA DE BLOQUES EN SIMULINK HACIA MATLAB", tal que quede de la siguiente manera:

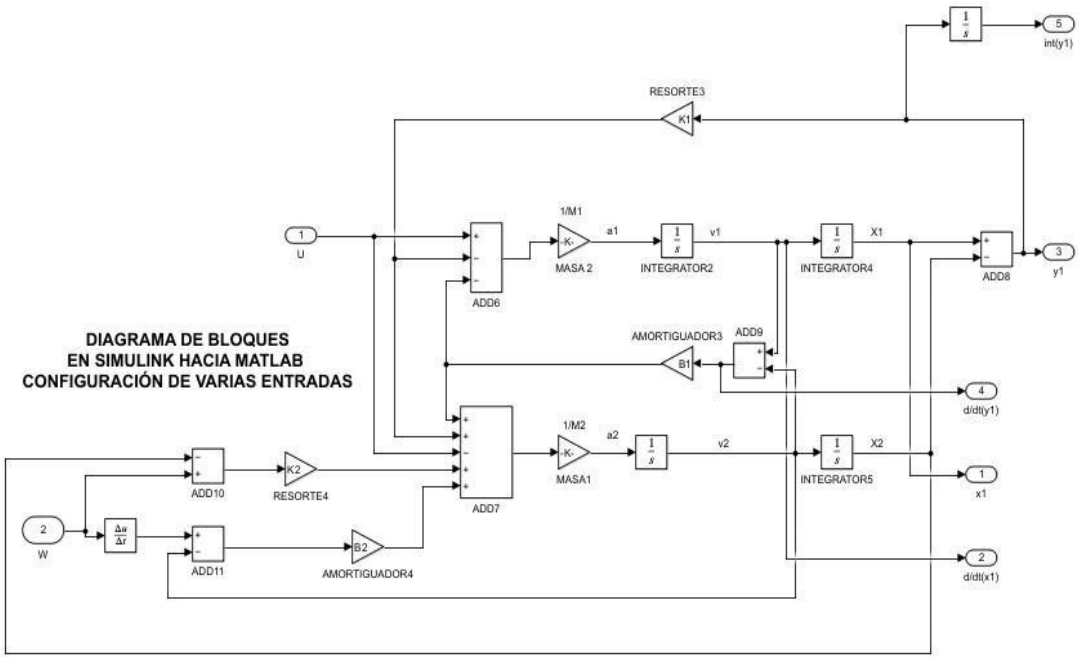


Ilustración 14 Diagrama de bloques en Simulink hacia Matlab de varias salidas

Creando las entradas correspondientes, se construye un controlador con retroalimentación para ello entonces, ya fuera del sistema se agrega un bloque MUX de 5 entradas para así poder visualizar de manera correcta el resto del ejercicio, se crean las entradas tipo Step para simular los baches de la carretera y la ganancia K multiplicado por la entrada U. El diagrama de bloques final queda de la siguiente manera:

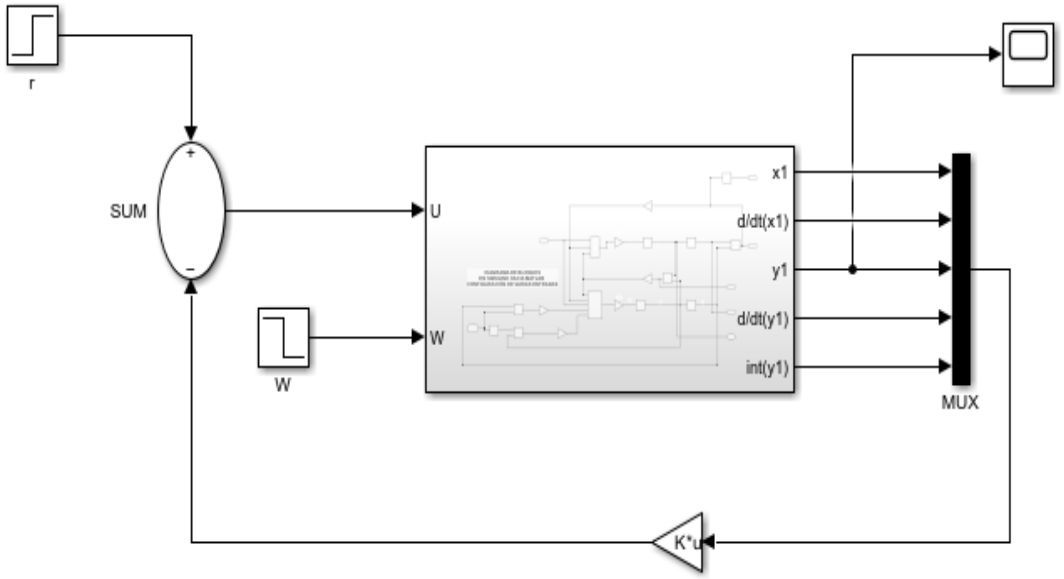


Ilustración 15 Diagrama de bloques con retroalimentación en Simulink

Es importante conocer si el sistema funcionó y si cumple con los requisitos planteados desde el comienzo, es por ello, que agregamos el Scope y poder ver el comportamiento, Se evidencia lo siguiente:

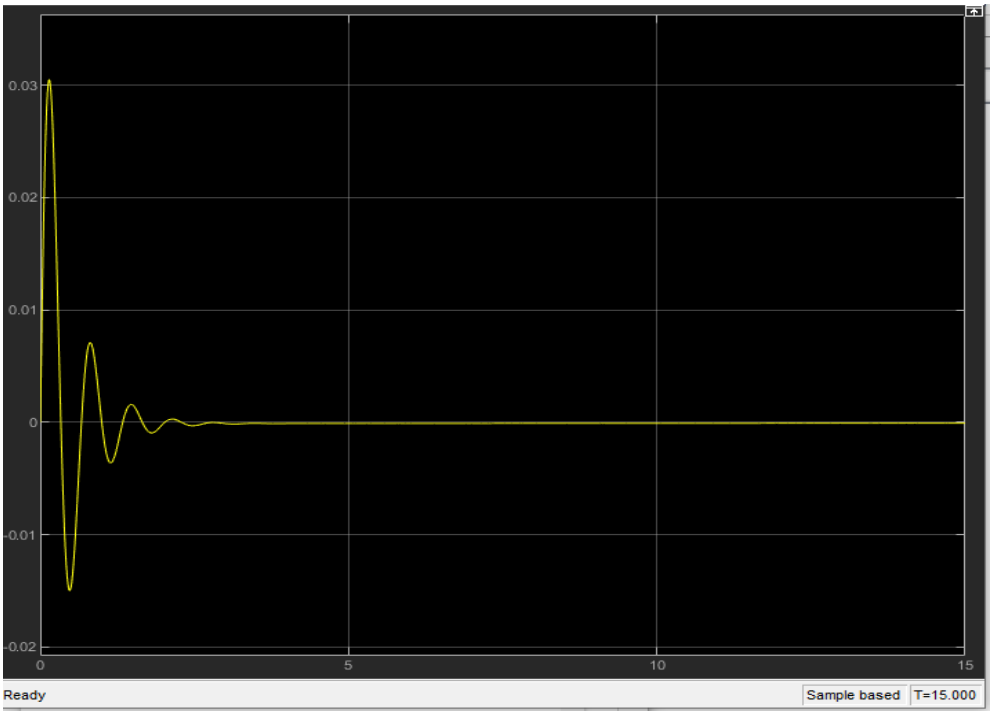


Ilustración 16 Respuesta al diagrama de bloques con retroalimentación

Finalmente podemos evidenciar que con el controlador interpuesto el sistema se controla en menos de los 5 segundos que es el requisito principal del sistema, en este caso el asentamiento de las oscilaciones está entre los 3 y 4.5 segundos.

4.1.11 MODELO DEL SISTEMA EN ENTORNO SIMSCAPE

En esta sección se explicará y se hará la representación del sistema de suspensión pero en el entorno de Simscape de Matlab. Simscape permite crear de manera rápida modelos de cualquier sistema físico dentro del apartado de Simulink, se pueden crear conexiones físicas que se integran con los diagramas de bloques propios y normales del entorno de Simulink.

Se creará el modelo físico de la suspensión, para ello, como primera medida es ejecutar el entorno de Simulink y allí abrir en el apartado de Simscape la ventana "Generic Simscape" a partir de allí empezaremos agregando los bloques y configurando de tal manera que se construya el sistema. Los nombres de los bloques y sus parámetros se explicarán de abajo hacia arriba a continuación.

- PULSE GENERATOR: Bloque que nos ayuda a simular la entrada de un bache o desnivel; amplitud=1, periodo=100, ancho de pulso=0.1.
- IDEAL TRANSLATIONAL VELOCITY SOURCE: Representa una fuente ideal de velocidad que genera un diferencial de velocidad, mantiene la velocidad especificada independiente de la fuerza ejercida sobre el sistema.
- SOLVER CONFIGURATION: Define las configuraciones usadas para la simulación. Parámetros por defecto.
- IDEAL TRANSLATIONAL MOTION SENSOR: Con este bloque es posible medir la aceleración, velocidad y/o desplazamiento el sensor es ideal ya que no tiene en cuenta la inercia, fricción, retrasos, consumo de energía etc.
- TRANSLATIONAL SPRING: Este bloque representa un resorte mecánico lineal ideal. Parámetros= valor que representa el resorte de la llanta, en este caso K1 y K2.
- TRANSLATIONAL DAMPER: Este bloque representa un amortiguador hidráulico translacional mecánico ideal. Parámetros= valor que representa el resorte de la llanta, en este caso B1 y B2.
- MASS: El bloque representa una masa traslacional mecánica ideal. Parámetros= valor que representa el resorte de la llanta, en este caso M1 y M2.
- SIMULINK PS CONVERTER: Sensor para visualizar de entorno simulink a simscape
- SCOPE: Visualizador grafico
- PS SIMULINK CONVERTER: Sensor para visualizar de entorno simscape a simulink

El modelo es representado de la siguiente manera:

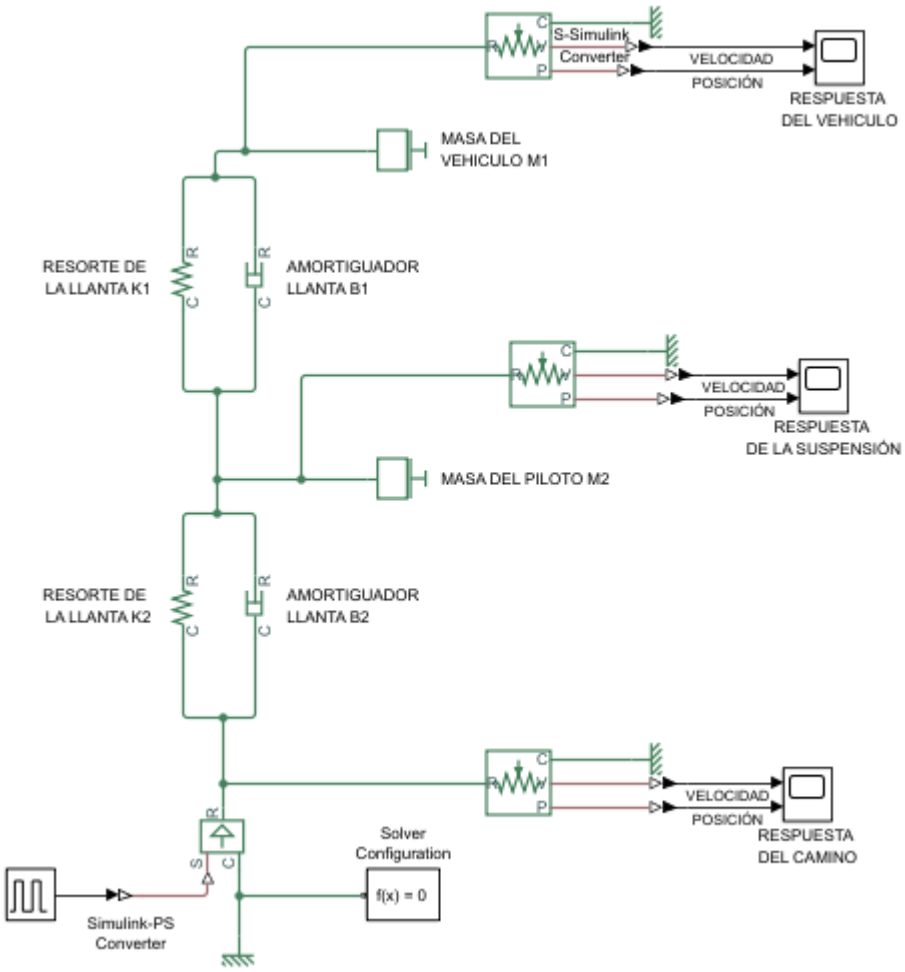


Ilustración 17 Modelo del sistema en entorno Simscape

Al simular podemos evidenciar las respuestas que se pueden generar, una de la más importante es la respuesta del vehículo obteniendo lo siguiente:

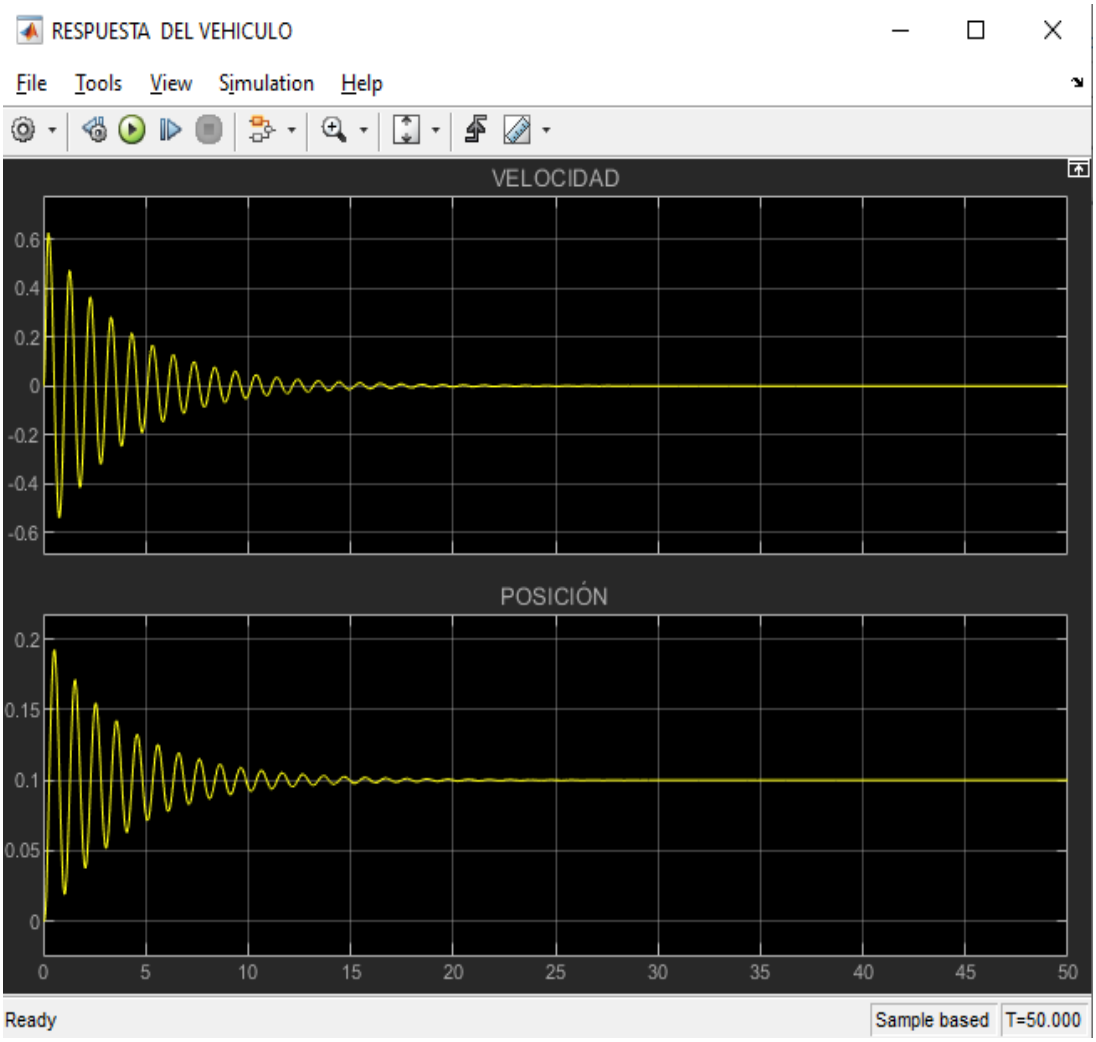


Ilustración 18 Respuesta del modelo de Simscape

Se evidencia que el vehículo oscila en un tiempo bastante largo, a partir de ahora, se va a controlar añadiendo los siguientes bloques.

4.1.12 DEFINIR LA ENTRADA DE CONTROL Y MEDICIÓN DE ESTADOS ADICIONALES

Se agrega una fuente de fuerza ideal para amortiguar rápidamente las oscilaciones del bache. Para ello agregamos un bloque Step con un tiempo de 0 y la amplitud de "Pulse Generator" de igual forma a 0. El diagrama de bloques queda de la siguiente manera:

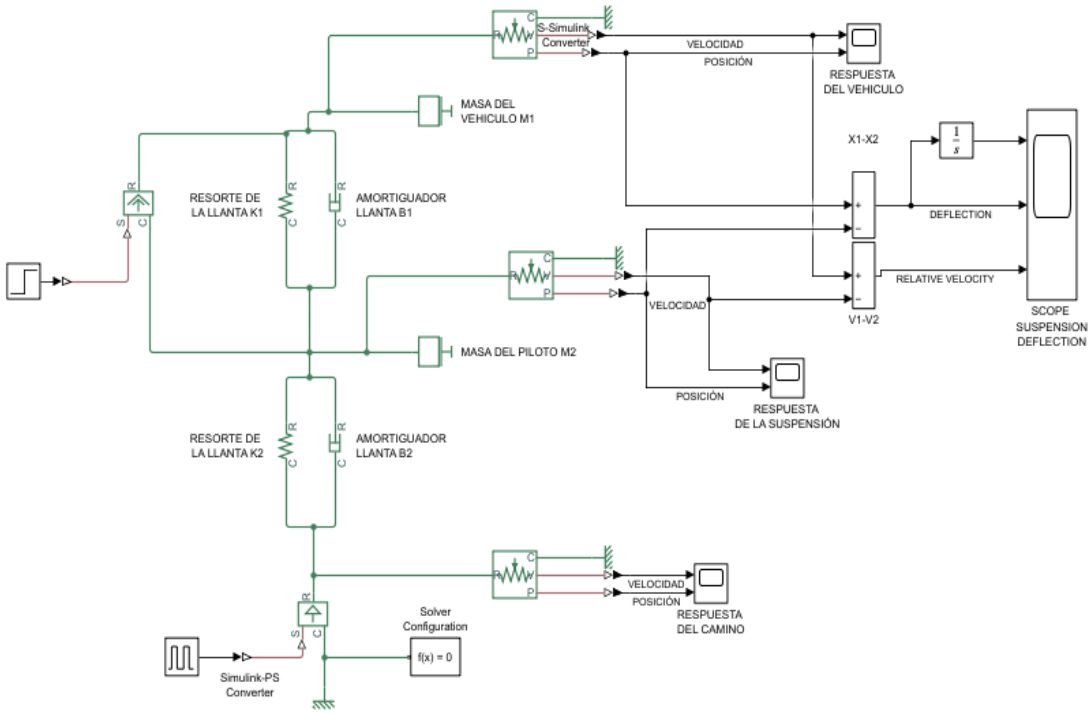


Ilustración 19 Diagrama de bloques en entorno de Simscape

Al ejecutar la simulación, la respuesta de la suspensión resultante en el osciloscopio de deflexión de suspensión se muestra de la siguiente forma:

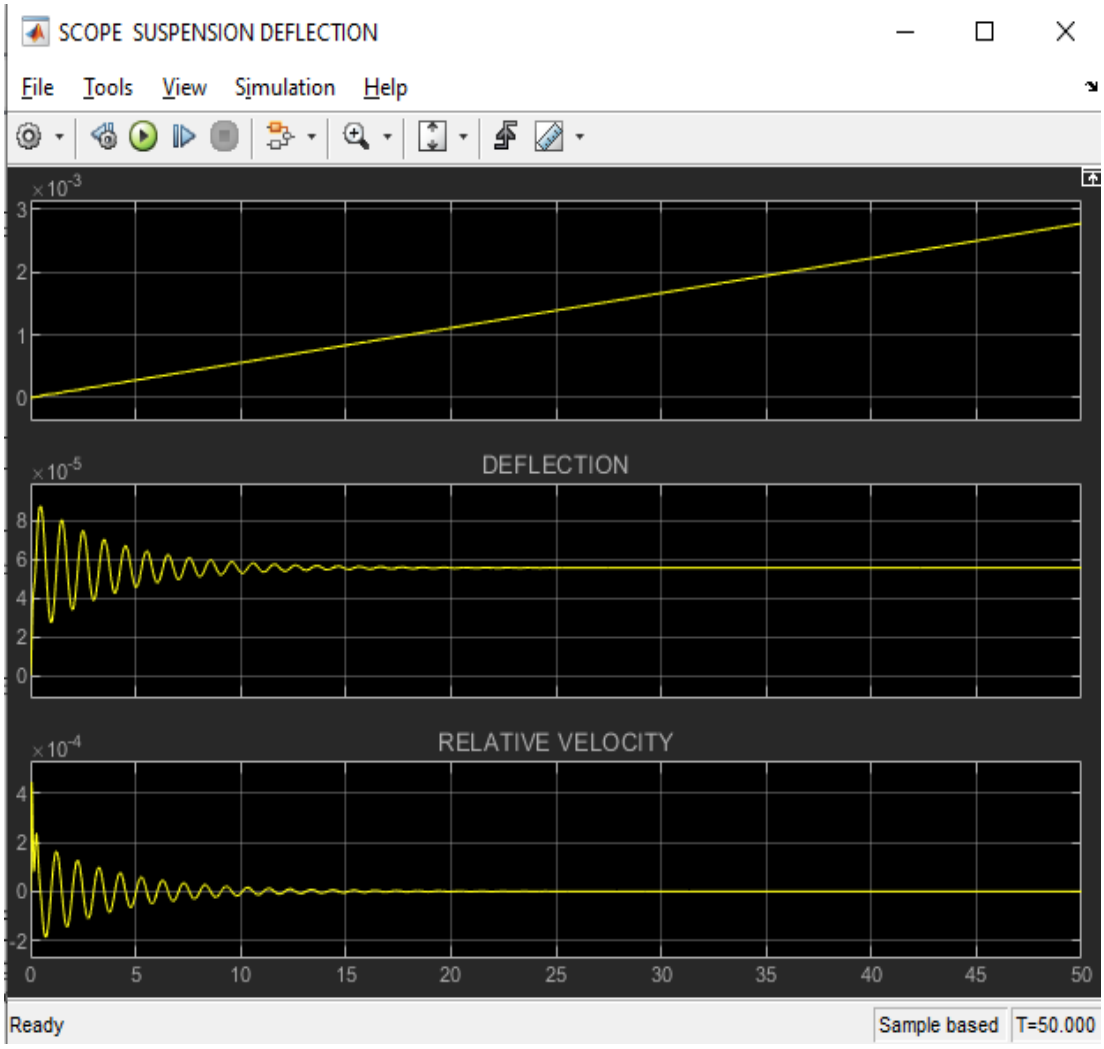


Ilustración 20 Respuesta al diagrama de bloque en entorno Simscape

4.1.13 CONTROL EN CIRCUITO CERRADO

Para poder controlar el sistema desde el espacio de simscape, se tiene que cambiar los bloques de entradas "Step" y "Pulse Generator" por entradas normales "In" y los bloques de salida del "Scope" por bloques tipo "Out". El diagrama de bloques entonces queda configurado así:

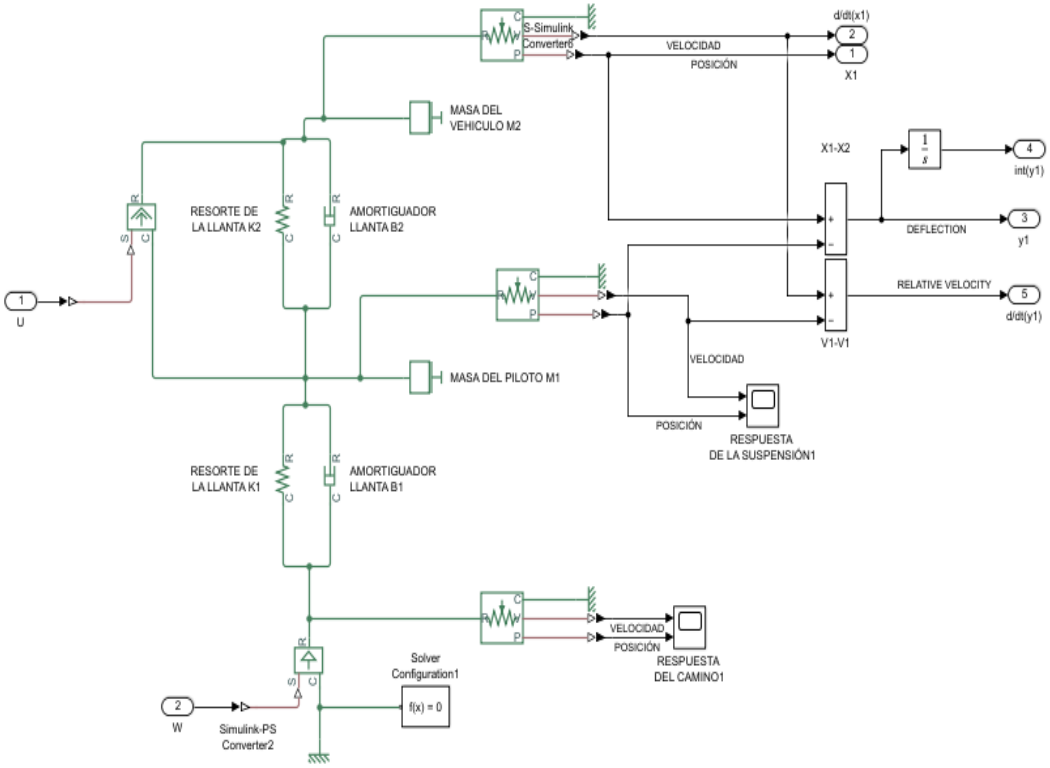


Ilustración 21 Diagrama de bloque en Simscape con varias salidas

Después, se crea un subsistema del diagrama, se eliminan las entradas y salidas que quedan, para posterior añadir un bloque MUX entradas Step y Pulse Generator como se ve a continuación:

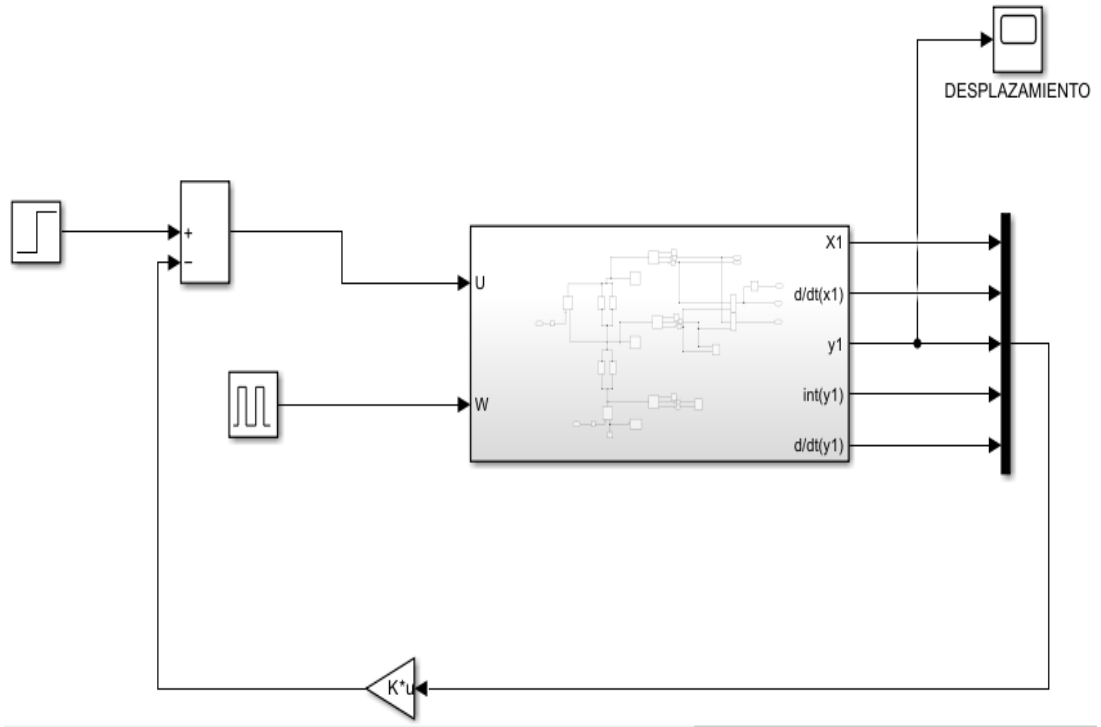


Ilustración 22 Diagrama de bloques con retroalimentación en Simscape

La configuración del bloque "Step" en el "Final Value=0 y en Pulse Generator el valor de la amplitud=1, cuando se ejecuta la simulación se ve el sistema controlado con los requerimientos deseados.

