UNIVERSIDAD TÉCNICA FEDERICO SANTA MARIA

DEPARTAMENTO DE ELECTRÓNICA

VALPARAISO - CHILE

“ IMPLEMENTACIÓN DE UN SISTEMA DE CONTROL PREDICTIVO USANDO EL TOOLBOX QUARC ”

MICHEL ANDRÉS AZÚA POBLETE

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRÓNICO, MENCIÓN CONTROL AUTOMÁTICO, SUBMENCIÓN COMPUTADORES

PROFESOR GUIA: DR. JUAN CARLOS AGÜERO

PROFESORES CORREFERENTE: DR. CESAR SILVA JIMÉNEZ

Septiembre 2022

Material de referencia, su uso no involucra responsabilidad del autor o de la Institución.

Agradecimientos

Agradezco a mi familia, en especial a mi madre y padre, cuyo apoyo ha sido permanente e indispensable para el desarrollo personal en todos los aspectos de mi vida. Mi proceso universitario es un resultado del fruto de su formación.

Agradezco a mis profesores y ayudantes, sus palabras fueron sabias, sus conocimientos rigurosos y precisos. Donde quiera que vaya, los llevaré conmigo en mí transitar profesional. Gracias por su paciencia, por compartir sus conocimientos de manera profesional e invaluable, por su dedicación perseverancia y tolerancia.

Agradezco a mis compañeros y compañeras de la universidad, en especial a las personas de "IRE", con los que vivimos muchas experiencias durante estos años de universidad. Gracias por su apoyo y constancia, al estar en las horas más difíciles, por compartir horas de estudio. Gracias por estar siempre allí.

Finalmente, agradezco el apoyo del Centro Basal "Advanced Center for Electrical and Electronic Engineering" (AC3E) - FB0008 y al proyecto ANID-FONDECYT 1211630.

DEPARTAMENTO DE ELECTRONICA
UNIVERSIDAD TECNICA FEDERICO SANTA MARIA

Implementación de un Sistema de Control Predictivo usando el toolbox Quarc

Michel Andrés Azúa Poblete

Memoria para optar al título de Ingeniero Civil Electrónico, mención Control Automático, submención Computadores.

Universidad Técnica Federico Santa María

Profesor Guía: Dr. Juan Carlos Agüero

Septiembre 2022

Resumen

Este trabajo aborda el problema del controlador por modelo predictivo en el motor de DC en tiempo real. Nos enfocaremos en realizar una acción integral y usar un algoritmo de optimización de ejecución llamado ADMM (Método de dirección alterna del multiplicador).

Para realizar esta técnica de control, se utilizó la dinámica del motor DC. Este modelo principal se obtiene a partir de datos empíricos del motor.

Se utilizaron dos métodos para implementar esta técnica de control: Toolbox Quarc y MicroLabBox - dSpace. Estos permiten realizar un prototipo rápido y simple de una técnica de control usando Matlab/Simulink.

Palabras Clave: MPC, ADMM, Servo, Observadores, Matlab, Simulink, Quarc, MicroLabBox

DEPARTAMENTO DE ELECTRONICA
UNIVERSIDAD TECNICA FEDERICO SANTA MARIA

Implementation of a Predictive Control System using a Quarc Toolbox

Michel Andrés Azúa Poblete

Memory for the fulfillment of the B.S. in Electronic Engineering, major in Automatic Control, minor in Computer (6 year program).

Universidad Técnica Federico Santa María

Advisor: Juan Carlos Agüero, PhD.

September 2022

Abstract

This work address of problem of model predictive control in DC motor in real time. We focus in utilize a integral action and use a perform optimization algorithm call ADMM (Alternating Direction Method of Multiplier).

To realize this technique of control, we used the dynamic of the DC motor. This principal model obtain from empirical data of the motor.

Be used two methods to implement this technique of control : Toolbox Quarc and MicroLabBox - dSpace. These allow to realize a rapid and simple prototype a technique of control using Matlab/Simulink.

Keywords: MPC, ADMM, Servo, Observers, Matlab, Simulink, Quarc, MicroLabBox

Índice de contenidos

1. Introducción **1**

1.1. Estado del arte 1

1.2. Objetivos específicos 3

1.3. Trabajos relacionados 3

2. Control por Modelo Predictivo (MPC) **4**

2.1. Metodología del Control Predictivo 5

2.2. Programación Cuadrática (QP) 7

2.3. Seguimiento de referencia 13

2.4. Restricción de Desigualdad 14

2.5. Planteamiento final del problema QP 15

3. Método de Optimización **17**

3.1. Método de Multiplicadores de Lagrange con Direcciones Alternadas (ADMM) . 17

3.2. ADMM formulado para MPC 19

3.3. Algoritmo de ADMM para MPC 23

4. Planta de interés **25**

4.1. Sistemas Electromecánicos 25

4.2. Motor de Corriente Continua (DC) 26

4.3. El Servomotor 29

4.4. Servomotor: *Rotatory Servo Base Unit* 30

4.5. Modelo matemático de Servo SRV02 31

4.5.1. Ecuaciones eléctricas del motor 31

4.5.2. Ecuaciones mecánicas del motor 33

4.5.3. Combinación de ecuaciones eléctrica y mecánica 35

5. Modelado del sistema **38**

5.1. Respuesta Escalón 38

5.2. Respuesta Escalón para distintas excitaciones 41

5.3. Modelo utilizado para técnica de Control 45

6. Muestreo del Sistema **47**

6.1. Muestreo de un sistema utilizando un Retenedor de Orden Cero (ZOH) 48

6.2. Representación del sistema dado un tiempo de muestreo especifico 49

7. Observadores **54**

7.1. Definición de Observadores 54

7.2. Observador para sistema básico 54

7.3. Observador extendido 56

7.4. Seguimiento de referencia considerando una perturbación estimada 58

8. Implementación de Método de Control **60**

8.1. Matlab/Simulink 60

8.1.1. Bloques útiles para implementar MPC 62

8.2. Quanser 64

8.2.1. Hardware Quanser 64

8.2.2. Software Quanser 66

8.2.3. Acotaciones al usar Quarc en Simulink 69

8.2.3.1. Encoder 70

8.2.4. Implementación de MPC en QUARC 75

8.2.5. Resultados obtenidos en Quarc 78

8.3. MicroLabBox/ dSpace 84

8.4. Hardware dSpace 84

8.5. Software para dSpace 87

8.5.1. Acotaciones al usar MicroLabBox con Simulink 90

8.5.2. Implementación de MPC con dSpace 92

8.5.3. Resultados obtenidos con dSpace 94

9. Conclusiones **98**

Referencias **99**

10. Anexos **101**

10.1. Códigos *MATLAB* 101

Índice de tablas

4.1. Características de SRV02 31
8.2. Resultados obtenidos sobre la implementación de MPC en Quarc. 1=Éxito, 0=Fallo 83
8.3. Resultados obtenidos sobre la implementación de MPC en Quarc (optimizado). 1=Éxito, 0=Fallo 83
8.4. Resultados obtenidos sobre la implementación de MPC en dSpace. 1=Éxito, 0=Fallo 97

Índice de figuras

2.1. Diagrama de Bloque para MPC 6
4.2. Sistema electromecánico con referencia de entrada $r(t)$ y salida $y(t)$ 25
4.3. Diagrama de bloques funcionales de un sistema electromecánico 26
4.4. Esquema de un motor DC, Fuente: [1] 27
4.5. Diagrama esquemático de un servo [2] 29
4.6. Srv02, Quanser 30
4.7. Circuito esquemático y sistema mecánico de Motor DC 32
4.8. Especificaciones de Servomotor SRV02 37
5.9. Sistema Servo Motor 38
5.10. Gráfico Respuesta Escalón con $\alpha = 1$ 41
5.11. Gráfico de entrada y salida estacionaria para $n = 23$ muestras 42
5.12. Gráfico de entrada y salida estacionaria para $n = 23$ muestras, con estimación. 44
5.13. Gráfico con estimaciones de τ obtenidos para $n = 23$ muestras 44
5.14. Gráfico con estimaciones de K obtenidos para $n = 23$ muestras 45
6.15. Diagrama de bloque de sistema de tiempo continuo conectado a convertidor A-D y D-A 47
6.16. Diagrama de bloque de sistema de un sistema muestreado con una planta en tiempo continuo y un ZOH 48
6.17. Diagrama de bloque de sistema muestreado considerando una planta discreta . 49

6.18. Representación de sistemas 50
7.19. Diagrama de Observador 55
8.20. Esquema de la plataforma Matlab 61
8.21. Biblioteca de Simulink 62
8.22. Bloque Simulink: Matlab Function, y su entorno de programación. 63
8.23. Bloque Simulink: Discrete State-Space. 63
8.24. Bloque Simulink: Demux y mux 63
8.25. Bloque Simulink: Bloques comunes 64
8.26. Hardware Quanser: Q8-USB Data Acquisition Device 65
8.27. Hardware Quanser: VoltPAQ-X2 Amplifier 65
8.28. Hardware Quanser: Servo SRV02 66
8.29. Software Quanser: HIL Initialize 67
8.30. Software Quanser: HIL Write Analog 67
8.31. Software Quanser: HIL Read Encoder 68
8.32. Software Quanser: Inverse Modulus 68
8.33. Software Quanser: To host file 69
8.34. Ejemplo de envío y recepción de señales en Servo utilizando Quarc 69
8.35. Salida de un encoder incremental cuando existe una rotación [3] 71
8.36. Gráficos obtenidos para primer ejemplo de uso de Quarc 72
8.37. Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 16. 73
8.38. Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 16. 73
8.39. Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 20. 73
8.40. Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 20. 74
8.41. Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 12. 74
8.42. Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 12. 74
8.43. Implementación de MPC en simulink con Quarc 76
8.44. Implementación de MPC en simulink con Quarc. Subsistema de seguimiento de referencia 76
8.45. Implementación de MPC en simulink con Quarc. Subsistema de MPC 77
8.46. Implementación de MPC en simulink con Quarc. Subsistema de conexiones con Servo motor real 77

8.47. Gráfico de salida obtenida con h=0.1[s] y N=5 en Quarc. 78
8.48. Gráfico de señal de entrada con h=0.1[s] y N=5 en Quarc. 79
8.49. Gráfico de señal de entrada con h=0.1[s] y N=5 en Quarc. 79
8.50. Gráfico de tiempos en resolver MPC con h=0.1[s] y N=5 en Quarc. 80
8.51. Gráfico de salida obtenida con h=0.01[s] y N=5 en Quarc. 81
8.52. Gráfico de señal de entrada con h=0.01[s] y N=5 en Quarc. 81
8.53. Gráfico de señal de entrada con h=0.01[s] y N=5 en Quarc. 82
8.54. Gráfico de tiempos en resolver MPC con h=0.01[s] y N=5 en Quarc. 82
8.55. Hardware dSpace: MicroLabBox 85
8.56. Hardware dSpace: Cable BNC-RCA 86
8.57. Hardware dSpace: Cable BNC-RCA 86
8.58. Software dSpace: Real-Time Interface (RTI) 87
8.59. Software dSpace: Control Desk 88
8.60. Bloque dSpace: RTI Data 88
8.61. Bloque dSpace: Sensor Supply 89
8.62. Bloque dSpace: DAC CLASS1 89
8.63. Bloque dSpace: EMC Encoder 90
8.64. Ejemplo de envío y recepción de señales en Servo utilizando dSpace 91
8.65. Gráfico de mediciones de ejemplo dSpace 91
8.66. Implementación de MPC en Simulink con dSpace 93
8.67. Captura de Datos en Control Desk 93
8.68. Gráfico de salida obtenida con h=0.1[s] y N=5 en dSpace. 94
8.69. Gráfico de señal de entrada con h=0.1[s] y N=5 en dSpace. 95
8.70. Gráfico de estimaciones de estados con h=0.1[s] y N=5 en dSpace. 95
8.71. Gráfico de salida obtenida con h=0.01[s] y N=5 en dSpace. 96
8.72. Gráfico de señal de entrada con h=0.01[s] y N=5 en dSpace. 96
8.73. Gráfico de estimación de estados con h=0.01[s] y N=5 en dSpace. 97

Índice de códigos

1. Introducción

Los avances de la robótica y de la automatización, por mencionar algunos que utilizan sistemas electromecánicos, están evolucionando constantemente. Con la intención de actualizar las metodologías de control y de realizar proyectos de diferente índole, los ingenieros buscan programar sistemas de control avanzados en plantas con versatilidad de funciones, en particular este trabajo aborda como planta los Servomotores. El uso de este tipo de motores es ampliamente difundido en la actualidad, pudiendo utilizarse en diversos procesos industriales.

Por otra parte, los procesos en los distintos ámbitos exigen no solo cumplir con resolver y desarrollar las tareas que se registran en el momento presente, sino bien, anticipar los acontecimientos para satisfacer ciertas especificaciones sujetas a una demanda normalmente variable. En este escenario, los sistema de control deben aplicar acciones directas sobre las variables manipulables, de tal forma que puedan satisfacerse los múltiples y cambiantes criterios de funcionamiento del sistema. Una poderosa herramienta para afrontar este reto consiste en aplicar el sistema de Control Predictivo por Modelo (MPC). A su vez, dentro de las herramientas y metodologías disponibles, es relevante considerar las que permitan optimizar los sistemas, en particular, se plantea conseguir eficiencia en el modelo de control a partir de la implementación del algoritmo del Método de los Multiplicadores de Lagrange con Direcciones Alternadas (ADMM).

A lo largo de este capitulo se presentará el contexto bajo el cual se desarrolló este trabajo de titulo. Preguntas como: ¿De donde surge la motivación a trabajar con MPC? o ¿Que efectos posee trabaja en tiempo real? serán analizados para comprender el escenario al cual nos enfrentamos y en que direcciones debemos concentrar nuestros avances.

1.1. Estado del arte

Un Modelo de Control Predictivo (MPC) es un método avanzado de control de procesos, que se basa en un algoritmo de predicción, lo cual lo diferencia de otro tipo de controladores como PID. Su uso en la industria se remonta mayormente a plantas químicas y refinerías de petróleo, pero en el último tiempo se utiliza en sistemas para potencia [4]. MPC no se utiliza normalmente en plantas simples, ya que para estas basta un PID, no obstante, su potencial

se ve expuesto ante plantas con comportamientos dinámicos complejos, como el caso plantas con retrasos de gran magnitud. MPC es un área en crecimiento, que ha recibido un interés continuo tanto para el ámbito industrial como para la investigación académica [5], todo esto debido a 4 aspectos fundamentales:

- Formulación del diseño bajo parámetros multivariable
- Capacidad de manejar restricciones duras y suaves
- Habilidad de optimización de procesos en línea
- Simplicidad del marco de diseño en el manejo de problemas complejos

Otra razón para hacer control avanzado sobre un control PID, es por la eficiencia que se puede obtener, sobre todo en términos de tiempos [6], por eso se ha desarrollado MPC Embedded [7], la implementación de las arquitecturas propuestas en FPGA muestra que se puede lograr un desempeño de control satisfactorio a una frecuencia de muestreo superior a 1 MHz incluso en dispositivos de gama baja, lo que abre nuevas posibilidades para la aplicación de MPC en sistemas integrados.

