Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ingeniería Eléctrica
IEE2613 - Control Automático 1-2023

PROYECTO

Cañón Contra Incendios

10 de Mayo de 2023

Juan Ignacio Lorca

---

1. Modelación

Para la modelación del sistema se utilizaron las ecuaciones entregadas por el profesor Miguel Torres. Estas fueron usadas en la simulación en python que pueden encontrar en la carpeta modelo en el archivo sim.py que contiene el código de la simulación.

1.1. Modelación movimiento carro y cañón

Con tal de indentificar nuestras variables controladas, manipuladas, de referencia y las perturbaciones, primero debemos encontrar las ecuaciones diferenciales que modelan el sistema. Para la modelación del sistema se utilizó el principio de Lagrange:

$$\frac{d}{dt}\frac{\partial \mathcal{L}(\mathbf{q},\dot{\mathbf{q}})}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}(\mathbf{q},\dot{\mathbf{q}})}{\partial q_i} = F_{q_i} \tag{1}$$

Con las respectivas variables de configuración:

$$q_i = \{q_1, q_2, ..., q_n\},$$

donde $\mathcal{L}(\mathbf{q},\dot{\mathbf{q}})$ es el Lagrangiano del sistema y se define como:

$$\mathcal{L}(\mathbf{q},\dot{\mathbf{q}}) = E_K(\mathbf{q},\dot{\mathbf{q}}) - E_P(\mathbf{q},\dot{\mathbf{q}})$$

En términos de la energía cinética $E_K(\mathbf{q},\dot{\mathbf{q}})$ y la energía potencial $E_P(\mathbf{q},\dot{\mathbf{q}})$.

Utilizando las coordenadas generalizadas:

$$\mathbf{q} = \begin{bmatrix} x \\ \theta \end{bmatrix} \tag{2}$$

y sus derivadas:

$$\dot{\mathbf{q}} = \begin{bmatrix} \dot{x} \\ \dot{\theta} \end{bmatrix} \tag{3}$$

De la figura (1) podemos calcular las posiciones de los centros de masa tanto para el carro y para el cañón. Notar que el ángulo θ se encuentra referido a una línea imaginara horizontal respecto a un cañón que se encuetra apuntando en las direcciones positivas (x, z).

$$\mathbf{p}_x = (x, 0, H),$$

$$\mathbf{p}_1 = \mathbf{p}_x + \frac{l_c}{2}(\cos(\theta), 0, \sin(\theta))$$

y sus derivadas:

$$\dot{\mathbf{p}}_x = (\dot{x}, 0, 0),$$

$$\dot{\mathbf{p}}_1 = \dot{\mathbf{p}}_x + \frac{l_c}{2}(-\sin(\theta)\dot{\theta}, 0, \cos(\theta)\dot{\theta})$$

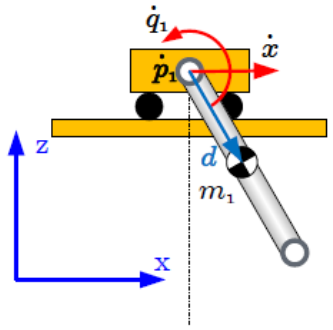


Figura 1: Modelo carro y cañón

De esta manera, siguendo los apuntes del profesor, las ecuaciones para la energía cinética:

$$E_K(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2}m_x\dot{\mathbf{p}}_x^T\dot{\mathbf{p}}_x + \frac{1}{2}m_c\dot{\mathbf{p}}_1^T\dot{\mathbf{p}}_1 + \frac{1}{2}{\omega_\mathbf{c}}^T\mathbf{J}_\mathbf{c}\omega_\mathbf{c},$$

tal que las normas al cuadrado de los vectores velocidad están dadas por:

$$\dot{\mathbf{p}}_x^T\dot{\mathbf{p}}_x = \dot{x}^2,$$

$$\dot{\mathbf{p}}_1^T\dot{\mathbf{p}}_1 = \dot{x}^2 - l_c\sin(\theta)\dot{\theta}\dot{x} + \frac{l_c^2}{4}\dot{\theta}^2$$

y la energía potencial:

$$E_P(\mathbf{q}, \dot{\mathbf{q}}) = m_x g H + m_c g(H + \frac{l_c}{2}\sin(\theta))$$

Donde $\omega_\mathbf{c} = \dot{\theta}$ y la masa de la bombarda m_b junto con su velocidad angular no fueron consideradas porque $\mathbf{J}_b = 0$ y la masa será utilizada en una modelación aparte. Además, el momento de inercia de la barra utilizado será $\mathbf{J_c} = m_c\frac{l_c^2}{12}$, lo que se debe a que estamos calculando las energías en los centros de masas presentes en el sistema.

Finalmente, el Lagrangiano:

$$\mathcal{L}(\mathbf{q},\dot{\mathbf{q}}) = \frac{1}{2}(m_x + m_c)\dot{x}^2 - \frac{1}{2}m_c l_c \sin(\theta)\dot{\theta}\dot{x} + \frac{l_c^2}{6}m_c\dot{\theta}^2 - m_c g\frac{l_c}{2}\sin(\theta) \tag{4}$$

Notamos que no se agregaron los términos $m_x gH$ y $m_c gH$ puesto que la altura H es una constante, por lo que para efectos del cálculo de (1) no es determinante.

Con los términos anteriores es posible calcular:

$$\frac{d}{dt}\frac{\partial\mathcal{L}(\mathbf{q},\dot{\mathbf{q}})}{\partial\dot{x}} = (m_x + m_c)\ddot{x} - \frac{1}{2}m_c l_c\cos(\theta)\dot{\theta}^2 - \frac{1}{2}m_c l_c\sin(\theta)\ddot{\theta},$$

$$\frac{d}{dt}\frac{\partial\mathcal{L}(\mathbf{q},\dot{\mathbf{q}})}{\partial\dot{\theta}} = -\frac{1}{2}m_c l_c\sin(\theta)\ddot{x} - \frac{1}{2}m_c l_c\cos(\theta)\dot{\theta}\dot{x} + \frac{l_c^2}{3}m_c\ddot{\theta}$$

Similarmente:

$$\frac{\partial\mathcal{L}(\mathbf{q},\dot{\mathbf{q}})}{\partial x} = 0,$$

$$\frac{\partial\mathcal{L}(\mathbf{q},\dot{\mathbf{q}})}{\partial\theta} = -\frac{1}{2}m_c l_c\cos(\theta)\dot{\theta}\dot{x} - m_c g\frac{l_c}{2}\cos(\theta)$$

Además, podemos modelar las fuerzas que actúan sobre estas variables como F_x y F_θ, y las resistencias viscosas que ejerce el aire sobre las variables x y θ. Luego las ecuaciones diferenciales de acuerdo al principio de lagrange quedan:

$$(m_x + m_c)\ddot{x} - m_c\frac{l_c}{2}\cos(\theta)\dot{\theta}^2 - m_c\frac{l_c}{2}\sin(\theta)\ddot{\theta} = F_x - c\dot{x} \tag{5}$$

$$-m_c\frac{l_c}{2}\sin(\theta)\ddot{x} + \frac{l_c^2}{3}m_c\ddot{\theta} + m_c\frac{l_c}{2}g\cos(\theta) = F_\theta - b\dot{\theta} \tag{6}$$

1.2. Identificación de variables

- **Variables Controladas Sistema Carro-Cañón:** $CV = (x, \theta)$; posición horizontal sistema carro más cañón y ángulo de rotación del cañón. Estas variables serán la salida **y** del sistema.
- **Variables Manipuladas Sistema Carro-Cañón:** $MV = (F_x, F_\theta)$; fuerza motor sistema carro más cañón y torque motor rotativo del cañón.
- **Variables de Referencia Sistema Carro-Cañón (Set Points):** $SP = (x_{sp}, z_{sp})$; el vector $\mathbf{r} = (x_{sp}, z_{sp})$ nos entregará la posición relativa del fuego respecto a la posición de salida de la bombarda de agua. Este vector me entragará la tupla (x_{sp}, θ_{sp}) para controlar el sistema carro-cañón.
- **Perturbaciones Sistema Carro-Cañón:** $DV = (c\dot{x}, b\dot{\theta})$; el roce aerodinámico que afectará tanto al movimiento horizontal del carro, a la rotación del cañón y a la trayectoria de la bombarda.

1.3. Punto de equilibrio

Con tal de identificar los puntos de equilibro u operación del sistema, primero se realizará una representación en el espacio de los estados:

1. Vector de estado: $\mathbf{x}$

$$\mathbf{x} = \begin{bmatrix} x \\ \theta \\ \dot{x} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

2. Vector de controles: $\mathbf{u}$

$$\mathbf{u} = \begin{bmatrix} F_x \\ F_\theta \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

3. Vector de variables controladas: $\mathbf{y} = \mathbf{g}(\mathbf{x}, \mathbf{u})$

$$\mathbf{y} = \begin{bmatrix} x \\ \theta \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

Ahora, para efectos de la linealización y definición de ecuaciones de estado definimos la ecuación de estado: $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$

$$\mathbf{f}(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \end{bmatrix}$$

$$\dot{x}_1 = x_3 = \dot{x}$$

$$\dot{x}_2 = x_4 = \dot{\theta}$$

$$\dot{x}_3 = \ddot{x} = \frac{4l_c(F_x - c\dot{x} + \frac{l_c}{2}m_c\cos(\theta)\dot{\theta}^2) + 6(F_c - b\dot{\theta} - g\frac{l_c}{2}m_c\cos(\theta))\sin(\theta)}{l_c(3m_c\cos^2(\theta) + m_c + 4m_x)}$$

$$\dot{x}_4 = \ddot{\theta} = \frac{6(l_c m_c(F_x - c\dot{x} + \frac{l_c}{2}\cos(\theta)\dot{\theta}^2)\sin(\theta) + 4(m_c + m_x)(F_c - b\dot{\theta}g\frac{l_c}{2}m_c\cos(\theta)))}{l_c^2 m_c(3m_c\cos^2(\theta) + m_c + 4m_x)}$$

El equilibrio del sistema se da cuando no hay un cambio de las variables de estado en el tiempo. Entonces definiendo el punto de equilibrio u operación cómo $(\mathbf{x}^e, \mathbf{u}^e)$, tenemos que:

$$\mathbf{f}(\mathbf{x}^e, \mathbf{u}^e) = \mathbf{0} \tag{7}$$

Luego, reemplazando $(\dot{x}, \dot{\theta}, \ddot{x}, \ddot{\theta}) = (0, 0, 0, 0)$ en (5) y (6) se tiene que el punto de equilibrio viene dado por:

$$\mathbf{x}^e = \begin{bmatrix} x_e \\ \theta_e \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \qquad \mathbf{u}^e = \begin{bmatrix} 0 \\ m_c g\frac{l_c}{2}\cos(\theta_e) \end{bmatrix} = \begin{bmatrix} 0 \\ m_c g\frac{l_c}{2} \end{bmatrix}$$

Donde (x_e, θ_e) es punto genérico para la posición horizontal del carro y el ángulo de elevación del cañón.