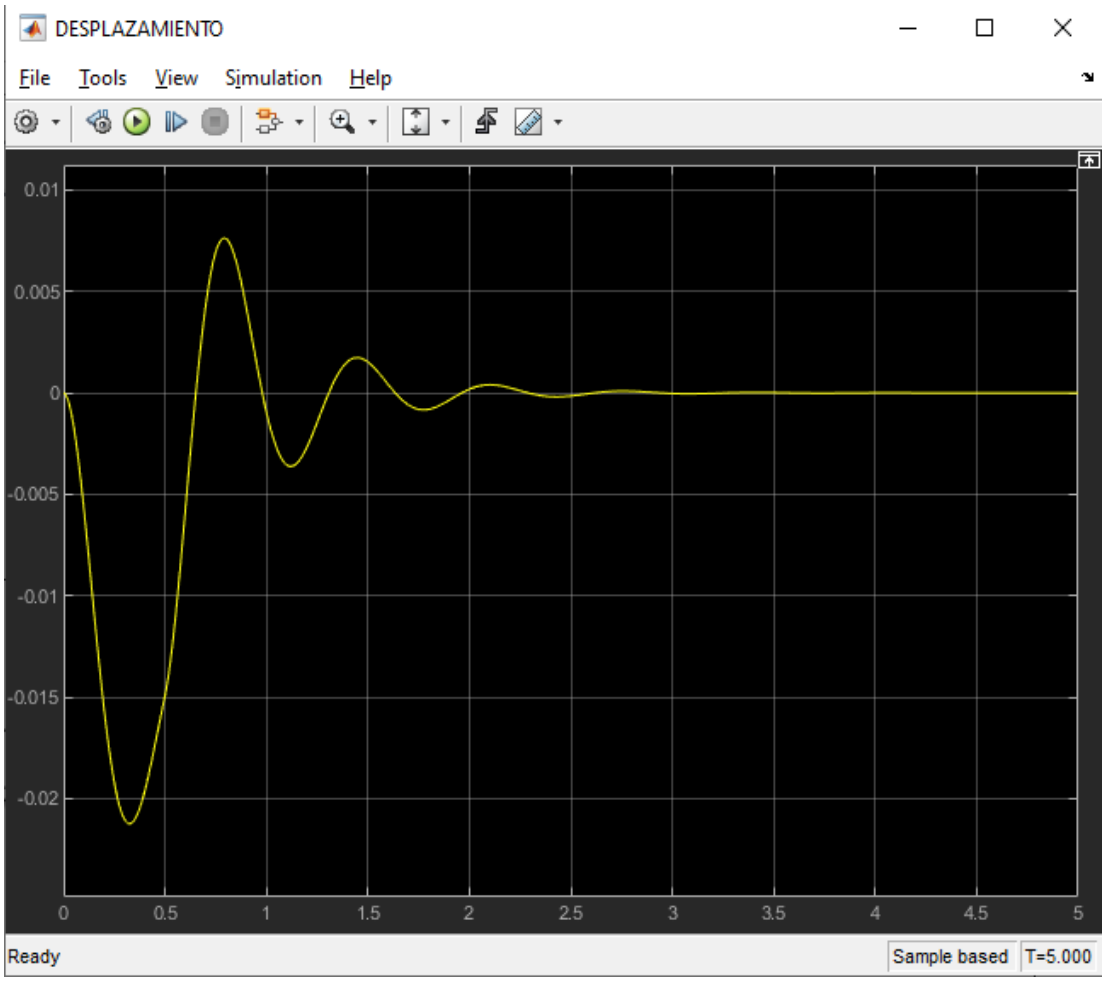


Ilustración 23 Respuesta al Diagrama de bloques con retroalimentación en Simscape

Se evidencia que el sistema es controlado en los espacios suministrados por Matlab, en código y en espacio de Simulink, Simscape por diagrama de bloques, cumpliendo con los parámetros de control dados desde un principio, fueron cumplidos a cabalidad.

En conclusión, el sistema es controlable y dependiendo las variables a considerar se puede controlar más o menos. Las variables acá expuestas están en un caso en general, sin embargo, el programa se presta para poderlas cambiar y visualizar que tipo de control se puede realizar con otras variables, el sistema es extrapolable con otras suspensiones o por lo menos el sistema y el análisis fisicomatemático es muy similar.

4.2 CONTROL DE MOVIMIENTO DE UN AVIÓN COMERCIAL

El control de movimiento de un avión comercial es un aspecto crítico de la operación en cualquier vuelo. Los sistemas de control de movimiento permiten al piloto precisamente dar control en la dirección, velocidad y altitud del avión lo que es esencial para mantener un vuelo seguro y cómodo.

Este controlo se logra mediante la combinación de múltiples sistemas, superficies de control, motores, sistemas hidráulicos y neumáticos. En general el control de movimiento de un avión es una tarea compleja que requiere la formación y habilidades especializadas por parte de ingenieros, pilotos y equipo de mantenimiento.

La ingeniería aeronáutica encarga del diseño, mantenimiento, y puesta a punto de todo tipo de aeronaves. En este caso el diseño a analizar es un modelo de avión comercial, usado para el transporte de pasajeros y equipaje. El modelo escogido está dado precisamente por las prestaciones que este tipo de aviones debe tener debido al transporte de personas, todo el conjunto del avión debe tener un factor de seguridad cerca de 1.5. Con esto se empieza el análisis.

Las ecuaciones que gobiernan el movimiento de un avión son un conjunto muy complicado de seis ecuaciones diferenciales no lineales. Sin embargo, bajo ciertas suposiciones, pueden desacoplarse y linealizarse en ecuaciones longitudinales y laterales. La inclinación del avión se rige por la dinámica longitudinal. En este ejemplo, diseñaremos un piloto automático que lo controla.

Los ejes coordenados y las fuerzas que interactúan sobre el avión se muestran a continuación:

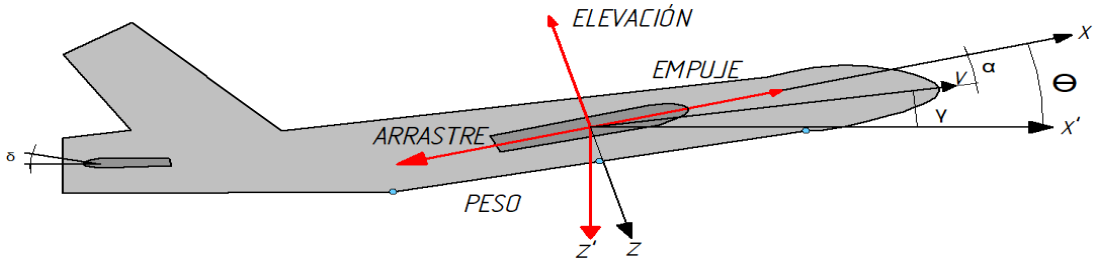


Ilustración 24 Modelo avión y diagrama de cuerpo, Autoría propia

Se supondrá que el avión se encuentra en un vuelo a altitud y velocidad constate, por ende las fuerzas de empuje, arrastre, elevación y peso se encuentran en equilibrio entre sí, en los ejes "X" y "Y", también se supondrá que cualquier cambio en el ángulo de elevación**,** no cambiará las condiciones del avión, en la realidad no sucede así, pero bajo efectos del ejercicio se simplifica el desarrollo, por tanto, las ecuaciones longitudinales de movimiento del avión se pueden escribir de la siguiente manera.

4.2.1 ECUACIONES DE MOVIMIENTO DEL SISTEMA

$$\dot{\alpha} = \eta\Omega\sigma\left[-(C_L + C_D)\alpha + \frac{1}{(\eta - C_L)}q - (C_W sin\gamma)\theta + C_L\right] \quad (1)$$

$$\dot{q} = \frac{\eta\Omega}{2i_{yy}}[[C_M - \eta(C_L + C_D)]\alpha + [C_M + \sigma C_M(1 - \eta C_L)]q + (\eta C_W sin\gamma)\delta] \quad (2)$$

$$\dot{\theta} = \Omega q \quad (3)$$

Las variables expuestas se pueden encontrar el cualquier texto relacionado con aviones, sin embargo, la definición es la siguiente:

$\alpha = Ángulo\ \ de\ \ ataque$
$q = coef\,iciente\ \ de\ \ elevación$
$\theta = Ángulo\ \ de\ \ elevación$
$\delta = Ángulo\ \ de\ \ def\,lexión\ \ de\ \ elevación$
$\mu = \frac{\rho S ë}{4m}$
$\rho = Densidad\ \ del\ \ aire$
$S = Área\ \ de\ \ la\ \ plataforma\ \ del\ \ ala$
$ë = Longitud\ \ media\ \ ala$
$m = Masa\ \ del\ \ avión$
$\Omega = \frac{2U}{ë}$
$U = Velocidad\ \ de\ \ vuelo\ \ de\ \ equilibrio$
$C_T = Coef\,iciente\ \ de\ \ empuje$
$C_D = Coef\,iciente\ \ de\ \ arrastre$
$C_L = Coef\,iciente\ \ de\ \ sustentación$
$C_W = Coef\,iciente\ \ de\ \ peso$
$C_M = Coef\,iciente\ \ de\ \ momento\ \ de\ \ elevación$
$\gamma = Ángulo\ \ de\ \ trayectoria\ \ de\ \ vuelo$
$\sigma = \frac{1}{1+\eta C_L} = Constante$
$i_{yy} = Momento\ \ de\ \ inercia$
$\eta = \mu\sigma C_M = Constante$

Para este sistema en específico, la entrada será el ángulo de deflexión de elevación (Delta) y la salida será el ángulo de elevación del avión (Theta).

Las ecuaciones acá descritas no satisfacen en su totalidad el sistema, sin embargo, conforme se vaya desarrollando el ejercicio se mostrarán otras que irán complementando.

4.2.2 MODELOS DE FUNCIÓN DE TRANSFERENCIA Y ESPACIO DE ESTADO

Antes de encontrar la función de transferencia y los modelos de espacio de estados, se introducen algunos valores numéricos para simplificar las ecuaciones del modelado que se mostraron anteriormente. Estos valores se toman de uno de los aviones comerciales de Boeing.

$\dot{a} = -0.313\alpha + 56.7q + 0.232\delta \quad (4)$
$\dot{q} = -0.0139\alpha - 0.426q + 0.0203\delta \quad (5)$
$\dot{\theta} = 56.7q \quad (6)$

FUNCIÓN TRANSFERENCIA

Para encontrar la función de transferencia del sistema anterior, se tomará la transformada de Laplace. Para ello es necesario asumir las condiciones iniciales en cero. Las ecuaciones se muestran a continuación:

$$sA(s) = -0.313A(s) + 56.7Q(s) + 0.232\Delta(s) \quad (7)$$

$$sQ(s) = -0.0139A(s) - 0.426Q(s) + 0.0203\Delta(s) \quad (8)$$

$$s\Theta(s) = 56.6(s) \quad (9)$$

Para obtener finalmente:

$$P(s) = \frac{\Theta(s)}{\Delta(s)} = \frac{1.151s + 0.1774}{s^3 + 0.739s^2 + 0.921s} \quad (10)$$

ESPACIO DE ESTADOS

Como las ecuaciones de movimiento anteriores ya están en la forma de variable de estado, lo que se hace es reescribirlas en forma matricial:

$$\begin{bmatrix} \dot{a} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -0.313 & 56.7 & 0 \\ -0.0139 & -0.426 & 0 \\ 0 & 56.7 & 0 \end{bmatrix} \begin{bmatrix} a \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} 0.232 \\ 0.0203 \\ 0 \end{bmatrix} [\delta] \quad (11)$$

Dado que la salida es el ángulo de elevación**,** la ecuación en forma matricial de la salida es la siguiente:

$$y = [0 \;\; 0 \;\; 1] \begin{bmatrix} \alpha \\ q \\ \theta \end{bmatrix} \quad (12)$$

Más adelante, conforme se vaya desarrollando el ejercicio se hará uso de las fórmulas ya descritas.

4.2.3 PARÁMETROS DEL SISTEMA E INTRODUCCIÓN DE FÓRMULAS A ENTORNO MATLAB

Con los parámetros descritos en código de MATLAB se pueden generar modelos de funciones transferencia y empezar a realizar el desarrollo lógico matemático de control.

Ejecute el siguiente código para generar el modelo de función transferencia de ciclo abierto anteriormente descrito

```
 s=tf('s'); %CONSTRUYE LA FUNCIÓN TRANSFERENCIA O CONVIERTE A FUNCIÓN
TRANSFERENCIA%
 P_pitch= (1.151*s+0.1774)/(s^3+0.739*s^2+0.921*s); %DEFINE UNA
VARIABLE CON LO DADO EN EC. 10%
```

Para generar el modelo de espacio de estados descrito anteriormente, ingrese los siguientes comandos:

```
 A=[-0.313 56.7 0; -0.0139 -0.426 0; 0 56.7 0]; %DEFINE CON VARIABLES
LAS MATRICES DADAS EN EC. 11%
 B=[0.232; 0.0203; 0]; %DEFINE CON VARIABLES LAS MATRICES DADAS EN EC.
11%
 C=[0 0 1]; %DEFINE CON VARIABLES LAS MATRICES DADAS EN EC. 11%
 D=[0]; %DEFINE CON VARIABLES LAS MATRICES DADAS EN EC. 11%
 pitch_ss=ss(A,B,C,D); %CONSTRUYE EL MODELO EN ESPACIO DE ESTADOS%
```

Del problema principal, las ecuaciones dinámicas en el dominio de Laplace y la función transferencia en lazo abierto de la dinámica de la elevación del avión, se relaciona las ecuaciones 7, 8, 9 y 10 y los requerimientos del sistema para empezar con el diseño.

4.2.4 REQUERIMIENTOS DE DISEÑO

El sistema a controlar es complejo de hacerlo, sin embargo, es un tema en común que tienen demasiados fabricantes puesto que cualquier perturbación afecta de manera clara el rendimiento del avión en general.

El siguiente paso es elegir criterios de diseño para controlar. Se diseñará un controlador de retroalimentación para que, en respuesta a un comando de ángulo de elevación tenga un sobreimpulso menor del 10%, un tiempo de subida menos de 2 segundos, un tiempo de estabilización de menos de 10 segundos y un error de estado estable de menos del 2%.

Ejemplo, si la referencia es de 0,2 radianes (11 grados), entonces el ángulo de elevación no excederá los 0,22 Rad, aumentará de 0,02 rad a 0,18 rad en 2 segundos, se estabilizará dentro del 2% de su valor de error de estado estable en 2 segundos. En resumen tenemos lo siguiente:

- Sobreimpulso inferior al 10%
- Tiempo de subida menor a 2 segundos

- Tiempo de estabilización inferior a 10 segundos
- Error de estado estacionario inferior al 2%

4.2.5 RESPUESTA EN LAZO ABIERTO

Se hará la representación del sistema en lazo abierto, se usará una entrada tipo "Step" en el comando de MATLAB precisamente para analizar la respuesta. La entrada a representar se hará con un ángulo de deflexión de elevación (delta) de 0,2 radianes o lo que vendría siendo 11 grados. El comando queda de la siguiente manera.

```
 t=[0:0.01:10];%TIEMPO DE SIMULACIÓN%
 step(0.2*P_pitch,t); %COMANDO PARA VISUALIZAR EL COMPORTAMIENTO DE LA
F.T CON EL TIEMPO Y UN PASO DE 0.2%
 axis([0 10 0 0.8]); %CONTROL DE ESCALA EN LOS EJES%
 ylabel('Ángulo de inclinación (rad)'); %AÑADE LEYENDA AL EJE Y%
 title('Respuesta de una entrada tipo Step en lazo abierto') %CONFIGURA
EL TITULO DE GRÁFICA%
```

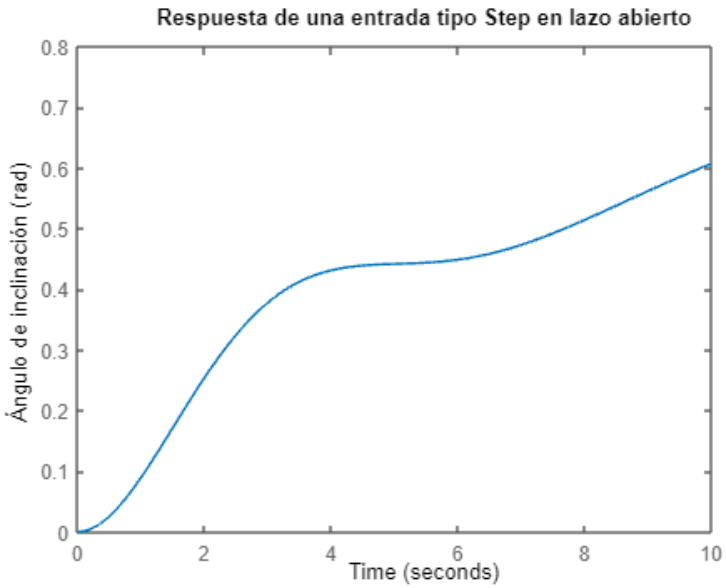


Ilustración 25 Respuesta al escalón unitario en lazo abierto

De la gráfica podemos evidenciar que el sistema no satisface en lo absoluto los requerimientos de diseño, es más, la respuesta en lazo abierto es demasiado inestable. Algo interesante es que en este tipo de sistemas la estabilidad se puede determinar examinando los polos de la función transferencia; los polos de una función transferencia se definen como valores

enteros que pertenecen a un conjunto que anulan y hacen infinito respectivamente la función transferencia, en general, los polos son un indicador de la capacidad del sistema para responder a las entradas de manera estable y predecible, además de predecir se puede ajustar la respuesta para mejorar el rendimiento. Para identificar los polos se ejecuta el siguiente comando:

```
 pole(P_pitch) %COMANDO PARA HALLAR LOS POLOS DE CUALQUIER FUNCIÓN DEFINIDA%
```

```
ans = 3×1 complex
   0.0000 + 0.0000i
  -0.3695 + 0.8857i
  -0.3695 - 0.8857i
```

Cómo se evidencia, uno de los polos de la función transferencia se encuentra en el eje imaginario, mientras que los otros dos polos se encuentra en la parte negativa (izquierda de un plano) esto nos indica que la respuesta del sistema es estable y su respuesta temporal es acotada y convergente. Un polo en el eje imaginario nos da como solución que la respuesta del sistema no crecerá ilimitadamente, pero tampoco decaerá en cero. Esto también depende de la entrada que se le está dando, es decir, un sistema con un polo en el eje imaginario puede crecer ilimitadamente cuando se le de cualquier entrada incluso si ésta está limitada y es precisamente lo que sucede en este caso en particular, el polo en el origen se comporta como un integrador, por lo tanto, cuando al sistema se le da una entrada escalonada, su salida continúa creciendo hasta el infinito de la misma manera que la integral de una constante crecerá hasta el infinito a medida que el límite superior de la integral se hace más grande.

Ahora bien, se verá el comportamiento del sistema en lazo cerrado.

4.2.6 RESPUESTA EN LAZO CERRADO

Para estabilizar el sistema y eventualmente cumplir con nuestros requerimientos de diseño, se agregará un controlador de retroalimentación, a continuación se ilustra la arquitectura del control que se empleará.

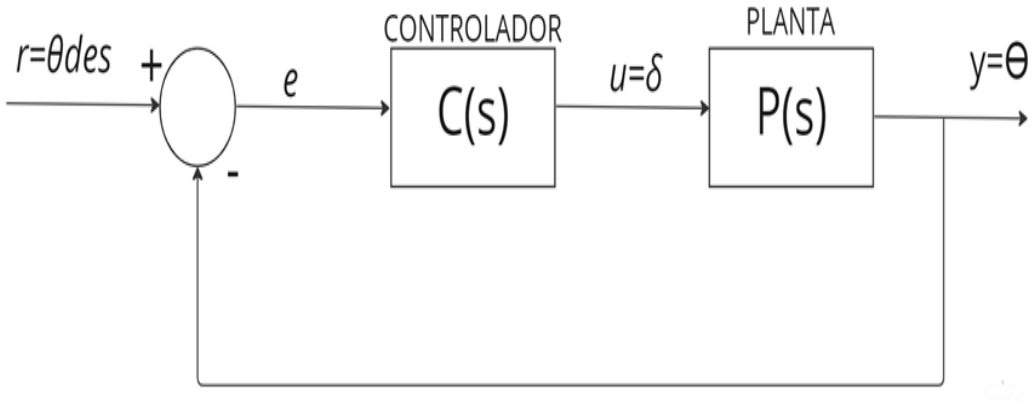


Ilustración 26 Esquema del sistema en lazo cerrado

La función transferencia en lazo cerrado para lo anterior con el controlador C(s) simplemente igualado a uno, se puede generar utilizando el código "*feedback"* como se presenta a continuación:

```
 sys_cl=feedback(P_pitch,1) %COMANDO PARA OBTENER LA RETROALIMENTACIÓN
DEL CONTROLADOR IGUALADO A 1%
```

```
sys_cl =

           1.151 s + 0.1774
  ---------------------------------
  s^3 + 0.739 s^2 + 2.072 s + 0.1774

Continuous-time transfer function.
```

La respuesta del Step correspondiente se puede generar agregando los comandos anteriormente usados en la respuesta de lazo abierto, así entonces tendríamos lo siguiente:

```
 step(0.2*sys_cl); %COMANDO PARA VISUALIZAR EL COMPORTAMIENTO DE LA F.T
Y UN PASO DE 0.2%
 ylabel('Ángulo de inclinación (rad)'); %AÑADE LEYENDA AL EJE Y%
 title('Respuesta de una entrada tipo Step en lazo cerrado') %CONFIGURA
EL TITULO DE GRÁFICA%
```

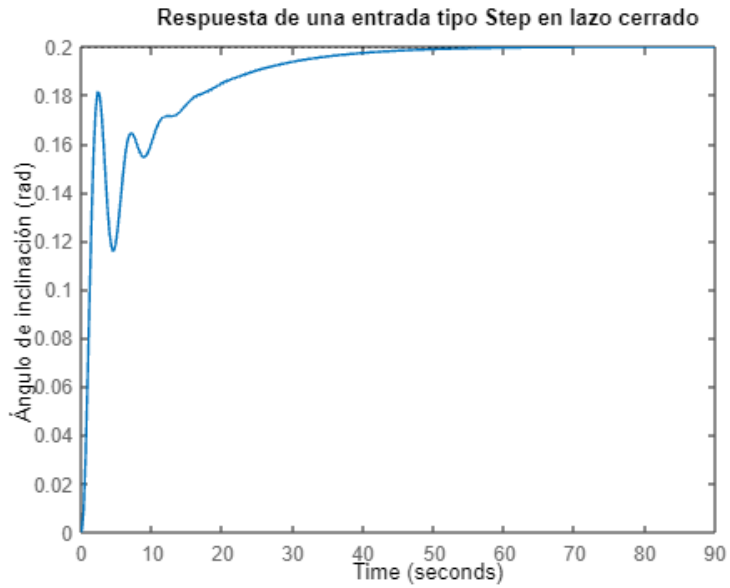


Ilustración 27 Respuesta del sistema de lazo cerrado

Análisis de grafica

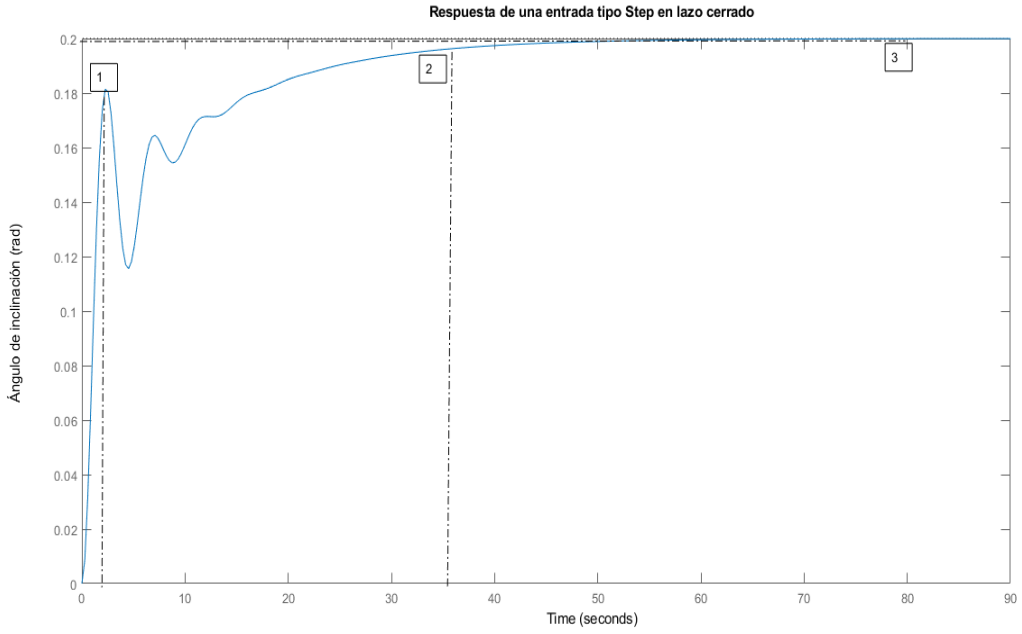


Ilustración 28 Análisis de la gráfica respuesta del sistema de lazo cerrado

1. Tiempo de subida en segundos: 1.79 segundos
2. Tiempo de estabilización en segundos: 35.1 segundos

3. Valor final obtenido: 0.2

Al examinar la respuesta tipo Step en lazo cerrado, la adición del bloque de retroalimentación ha estabilizado el sistema, de hecho el error de estado estacionario parece llevarse a cero, no hay un sobreimpulso en la respuesta, aunque no se cumplen los requerimientos de tiempo de subida y de estabilización. Esto está dado por la respuesta al Step indicado por la ubicación de los polos y ceros de la función transferencia del sistema, de manera similar a como eran las propiedades de estabilización del sistema. MATLAB con algunos comandos nos ayuda a revelar cuales son los polos y ceros de la función transferencia en lazo cerrado como se muestra a continuación

```
 poles=pole(sys_cl) %COMANDO PARA VISUALIZAR LOS POLOS DE UNA FUNCIÓN DEFINIDA%
```

```
poles = 3×1 complex
  -0.3255 + 1.3816i
  -0.3255 - 1.3816i
  -0.0881 + 0.0000i
```

```
 zeros=zero(sys_cl) %COMANDO PARA VISUALIZAR LOS CEROS DE UNA FUNCIÓN DEFINIDA%
```

```
zeros = -0.1541
```

Los resultados anteriores demuestran que la función de transferencia en lazo cerrado es de tercer orden con un cero. La mayoría de las relaciones con las que se están familiarizados para predecir el carácter de la respuesta tipo Step de un sistema suponen un sistema estándar de segundo orden subamortiguado sin ceros. Por lo tanto no es confiable estas relaciones para este sistema en particular, sin embargo, podemos volver a transformar la salida al dominio del tiempo para generar una función de tiempo para la respuesta del sistema para obtener una idea de cómo los polos y ceros de la función de transferencia de lazo cerrado afectan la respuesta del sistema.

Suponiendo que la función transferencia de lazo cerrado tiene la forma $Y(s)/R(s)$, la salida $Y(s)$ en el dominio de Laplace se calcula de la siguiente manera donde $R(s)$ es un paso de magnitud 0,2.

$$Y(s) = \frac{1.151s + 0.1774}{s^3 + 0.739s^2 + 2.072s + 0.1774} R(s) \quad (13)$$

$$Y(s) = \frac{0.2(1.151s + 0.1774)}{s^4 + 0.739s^3 + 2.072s^2 + 0.1774s} \quad (14)$$

Luego, podemos realizar una expansión de fracciones parciales para dividir esta expresión en términos más simples que, con suerte, reconocemos y podemos convertir del dominio de Laplace al dominio del tiempo. Primero, usaremos el comando MATLAB zpk para factorizar el numerador y el denominador de nuestra salida *Y*(*s*) en términos más simples.

```
 R=0.2/s; %SE DEFINE UNA VARIABLE%
 Y=zpk(sys_cl*R) %CONSTRUTE EL MODELO DE GANANCIA DE POLOS A CEROS O SE
CONVIERTE A FORMATO DE GANANCIAS DE POLOS A CEROS%
```

```
Y =

           0.2302 (s+0.1541)
  -------------------------------------
  s (s+0.08805) (s^2 + 0.6509s + 2.015)

Continuous-time zero/pole/gain model.
```

Con base en lo anterior, el denominador de nuestra salida *Y*(*s*) se puede factorizar en un término de primer orden para el polo real de la función de transferencia, un término de segundo orden para los polos conjugados complejos de la función de transferencia y un término de polo en el origen para la entrada tipo Step. Por lo tanto, se desea expandir *Y*(*s*) como se muestra a continuación.

$$Y(s) = \frac{0.2302(s + 0.1541)}{s(s + 0.08805)(s^2 + 0.6509s + 2.105)} \quad (15)$$

$$Y(s) = \frac{A}{s} + \frac{B}{s + 0.08805} + \frac{Cs + D}{s^2 + 0.6509s + 2.015} \quad (16)$$

Los valores específicos de las constantes *A*, *B*, *C* y *D* se pueden determinar mediante cálculo manual o utilizando el *residue* de comando de MATLAB para realizar la expansión de fracciones parciales como se muestra a continuación. Aquí la sintaxis es *[r,p,k] = residue(num,den)* donde *num* y *den* son matrices que contienen los coeficientes del numerador y el denominador, respectivamente, de la función de Laplace que se está expandiendo. Observe que la matriz del denominador incluye un marcador de posición cero ya que no hay un término constante en el denominador de *Y*(*s)*. En código queda hecho de la siguiente manera:

```
 [r,p,k]=residue(0.2*[1.151 0.1774],[1 0.739 2.072 0.1774 0]) %COMANDO
PARA ENCONTRAR LOS RESIDUOS DE FRACCIONES PARCIALES%
```

```
r = 4×1 complex
```

```
  -0.0560 + 0.0160i
  -0.0560 - 0.0160i
  -0.0879 + 0.0000i
   0.2000 + 0.0000i
p = 4×1 complex
  -0.3255 + 1.3816i
  -0.3255 - 1.3816i
  -0.0881 + 0.0000i
   0.0000 + 0.0000i
k =

    []
```

Teniendo en cuenta lo anterior, "r" es una matriz que contiene los residuos de la expansión en fracciones parciales, es decir, los coeficientes del numerador en la expansión. En "p" contiene los polos del sistema donde el orden corresponde al orden de los residuos en "r". Finalmente el término "k" está vacío en este caso lo será en general, ya que el polinomio del numerador generalmente será de un orden menor que el polinomio del denominador. Con base en lo anterior, los coeficientes *A* y *B* en la expresión son iguales a 0.2 y -0.0881, respectivamente. Los coeficientes *C* y *D* se pueden determinar combinando los términos de los polos conjugados complejos en una sola expresión como se muestra a continuación.

```
 [num,den]=residue(r(1:2),p(1:2),k); %COMANDO PARA ENCONTRAR LOS
RESIDUOS DE FRACCIONES PARCIALES%
 tf(num,den) %HALLA FUNCIÓN TRANSFERENCIA DE VARIABLES DEFINIDAS%
```

```
ans =

   -0.1121 s - 0.08071
  ----------------------
  s^2 + 0.6509 s + 2.015

Continuous-time transfer function.
```

Con lo anterior, *C* = -0.1121 y *D* = -0.08071 y nuestra expansión en fracción parcial resultante se puede expresar de la siguiente manera.

$$Y(s) = \frac{0.2}{s} + \frac{0.0881}{s + 0.08805} + \frac{0.1121s + 0.08071}{s^2 + 0.6509s + 2.015} \quad (17)$$

Empleando una tabla de transformada de Laplace, la transformada de Laplace inversa de la expresión anterior se puede tomar para generar la expresión de dominio de tiempo correspondiente que se muestra a continuación:

$$y(t) = 0.2 - 0.0881e^{-0.08805t} - e^{-0.3255t}(0.1121cos(1.3816t) + 0.0320sin(1.3816t)) \quad (18)$$

Examinando lo anterior, cada término corresponde a un polo de $Y(s)$ donde la parte real del polo describe el decaimiento (o crecimiento) exponencial, de ese modo y la parte imaginaria del polo corresponde a la frecuencia de oscilación del modo. El efecto de los ceros es alterar los coeficientes multiplicando cada uno de los términos. En otras palabras, los ceros afectan la contribución relativa de cada uno de los modos. El ejemplo anterior ayuda a dar una idea de cómo los polos y ceros en el dominio de Laplace indican el comportamiento correspondiente del sistema en el dominio del tiempo.

