|  |  |
| --- | --- |
| **horizontal-naranja** | **PROGRAMA DE INGENIERÌA MECATRÓNICA**  **UNIVERSIDAD AUTONOMA DE BUCARAMANGA - UNAB** |
| **CONTROL DE UN SISTEMA DE POSICIÓN – MODELO MATEMÁTICO**  ***M.Sc. Hernando González Acevedo*** |

|  |  |
| --- | --- |
| **INTEGRANTES** | **CÓDIGO** |
| **Neil Sebastián Castro Caicedo** | **U00131867** |
| **Hubert Armando Delgado Maestre** | **U00091396** |
| **Paula Andrea Portilla Corredor** | **U00139413** |

***TRABAJO PREVIO***

1. Investigar que métodos experimentales existen para calcular los parámetros del modelo matemático de un motor DC: inercia, fricción viscosa, resistencia, inductancia, constante de par y la constante de fuerza contraelectromotriz.

Se investigaron los parámetros necesarios para realizar un correcto modelo matemático del comportamiento de nuestra planta.

**Resistencia:** también considerado como resistencia de armadura denominada como “ra”, encontramos que existen dos métodos para determinarla en los cuales ambos dan el mismo valor.

El primer método mide la resistencia de los devanados de la armadura del motor (cables rojos y negros) mediante un multímetro.

El segundo método consiste en generar un voltaje mínimo para alimentar el motor, donde con un amperímetro determinaremos la corriente antes de que el motor empiece a rotar, para último, con la ley de OHM calcular la resistencia.

**Constante contraelectromotriz (ke):** La constante electromotriz del motor DC se define como el valor de la FCEM cuando el motor gira a una velocidad específica. Es una medida de la eficiencia del motor para convertir la energía eléctrica en energía mecánica. Se expresa en voltios por velocidad angular, generalmente en unidades de voltios por revolución por minuto (V/rpm) o voltios por radián por segundo (V/rad/s).

*Ke=Ebn=Vw [V/rpm] o [ V/rad/s]*

*Ke= constante contraelectromotriz [V/rpm]*

*Eb=V= Fuerza contra electromotriz o Tensión del motor [V]*

*w= velocidad en rad/s*

*n= velocidad en RPM*

**Inductancia:** se calcula por medio de un LCR meter el cual mide la inductancia en los devanados del motor y se denominará con “La”

**Constante de torque o par (Kt):** Se utiliza la técnica llamada parámetro dimensional que no recurre a la prueba experimental, pero se reconoce como útil y confiable para los motores DC, se basa en utilizar expresiones que guardan una relación paramétrica dimensional directa entre (Kt) con la constante de voltaje (Ke) y cuya comprobación empírica ha sido sustentada.

Dichas relaciones paramétricas son:

*Kt (Nm/A) =Ke (V/rad s)*

*Kt (Nm/A) =9.5493x10 Ke (V/Krpm)*

*Kt (oz-in/A) =1.3524 Ke (V/Krpm)*

**Inercia (Jm):** Se utiliza un método paramétrico ya que apela al conocimiento de los parámetros ya obtenidos o conocidos experimentalmente tales como:

*tm (s) = Jm\*RaKt\*Ke\*(kg-m2,ohm)(Nm/A,V/rad s)*

*Jm=(tm)(Kt)(Ke)Ra [Kg\*-m2]*

*Donde Ra es la resistencia de armadura, Kt es constante torque, Ke es constante de voltaje y tm es la constante de tiempo mecánica.*

**Constante de tiempo mecánica (tm):** Se mide el tiempo requerido para que la señal de salida alcance el 63,2% de su valor final, dicho tiempo corresponde a la constante de tiempo mecánica.

**Torque fricción y corriente arranque:** La corriente arranque se determina cuando el motor va a girar y se halla por medio del amperímetro.

Torque de fricción:

*Tf=Kt\*Iarr [Nm]*

*Donde Kt es constante torque e Iarr es la corriente de arranque.*

**Constante de fricción de coulomb (B):** La B se determina cuando el sistema se encuentra en estado estable, el motor alcanza una velocidad constante.