1.4. Linealización

Buscamos linealizar el modelo de la forma:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) = A\mathbf{x} + B\mathbf{u}$$

$$\mathbf{y} = \mathbf{g}(\mathbf{x}, \mathbf{u}) = C\mathbf{x} + D\mathbf{u}$$

Luego, matrices $A,\ B,\ C,\ D$ se calculan de la siguiente forma:

$$A = \frac{\partial \mathbf{f}(\mathbf{x}^e, \mathbf{u}^e)}{\partial \mathbf{x}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{3gm_c \sin^2(\theta_e)}{3m_c \cos^2(\theta_e)+m_c+4m_x} & \frac{3gm_c \sin^2(\theta_e)}{3m_c \cos^2(\theta_e)+m_c+4m_x} & -\frac{6c \sin(\theta_e)}{l_c(3m_c \cos^2(\theta_e)+m_c+4m_x)} \\ 0 & -\frac{3g(-3m_c-2m_x) \sin(\theta_e)}{l_c(3m_c \cos^2(\theta_e)+m_c+4m_x)} & -\frac{6b \sin(\theta_e)}{l_c(3m_c \cos^2(\theta_e)+m_c+4m_x)} & \frac{6c(-2m_c-2m_x)}{l_c^2 m_c(3m_c \cos^2(\theta_e)+m_c+4m_x)} \end{bmatrix}$$

$$B = \frac{\partial \mathbf{f}(\mathbf{x}^e, \mathbf{u}^e)}{\partial \mathbf{u}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{4}{3m_c \cos^2(\theta_e)+m_c+4m_x} & \frac{6 \sin(\theta_e)}{l_c(3m_c \cos^2(\theta_e)+m_c+4m_x)} \\ \frac{6 \sin(\theta_e)}{l_c(3m_c \cos^2(\theta_e)+m_c+4m_x)} & \frac{12(m_c+m_x)}{l_c^2 m_c(3m_c \cos^2(\theta_e)+m_c+4m_x)} \end{bmatrix}$$

$$C = \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$D = \frac{\partial \mathbf{y}}{\partial \mathbf{u}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

1.5. Funciones de Transferencia

Con el sistema ya linealizado, podemos encontrar la función de transferencia utilizando la siguiente fórmula:

$$F(s) = \frac{Y(s)}{U(s)} = C[sI - A]^{-1}B + D \tag{8}$$

obteniendo:

$$X(s) = F_x \frac{4l_c^2 m_c s^2 + 12cs - 6gl_c m_c \sin(\theta_e)}{sq(s)} + F_c \frac{6l_c m_c \sin(\theta_e)s}{q(s)},$$

$$\Theta(s) = F_x \frac{6l_c m_c \sin(\theta_e)s}{q(s)} + F_c \frac{12(m_c + m_x)s + 12b}{q(s)}$$

con:

$$q(s) = l_c^2 m_c^2 (1 + 4\frac{m_x}{m_c} + 3\cos^2(\theta_e))s^3 + (12c(m_c + m_x) + 4bl_c^2 m_c)s^2$$
$$+6(2bc - gl_c m_c(m_c + m_x)\sin(\theta_e))s - 6bgl_c m_c \sin(\theta_e)$$

Podemos notar que tanto cambios en escalón de F_x o F_θ, perturban de manera curzada las variables θ y x, respectivamente, mostrando que realmente existe un acoplamiento entre las variables del sistema. Ahora, al linealizar en torno al punto de equilibrio, observamos que ambas variables controladas no dependen de las variables manipuladas cruzadas ya que al ser entradas de tipo escalón, se les aplica un derivador s que hace su efecto en tales variables sea nulo. De esta manera, para la implementación de un controlador básico sin acciones de desacoplamiento, es posible trabjar el diseño del control como dos subsistemas independientes que controlan el desplazamiento horizontal del carro y la elevación del cañón.

1.6. Diagrama esquemático

El diagrama esquemático **genérico** que representa al sistema descrito por las ecuaciones diferenciales en (5) y (6) en tiempo discreto es:

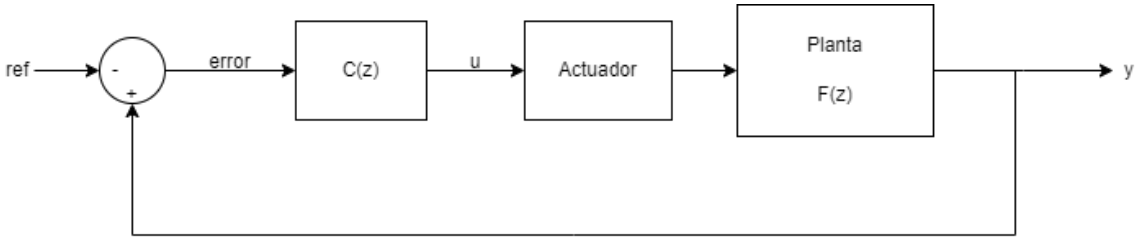


Figura 2: Diagrama esquemático

2. Simulación y Pruebas en Lazo Abierto

En el en la carpeta modelo dentro del archivo Entrega 3.zip podrán encontrar los códigos que llevaran a cabo la simulación. Para la programación de esta, el lenguaje *Python* fue utilizado. El script principal (que corre la simulación) tiene por nombre **sim.py**.

La modelación del sistema fue implementada en el script **model.py**. En él podrán encontrar la clase **Model** la cuál posee los atributos referidos a: el modelo del espacio de los estados, las referencias que tendrán las variables x y θ a la hora de hacer click en la pantalla o presionar la tecla "F" para la aparición de fuegos aleatorios, los errores que entregarán tales referencias, los modos manual y automático del sistema y, por último, las variables para almacenar la data entregada por la simulación. Cada uno de estos atributos se encuentra comentado en el código. Luego, este modelo posee de métodos que harán que la simulación funcione correctamente. Los métodos más destacados son: **update_state_model** que posee de una función que entrega las ecuaciones de cinemática del modelo y realiza los cálculos del vector de estado **x** por medio de la función **odeint**, **x_controller** y **theta_controller** que representan los controladores que manipulan los actuadores F_x y F_c del sistema, **shoot_projectile**, que entrega las condiciones iniciales para que el modelo genere bombaradas de agua y **update_model**, que actualiza el modelo de acuerdo al modo de funcionamiento manual o automático. Para el posicionamiento de los distintos elementos en la pantalla, la siguiente figura fue la referencia:

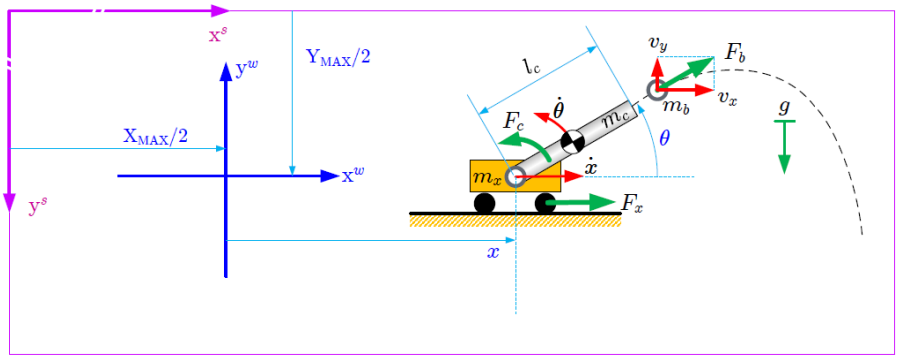


Figura 3: Referencia posicionamiento de elementos en **sim.py**

Además, en la carpeta podrán encontrar los scripts **fire.py**, **projectile.py** y **plot_data.py** que completan la simulación y **Projectile**, que también contempla una modelación cinemática similar a la utilizada en **Model**. También, el script **plot_data.py** permite tener una vizualización de los datos: variables controladas x y θ, variables manipuladas F_x y F_c, referencias y errores respecto a las variables controladas. El código está configurado para que los datos se guarden después de aproximadamente 100 segundos de simulación (5000 muestras). Por último, el programa posee de un **README.md** que contiene indicaciones sobre la manipulación del código.