El siguiente código generará el gráfico con una entrada tipo Step y así poder visualizar si el sistema es controlable con el procedimiento anteriormente hecho

```
 t=[0:0.1:70]; %VARIABLE QUE DEFINE TIEMPO DE SIMULACIÓN%
 y = 0.2 - 0.0881*exp(-0.08805*t) - exp(-
0.3255*t).*(0.1121*cos(1.3816*t)+0.0320*sin(1.3816*t)); %DEFINE EN
CODIGO EC. 18%
 plot(t,y) %COMANDO PARA GRAFICAR%
 xlabel('tiempo (seg)'); %AÑADE LEYENDA AL EJE X%
 ylabel('Ángulo de elevación (rad)');%AÑADE LEYENDA AL EJE Y%
 title('Respuesta en lazo cerrado'); %CONFIGURA EL TITULO DE LA
GRAFICA%
```

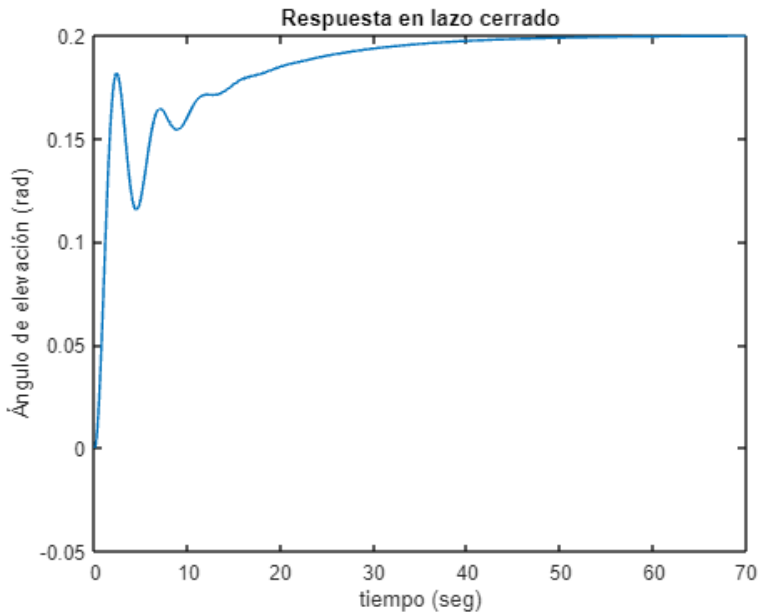


Ilustración 29 Respuesta en lazo cerrado

El gráfico anterior demuestra que nuevamente el sistema en lazo cerrado no cumple con los requerimientos de diseño planteados en el principio.

4.2.7 IMPLEMENTACIÓN DE UN CONTROLADOR PID

Recordemos que la función para un controlador PID está dada por la función transferencia

$$C(s) = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s} \qquad (19)$$

En este caso se implementará combinaciones de control proporcional (Kp), integral (Ki), y derivativo (Kd) en la retroalimentación unitaria que se muestra a continuación para lograr el comportamiento deseado del sistema.

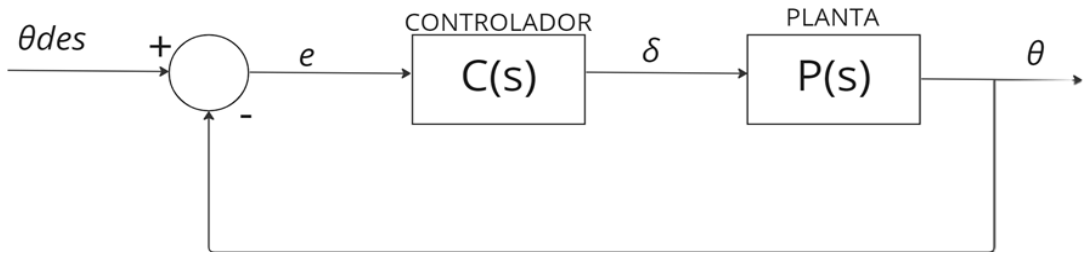


Ilustración 30 Esquema del sistema con retroalimentación

Se aprovecharán las capacidades de ajuste automatizado de MATLAB del "Control System Designer" para el diseño del controlador PID, para ello, se escriben en código el modelo de nuestra planta P(s) anteriormente ya descrito, tenemos:

```
 s=tf('s'); %SE LLAMA VARIABLES YA DEFINIDAS%
 P_pitch=(1.151*s+0.1774)/(s^3+0.739*s^2+0.921*s); %SE LLAMA VARIABLES
YA DEFINIDAS%
```

4.2.8 CONTROL PROPORCIONAL (Kp)

Se empezará diseñando un controlador proporcional de la forma C(s)=Kp. el "Control System Designer" que se usará se puede abrir escribiendo el siguiente código:

```
 controlSystemDesigner(P_pitch); %ABRE LA APLICACIÓN DISEÑADOR DE
SISTEMA DE CONTROL%
```

Al ejecutarlo se abrirá un gráfico del lugar geométrico de las raíces, el gráfico de Bode en lazo abierto y el gráfico de respuesta tipo Step en lazo cerrado, todo esto para la función transferencia de planta proporcionada anteriormente con el controlador C(s)=1, de forma predeterminada. La ventana que se muestra es la siguiente:

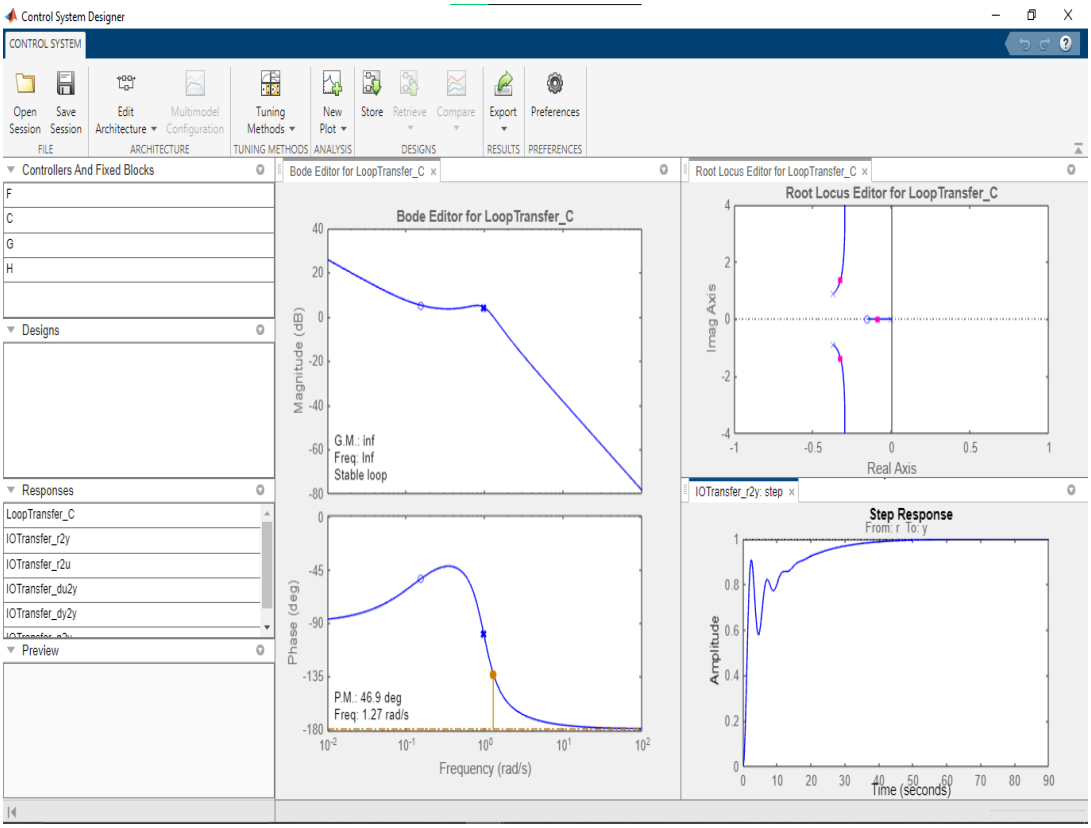


Ilustración 31 Interfaz gráfica apartado Control System Designer

Dado que nuestra referencia es la función tipo Step de 0,2 rad, se puede establecer el bloque "precompensator block F(s)" es igual a 0.2 escalar para una entrada tipo Step. Se configura en la ventana de "Controllers And Fixed Blocks", quedando de la siguiente manera:

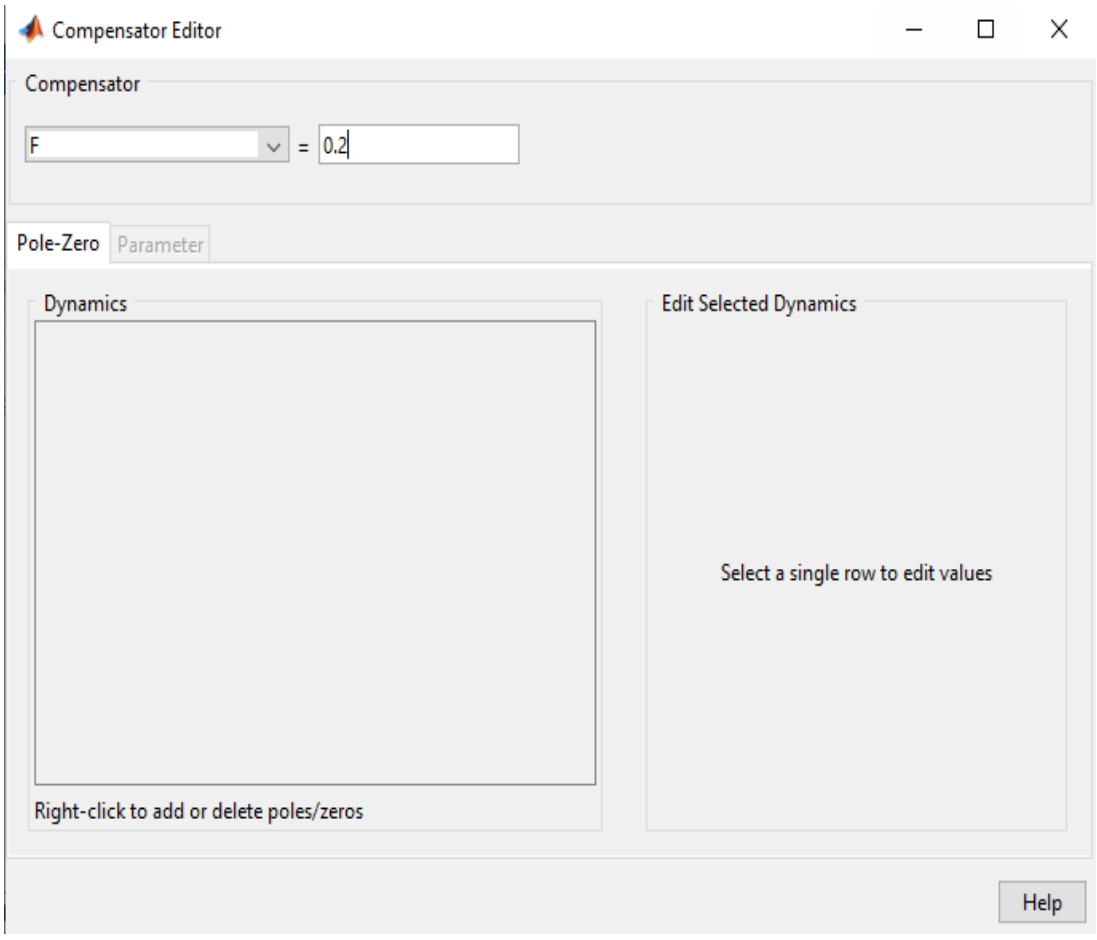


Ilustración 32 Configuración compensador F

Para empezar, visualice cómo funciona el sistema con un controlador proporcional establecido en 2. El compensador C(s) se puede definir de la misma manera que se hizo anteriormente, solamente escoja C en el menú desplegable y configúrelo en 2, como se muestra a continuación:

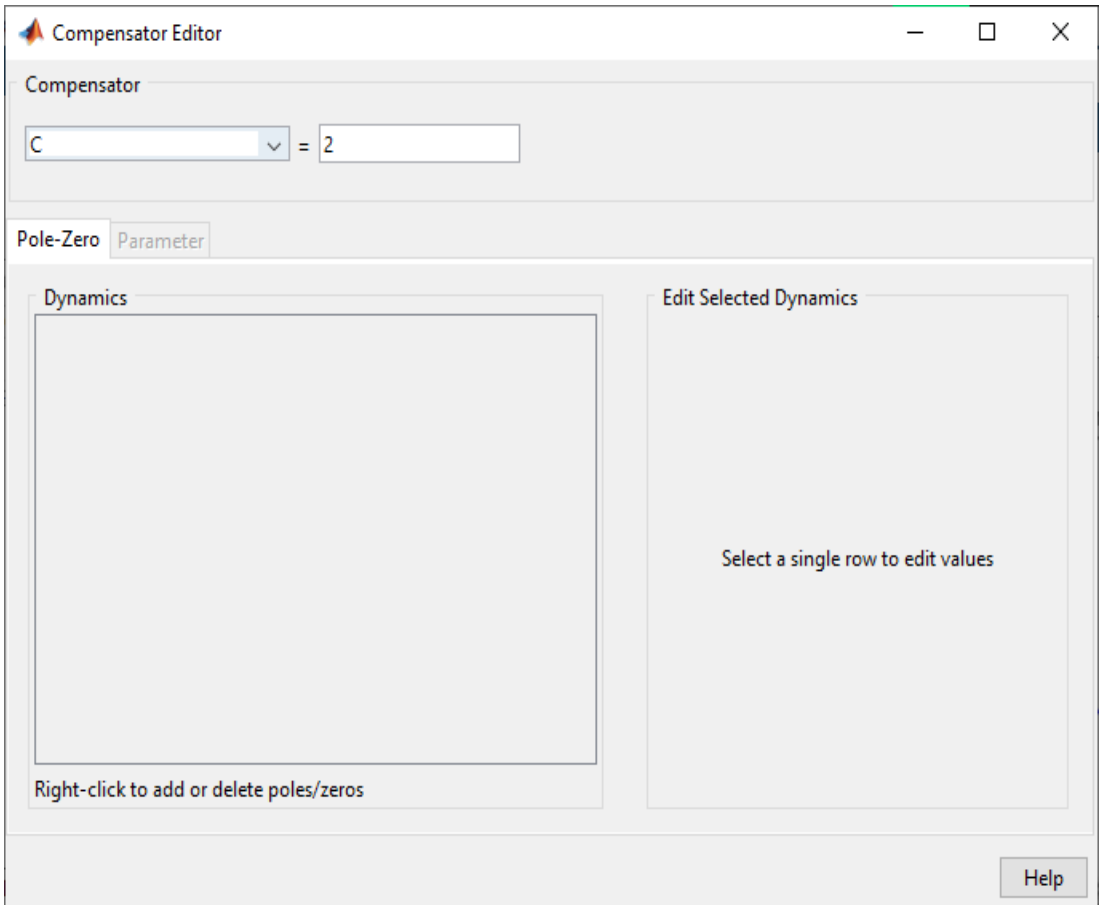


Ilustración 33 Configuración compensador C

Para poder visualizar el rendimiento del sistema con este controlador, deberá seguir los siguientes pasos:

1. En el apartado CONTROL SYSTEM, seleccione New Plot, ejecutando New Step

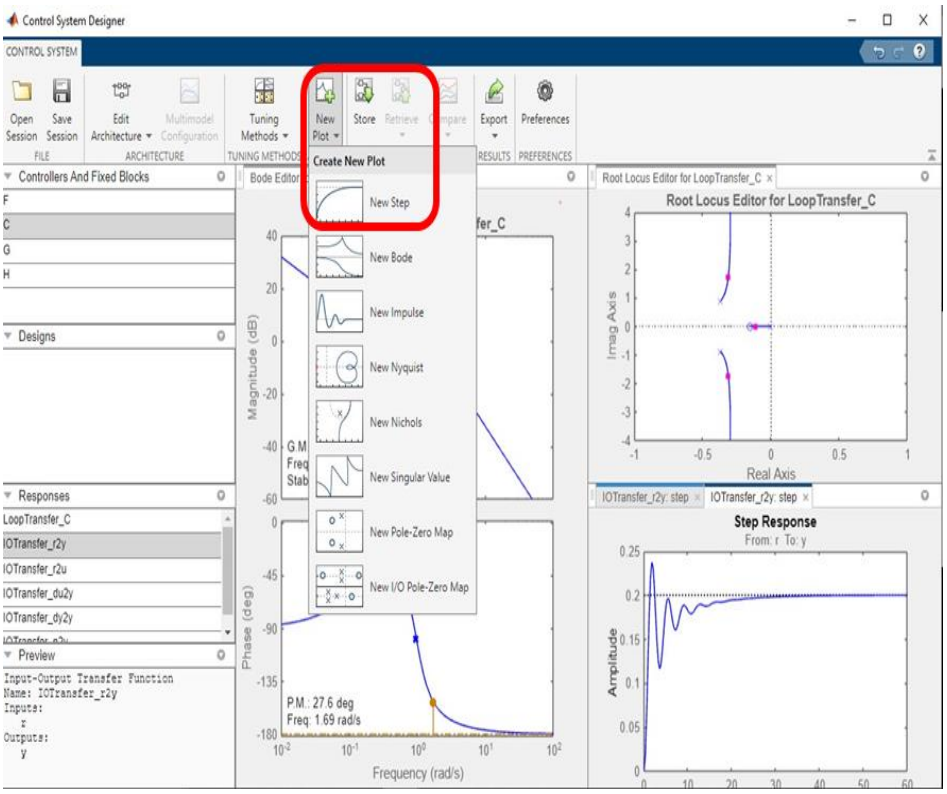


Ilustración 34 Creación de nueva gráfica en entorno del control system

2. Se desplegará otra ventana donde tendrá que elegir qué tipo de respuesta es la que quiere visualizar, tendrá que escoger "IOTransfer_r2y"

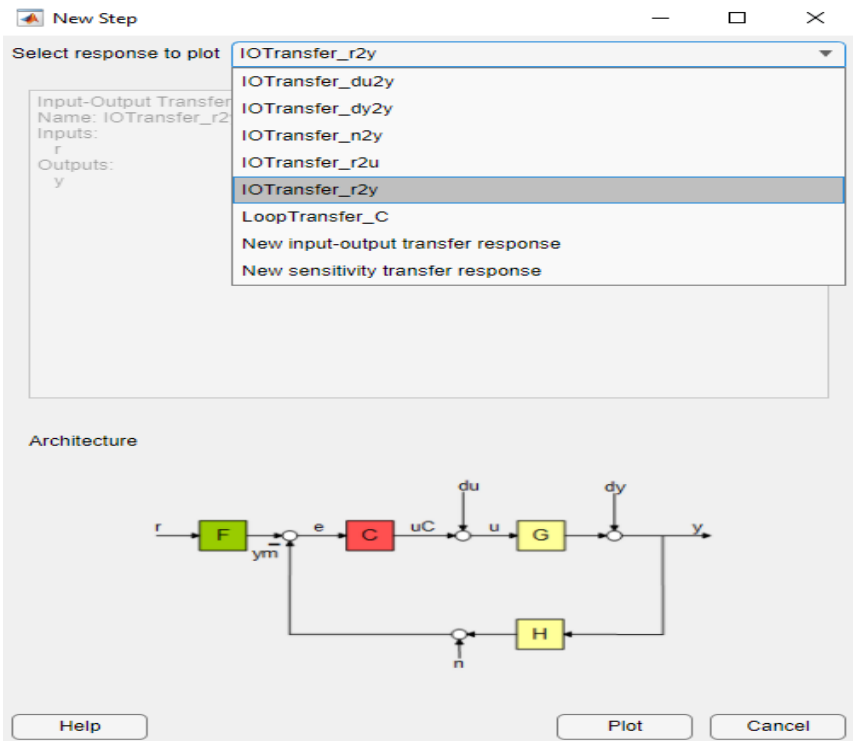


Ilustración 35 Configuración de la gráfica

3. Finalmente se abrirá una respuesta tipo Step de la siguiente manera:

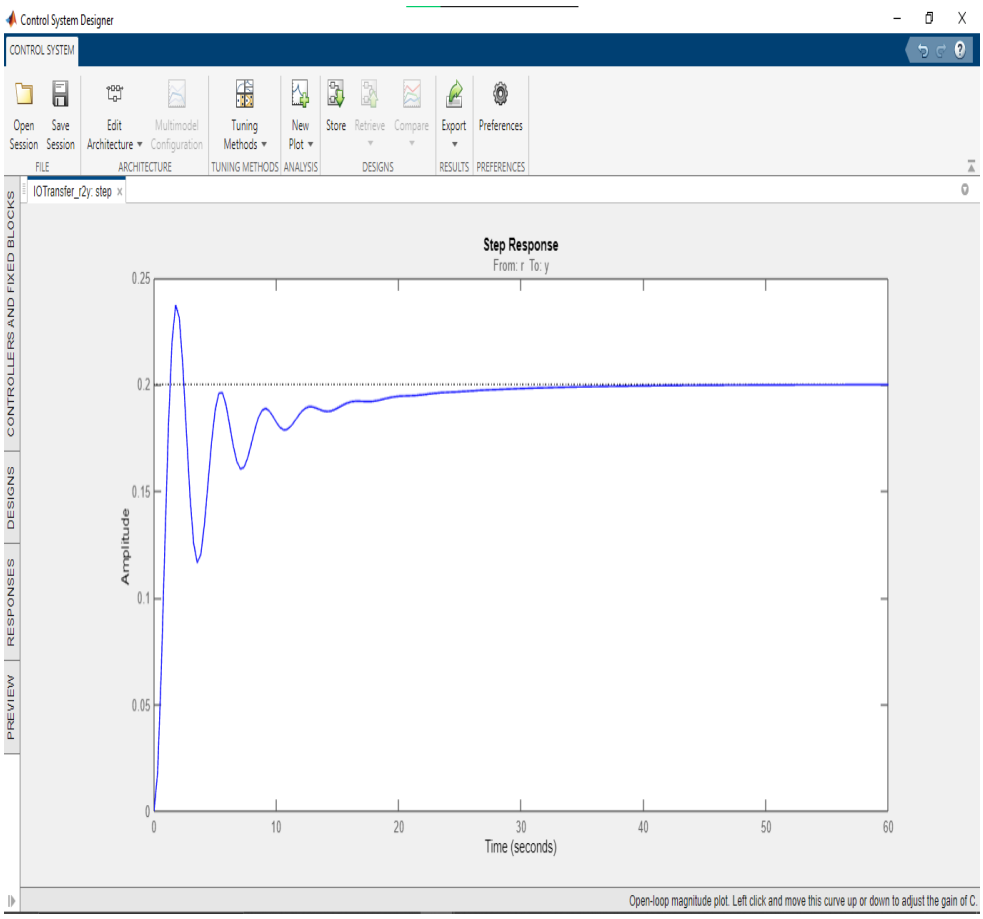


Ilustración 36 Resultado de la gráfica resultante

La gráfica anterior muestra que, a parte del error de estado estable, aún no se cumple con los requerimientos del sistema. La ganancia elegida Kp se puede ajustar en un intento acelerado para modificar el rendimiento resultante a través de la ventana "Compensator Editor" en el menú desplegable de "Controllers and fixes blocks". En su lugar, utilizaremos el "Control System Designer" para ajustar automáticamente la ganancia proporcional, Para usar esta función, diríjase a la ventana "Tuning Methods" y escoja la opción "PID tuning" en el menú de "Automated Tuning". El procedimiento se ilustra a continuación:

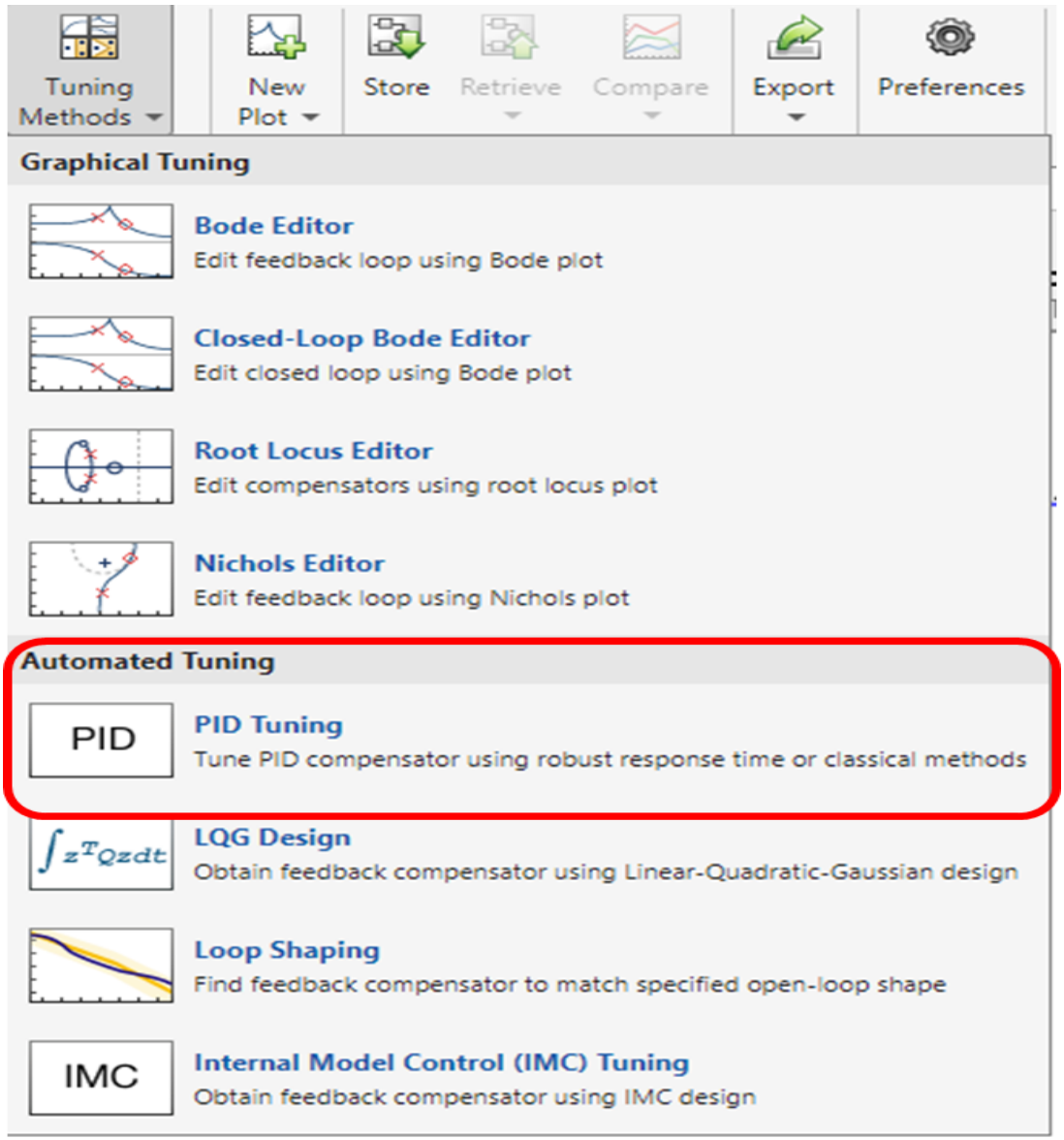


Ilustración 37 Menú tuning methods

Después de seleccionado, del programa emergerá otra ventana, como se muestra a continuación:

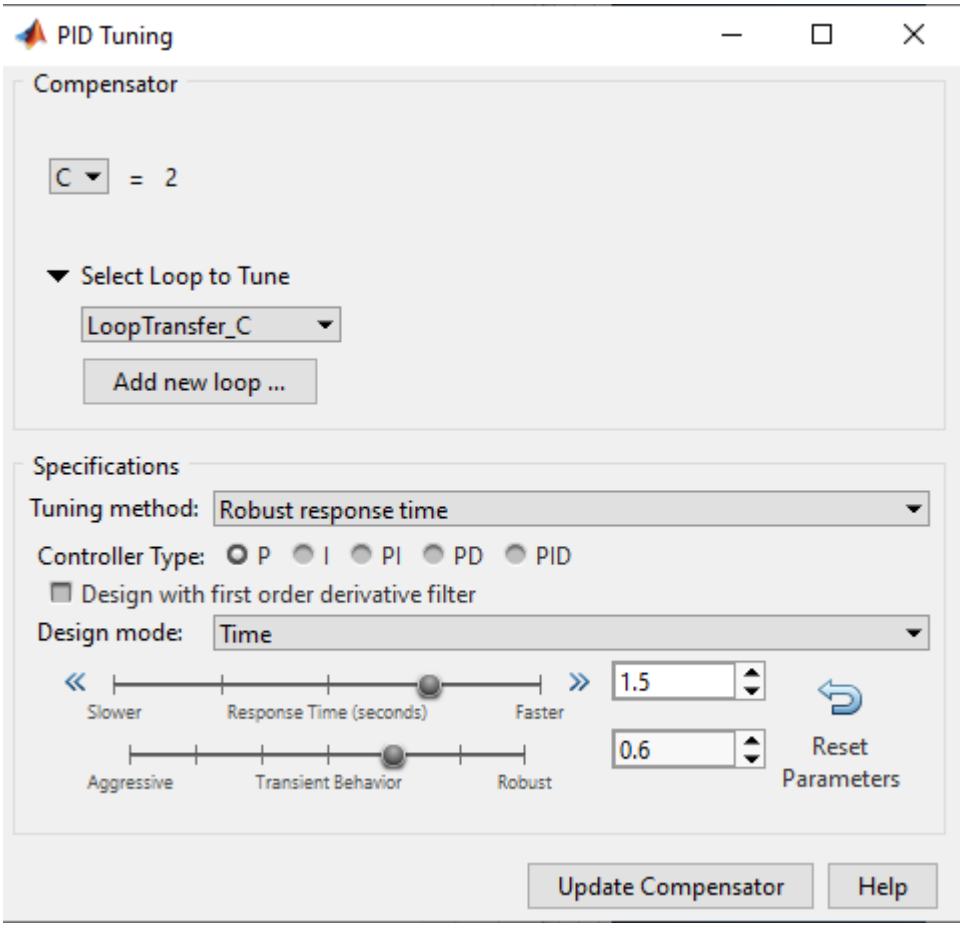


Ilustración 38 Configuración compensador C

Asegúrese de escoger la ganancia proporcional en el tipo de controlador.