Los sistemas en tiempo real se pueden definir como cualquier sistema en el cual el tiempo en el que se produce la salida es significativo, causado generalmente porque la entrada corresponde a alguna perturbación en el mundo físico. Una opción para trabajar con estos sistemas es utilizar las capacidades de Matlab/Simulink en conjunto con a el Toolbox Quarc. El software Quarc de Quanser agrega herramientas y capacidades poderosas a Matlab/Simulink que facilitan el desarrollo y la implementación de sofisticadas aplicaciones de control y mecatrónica en tiempo real, ya que genera código en tiempo real directamente desde los controladores diseñados por Simulink y lo ejecuta en tiempo real, todo sin procesamiento de señal digital o sin escribir una sola línea de código [8]. Además, posee las herramientas para poder trabajar bajo red [9].

Otra opción para realizar implementaciones es MicroLabBox [10], este es un sistema de desarrollo todo en uno para el laboratorio que combina el tamaño compacto y los bajos costos

del sistema con un alto rendimiento y versatilidad. Permite configurar aplicaciones de control, prueba o medición de forma rápida y sencilla. Este posee una alta potencia de cómputo combinada con latencias de E/S muy bajas proporciona un gran rendimiento en tiempo real.

1.2. Objetivos específicos

Se esperan los siguientes objetivos principales:

- Implementar un control por modelo predictivo utilizando Quarc y MicroLabBox
- Validar la utilización del Algoritmo de Multiplicadores Aumentados de Lagrange (ADMM) en una implementación de MPC en tiempo real.
- Implementar una acción integral en un Control por Modelo Predictivo para corregir el error en estado estacionario.
- Documentar el uso de Quarc y MicroLabBox para implementaciones de técnicas de control.

1.3. Trabajos relacionados

MPC puede ser trabajado de diversas maneras, y pese a que se ha mencionado el Algoritmo del Método de los Multiplicadores de Lagrange con Direcciones Alternadas (ADMM), una alternativa que guarda una cercana relación consiste en el diseño de un modelo que posee una función de costos cuadráticos, pero con restricciones lineales. Este se puede realizar con un algoritmo de programación cuadrática (QP), el cual minimiza la función de costo por cada iteración. Más específicamente dentro de los algoritmos (QP), se puede mencionar Punto Interior.

Los problemas de optimización surgen en una gran variedad de aplicaciones, entre las que destacan, la ingeniería, las finanzas, la investigación de operaciones, el aprendizaje automático (machine learning), entre otros campos. Particularmente las aplicaciones específicas que despiertan interés en el campo de la ingeniería electrónica en el área de control y automatización dicen relación con el desarrollo de Modelos Predictivos (MPC) y estimación de Horizonte Móvil (MHE) que utilizan soluciones del tipo QP.

2. Control por Modelo Predictivo (MPC)

El control Predictivo basado en Modelo o MPC es una técnica avanzada de control que, utilizando un modelo matemático del sistema a controlar y el valor de los estados del sistema en un instante de tiempo dado, es capaz de determinar el valor óptimo de las entrada para que el sistema siga una referencia a lo largo de una secuencia de muestras futuras. Esto lo realiza sujeto a restricciones que pueden tomar la forma de valores máximos y mínimos permitidos para las entradas, salidas y estados [11].

El Control por Modelo Predictivo de basa en cuatro componentes:

- Modelo matemático del sistema a controlar: Se utiliza la representación en estados de un sistema linealizado en tiempo discreto para estimar la dinámica del sistema.
- Horizonte de control de N muestras: Es el número de acciones de controles futuras que se calcularan en cada instante para realizar un control en el sistema.
- Conjunto de restricciones de las entradas, salidas y estados del sistema: Limitaciones de rangos de actuaciones en las variables mencionadas.
- Función de costos: Problema de optimización que penaliza los errores de las entradas y los estados con sus valores deseados.

El desempeño de esta técnica de control dependerá principalmente del modelo matemático que uno posea del sistema a controlar, debido a que las señales para realizar control provienen de las predicciones que se realizan a partir de este modelo. Si el modelo es poco exacto, las señales de control no serán óptimas y en casos extremos podrían provocar inestabilidad en el sistema, pero si el modelo se comporta semejante a el sistema a controlar, las predicciones serán más acertadas, de modo que MPC podría generar señales de control óptimas si es que se implementa correctamente.

Otro factor importante respecto al desempeño del controlador es el método que se utiliza para optimizar la función de costos, esto debido a que algunos métodos pueden tomar mucho tiempo en converger o utilizar demasiados recursos computacionales para en terminar de optimizar la función de costos.

2.1. Metodología del Control Predictivo

En cada instante de muestreo, MPC utiliza el modelo matemático del sistema para predecir el comportamiento futuro del sistema a lo largo de N instantes futuros (estos N instantes lo llamaremos Horizonte de predicción). A partir de estos N instantes predichos, el controlador determina una secuencia de actuaciones que minimizan la función de costos y a la vez, respetan el conjunto de restricciones establecidas. Finalmente, se aplica el primer elemento de esta secuencia y se espera a el siguiente intervalo de control para nuevamente predecir el comportamiento futuro del sistema, repitiendo el proceso de optimización en este nuevo instante de tiempo y así sucesivamente hasta llegar al final del horizonte. Cabe destacar que no siempre se posee información sobre los estados de la planta, por lo que se deberán utilizar técnicas matemáticas para estimar estos valores.

La metodología de sólo utilizar la primera entrada de la secuencia entradas optimizadas y desechar el resto de las actuaciones se le llama 'Receding Horizon'. A partir de esta técnica, se dice que el sistema de control opera sobre un 'horizonte móvil' debido a que en cada instante de muestreo se calcula la actuación óptima para ese instante de tiempo. Esto es una ventaja al momento de realizar implementaciones en plantas, ya que permite tomar en cuenta las posibles variaciones que el sistema puede experimentar en el tiempo y corregir estos errores.

El esquema de MPC se encuentra representado en la Figura 2.1:

Teniendo en cuenta el esquema de MPC, se reescribirán los pasos de MPC en formato de pseudo-algoritmo. El proceso del controlador sigue los siguientes pasos, considerando un intervalo de tiempo desde k hasta k+N-1:

1. En el instante k , el bloque optimizador resuelve el problema de control óptimo en el intervalo [k,k+N-1], considerando una función de coste, errores obtenidos por el modelo matemático y las restricciones a las que se encuentra sometido el sistema.

2. Se obtiene el vector óptimo $\vec{u_k}$, del cual sólo se aplicará el primer elemento de este vector.

3. Se obtiene el nuevo estado del sistema a través del modelo matemático, es decir, x_{k+1}

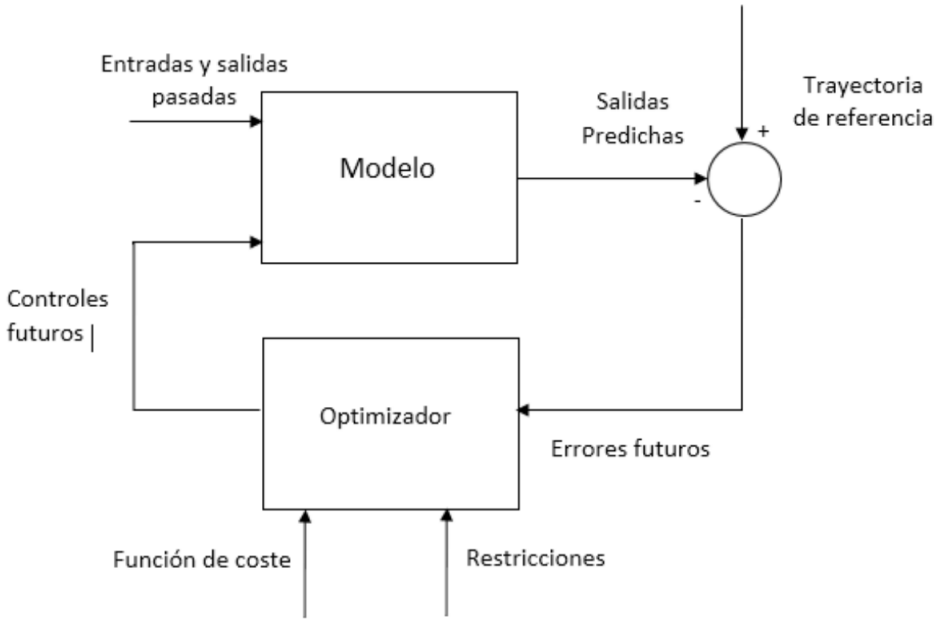


Figura 2.1: Diagrama de Bloque para MPC

4. Se repite el procedimiento, pero sobre el intervalo [k+1,k+N], considerando estado actual del sistema x_{k+1}.

El problema a resolver en el bloque optimizador se puede escribir como:

$$J_N^{\text{opt}}(x) = \text{mín}\, J_N(x_k, u_k) = \text{mín}\, P(x_N) + \sum_{k=0}^{N-1} L(x_k, u_k) \tag{2.1}$$

Sujeto a las siguientes restricciones:

$$x_{k+1} = f(x_k, u_k), \text{ para } k = 0, ..., N-1 \tag{2.2}$$

$$x_0 = x \tag{2.3}$$

$$u_k \in U, \text{ para } k = 0, ..., N-1 \tag{2.4}$$

$$x_k \in X, \text{ para } k = 0, ..., N \tag{2.5}$$

$$x_N \in X_f \subset X \tag{2.6}$$

La ecuación (2.1) corresponde a la función objetivo que se desea minimizar. Esta consta de dos

partes: la función P que pondera el estado final (N), y L, que corresponde a la ponderación por etapa. Por lo general estas funciones son representadas en forma cuadrática, esto se describirá más adelante en la sección 'Programación Cuadratica (QP)'.

Las ecuaciones (2.2)-(2.6) representan a x_k y u_k, que corresponden a la secuencia de estados y de control. Estas deben respetar las restricciones representadas por los conjuntos U, X y X_f. Si alguna de estas restricciones no se cumple, no será posible encontrar una solución óptima al problema definido para MPC.

2.2. Programación Cuadrática (QP)

Existen diversos problemas de optimización, uno de ellos, que será utilizado en esta memoria, es la Programación Cuadrática (Quadratic Programming).

Matemáticamente, el problema QP a resolver en cada intervalo se puede plantear de distintas formas según la elección de las variables de decisión. Cuando se elige como única variable de decisión la entrada u_k, entonces se obtiene la forma denominada 'Formulación Densa', y si se eligen como variable de decisión tanto la entrada u_k como el estado x_k, se denomina como 'Formulación Sparse'. La selección de la forma de representar el problema QP normalmente influye en el costo computacional, requerimientos de memoria y tiempos de convergencia asociados al problema QP [12].

La programación cuadrática utilizando notación matricial es una función de la forma:

$$\min J(x) = \frac{1}{2}x^T H x + h^T x \tag{2.7}$$

sujeto a las restricciones

$$Fx = f \tag{2.8}$$

$$Gx \leq g \tag{2.9}$$

donde H tiene que ser semi definida positiva para que el problema sea convexo y así sea posible encontrar un mínimo global.

A continuación, se detallará como se puede plantear el problema MPC como un problema QP utilizando la formulación densa (sólo en función de la entrada). Se debe tener en cuenta que la función objetivo expresada en (2.1) de MPC es una función cuadrática, por ende, basta con reescribirla de la forma en que se utiliza en programación cuadrática.

Se considera el siguiente sistema lineal discretizado e invariante en el tiempo que describe la dinámica de la planta a controlar.

$$x_{k+1} = Ax_k + Bu_k \tag{2.10}$$

$$y_k = Cx_k \tag{2.11}$$

Cabe destacar que la matriz D que relaciona directamente la salida con la entrada del sistema se desprecia debido a que típicamente es 0 en los sistemas reales.

Tomando la ecuación que relaciona la entrada con los estados estados:

$$x_{k+1} = Ax_k + Bu_k \quad ; \quad A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times m} \tag{2.12}$$

Esta ecuación se puede desarrollar de manera recursiva, desde $k = 0, ..., N$ por ejemplo. Para el instante $k = 0$ y $k = 1$ se tiene que:

$$x_1 = Ax_0 + Bu_0 \tag{2.13}$$

$$x_2 = Ax_1 + Bu_1 \tag{2.14}$$

Reemplazando (2.14) en (2.13):

$$x_2 = A(Ax_0 + Bu_0) + Bu_1 \tag{2.15}$$

$$x_2 = A^2x_0 + ABu_0 + Bu_1 \tag{2.16}$$

Para un $k = 2$, utilizando la relación obtenida anteriormente se obtiene que:

$$x_3 = A^3x_0 + A^2Bu_0 + ABu_1 + Bu_2 \tag{2.17}$$

Si uno sigue realizando estos pasos, obteniendo los estados para los siguientes instantes k_i, se puede notar una secuencia que permitirá obtener las predicciones de estos estados. Entonces, siendo x_0 la condición inicial y considerando:

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ x_N \end{bmatrix}, \vec{u} = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-2} \\ u_{N-1} \end{bmatrix} \tag{2.18}$$

Se obtiene la siguiente expresión que permite predecir los estados del sistema en su forma matricial:

$$\vec{x} = \begin{bmatrix} A \\ A^2 \\ \vdots \\ \vdots \\ A^N \end{bmatrix} x_0 + \begin{bmatrix} B & 0 & 0 & \dots & 0 \\ AB & B & 0 & \dots & 0 \\ \dots & \dots & \ddots & \vdots & 0 \\ \dots & \dots & \vdots & \ddots & 0 \\ A^{N-1}B & A^{N-2}B & \dots & AB & B \end{bmatrix} \vec{u} \tag{2.19}$$

$$\tag{2.20}$$

De forma simplificada, el vector $\vec{x}$ se puede expresar de la siguiente forma, en donde $\mathscr{A} \in \mathbb{R}^{(n \cdot N) \times n}$ y $\mathscr{O} \in \mathbb{R}^{(n \cdot N) \times (m \cdot N)}$ corresponden a matrices de predicción descritas en la ecuación (2.14):

$$\vec{x} = \mathscr{A} x_0 + \mathscr{O} \vec{u} \tag{2.21}$$

Por otra parte, la función objetivo queda descrita de manera matricial como:

$$J(x_0, \vec{u}) = x_0^T \Omega x_0 + \vec{x}^T \mathrm{diag}(\Omega, \Omega, ..., \Omega, \Omega_N) \vec{x} + \vec{u}^T \mathrm{diag}(\Gamma, \Gamma, ..., \Gamma) \vec{u} \tag{2.22}$$

En donde $\Omega \in \mathbb{R}^{n \times n}$ es una matriz de pesos que refleja la importancia relativa de cada estado en la determinación de la salida del sistema, que es equivalente a $\Omega = C^T C$. Además, Ω_N

corresponde a la matriz de peso para el estado final.

La matriz $\Gamma \in \mathbb{R}^{m\times m}$ es una matriz de pesos que refleja la importancia de la entrada del sistema en la función de costos al momento de optimizar. Para el caso de interés de múltiples entradas, se puede considerar por simplicidad una matriz diagonal $\Gamma = Iq$, donde $I \in \mathbb{R}^{m\times m}$ es la matriz identidad y q es un escalar que pondera la entrada.