*Tm=Kt\*Ia=jw'+Bw+Tf*

*Donde w’=0*

*Tm=Kt\*Ia=Bw+Tf*

*w es velocidad, w’ es aceleración,Tm es torque mecánico.*

***MODELO MATEMÁTICO***

1. Determinar el modelo matemático de un sistema de control de posición de una barra y realizar la representación en espacio de estados. La variable de entrada al modelo es la señal de modulación de ancho de pulso (PWM) que modifica el voltaje de alimentación del motor DC (), las variables de salida son la posición angular de la barra y la velocidad angular del eje del motor . En la figura 1 se observa un esquemático del prototipo.

|  |
| --- |
|  |

*Figura 1. Control de posición de una barra*

|  |  |
| --- | --- |
| **Ecuaciones diferenciales** | Ecuaciones diferenciales observadas:  Como tenemos cuatro derivadas, tenemos cuatro estados, de los cuales pueden ser 𝜃𝐿, 𝜃𝑚, 𝑊𝑚, 𝑊𝐿 y 𝑖𝑎(t) pero, dado que las W y la theta guardan proporcional entre sí, la cual está definida por la relación de engranajes n.  Entonces tomamos como estados 𝑖𝑎(𝑡), 𝜃𝐿 𝑦 𝑊𝐿, quedando:  𝒙𝟏 = 𝒊𝒂  𝒙𝟐 = 𝜽𝑳  𝒙𝟑 = 𝑾𝑳  La entrada del sistema será el voltaje 𝑉𝑖𝑛(𝑡), procedemos con la derivación de los estados, obteniendo: |
| **Nombre de las variables y parámetros del modelo matemático** | *Vcc : [V] Voltaje*  *Ro : [Ohms] Resistencia motor*  *Lo : [H] Inductancia motor*  *Kvo : [V/rad s ] Cte contraelectromotriz motor*  *Kio : [Nm/A]  Cte par del motor*  *no : Relación caja de engranajes*  *Jmo : [Kg.m2] Inercia motor*  *Jlo : [Kg.m2] Inercia barra*  *Bmo : [Kg/s] Fricción viscosa motor*  *Blo : [Kg/s] Fricción viscosa barra*  *mo : [kg] masa de la barra*  *do : [m] Distancia del eje al centro de gravedad*  *go : [m/s^2] Gravedad* |
| **Vector de estados** | **Vectores de Estados:**  **Vectores**  **Entradas:**  **Salidas:** |
| **Punto de equilibrio (Expresión simbólica en función de los parámetros)** | **Asumiendo valor de x20:** |
| **Representación en espacio de estados del modelo lineal (Expresión simbólica en función de los parámetros)** | **.** |

1. Indicar las características más importantes de cada uno de los elementos que conforman el prototipo. Para la selección de la barra se debe cumplir que el producto entre el peso de la barra y la ubicación del centro de masa de la barra, lo cual corresponde al torque que se necesita para sostener la barra en posición horizontal, debe ser mayor o igual al 50% del torque máximo del motor.

|  |  |
| --- | --- |
| **Especificaciones técnicas del motor** | **Motorreductor DC con Encoder-SJ01, compatible con Arduino**    Voltaje de operación del motor: 3V ~ 7.5V  Voltaje: 6V a 160 RPM  Voltaje de operación del encoder: 4.75V ~ 7.5V  Corriente sin carga a 6V: 17mA  RPM: 160  Relación: 120:1  Máximo torque: 0.8kgf\*cm  Resolución: 16 pulsos por vuelta  Temperatura de operación: - 10°C ~ 60°C  Peso: 50g |
| **Especificaciones técnicas de la barra** | Masa = 0.1.7244 [Kg]  Material = Hierro dúctil  Volumen = 0.00002429 [m3]  Longitud = 0.13261 [m]  Centro de gravedad = 0.06861 [m]  Distancia c.gravedad-eje = 0.04762 [m]  Momento de inercia = 8.403161000000001e-05 [kg\*m^2] |
| **Etapa de potencia** | **Puente H L298n**  Modulo Driver Motor Puente H L298N 5v 2A - TresD Print Tech  Modelo: L298N  Tensión de funcionamiento: 6V dc a 24Vdc  Corriente de Salida: 2A  Potencia máxima: 25WCorriente máxima: 2ATamaño: 76mm\*44mm\*26mm |

1. Aplicando los métodos que investigaron en el trabajo previo, determinar el valor de los parámetros del motor Dc, además calcular los parámetros de la barra. Completar la siguiente tabla.