Hay 2 opciones que se pueden escoger para el método de control, está "Robust response time" y "Classical design formulas". El primero "Robust response time" ajusta automáticamente los parámetros PID para equilibrar la velocidad de respuesta, se puede ajustar todos los parámetros para cualquier tipo de controlador PID y además, también es usado para el diseño de plantas que son estables, inestables o integradoras. Por otro lado el segundo método en lista "Classical design formulas" requiere una planta estable o integradora y no puede ajustar el filtro derivado. Si selecciona el algoritmo de "Classical design formulas", entonces en el menú desplegable verá una gama de opciones. Estas opciones van desde técnicas heurísticas, como Ziegler-Nichols, hasta enfoques numéricos que buscan todas las ganancias de control posibles para minimizar algún índice de rendimiento identificado.

En este caso se usará el primer método, que además, viene por defecto, en el menú desplegable "Desing Mode" podrá elegir tiempo o frecuencia, debido a la configuración del sistema y los requerimientos del sistema el "Desing Mode" estará configurado en tiempo.

Dado que se espera que el tiempo de elevación sea inferior a 2 segundos, según los requerimientos, especifique un "Response Time" de 1.5 segundos. Finalmente de click en el recuadro "Update Compensator".

El algoritmo elige una ganancia proporcional de Kp=1.1269, con esta ganancia el controlador cumple con el requerimiento del tiempo de subida, pero no con el tiempo de estabilización que aún es demasiado largo, se puede evidenciar en la siguiente gráfica el comportamiento.

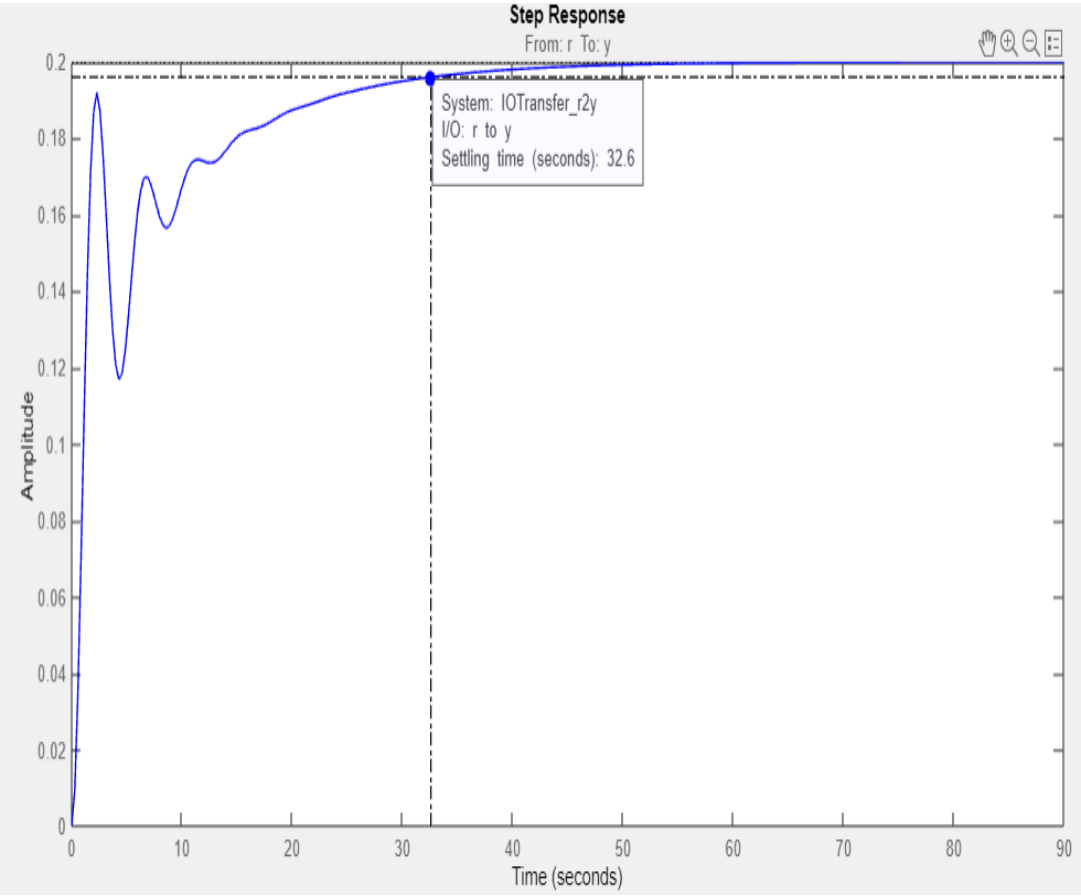


Ilustración 39 Comportamiento del sistema con ganancia Kp

Si se fija el tiempo de asentamiento esta alrededor de los 32 segundos, un tiempo considerablemente excesivo

4.2.9 CONTROL PI

Se va a realizar un control PI (Proporcional, que fue el que se hizo anteriormente e integral para reducir el error en estado estable) en este caso en particular el error en estado estable está controlado, sin embargo con fines ilustrativos se va a realizar. Para ello vuelva a ejecutar la ventana de PID TUNING y seleccione en "Controller Type" PI, quedando de la siguiente manera, actualice el compensador y verá el siguiente resultado:

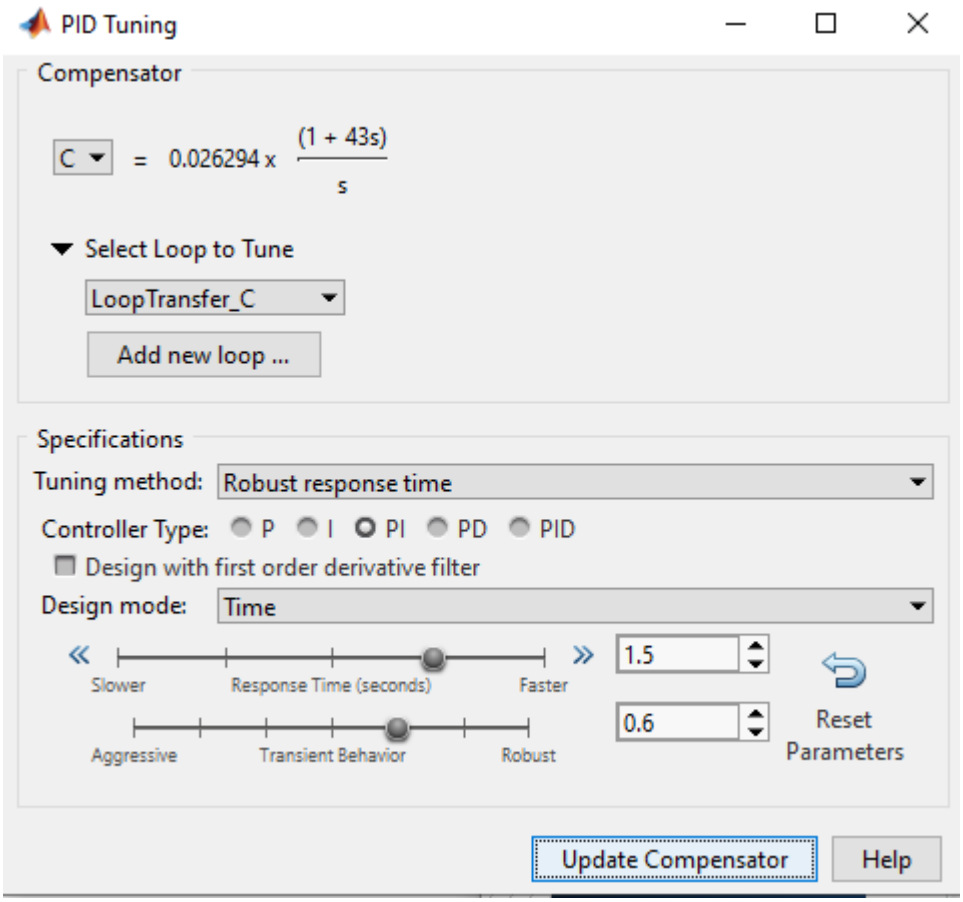


Ilustración 40 Fórmula compensador C y controlador PI

La fórmula del controlador queda de la siguiente forma:

$$C(s) = 0.026294\frac{1+43s}{s} \approx \frac{0.0263}{s} + 1.13 \qquad (20)$$

De este modo tenemos que la función transferencia con un controlador PI tenemos las ganancias Ki=0.0263 y Kp=1.13, el resultado grafico en lazo cerrado es el siguiente:

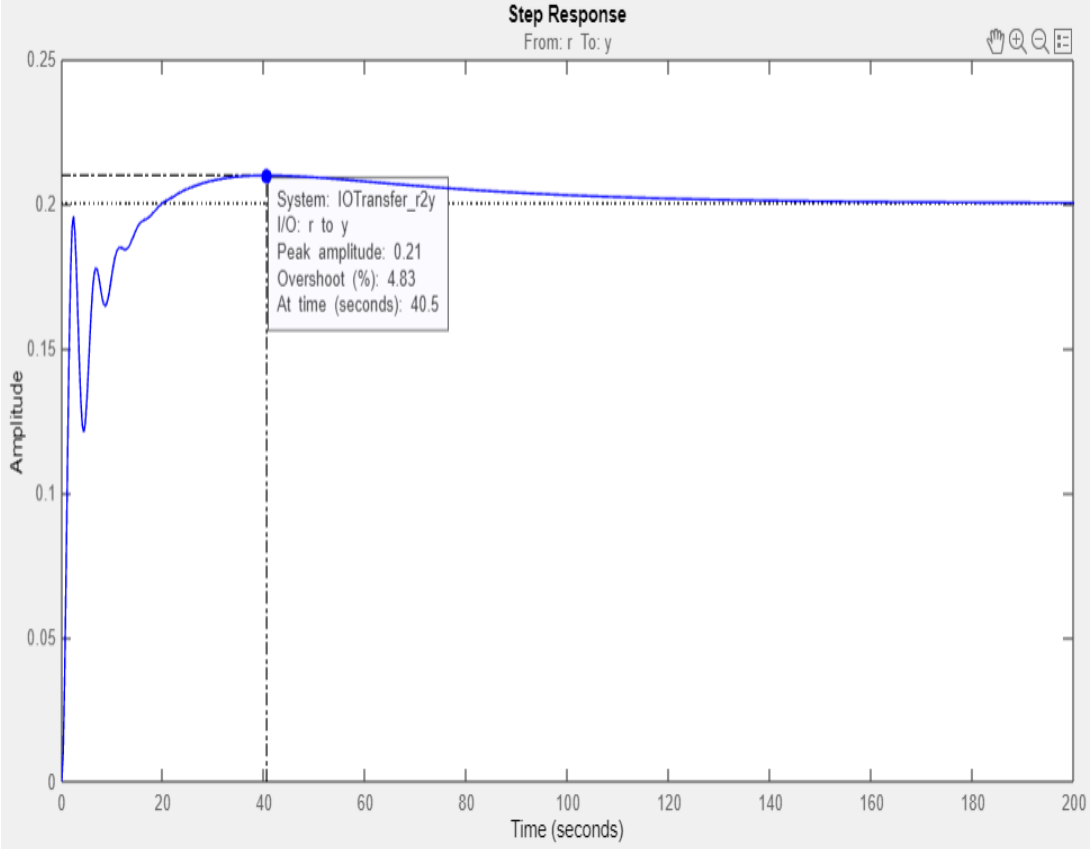


Ilustración 41 Comportamiento del sistema con ganancia Kp y Ki

Del resultado anterior, observe que la adición del controlador PI ayudó a reducir el error promedio en la señal más rápidamente, sin embargo no ayudó a reducir la oscilación. Para ello se agregará una ganancia derivativa a nuestro controlador

4.2.10 CONTROL PID

Ya sabemos la funcionalidad de las ganancias proporcionales e integral, ahora bien, una ganancia derivativa ayuda a reducir el sobreimpulso generado, por lo tanto, al agregar una ganancia derivativa, se puede reducir las oscilaciones en la respuesta lo suficiente para que después se puedan aumentar las otras ganancias y así reducir el tiempo de asentamiento.

Nuevamente vuelva a ejecutar la ventana de PID TUNING y seleccione en "Controller Type" PID, quedando de la siguiente manera, actualice el compensador y verá el siguiente resultado:

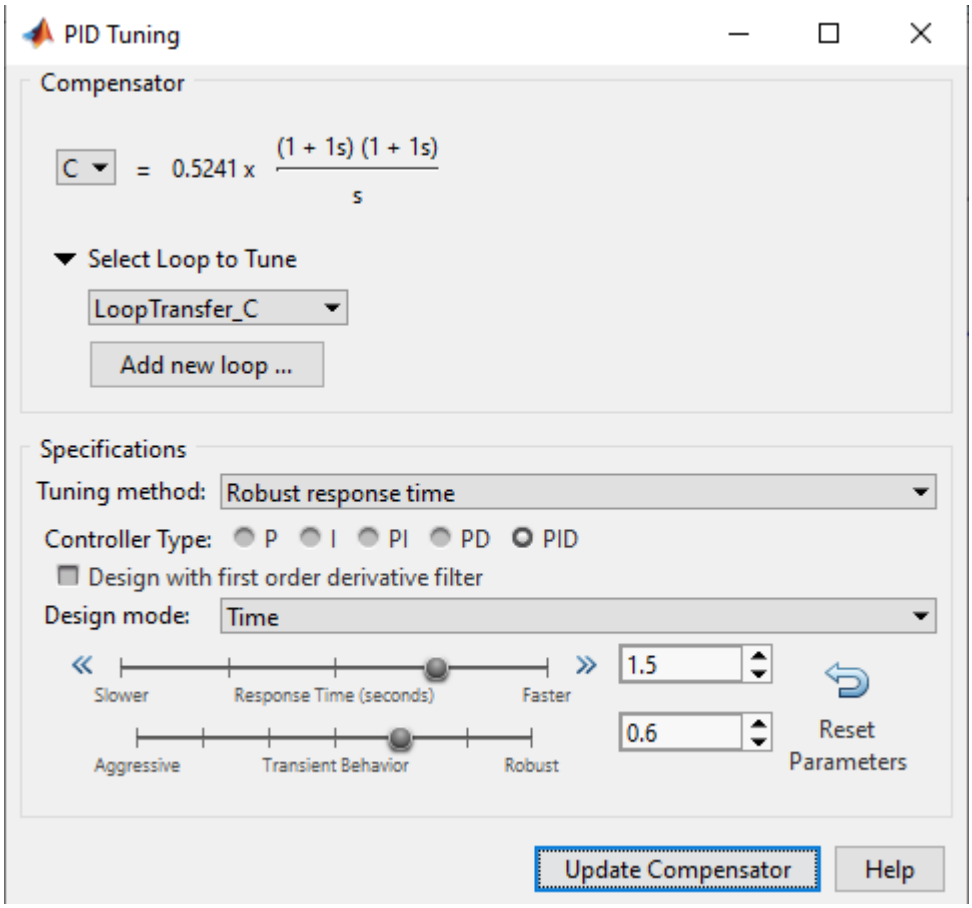


Ilustración 42 Fórmula compensador C y controlador PID

La fórmula del controlador queda de la siguiente forma:

$$C(s) = 0.5241\frac{(1+1s)(1+1s)}{s} \approx \frac{0.5241}{s} + 1.0482 + 0.5241s \quad (21)$$

De este modo la función transferencia con un controlador PID se generan las ganancias Ki=0.5241, Kp=1.0482 y Kd=0.5241. El resultado grafico en lazo cerrado es el siguiente:

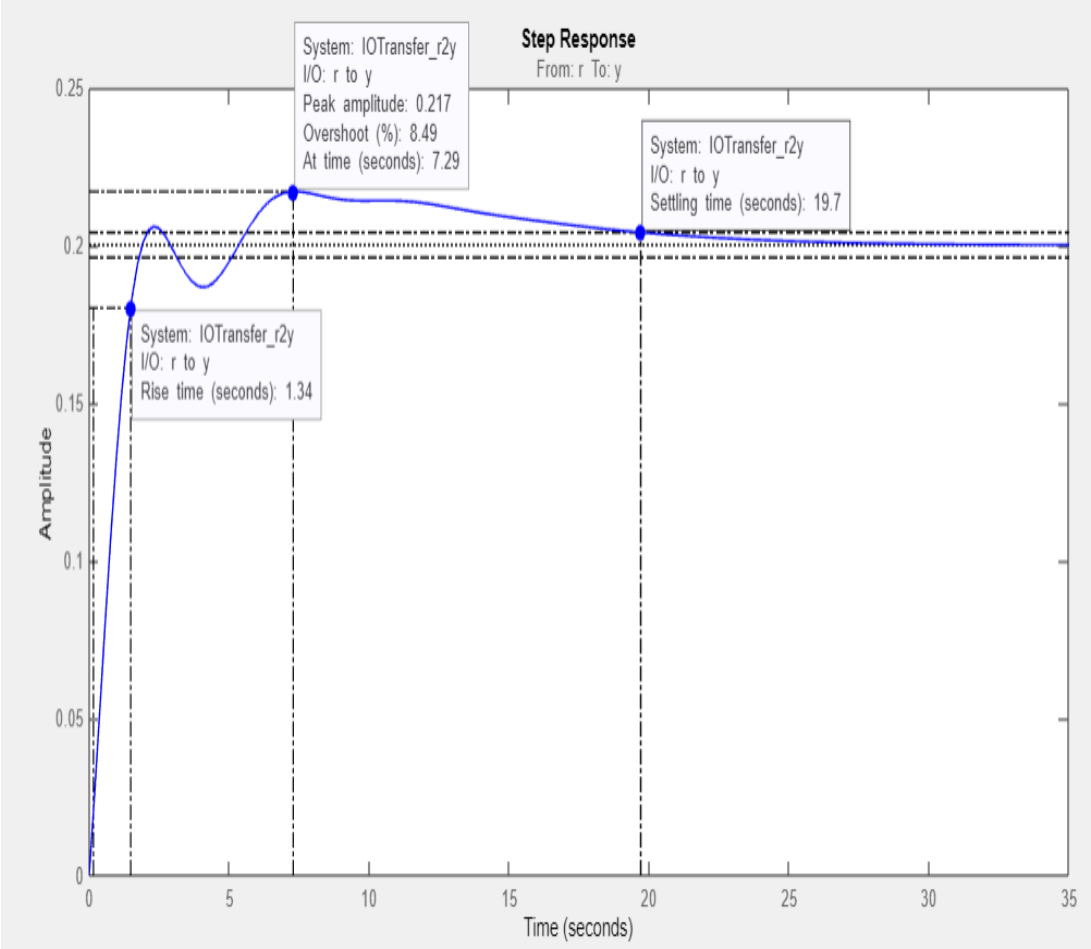


Ilustración 43 Comportamiento del sistema con ganancia Kp, Ki y Kd

Esta respuesta cumple con todos los requerimientos del sistema excepto por el tiempo de estabilización que en este caso es de 19.7s mucho mayor a los 10s que requiere el sistema. De este modo se tendrá que hacer el ajuste manual con la ventana emergente de "PID TUNING", se muestra a continuación el ajuste manual que debe hacer

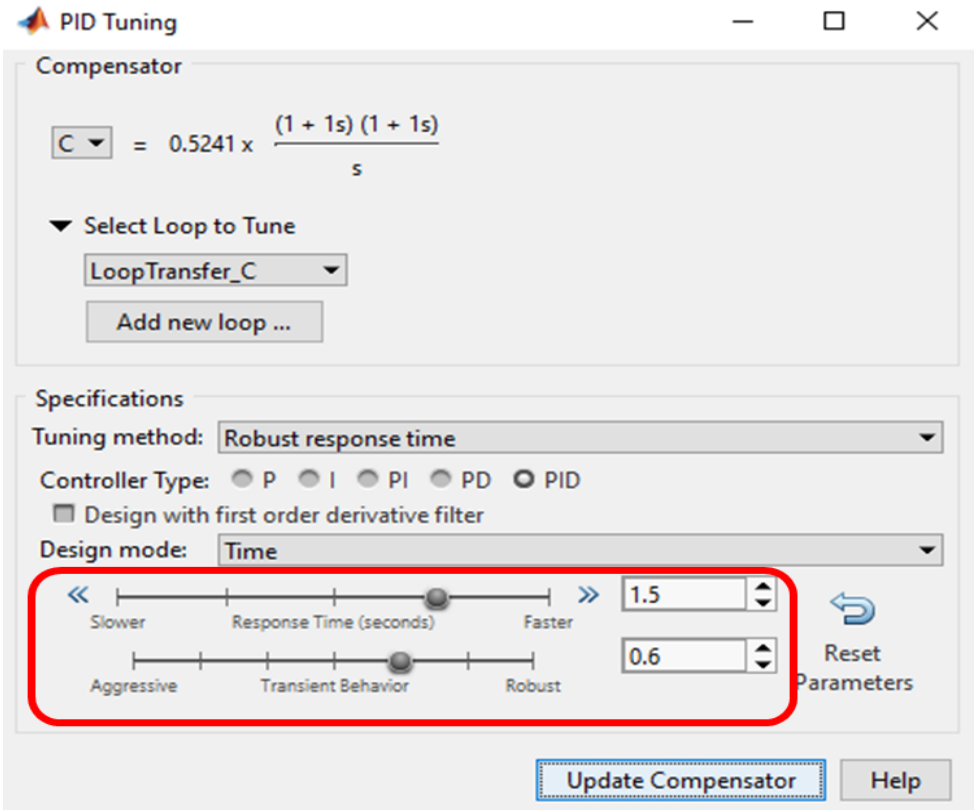


Ilustración 44 Ajuste manual controlador PID

Reducir el "Response Time (seconds)", es decir, moverlo hacia la derecha hará que la respuesta sea más rápida, mientras que mover la barra "Transient Behavior" hacia "Robust" ayudará a reducir la oscilación, El controlador PID resultante para los ajustes mostrados se proporciona a continuación:

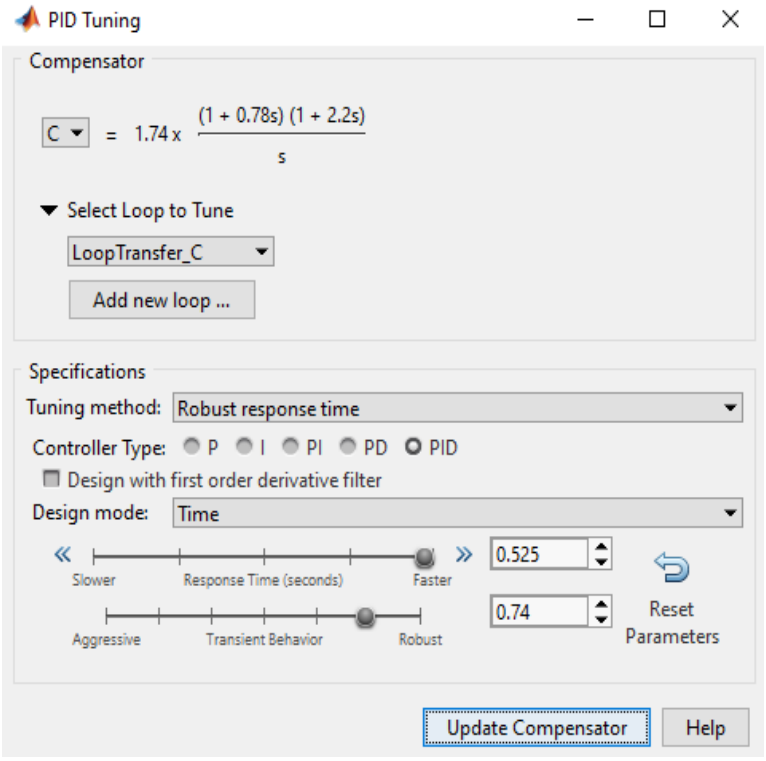


Ilustración 45 Ajuste manual del controlador final PID

La fórmula del controlador queda de la siguiente forma

$$C(s) = 1.2882\frac{(1+0.78s)(1+2.2s)}{s} \approx \frac{1.74}{s} + 5.1852 + 2.98s \quad (22)$$

Finalmente tenemos la respuesta gráfica de la siguiente manera:

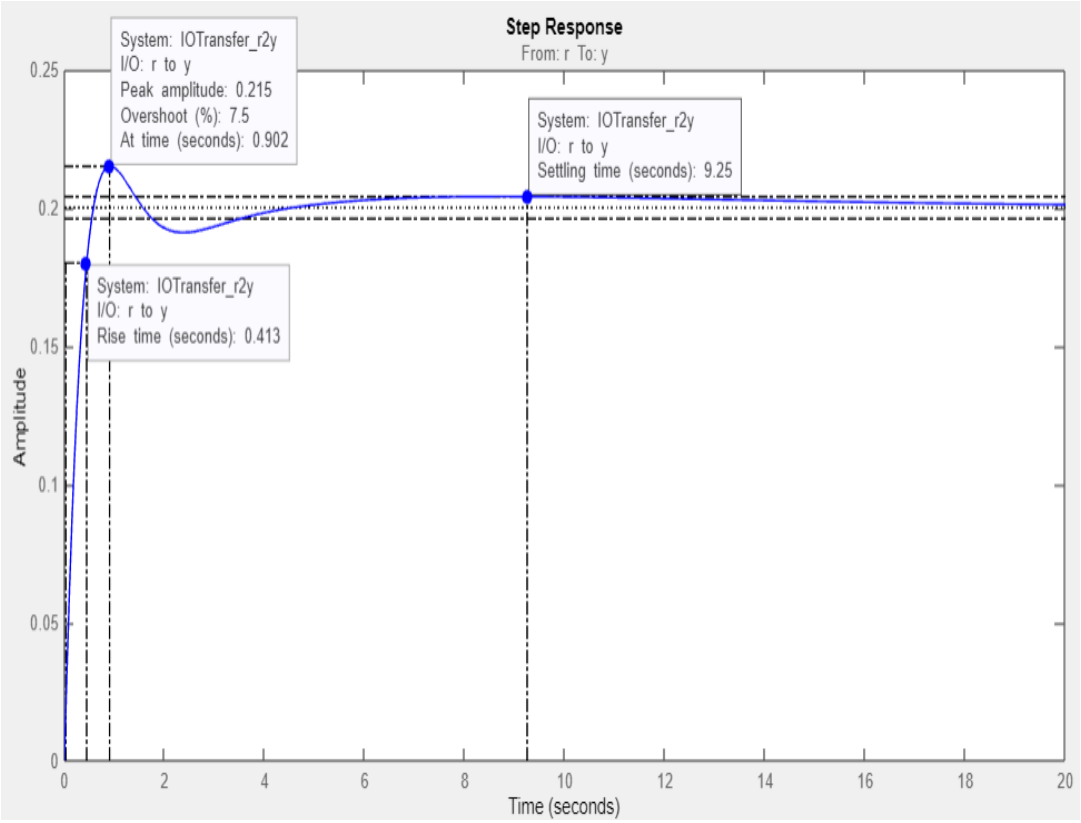


Ilustración 46 Comportamiento final con controlador PID

Con ello podemos evidenciar que finalmente el sistema es controlable cumpliendo con los requerimientos del sistema, se hará un resumen.

- Sobreimpulso = 7.5% < 10%
- Tiempo de subida = 0.413 segundos < 2 segundos
- Tiempo de estabilización = 9.25 segundos < 10 segundos
- Error de estado estacionario = 0% < 2%

Con ello, podemos definir que el controlador PID con las ganancias obtenidas proporcionará el rendimiento deseado al ángulo de elevación del avión.