2.1. Modelos y Diagrama de Flujo

El siguiente diagrama de flujo representa el funcionamiento y la relación entre scripts del programa:

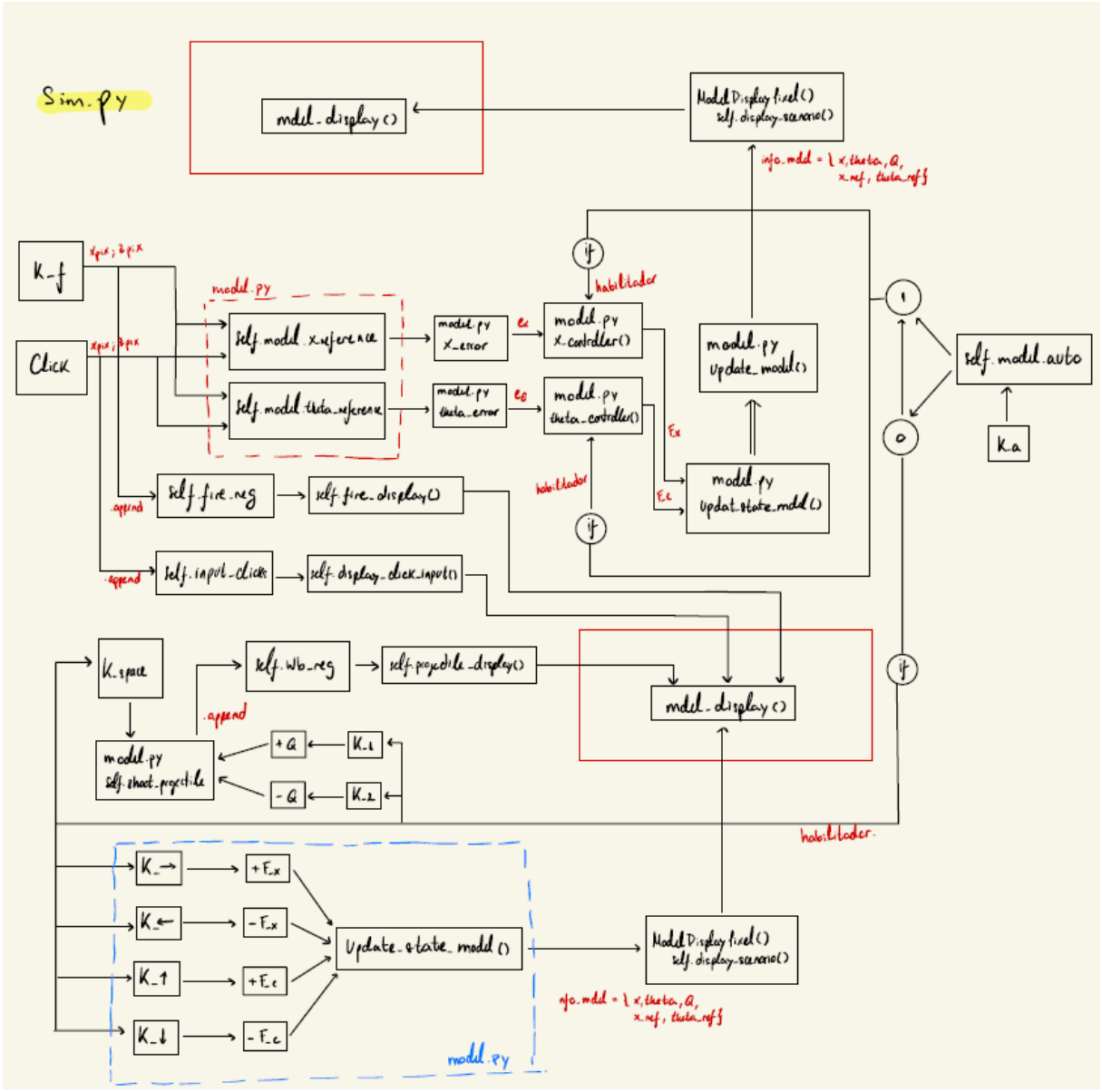


Figura 4: Diagrama de flujo script **sim.py**

3. Diseño del Controloador y Análisis de Desempeño

3.1. Controlabilidad y Observabilidad del Sistema

Para el desarrollo de este punto utilizaremos el modelo linealizado en el punto de equilibrio descrito en las secciones 1.3 y 1.4. Para ello analizaremos las mediciones hechas por el sensor para las variables de estado $(\dot{x}, \dot{\theta})$ y (x, θ).

Para determinar la controlabilidad del sistema necesitamos saber si nuestras variables controladas afectan al sistema al cerrar el lazo. De esta manera, existirá un vector de variables manipuladas o vector de controles:

$$\mathbf{u} = \begin{bmatrix} F_x \\ F_\theta \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

que actúen como señales que le permitan al sistema alcanzar cualquier estado en un tiempo definido. En este sentido, el vector $\mathbf{u}$ viene a ser la salida del actuador de la planta.

Para determinar la controlabilidad de nuestro sistema linealizado en torno al punto de equilibrio $(\mathbf{x}^e, \mathbf{u}^e)$, primero determinamos las matrices A y B reemplazando los valores dados por el enunciado:

$$A = \frac{\partial \mathbf{f}(\mathbf{x}^e, \mathbf{u}^e)}{\partial \mathbf{x}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{5}{24} \end{bmatrix}$$

$$B = \frac{\partial \mathbf{f}(\mathbf{x}^e, \mathbf{u}^e)}{\partial \mathbf{u}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{44000} & 0 \\ 0 & \frac{1}{48000} \end{bmatrix}$$

luego, definimos la matriz de controlabilidad:

$$\mathcal{C} = \begin{bmatrix} B & AB & A^2B & ... & A^{n-1}B \end{bmatrix} \quad \mathbf{x} \in \mathbb{R}^n \tag{9}$$

si el rango de esta matriz es igual a la cantidad de variables de estado:

$$rank(\mathcal{C}) = n$$

entonces posee esa cantidad de columnas LI y el sistema es controlable para las variables de estado:

$$\mathbf{x} = \begin{bmatrix} x \\ \theta \\ \dot{x} \\ \dot{\theta} \end{bmatrix} \quad \mathbf{x} \in \mathbb{R}^4$$

Calculamos la matriz de controlabilidad usando al función **ctrb()** de MATLAB o de la librería de control en python, de la siguiente forma: **ctrb(A, B)**

$$\mathcal{C} = 10^{-4} \cdot \begin{bmatrix} 0 & 0 & 0.2273 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.2083 & 0 & -0.0386 & 0 & 0.0071 \\ 0.2273 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.2083 & 0 & -0.0386 & 0 & 0.0071 & 0 & -0.0013 \end{bmatrix}$$

Luego, en python usando **np.linalg.matrix_rank()** obtenemos que el **rango de la matriz de controlabilidad es 4** igual que el número de variables de estado. Por lo tanto, podemos concluir que existe una señal de controles (variables manipuladas) que nos permiten dirijir desde un estado inicial del sistema $\mathbf{x}_0$ a cualquier estado final $\mathbf{x}_f$.

Para determinar la observabilidad del sistema podemos hacer uso de la condición de observabilidad de Kalman:

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix} \quad \mathbf{x} \in \mathbb{R}^n \tag{10}$$

y si el rango de esta matriz es igual a la cantidad de variables de estado:

$$rank(\mathcal{O}) = n$$

entonces el sistema es observable. Para ello utilizaremos la función **obsv()** de MATLAB o de la librería de control el python, de la siguiente forma: **obsr(A, C)**. Luego, en python usaremos **np.linalf.matrix_rank()** nuevamente, con tal de determinar el rango de la matriz $\mathcal{O}$.

Con tal de determinar la observabilidad, dispondremos de dos conjuntos de mediciones por parte del sensor:

$$\mathbf{y}_1 = \begin{bmatrix} \dot{x} \\ \dot{\theta} \end{bmatrix} \quad \mathbf{y}_2 = \begin{bmatrix} x \\ \theta \end{bmatrix}$$

(a) $\mathbf{y} = (\dot{x}, \dot{\theta})$

La matriz de controles C nos queda:

$$C = \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Entonces, la matriz de observabilidad para este caso:

$$\mathcal{O} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.20833 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0434 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.00904 \end{bmatrix}$$

y su **rango es 2** por lo que **no** es observable.

(b) $\mathbf{y} = (x, \theta)$

La matriz de controles C nos queda:

$$C = \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Entonces, la matriz de observabilidad para este caso:

$$\mathcal{O} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.20833 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0434 \end{bmatrix}$$

y su **rango es 4** por lo que **sí** es observable.

Adjunto a este informe está el achivo **ctrb&obsv.py** que contienen el código con los cálculos realizados.

3.2. LGR, Nyquist y Bode

Para los gráficos de LGR, Nyquist y Bode se utilizarán las funciones de transferencia definidas para el espacio de los estados linealizado en el punto de equilibrio y entregadas en el modelo corregido por el profesor Miguel Torres:

$$F_x(s) = \begin{bmatrix} 10^{-5}\frac{2.3121s^2+48.1696s-2.8353}{s^4+21.4258s^3+3.5694s^2-0.2835s} & 10^{-6}\frac{2.8902}{s^3+21.4258s^2+3.5694s-0.2835} \end{bmatrix}$$

$$F_\theta(s) = \begin{bmatrix} 10^{-6}\frac{2.8902}{s^3+21.4258s^2+3.5694s-0.2835} & 10^{-5}\frac{2.1195s+0.4817}{s^3+21.4258s^2+3.5694s-0.2835} \end{bmatrix}$$

las que incluyen un término de acoplamiento que modele las perturbaciones cruzadas de las variables x y θ:

$$F_{coupling}(s) = 10^{-6}\frac{2.8902}{s^3 + 21.4258s^2 + 3.5694s - 0.2835}$$

Con tal de obtener las gráficas solicitadas por el enunciado se utilizará MATLAB con sus funciones **rlocus(sys)**, **nyquist(sys)** y **bode(sys)** para graficar el lugar geométrico de las raíces y las gráficas de Nyquist y Bode. De esta manera, se obtuvieron los siguientes plots:

1. **Sistema control movimiento horizontal carro:**

$$F_x(s) = 10^{-5}\frac{2.3121s^2 + 48.1696s - 2.8353}{s^4 + 21.4258s^3 + 3.5694s^2 - 0.2835s}$$