|  |  |  |
| --- | --- | --- |
| **Parámetro del motor** | **Procedimiento para calcular el parámetro** | **Valor (Unidad)** |
| Resistencia del motor R | Utilizando un multímetro y conectando entre los devanados | 3 [Ohms] |
| Inductancia del motor L | Utilizando el medidor de inductancia y conectando entre los extremos de la bobina del motor | 0.010 [H] |
| Corriente de arranque del motor I | Subimos la alimentación del motor y medimos la corriente a la cual gira el motor. | 0.14 [A] |
| Cte contraelectromotriz del motor Kv | Reemplazar en la fórmula de  Ke=Ebw=0.007054 V921.46 rad/s | 0.007054 [V/rads ] |
| Constante de par del motor Ki | Conociendo la relación establecida anteriormente para constante par y contraelectromotriz. | 0.004119 [Nm/A] |
| Torque de fricción Tf | Reemplazar en la fórmula de  Tf=Kt\*Iarr  [Nm]  donde *Kt* es el *Kvm* y *Iarr* es la corriente de arranque. | 0.0009875 Nm |
| Voltaje de operación Vin | Se alimenta el motor con 7V y se mide la tensión. | 7 [V] |
| Constante de tiempo mecánica tm | Valor tomado de  Cristian Julian Solarte Rosas y. Jhon Edinson Muñoz Ordoñez. (2015, marzo). *Guía para la Parametrización de un motor DC de imán permanente.* Principal. <http://repositorio.unicauca.edu.co:8080/xmlui/bitstream/handle/123456789/1488/ANEXO%20A.%20Guia%20parametrizacion%20de%20un%20motor%20DC%20de%20iman%20permanente%20.pdf?sequence=2&amp;isAllowed=y> | 125 [ms] |
| Torque mecánico Tm | Esto se obtiene de los parámetros nominales del motor donde nos da en *g\*cm* y lo pasamos a *Nm* | 0.0784532 [Nm] |
| Relación caja de engranajes | Se obtiene de la ficha tecnica del motor | 1:120 |
| Inercia motor Jm | Reemplazar los valores obtenidos anteriormente como son la constante de tiempo mecánica, las constante par y contraelectromotriz, la resistencia del motor en  la fórmula de  Jt=(tm)(Kt)(Ke)Ra [Kg\*-m2] | 525.83568e-7 [Kg.m2] |
| Fricción viscosa motor Bm | Con la relación entre parámetros conocidos se despeja la fórmula para obtener de este modo la fricción viscosa, obteniendo la relación: 𝐵𝐿=𝐽𝑚𝑡𝑚−𝑘𝑖∗𝑘𝑣𝑅 | 5.873e-9 [Kg/s] |

|  |  |  |
| --- | --- | --- |
| **Parámetro de la barra** | **Procedimiento para calcular el parámetro** | **Valor (Unidad)** |
| Masa [m] | Se calcula con ayuda de una gramera y se confirma mediante Solidworks. | 0.17244 [Kg] |
| Momento de inercia de la barra [Jl] | Solidworks ofrece un valor de momento de inercia con respecto al eje de origen, mediante el teorema de ejes paralelos se calcula el momento de inercia con respecto al eje de giro. | 8.403161000000001e-05 [kg\*m^2] |
| Distancia al centro [d] | Solidworks ofrece la distancia de origen hasta el cento de masa, a este valor le restamos la distancia del origen al eje de giro y obtenemos el valor real de centro de masa con respecto al eje de giro. | 0.04762 [m] |
| Fricción viscosa [Bl] | Este parámetro no ejerce un impacto directo en el sistema debido a su proximidad a cero. Además, en la ecuación diferencial, se encuentra dividido por n al cuadrado, lo que reduce aún más su valor. | 0 [Kg/s] |

* ¿Cuáles parámetros consideran que presentan mayor incertidumbre? Justificar la respuesta.

|  |  |
| --- | --- |
| **Análisis de los resultados** | Para el motor, entre los parámetros que pueden presentar mayor incertidumbre, primero mencionamos la resistencia de la armadura del motor. En las pruebas realizadas con el motor en funcionamiento al medir, puede haber alguna variación. Además, el motor incluye una placa soldada en su circuito, lo que podría influir en la medida.    Por otro lado, la constante mecánica del tiempo también genera incertidumbre. Dado que la medición se realiza con un osciloscopio, calcular el tiempo exacto resulta difícil, especialmente cuando se trata de órdenes de milisegundos.    Teniendo en cuenta lo anterior, los demás parámetros, que no pueden medirse directamente con una herramienta y deben calcularse mediante fórmulas, son los que generan mayor incertidumbre. Estos se basan en medidas que ya de por sí poseen cierto margen de incertidumbre, y se suma a ello el factor de las cifras significativas. Entre estos, los más críticos son la inercia del motor y las constantes electromotriz y de par.    En cuanto a la barra, al utilizar un tornillo disponible del cual no se dispone mucha información y no se puede verificar la exactitud de sus procesos de fabricación, los dos parámetros que generan mayor incertidumbre son el centro de gravedad y el momento de inercia, siendo este último dependiente del primero. |

1. Programar en un sistema embebido el algoritmo para generar la señal PWM y, registrar la posición angular de la barra y la velocidad angular del eje del motor , alrededor del punto de equilibrio. Justificar la selección del periodo de muestreo para la toma de datos y los métodos para calibrar la lectura registrada por los sensores. Anexar la respuesta transitoria de posición y la velocidad angular de la barra, dada una señal PWM variable. A partir de los datos registrados, identificar la zona muerta del sistema.