4.2.11 ANÁLISIS DEL SISTEMA EN ENTORNO DE SIMULINK

El sistema se modelará en entorno de Simulink para ello es necesario tener en cuenta las fórmulas que gobiernan el movimiento del mismo, en este ejemplo simularemos el modelo del avión linealizando con el controlador de retroalimentación diseñado anteriormente, así entonces tenga en cuenta las

ecuaciones 11 y 12, además, estas ecuaciones coinciden con la forma de espacio de estado lineal general de la forma

$$\frac{dx}{dt} = Ax + Bu \quad (23)$$
$$y = Cx + Du \quad (24)$$

Construyendo el modelo de espacio en forma de diagrama de bloque de las ecuaciones anteriores 23 y 24. Una opción es construir un modelo de la planta con retroalimentación que se muestra a continuación:

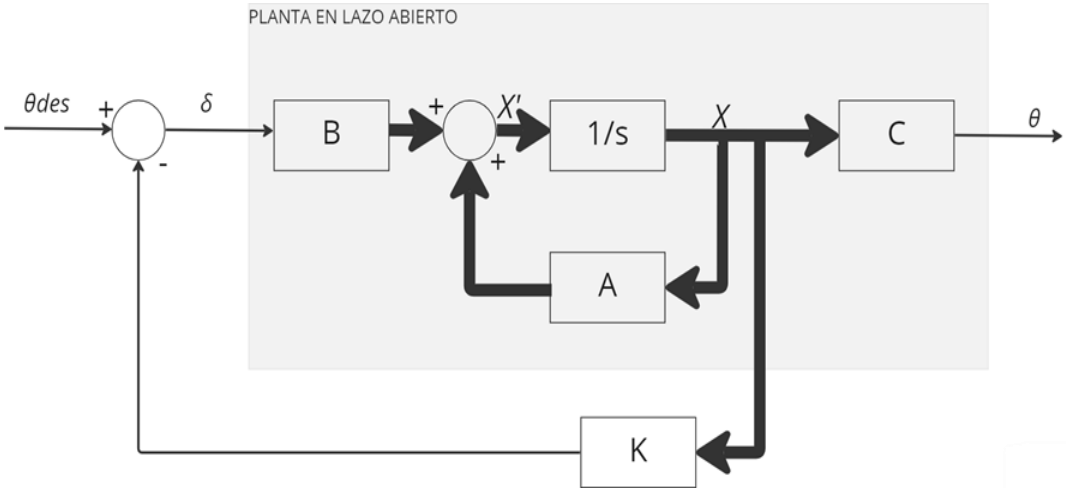


Ilustración 47 Esquema del sistema de la planta en lazo abierto

Se incluirá un bloque "State-Space" disponible en la biblioteca de Simulink, para modelar la planta en lazo abierto, para ello se realizará lo siguiente:

- Abrir un modelo de Simulink nuevo y en blanco.
- Inserte un bloque tipo Step.
- Configúrelo dando doble click en el con los siguientes datos "Step time=0" "Final Value=0.2" para representar los 0.2 rad de referencia que se han asumido a lo largo del ejercicio.
- Inserte un bloque "Demux" y configúrelo para que el número de salidas sea de tres, para las tres variables de estado.
- Inserte un Scope, las conexiones las verá representadas de manera gráfica al final.
- Agregue dos bloques Terminator.
- Inserte un bloque State-Space, configúrelo de tal forma que quede como se muestra a continuación:

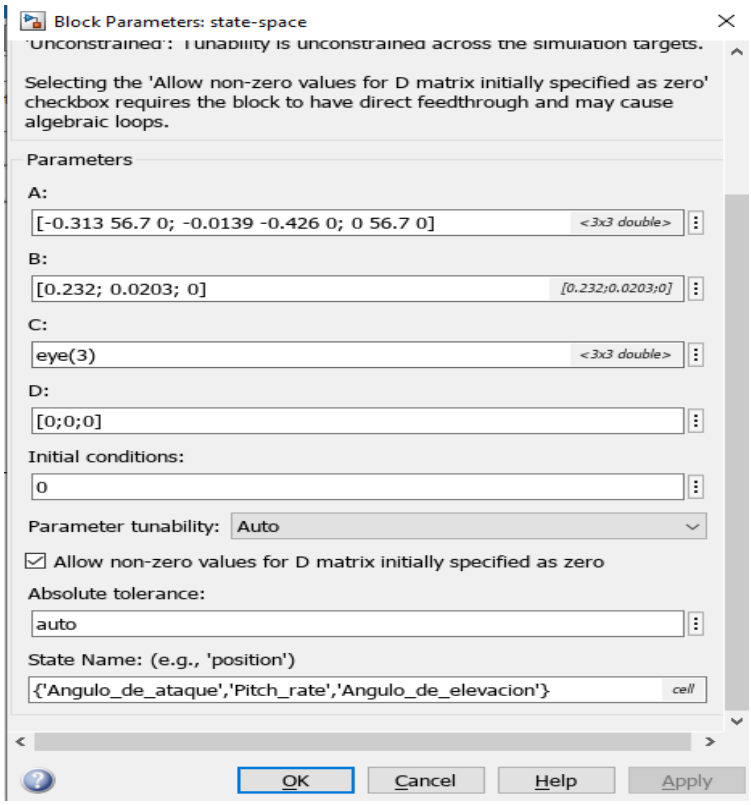


Ilustración 48 Parámetros de bloque State-Space

Tenga en cuenta que en la anterior figura (Configuración de State-Space) la matriz C se ingresa como una matriz de identidad de 3x3 usando el comando "eye" en lugar de [0 0 1]. Finalmente conecte todos los bloques de la siguiente manera:

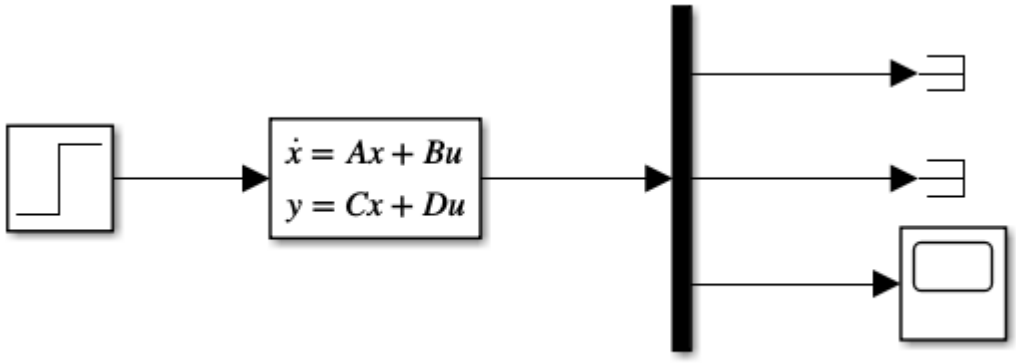


Ilustración 49 Diagrama de bloques con bloque State-Space

Se puede visualizar la respuesta por medio del Scope, así:

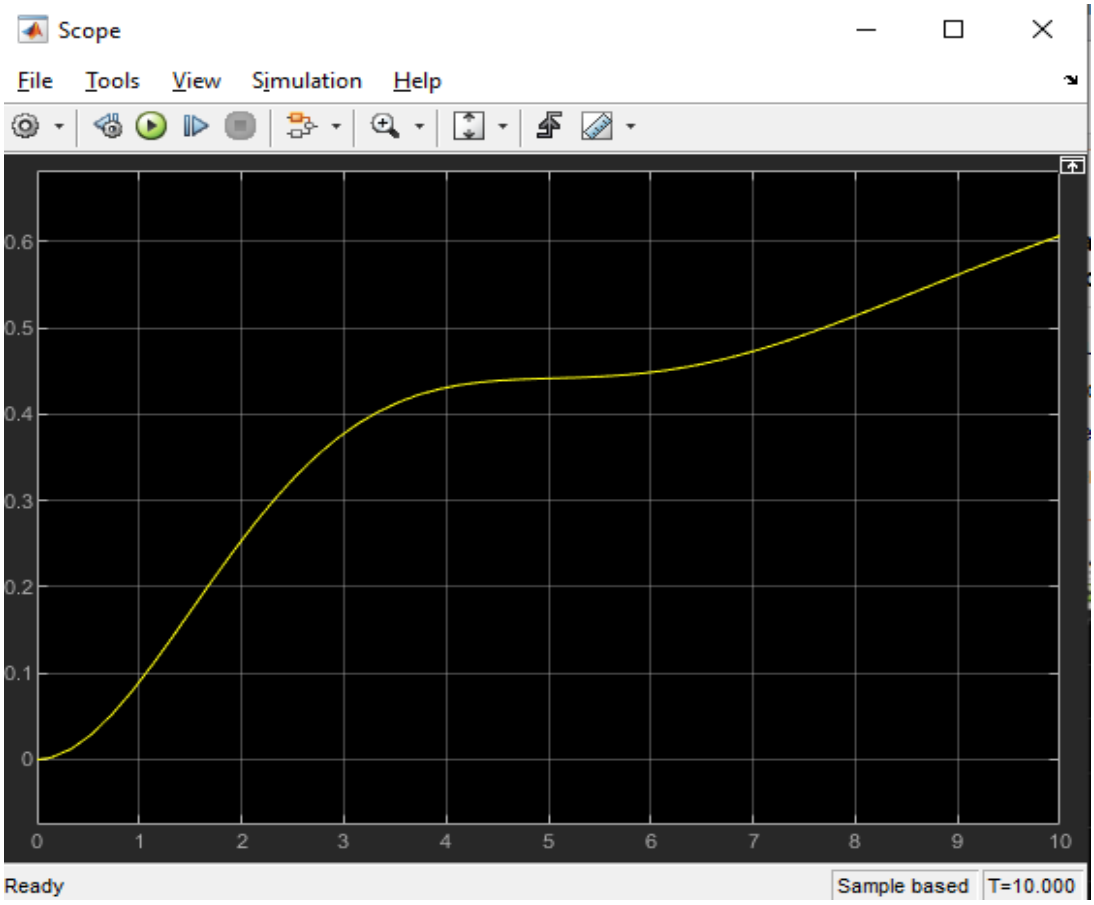


Ilustración 50 Respuesta del sistema de diagrama de bloques con State-Space

La respuesta generada en la gráfica es inestable, claramente no cumple con los requerimientos del sistema, está en lazo abierto y además es idéntica a la obtenida dentro del entorno de MATLAB. Para ver una respuesta estable, se agrega una ganancia de control de retroalimentación de estado K. Está ganancia se diseñó utilizando el método del regulador cuadrático lineal y dio como resultado lo siguiente

```
K=[-0.6435 169.6950 7.0711]; %DEFINE UNA VARIABLE%
```

Así entonces, tome como guía el diagrama de bloques anterior, agregue un bloque de suma y un bloque de ganancia, este último bloque configúrelo con la ganancia K ya definida en entorno MATLAB, como se muestra a continuación:

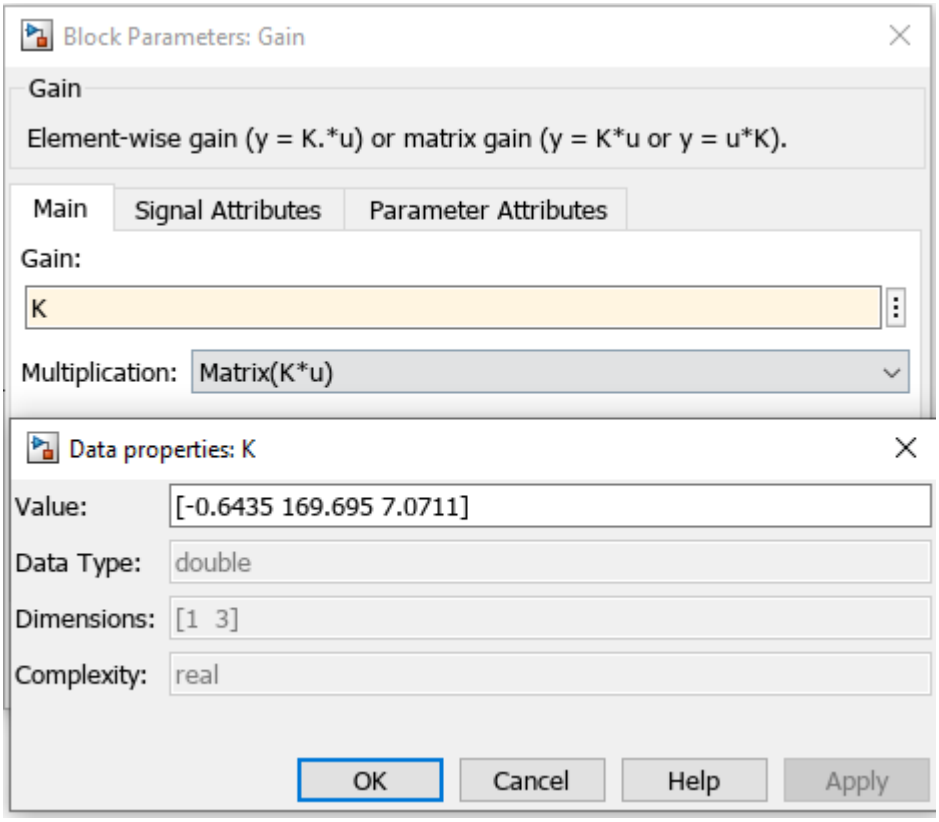


Ilustración 51 Parámetros bloque ganancia

El diagrama de bloques con las conexiones requeridas queda de la siguiente manera:

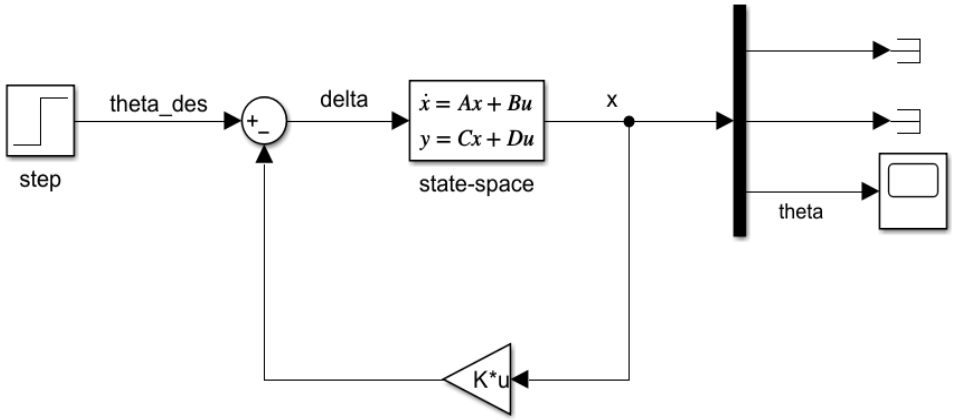


Ilustración 52 Diagrama de bloques con retroalimentación

Al ejecutar la simulación y dando Click en Scope para así poder visualizar el resultado, se puede observar:

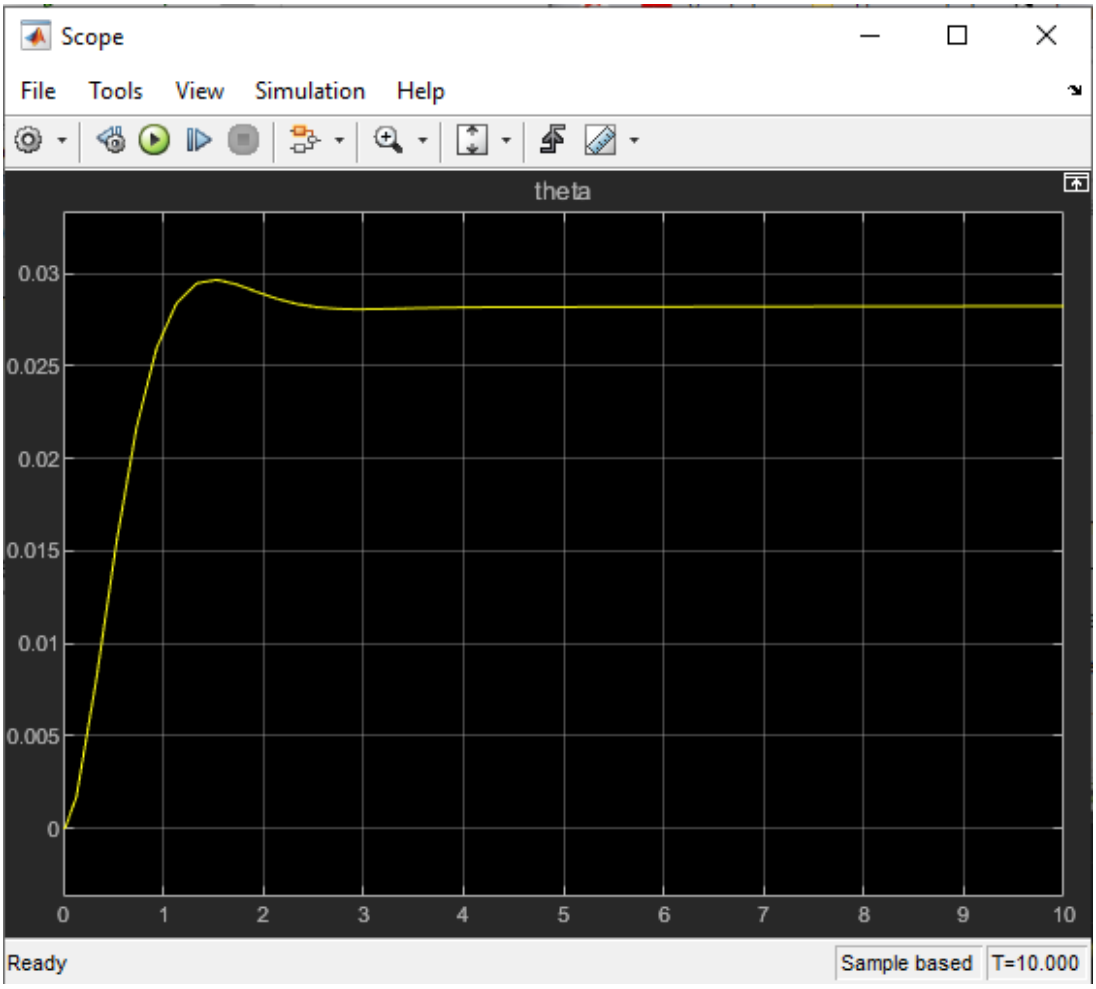


Ilustración 53 Respuesta diagrama de bloques con retroalimentación

Se evidencia que la respuesta ya está controlada y que además cumple con los requerimientos del sistema, el tiempo de elevación, tiempo de asentamiento y sobreimpulso, sin embargo el error de estado estable no se cumple ya que no se estabiliza dentro del 2% de la referencia tenida de 0,2 rads. Esta deficiencia en el sistema se soluciona agregando un precompensador constante de N=7.0711 para escalar la salida al nivel deseado. Valor obtenido debido al análisis de la regulación cuadrática lineal.

```
N=7.0711;
```

Este precompensador se puede implementar un bloque de ganancia al sistema de diagrama de bloques que se ha venido trabajando, así entonces la conexión es la siguiente:

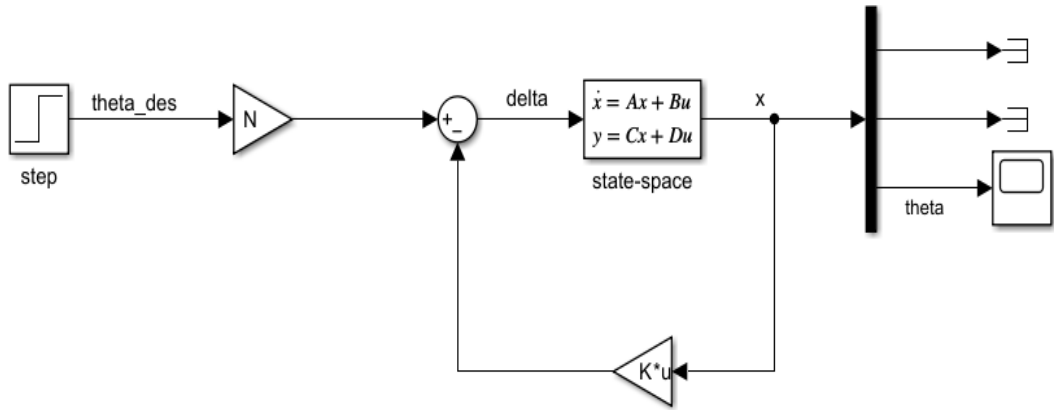


Ilustración 54 Diagrama de bloques con retroalimentación y bloque N

Fíjese en el bloque de ganancia N después de la entrada conectado al bloque de suma, el resultado obtenido por medio del Scope es el siguiente:

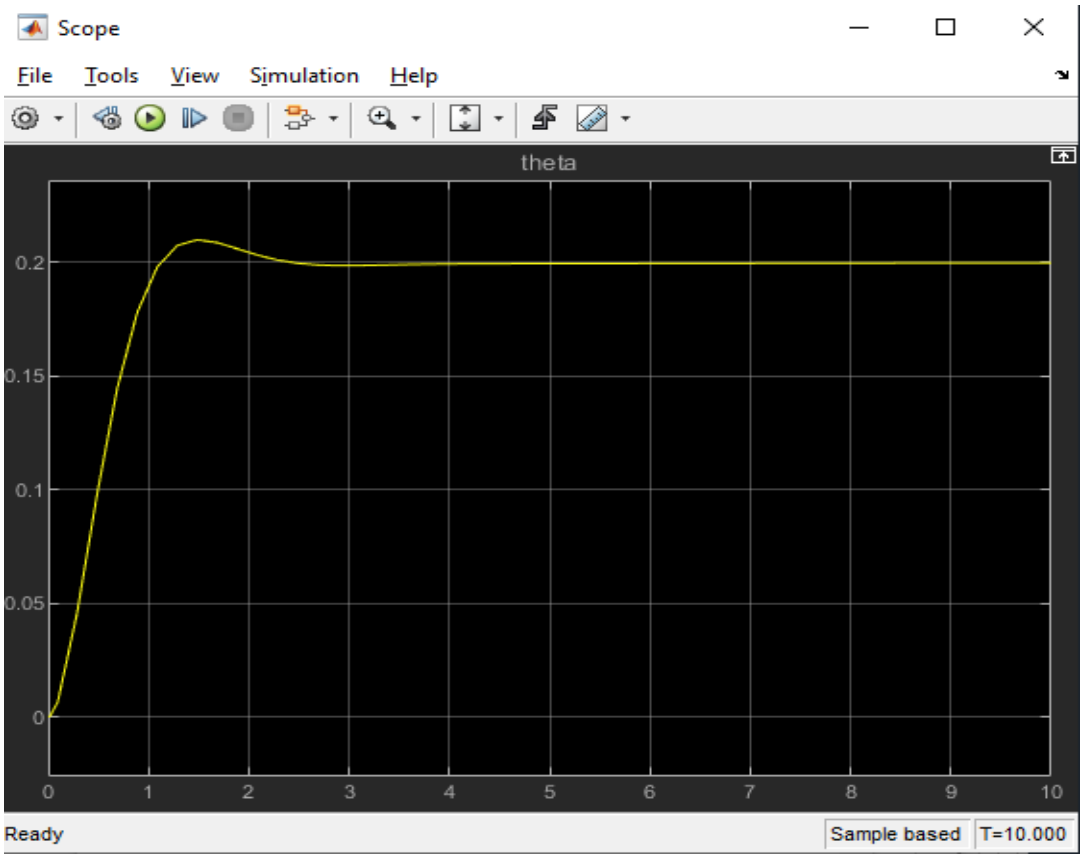


Ilustración 55 Respuesta diagrama de bloques con retroalimentación y bloque N

Con la adición de la ganancia del precompensador logró llevar el error de estado estable a cero y con esto cumplir en su totalidad los requerimientos del sistema, sin embargo, el inconveniente de usar este tipo de bloque como ganancia es que se calcula sobre la base de un modelo de la planta y está ubicado fuera del circuito de retroalimentación, de modo que la salida de la unión del bloque suma en el modelo anterior ya no es verdadera, esto influye en que si hay una perturbación desconocida, el bloque con la ganancia no los corregirá y se generará un error de estado estable.

Como ejemplo didáctico, se hará el ejemplo cuando una perturbación afecte al sistema. Esta perturbación se simulará añadiendo un bloque tipo "Step" con un "Value Final=0.2" y un "Step Time=3", esta perturbación se modela como si entrara en el sistema de misma forma como la entrada de control "delta". Así pues, el modelo del diagrama de bloques lo verá de la siguiente manera:

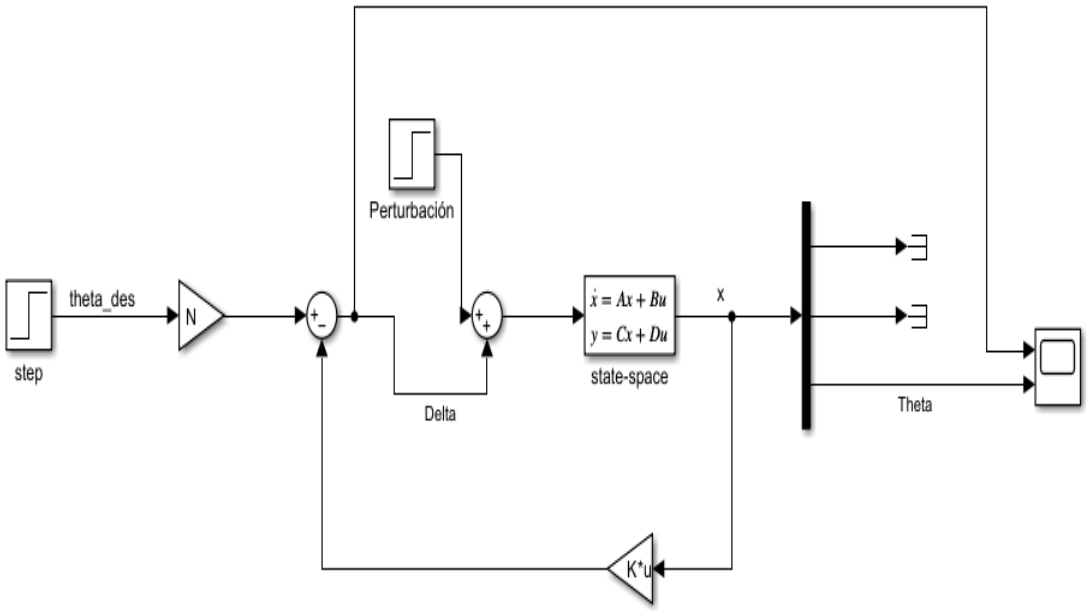


Ilustración 56 Diagrama de bloques agregando una perturbación

Cuando se ejecuta la simulación se visualiza el resultado gráficamente obtenemos lo siguiente:

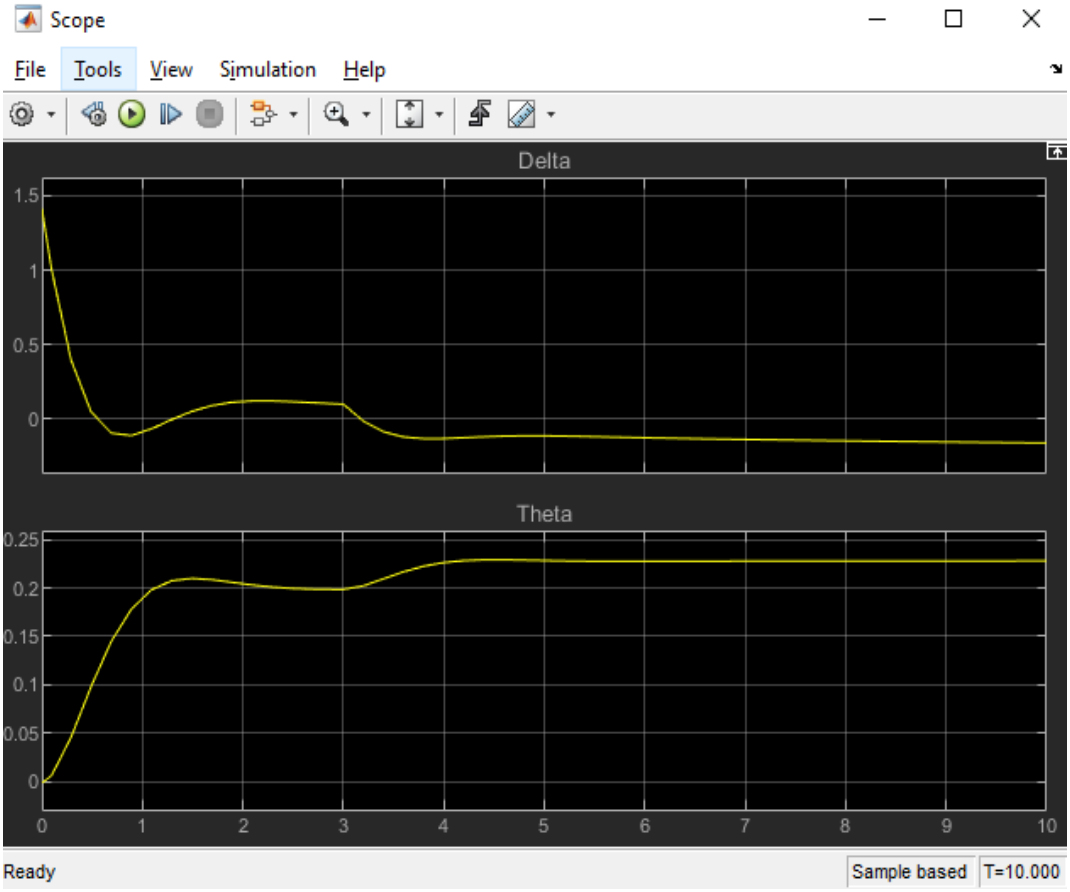


Ilustración 57 Respuesta diagrama de bloques agregando una perturbación

El análisis de la gráfica Theta entonces muestra que ante una perturbación en un tiempo de 3 segundos aleja al sistema del valor de estado estable de 0.2rad, exactamente a los 3 segundos se controla pero después el error aumenta nuevamente, el bloque con la ganancia K no puede corregir el efecto de esta perturbación. Se investiga la forma de controlar el sistema de otra forma.

Si el modelo del sistema actual tuviera una matriz B= [0.232*1.4 0.0203*0.6 0], con esto el controlador K usado anteriormente conduciría a un error de inestabilidad del sistema. Para ello se hace el ajuste con un PID automatizado con Simulink

4.2.12 AJUSTE CON PID AUTOMATIZADO EN SIMULINK

Agregar un controlador integral ayuda a reducir el error de estado estable que surge debido a las perturbaciones del sistema. Se implementará un

controlador PID asumiendo que Theta es medido a la salida, Además se utilizarán las prestaciones de Simulink para ajustar automáticamente.