La ecuación (2.22) puede ser escrita como una función cuadrática de $\vec{u}$, con $Q = \text{diag}(\Omega, \Omega, ..., \Omega, \Omega_N)$ con dimensión $\mathbb{R}^{(n\cdot N)\times(n\cdot N)}$ y $R = \text{diag}(\Gamma, \Gamma, ..., \Gamma)$ con dimensión $\mathbb{R}^{(m\cdot N)\times(m\cdot N)}$. Reescribiendo la función cuadrática se tiene que:

$$J(x_0, \vec{u}) = x_0^T \Omega x_0 + \vec{x}^T Q \vec{x} + \vec{u}^T R \vec{u} \tag{2.23}$$

Utilizando la predicción obtenida en la ecuación (2.21), se obtiene:

$$J(x_0, \vec{u}) = x_0^T \Omega x_0 + (\mathscr{A} x_0 + \mathscr{O}\vec{u})^T Q (\mathscr{A} x_0 + \mathscr{O}\vec{u}) + \vec{u}^T R \vec{u} \tag{2.24}$$

$$= x_0^T \Omega x_0 + x_0^T \mathscr{A}^T Q \mathscr{A} x_0 + x_0^T \mathscr{A}^T Q \mathscr{O} \vec{u} + \vec{u}^T \mathscr{O}^T Q \mathscr{A} x_0 + \vec{u}^T \mathscr{O}^T Q \mathscr{O}^T \vec{u} + \vec{u}^T R \vec{u} \tag{2.25}$$

los términos $x_0^T \mathscr{A}^T Q \mathscr{O} \vec{u}$ y $\vec{u}^T \mathscr{O}^T Q \mathscr{A} x_0$ son iguales ya que uno es el traspuesto del otro, por lo que se puede simplificar la expresión como:

$$J(x_0, \vec{u}) = x_0(\Omega + \mathscr{A}^T Q \mathscr{A}) x_0 + \vec{u}(R + \mathscr{O}^T Q \mathscr{O})\vec{u} + 2x_0^T \mathscr{A}^T Q \mathscr{O} \vec{u} \tag{2.26}$$

Se define $H \in \mathbb{R}^{(m\cdot N)\times(m\cdot N)}$ y $h^T \in \mathbb{R}^{1\times(m\cdot N)}$ como:

$$\frac{1}{2} H = R + \mathscr{O}^T Q \mathscr{O} \tag{2.27}$$

$$h^T = 2x_0^T \mathscr{A}^T Q \mathscr{O} \tag{2.28}$$

Reemplazando estas expresiones, finalmente se obtiene la función de costo de la forma QP como:

$$J(x_0, \vec{u}) = \frac{1}{2} \vec{u}^T H \vec{u} + h^T \vec{u} \tag{2.29}$$

Nótese que el termino $x_0(\Omega + \mathscr{A}^T Q \mathscr{A}) x_0$ no aparece en la función de costos (2.23) debido a

que su valor es constante para cada instante de muestreo, por lo que no va a afectar en la búsqueda de un $\vec{u}$ que minimice la función de costos.

Respecto a la restricciones, estas se deben reescribir en términos de $\vec{x}$ y $\vec{u}$ como se describe a continuación:

- Limitaciones en la señal de estado:

$$\vec{x}^{\min} \leq \vec{x} \leq \vec{x}^{\max} \tag{2.30}$$

$$\vec{x}^{\min} \leq (\mathscr{A} x_0 + \mathscr{O} \vec{u}) \leq \vec{x}^{\max} \tag{2.31}$$

  Despejando y separando las desigualdades se obtiene que:

$$\mathscr{O} \vec{u} \leq (\vec{x}^{\max} - \mathscr{A} x_0) \tag{2.32}$$

$$-\mathscr{O} \vec{u} \leq -(\vec{x}^{\min} - \mathscr{A} x_0) \tag{2.33}$$

  donde:

$$\vec{x}^{\min} = \begin{bmatrix} x^{\min} \\ \vdots \\ x^{\min} \end{bmatrix} \qquad \vec{x}^{\max} = \begin{bmatrix} x^{\max} \\ \vdots \\ x^{\max} \end{bmatrix} \tag{2.34}$$

  tal que $\vec{x}^{\min}$ y $\vec{x}^{\max} \in \mathbb{R}^{n \cdot N}$.

- Región final:

$$\vec{x}_N^{\min} \leq \vec{x}_N \leq \vec{x}_N^{\max} \tag{2.35}$$

$$\vec{x}_N^{\min} \leq (\mathscr{A}_N x_0 + \mathscr{O}_N \vec{u}) \leq \vec{x}_N^{\max} \tag{2.36}$$

Despejando y separando las desigualdades se obtiene que:

$$\mathscr{O}_N \vec{u} \leq (\vec{x}_N^{\max} - \mathscr{A}_N x_0) \tag{2.37}$$

$$-\mathscr{O}_N \vec{u} \leq -(\vec{x}_N^{\min} - \mathscr{A}_N x_0) \tag{2.38}$$

$$\tag{2.39}$$

donde

$$\mathscr{O}_N = \begin{bmatrix} A^{N-1}B & A^{N-2}B \dots & AB & B \end{bmatrix}, \quad \mathscr{A}_N = A^N \tag{2.40}$$

tal que $\mathscr{O}_N \in \mathbb{R}^{n \times (m \cdot N)}$ y $\mathscr{A}_N \in \mathbb{R}^{nxn}$.

- Limitaciones en la señal de entrada:

$$\vec{u}^{\min} \leq \vec{u} \leq \vec{u}^{\max} \tag{2.41}$$

donde

$$\vec{u}^{\min} = \begin{bmatrix} u^{\min} \\ \vdots \\ u^{\min} \end{bmatrix} \qquad \vec{u}^{\max} = \begin{bmatrix} u^{\max} \\ \vdots \\ u^{\max} \end{bmatrix} \tag{2.42}$$

tal que $\vec{u}^{\min}$ y $\vec{u}^{\max} \in \mathbb{R}^{m \cdot N}$.

Luego de reescribir las restricciones y escribirlas de forma matricial, estas se pueden agrupar y escribir como:

$$\underbrace{\begin{bmatrix} \mathscr{O}_N \\ -\mathscr{O}_N \\ \mathscr{O} \\ -\mathscr{O} \end{bmatrix}}_{M} \vec{u} \leq \underbrace{\begin{bmatrix} \vec{x}_N^{\max} - \mathscr{A}_N x_0 \\ -\vec{x}_N^{\min} + \mathscr{A}_N x_0 \\ \vec{x}^{\max} - \mathscr{A} x_0 \\ -\vec{x}^{\min} + \mathscr{A} x_0 \end{bmatrix}}_{c} \qquad \underbrace{\begin{bmatrix} u^{\min} \\ \vdots \\ u^{\min} \end{bmatrix}}_{a} \leq \vec{u} \leq \underbrace{\begin{bmatrix} u^{\max} \\ \vdots \\ u^{\max} \end{bmatrix}}_{b} \tag{2.43}$$

donde $M \in \mathbb{R}^{(2 \cdot n + 2 \cdot n \cdot N) \times (m \cdot N)}$, $c \in \mathbb{R}^{(2 \cdot n + 2 \cdot n \cdot N)}$, $a \in \mathbb{R}^{(m \cdot N)}$ y $b \in \mathbb{R}^{(m \cdot N)}$.

Nótese que las restricciones se transforman en restricciones lineales en $\vec{u}$. Finalmente, el problema de optimización QP se define como:

$$\vec{u}^* = \min_{\vec{u}} J(x_0, \vec{u}) \tag{2.44}$$

sujeto a

$$M\vec{u} \leq c \tag{2.45}$$

$$a \leq \vec{u} \leq b \tag{2.46}$$

2.3. Seguimiento de referencia

Para realizar un seguimiento de referencia para un $r \neq 0$, en la salida se requiere que $y_\infty \to r$, donde y_∞ representa el valor de la salida en estado estacionario. Para lograr esto, se analizaran las señales del sistema en estado en estado estacionario, a partir de esto, se podrá mover el punto de operación del sistema para que el controlador (MPC) funcione como un regulador que sólo se encargue de llevar el sistema a este nuevo punto de operación con una referencia 0. Para determinar el valor de los estados y entradas que permiten llevar la salida al valor de referencia, se reescribe la expresión en estado estacionario para el sistema como:

$$x_\infty = Ax_\infty + Bu_\infty \tag{2.47}$$

$$y_\infty = Cx_\infty \tag{2.48}$$

Teniendo en cuenta que $y_\infty \approx r$ en estado estacionario, se despeja x_∞ y u_∞ obteniendo:

$$\underbrace{\begin{bmatrix} I - A & -B \\ C & 0 \end{bmatrix}}_{L} \begin{bmatrix} x_\infty \\ u_\infty \end{bmatrix} = \begin{bmatrix} 0 \\ r \end{bmatrix} \tag{2.49}$$

$$\begin{bmatrix} x_\infty \\ u_\infty \end{bmatrix} = L^{-1} \begin{bmatrix} 0 \\ r \end{bmatrix} \tag{2.50}$$

donde $L \in \mathbb{R}^{(n+p)\times(n+m)}$.

Se realiza el siguiente cambio de variable que permite desplazarse del punto de operación:

$$\tilde{x} = x - x_\infty \tag{2.51}$$

$$\tilde{u} = u - u_\infty \tag{2.52}$$

donde $\tilde{x}$ y $\tilde{u}$ representan la desviación de los estados y entradas de los valores que permiten llevar la salida a su valor de referencia en estado estacionario. Considerando que las variables de desviación satisfacen el mismo modelo dinámico que las variables originales, se obtienen nuevas restricciones de forma matricial que permiten realizar un seguimiento de referencia:

$$\underbrace{\begin{bmatrix} \mathscr{O}_N \\ -\mathscr{O}_N \\ \mathscr{O} \\ -\mathscr{O} \end{bmatrix}}_{\tilde{M}} \vec{\tilde{u}} \leq \underbrace{\begin{bmatrix} \vec{x}_N^{\max} - x_\infty - \mathscr{A}_N x_0 \\ -\vec{x}_N^{\min} + x_\infty + \mathscr{A}_N x_0 \\ \vec{x}^{\max} - x_\infty - \mathscr{A} x_0 \\ -\vec{x}^{\min} + x_\infty + \mathscr{A} x_0 \end{bmatrix}}_{\tilde{c}} \qquad \underbrace{\begin{bmatrix} u^{\min} - u_\infty \\ \vdots \\ u^{\min} - u_\infty \end{bmatrix}}_{\tilde{a}} \leq \vec{\tilde{u}} \leq \underbrace{\begin{bmatrix} u^{\max} - u_\infty \\ \vdots \\ u^{\max} - u_\infty \end{bmatrix}}_{\tilde{b}} \tag{2.53}$$

donde $\tilde{M} \in \mathbb{R}^{(2\cdot n+2\cdot n\cdot N)\times(m\cdot N)}$, $\tilde{c} \in \mathbb{R}^{(2\cdot n+2\cdot n\cdot N)}$, $\tilde{a} \in \mathbb{R}^{(m\cdot N)}$ y $\tilde{b} \in \mathbb{R}^{(m\cdot N)}$.

De este modo, se debe encontrar la entrada óptima $\vec{\tilde{u}}^*$ bajo a las restricciones (2.53). Nótese que esta entrada óptima obtenida corresponde a la desviación óptima de la entrada que permite seguir la referencia deseada. La entrada óptima a aplicar al sistema corresponde a:

$$\vec{u}^* = \vec{\tilde{u}}^* + \vec{u}_\infty \tag{2.54}$$

Al utilizar este seguimiento de referencia no se asegura obtener un error cero en estado estacionario. Para arreglar este problema, en el Capítulo 7, se hablará sobre los observadores y estimación de perturbaciones, estos permitirán realizar una acción integral en el controlador. Esto permitirá minimizar el error a cero.

2.4. Restricción de Desigualdad

La restricción en caja de (2.53) se puede reescribir en una sola desigualdad para que así, todas las restricciones se puedan escribir en una sola ecuación matricial. Esto se plantea de la

siguiente forma:

$$\tilde{a} \leq \vec{\tilde{u}} \leq \tilde{b} \tag{2.55}$$

$$\tilde{a} \leq \vec{\tilde{u}} \qquad \vec{\tilde{u}} \leq \tilde{b} \tag{2.56}$$

$$-\vec{\tilde{u}} \leq -\tilde{a} \qquad \vec{\tilde{u}} \leq \tilde{b} \tag{2.57}$$

Agrupando estas restricciones de desigualdad de la ecuación (2.57), se juntan de la siguiente manera:

$$\begin{bmatrix} I \\ -I \end{bmatrix} \vec{\tilde{u}} \leq \begin{bmatrix} \tilde{b} \\ -\tilde{a} \end{bmatrix} \tag{2.58}$$

Utilizando la desigualdad de la ecuación (2.53) y juntándola con la restricción en caja obtenida en la ecuación (2.58), se definen las siguientes matrices:

$$\hat{M} = \begin{bmatrix} \tilde{M} \\ I \\ -I \end{bmatrix} \qquad \hat{c} = \begin{bmatrix} \tilde{c} \\ \tilde{b} \\ -\tilde{a} \end{bmatrix} \tag{2.59}$$

De esta manera, $\vec{\tilde{u}}$ queda restringido de la siguiente manera:

$$\hat{M}\vec{\tilde{u}} \leq \hat{c} \tag{2.60}$$

donde $\hat{M} \in \mathbb{R}^{(2\cdot n+2\cdot n\cdot N+2\cdot m\cdot N)\times(m\cdot N)}$ y $\hat{c} \in \mathbb{R}^{(2\cdot n+2\cdot n\cdot N+2\cdot m\cdot N)}$

2.5. Planteamiento final del problema QP

Finalmente, el problema QP en forma densa, con seguimiento de referencia r distinta a 0 y con restricciones en forma de desigualdad para encontrar una entrada $\vec{\tilde{u}}^*$ que minimice la función de costo, queda como:

$$J_N(\tilde{x_0}, \vec{\tilde{u}}) = \frac{1}{2}\vec{\tilde{u}}^T H \vec{\tilde{u}} + \tilde{h}^T \vec{\tilde{u}} \tag{2.61}$$

donde:

$$\frac{1}{2}H = R + \mathscr{O}^T Q \mathscr{O} \tag{2.62}$$

$$\tilde{h}^T = 2\tilde{x_0}^T \mathscr{A}^T Q \mathscr{O} \tag{2.63}$$

sujeto a:

$$\hat{M}\vec{\tilde{u}} \leq \hat{c} \tag{2.64}$$

Una vez encontrado $\vec{\tilde{u}}^*$, es necesario revertir el cambio de variables realizado en (2.52) para seguir la referencia, y así obtener la entrada que se debe aplicar a la planta. Esto se realiza con la siguiente relación:

$$\vec{u}^* = \vec{\tilde{u}}^* + \vec{u}_\infty \tag{2.65}$$

De esta forma, se obtiene el resultado final de MPC, este corresponde a el vector $\vec{u}^*$ que contiene las entradas para N instantes de tiempo hacia adelante necesarias para llevar a la salida de la planta a la referencia deseada. Se recuerda que sólo se aplica el primer elemento de este vector y en el siguiente instante de muestreo se repite el proceso, esto se debe a la metodología llamada 'Receding Horizon' de MPC.

Habiendo planteado el problema de MPC, como un problema QP utilizando la formulación densa, se debe utilizar un método de optimización para este problema cuadrático que encuentre la señal de entrada, que minimice el error. En el capítulo siguiente se describirá el método de Multiplicadores de Lagrange con Direcciones Alternadas (ADMM) como método de optimización a utilizar en esta memoria.

3. Método de Optimización

La teoría de optimización clásica o programación matemática está constituida por un conjunto de resultados, métodos analíticos y métodos numéricos enfocados a encontrar e identificar al mejor candidato entre diversas alternativas, sin tener que enumerar y evaluar de manera explicada todas ellas. Un problema de optimización es, en general, un problema de decisión [13].