|  |  |
| --- | --- |
| **Periodo de muestreo (Justificar la selección del periodo)** | Se utiliza el diagrama de Bode para seleccionar la frecuencia de corte a -3 dB, una década después, con el fin de determinar el periodo máximo de muestreo permitido por la dinámica del sistema antes de la estimación de parámetros. Esto se hace con el propósito de establecer un valor inicial de periodo de muestreo que garantice una adquisición adecuada de datos para llevar a cabo el laboratorio en condiciones óptimas.    Tm = 0.104371848956472 [s]  Considerando que el sistema embebido utilizado permite periodos de muestreo más bajos, hemos definido un periodo de muestreo de Tm = 0.03 para lograr una mayor resolución en los datos obtenidos.    Se observa que el periodo de muestreo presenta pequeñas variaciones alrededor de 0.030, dentro de un rango de +/-, lo cual se considera normal. Esto garantiza el periodo de muestreo definido con anterioridad. |
| **Punto de equilibrio** |  |
| **Metodología para la calibración de los sensores (posición y velocidad angular)** | Para la calibración de la velocidad, se aplicaron distintos voltajes utilizando un tacómetro. Esta calibración se llevó a cabo con la colaboración de otro grupo de laboratorio para obtener un valor de revoluciones por minuto (RPM) estandarizado. Después de esta calibración, se procedió a realizar la medición de velocidad en el sistema embebido, teniendo en cuenta la relación de engranajes. De esta manera, se verificó que el sistema embebido coincidiera con los datos proporcionados por el tacómetro.    Posteriormente, se procede a calibrar la posición del sistema embebido utilizando un compás. Esta calibración se realiza de manera análoga, ya que no se dispone de herramientas más precisas para llevar a cabo el proceso. Se efectuó la calibración para los ejes principales, es decir, cada 90 grados, ya que este método se considera más fiable para garantizar la precisión de la calibración. |
| **Señal de PWM** |  |
| **Señal de posición** |  |
| **Señal de velocidad** |  |
| **Análisis de los resultados** | Una vez que los sensores están calibrados, se procede a programar en el sistema embebido una señal de control alternante entre valores positivos y negativos con el fin de generar una oscilación alrededor del punto de equilibrio.    Para lograr esto, se genera una señal de control con el mismo ancho de pulso, pero con distintos valores, con el objetivo de explorar la zona muerta y de saturación del péndulo.    Se puede concluir que los pulsos de ancho modulado (PWM) menores al 25% se consideran zona muerta, ya que el motor no puede superar la inercia y no se registran valores de posición ni velocidad. Respecto a la zona de saturación, no se logra percibir debido a que a valores de PWM mayores al 55%, el péndulo comienza a girar de manera descontrolada, lo cual se aleja del objetivo inicial de la toma de datos.    Para un PWM del 25%, se observa en la señal de posición y velocidad que los primeros datos son ligeramente menores que los posteriores, ya que el péndulo parte desde el reposo y la potencia es apenas suficiente para romper la inercial.    En cuanto a los valores de PWM superiores al 25%, se nota que al cambiar el PWM, los primeros datos son ligeramente mayores, debido a que el péndulo ya está en movimiento y al aumentar el voltaje del motor, adquiere mayor impulso. Posteriormente, se estabiliza, como se puede observar en las gráficas. |

1. Ajustar los parámetros del modelo matemático del sistema, para lo cual deben seguir la siguiente metodología:

* Implementar en simulink el modelo no lineal. Fijar como punto de equilibrio el determinado en el numeral cinco.

|  |  |
| --- | --- |
| **Código del bloque de Matlab function con el modelo no lineal** | function dx = fcn(x,PWM,parametros)    %ESTADOS  x1 = x(1);  x2 = x(2);  x3 = x(3);    %parámetros  R = parametros(1);  L = parametros(2);  Ki = parametros(3);  Kv = parametros(4);  Jm = parametros(5);  Jl = parametros(6);  Bm = parametros(7);  Bl = parámetros(8);  n = parametros(9);  g = parametros(10);  d = parametros(11);  m = parametros(12);    % MODELO 2 ESTADOS I TL WL SALIDA TL WL  f1 = (((7\*PWM)/100)-Kv\*x3\*n-R\*x1)/(L); %di/dt  f2 = x3; %d0l/dt = Wl  f3 = ((Ki\*x1\*n)-x3\*(Bm\*n^2+Bl)-(m\*g\*d\*sin(x2)))/(Jl+Jm\*n^2); %dwl/dt    dx = [f1; f2; f3]; |