Para ajustar un bloque PID realice lo siguiente:

- Tenga en cuenta que se va a seguir manejando el modelo de Simulink con el que se ha estado trabajando. Para ello, elimine los bloques de las ganancias K y N. Elimine la señal de retroalimentación del bloque State-Space y conéctela nuevamente a la salida theta.
- Inserte un bloque de controlador PID desde la biblioteca y colóquelo después del primer bloque de suma con una retroalimentación negativa. El modelo del sistema queda representado de la siguiente manera:

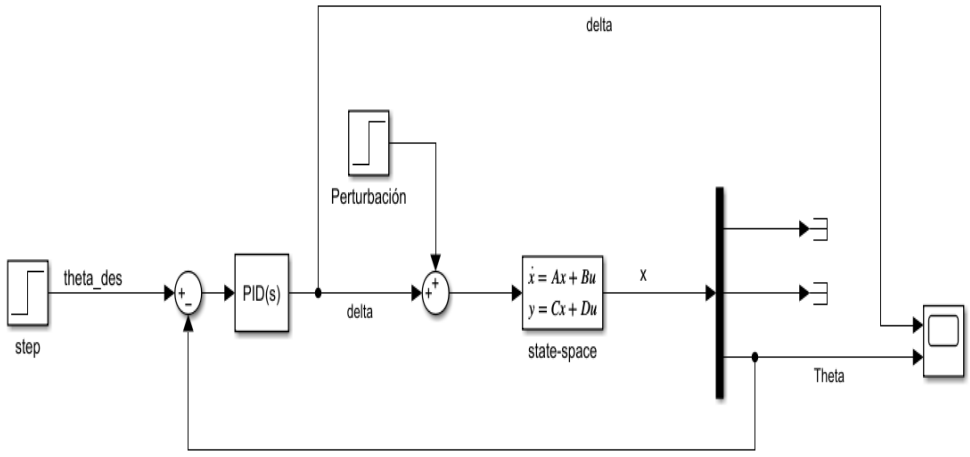


Ilustración 58 Diagrama de bloques con perturbación y bloque PID

- Los parámetros del bloque PID se dejarán los que están por defecto, sin embargo, no van hacer con las que se simule el programa, en los parámetros del bloque se realizará la sintonización automática o Tune, dando click precisamente en este botón. El resultado obtenido será el siguiente:

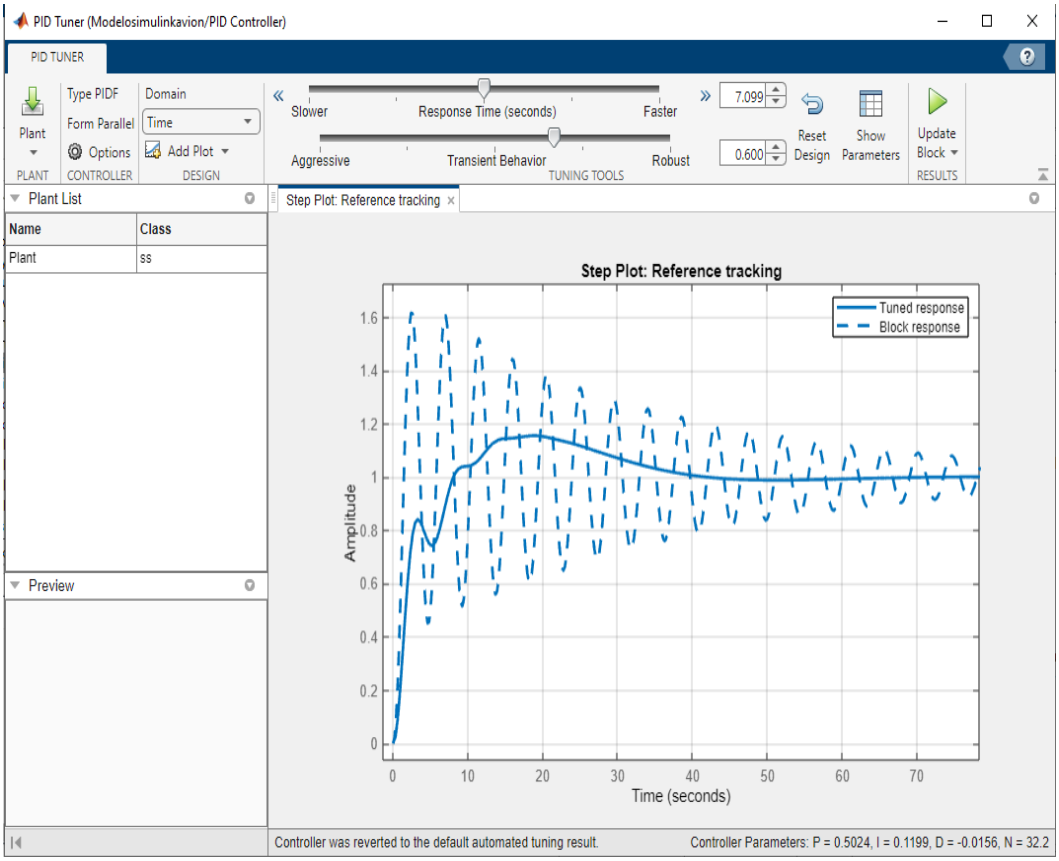


Ilustración 59 Respuesta diagrama de bloques con perturbación y bloque PID

Inicialmente, la respuesta del sistema con las ganancias actuales del controlador PID son las que se detallan en una línea azul punteada y la respuesta del sistema con las ganancias ajustadas propuestas se detallan con la línea azul continua. Con esto entonces actualice el bloque con las ganancias generadas por la simulación. La grafica generada después de actualizar el bloque es la siguiente:

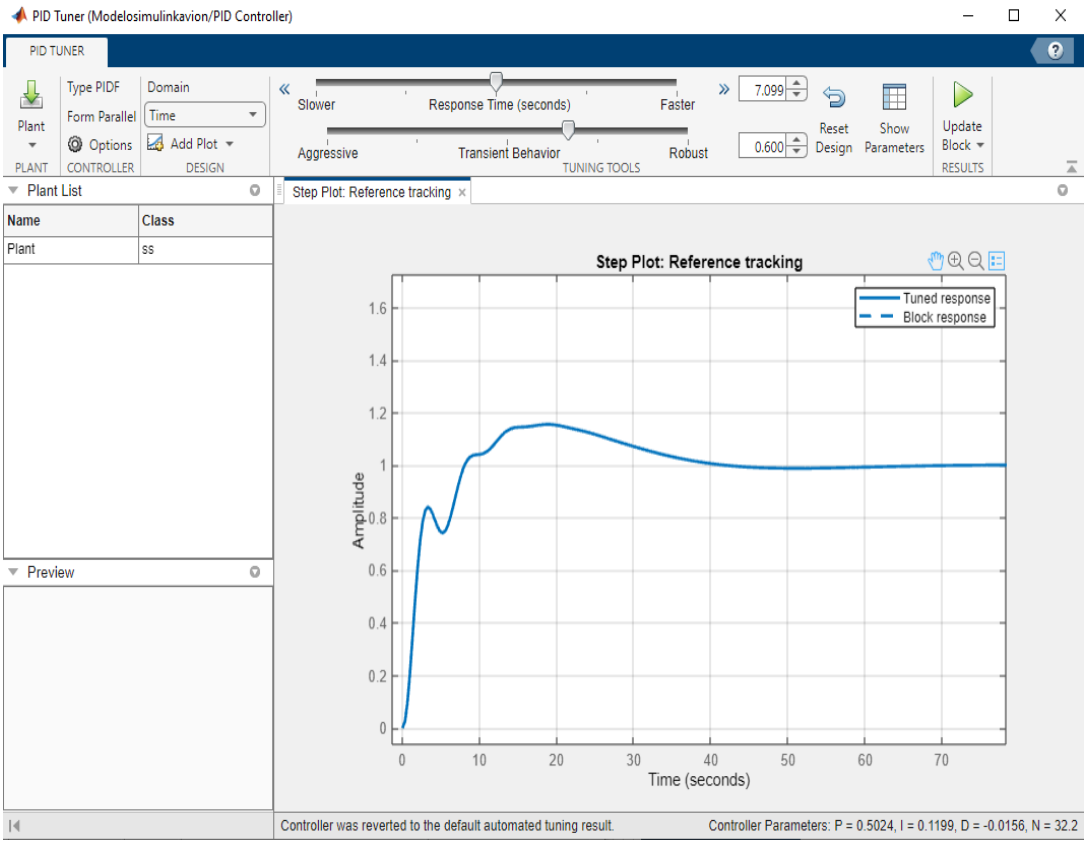


Ilustración 60 Respuesta generada por controlador PID Tuning

Sin embargo, aunque estabiliza bastante la señal, no cumple con los requerimientos del sistema, se tiene que hacer la sintonización manual y así ajustar la gráfica hasta que se cumplan los requerimientos, este ajuste manual se logra desplazando las barras ubicadas en TUNING TOOLS, nos ubicamos en "Response Time". En general, se tiene que hacer que el sistema sea lo suficientemente rápido para cumplir con los requerimientos del sistema.

El ajuste en un tiempo de respuesta de 0.0894 segundos cumple con los requerimientos del sistema, con ello también se cumple el objetivo que el tiempo de estabilización se cumpla incluso cuando exista una perturbación, la configuración es la siguiente:

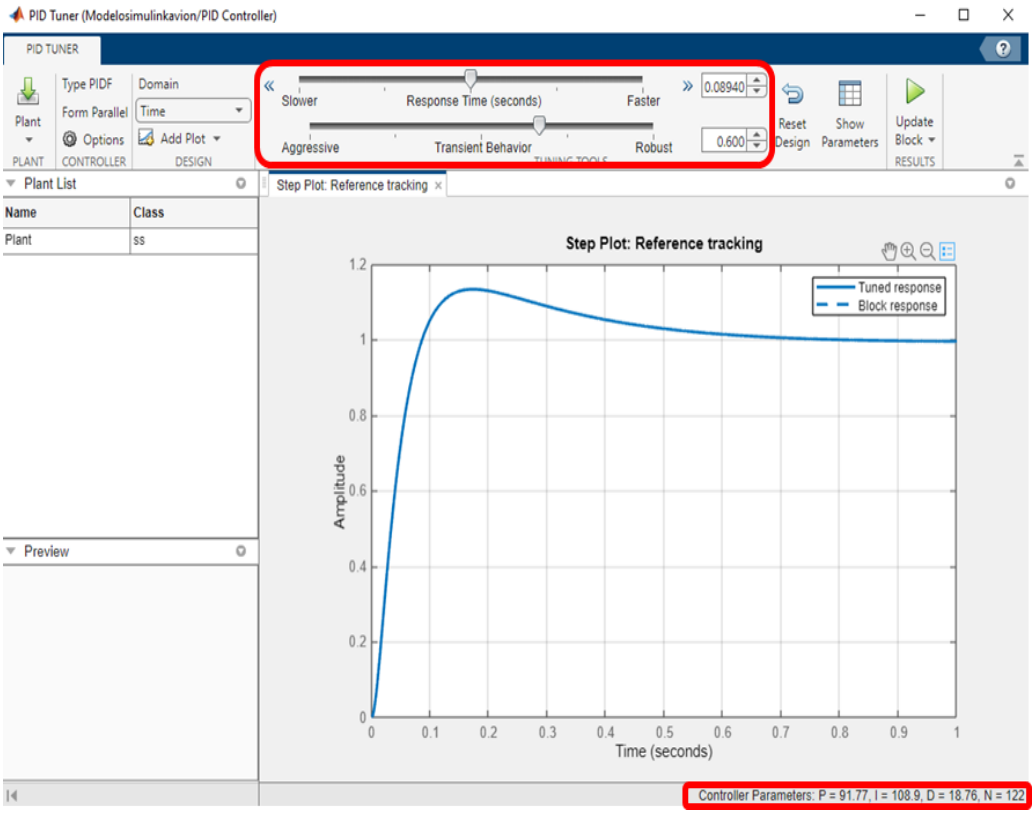


Ilustración 61 Ajuste manual PID Tuning

Además, las ganancias obtenidas se pueden visualizar en la parte inferior izquierda de la gráfica, y así ver el rendimiento del controlador. Actualice nuevamente el bloque, ejecute la simulación en entorno de Simulink y

visualice por medio del Scope. La respuesta generada debe ser la siguiente:

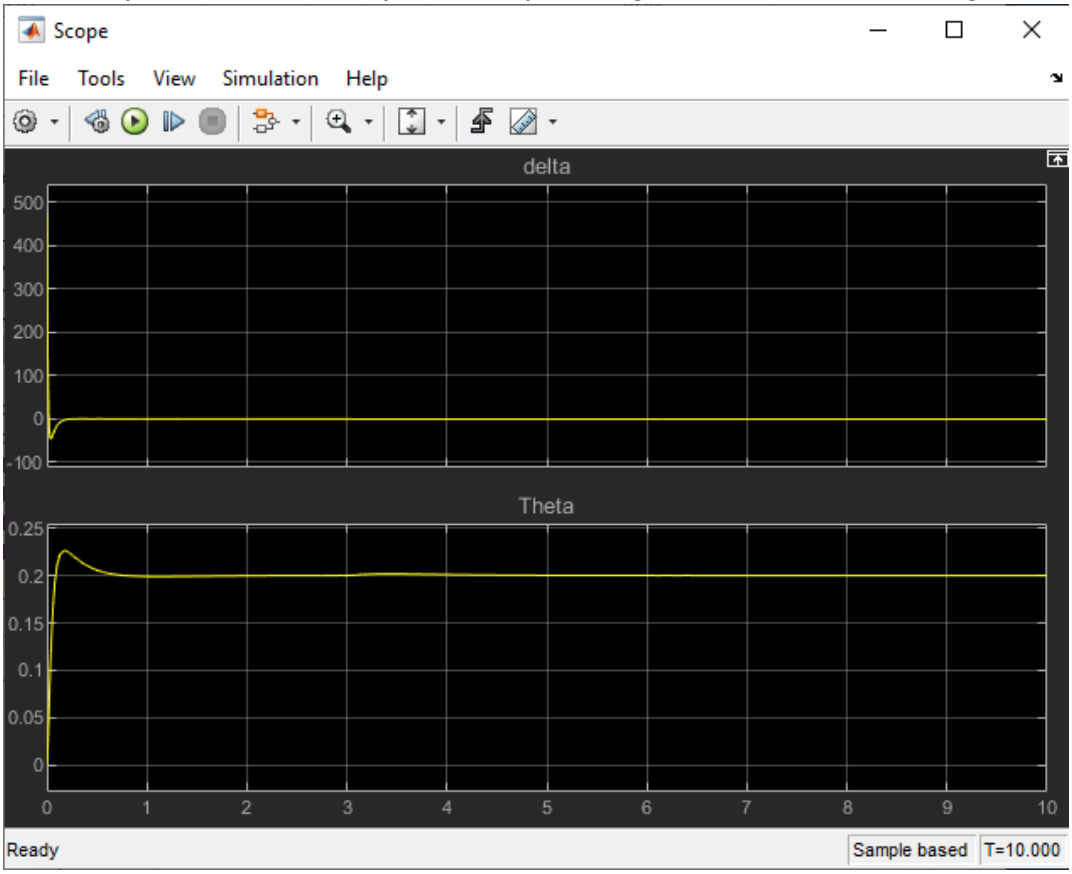


Ilustración 62 Respuesta del ajuste manual PID Tuning

Teniendo en cuenta las gráficas obtenidas se destaca que la perturbación en estado estable se corrige, esto debido al controlador PID que se está usando y que incluye un término integral. Se puede evidenciar en el resultado de Theta.

En la práctica, es probable que el ángulo de elevación delta del avión se limite como +/-25° en radianes vendría siendo +/-0.4364, para no ir más allá de este límite, se puede configurar el bloque, esto se configura en la ventana "Saturation" activando "Output saturation" y definir los limites como se muestra a continuación:

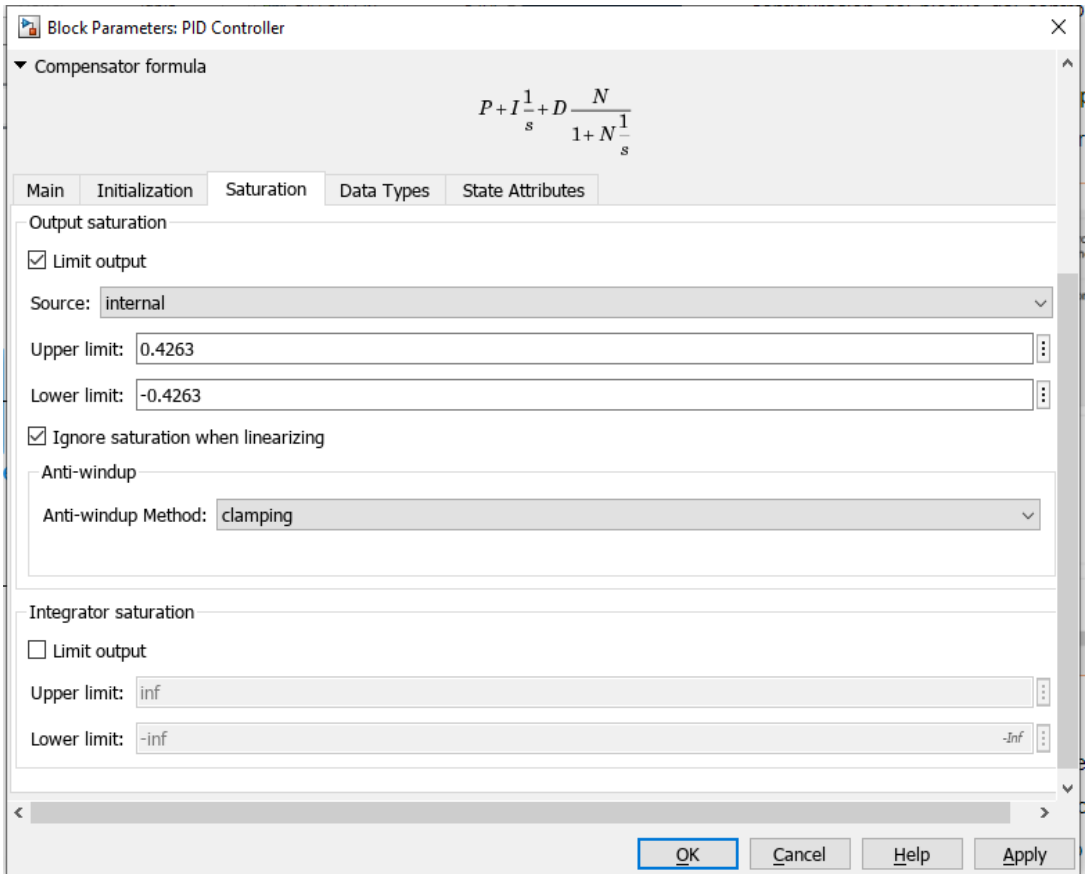


Ilustración 63 Parámetros bloque PID saturación

Además, tiene que habilitarle la opción anti-windup y escoger el método clamping, de no hacerlo surge un problema con el control PID y es que el término integrador puede continuar integrando el error, esencialmente solicitando un control cada vez mayor, incluso cuando el esfuerzo del control se ha saturado, el problema es que el error eventualmente comienza a disminuir, el esfuerzo del control seguirá estando saturado debido a que el integrador ha acumulado un valor grande.

Esto hace que el sistema sea lento porque el integrador puede tardar mucho tiempo en ejecutarse. Esta estrategia evita que el integrador cuando integre no se sature. Tenemos el siguiente comportamiento:

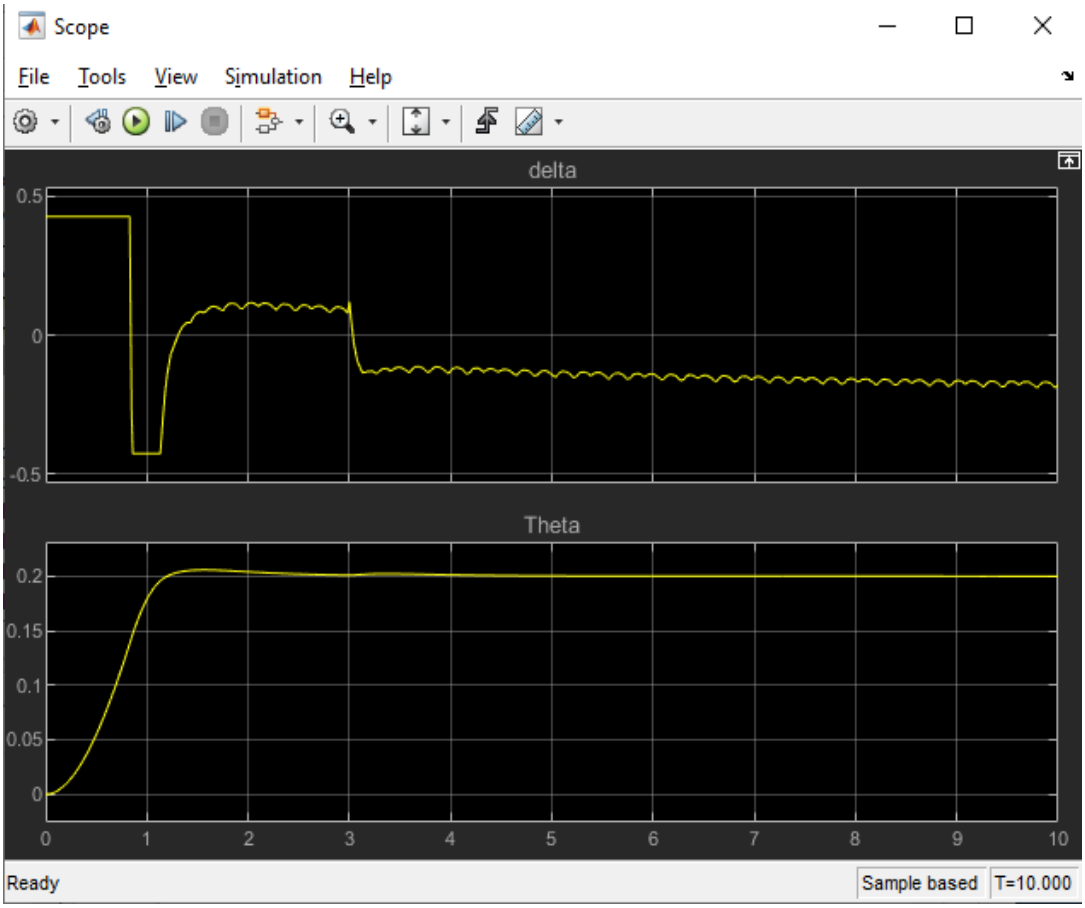


Ilustración 64 Respuesta del sistema con bloque PID limitado en la saturación

La salida Theta es la respuesta deseada y la que cumple con los requerimientos del sistema.

Con esto se concluye que el control de movimiento de un avión es un proceso crucial y que implica una serie casi que infinitas de sistemas y tecnologías diseñadas para garantizar siempre el bienestar de todos los pasajeros y aunque en este ejercicio no se tuvieron en cuenta muchas variables aun así no deja de serlo.

El objetivo principal era que el avión volara de manera segura y estable durante todo el trayecto y si había alguna perturbación este tuviera la capacidad en el menor tiempo posible de volver a las condiciones ideales de vuelo. El desarrollo del ejercicio cumplió a cabalidad lo estipulado desde el comienzo del ejercicio.

4.3 CONTROL DE EQUILIBRIO A UN PÉNDULO INVERTIDO

El sistema de este ejemplo consta de un péndulo invertido montado en un carro motorizado. El sistema de péndulo invertido es un ejemplo que se encuentra comúnmente en los libros de texto de sistemas de control y en la literatura de investigación. Su popularidad se debe en parte al hecho de que es inestable sin control, es decir, el péndulo simplemente se caerá si no se mueve el carro para equilibrarlo. Además, la dinámica del sistema es no lineal. El objetivo del sistema de control es equilibrar el péndulo invertido aplicando una fuerza al carro al que está sujeto el péndulo. En este caso, consideraremos un problema bidimensional en el que el péndulo está obligado a moverse en el plano vertical que se muestra en la figura siguiente.

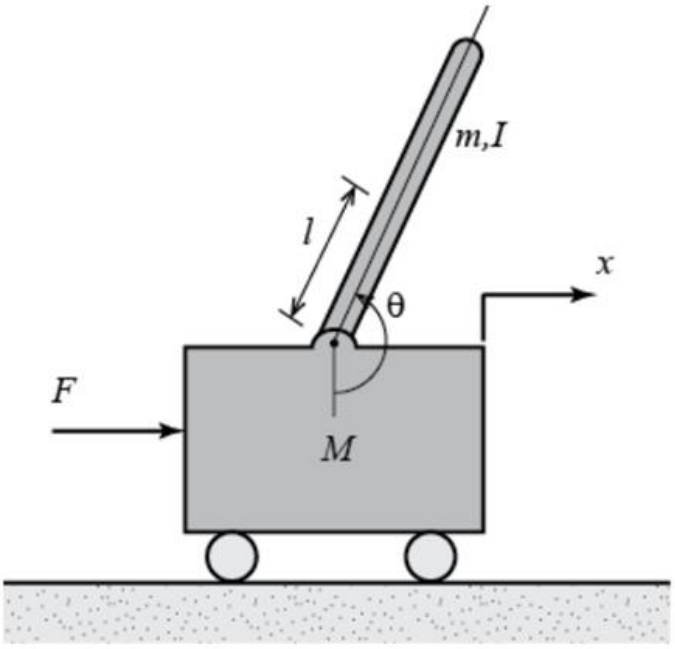


Ilustración 65 Modelo péndulo invertido, imagen tomada y adaptada de https://ctms.engin.umich.edu/CTMS/Content/InvertedPendulum/System/Modeling/figures/pendulum.png

Para este sistema, la entrada de control es la fuerza *F* que mueve el carro horizontalmente y las salidas son la posición angular del péndulo *Theta* y la posición horizontal del carro *x.*

Para este ejemplo, supongamos las siguientes cantidades:

(M) masa del carro 0,5 kg

(m) masa del péndulo 0,2 kg

(b) coeficiente de fricción para carro 0.1 N/m/seg

(l) longitud al centro de masa del péndulo 0,3 m

(I) Momento de inercia de la masa del péndulo 0.006 kg.m^2

(F) fuerza aplicada al carro

(x) coordenada de posición del carro

(Theta) ángulo del péndulo desde la vertical (hacia abajo)

Para las secciones de PID, lugar geométrico de las raíces y respuesta de frecuencia de este problema, sólo nos interesará el control de la posición del péndulo. Esto se debe a que las técnicas utilizadas en estas secciones son las más adecuadas para sistemas de entrada única, salida única (SISO). Por lo tanto, ninguno de los criterios de diseño tiene que ver con la posición del carro. Sin embargo, investigaremos el efecto del controlador en la posición del carro después de que se haya diseñado el controlador. Para estas secciones, diseñaremos un controlador para restaurar el péndulo a una posición vertical hacia arriba después de que haya experimentado un "golpe" impulsivo en el carro. Específicamente, los criterios de diseño son que el péndulo regrese a su posición vertical en 5 segundos y que el péndulo nunca se mueva más de 0. 05 radianes de distancia de la vertical después de ser perturbado por un impulso de magnitud 1 Nseg. El péndulo comenzará inicialmente en el equilibrio vertical ascendente, *Theta* = *pi*.

En resumen, los requisitos de diseño para este sistema son:

- Tiempo de asentamiento de menos de *Theta* = 5 segundos.
- Ángulo del péndulo *Theta* no menor de 0,05 radianes de la vertical

Empleando técnicas de diseño de espacio de estado, somos más capaces de abordar un sistema de múltiples salidas. En nuestro caso, el sistema de péndulo invertido es de entrada única, salida múltiple (SIMO). Por lo tanto, para la sección de espacio de estado del ejemplo del péndulo invertido, intentaremos controlar tanto el ángulo del péndulo como la posición del carro. Para hacer que el diseño sea más desafiante en esta sección, ordenaremos un paso de 0,2 metros en la posición deseada del carro. En estas condiciones, se desea que el carro alcance su posición ordenada en 5 segundos y tenga un

tiempo de subida inferior a 0,5 segundos. También se desea que el péndulo se asiente en su posición vertical en menos de 5 segundos y, además, que el ángulo del péndulo no se desplace más de 20 grados.

En resumen, los requisitos de diseño para el ejemplo de espacio de estado del péndulo invertido son:

- Tiempo de establecimiento para *x* y *Theta* de menos de 5 segundos
- Tiempo de subida para *x* de menos de 0,5 segundos
- Ángulo del péndulo *Theta* no menor de 20 grados (0,35 radianes) de la vertical
- Error de estado estacionario de menos del 2% para *x* y *Theta*

4.3.1 ANÁLISIS DE FUERZAS Y ECUACIONES DEL SISTEMA

A continuación se muestran los diagramas de cuerpo libre de los dos elementos del sistema de péndulo invertido.

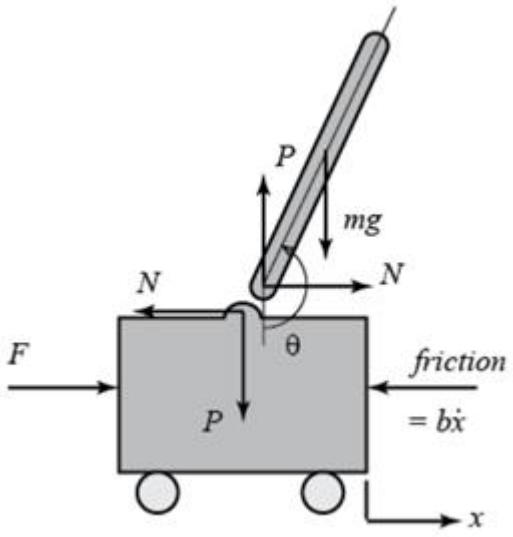


Ilustración 66 DCL para péndulo invertido, imagen tomada y adaptada de https://ctms.engin.umich.edu/CTMS/Content/InvertedPendulum/System/Modeling/figures/pendulum2.png

Sumando las fuerzas en el diagrama de cuerpo libre del carro en la dirección horizontal, se obtiene la siguiente ecuación de movimiento.

$$M\ddot{x} + b\dot{x} + N = F \qquad (1)$$

Tenga en cuenta que también puede sumar las fuerzas en la dirección vertical del carro, pero no se obtendrá información útil.