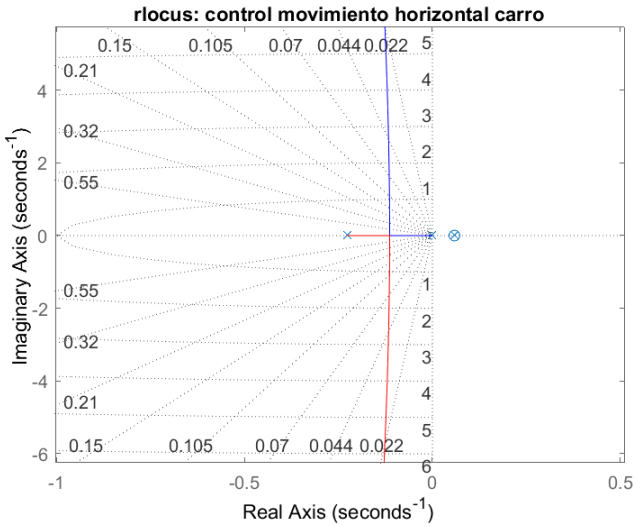
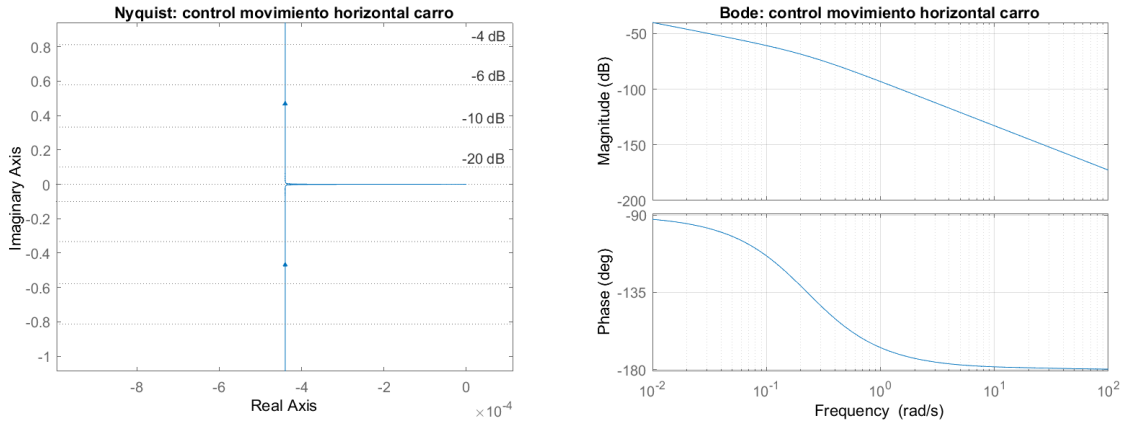


Figura 5: LGR $F_x(s)$

Nota: existe un polo y un cero cercanos en el semiplano positivo derecho del plano-s.



(a) Diagrama de Nyquist (b) Diagrama de Bode

Figura 6: Nyquist y Bode $F_x(s)$

2. **Sistema control de elevación cañón:**

$$F_\theta(s) = 10^{-5}\frac{2.1195s + 0.4817}{s^3 + 21.4258s^2 + 3.5694s - 0.2835}$$

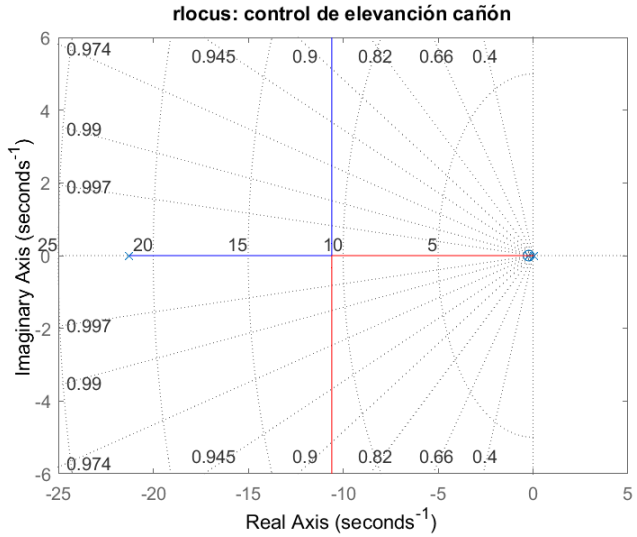
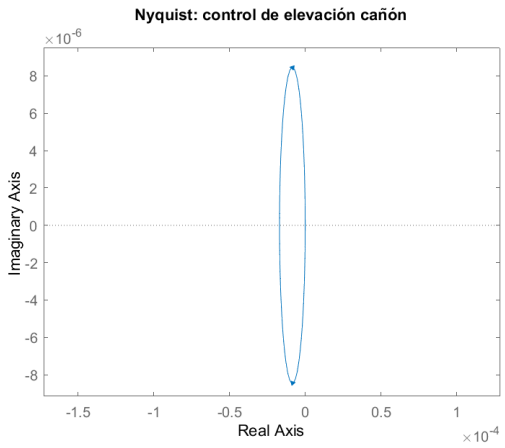
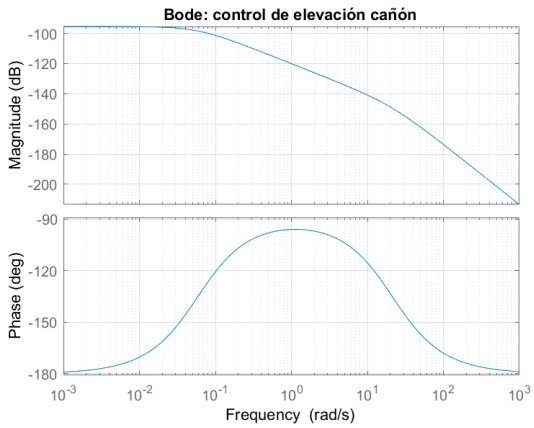


Figura 7: LGR $F_\theta(s)$



(a) Diagrama de Nyquist



(b) Diagrama de Bode

Figura 8: Nyquist y Bode $F_\theta(s)$

3. Sistema de control acoplamiento señales cruzadas

$$F_{coupling}(s) = 10^{-6}\frac{2.8902}{s^3 + 21.4258s^2 + 3.5694s - 0.2835}$$

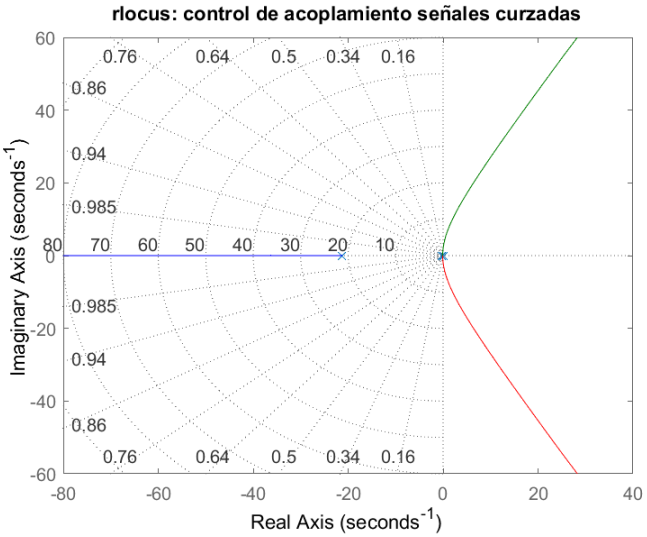
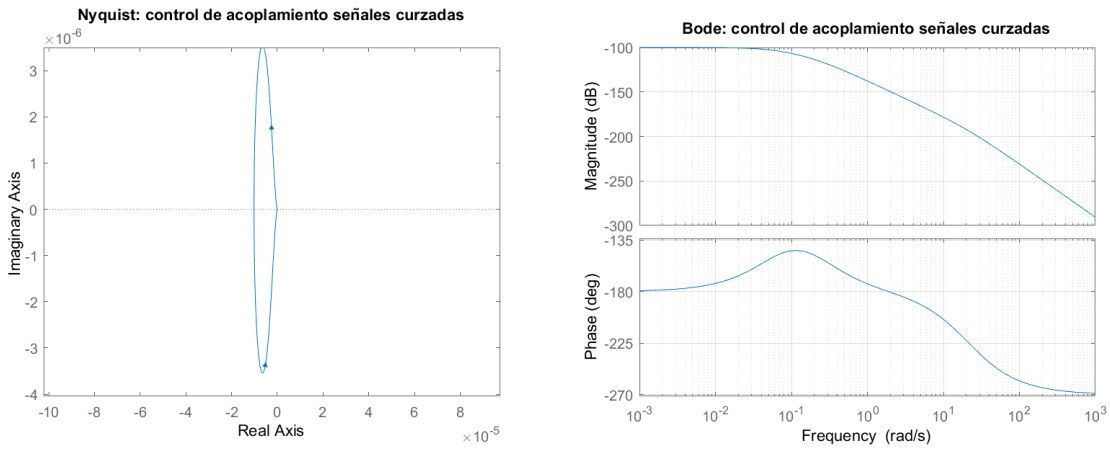


Figura 9: LGR $F_{coupling}(s)$



(a) Diagrama de Nyquist (b) Diagrama de Bode

Figura 10: Nyquist y Bode $F_{coupling}(s)$

3.3. Diseño del controlador

Para el diseño de un controlador discretro existen dos formas muy comunes para su implementación. Primero está la idea que dada la función de transferencia contínua $F(s)$ de la planta que representa el sistema, se puede diseñar un controlador contínuo para esta mediante el posicionamiento del polo en $s = 0$ y los dos ceros que entrega este este, y luego discretizar el controlador para que sea usado en la parte digital del lazo de control. La otra forma es obtener la representación de la planta en tiempo discreto $F(z)$ mediante la retención de orden cero (ZOH) y en base a esta diseñar el controlador en tiempo discreto. Para esta ocasión, el segundo camino será utilizado en la implementación de un **controlador PID discreto de la forma incremental**. Este controlador actuará sobre el error del sistema manifestando su respuesta a través del actuador de este, o sea, a través las variables manipuladas del sistema. Dada las variables manipuladas del sistema carro-cañón móvil:

$$\mathbf{u} = \begin{bmatrix} F_x \\ F_\theta \end{bmatrix}$$

podemos definir el controlador de la siguiente manera:

$$u[n] = u[n-1] + K_p(e[n] - e[n-1]) + K_i T_s e[n] + \frac{K_d}{T_s}(e[n] - 2e[n-1] + e[n-2]), \quad (11)$$

dónde $u[n]$ es la varible manipulada y $e[n]$ es el error dado por la referencia y la variable controlada, ambos en el instante n, T_s es el periodo de muestreo del sistema discreto y los parámetros K_p, K_i y K_d son las ganancias para la parte proporcional, integral y derivativa del controlador. Tomando transformada $\mathcal{Z}$ en ambos lados de la ecuación podemos despejar para el controlador $C(z)$:

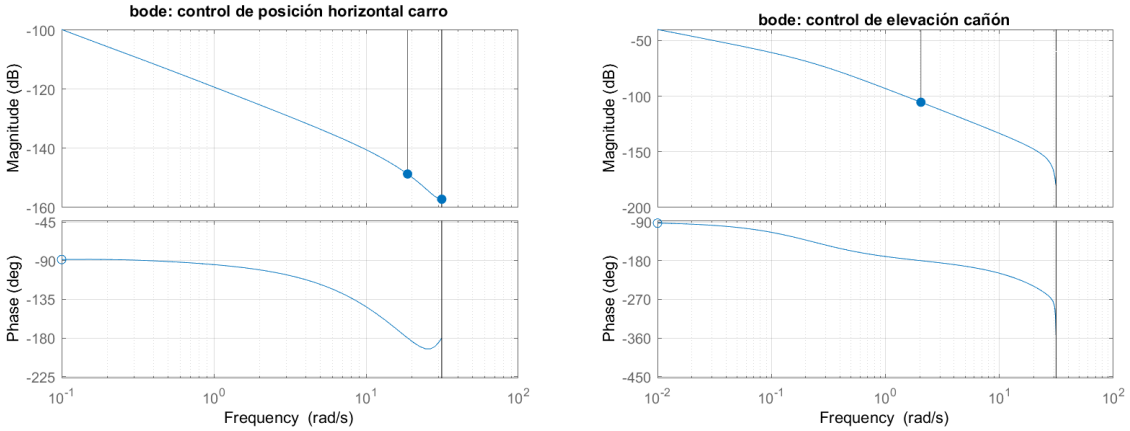
$$U(z) = z^{-1}U(z) + K_p E(z)(1 - z^{-1}) + K_i T_s E(z) + \frac{Kd}{Ts} E(z)(1 - 2z^{-1} + z^{-2})$$

$$C(z) = \frac{U(z)}{E(z)} = \frac{(T_s^2(K_p + K_i) + K_d)z^2 + (T_s K_p - 2K_d)z + K_d}{T_s z(z-1)} \tag{12}$$

Notamos que tal controlador nos entrega 2 polos, un en el origen y el otro justo sobre la circunsferencia unitaria del plano-z en $z = 0$. El diseño del controlador será encontrar tales ganancias que estabilicen el sistema y que hagan que el error en régimen permanente sea cero lo más rápido posible. Para ello utilizaremos un periodo de muestreo $T_s = 0.1$ $[s]$

3.3.1. Respuesta en frecuencia: Diagrama de Bode

En esta sección determinaremos los márgenes de ganancia y fase utilizando el diagrama de bode de las funciones de transferencia en lazo abierto $F_x(z)$ y F_θ. Para ello utilizaremos el comando **bode(sys)** de MATLAB y la herramienta que este presenta para identificar los márgenes mencionados. Bosquejando los diagramas de bode:



(a) $F_x(z)$ (b) $F_\theta(z)$

Figura 11: Diagrama de Bode plantas discretizadas

Notar: la línea negra vertical representa la frecuencia de Nyquist de ambos sistemas.

El margen de ganancia es aquel que nos mueve nuestra curva de magnitud a una ganancia de 0 $[dB]$ a la altura de la frecuencia que nos da una fase de 180°. Así, el margen de ganancia para $F_x(z)$ es:

$$GM_x = 157 \ [dB]$$

y para $F_\theta(z)$:

$$GM_\theta = 105 \ [dB]$$

Por otra lado, el margen de fase representa cuanto retardo debe haber en la fase para que esta sea $-180°$ a 0 $[dB]$, por lo que para ambos sistemas el margen de fase es:

$$PM_x = PM_\theta = 90°$$

Podemos utilizar los márgenes de ganancia para saber aproximadamente qué ganancia hará que las plantas $F_x(z)$ y $F_\theta(z)$ se vuelvan inestables por medio de la fórmula:

$$|F(j\omega)|_{dB} = 20\log(|F(j\omega)|) \tag{13}$$

3.3.2. Ganancia crítica: K_{cr}

Para esta sección utilizaremos las funciones de transferencia discretizadas $F_x(z)$ y $F_\theta(z)$ de las plantas para determinar la ganancias $K_{cr,x}$ y $K_{cr,\theta}$ que sitúen a los polos sobre el círculo unitario del plano-z. Utilizando la función **c2d(sys, Ts)** en MATLAB podemos obetener las funciones de transferencia discretizadas:

$$F_x(z) = 10^{-8}\frac{6.32z^2 - 2.969z - 3.155}{z^3 - 2.102z^2 + 1.224z - 0.1217}$$

$$F_\theta(z) = 10^{-7}\frac{1.076z^2 + 1.005z - 0.1129}{z^3 - 2.102z^2 + 1.224z - 0.1217}$$

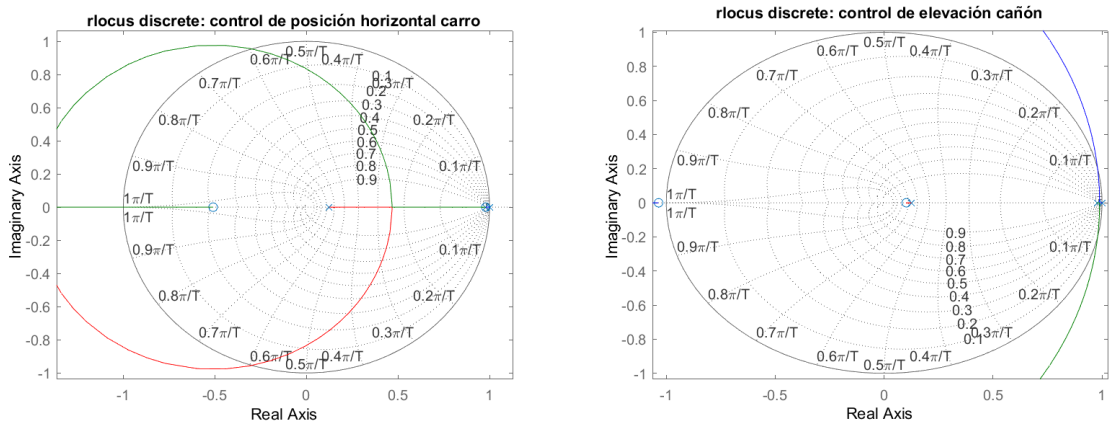
Es de suma importancia mencionar que esta discretización fue hecha a las siguientes funciones de transferencia:

$$F_x(s) = 10^{-6}\frac{5.76s^2 + 1.2s}{0.2534s^4 + 5.338s^3 + 1.2s^2}$$

$$F_\theta(s) = 10^{-4}\frac{0.0528s + 1.2}{0.2534s^3 + 5.338s^2 + 1.2s}$$

que fueron obtenidas al evaluar las plantas que acompañan a las señales de control directas y cruzadas en el punto de equilibrio de la sección **1.3**. Notar que estas plantas fueron las que nos entregaron los sistemas $X(s)$ y $\Theta(s)$ en la sección **1.5**. La razón por la cual se decidió utilizar estas últimas funciones de transferencia fue porque las descritas por el profesor Miguel Torres carecían de estabilidad al incorporar, una de ellas, un polo y un cero en el semiplano positivo del plano-s. Además, es importante notar que al utilizar $F_x(z)$ y $F_\theta(z)$ no estamos utilizando la acción de acoplamiento para las señales de control cruzadas, dada en las ecuaciones corregidas por el mismo profesor.

Podemos encontrar el LGR de las funciones $F_x(z)$ y $F_\theta(z)$ para determinar la ganancia crítica de ambos sistemas. Para ello, utilizando el mismo comando **rlocus(sys)** de MATLAB, obtenemos los siguientes gráficos:



(a) $F_x(z)$ (b) $F_\theta(z)$

Figura 12: LGR plantas discretizadas

Estos gráficos muestran que ambos sistemas poseen de polos justo sobre el círculo unitario ($z = 1$) lo que no brinda mayor información acerca de la estabilidad de ambos sistemas.

MATLAB provee información de la ganancia en lazo cerrado al ir haciendo "click" sobre la trayectoria que van tomando los polos. De esta manera, de los gráficos podemos extraer las siguientes ganancias críticas aproximadas para las plantas $F_x(z)$ y $F_\theta(z)$:

$$K_{cr,x} \approx 2.7 \cdot 10^7 \qquad K_{cr,\theta} \approx 1.8 \cdot 10^5$$

Además, como ya se mencionó, existen polos justo en la perisferia de la inestabilidad por lo que queremos que estos también migren a un cero cercano y dentro de la circunsferencia unitaria. Para ello encontramos que las ganancias mínimas aproximadas para lograr esto:

$$K_{min,x} \approx 5.0 \cdot 10^3 \qquad K_{min,\theta} \approx 60$$

De esta manera, podemos imponer ciertas restricciones sobre la ganacia total que tendrán los controladores para los errores:

$$e_x[n] = x_{ref} - x[n] \qquad e_\theta[n] = \theta_{ref} - \theta[n]$$

También nos podemos apoyar en los diagramas de

3.3.3. Ganancias proporcional, integral y derivativa: K_p, K_i y K_d

Dada una ganancia específica, definida de acuerdo a los márgenes de la sección **3.3.1**, para cada controlador discreto $C_x(z)$ y $C_\theta(z)$ asignaremos un valor porcentual de esta a cada parámetro K_p, K_i y K_d siguiendo la siguiente tabla:

TABLE I
EFFECTS OF INDEPENDENT P, I, AND D TUNING

Closed-Loop Response	Rise Time	Overshoot	Settling Time	Steady-State Error	Stability
Increasing K_P	Decrease	Increase	Small Increase	Decrease	Degrade
Increasing K_I	Small Decrease	Increase	Increase	Large Decrease	Degrade
Increasing K_D	Small Decrease	Decrease	Decrease	Minor Change	Improve

Figura 13: Efectos independientes de las ganacias para las acciones P, I y D

Así, modificando las ganancias K_p, K_i y K_d uno puede obtener los distintos comportamientos enseñados por la figura (12).