* Generar dos señales PWM aleatorias, evitando la zona muerta y la zona de saturación del sistema y, registrar la posición angular de la barra y la velocidad angular del eje del motor . El primer grupo de datos experimentales se denominan *datos de identificación* y el segundo grupo *datos de validación.*

|  |  |
| --- | --- |
| **Datos de identificación** | |
| **Señal PWM** |  |
| **Posición angular de la barra** |  |
| **Velocidad angular** |  |
| **Datos de validación** | |
| **Señal PWM** |  |
| **Posición angular de la barra** |  |
| **Velocidad angular** |  |

* Ajustar los parámetros del modelo matemático utilizando la toolbox *Parameter Estimation*, de forma que la respuesta transitoria simulada coincida con los datos experimentales, para los dos grupos de señales registrados en el numeral anterior. Asuma como valores iniciales de los parámetros los calculados en el numeral cuatro. Los parámetros que consideran conocidos, los deja fijos al ejecutar el algoritmo de la toolbox *Parameter Estimation*.

|  |  |  |
| --- | --- | --- |
| **Parámetros iniciales del modelo matemático** |  | |
| **Parámetros finales del modelo matemático** |  | |
| **Comparación de los datos simulados (modelo no lineal) con los datos de identificación** | **Posición angular de la barra** | Amarillo: Datos experimentales  Azul: Datos simulados  Eje X: Tiempo [s]  Eje Y: Grados [rads] |
| **Velocidad angular** | Amarillo: Datos simulados  Azul: Datos experimentales  Eje X: Tiempo [s]  Eje Y: Velocidad [rads/s] |
| **Comparación de los datos simulados (modelo no lineal) con los datos de validación** | **Posición angular de la barra** | Amarillo: Datos experimentales  Azul: Datos simulados  Eje X: Tiempo [s]  Eje Y: Grados [rads] |
| **Velocidad angular** | Amarillo: Datos simulados  Azul: Datos experimentales  Eje X: Tiempo [s]  Eje Y: Velocidad [rads/s] |
| **Análisis de los resultados** | En resumen, podemos concluir que la estimación de parámetros fue exitosa, ya que proporcionó valores consistentes con la realidad. Además, para un PWM del 50%, el modelo matemático ajustado logra que los datos simulados se alineen con los datos experimentales. No obstante, dependiendo de la complejidad del modelo matemático, podemos obtener mejores o peores resultados simulados. Por lo tanto, si deseamos mejorar la simulación en la zona muerta, sería necesario incluir factores como las fricciones en el modelo.  Como mencionamos anteriormente, la estimación de parámetros se realizó alrededor del 50% de PWM. Al comparar los datos experimentales con los simulados durante la identificación de parámetros, las gráficas son prácticamente idénticas, con algunas variaciones que se consideran normales.    Sin embargo, al analizar la comparativa de datos de validación, llevada a cabo alrededor del 30% de PWM, se observa un alto porcentaje de error entre los datos experimentales y los simulados, generando un error del 50%. Esto se debe a la naturaleza del modelo matemático, el cual no tiene en cuenta factores como las fricciones del sistema, entre otros. Por esta razón, al acercarnos a la zona muerta del sistema, los datos simulados se distancian de los experimentales.    Al simular los datos a diferentes PWM, se evidencia que la simulación genera valores de posición y velocidad en la zona muerta, lo cual no ocurre en el experimento real, como se mencionó anteriormente. Sin embargo, al acercarse a la zona operativa óptima, los datos simulados empiezan a ajustarse a los datos experimentales.    Amarillo: Datos experimentales  Azul: Datos simulados  Eje X: Tiempo [s]  Eje Y: Grados [rads] | |

1. Determinar la respuesta transitoria del modelo lineal ante una entrada escalón unitario y el diagrama de bode. De acuerdo con las gráficas:

* ¿La respuesta transitoria ante una entrada escalón corresponde a la dinámica del sistema?
* ¿El periodo de muestreo seleccionado para la toma de datos fue el adecuado?