Sumando las fuerzas en el diagrama de cuerpo libre del péndulo en la dirección horizontal, se obtiene la siguiente expresión para la fuerza de reacción *N*.

$$N = m\ddot{x} + ml\ddot{\theta}cos\theta - ml\dot{\theta}^2 sen\theta \qquad (2)$$

Si sustituye esta ecuación en la primera ecuación, obtiene una de las dos ecuaciones gobernantes para este sistema

$$(M + m)\ddot{x} + b\dot{x} + ml\ddot{\theta}cos\theta - ml\dot{\theta}^2 sen\theta = F \qquad (3)$$

Para obtener la segunda ecuación de movimiento de este sistema, suma las fuerzas perpendiculares al péndulo. Resolver el sistema a lo largo de este eje simplifica enormemente las matemáticas. Deberías obtener la siguiente ecuación

$$Psen\theta + Ncos\theta - mgsen\theta = ml\ddot{\theta} + m\ddot{x}cos\theta \qquad (4)$$

Para deshacerse de los términos *P* y *N* en la ecuación anterior, sume los momentos sobre el centroide del péndulo para obtener la siguiente ecuación

$$-Plsen\theta - Nlcos\theta = I\ddot{\theta} \qquad (5)$$

Combinando estas dos últimas expresiones, se obtiene la segunda ecuación gobernante.

$$(I + ml^2)\ddot{\theta} + mlsen\theta = -ml\ddot{x}cos\theta \qquad (6)$$

Dado que las técnicas de análisis y diseño de control que emplearemos en este ejemplo se aplican solo a sistemas lineales, este conjunto de ecuaciones debe linealizarse. Específicamente, se lineal izará las ecuaciones sobre la posición de equilibrio verticalmente hacia arriba, *Theta = pi*, y supondremos que el sistema permanece dentro de una pequeña vecindad de este equilibrio. Esta suposición debería ser razonablemente válida ya que bajo control deseamos que el péndulo no se desvíe más de 20 grados de la posición vertical hacia arriba. Sea *Phi* la desviación de la posición del péndulo del equilibrio, es decir, *Theta = pi+ Phi*. De nuevo suponiendo una pequeña desviación (*Phi*) del equilibrio, podemos usar las siguientes aproximaciones de ángulo pequeño de las funciones no lineales en las ecuaciones de nuestro sistema:

$$cos\theta = \cos(\pi + \varphi) \approx -1 \qquad (7)$$

$$sen\theta = sen(\pi + \varphi) \approx -\varphi \qquad (8)$$

$$\dot{\theta}^2 = \dot{\varphi}^2 \approx 0 \qquad (9)$$

Después de sustituir las aproximaciones anteriores en nuestras ecuaciones gobernantes no lineales, llegamos a las dos ecuaciones de movimiento linealizadas. Nótese que *u* ha sido sustituida por la entrada *F*.

$$\left(l + ml^2\right)\varphi - mgl\ddot{\varphi} = ml\ddot{x} \qquad (10)$$

$$(M + m)\ddot{x} + b\dot{x} - ml\ddot{\varphi} = u \qquad (11)$$

4.3.2 REPRESENTACIÓN POR FUNCIONES DE TRANSFERENCIA

Para obtener las funciones de transferencia de las ecuaciones del sistema linealizadas, primero debemos tomar la transformada de Laplace de las ecuaciones del sistema asumiendo condiciones iniciales cero. Las transformadas de Laplace resultantes se muestran a continuación.

$$(l + ml^2)\Phi(s)s^2 - mgl\Phi(s) = mlX(s)s^2 \qquad (12)$$

$$(M + m)X(s)s^2 + bX(s)s - ml\Phi(s)s^2 = U(s) \qquad (13)$$

Recuerde que una función de transferencia representa la relación entre una sola entrada y una sola salida a la vez. Para encontrar nuestra primera función de transferencia para la salida *Phi(s)* y una entrada *U(s)* de necesitamos eliminar *X(s)* de las ecuaciones anteriores. Resuelve la primera ecuación para *X(s)*.

$$X(s) = \left[\frac{l + ml^2}{ml} - \frac{g}{s^2}\right]\Phi(s) \qquad (14)$$

Luego sustituya lo anterior en la segunda ecuación

$$(M + m)\left[\frac{l + ml^2}{ml} - \frac{g}{s^2}\right]\Phi(s)s^2 + b\left[\frac{l + ml^2}{ml} - \frac{g}{s^2}\right]\Phi(s)s - ml\Phi(s)s^2 = U(s) \qquad (15)$$

Reordenando, la función de transferencia es entonces la siguiente

$$\frac{\Phi(s)}{U(s)} = \frac{\dfrac{ml}{[(M+m)(I+ml^2)-(ml)^2]}s^2}{s^4 + \dfrac{b(I+ml^2)}{[(M+m)(I+ml^2)-(ml)^2]}s^3 - \dfrac{(M+m)mgl}{[(M+m)(I+ml^2)-(ml)^2]}s^2 - \dfrac{bmgl}{[(M+m)(I+ml^2)-(ml)^2]}s} \quad (16)$$

De la función de transferencia anterior se puede ver que hay un polo y un cero en el origen. Estos se pueden cancelar y la función de transferencia se convierte en la siguiente.

$$\frac{\Phi(s)}{U(s)} = \frac{\dfrac{ml}{[(M+m)(I+ml^2)-(ml)^2]}s}{s^3 + \dfrac{b(I+ml^2)}{[(M+m)(I+ml^2)-(ml)^2]}s^2 - \dfrac{(M+m)mgl}{[(M+m)(I+ml^2)-(ml)^2]}s - \dfrac{bmgl}{[(M+m)(I+ml^2)-(ml)^2]}} \quad (17)$$

En segundo lugar, la función de transferencia con la posición del carrito *X(s)* como salida se puede derivar de manera similar para llegar a lo siguiente.

$$\frac{X(s)}{U(s)} = \frac{\dfrac{(l+ml^2)s^2 - gml}{[(M+m)(I+ml^2)-(ml)^2]}}{s^4 + \dfrac{b(I+ml^2)}{[(M+m)(I+ml^2)-(ml)^2]}s^3 - \dfrac{(M+m)mgl}{[(M+m)(I+ml^2)-(ml)^2]}s^2 - \dfrac{bmgl}{[(M+m)(I+ml^2)-(ml)^2]}s} \quad (18)$$

4.3.3 REPRESENTACIÓN EN MATLAB DE LAS FUNCIONES DE TRANSFERENCIA

Podemos representar las funciones de transferencia derivadas anteriormente para el sistema de péndulo invertido dentro de MATLAB empleando los siguientes comandos. Tenga en cuenta que puede dar nombres a las salidas (y entradas) para diferenciar entre la posición del carro y la posición del péndulo. Ejecutar este código en la ventana de comandos produce el resultado que se muestra a continuación.

```
M=0.5; %Masa del carro en Kg%
m=0.2; %Masa del péndulo en Kg%
b=0.1; %Coeficiente de fricción del carro en Nm/s%
I=0.006; %Momento de inercia del péndulo en Kg/m^2%
g=9.8; %Gravedad en m/s^2%
l=0.3; %Longitud del péndulo en m%
q=(M+m)*(I+m*l^2)-(m*l)^2; %Definición de variable q%
s=tf('s'); %Definición función transferencia%
```

```
 P_cart=(((I+m*l^2)/q)*s^2 - (m*g*l/q))/(s^4 + (b*(I + m*l^2))*s^3/q -
((M + m)*m*g*l)*s^2/q - b*m*g*l*s/q); %Definición de variable P_cart%

 P_pend=(m*l*s/q)/(s^3 + (b*(I + m*l^2))*s^2/q - ((M + m)*m*g*l)*s/q -
b*m*g*l/q); %Definición de variable P_pend%

 sys_tf=[P_cart ; P_pend]; %Función transferencia para variables
definidas P_cart y P_pend%

 entradas={'u'}; %Definición de entradas%
 salidas={'x'; 'phi'}; %Definición de salidas%

 set(sys_tf,'InputName',entradas) %Proporcionar propiedades a las
variables%
 set(sys_tf,'OutputName',salidas) %Proporcionar propiedades a las
variables%

 sys_tf
```

```
sys_tf =

  From input "u" to output...
                      4.182e-06 s^2 - 0.0001025
   x:  ---------------------------------------------------------
       2.3e-06 s^4 + 4.182e-07 s^3 - 7.172e-05 s^2 - 1.025e-05 s

                              1.045e-05 s
   Phi:  ---------------------------------------------------
         2.3e-06 s^3 + 4.182e-07 s^2 - 7.172e-05 s - 1.025e-05

Continuous-time transfer function.
```

4.3.4 REPRESENTACIÓN EN EL ESPACIO DE ESTADOS

Las ecuaciones de movimiento linealizadas de arriba también se pueden representar en forma de espacio de estado si se reorganizan en una serie de ecuaciones diferenciales de primer orden. Dado que las ecuaciones son lineales, se pueden poner en la forma matricial estándar que se muestra a continuación.

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(I+ml^2)b}{I(M+m)+Ml^2} & \frac{m^2gl^2}{I(M+m)+Ml^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-mlb}{I(M+m)+Ml^2} & \frac{mgl(M+m)}{I(M+m)+Ml^2} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{I+ml^2}{I(M+m)+Ml^2} \\ 0 \\ \frac{ml}{I(M+m)+Ml^2} \end{bmatrix} u \qquad (20)$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u \qquad (21)$$

La matriz *C* tiene 2 filas porque tanto la posición del carro como la posición del péndulo son parte de la salida. Específicamente, la posición del carro es el primer elemento de la salida *y* la desviación del péndulo de su posición de equilibrio es el segundo elemento de *y*.

```
p = I*(M+m)+M*m*l^2; % denominador para matrices A y B

A = [0      1              0           0;
     0 -(I+m*l^2)*b/p  (m^2*g*l^2)/p   0;
     0      0              0           1;
     0 -(m*l*b)/p       m*g*l*(M+m)/p  0];
B = [     0;
     (I+m*l^2)/p;
          0;
        m*l/p];
C = [1 0 0 0;
     0 0 1 0];
D = [0;
     0];

states = {'x' 'x_punto' 'phi' 'phi_punto'};
inputs = {'u'};
outputs = {'x'; 'phi'};

sys_ss =
ss(A,B,C,D,'statename',states,'inputname',inputs,'outputname',outputs)
```

```
sys_ss =

  A =
```

```
                    x    x_punto        phi  phi_punto
 x                  0          1          0          0
 x_punto            0     -0.1818      2.673          0
 phi                0          0          0          1
 phi_punto          0     -0.4545      31.18          0

B =
                u
 x              0
 x_punto    1.818
 phi            0
 phi_punto  4.545

C =
               x    x_punto        phi  phi_punto
 x             1          0          0          0
 phi           0          0          1          0

D =
       u
 x     0
 phi   0

Continuous-time state-space model.
```

El modelo de espacio de estado anterior también se puede convertir en forma de función de transferencia empleando el comando *tf* como se muestra a continuación.

```
 sys_tf = tf(sys_ss) %Función transferencia desde un modelo en espacio de estados%
```

```
sys_tf =

  From input "u" to output...
          1.818 s^2 + 4.845e-15 s - 44.55
   x:  ---------------------------------------
       s^4 + 0.1818 s^3 - 31.18 s^2 - 4.455 s

                4.545 s - 7.774e-16
   phi:  ---------------------------------
         s^3 + 0.1818 s^2 - 31.18 s - 4.455

Continuous-time transfer function.
```

Por el contrario, el modelo de función de transferencia se puede convertir en forma de espacio de estado utilizando el comando *ss*.

4.3.5 MODELADO EN SIMULINK/SIMSCAPE MULTIBODY

Simscape Multibody es una herramienta apropiada para modelar, simular y controlar el péndulo invertido. Mediante el comando "*smnew"* se abre un lienzo para nuevo modelo y se colocan todos los elementos que ya se han mencionado. El modelo resultante debe aparecer de la siguiente manera:

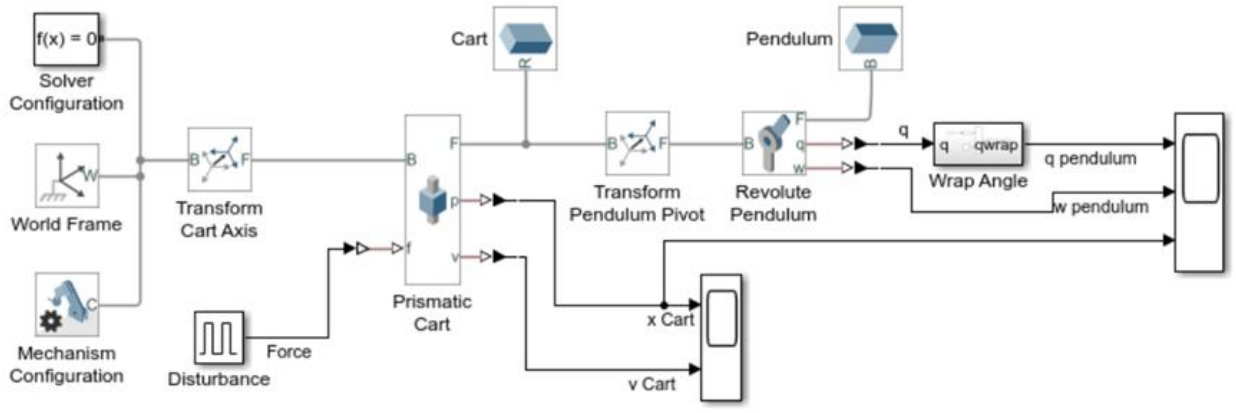


Ilustración 67 Modelado del sistema péndulo invertido en entorno simscape-multibody

Ahora hemos creado con éxito todos los elementos del sistema de péndulo invertido, convertiremos esto en un subsistema se muestra en la siguiente figura:

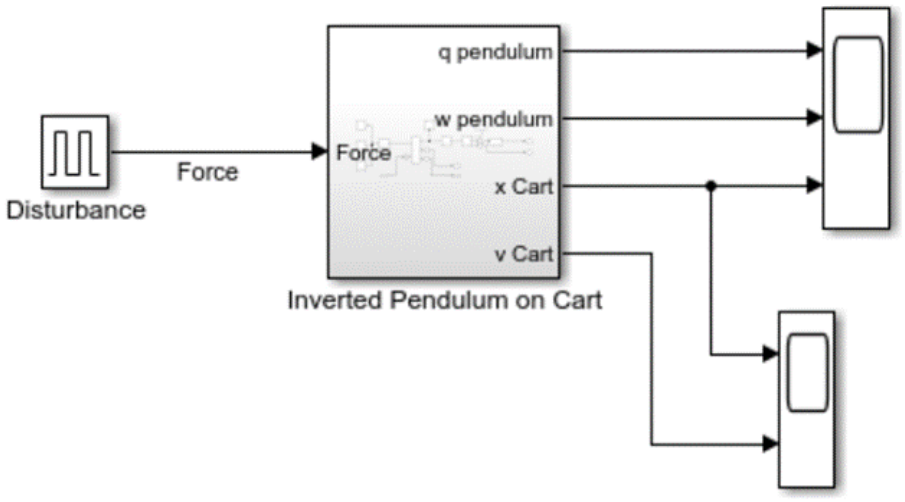


Ilustración 68 Representación por diagrama de bloques del sistema

4.3.6 ANÁLISIS DEL COMPORTAMIENTO

Considerando la respuesta del péndulo a un impulso de 1 Nseg aplicado al carro, los requisitos de diseño para el péndulo son:

- Tiempo de asentamiento *Theta* de menos de 5 segundos
- Ángulo del péndulo *Theta* no menor de 0,05 radianes de la vertical

Adicionalmente, los requisitos para la respuesta del sistema a un comando de paso de 0,2 metros en posición de carro son:

- Tiempo de establecimiento para *x* y *Theta* de menos de 5 segundos
- Tiempo de subida para *x* de menos de 0,5 segundos
- Ángulo del péndulo *Theta* no menor de 20 grados (0,35 radianes) de la vertical

Ahora podemos examinar la respuesta de impulso en lazo abierto del sistema. Específicamente, examinaremos cómo responde el sistema a una fuerza impulsiva aplicada al carro empleando el comando *impulse* de MATLAB. Agregue los siguientes comandos al final del archivo m y ejecútelo en la ventana de comandos de MATLAB para obtener el gráfico asociado que se muestra a continuación.

```
 t=0:0.01:1; %Definición de un tiempo en especifico%
 impulse(sys_tf,t); %Respuesta de un impulso%
 title('Open-Loop Impulse Response') %Define un titulo en específico a
una gráfica%
```

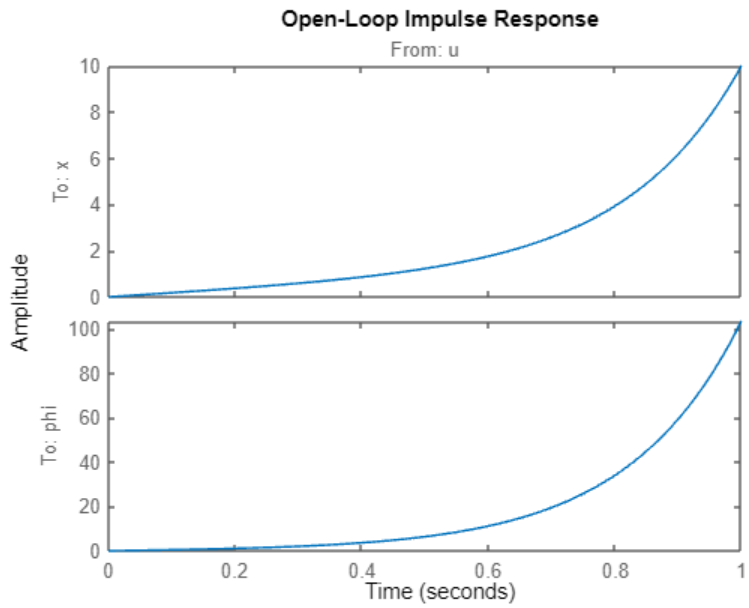


Ilustración 69 Respuesta en lazo abierto del sistema

Como puede ver en el gráfico, la respuesta del sistema es totalmente insatisfactoria. De hecho, no es estable en lazo abierto. Aunque se muestra que la posición del péndulo aumenta más allá de los 100 radianes (15 revoluciones), el modelo solo es válido para pequeños *Phi*. También puede ver que la posición del carro se mueve infinitamente hacia la derecha, aunque no hay ningún requisito en la posición del carro para una entrada de fuerza impulsiva.

Los polos de un sistema también pueden informarnos sobre su respuesta en el tiempo. Dado que nuestro sistema tiene dos salidas y una entrada, se describe mediante dos funciones de transferencia. En general, todas las funciones de transferencia de cada entrada a cada salida de un sistema de múltiples entradas y múltiples salidas (MIMO) tendrán los mismos polos (pero diferentes ceros) a menos que haya cancelaciones de polos y ceros. Examinaremos específicamente los polos y ceros del sistema utilizando la función "*zpkdata"* de MATLAB. El parámetro 'v' que se muestra a continuación devuelve los polos y ceros como vectores de columna en lugar de matrices de celdas. Los ceros y polos del sistema donde la posición del péndulo es la salida se encuentran como se muestra a continuación:

```
 [ceros polos] = zpkdata(P_pend, 'v' ) %Definición de matriz de ceros y
polos%
```

```
ceros = 0
```

```
polos = 3×1
     5.5651
    -5.6041
    -0.1428
```

Así mismo, los ceros y polos del sistema donde la posición del carro es la salida se encuentran de la siguiente manera:

```
 [ceros polos] = zpkdata(P_cart, 'v' ) %Definición de matriz de ceros y
polos%
```

```
ceros = 2×1
     4.9497
    -4.9497
polos = 4×1
          0
     5.5651
    -5.6041
    -0.1428
```

Como se predijo, los polos de ambas funciones de transferencia son idénticos. El polo en 5.5651 indica que el sistema es inestable ya que el polo tiene parte real positiva. En otras palabras, el polo está en la mitad derecha del plano s complejo. Esto concuerda con lo que observamos anteriormente.

Dado que el sistema tiene un polo con parte real positiva, su respuesta a una entrada escalonada también crecerá sin límites. Verificaremos esto usando el comando *"lsim"* que se puede emplear para simular la respuesta de los modelos LTI a entradas arbitrarias. En este caso, se utilizará una entrada escalón de 1 Newton. Agregar el siguiente código a su archivo m y ejecutarlo en la ventana de comandos de MATLAB generará el gráfico que se muestra a continuación.

```
 t = 0:0.05:10; %Definición de un tiempo en especifico%
 u = ones(size(t)); %Definición de variable u, la entrada%
 [y,t] = lsim(sys_tf,u,t);
 plot(t,y) %Código para la realización de una grafica%
 title('Open-Loop Step Response') %Define un titulo en específico a una
gráfica%
 axis([0 3 0 50]) %Define los saltos numericos en el eje X & Y%
 legend('x','phi') %Define una leyenda en específico en una gráfica%
```

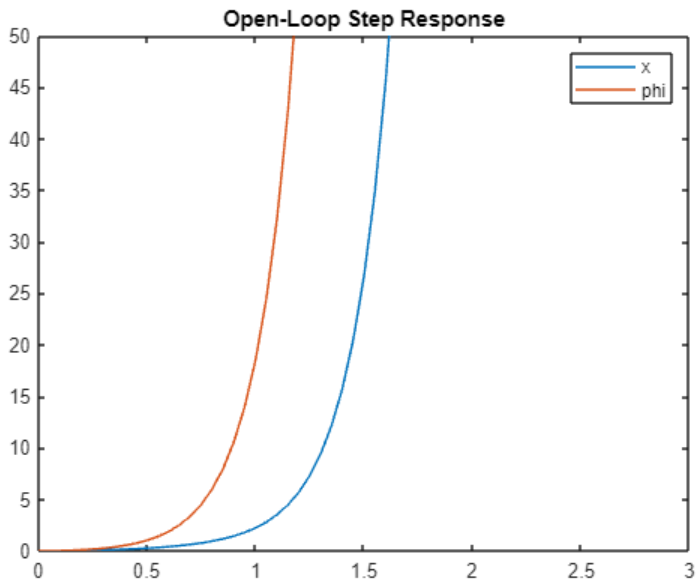


Ilustración 70 Respuesta del sistema en lazo abierto del sistema con dos entradas

También puede identificar algunas características importantes de la respuesta usando el comando "*lsiminfo"* como se muestra.

```
 step_info = lsiminfo(y,t); %Computa y muestra información de una
entrada o salida%
 cart_info = step_info(1) %Computa y muestra información de una entrada
o salida%
```

```
cart_info = struct with fields:
    TransientTime: 9.9959
     SettlingTime: 9.9959
              Min: 0
          MinTime: 0
              Max: 8.7918e+21
          MaxTime: 10
```

```
 pend_info = step_info(2) %Computa y muestra información de una entrada
o salida%
```

```
pend_info = struct with fields:
    TransientTime: 9.9959
     SettlingTime: 9.9959
              Min: 0
          MinTime: 0
              Max: 1.0520e+23
          MaxTime: 10
```

Los resultados anteriores confirman nuestra expectativa de que la respuesta del sistema a una entrada de paso es inestable. Del análisis anterior se desprende que será necesario diseñar algún tipo de control para mejorar la respuesta del sistema.

4.3.7 DISEÑO DE CONTROLADOR PID

Más específicamente, el controlador intentará mantener el péndulo verticalmente hacia arriba cuando el carro esté sujeto a un impulso de 1 Nseg. En estas condiciones, los criterios de diseño son:

- Tiempo de establecimiento de menos de 5 segundos
- El péndulo no debe alejarse más de 0,05 radianes de la vertical

La estructura del controlador para este problema es un poco diferente a los problemas de control estándar a los que puede estar acostumbrado. Dado que estamos intentando controlar la posición del péndulo, que debería volver a la vertical después de la perturbación inicial, la señal de referencia que estamos rastreando debería ser cero. Este tipo de situación a menudo se denomina problema del regulador. La fuerza externa aplicada al carro se puede considerar como una perturbación impulsiva. El esquema de este problema se muestra a continuación.

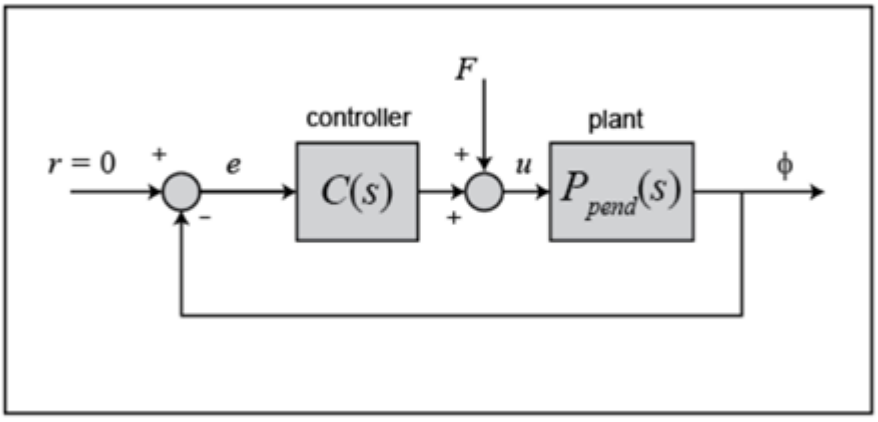


Ilustración 71 Esquema por diagrama de bloques del sistema sin control de un péndulo invertido

Puede que le resulte más fácil analizar y diseñar para este sistema si primero reorganizamos el esquema de la siguiente manera.

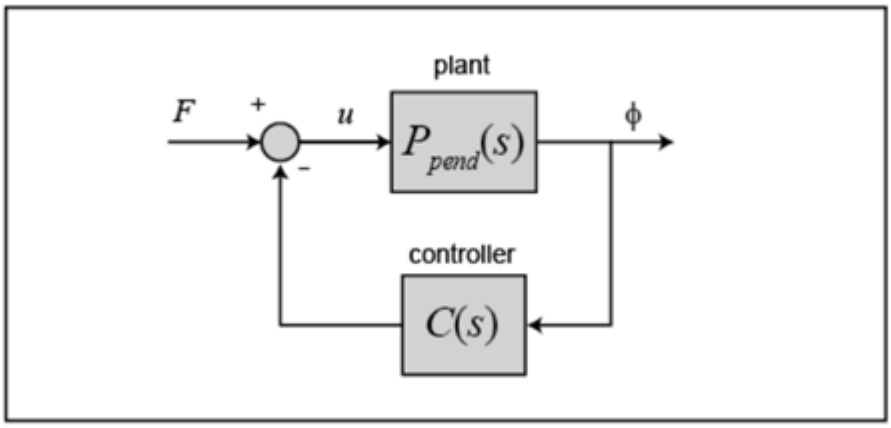


Ilustración 72 Reorganización del esquema por diagrama de bloques del sistema sin control de un péndulo invertido

Entonces se determina que la función de transferencia resultante para el sistema de lazo cerrado desde una entrada de fuerza *F* hasta una salida de ángulo de péndulo *Phi* es la siguiente.

$$\frac{\Phi(s)}{F(s)} = \frac{P_{pend}(s)}{1 + C(s)P_{pend}(s)} \qquad (22)$$

Esta función de transferencia de ciclo cerrado se puede modelar en MATLAB copiando el siguiente código al final de su archivo m (ya sea que esté utilizando el formulario de función de transferencia o la representación de espacio de estado de la planta). Específicamente, definimos nuestro controlador usando el objeto *PID* dentro de MATLAB. Luego usamos el comando "*feedback"* para generar la función de transferencia de lazo cerrado como se muestra en la figura anterior, donde la fuerza de perturbación *F* es la entrada y la desviación del ángulo del péndulo con respecto a la vertical *Phi* es la salida.

```
Kp = 1; %Control proporcial Kp%
Ki = 1; %Control integral Ki%
Kd = 1; %Control derivativa Kd%
C1 = pid(Kp,Ki,Kd); %Definición de Variable%
T = feedback(P_pend,C1); %Definición de Variable%
```

Ahora podemos comenzar a sintonizar nuestro controlador. Primero, examinemos la respuesta del sistema de lazo cerrado a una perturbación de impulso para este conjunto inicial de ganancias de control. Ingrese el siguiente código al final de su archivo m y ejecútelo en la ventana de comandos de

MATLAB. Debe generar el gráfico de respuesta que se muestra a continuación.

```
 t=0:0.01:10; %Definición de un tiempo en especifico%
 impulse(T,t) %Definición de un impulso a una entrada%
 title({'Respuesta de la posición del pendulo ante una perturbacion
impulsiva';'con control PID: Kp = 1, Ki = 1, Kd = 1'}); %Define un
titulo en específico a una gráfica%
```

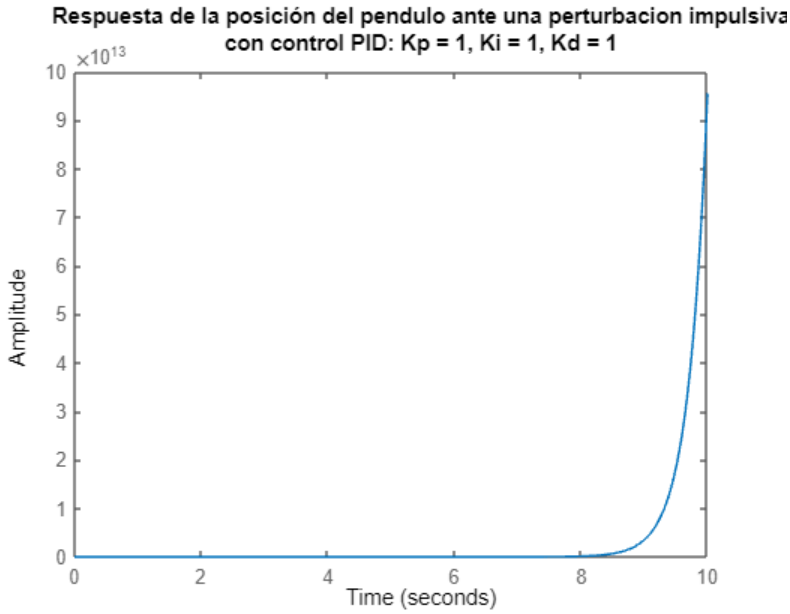


Ilustración 73 Respuesta de la posición del péndulo con parámetros iguales a 1

Esta respuesta aún no es estable. Comencemos a modificar la respuesta aumentando la ganancia proporcional. Aumente la variable *Kp* para ver qué efecto tiene en la respuesta. Si modifica su archivo m a lo siguiente donde *Kp* = 100 y lo ejecuta en la ventana de comandos, debería obtener el gráfico de respuesta que se muestra a continuación.

```
 Kp = 100; %Control proporcial Kp%
 Ki = 1; %Control integral Ki%
 Kd = 1; %Control derivativo Kd%
 C2 = pid(Kp,Ki,Kd); %Definición de una variable%
 T = feedback(P_pend,C2); %Definición de una variable%
 t=0:0.01:10; %Definición de un tiempo en especifico%
 impulse(T,t) %Definición de un impulso a una entrada%
 axis([0, 2.5, -0.2, 0.2]); %Define los saltos numericos en el eje X &
Y%
```

```
 title({'Respuesta de la posicion del pendulo ante una perturbacion
impulsiva';'con control PID: Kp = 100, Ki = 1, Kd = 1'}); %Define un
titulo en específico a una gráfica%
```

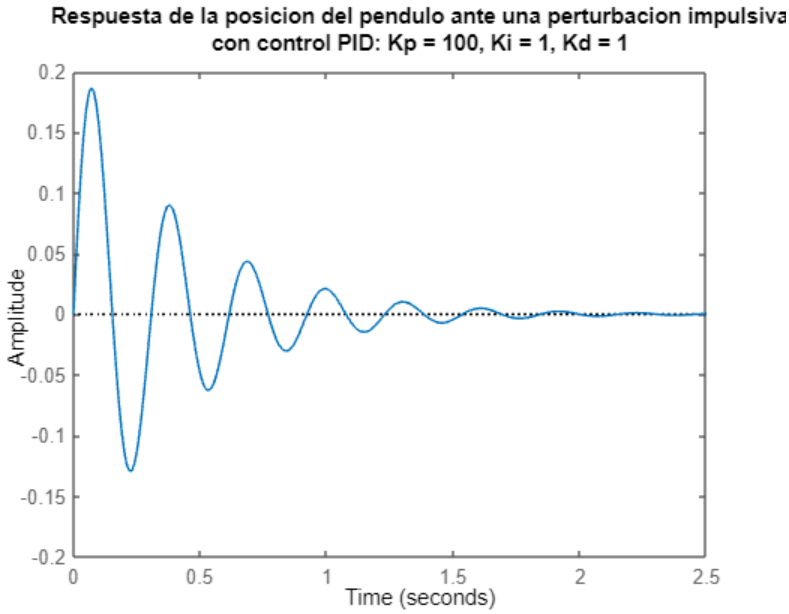


Ilustración 74 Respuesta de la posición del péndulo variando la variable proporcional

Si hace clic con el botón derecho en el gráfico resultante y selecciona características en el menú resultante, podrá identificar características importantes de la respuesta. Específicamente, se determina que el tiempo de establecimiento de la respuesta es de 1,64 segundos, que es inferior al requisito de 5 segundos. Dado que el error de estado estacionario se aproxima a cero con suficiente rapidez, no se necesita ninguna acción integral adicional. Puede establecer la constante de ganancia integral *Ki* = 0 para ver por sí mismo que se necesita algún control integral. Sin embargo, la respuesta máxima es mayor que el requisito de 0,05 radianes. Recuerde que el sobreimpulso a menudo se puede reducir aumentando la cantidad de control derivado. Después de algunas pruebas y errores, se encuentra que una ganancia derivada *Kd* = 20 proporciona una respuesta satisfactoria. Modificar su archivo m de la siguiente manera y volver a ejecutar debería producir el gráfico de respuesta que se muestra a continuación

```
Kp = 100; %Control proporcial Kp%
Ki = 1; %Control integral Ki%
Kd = 20; %Control derivativo Kd%
C3 = pid(Kp,Ki,Kd); %Definición de una variable%
```

```
 T = feedback(P_pend,C3); %Definición de una variable%
 t=0:0.01:10; %Definición de un tiempo en especifico%
 impulse(T,t) %Definición de un impulso a una entrada%
 axis([0, 2.5, -0.2, 0.2]); %Define los saltos numericos en el eje X &
Y%
 title({'Respuesta de la posición del pendulo ante una perturbacion
impulsiva';'con control PID: Kp = 100, Ki = 1, Kd = 20'}); %Define un
titulo en específico a una gráfica%
```