Existen diversos métodos para la afinación de los valores dados a las ganancias proporcional, integral y derivativa. Uno de estos es el método de Zigler-Nichols para la respuesta en lazo cerrado al realimentar la planta con la ganancia crítica de esta. Para este proyecto, tanto las ganancias para $C_x(z)$: $K_{p,x}$, $K_{i,x}$ y $K_{d,x}$, como para $C_\theta(z)$: $K_{p,\theta}$, $K_{i,\theta}$ y $K_{d,\theta}$, fueron ajustadas experimentalmente con variadas simulaciones hechas en el script de python **sim.py** y observando las repuestas del sistema a una entrada escalón referencial en el script **plot_data.py**. La ganancia total utilizada para los controladores $C_x(z)$ y $C_\theta(z)$ es:

$$K_x = 1.55 \cdot 10^5 \qquad K_\theta = 1.65 \cdot 10^5$$

Luego, para la ganancias específicas de cada controlador tendríamos:

$$K_{p,x} = 1.0 \cdot K_x \quad K_{i,x} = 0.275 \cdot K_x \quad K_{d,x} = 0.55 \cdot K_x$$

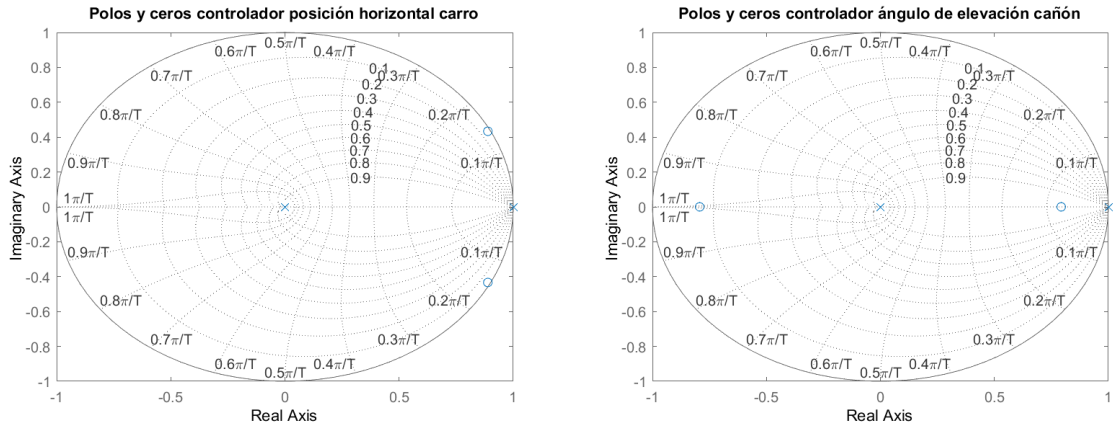
$$K_{p,\theta} = 1.0 \cdot K_\theta \quad K_{i,\theta} = 0.8 \cdot K_\theta \quad K_{d,\theta} = 0.3 \cdot K_\theta$$

Podemos notar que ambos controladores poseen acción integral con tal de eliminar el error en régimen permanente y que la acción derivativa es mayor que la integral para el caso del controlador $C_x(z)$. Estas ganancias para un periodo de muestreo de 0.1 $[s]$ nos entregan los siguientes controladores:

$$C_x(z) = \frac{87226.3z^2 - 155000z + 85250}{0.1z(1-z)}$$

$$C_\theta(z) = \frac{52470z^2 - 82500z + 495000}{0.1z(1-z)}$$

y sus respectivos diagramas de polos y ceros:



(a) $C_x(z)$ (b) $C_\theta(z)$

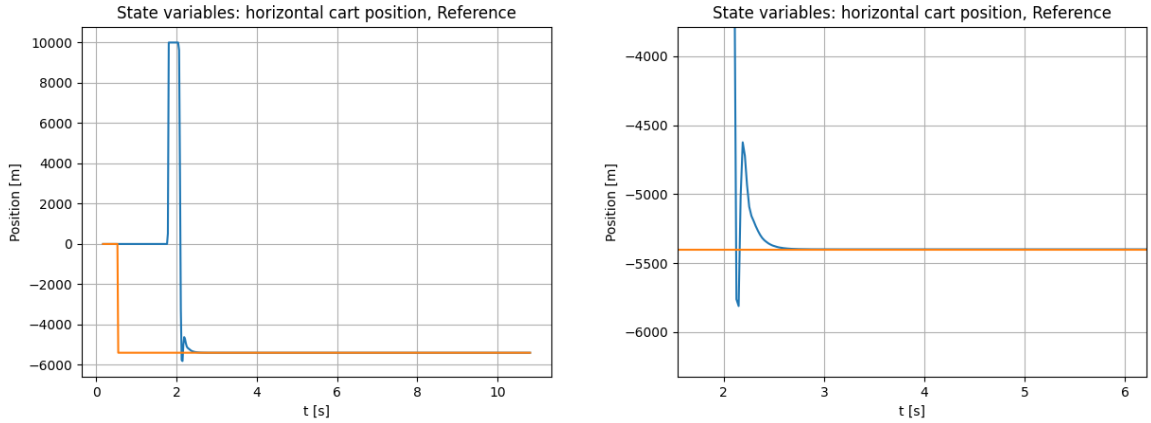
Figura 14: Diagrama de polos y ceros controladores discretos

3.3.4. Respuesta a entrada escalón: cañón contra incendios

Para esta sección, las entradas del sistema serán las referencias que obtenga el sistema carro-cañón para la posición horizontal y el ángulo de elevación: (x_{ref}, θ_{ref}).

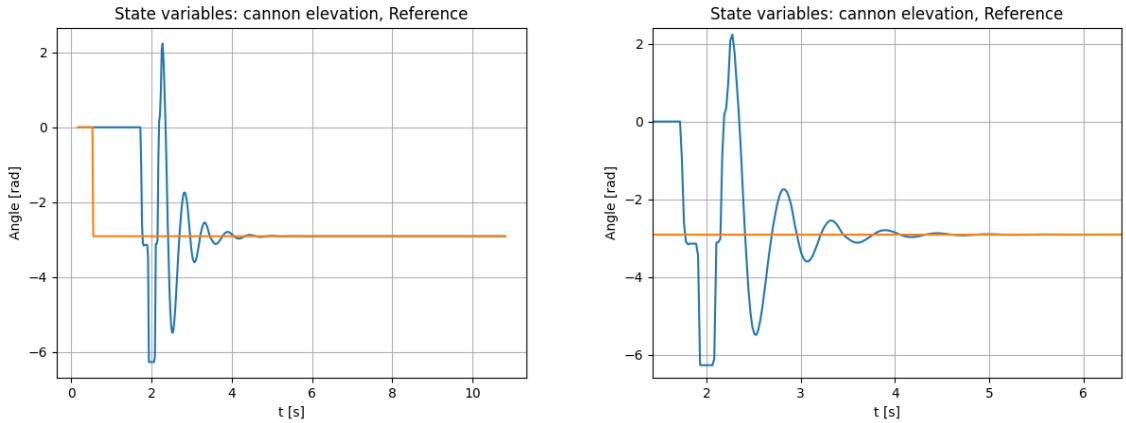
Antes de graficar, es importante mencionar qué es lo que se prioriza en la acción de ambos controladores. Para esta ocasión, en $C_x(z)$ se busca priorizar *overshoot* por sobre el *settling time* ya que queremos que el carro dispare desde una distancia definida al fuego y que no se desvíe mucho de esta. En cambio, en $C_\theta(z)$ se busca priorizar *settling time*, lo que se debe a que queremos que el cañón alcance la elevación necesaria rápidamene para poder apagar el fuego lo más rápido posible. Graficando para la referencia dada por el gráfico del fuego:

$$(x_{ref}, \theta_{ref}) = (-5400\ [m], -2.91\ [rad])$$



(a) Zoom out (b) Zoom in

Figura 15: Respuesta al escalón para x



(a) Zoom out (b) Zoom in

Figura 16: Respuesta al escalón para θ

Podemos medir las especificaciones en el dominio del tiempo *overshoot* M_p y *settling time* t_{ss} obteniendo lo siguiente:

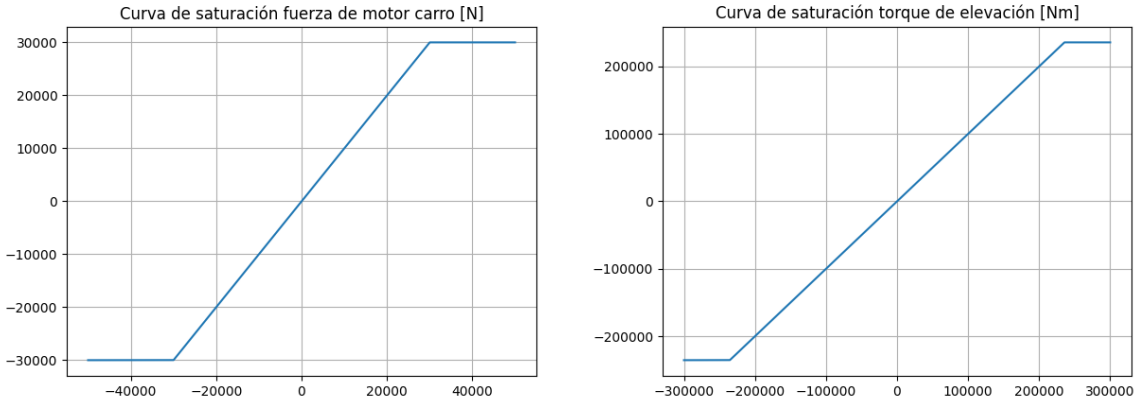
$$M_{p,x} = \frac{-5805 - (-5400)}{-5400} \cdot 100 = 7.5\,\% \qquad t_{ss,x} = \frac{2.7 - 2.3}{2.7} \cdot 100 = 14.81\,\%$$

$$M_{p,\theta} = \frac{-3.58 - (-2.91)}{-2.91} \cdot 100 = 23\,\% \qquad t_{ss,\theta} = \frac{5.4 - 3.32}{5.4} \cdot 100 = 38.52\,\%$$

Lo que demuestra la priorización del *overshoot* para el controlador de la posición horizontal y no logra demostrar que se prioriza el *settling time* para el controlador de la elevación del cañón.