|  |  |
| --- | --- |
| **Representación en espacio de estados del modelo lineal** |  |
| **Respuesta transitoria ante una entrada escalón (Indicar los parámetros transitorios más importantes de cada señal)** | |  |  | | --- | --- | | **RESPUESTA TRANSTORIA POSICIÓN** | | | Sobrepaso | 53.8% | | Tiempo levantamiento | 0.242 [s] | | Tiempo establecimiento | 4 [s] | | Valor estado estable | 0.199 [rad] |  |  |  | | --- | --- | | **RESPUESTA TRANSITORIA VELOCIDAD** | | | Amplitud MAX | 0.757 [rad] | | Tiempo levantamiento | - | | Tiempo establecimiento | 4.28 [s] | | Valor estado estable | 0 [rad/s] | |
| **Diagrama de bode (Indicar la frecuencia de , el margen de fase y el margen de ganancia)** | |  |  | | --- | --- | | **BODE POSICIÓN** | | | W-3dB | -17 [dB] | | Frecuencia -3dB | 7.56 [rad/s] |  |  |  | | --- | --- | | **BODE VELOCIDAD** | | | W-3dB | 5.29 [dB] | | Frecuencia -3dB | 6.03 [rad/s] | |
| **Análisis de los resultados** | **¿La respuesta transitoria ante una entrada escalón corresponde a la dinámica del sistema?**  Si corresponde, al ajustar los parámetros, la respuesta transitoria ante una entrada escalón mantiene la misma forma, pero con más oscilaciones. Al realizar la prueba en lazo abierto, observamos que al simular una entrada escalón del 30% de PWM, obtenemos una salida de posición de 0.41 [rads] (23.49 [grados]). Este resultado concuerda con lo observado experimentalmente, donde la barra se mantiene en una posición de 27.09 [grados] y con una velocidad de 0.    Amarillo: Datos simulados  Eje X: Tiempo [s]  Eje Y: Grados [rads]    Amarillo: Datos simulados  Eje X: Tiempo [s]  Eje Y: Velocidad [rads/s]      [Tiempo, Velocidad, Grados, PWM]  [Datos obtenidos de arduino]  **¿El periodo de muestreo seleccionado para la toma de datos fue el adecuado?**  Basándonos en el diagrama de Bode con los parámetros estimados, observamos que la salida crítica en este caso es la posición. Por lo tanto, calculamos el periodo de muestreo (Tm) utilizando su frecuencia de corte (W-3dB), de la siguiente manera:    Anteriormente, habíamos seleccionado un periodo de muestreo de Tm = 0.030, el cual consideramos adecuado para la dinámica del sistema. En este caso, no es necesario realizar ninguna corrección, ya que el periodo de muestreo seleccionado sigue siendo apropiado para las necesidades del sistema. |

1. Establecer los parámetros del modelo matemático que presentan mayor incertidumbre. Asumir una tolerancia del en los parámetros y realizar un análisis de la forma como se ve afectada los parámetros transitorios y de frecuencia del sistema.

|  |  |
| --- | --- |
| **Parámetros con mayor incertidumbre (Justificar la respuesta)** | Incertidumbre en Rm:    Incertidumbre en Lm:    Incertidumbre en Jm:    Incertidumbre en Kim:    Incertidumbre en Kvm:    Incertidumbre en Bm:    Incertidumbre en Jl:    Incertidumbre en Bl: |
| **Respuesta transitoria ante una entrada escalón** | Anteriormente, realizamos variaciones individuales en cada parámetro. Ahora, llevaremos a cabo cambios simultáneos en todos los parámetros para determinar la máxima y mínima variación posible en las respuestas transitorias del sistema. |
| **Variación en el sobrepaso, tiempo de levantamiento, tiempo de establecimiento y valor en estado estable** | |  |  |  |  | | --- | --- | --- | --- | | **RESPUESTA TRANSTORIA POSICIÓN** | | | | |  | ESTIMADO | MINIMO | MAXIMO | | Sobrepaso | 53.8% | 49.9% | 58% | | Tiempo levantamiento | 0.242 [s] | 0.241 [s] | 0.249 [s] | | Tiempo establecimiento | 4 [s] | 3.26 [s] | 4.76 [s] | | Valor estado estable | 0.199 [rad] | 0.166 [rad] | 0.236 [rad] |  |  |  |  |  | | --- | --- | --- | --- | | **RESPUESTA TRANSTORIA VELOCIDAD** | | | | |  | ESTIMADO | MINIMO | MAXIMO | | Amplitud MAX | 0.757 [rad/s] | 0.634 [rad/s] | 0.869 [rad/s] | | Tiempo levantamiento | - | - | - | | Tiempo establecimiento | 4.28 [s] | 3.53 [s] | 5.37 [s] | | Valor estado estable | 0 [rad/s] | 0 [rad/s] | 0 [rad/s] |   Nota: Cuando hablamos de “mínimo” y “máximo” en las tablas hacemos referencia al rango de valores que puede tomar por la incertidumbre de los parámetros. |
| **Diagrama de bode** |  |
| **Variación en la frecuencia de , margen de fase y margen de ganancia** | |  |  |  |  | | --- | --- | --- | --- | | **BODE POSICIÓN** | | | | |  | ESTIMADO | MINIMO | MAXIMO | | W-3dB | -17 [dB] | -18.3 [dB] | -16 [db] | | Frecuencia -3dB | 7.56 [rad/s] | 7.43 [rad/s] | 7.71 [rad/s] |  |  |  |  |  | | --- | --- | --- | --- | | **BODE VELOCIDAD** | | | | |  | ESTIMADO | MINIMO | MAXIMO | | W-3dB | 5.29 [dB] | 4.51 [dB] | 5.99 [db] | | Frecuencia -3dB | 6.03 [rad/s] | 5.91 [rad/s] | [6.27] [rad/s] |   Nota: Cuando hablamos de “mínimo” y “máximo” en las tablas hacemos referencia al rango de valores que puede tomar por la incertidumbre de los parámetros. |
| **Análisis de los resultados** | De todos los parámetros que se pusieron a variar con 10% de tolerancia se observa como los menos influyentes en la respuesta transitoria ante una entrada escalón son Jl, Bl, Bm y Lm.  Se confirma que la fricción viscosa de la barra no afecta significativamente la respuesta transitoria dado que el valor es cero o cercano a cero.  Por otra parte, se esperaba que la fricción viscosa del motor influyera significativamente en la respuesta transitoria dado que este parámetro se calcula matemáticamente teniendo (Jm, Tm, Ki, Kv y R).  Por último, se confirma que los parámetros que más variación generan son Rm, Kim, Kvm y Jm, son los valores que más influyen en la respuesta transitoria del sistema.  A razón de estudio se pone a variar el parámetro de la masa de la barra para observar cómo influye en la respuesta transitoria, se observa que, de los anteriores parámetros estudiados, la masa es el más crítico. Por último, queda estudiar la razón por la cual la inercia de la barra no afecta significativamente cuando la masa si lo hace. |