Los problemas de optimización para MPC corresponden a problemas cuadráticos, por lo que se necesitan métodos de optimización que puedan resolver este tipo de problemas. En anteriores memorias, se utilizaron los métodos de Hildret [14] y método de punto interior [15] para resolver el problema de MPC. Para esta memoria, se propone utilizar el método de Multiplicadores de Lagrange con direcciones alternadas (ADMM) para implementaciones de MPC.

3.1. Método de Multiplicadores de Lagrange con Direcciones Alternadas (ADMM)

El método de Multiplicadores de Lagrange con Direcciones Alternadas (ADMM) comenzó a desarrollarse desde la década de 1960, hasta obtener un nombre propio en los albores de 1970. Está estrechamente relacionado con varios otros métodos, como la *Descomposición Dual*, los *Métodos Proximales*, la división de *Douglas-Rachford*, el método de inversas parciales de *Spingarn*, las proyecciones alternas de *Dykstra*, los algoritmos iterativos de *Bregman*, entre otros [16].

En particular, ADMM proviene de la combinación de dos métodos. Uno de ellos es el principio de Descomposición Dual (*Dual Ascent*) el cual permite dividir el problema de optimización, y el segundo corresponde al Método de multiplicadores Aumentados de Lagrange, el cual se basa en el método de los lagrangianos, pero agrega un término extra denominado 'aumentado' obteniendo de este método la estructura y propiedades favorables para conseguir la convergencia deseada.

El método de optimización ADMM, a diferencia de otros métodos donde sólo se enfocan en optimizar una única variable en su función objetivo, a través de una división posee como

variables de decisión a x y $z \in \mathbb{R}^n$, lo que conlleva a que la función objetivo sea separable. Gracias a esto, se tienen $f(x)$ y $f(z)$ como funciones convexas, cerradas y propias, expresadas de la forma:

$$\text{mín}\, f(x) + g(z) \tag{3.66}$$

La función expresada en la ecuación (3.66) está sujeta a una restricción lineal para las variables de decisión, con A y $B \in \mathbb{R}^{l \times n}$, mientras que $c \in \mathbb{R}^l$, así, se tiene:

$$Ax + Bz = c \tag{3.67}$$

A partir del problema planteado en la ecuación (3.66) se propone, a continuación, la naturaleza del Método de Multiplicadores Aumentados de Lagrange teniendo como lagrangiano a:

$$L_p(x, y, z) = \underbrace{f(x) + g(z)}_{\text{Función de Costo}} + \underbrace{y^T(Ax + Bz - c)}_{\text{Multiplicador de Lagrange}} + \underbrace{\frac{\rho}{2}||Ax + Bz - c||_2^2}_{\text{Término Proximal Aumentado}} \tag{3.68}$$

en donde el lagrangiano L_p descrito en la ecuación (3.68) se compone de tres sumandos, la *Función de Costo* que corresponde a el problema original a minimizar, el *Multiplicador de Lagrange* que corresponde al producto entre la penalización de las restricciones y la variable dual y finalmente el *Termino Proximal Aumentado* que corresponde a la norma de la restricción multiplicada por un parámetro de penalización ρ.

La *Descomposición Dual* aprovecha que el problema dual siempre es convexo, por lo que se pueden aplicar técnicas de minimización convexa. Habiendo definido el lagrangiano de este método de optimización, se puede usar el método del sub-gradiente descendiente en las variables duales [17]. Esto permite dividir la función $L_p(x, y, z)$ en tres términos para así optimizar cada variable por separado.

Para realizar esto, primero se reescribirá la ecuación (3.68) utilizando la variable auxiliar $u = y/\rho$. (esto se puede realizar debido a que ρ es una constante). El lagrangiano queda de la siguiente forma:

$$L_p(x, u, z) = f(x) + g(z) + \frac{\rho}{2}||Ax + Bz - c + u||^2 - \frac{\rho}{2}||u||^2 \tag{3.69}$$

De esta forma podemos obtener el sub-gradiente descendiente de la ecuación (3.69), obteniendo el algoritmo del método de optimización para un problema general de ADMM:

$$x_{k+1} = \arg \min_{x} f(x) + \frac{\rho}{2}||Ax + Bz_k - c + u_k||^2 \tag{3.70}$$

$$z_{k+1} = \arg \min_{z} g(z) + \frac{\rho}{2}||Ax_{k+1} + Bz - c + u_k||^2 \tag{3.71}$$

$$u_{k+1} = u_k + Ax_{k+1} + Bz_{k+1} - c \tag{3.72}$$

Notar que al utilizar el concepto de *Descomposición Dual*, para la ecuación (3.71) se utiliza la expresión x_{k+1} obtenida en la expresión anterior, esto con el fin de avanzar en dirección al gradiente para así lograr converger en un mínimo local, esto mismo ocurre en la ecuación (3.72). De este modo, el método de optimización en cada iteración da un paso en la dirección del gradiente de cada variable con el fin de encontrar el mínimo global.

3.2. ADMM formulado para MPC

En esta sección, se realizará la formulación de ADMM para *Quadratic Programing (QP)* de forma general, esta solución está basada en el paper *Embedded ADMM-based QP Solver for MPC with polytopic constraints* [18].

El problema planteado para MPC como un problema QP de forma general corresponde al siguiente:

$$\min f(x) = \frac{1}{2}x^T Hx + h^T x \tag{3.73}$$

$$\text{s.t. } Fx = f \tag{3.74}$$

$$Gx \leq g \tag{3.75}$$

La ecuación (3.75) corresponde a una restricción de desigualdad, pero ADMM sólo está definido para restricciones de igualdad. Para solucionar este problema, se utilizará la variable auxiliar z el cual permitirá reescribir la desigualdad como una igualdad. Las restricciones se pueden

reescribir de la siguiente forma:

$$\text{s.t. } Fx = f \tag{3.76}$$

$$Gx - g = z \tag{3.77}$$

$$z \geq 0 \tag{3.78}$$

Esta variable definida en (3.78) se puede plantear en la función objetivo utilizando una función indicatriz, en donde puede tomar dos valores $\{0, 1\}$. Esta función tiene valor 1 si z pertenece al conjunto definido, pero si no pertenece al conjunto, esta toma valor 0. Realizando este reemplazo y agrupando las restricciones, el problema QP general se puede representar de la siguiente forma:

$$\text{mín } f(x) + \overbrace{I(z \geq 0)}^{g(z)} \tag{3.79}$$

$$\text{s.t. } \underbrace{\begin{bmatrix} G \\ F \end{bmatrix}}_{A} x + \underbrace{\begin{bmatrix} I \\ 0 \end{bmatrix}}_{B} z = \underbrace{\begin{bmatrix} g \\ f \end{bmatrix}}_{C} \tag{3.80}$$

De esta forma, el algoritmo de ADMM para un problema general QP está dado por:

$$x^{k+1} = \arg \min_{x} f(x) + \frac{\rho}{2}||Ax + Bz^k - c + u^k||^2 \tag{3.81}$$

$$z^{k+1} = \arg \min_{z} g(z) + \frac{\rho}{2}||Ax^{k+1} + Bz - c + u^k||^2 \tag{3.82}$$

$$u^{k+1} = u^k + Ax^{k+1} + Bz^{k+1} - c \tag{3.83}$$

Estos sub problemas descritos en las ecuaciones (3.81)-(3.82) poseen una única solución por si solas y estas soluciones se puede obtener encontrando su punto crítico. Un método común y simple de implementar, es derivar la expresión e igualándola a cero. A continuación, se encontrarán los óptimos de estas ecuaciones:

La ecuación (3.81) en función de la variable x correspondiente al primer paso del algoritmo de

ADMM, para un problema QP se describe como:

$$x^{k+1} = \arg \min_{x} \frac{1}{2} x^T H x + h^T x + \frac{\rho}{2} ||Ax + Bz^k - c + u^k||^2 \tag{3.84}$$

Expandiendo la expresión cuadrática y realizando simplificaciones (se debe tener en cuenta que se manejan matrices), se obtiene:

$$x^{k+1} = \arg \min_{x} \frac{1}{2} x^T \{H + \rho A^T A\} x + \rho (Bz^k - c + u^k)^T Ax + \frac{\rho}{2} ||Bz^k - c + u^k||^2 \tag{3.85}$$

Derivando la ecuación (3.85) e igualando la a 0, se obtiene la expresión que minimiza la variable x^{k+1}:

$$x^{k+1} = (H + \rho A^T A)^{-1} \cdot \{-h - \rho A^T (Bz^k - c + u^k)\} \tag{3.86}$$

El segundo paso de ADMM corresponde a la ecuación (3.82) en función de la variable z. Para un problema QP está dada por:

$$z^{k+1} = \arg \min_{z} I(z \geq 0) + \frac{\rho}{2} ||Ax^{k+1} + Bz - c + u^k||^2 \tag{3.87}$$

Al buscar la expresión que minimiza z^{k+1}, se puede notar que esta ecuación se puede plantear como un problema de optimización dado por:

$$z^{k+1} = \arg \min_{z} \frac{\rho}{2} ||Ax^{k+1} + Bz - c + u^k||^2 \tag{3.88}$$

$$\text{s.t. } z \geq 0 \tag{3.89}$$

en donde $g(z)$ correspondiente a la función indicatriz se puede plantear como una restricción de desigualdad.

El término Bz, equivale a la siguiente matriz

$$Bz = \begin{bmatrix} I \\ 0 \end{bmatrix} z = \begin{bmatrix} z \\ 0 \end{bmatrix} \tag{3.90}$$

Reemplazando por la matriz obtenida en (3.90) se obtiene

$$z^{k+1} = \arg \min_{z} \frac{\rho}{2}||Ax^{k+1} + \begin{bmatrix} z \\ 0 \end{bmatrix} - c + u^k||^2 \tag{3.91}$$

$$\text{s.t. } z \geq 0 \tag{3.92}$$

Debido a que se está optimizando en función de z, no se tomará en cuenta la segunda fila de las matrices al no poseer esta variable. De este análisis, la función a optimizar se reduce a:

$$z^{k+1} = \arg \min_{z} \frac{\rho}{2}||Gx^{k+1} + z - g + u^k||^2 \tag{3.93}$$

$$\text{s.t. } z \geq 0 \tag{3.94}$$

Para el siguiente análisis, se cambiará la notación utilizada. Esto se realiza para facilitar la descripción del método utilizado para encontrar el óptimo de z^{k+1}.

Esta función de costo (3.93) se puede reescribir agrupando términos, para esto se define $z_i > d - a_i$ en donde se utiliza la variable auxiliar $a_i = Gx^{k+1} - g + u_1^k$ donde $u^k = [u_1^k \quad u_2^k]^T$. De esta forma la función de costos se puede expresar de la forma:

$$z_i^{k+1} = \arg \min_{z_i} \frac{\rho}{2}||z_i - a_i||^2 \tag{3.95}$$

$$\text{s.t. } z_i \geq 0 \tag{3.96}$$

La solución de este problema de optimización es igual a $max(0, a_i)$, en donde el óptimo de la función de costos tiene valor a_i y en caso de que este valor sea negativo, debido a la restricción de no negatividad, la solución del problema de optimización seria 0. Devolviendo a las expresiones originales, se obtiene que la solución de este paso de ADMM corresponde a:

$$z^{k+1} = \max\{0, g - Gx^{k+1} - u^k\} \tag{3.97}$$

Finalmente, el algoritmo de ADMM para un problema QP está dado por:

$$x^{k+1} = (H + \rho A^T A)^{-1} \cdot \{-h - \rho A^T (Bz^k - c + u^k)\} \tag{3.98}$$

$$z^{k+1} = \text{máx}\{0, g - Gx^{k+1} - u^k\} \tag{3.99}$$

$$u^{k+1} = u^k + Ax^{k+1} + Bz^{k+1} - c \tag{3.100}$$

3.3. Algoritmo de ADMM para MPC

Del análisis anterior extraído del paper de Dang [18]. Se obtiene el siguiente algoritmo de ADMM para MPC en forma densa:

Algorithm 1 Algoritmo de ADMM para MPC como un problema QP

Require: A es Invertible
  x_0, z_0, u_0 (establecer valor inicial)
  while $||r_k|| \geq \epsilon_{primal}$ or $||s_k|| \geq \epsilon_{dual}$ **do**
    $x_{k+1} = (H + \rho A^T A)^{-1} \cdot \{-h - \rho A^T (Bz_k - c + u_k)\}$
    $z_{k+1} = \max\{0, g - Gx_{k+1} - u_k\}$
    $u_{k+1} = u_k + Ax_{k+1} + Bz_{k+1} - c$
  end while

* $r_k = Ax_k + Bz_k - c$, *y* $s_k = \rho A^T B(z_{k+1} - z_k)$ *son el residuo primal y dual respectivamente.*

En donde el *residuos primal* (r_k) y *residuo dual* (s_k) se definen para diseñar el criterio de parada del algoritmo.

Debido a que este algoritmo está diseñado para una formulación QP general con restricciones de igualdad y desigualdad y la formulación de MPC descrita en el capítulo anterior sólo posee restricciones de desigualdad al estar en forma densa, se debe hacer el siguiente arreglo en las matrices de restricciones:

$$\text{s.t.} \underbrace{\begin{bmatrix} G \\ 0 \end{bmatrix}}_{A} x + \underbrace{\begin{bmatrix} I \\ 0 \end{bmatrix}}_{B} z = \underbrace{\begin{bmatrix} g \\ 0 \end{bmatrix}}_{c} \tag{3.101}$$

En donde los elementos de la fila correspondiente a las restricciones de igualdad son 0. De esta forma, las restricciones se pueden expresar utilizando la restricción definida en (2.64) para

MPC de forma densa:

$$A = G = \hat{M} \quad ; \quad B = I \quad ; \quad c = g = \hat{c} \tag{3.102}$$

$$\text{s.t.} \ \underbrace{\begin{bmatrix} \hat{M} \end{bmatrix}}_{A} x + \underbrace{\begin{bmatrix} I \end{bmatrix}}_{B} z = \underbrace{\begin{bmatrix} \hat{c} \end{bmatrix}}_{c} \tag{3.103}$$

4. Planta de interés

En este capitulo se estudiará la estructura básica de las máquinas electromecánicas, profundizando en el Motor de Corriente Continua (DC).

El objetivo de modelar el sistema es presentar un formato que se ajuste a la configuración estándar para aplicar control sobre él, que en este caso corresponde a MPC.

4.1. Sistemas Electromecánicos

La electromecánica es la aplicación híbrida que surge de la combinación de distintas áreas del conocimiento como son: electromagnetismo, electrónica, electricidad y la mecánica. Se aplica principalmente en mecanismos eléctricos, máquinas industriales, generación y transformación de energía. [19].

El rendimiento y capacidades de los sistemas electromecánicos son medidos utilizando criterios como funcionabilidad, eficiencia, estabilidad, robustez, sensibilidad, comportamiento transitorio, precisión, atenuación por perturbación, inmunidad al ruido, termodinámica, entre otros.

Las especificaciones de estos sistemas dependen de los requisitos impuestos considerando el funcionamiento completo. Por esto, para controlar un sistema electromecánico se procede a examinar la dinámica del sistema, con el objetivo de analizar y optimizar la dinámica transitoria de la relación entre la entrada, que generalmente se utiliza como una referencia y la salida.