3.3.5. Respuesta de los actuadores: motor de tracción del carro y motor de elevación del cañón

El la respuesta del actuador de un sistema real nunca será lineal. Esto se debe a que siempre existirá un punto en el cuál la salida de este se satura y no permite que esta varíe más allá de los márgenes dados por la saturación. Los gráficos dado los márgenes entregados por enunciado para el motor del carro y el motor del cañón son:

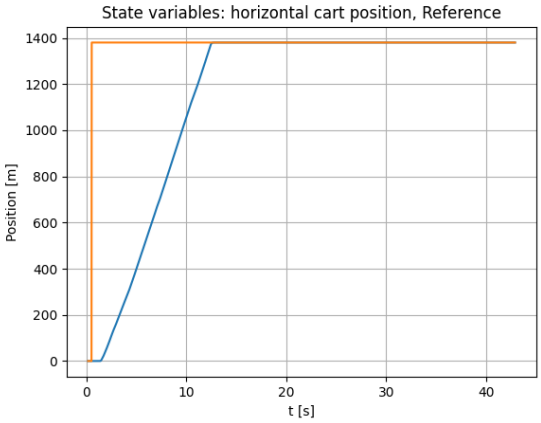


(a) F_x (b) F_c

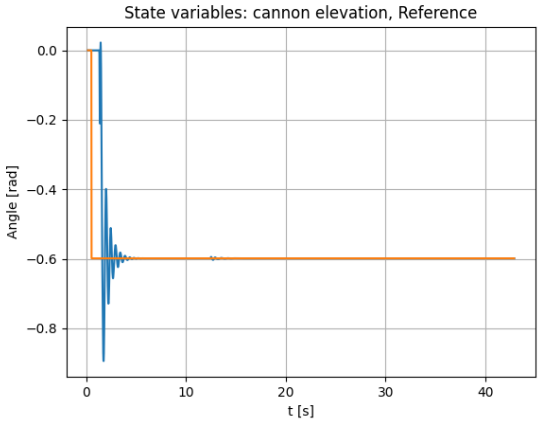
Figura 17: Fuerza y torque de saturación de ambos motores

El comportamiento entregado por estas curvas, sumado a la acción integral entregada por el controlador PID, puede generar el fenómeno de *integral windup*. Este último puede hacer que nuestro sistema se aleje mucho de la referencia y que le tome un largo tiempo en volver a un un punto donde comience el tiempo de respuesta a tal referencia. De esta manera, las curvas de la sección **3.3.4** presentarían un cambio. Graficando la respuesta de ambos sistemas dada la saturación de los actuadores para la posición de referencia:

$$(x_{ref}, \theta_{ref}) = (1380\ [m], -0.5988\ [rad])$$



(a) Respuesta al escalón para el control de posición x



(b) Respuesta al escalón para el control de elevación θ

Figura 18: Respuesta a entrada referencia escalón con saturación de motores

3.3.6. Cifras de desempeño: IQE

Para esta sección será utilizada la cifras o índice de desempeño para el error $e(t)$: **Integral of Quadratic Error: IQE** cuya fórmula es:

$$IQE = \int_0^T e^2(t)dt \tag{14}$$

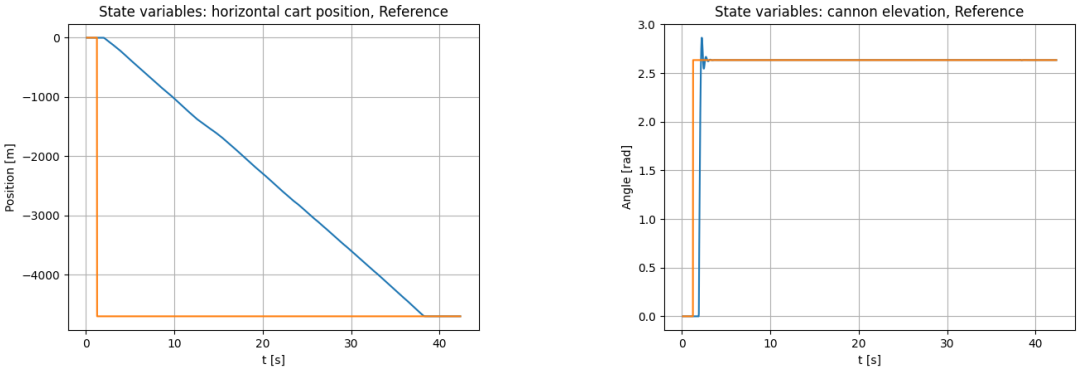
Dado que el error es discreto $e[n]$ la cifra se vuelve:

$$IQE = \int_0^T e^2(t)dt \approx \sum_{n=0}^{N}(e[n] \cdot T_s)^2 \tag{15}$$

Se calculará esta cifra en el script de python **plot_data** y se graficará la respuesta al escalón, utilizando la curva de saturación para los actuadores, para 3 referencias distintas:

1. $\mathbf{x}_{ref} = (-4700\ [m],\ 2.634[rad])$

$$IQE_x = \sum_{n=0}^{N}(e_x[n] \cdot T_s)^2 = 13.624 \cdot 10^8 \quad IQE_\theta = \sum_{n=0}^{N}(e_\theta[n] \cdot T_s)^2 = 2.394$$

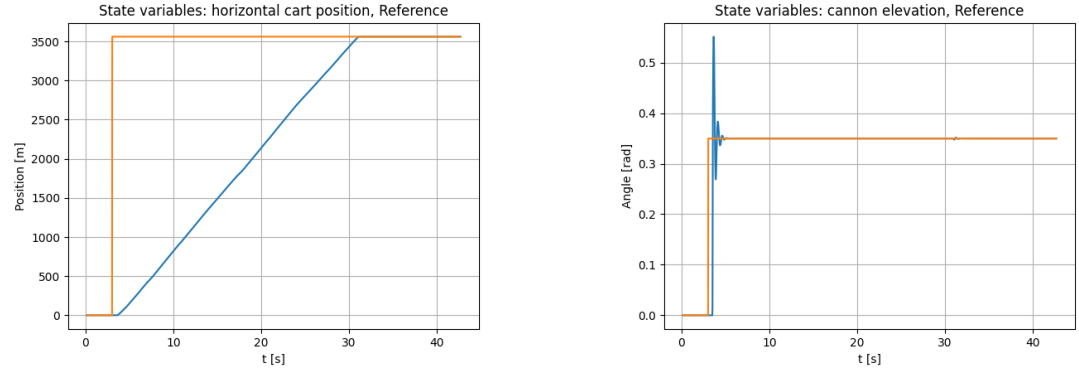


(a) $X(s)$ (b) $\Theta(s)$

Figura 19: Respuesta al escalón para $\mathbf{x}_{ref}$

2. $\mathbf{x}_{ref} = (3560\ [m], 0.365\ [rad])$

$$IQE_x = \sum_{n=0}^{N} (e_x[n] \cdot T_s)^2 = 58.45 \cdot 10^6 \quad IQE_\theta = \sum_{n=0}^{N} (e_\theta[n] \cdot T_s)^2 = 0.0322$$

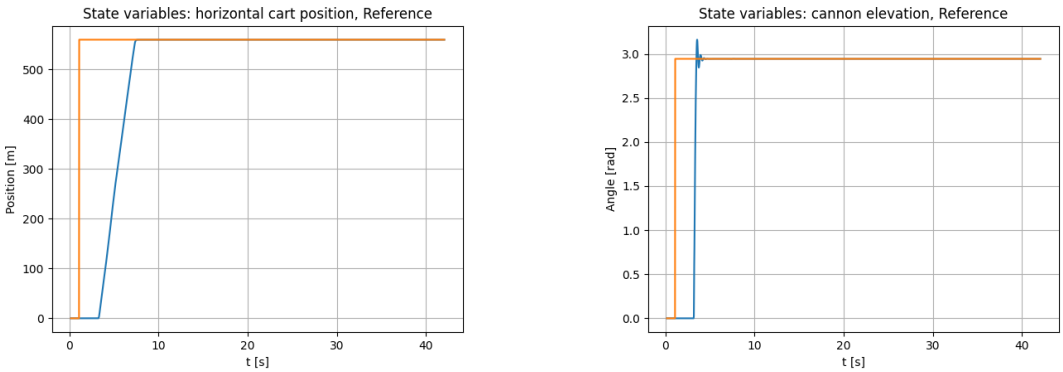


(a) $X(s)$ (b) $\Theta(s)$

Figura 20: Respuesta al escalón para $\mathbf{x}_{ref}$

3. $\mathbf{x}_{ref} = (560\ [m], 9.116\ [rad])$

$$IQE_x = \sum_{n=0}^{N} (e_x[n] \cdot T_s)^2 = 5.335 \cdot 10^5 \quad IQE_\theta = \sum_{n=0}^{N} (e_\theta[n] \cdot T_s)^2 = 9.116$$



(a) $X(s)$ (b) $\Theta(s)$

Figura 21: Respuesta al escalón para $\mathbf{x}_{ref}$

Dado que esta cifra sólo suma el error cuadrático en el tiempo, no es una cifra muy buena para analizar este sistema, principalmente al evaluarla en el control de posición horizontal $X(s)$ ya que su IQE_x es muy distinto para los 3 casos. Esta diferencia es muy notable al ver las respuestas al escalón en $X(s)$. Se debe observar que se está tomando un área muy grande dada por: $e_x[n] = x_{ref} - x[n]$ y, en cada instante, se está elevando al cuadrado y multiplicando por el periodo de muestreo. Es claro que si la referencia es muy lejana al punto de partida esta área será mayor y por ende un mayor valor para IQE_x.