1. Indicar las conclusiones más importantes del desarrollo de la práctica de laboratorio (máximo 200 palabras).

|  |
| --- |
| **Conclusiones** |
| En la programación del sistema embebido para la lectura del enconder hall, se encontraron nuevas complejidades en comparación con el semestre pasado. Esta vez, se trabajó con dos interrupciones y se prestó especial atención a la dirección de giro al leer la velocidad y la posición, esto fue crucial para sincronizar las gráficas, evitando que una quedara desfasada con respecto a la otra.    Durante el laboratorio, se destacó la importancia de la medición de los parámetros del sistema. Aunque MATLAB ofrece herramientas para estimar y ajustar estos parámetros a los datos experimentales, es fundamental proporcionar valores iniciales precisos para que el proceso de optimización funcione de manera óptima. De lo contrario, el programa de optimización puede generar estimaciones no ideales.    Se evidenció que el modelo matemático no considera parámetros importantes como las fricciones, que son relevantes en el mundo real y pueden generar lo que se conoce como "zona muerta". Esto significa que las simulaciones y los datos experimentales pueden no coincidir hasta que el sistema opere dentro de una zona óptima.    Se confirmo experimentalmente y simulado que, ante una entrada escalón en lazo abierto en el modelo lineal, en este caso 30% de PWM, el péndulo lograba romper la inercia, pero no era lo suficiente para empezar a girar de manera descontrolada, por otro lado, el ángulo al que llegaba con este PWM en la prueba experimental no siempre era el mismo (pero era próximo al simulado), se evidencia la necesidad de un controlador en lazo cerrado para garantizar un ángulo deseado. Esto será objeto de estudio más adelante.    Por último, es importante recalcar la relevancia del análisis de incertidumbres en la aplicación. Esta herramienta permite identificar los parámetros más críticos del modelo y prestar especial atención a ellos al tomar medidas o confirmarlas. Esto conduce a un modelo matemático más preciso y confiable. |