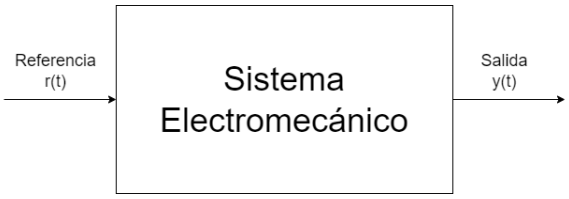


Figura 4.2: Sistema electromecánico con referencia de entrada $r(t)$ y salida $y(t)$

A partir de estos términos, se define el error de seguimiento como $e(t) = r(t) - y(t)$ el cual se busca minimizar. El comportamiento de la salida $y(t)$ será óptimo dependiendo de los criterios de rendimiento. Por ejemplo, para optimizar la dinámica del sistema de salida, uno puede minimizar el error se seguimiento y el tiempo de asentamiento, asegurando robustez.

Las respuestas transitorias se optimizan diseñando sistemas de circuito cerrado con leyes de control aplicando las funciones de desempeño que incluyen tiempo y error de seguimiento.

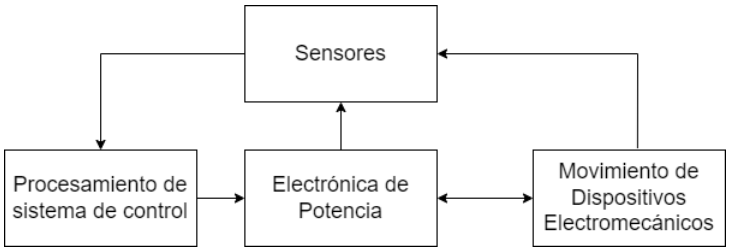


Figura 4.3: Diagrama de bloques funcionales de un sistema electromecánico

Estos sistemas electromecánicos se pueden dividir en dos categorías dependiendo de la energía que convierten estas máquinas.

- Cuando este dispositivo se utiliza principalmente para convertir energía mecánica en energía eléctrica se denomina generador.
- Cuando se utiliza principalmente para convertir energía eléctrica en energía mecánica se llama motor.

Casi todos los motores y generadores pueden convertir la energía de una a otra forma a través de la acción de campos magnéticos. [20].

En esta memoria, se enfocará en implementar una técnica de control en un motor que funciona con corriente continua.

4.2. Motor de Corriente Continua (DC)

Los motores que corriente continua, son una máquina que convierte energía eléctrica DC en energía mecánica, a partir de un flujo de corriente eléctrica que induce la acción de un campo magnético, ocasionando un movimiento rotatorio. Los fenómenos de inducción y torque electromagnéticos son los responsable de transformar la corriente continua (DC) en energía.

Basándose en el libro de Franklin [1], se describirá el funcionamiento de un motor DC.

El principio de funcionamiento se basa en la repulsión que ejercen los polos magnéticos de un imán cuando, de acuerdo con la Ley de Lorentz, interactúan con los polos magnéticos de

un electroimán (Rotor) que se encuentra montado en un eje (Eje de transmisión). De esta manera, los dispositivos de movimiento electromecánico giran y transforman energía. De la

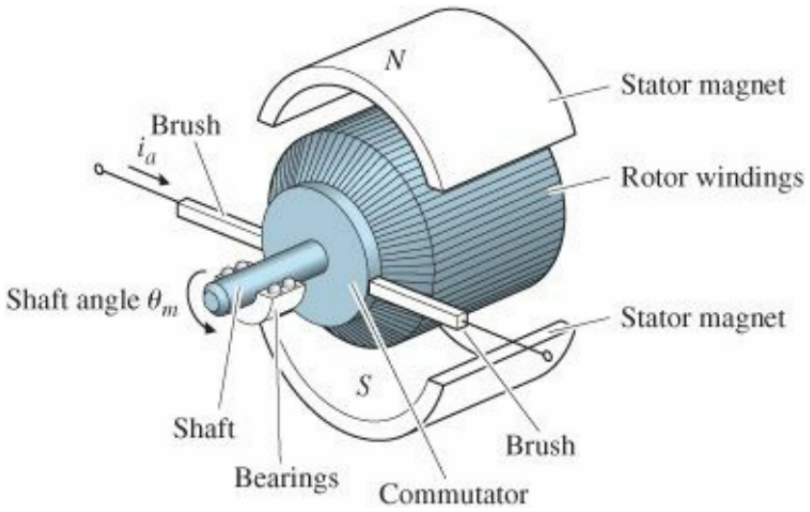


Figura 4.4: Esquema de un motor DC, Fuente: [1]

Figura 4.4, se puede apreciar el esquema de un motor DC constituido por una serie de piezas. A continuación, se realizará una breve descripción de cada una.

- *Estator:* Corresponde a parte inmóvil formada por un par de imanes permanentes que se sitúan en los extremos y recubren el resto del motor con posición fija, orientados y definidos como Norte y Sur. La función del estator es producir un campo magnético que se concentra en el inferior del motor.

- *Eje de Transmisión:* Es una pieza que atraviesa el centro del motor. Su función es transferir energía mecánica.

- *Rotor:* Este se encuentra unido al eje, este corresponde a la parte móvil del motor (giratoria). Su composición consta de un número de discos laminados en yuxtaposición, la forma de los discos presenta un espacio interior en el cual se acoplan un conjunto de bobinas que llevan la corriente eléctrica que se aplica en los terminales, así, a medida que la corriente atraviesa dichas bobinas, se produce un campo electromagnético del cual es posible controlar el tiempo y la polaridad.

- *Conmutador:* Corresponde a un anillo concéntrico situado al rededor del eje de transmisión, que ha sido segmentado en un número de placas. Los extremos de cada bobina se conectan a diferentes placas del conmutador. Gracias a esto se forma un circuito en el que fluye electricidad a distintas partes pasando por la escobilla, que luego pasan por otro segmento del conmutador ubicado en el terminal opuesto hasta llegar a la escobilla contraria.

- *Escobilla:* La electricidad que fluye por el conmutador pasa por la escobilla. La escobilla permite que, si la corriente de campo se mantiene constante, el torque desarrollado por el motor será proporcional a la corriente de armadura. Las escobillas permiten el flujo de corriente desde la fuente al devanado del rotor.

- *Corriente de campo* i_f : corresponde a la corriente que fluye a través de los devanados del estator. Esta corriente permite generar un campo magnético que interactúa con el campo eléctrico del rotor.

  Cabe destacar que existen motores que poseen imanes permanentes, estos utilizan la naturaleza de los imanes para generar este campo magnético, de modo que no necesiten esta corriente de campo para funcionar.

- *Corriente de armadura* i_a : es la corriente que fluye a través del rotor. La corriente de armadura crea un campo magnético que gira a velocidad sincrónica, este campo reacciona con el campo magnético producido por la corriente continua del devanado de campo produciéndose el torque electromagnético, por la tendencia que tienen los campos alinearse.

Estas piezas son los componentes básicos de un motor DC.

Respecto a los tamaños de estos motores, se tiene que las máquinas de corriente continua tipo servo (Motores DC pequeños), normalmente presentan excitación magnética para el campo medienta imanes permanentes, mientras que los motores de mayor tamaño tienden a tener un suministro de campo independiente para la excitación a través de una corriente de campo que deber ser suministrada externamente. En esta memoria, se utilizó un motor tipo servo debido a facilidad de manipular la instrumentación de forma cómoda y segura.

4.3. El Servomotor

Un *servomotor*, también llamado *Servo* por simplicidad, es un dispositivo de accionamiento que posee incorporado un sistema de regulación, permitiendo así que el eje del motor DC sea controlado. En particular, el objetivo propuesto consiste en regular la corriente y controlar tanto su velocidad angular, como su posición angular. Los servomotores con rangos de corriente muy pequeños se usan en instrumentos y equipo relacionado con computadores, tales como impresoras, unidades de disco y unidades de cinta. Los servomotores de corriente media y grande, se usan en sistemas robóticos, en máquinas de fresado controladas numéricamente, etcétera.

El servomotor normalmente está compuesto de un motor eléctrico de corriente continua, un potenciómetro (para medir la posición), un tacómetro (para medir la velocidad de giro) y codificador rotatorio (*encoder* en ingles). Gracias a estos componentes se consigue controlar el motor, permitiéndole al programador comandar el eje del servo, con el torque y velocidad que especifique, hacia cualquier ubicación dentro de su rango de operación e inclusive mantenerlo en dicha posición de forma estable.

En la siguiente Figura, se presenta un servo sistema que incluye un motor DC, que como se ha mencionado, al poseer la capacidad de control este adquiere la denominación de Servomotor

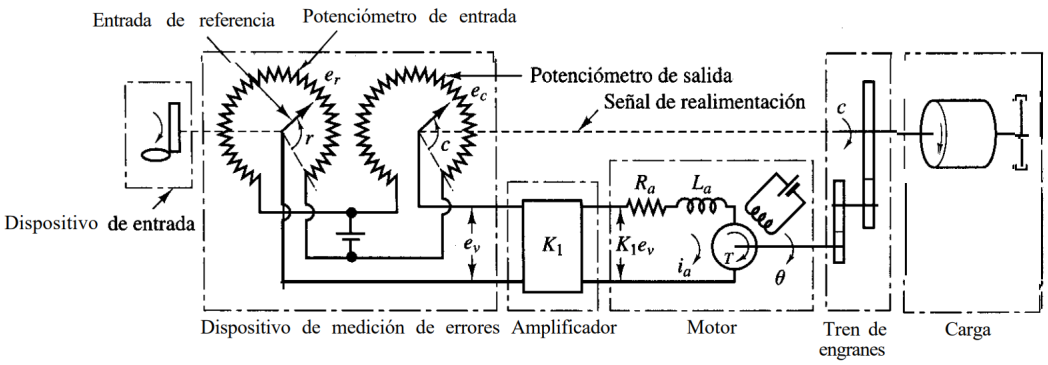


Figura 4.5: Diagrama esquemático de un servo [2]

Respecto al funcionamiento del Servo de la Figura 4.5, radica inicialmente en la acción de un par de potenciómetros que permiten medir el error. Para posteriormente transformar las posiciones de entrada y salida en señales eléctricas proporcionales.

La entrada de referencia de este sistema, es la posición angular r y el potencial eléctrico del brazo, es proporcional a r. La posición angular de salida c, depende directamente de la posición de salida del potenciómetro. Finalmente se tiene que el error de posición angular, es la diferencia entre la posición angular de entrada y la posición angular de salida, $e = r - c$.

El error de voltaje corresponde a la variable de mayor interés y se expresa como $e_v = e_r - e_c$. Existe una constante de proporcionalidad K_0 que permite obtener tanto, e_r como e_c, a partir de r y c, respectivamente. De esta forma, se tiene que $e_v = K_0 r - K_0 c$. Posteriormente ese voltaje es amplificado por una ganancia continua K_1. Siendo en definitiva ese voltaje, $e_v K_1$, el que se aplica a los terminales del circuito del motor DC.

4.4. Servomotor: *Rotatory Servo Base Unit*

Un servo disponible en la dependencia de la Universidad Técnica Federico Santa Maria, es el *Rotatory Servo Base Unit* o también nombrado como *SRV02* de la empresa Quanser. Este servomotor es una unidad para realizar experimentos de control rotatorio en laboratorios. Es ideal para introducir conceptos y teoría básica de control debido a la facilidad manipulación. El motor de corriente continua de este servomotor, impulsa un engranaje de piñón más pequeño, fijado a un engranaje medio más grande que gira sobre el eje de carga. La posición del eje de carga se mide utilizando una alta resolución de un codificador óptico (encoder) o un potenciómetro, ambos disponibles como parte del sistema.

Figura 4.6: Srv02, Quanser

Las especificaciones de este servomotor presentadas por la empresa Quanser son:

Tabla 4.1: Características de SRV02

Dimensiones (LxWxH)	15cm x 15cm x 18cm
Masa	$1.2Kg$
Voltaje Nominal de entrada	$6V$
Máxima corriente continua	$1A$
Máxima velocidad de motor	$6000RPM$
Potencia de polarización del Potenciómetro	$\pm 12V$
Rango de Medición del Potenciómetro	$\pm 5V$
Resolución Encoder (en cuadratura)	4096 $cuentas/rev$

A continuación, se describirá el modelo del servomotor SRV02.

4.5. Modelo matemático de Servo SRV02

Basándose en el *workbook* realizado por Quanser , el circuito esquemático y sistema mecánico de un motor DC se puede apreciar en la siguiente Figura 4.7

A partir del sistema eléctrico y mecánico planteado en la Figura 4.7, se realizará un análisis eléctrico y mecánico por separado. Teniendo descrito ambos sistemas, se obtendrá un modelo electromecánico del servomotor.

4.5.1. Ecuaciones eléctricas del motor

Respecto a la Figura 4.7, se tiene que R_m es la resistencia del motor, L_m es la inductancia del motor, k_m es la constante del *back-EMF*.

Analizando la parte eléctrica del motor, se tiene que el voltaje $e_b(t)$ de *back-EMF* depende de la velocidad del eje del motor w_m, y la constante *back-emf* del motor k_m. Si se opone el flujo

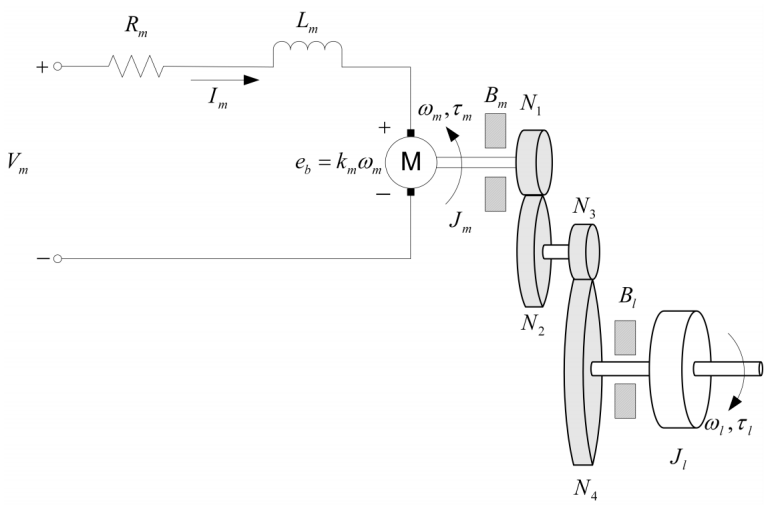


Figura 4.7: Circuito esquemático y sistema mecánico de Motor DC

de corriente, el voltaje de la fuerza contra electromotiz esta dada por:

$$e_b(t) = k_m w_m(t) \tag{4.104}$$

Usando la ley de voltaje de Kirchoff, podemos escribir la siguiente ecuación del sistema eléctrico:

$$V_m(t) - R_m I_m(t) - L_m \frac{\partial I_m(t)}{\partial t} - k_m w_m(t) = 0 \tag{4.105}$$

Debido a que la inductancia del motor L_m es mucho más pequeña que la resistencia del motor, se puede despreciar esta variable. De esta forma, se tiene la siguiente ecuación:

$$V_m(t) - R_m I_m(t) - k_m w_m(t) = 0 \tag{4.106}$$

Despejando la corriente $I_m(t)$, describimos la relación de esta como

$$I_m(t) = \frac{V_m(t) - k_m w_m(t)}{R_m} \tag{4.107}$$

4.5.2. Ecuaciones mecánicas del motor

Respecto al sub sistema mecánico, este describirá la velocidad del eje de carga w_l respecto a torque τ_m aplicado a este. Debido a que el sistema del motor DC es un sistema rotativo de un grado de libertad, según la según la segunda ley de Newton podemos describir la velocidad de este como:

$$J \cdot \alpha = \tau \tag{4.108}$$

donde J es el momento de inercia del cuerpo (respecto al centro de masa), α es la aceleración angular del sistema, y tau es la suma de los torques aplicados el cuerpo.