3.4. Controlador a distintos tiempos de muestreo: Ts

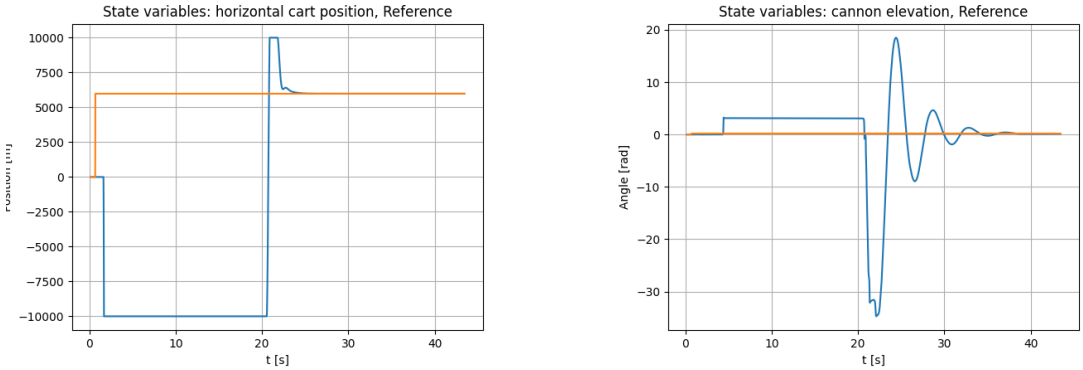
En esta sección se comparará el desempeño de los controladores $C_x(z)$ y $C_\theta(z)$ diseñados en la sección **3.3.** a partir del comportamiento que presenta el sistema linealizado carro-cañón. Para ello, se analizarán dos tiempos de muestreo distintos al expuesto en el enunciado del proyecto.

Como bien sabemos, la selección del tiempo de muestreo depende de la dinámica del proceso y de la frecuencia con la que varían las variables que se quieren controlar. Además, también se sabe que a mayor tiempo de muestreo se vuelven más apreciables los efectos del ruido y especialmente tienden a hacer que las componentes de alta frecuencia se capturen y amplifiquen en la parte derivativa del control. Por último, una mala elección para el periodo de muestreo puede comprometer la estabilidad del sistema discreto, haciendo de este un sistema inestable.

A continuación, se graficará el comportamiento de las variables discretas x y θ sin la saturación del actuador, al realizar una simulación para los siguientes tiempos de muestreo:

1. $T_s = 0.01$ $[s]$ (2000 muestras; 40 segundos de simulación)

$$\mathbf{x}_{ref} = (5980\ [m], 0.1974\ [rad])$$



(a) $X(s)$ (b) $\Theta(s)$

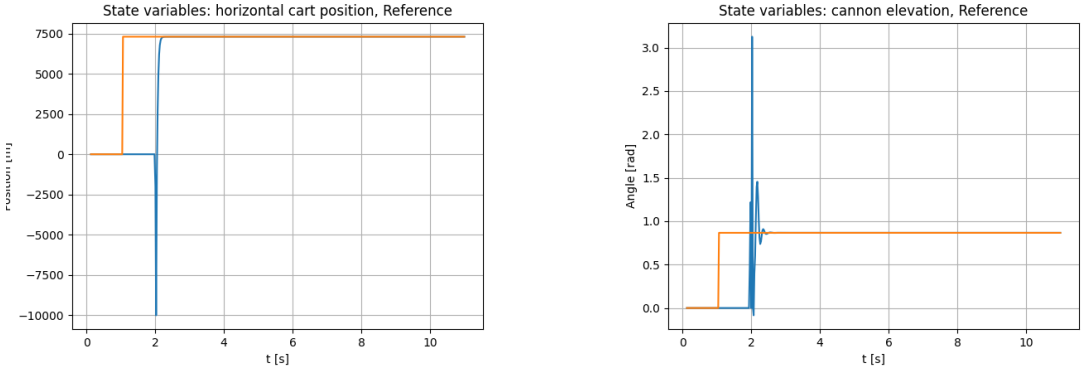
Figura 22: Respuesta al escalón para $\mathbf{x}_{ref}$

Para estas respuestas es normal que las cifras de desempeño nos den valores altos. Esto último se debe a que el tiempo de respuesta aumenta (el controlador se hace más lento) y por ende hay más error que integrar. Así el valor de la integral del error cuadrático para ambos sistemas:

$$IQE_x = 2.3245 \cdot 10^{10} \qquad IQE_\theta = 1.1795 \cdot 10^3$$

2. $T_s = 0.3\ [s]$ (500 muestras; 10 segundos de simulación)

$$\mathbf{x}_{ref} = (7300\ [m], 0.866\ [rad])$$



(a) $X(s)$ (b) $\Theta(s)$

Figura 23: Respuesta al escalón para $\mathbf{x}_{ref}$

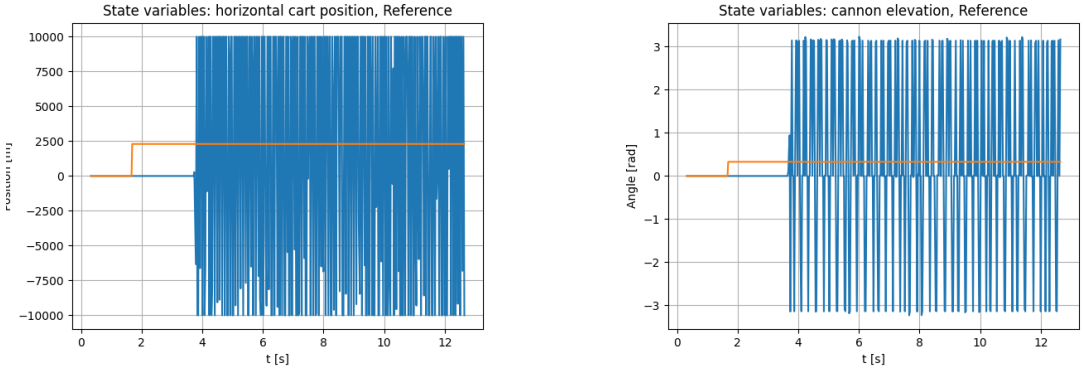
Para estas respuestas, podemos ver que el tiempo de respuesta disminuye, por lo que esperaríamos tener valores más bajos para el IQE del error de las variables:

$$IQE_x = 28.79 \cdot 10^6 \qquad IQE_\theta = 0.412$$

Además, podemos notar que después del gran *overshoot* "negativo" de la variable x esta se estabiliza y converge a la referencia rápidamente y sin oscilación. En cambio para la variable θ el *overshoot* se incrementa.

3. $T_s = 1.0\ [s]$ (500 muestras; 10 segundos de simulación)

$$\mathbf{x}_{ref} = (2300\ [m], 0.3288\ [rad])$$



(a) $X(s)$ (b) $\Theta(s)$

Figura 24: Respuesta al escalón para $\mathbf{x}_{ref}$

Podemos observar que ambos sistemas se vuelven completamente inestables para este tiempo de muestreo por lo que no es necesario calcular los valores del IQE.

Los cambios en las diferentes respuestas al escalón se deben principalmente al cambio en la ubicación de los polos en el plano-z, ya que un cambio en T_s afecta directamente el controlador discreto de la ecuación (12). Un ejemplo claro de esto es la simulación hecha para el último tiempo de muestreo, donde queda claro que para $T_s = 1\ [s]$ los polos se encuentran fuera del circulo imaginario en el plano-z.

3.5. Pertubaciones externas y ruidos para el sensor: control de elevación del cañón

Para la modelación del ruido en la medición de las variables por parte del sensor utilizaremos una variable aleatoria $\mathcal{N}$ que distribuye normal con media cero y varianza unitaria:

$$\mathcal{N} \sim Normal(0, 1)$$

Esta perturbación afectará al momento de retroalimentar la salida del sistema por lo que se verá reflejada en el error que presenta el sistema de control al cerrar el lazo de la siguiente manera:

$$e_\theta = \theta_{ref} - \theta + \mathcal{N}$$

Podemos implementar esto último con la función **random.normal(0,1)** de la librería Numpy en Python y luego graficar la respuesta que tendrá el ángulo de elevación al cañón. Para la siguente referencia de ángulo:

$$\theta_{ref} = 2.724\ [rad]$$

se obtuvo el siguiente gráfico al guardar 2000 muestras:

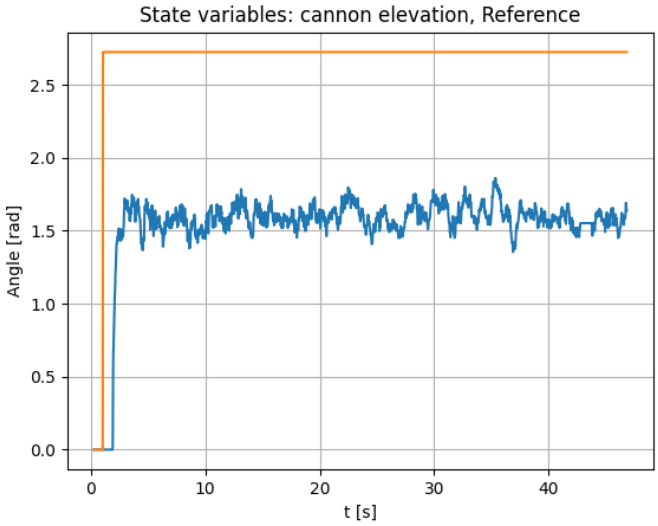


Figura 25: Ruido en el sensor para θ

Donde se puede observar que el controlador $C_\theta(z)$ no logra superar el ruido impuesto por $\mathcal{N}$ para alcanzar la referencia dada en la parte superior que representa una señal escalón.

3.6. Nivel máximo de pertubaciones a tolerar por el controlador

De representar esta como una combinación lineal de diferentes variables aleatorias $\mathcal{N}_i$, con diferentes ponderadores o amplitudes para estos ruidos, podemos agregar la siguiente sumatoria:

$$\sum_{i=0}^{N} A_i \mathcal{N}_i$$

con N el número de perturbaciones y A_i como la amplitud del ruido i, a las mediciones del sensor y con ello a los errores e_x y e_θ de los sistemas de control.

Dado que no se hizo una investigación profunda de como abordar esta parte del proyecto, esta **no** fue implementada.

Referencias

[1] M. Torres, *Proyecto Cañón contra Incendios*. Control Automático IEE2613, Pontificia Universidad Católica de Chile, 2023.

[2] M. Torres, *Ecuaciones Cañón contra Incendios*. Control Automático IEE2613, Pontificia Universidad Católica de Chile, 2023.

[3] M. Torres, *L08_Control_PID*, Lectura 8, ppt. 8, Control Automático IEE2613, Pontificia Universidad Católica de Chile, 2023.

[4] Kiam Heong Ang, Chong, G; Yun Li (2005). *PID control system analysis, design and technology*. IEEE Transactions on Control Systems Technology. 13 (4): 559–576.