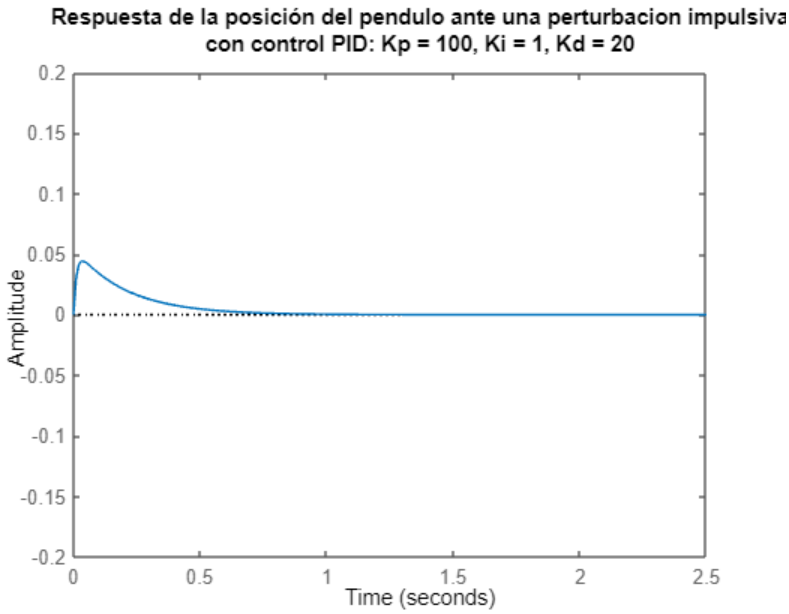


Ilustración 75 Respuesta de la posición del péndulo variando la variable proporcional y la derivativa

Como se puede ver, el sobreimpulso se ha reducido para que el péndulo no se aleje más de 0,05 radianes de la vertical. Dado que se han cumplido todos los requisitos de diseño dados, no se necesita más iteración.

```
% Convert Response Time to Bandwidth
% Bandwidth is equivalent to 2 divided by the Response Time
wc = 2/7.74461;

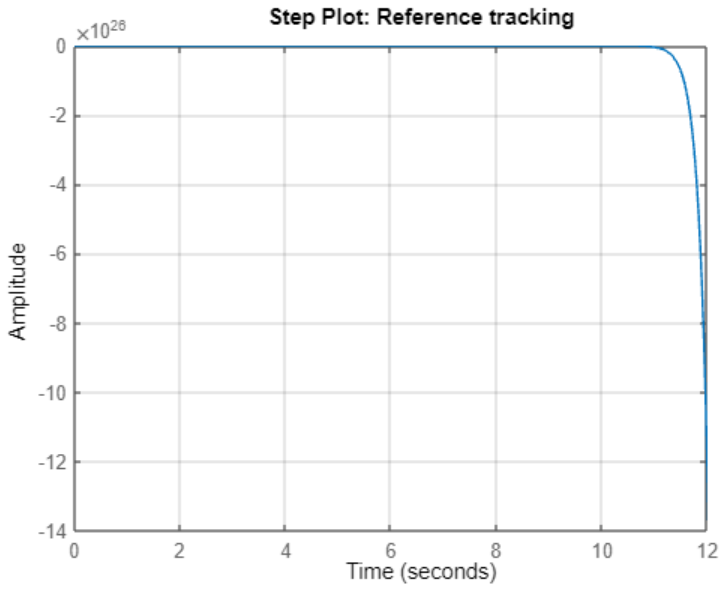
% PID tuning algorithm for linear plant model
[C4,pidInfo] = pidtune(P_cart,'PID',wc);

% Clear Temporary Variables
clear wc

% Get desired loop response
```

```
Response = getPIDLoopResponse(C4,P_cart,'closed-loop');

% Plot the result
stepplot(Response)
title('Step Plot: Reference tracking')
grid on
```




```
% Clear Temporary Variables
clear Response
```

4.3.8 CONTROL PID EN SIMULINK/SIMSCAPE MULTIBODY

Después del modelo del péndulo realizado en Simulink/Simscape Multibody ahora agregaremos bloques para pruebas en lazo abierto y cerrado, agregando los bloques como en la siguiente figura

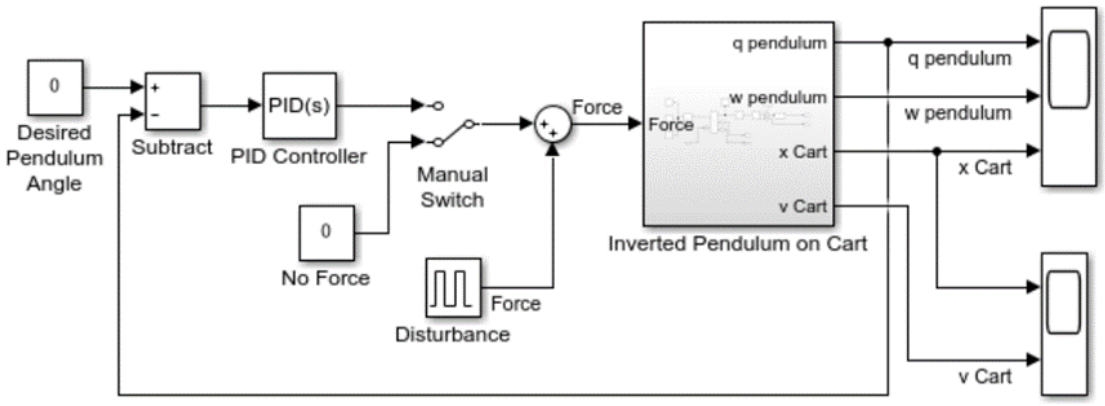


Ilustración 76 Diagrama de bloques del sistema con controlador PID

Ahora implementaremos las ganancias de control PID

- Haga doble clic en el bloque Controlador PID
- Establecer parámetro Proporcional (P) a "100"
- Establezca el parámetro Integral (I) en "1"
- Establezca el parámetro Derivada (D) en "20"

Haciendo doble clic en Cambio manual hasta que se seleccione la entrada del controlador PID.

Ejecutar una simulación produce los siguientes dos gráficos que muestran la respuesta controlada del sistema. Después del impacto inicial, el controlador pudo reducir rápidamente el ángulo del péndulo a cero y la velocidad del péndulo también es cero. El carro se mueve lentamente y con velocidad constante en la dirección X negativa para mantener el péndulo equilibrado.

4.3.9 CONTROL EN EL ESPACIO DE ESTADOS

El siguiente paso en el proceso de diseño es encontrar el vector de ganancias de control de retroalimentación de estado suponiendo que se tiene acceso (es decir, que se puedan medir) a las cuatro variables de estado. Esto se puede lograr de varias maneras. Si conoce las ubicaciones deseadas de los polos en lazo cerrado, puede usar los comandos "*place"* o "*acker"* de MATLAB. Otra opción es usar el comando "*lqr"* que devuelve la ganancia óptima del controlador suponiendo una planta lineal, una función de costo cuadrática y una referencia igual a cero.

Antes de diseñar el controlador, primero hay que verificar que el sistema sea controlable. La satisfacción de esta propiedad significa que podemos conducir el estado del sistema a donde queramos en un tiempo finito (bajo las restricciones físicas del sistema). Para que el sistema sea de estado completamente controlable, la matriz de controlabilidad debe tener un rango *n*, donde el rango de una matriz es el número de filas (o columnas) linealmente independientes. La matriz de controlabilidad del sistema toma la forma que se muestra a continuación. El número corresponde al número de variables de estado del sistema. Adición de términos adicionales a la matriz de controlabilidad con mayores potencias de la matriz no aumentará el rango de la matriz de controlabilidad ya que estos términos adicionales serán solo combinaciones lineales de los términos anteriores.

$$\mathrm{CO} = \begin{bmatrix} B & \mathrm{AB} & A^2B & \cdots & A^{N-1}B \end{bmatrix} \quad (22)$$

Dado que la matriz de controlabilidad es 4x4, el rango de la matriz debe ser 4. Se usa el comando "*ctrb"* para generar la matriz de controlabilidad y el comando "*Rank"* para probar el rango de la matriz. Agregando los siguientes comandos adicionales al archivo .m producirá el siguiente resultado

```
 co = ctrb(sys_ss); %Código para computar la controlabilidad de una
matriz%
 controllability = rank(co) %Código para computar la controlabilidad de
una matriz%
```

```
controllability = 4
```

Por lo tanto, se ha verificado que el sistema es controlable y, por lo tanto, se debe poder diseñar un controlador que cumpla con los requisitos dados. Específicamente, se utilizará un regulador lineal cuadrático para determinar la matriz de ganancia de control de retroalimentación de estado *K*. La función *lqr* permite elegir dos parámetros, *R* y *Q*, que equilibrarán la importancia relativa del esfuerzo de control *u* y del error (desviación de 0), respectivamente, en la función de costo que está tratando de optimizar. El caso más simple es asumir *R* = 1, y *Q* = *C'C*. La función de coste correspondiente a este *R* y *Q* da igual importancia al control y a las variables de estado que son salidas (el ángulo del péndulo y la posición del carro). Esencialmente, el método *lqr* permite el control de ambas salidas. En este caso, es bastante fácil de hacer. El controlador se puede ajustar cambiando los elementos distintos de cero en la matriz *Q* para lograr una respuesta deseable. Para observar la estructura de *Q*, se ingresa lo siguiente en la ventana de comandos:

```
 Q = C'*C %Definición de una operación con variables definidas%
```

```
Q = 4×4
     1     0     0     0
     0     0     0     0
     0     0     1     0
     0     0     0     0
```

El elemento en la posición (1,1) de *Q* representa el peso en la posición del carro y el elemento en la posición (3,3) representa el peso en el ángulo del péndulo. La ponderación de entrada *R* permanecerá en 1. En última instancia, lo que importa es el valor relativo de *Q* y *R*, no sus valores absolutos. Ahora que se sabe cómo interpretar la matriz *Q*, se puede experimentar para encontrar la matriz *K* que dará un "buen" controlador. Agregando los siguientes comandos al final del archivo .m para obtener el siguiente valor para *K* y el gráfico de respuesta que se muestra a continuación.

```
 R = 1; %Definición de una variable%
 K = lqr(A,B,Q,R) %Regulador cuadrático lineal diseñado para sistemas
en espacio de estados%
```

```
K = 1×4
   -1.0000   -1.6567   18.6854    3.4594
```

```
 Ac = [(A-B*K)]; %Operaciones entre matrices%
 Bc = [B]; %Operaciones entre matrices%
 Cc = [C]; %Operaciones entre matrices%
 Dc = [D]; %Operaciones entre matrices%

 states = {'x' 'x_dot' 'phi' 'phi_dot'}; %Definición de estados a
entradas especificas%
 inputs = {'r'}; %Definición de entradas%
 outputs = {'x'; 'phi'}; %Definición de salidas%

 sys_cl =
ss(Ac,Bc,Cc,Dc,'statename',states,'inputname',inputs,'outputname',outpu
ts); %Función de transferencia a entradas y salidas especificas%

 t = 0:0.01:5; %Definición de un tiempo en especifico%
 r =0.2*ones(size(t));
```

```
 [y,t,x]=lsim(sys_cl,r,t); %Simulación del tiempo de respuesta a
sistemas dinámicos sin importar las entradas%
 [AX,H1,H2] = plotyy(t,y(:,1),t,y(:,2),'plot'); %Código para gestionar
una gráfica%
 set(get(AX(1),'Ylabel'),'String','cart position (m)') %Proporcionar
propiedades a las variables%
 set(get(AX(2),'Ylabel'),'String','pendulum angle (radians)')
%Proporcionar propiedades a las variables%
 title('Step Response with LQR Control') %Define un titulo en
específico a una gráfica%
```

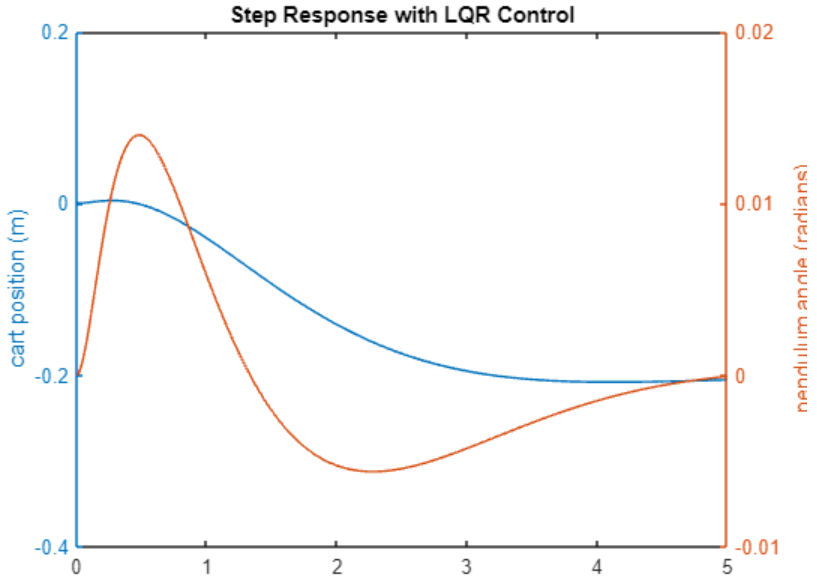


Ilustración 77 Respuesta del sistema con un controlador LQR

La curva en rojo representa el ángulo del péndulo en radianes y la curva en azul representa la posición del carro en metros. Como se puede ver, este comportamiento no es satisfactoria. El rebasamiento del péndulo y del carro parece estar bien, pero sus tiempos de estabilización necesitan mejorarse y el tiempo de subida del carro debe reducirse. Como seguramente habrá notado, la posición final del carrito tampoco está cerca de la ubicación deseada, sino que se ha movido en la dirección opuesta. Regresando al archivo .m se cambia la matriz *Q* para ver si puede obtener una mejor respuesta. Finalmente, se encuentra que al aumentar los elementos (1,1) y (3,3) hace que los tiempos de asentamiento y subida disminuyan, y reduce el ángulo que se mueve el péndulo. En otras palabras, está poniendo más peso en los errores a costa de

un mayor esfuerzo de control. Modificando el archivo .m para que el elemento (1,1) de *Q* sea 5000 y el elemento (3,3) sea 100, producirá el siguiente valor de *K* y la respuesta de paso que se muestra a continuación

```
 Q = C'*C; %Operación entre variables%
 Q(1,1) = 5000;
 Q(3,3) = 100
```

```
Q = 4×4
        5000           0           0           0
           0           0           0           0
           0           0         100           0
           0           0           0           0
```

```
 R = 1; %Definición de una variable%
 K = lqr(A,B,Q,R) %Regulador cuadrático lineal diseñado para sistemas
en espacio de estados%
```

```
K = 1×4
  -70.7107  -37.8345  105.5298   20.9238
```

```
 Ac = [(A-B*K)]; %Operaciones entre matrices%
 Bc = [B]; %Operaciones entre matrices%
 Cc = [C]; %Operaciones entre matrices%
 Dc = [D]; %Operaciones entre matrices%

 states = {'x' 'x_dot' 'phi' 'phi_dot'}; %Definición de estados a
entradas especificas%
 inputs = {'r'}; %Definición de entradas%
 outputs = {'x'; 'phi'}; %Definición de salidas%

 sys_cl =
ss(Ac,Bc,Cc,Dc,'statename',states,'inputname',inputs,'outputname',outpu
ts); %Función de transferencia a entradas y salidas especificas%

 t = 0:0.01:5; %Definición de un tiempo en especifico%
 r =0.2*ones(size(t));
 [y,t,x]=lsim(sys_cl,r,t); %Simulación del tiempo de respuesta a
sistemas dinámicos sin importar las entradas%
 [AX,H1,H2] = plotyy(t,y(:,1),t,y(:,2),'plot'); %Código para gestionar
una gráfica%
 set(get(AX(1),'Ylabel'),'String','cart position (m)')  %Proporcionar
propiedades a las variables%
 set(get(AX(2),'Ylabel'),'String','pendulum angle (radians)')
%Proporcionar propiedades a las variables%
```

```
 title('Step Response with LQR Control') %Define un titulo en
específico a una gráfica%
```

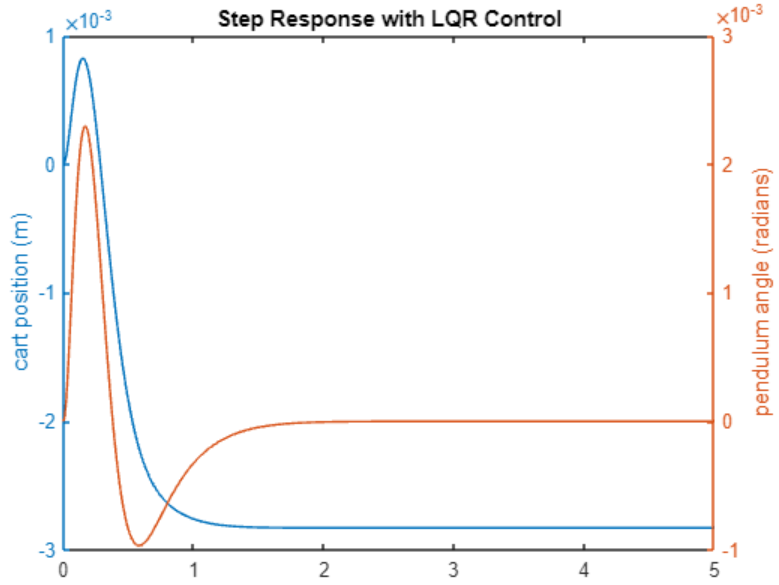


Ilustración 78 Respuesta del sistema con varias entradas con controlador LQR

Es posible que haya notado que si aumenta los valores de los elementos de *Q* aún más, podría mejorar la respuesta aún más. La razón por la que se eligió esta ponderación fue porque solo satisface los requisitos de diseño transitorio. Aumentar la magnitud de *Q* más haría que el error de seguimiento fuera más pequeño, pero requeriría una mayor fuerza de control. Más esfuerzo de control generalmente corresponde a un mayor costo (más energía, actuador más grande, etc.).

5. ANÁLISIS DE RESULTADOS

Se evidencia que la metodología mostrada desde el inicio del capítulo 3 da primero, un avance con respecto a definir un orden para realizar los múltiples ejercicios que se presentaron, además, como se define un paso a paso, es decir, se está aplicando la definición de lo que es un algoritmo se ratifica cada vez más la finalidad de este proyecto. Sin un orden especifico a seguir el desarrollo de los ejercicios tienden a ser más complejos de realizar y más confusos, que es lo que sucede en la mayoría de los casos cuando se quiere hacer un compilado de ejercicios.

La optimización del tiempo también es un factor clave a la hora de haber definido la metodología propuesta, pues cuando se tiene claro que es y cómo se va a realizar no se pierde el tiempo precisamente en pensar y analizar cómo se tienen que llevar a cabo los ejercicios. En los ejercicios desarrollados en el capítulo 4 el desarrollo físico-matemático es diferente para cada uno de los sistemas de control, sin embargo, en la metodología propuesta está plasmado como llevar a cabo el análisis desde una perspectiva general que irá facilitando la solución del ejercicio, desarrollo que sin la metodología puede variar y se pueden realizar análisis que no terminan siendo propios del sistema.

Uno de los factores claves fue someter la metodología en cuestión en un curso que esté siendo involucrado de manera directa con los sistemas de control y sus aplicaciones, así bien, se hizo una encuesta después de presentarla para medir los resultados de la metodología, se hicieron varias preguntas entre ellas el conocimiento del software, el uso y el tiempo del mismo obteniendo los siguientes resultados. Véase ilustración 79, 80, 81.

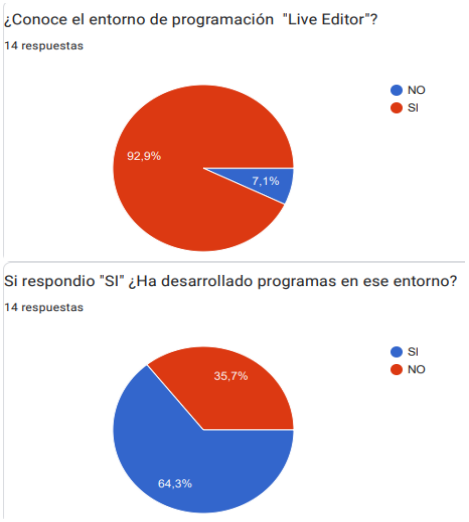


Ilustración 79 Encuesta de satisfacción, conocimiento de la herramienta

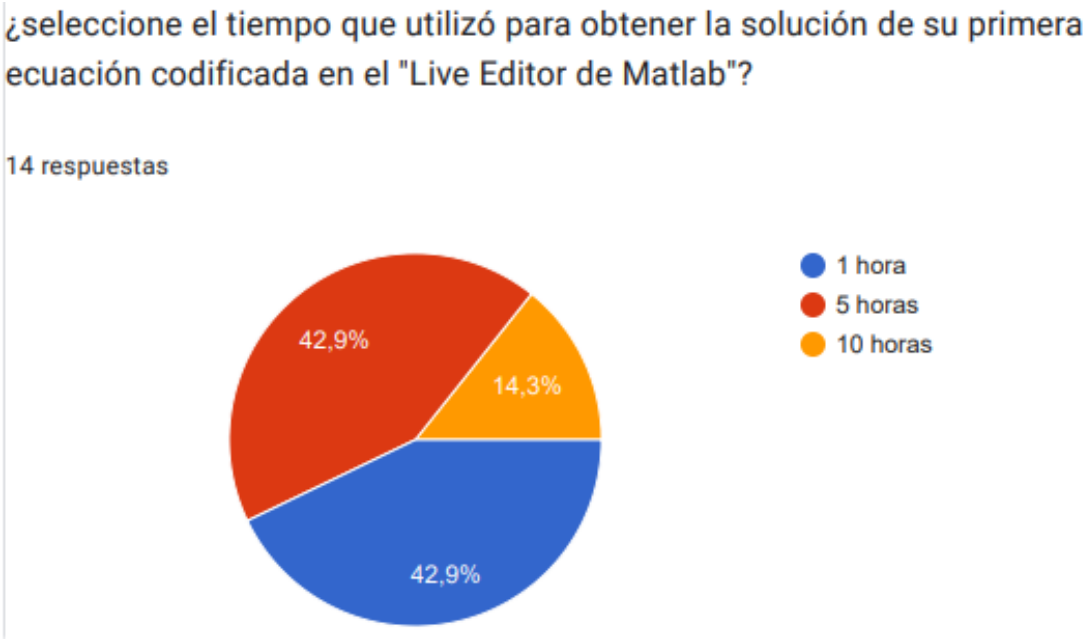


Ilustración 80 Encuesta de satisfacción, tiempo y uso de la herramienta

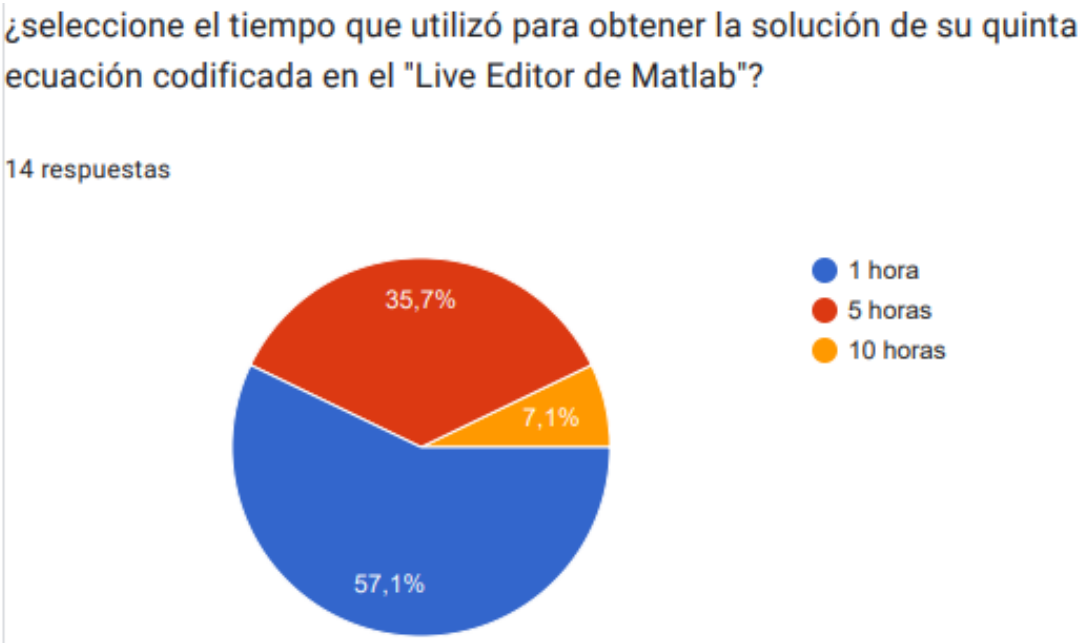


Ilustración 81 Encuesta de satisfacción, mayor tiempo de uso de la herramienta

Se puede evidenciar que el tiempo que tardan los estudiantes a la hora de realizar los ejercicios o de programarlos es reducido a medida que los estudiantes van usando el software y la metodología presentada. Los resultados obtenidos fueron tenidos en cuenta para evidenciar y ajustar algunos parámetros dentro de la metodología.

Además, como una de las opciones de descarga dentro del Live Script, es .HTML, en formato de página, se puede hacer la estandarización de los ejercicios haciendo una distribución libre en una página que tenga la opción de descargar los ejercicios desarrollados con la metodología y también recibir ejercicios si y solo si están desarrollados con la misma.

Finalmente, El análisis del sistema se centró en su estabilidad, es decir, en mantener las variables de entrada y salida dentro de límites aceptables. La respuesta transitoria del sistema también fue considerada, incluyendo el tiempo de respuesta, sobreimpulso y oscilaciones. Los requerimientos de cada sistema incluyeron el margen de ganancia y fase para asegurar un diseño óptimo. Finalmente, la sensibilidad del sistema a las perturbaciones fue evaluada y puesta a prueba, obteniendo resultados satisfactorios el ejemplo más evidente es el control de movimiento del avión cuando se agregó al

diagrama de bloques una perturbación intencional y con esta perturbación controlar de nuevo el sistema con esta perturbación, agregar el control del avión fue la opción más acertada obteniendo unos resultados muy por debajo a los requerimientos de diseño que se plantearon en el sistema inicialmente.

6. CONCLUSIONES

Desde un principio se quiso dejar plasmada una metodología que rigiera la mayor parte de este proyecto y precisamente es uno de los objetivos que se han cumplido. Con la utilización de la metodología propuesta se define un orden para llevar a cabo los ejercicios propuestos, la reducción de tiempo de ejecución y la facilidad de desarrollar los mismos.

La incorporación de conceptos, ilustraciones, ecuaciones y demás dentro del desarrollo de los ejercicios ayuda a la descripción de los sistemas y así también entender de manera clara lo que paralelamente se está haciendo que es la explicando y solución del mismo dentro de los ejercicios o sistemas. Utilizar la metodología descrita nos da la seguridad de cómo se va a llevar a cabo el ejercicio y como se compone el mismo, es por ello que la metodología propuesta se sometió a prueba en cada uno de los desarrollos de los ejercicios para validar su confiabilidad.

Como la metodología del sistema fue pensada para que el desarrollo de los sistemas fuera extrapolable hacia otros sistemas, la incorporación de código dentro la explicación nos ayuda a solventar este tipo de inconvenientes y se puede presentar de modo que sea usada como un instrumento estándar y de manera generalizada frente a cualquier tipo de sistema que tenga algo de similitud con los desarrollados acá.

7. REFERENCIAS

Regalón Anias, O., Rodríguez Diez, V., Diez Rodríguez, M., & Báez Prieto, R. (2012). Aplicación de algoritmos de control clásico, adaptable y robusto a sistemas dinámicos de parámetros variables. Ingeniería Energética, XXXIII(3), 184-195.

Mathworks. (28 de 02 de 2023). Obtenido de https://la.mathworks.com/

VOSviewer - Visualizing scientific landscapes. (28 de 02 de 2023). Obtenido de https://www.vosviewer.com/

Marek Kowalkiewicz. (2019, Octubre 10). How did we get here? The story of algorithms. - Towards Data Science. Medium; Towards Data Science. https://towardsdatascience.com/how-did-we-get-here-the-story-of-algorithms-9ee186ba2a07

Ahmet Fatih Turkel. (2023, enero 8). Getting to Know the Algorithm. - Ahmet Fatih Turkel - Medium. Medium; Medium. https://medium.com/@afturkel/getting-to-know-the-algorithm-2bfe4cacfdf1

Codecrucks. (2021, septiembre 5). Evolution of Algorithm - Origin and History. Codecrucks. https://codecrucks.com/evolution-of-algorithm-origin-and-history/

Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. Electronic Markets, 31(3), 685–695. https://doi.org/10.1007/s12525-021-00475-2

Losing, V., Hammer, B., & Wersing, H. (2018). Incremental on-line learning: A review and comparison of state of the art algorithms. Neurocomputing, 275, 1261–1274. https://doi.org/10.1016/j.neucom.2017.06.084

Burke, A. (2019). Occluded algorithms. Big Data and Society, 6(2). https://doi.org/10.1177/2053951719858743

Nwankpa, C. E. (2020). Advances in optimisation algorithms and techniques for deep learning. Advances in Science, Technology and Engineering Systems, 5(5), 563–577. https://doi.org/10.25046/AJ050570

Sabato, S., & Hess, T. (2018). Interactive algorithms: Pool, stream and precognitive stream. Journal of Machine Learning Research, 18, 1–39.

Chen, J., & Zhao, J. (2018). An Educational Data Mining Model for Supervision of Network Learning Process. International Journal of Emerging Technologies in Learning (IJET), 13(11), 67. https://doi.org/10.3991/ijet.v13i11.9599

Zhou, Y., Zhang, S., Luo, Q., & Abdel-Baset, M. (2019). CCEO: cultural cognitive evolution optimization algorithm. Soft Computing, 23(23), 12561–12583. https://doi.org/10.1007/s00500-019-03806-w

Yamashita, K., Fujioka, R., Kogure, S., Noguchi, Y., Konishi, T., & Itoh, Y. (2016). Practices of algorithm education based on discovery learning using a program visualization system. Research and Practice in Technology Enhanced Learning, 11(1). https://doi.org/10.1186/s41039-016-0041-5

Baker, R. S., & Hawn, A. (2021). Algorithmic Bias in Education. International Journal of Artificial Intelligence in Education, 32(4), 1052–1092. https://doi.org/10.1007/s40593-021-00285-9

Control Tutorials for MATLAB and Simulink - Suspension: System Modeling. (2023). Umich.edu. https://ctms.engin.umich.edu/CTMS/index.php?example=Suspension&section=SystemModeling

Control Tutorials for MATLAB and Simulink - Inverted Pendulum: System Modeling. (2023). Umich.edu. https://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum&section=SystemModeling

Control Tutorials for MATLAB and Simulink - Aircraft Pitch: System Modeling. (2023). Umich.edu. https://ctms.engin.umich.edu/CTMS/index.php?example=AircraftPitch&section=SystemModeling