Según la Figura 4.7, la base de los engranajes posee una fricción viscosa B_m que actúa sobre el eje del motor. Además se considera una fricción viscosa en el eje de la carga B_l. La ecuación de esta carga esta dada por:

$$J_l \frac{dw_l(t)}{dt} + B_l w_l(t) = \tau_l(t) \tag{4.109}$$

donde J_l es el momento de inercia de la carga y τ_l es el torque total aplicado a la carga. La inercia en la carga incluye la inercia desde los engranajes y desde alguna carga externa anclada.

El eje del motor se puede representar como:

$$J_m \frac{dw_m(t)}{dt} + B_m w_m(t) + \tau_{ml}(t) = \tau_m(t) \tag{4.110}$$

donde J_m es el momento de inercia del eje del motor y τ_{ml} es el torque resultante en el eje del motor hasta el torque de carga. El toque de carga del eje aplicado en el motor puede ser descrito por:

$$\tau_l(t) = \eta_g K_g \tau_{ml}(t) \tag{4.111}$$

donde K_g es el radio de los engranaje y η_g es la eficiencia de la caja de engranaje. La caja de cambio es directamente montada en la base del motor, este es representado por N_1 y N_2 que

corresponde a la relación de engranajes de

$$K_{gi} = \frac{N_2}{N_1} \tag{4.112}$$

la cual corresponde al radio interno de la caja de engranaje. El engranaje del motor N_3 y los engranajes de la carga N_4 están directamente endentados juntos y son visibles desde fuera. Estos engranajes poseen la siguiente relación:

$$K_{ge} = \frac{N_4}{N_3} \tag{4.113}$$

Juntando amabas relaciones entre las cajas de engranajes, se tiene que:

$$K_g = K_{ge} K_{gi} \tag{4.114}$$

Entonces, el torque visto en el eje del motor puede ser expresado como:

$$\tau_{ml}(t) = \frac{\tau_l(t)}{\eta_g K_g} \tag{4.115}$$

Además, aprovechando la relación definida entre los engranajes, se puede decir que

$$\theta_m(t) = K_g \theta_l(t) \tag{4.116}$$

A partir de la anterior relación, podemos definir la relación entre las velocidades angular del eje del motor y el eje de la carga derivando en el tiempo. Se tiene

$$w_m(t) = K_g w_l(t) \tag{4.117}$$

Utilizando las definiciones realizadas para cada componente mecánico del sistema, se pueden relacionar con la ecuación de balance de energías del motor. Esta se describe como

$$J_m K_g \frac{dw_l(t)}{dt} + B_m K_g w_l(t) + \frac{J_l \left(\frac{dw_l(t)}{dt}\right) + B_l w_l(t)}{\eta_g K_g} = \tau_m(t) \tag{4.118}$$

Reescribiendo esta expresión y agrupando términos, el sistema mecánico se puede describir como

$$J_{eq}\frac{dw_l(t)}{dt} + B_{eq}w_l(t) = \eta_g K_g \tau_m(t) \tag{4.119}$$

con:

$$J_{eq} = \eta_g K_g^2 J_m + J_l \tag{4.120}$$

$$B_{eq} = \eta_g K_g^2 B_m + B_l \tag{4.121}$$

4.5.3. Combinación de ecuaciones eléctrica y mecánica

En esta sección se combinarán las ecuaciones eléctricas con las ecuaciones mecánicas descritas anteriormente con el fin de describir la velocidad angular del motor respecto al voltaje de entrada en este.

El torque del motor es proporcional al voltaje aplicado, esto se describe como:

$$\tau_m(t) = \eta_m k_t I_m(t) \tag{4.122}$$

donde k_t es la constante de corriente-torque (Nm/A), η_m es la eficiencia del motor y I_m es la corriente de armadura del motor. Se puede expresar este torque respecto a $V_m(t)$ y la velocidad en la carga del eje utilizando la ecuación (4.107). Por lo tanto, se tiene que:

$$\tau_m(t) = \frac{\eta_m k_t (V_m(t) - k_m w_m(t))}{R_m} \tag{4.123}$$

Utilizando la ecuación (4.117), que relaciona $w_m(t)$ con $w_l(t)$, se obtiene

$$\tau_m(t) = \frac{\eta_m k_t (V_m(t) - k_m K_g w_l(t))}{R_m} \tag{4.124}$$

Utilizando la ecuación (4.119) obtenido al final del modelado mecánico, se cumple que:

$$J_{eq}\frac{dw_l(t)}{dt} + B_{eq}w_l(t) = \frac{\eta_g K_g \eta_m k_t (V_m(t) - k_m K_g w_l(t))}{R_m} \tag{4.125}$$

De esta forma, se obtiene una expresión que posee como variable el voltaje de entrada $V_m(t)$ y la velocidad angular de la carga $w_l(t)$. Agrupando y reordenando términos, se obtiene la siguiente ecuación:

$$J_{eq}\frac{dw_l(t)}{dt} + B_{eq,v}w_l(t) = A_m V_m(t) \tag{4.126}$$

donde:

$$B_{eq,v} = \frac{\eta_g K_g^2 \eta_m k_t k_m + B_{eq} R_m}{R_m} \tag{4.127}$$

$$A_m = \frac{\eta_g K_g \eta_m k_t}{R_m} \tag{4.128}$$

$$J_{eq} = \eta_g K_g^2 J_m + J_l \tag{4.129}$$

$$B_{eq} = \eta_g K_g^2 B_m + B_l \tag{4.130}$$

En donde (4.126) corresponde a un sistema de primer orden.

Los parámetros definidos para este sistema electromecánico, según el datasheet del servo SRV02 son:

Symbol	Description	Matlab Variable	Value	Variation
V_{nom}	Motor nominal input voltage		6.0 V	
R_m	Motor armature resistance	Rm	2.6 Ω	$\pm$ 12%
L_m	Motor armature inductance	Lm	0.18 mH	
k_t	Motor current-torque constant	kt	7.68×10^{-3} N m/A	$\pm$ 12%
k_m	Motor back-emf constant	km	7.68×10^{-3} V/(rad/s)	$\pm$ 12%
K_g	High-gear total gear ratio	Kg	70	
	Low-gear total gear ratio	Kg	14	
η_m	Motor efficiency	eta_m	0.69	$\pm$ 5%
η_g	Geabox efficiency	eta_g	0.90	$\pm$ 10%
$J_{m,rotor}$	Rotor moment of inertia	Jm_rotor	3.90×10^{-7} kg $\cdot$ m^2	$\pm$ 10%
J_{tach}	Tachometer moment of inertia	Jtach	7.06×10^{-8} kg $\cdot$ m^2	$\pm$ 10%
J_{eq}	High-gear equivalent moment of inerta without external load	Jeq	9.76×10^{-5} kg $\cdot$ m^2	
	Low-gear equivalent moment of inerta without external load	Jeq	2.08×10^{-5} N $\cdot$ m / (rad/s)	
B_{eq}	High-gear Equivalent viscous damping coefficient	Beq	0.015 N $\cdot$ m / (rad/s)	
	Low-Gear Equivalent viscous damping coefficient	Beq	1.50×10^{-4} kg $\cdot$ m^2	
m_b	Mass of bar load	m_b	0.038 kg	
L_b	Length of bar load	L_b	0.1525 m	
m_d	Mass of disc load	m_d	0.04 kg	
r_d	Radius of disc load	r_d	0.05 m	
m_{max}	Maximum load mass		5 kg	
f_{max}	Maximum input voltage frequency		50 Hz	
I_{max}	Maximum input current		1 A	
ω_{max}	Maximum motor speed		628.3 rad/s	

Figura 4.8: Especificaciones de Servomotor SRV02

5. Modelado del sistema

A partir del modelo matemático definido para el Servo SRV02, este corresponde a un sistema de primer orden con entrada $V_m(t)[V]$ y salida $w_l(t)[rad/s]$. Se utilizará $w = w_l$ para el siguiente análisis en donde este sistema se puede representar de la forma:

Figura 5.9: Sistema Servo Motor

Según las especificaciones definidas por el fabricante del servomotor SRV02, hay parámetros que pueden variar hasta un $\pm 12\,\%$. Estas variaciones podrían afectar en la técnica de control empleada, debido a que las predicciones se obtienen a partir del modelo matemático del sistema. Es por esto por lo que se obtendrá un modelo del sistema a partir de la respuesta escalón.

5.1. Respuesta Escalón

El bloque 'servo motor' se puede modelar como un sistema de primer orden. Este se modelará como una función de transferencia $H(s)$ en el dominio de Laplace, relacionado como

$$\frac{w(s)}{V_m(s)} = H(s) = \frac{K}{\tau s + 1} \tag{5.131}$$

en donde K corresponde a la ganancia del sistema y τ corresponde a la ubicación del polo del sistema.

Para modelar este sistema, se utilizará como entrada una señal escalón de la forma:

$$V_m(s) = \frac{\alpha}{s} \tag{5.132}$$

en donde α corresponde a la ganancia de la señal escalón. De esta forma, reemplazando la

señal de entrada se tiene:

$$w(s) = \frac{K}{\tau s + 1} \cdot V_m(s) = \frac{K}{\tau s + 1} \cdot \frac{\alpha}{s} \tag{5.133}$$

Para obtener K, se debe analizar el sistema en estado estacionario. A partir de la señal de salida y utilizando el teorema del valor final se cumple que

$$\lim_{t\to\infty} w(t) = \lim_{s\to 0} w(s) \cdot s = \lim_{s\to 0} H(s) \cdot V_m(s) \cdot s = \lim_{s\to 0} \frac{K}{\tau s + 1} \cdot \frac{\alpha}{s} \cdot s = K \cdot \alpha \tag{5.134}$$

De esta forma se obtiene que $w_\infty = K \cdot \alpha$. Debido a que α es la ganancia de la señal de entrada utilizada en el experimento, y w_∞ es la velocidad angular en estado estacionario medida con un sensor, se puede calcular K. Se obtiene:

$$K = \frac{w_\infty}{\alpha} \tag{5.135}$$

Para obtener τ, se debe analizar el transiente del sistema. A partir de la respuesta escalón definida en (5.133), utilizando fracciones parciales se tiene el sistema como:

$$w(s) = -\frac{K \cdot \tau \cdot \alpha}{\tau s + 1} + \frac{K \cdot \alpha}{s} \tag{5.136}$$

Aplicando la transformada inversa de laplace, en el dominio del tiempo se cumple

$$w(t) = \mu(t)\left(-K \cdot \alpha \cdot e^{\frac{-t}{\tau}} + K \cdot \alpha\right) = \mu(t) \cdot K \cdot \alpha \left(1 - e^{\frac{-t}{\tau}}\right) \tag{5.137}$$

Analizando el transiente del sistema a partir de la ecuación (5.137), este se rige por $e^{\frac{-t}{\tau}}$. Este retardo depende directamente del valor de τ.

Los sistemas de primer orden dependen directamente del instante $t = t_x \cdot \tau$, este permite informar en que punto el sistema converge al estado estacionario. Algunos casos útiles para estimar τ son:

- Caso 1: Para $t_1 = 4\tau$, se cumple que $w(t_1) = w_\infty \cdot 98.17\,\%$
- Caso 2: Para $t_2 = \tau$, se cumple que $w(t_2) = w_\infty \cdot 63.21\,\%$

Se descartó el Caso 1, debido a que puede existir ruido en la señal de medición, provocando un grado de incertidumbre al analizar el sistema cerca del estado estacionario. De esta forma, se escoge el Caso 2 para obtener el valor de τ del sistema, debido a que este caso es menos propenso al ruido.

A partir del análisis realizado en esta sección, utilizando la respuesta escalón, podemos modelar un sistema de primer orden. Los pasos para modelar el sistema utilizando la respuesta escalón son:

1. Mantener el sistema sin excitación externa ($u(t) = 0$) durante un tiempo, para que las condiciones iniciales sean 0.
2. Aplicar una señal escalón al sistema de ganancia α.
3. Determinar la salida en estado estacionaria del sistema, w_∞.
4. A partir de w_∞ y un α dado, calcular $K = \frac{w_\infty}{\alpha}$.
5. Calcular $w(t_2) = w_\infty \cdot 63.21\,\%$
6. Buscar punto en el gráfico que posea $w(t_2)$. Este punto posee coordenada $\{t_2, w(t2)\}$ en donde $\tau = t_2$. (en caso de que exista un delay en la señal escalón utilizada, se debe sustraer este retardo).

Modelando el sistema utilizando un escalón de ganancia $\alpha = 1$, se obtuvo el gráfico 5.10.

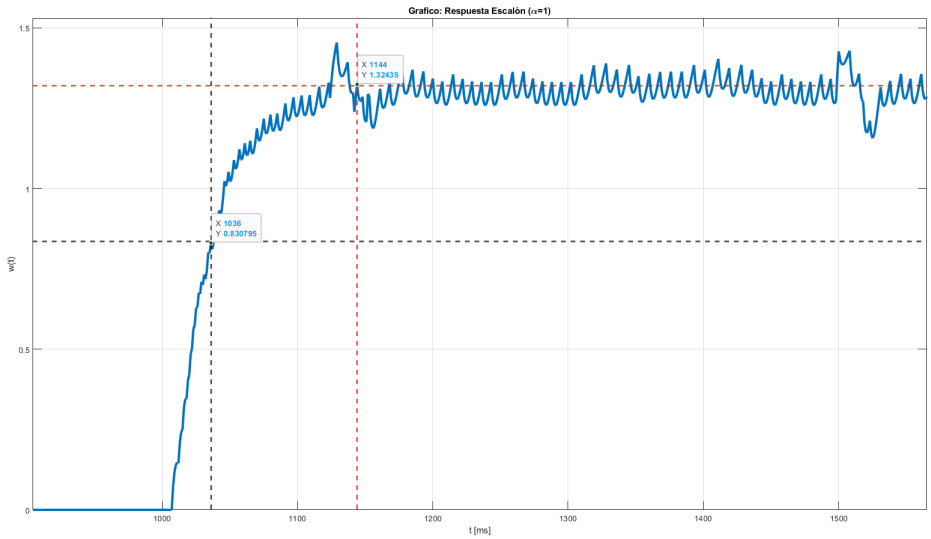


Figura 5.10: Gráfico Respuesta Escalón con $\alpha = 1$

La traza azul corresponde a la velocidad angular $w(t)$, las trazas rojas corresponden al punto donde $t = 4\tau$ y las trazas negras corresponden al punto donde $t = \tau$. Se obtuvo $w_\infty = 1.3208$ calculando el promedio de $w(t)$ posterior a las trazas rojas. Luego, a partir de w_∞ se calculó $K = 1.3208$. Finalmente, a partir del punto obtenido en las trazas negras, se obtiene $\tau = 0.034$. A partir de esta respuesta escalón, se obtiene un modelo lineal del sistema como:

$$\frac{w(s)}{V_m(s)} = H(s) = \frac{1.3208}{0.034 \cdot s + 1} \tag{5.138}$$

5.2. Respuesta Escalón para distintas excitaciones

Debido a que las mediciones realizadas por el encoder $w(t)$, se encuentran contaminadas por ruido como se puede apreciar en la Figura 5.10 obtenida en la sección anterior, se debe obtener una mejor estimación de los parámetros K y τ. El transiente definido por τ no debería variar mucho debido a que los polos dependen de la dinámica del sistema, pero la ganancia K puede variar dependiendo de la amplitud α aplicada al sistema.