**Anexos.**

|  |  |
| --- | --- |
| **Código del sistema embebido** | const byte Encoder\_C1 = 2; // Cable amarillo pin 3 digital  const byte Encoder\_C2 = 3; // Cable verde al pin 4 digital  byte Encoder\_C1Last;  int paso = 0;  boolean direccion, flagPWM, autoincrement;    int entrada3 = 13;  int entrada4 = 12;  int enableB = 11;    int maxPWM = 130; // VCC 7V    int userE, arduinoPWM;    const int tMuestro = 30;  unsigned long tiempoDatos = 3000;    volatile int n = 0;  volatile byte ant = 0;  volatile byte act = 0;    double P = 0;  double R = 3840;    float rpm = 0;  unsigned long timeold = 0;  unsigned long timeoldPWM = 0;  unsigned long startTime = 0;  unsigned long elapsedTime = 0;      void setup()  {  Serial.begin(9600);  pinMode(entrada3, OUTPUT);  pinMode(entrada4, OUTPUT);  pinMode(enableB, OUTPUT);  pinMode(Encoder\_C2, INPUT);  flagPWM = false;  autoincrement = false;  arduinoPWM = 5;    attachInterrupt(digitalPinToInterrupt(Encoder\_C1), calculapulso, CHANGE);  attachInterrupt(digitalPinToInterrupt(Encoder\_C2), calculapulso, CHANGE);  }    void loop()  {  // digitalWrite(entrada3, HIGH);  // digitalWrite(entrada4, LOW);  //  // analogWrite(enableB, map(25, 0, 100, 0, maxPWM)); // 37.5  // analogWrite(enableB, map(50, 0, 100, 0, maxPWM)); // 129        if (Serial.available()) { // Verificar si hay datos disponibles en el puerto serial  userE = Serial.parseInt(); // Leer el valor entero enviado por el puerto serial  // Verificar si el valor recibido está dentro del rango válido (0-100)  if (userE > 5 && userE != 101 ) autoincrement = true;    if (userE == 101) {  autoincrement = false;  startTime = 0; // Reiniciar el tiempo de inicio si autoincrement es false  elapsedTime = 0;  timeold = 0;  timeoldPWM = 0;  tiempoDatos = 3000;  rpm = 0;  P = 0;  analogWrite(enableB, 0); // Equivalente a 50% del máximo (128 de 255)    }    }    if (autoincrement == true) {  if (startTime == 0) {  startTime = millis(); // Registrar el tiempo de inicio  }    elapsedTime = millis() - startTime; // Calcular el tiempo transcurrido      if (elapsedTime > 3000) {    // PWM FIJO  // if (elapsedTime == 3001) arduinoPWM = 10; // PWMX60x80  arduinoPWM = 30;        // if (elapsedTime > tiempoDatos) {  // arduinoPWM = arduinoPWM + 10; // PWMX60x80  // tiempoDatos = tiempoDatos + 10500;  // }      // if (elapsedTime == 3001) arduinoPWM = 10; // PWMX60x80  // if (elapsedTime == 3001) arduinoPWM = 15; // PWMX60x80  // if (elapsedTime == 3001) arduinoPWM = 20; // PWMX60xx80  // if (elapsedTime == 3001) arduinoPWM = 25; // PWMX60x80  // if (elapsedTime == 3001) arduinoPWM = 30; // PWMX60x80  // if (elapsedTime == 3001) arduinoPWM = 35; // PWMX60x80  // if (elapsedTime == 3001) arduinoPWM = 40; // PWMX60x80  // if (elapsedTime == 3001) arduinoPWM = 45; // PWMX60x80  // if (elapsedTime == 3001) arduinoPWM = 50; // PWMX60x80  // if (elapsedTime == 3001) arduinoPWM = 55; // PWMX60x80  // if (elapsedTime == 3001) arduinoPWM = 60; // PWMX60x80          if (millis() - timeoldPWM >= 700) {  // Cambia la dirección del motor y establece la velocidad  if (flagPWM) {  digitalWrite(entrada3, LOW);  digitalWrite(entrada4, HIGH);    } else {    digitalWrite(entrada3, HIGH);  digitalWrite(entrada4, LOW);  }  analogWrite(enableB, map(arduinoPWM, 0, 100, 0, maxPWM));    // Invierte el estado de flagPWM para la próxima iteración  flagPWM = !flagPWM;    // Actualiza el tiempo anterior  timeoldPWM = millis();  }      if (millis() - timeold >= tMuestro) {  rpm = (paso \* (0.100 / (millis() - timeold)) \* (60000 / 360))\*-1 ;  P = (n \* 360.0) / R;  impresionSerial();  if (flagPWM) {  Serial.print(arduinoPWM); Serial.print("\n");  } else {  Serial.print("-"); Serial.print(arduinoPWM); Serial.print("\n");    }  timeold = millis();  paso = 0;  }    if (elapsedTime >= 66000 || arduinoPWM > 55 ) {  autoincrement = false;  startTime = 0; // Reiniciar el tiempo de inicio si autoincrement es false  elapsedTime = 0;  timeold = 0;  timeoldPWM = 0;  tiempoDatos = 3000;  rpm = 0;  P = 0;  analogWrite(enableB, 0); // Equivalente a 50% del máximo (128 de 255)  }  }        }  }    void impresionSerial() {  Serial.print(elapsedTime ); Serial.print(", "); Serial.print(rpm ); Serial.print(", "); Serial.print(P ); Serial.print(", ");    }    void calculapulso() {    int Lstate = digitalRead(Encoder\_C1);  if ((Encoder\_C1Last == LOW) && Lstate == HIGH)  {  int val = digitalRead(Encoder\_C2);  if (val == LOW && direccion)  {  direccion = false; //Reverse  }  else if (val == HIGH && !direccion)  {  direccion = true; //Forward  }  }  Encoder\_C1Last = Lstate;    if (!direccion) paso++;  else paso--;      ant = act;    if (digitalRead(Encoder\_C1)) bitSet(act, 1); else bitClear(act, 1);  if (digitalRead(Encoder\_C2)) bitSet(act, 0); else bitClear(act, 0);    } |