Para obtener una mejor estimación, se repitió el experimento realizado en la sección anterior

utilizando distintos valores de α (señal escalón). Se obtuvieron $n = 23$ muestras utilizando un $\alpha = [0.6, 0.8, 1.0, 1.2, ..., 4.6, 4.8, 5.0]$ como señal de entrada $V(s) = \alpha/s$.

Para el caso de τ, a partir de las n muestras obtenidas, se calculó la varianza de estas muestras. Se obtuvo $P_\tau = 6.0079 \cdot 10^{-7}$, el cual representa a una desviación estándar de $\sigma_\tau = 7.7511 \cdot 10^{-4}$. Estos resultados demuestran que la variación de los τ_n obtenidos es muy pequeña, por lo que se considerará que τ es constante. Se utilizará la media muestral para calcular τ_n. Este calculo está definido por:

$$\tau_n = \frac{1}{n} \sum_{i=1}^{n} \tau_i \tag{5.139}$$

A partir de la media muestral se obtiene $\tau_n = 0.0347$.

Respecto a K, del análisis realizado en la sección anterior, se tiene que este está dado por $w_\infty = K \cdot \alpha$. Al realizar un gráfico XY entre la entrada y salida para n experimentos se tiene:

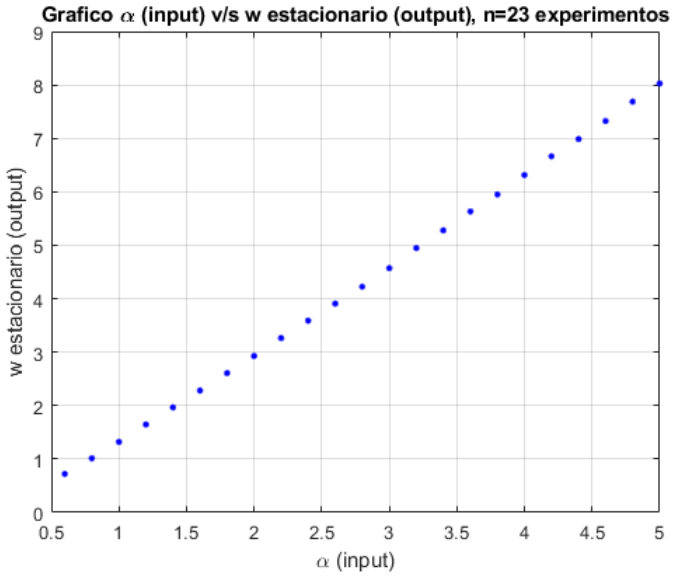


Figura 5.11: Gráfico de entrada y salida estacionaria para $n = 23$ muestras

De la Figura 5.11, se puede apreciar que la relación está dada por una recta con pendiente positiva mayor a 0. Nótese que esta recta no toca el cero. Es por esto que se estimará K a

partir de n experimentos, como una recta dada por

$$w_\infty(n) = \alpha(n) \cdot K + \bar{E}(n) \tag{5.140}$$

donde $\bar{E}$ corresponde a una variable aleatoria que modela errores de medición. Se supone que posee media $\mu_{\bar{E}} = 0$ y varianza $P_{\bar{E}} = \sigma^2$.

Se utilizó el estimador de mínimos cuadrados para obtener una estimación de $\hat{K}$. Este es un estimador BLUE (best linear unbiased estimator) definido como:

$$\hat{K}_{\text{LS}} = (M^T M)^{-1} M^T Y \tag{5.141}$$

donde

$$M = \begin{bmatrix} \alpha(1) \\ \alpha(2) \\ \vdots \\ \alpha(n) \end{bmatrix} \quad ; \quad Y = \begin{bmatrix} w_\infty(1) \\ w_\infty(2) \\ \vdots \\ w_\infty(n) \end{bmatrix} \tag{5.142}$$

Al implementar mínimos cuadrados, se obtiene $\hat{K}_{\text{LS}} = 1.55953$. Repitiendo el gráfico entrada v/s salida, pero incorporando la estimación del sistema utilizando $\hat{K}_{\text{LS}}$, se obtuvo el gráfico de la Figura 5.12 . Se evidencia que esta estimación se comporta similar a la original, pero debido a la estimación utilizada, esta recta sí para por el 0.

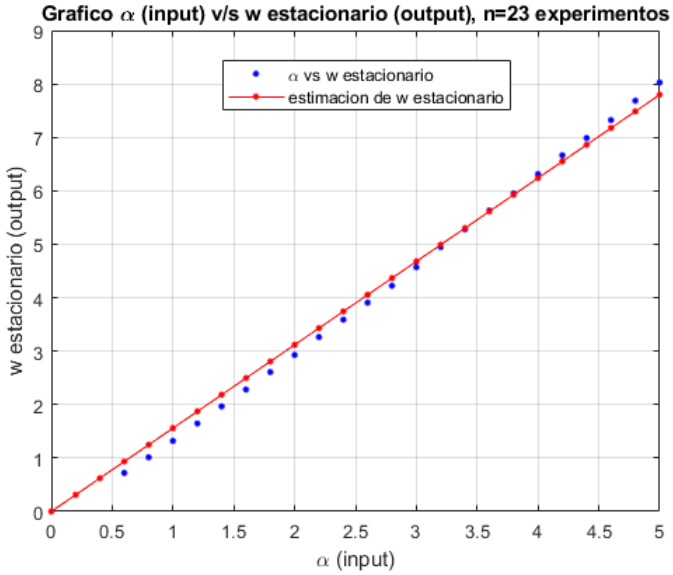


Figura 5.12: Gráfico de entrada y salida estacionaria para $n = 23$ muestras, con estimación.

Las muestras obtenidas de $\tau(n)$ y $K(n)$ se puede apreciar en la Figura 5.13 y Figura 5.14 respectivamente. Donde los puntos azules corresponden a la muestras obtenidas, la traza naranja corresponde a la estimación a partir de la media y la traza amarilla corresponde a la estimación a partir de mínimos cuadrados.

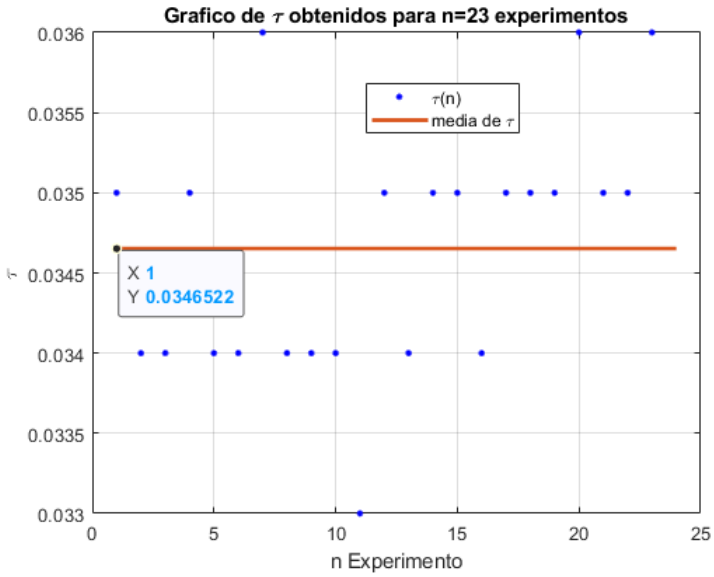


Figura 5.13: Gráfico con estimaciones de τ obtenidos para $n = 23$ muestras

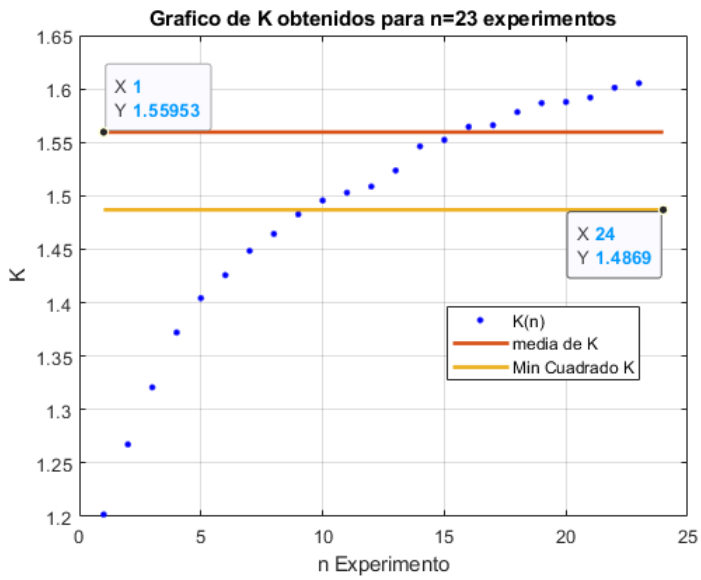


Figura 5.14: Gráfico con estimaciones de K obtenidos para $n = 23$ muestras

Finalmente, a partir de las estimaciones realizadas, se obtiene el siguiente modelo lineal del servo motor SRV02:

$$\frac{w(s)}{V_m(s)} = H(s) = \frac{1.55953}{0.0347 \cdot s + 1} \tag{5.143}$$

5.3. Modelo utilizado para técnica de Control

Para implementar la técnica de control MPC, se utilizó el modelo en su forma no normalizada. Esta representa a la misma función de transferencia, pero escrito de otra forma.

Esta función de transferencia está dada de la forma:

$$H(s) = \frac{b}{s + a} \tag{5.144}$$

Reescribiendo (5.143) de la forma planteada en (5.144), se obtiene:

$$H(s) = \frac{45.0051}{s + 28.8582} \tag{5.145}$$

Además, en esta memoria se desea controlar el sistema según la posición angular del servo motor $\theta(t)$, pero la función de transferencia planteada en (5.145) tiene como salida la velocidad

angular $w(t)$. Para compensar esto, sabemos que:

$$w(t) = \frac{d\theta(t)}{dt} \tag{5.146}$$

Esta relación expresada en el dominio de Laplace corresponde a

$$w(s) = s \cdot \theta(s) \tag{5.147}$$

Reemplazando (5.147) en la función de transferencia planteada en (5.145), se tiene

$$H(s) = \frac{w(s)}{V_m(s)} = \frac{s \cdot \theta(s)}{V_m(s)} \tag{5.148}$$

De esta forma, moviendo el término s hacia el otro lado de la expresión, se obtiene la relación entre la posición angular $\theta(s)$ y el voltaje de salida $V_m(s)$

$$\frac{\theta(s)}{V_m(s)} = \frac{H(s)}{s} \tag{5.149}$$

Finalmente, se define $G(s)$ como el modelo del servo motor de segundo orden dado por

$$G(s) = \frac{45.0051}{s(s + 28.8582)} \tag{5.150}$$

con entrada $V_m(s)$ y salida $\theta(s)$.

6. Muestreo del Sistema

Para conseguir que un sistema sea controlado de forma digital, por ejemplo, a través de un computador, es necesario que exista una comunicación coordinada entre la planta y el controlador.

Al realizar implementaciones en un computador, este recibe mediciones del proceso en tiempo discreto y genera una señal de entrada en tiempo discreto. Por otro lado, la dinámica del sistema físico corresponde a un proceso de tiempo continuo, el cual no se mide en momentos concretos.

Al utilizar un controlador en un computador, es imprescindible convertir las señales continuas medio físico en la planta (servo motor en este caso), en señales discretas que pueda interpretar el controlador. Esto se consigue utilizando una técnica de muestreo.

Existen recursos como los que ofrecen los conversores digital analógicos, los cuales permiten transformar las acciones de control de carácter digital, en señales analógicas continuas en el tiempo con sigla *D-A*. También, existen conversores que permiten realizar el proceso inverso, es decir, convierten una señal analógica continua en una señal digital con sigla *A-D*. Esta idea está plasmada en el diagrama de la Figura 6.15 en donde las señales $u(k)$ y $y(k)$ corresponde a las señales discretas que utiliza el computador, y las señales $u(t)$ y $y(t)$ corresponden a las señales de tiempo continuo que operan en el sistema real.

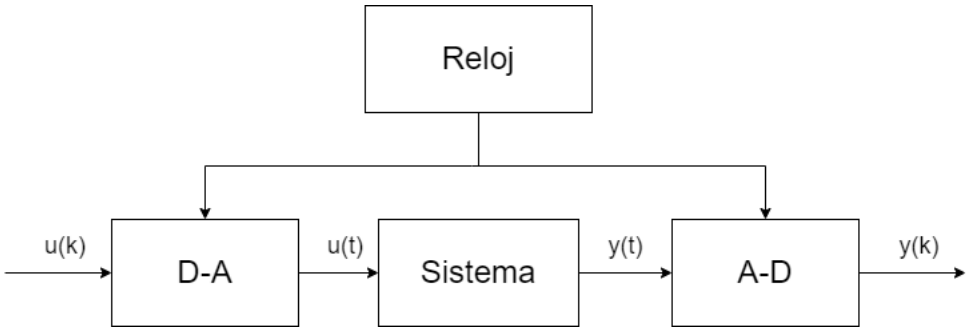


Figura 6.15: Diagrama de bloque de sistema de tiempo continuo conectado a convertidor A-D y D-A

Este proceso de reconstruir la señal, para este caso, viene dado por un Retenedor de Orden Cero (ZOH, por sus siglas en inglés Zero Order Hold), el cual se describirá en la siguiente

sección.

6.1. Muestreo de un sistema utilizando un Retenedor de Orden Cero (ZOH)

El retenedor de orden cero es el más sencillo de los reconstructores. Su funcionamiento se basa en congelar el valor de la muestra presente hasta que se tiene la siguiente muestra. El retenedor de orden cero no solamente se utiliza para reconstruir señales.

De esta forma, empleando en la Figura 6.16, el cual presenta el procedimiento para realizar el control digital de una planta $G(s)$.

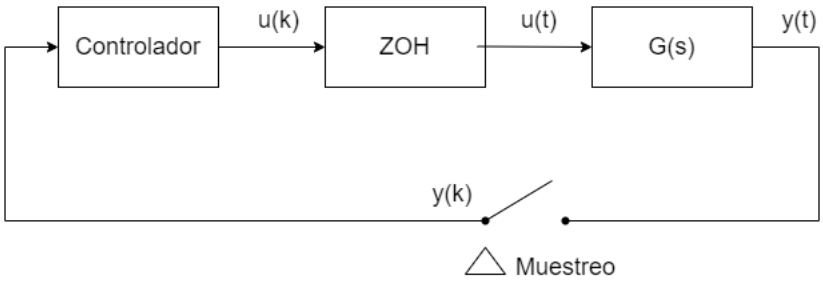


Figura 6.16: Diagrama de bloque de sistema de un sistema muestreado con una planta en tiempo continuo y un ZOH

Cabe mencionar que todo este sistema de control debe funcionar coordinadamente, para ello, es necesario sincronizar los relojes del muestreador, del retenedor (ZOH) y del controlador.

El muestreo tiene la función de convertir la señal continua (analógica) proveniente del "mundo real" en una señal discreta (digital). Esto lo consigue realizando un proceso de muestreo de la señal continua en ciertos instantes de tiempo discreto, para ese caso definido como "Tiempo de muestreo h".

El controlador, a partir de la señal $y(k)$ utiliza un algoritmo de control para generar la señal de entrada $u(k)$. Para que esta señal pueda ingresar en la planta $G(s)$ es necesario convertir dichos valores en una señal continua en el tiempo, en particular y por simplicidad, se utiliza un retenedor de orden cero (ZOH) para mantener constante el valor entregado por el controlador

durante un tiempo específico de muestreo h.

De esta forma, la planta equivalente discreta está dada por

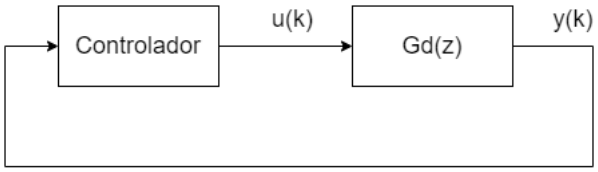


Figura 6.17: Diagrama de bloque de sistema muestreado considerando una planta discreta

donde $G_d(z)$ corresponde a la planta $G(s)$ discretizada en un tiempo de muestreo h segundos. Está dada por

$$G_d(z) = (1 - z^{-1}) \cdot \mathcal{Z}\left\{\mathcal{L}^{-1}\left\{\frac{b}{s(s+a)}\right\}_{t=h}\right\} \tag{6.151}$$

6.2. Representación del sistema dado un tiempo de muestreo especifico

Según el capítulo anterior, se tiene un modelo del servo motor de la forma

$$G(s) = \frac{b}{s(s+a)} \tag{6.152}$$

con $b = 45.0051$ y $a = 28.8582$.

Como se describió en este capítulo, se requiere poseer un modelo discreto del sistema para implementar una técnica de control utilizando un computador. Además, el modelo matemático que utiliza MPC debe corresponde a una representación en espacio de estados. Por lo que se debe obtener una representación de estados en tiempo discreto a partir de la función de transferencia en tiempo continuo planteada en (6.152). Se sabe que existen 4 formas de representar un sistema, esto está plasmado en la siguiente figura:

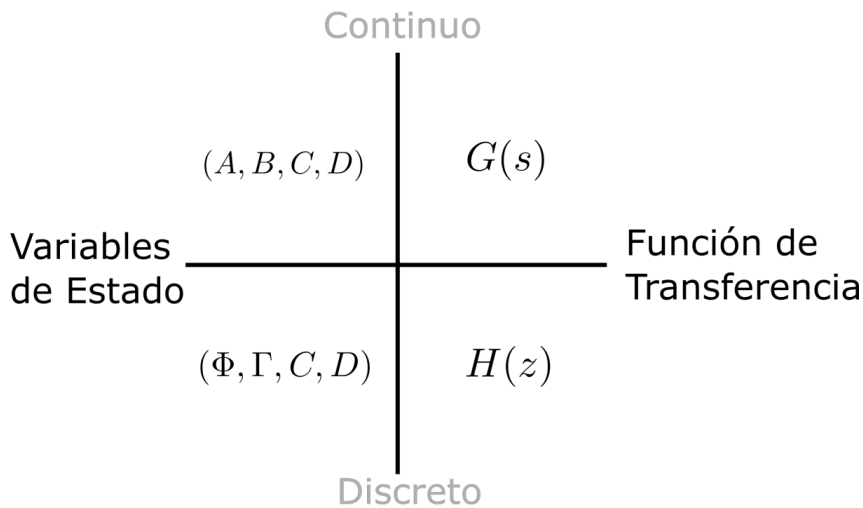


Figura 6.18: Representación de sistemas

Según la Figura 6.18, existen dos caminos para poder obtener la representación del sistema en variables de estado en tiempo discreto a partir de $G(s)$. Se utilizará el camino antihorario planteado según la Figura 6.18.

Se tiene que

$$G(s) = \frac{b}{s(s+a)} \quad ; \quad Y(s) = G(s)U(s) \tag{6.153}$$

en donde $y = \theta$ e $u = V_m$ representan a la salida y entrada del sistema respectivamente (se utilizará estas variables para obtener una representación general de un modelo de segundo orden).

Primero, se obtendrá la representación de estados en tiempo continuo. Utilizando la variable auxiliar $Q(s)$ y reordenando términos, se tiene

$$\frac{Y(s)}{U(S)} = \frac{1}{\frac{s(s+a)}{b}} \cdot \frac{Q(s)}{Q(s)} \tag{6.154}$$

A partir de esta variable auxiliar, la ecuación (6.154) se divide en dos expresiones dadas por

$$U(s) = \frac{s(s+a)}{b} \cdot Q(s) \quad ; \quad Y(s) = Q(s) \tag{6.155}$$

Aplicando la transformada inversa de laplace, se tiene

$$b \cdot u(t) = \ddot{q}(t) + a \cdot \dot{q}(t) \quad ; \quad y(t) = q(t) \tag{6.156}$$

Definiendo el vector de estados como $x(t) = [x_1(t) \quad x_2(t)]^T$, dado por

$$x_1(t) = \dot{q}(t) \quad \Rightarrow \quad \dot{x}_1(t) = \ddot{q}(t) \tag{6.157}$$

$$x_2(t) = q(t) \quad \Rightarrow \quad \dot{x}_2(t) = \dot{q}(t) = x_1(t) \tag{6.158}$$

Reemplazando las variables de estado en (6.156) y agrupando términos, se tiene la siguiente representación de estados de $G(s)$:

$$\dot{x}(t) = \underbrace{\begin{bmatrix} -a & 0 \\ 1 & 0 \end{bmatrix}}_{A} x(t) + \underbrace{\begin{bmatrix} b \\ 0 \end{bmatrix}}_{B} u(t) \tag{6.159}$$

$$y(t) = \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{C} x(t) \tag{6.160}$$

Teniendo la representación de estados del sistema en tiempo continuo, faltaría discretizar esta representación utilizando un tiempo de muestreo h. La representación de variables de estado en tiempo discreto corresponde a:

$$x_{k+1} = A_d x_k + B_d u_k \tag{6.161}$$

$$y_k = C_d x_k \tag{6.162}$$

Se sabe que la matriz $C_d = C$. Para obtener la matriz A_d y B_d, primero se debe presentar la matriz exponencial, cuya definición generalizada se funda a partir de la Serie de Taylor.

En particular, esta matriz exponencial corresponde a la función exponencial de la serie de Maclaurin, que corresponde a una particularidad de la serie de Taylor, cuya característica es

que se encuentra centrada en el origen. Esta dada por

$$e^A = I + A + \frac{A^2}{2!} + ... \tag{6.163}$$

Algunas propiedades matemáticas que conviene destacar de esta matriz exponencial son:

- Al derivar una exponencial que posee como exponente una matriz, es válida la propiedad de conmutatividad entre la matriz exponencial y la matriz constante.

$$\frac{d(e^{At})}{dt} = Ae^{At} = e^{At}A \tag{6.164}$$

- Considerando un tiempo de muestre Δh, la equivalencia de la siguiente ecuación, puede ser comprendida como que el límite de Δh tendiendo a 0 provoca un desfase en la serie de Taylor.

$$\lim_{h \to 0} \frac{e^{Ah} - I}{h} = A \tag{6.165}$$

- En el supuesto de que e^{-A} es el inverso de e^A es posible utilizar propiedades de las potencias, además, es admisible trabajar aplicando conmutatividad entre las matrices.

$$AB = BA \Rightarrow e^A e^B = e^B e^A = e^{A+B} \tag{6.166}$$

Continuando con el análisis, para obtener las matrices en su forma discreta, se debe multiplicar la ecuación de estado $\dot{x}(t)$ por la matriz exponencial e^{-A} a ambos lados de la igualdad. Desarrollando esta idea, se tiene

$$e^{-At}\dot{x}(t) = e^{-A}Ax(t) + e^{-At}Bu(t) \Rightarrow \frac{d}{dt}\left\{e^{-At}x(t)\right\} = e^{-A}Bu(t) \tag{6.167}$$

$$\Rightarrow e^{-At_{k+1}}x(t_{k+1}) - e^{-At_k}x(t_k) = \int_{t_k}^{t_{k+1}} e^{-At}Bu(t)dt \tag{6.168}$$

$$\Rightarrow x(t_{k+1}) = e^{-A(t_{k+1}-t_k)}x(t_k) + u(t_k)\int_{t_k}^{t_{k+1}} e^{-A(t_{k+1}-t)}Bdt \tag{6.169}$$

De la ecuación (6.169) y considerando que $h = t_{k+1} - t_k$ y $\eta = t_{k+1} - t$, se desprende lo

siguiente:

$$A_d = e^{-A(t_{k+1}-t_k)} = e^{Ah} \tag{6.170}$$

$$B_d = \int_{t_k}^{t_{k+1}} e^{-A(t_{k+1}-t)} B dt = \int_h^0 e^{A\eta} B d(-\eta) = \int_0^h e^{A\eta} d(\eta) B \tag{6.171}$$

Debido a que A es invertible, se tiene que:

$$B_d = \int_0^h e^{A\eta} d(\eta) B = A^{-1}(e^{Ah} - I)B \tag{6.172}$$

De esta forma, del análisis realizado para obtener las matrices A_d y B_d en (6.170) y (6.172) respectivamente, para la representación en estado planteada en (6.162) se obtiene que

$$A_d = \begin{bmatrix} e^{-ah} & 0 \\ \frac{1-e^{-ah}}{a} & 1 \end{bmatrix} \quad ; \quad B_d = \begin{bmatrix} \frac{b}{a} \cdot \left(1 - e^{-ah}\right) \\ \frac{b}{a} \cdot \left(\frac{e^{-ah}}{a} - \frac{1}{a} + h\right) \end{bmatrix} \tag{6.173}$$

Utilizando las constantes a y b obtenidas en el capítulo anterior, se obtienen las siguientes matrices A_d y B_d con distintos tiempos de muestreo h.

- $h = 100[ms]$

$$A_d = \begin{bmatrix} 0.0558 & 0 \\ 0.0327 & 1 \end{bmatrix}_{h=100[ms]} \quad ; \quad B_d = \begin{bmatrix} 1.4725 \\ 0.1049 \end{bmatrix}_{h=100[ms]} \tag{6.174}$$

- $h = 10[ms]$

$$A_d = \begin{bmatrix} 0.7493 & 0 \\ 0.0087 & 1 \end{bmatrix}_{h=10[ms]} \quad ; \quad B_d = \begin{bmatrix} 0.3909 \\ 0.002 \end{bmatrix}_{h=10[ms]} \tag{6.175}$$

- $h = 1[ms]$

$$A_d = \begin{bmatrix} 0.9716 & 0 \\ 0.001 & 1 \end{bmatrix}_{h=1[ms]} \quad ; \quad B_d = \begin{bmatrix} 0.0444 \\ 0 \end{bmatrix}_{h=1[ms]} \tag{6.176}$$

7. Observadores

7.1. Definición de Observadores

Para realizar una técnica de control por realimentación de estados se asume que se posee información sobre los estados del sistema. Se sabe que un sistema en representación de estados en tiempo discreto se representa como:

$$x_{k+1} = A_d x_k + B_d u_k \tag{7.177}$$

$$y_k = C_d x_k \tag{7.178}$$

siendo las matrices A_d, B_d, D_d conocidas (se omite D_d).

Al realizar implementaciones, las señales u_k y y_k suelen ser conocida al utilizar sensores en el sistema de control, pero los estados x_k suelen no ser accesible directamente o que su medida no sea económicamente viable. Una alternativa para los casos en que estos se desconozcan, es obtener una estimación de estos estados a través de un observador de estados.

Un observador de estados es un sistema dinámico cuyos estados convergen a los del sistema observado. Dependiendo del número de estados observados, el observador es de orden completo o reducido. [21]

7.2. Observador para sistema básico

El observador de orden reducido, también llamado como observador de Luenberger, considera la aproximación de estado como $\hat{x}_k$ del modelo:

$$\hat{x}_{k+1} = A_d \hat{x}_k + B_d u_k \tag{7.179}$$

Esta reconstrucción de estado, se puede realizar usando las mediciones de las salidas. Esto se puede realizar introduciendo la realimentación de la diferencia entre, las mediciones y las salidas estimadas, $y_k - C_d \hat{x}_k$, teniendo

$$\hat{x}_{k+1} = A_d \hat{x}_k + B_d u_k + L(y_k - C_d \hat{x}_k) \tag{7.180}$$

donde L es una matriz de ganancia.

Este término de realimentación $L(y_k - C_d\hat{x}_k)$ no contribuye en la expresión (7.180) si la salida medida es igual a la estimación de la salida generada por la estimación del estado. Además, se define el error los estados reales y la estimación del estado como:

$$\tilde{x}_k = x_k - \hat{x}_k \tag{7.181}$$

Reescribiendo esta expresión del error a partir de (7.180) se tiene:

$$\tilde{x}_{k+1} = A_d\tilde{x}_k - L(y_k - C_d\hat{x}_k) = (A_d - LC_d)\tilde{x}_k \tag{7.182}$$

Debido a que la matriz A y C son constantes dadas por el sistema, se debe escoger la matriz L de modo que los autovalores del observador dado por $(A_d - LC_d)$ se comporten asinstoticamente estable, esto forzará a que el error $\tilde{x}_k$ converja a cero; provocando que $\hat{x}_k \approx x_k$ logrando reconstruir los estados del sistema con la estimación $\hat{x}_k$.

De este modo, se utiliza el bloque 'observador' que posee la ecuación (7.180) que permite reconstruir los estados del sistema a partir de las entradas y salidas del sistema como se puede apreciar en la figura 7.19.

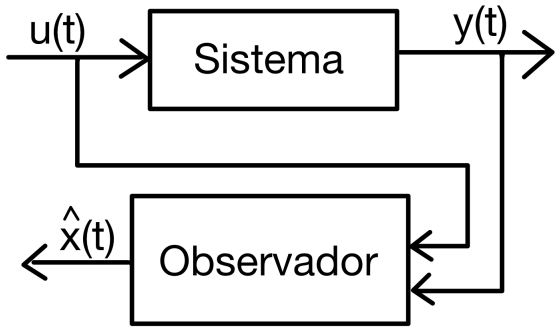


Figura 7.19: Diagrama de Observador

Para facilitar el calculo de L se utiliza el comando *acker(A',C',p)* en matlab. Este comando posee como entrada la matriz A y C traspuestos que están dadas por el sistema, y p corresponde a una matriz fila que contiene la ubicación de los polos deseados. Este nos entrega la matriz

L' definida anteriormente.

7.3. Observador extendido

A partir del observador definido anteriormente, es posible reconstruirlos estados del sistema $\hat{x}_k$, pero esto no es muy útil en la practica, debido a que pueden existir perturbaciones en las entradas y/o salidas del sistema, las cuales pueden afectar la estimación. Generalizando este problema, un modelo con perturbaciones esta descrito como

$$x_{k+1} = A_d x_k + B_d u_k + B_d d_k \tag{7.183}$$

$$y_k = C_d x_k + e_k \tag{7.184}$$

en donde d_k es una perturbación en la entrada y e_k es una perturbación en la salida.

La planta a controlar en esta memoria, posee una zona muerta la cual genera una señal de salida distinta a cero a partir de señales de entradas superiores a 0.3[V]. Para estimar esta zona muerta, se reconstruirá como una perturbación de entrada d_k en el sistema.

La perturbación d_k al ser constante en el tiempo cumple que:

$$d_{k+1} = d_k \tag{7.185}$$

Aumentando el grado del sistema considerando esta perturbación, se define el nuevo vector de estado como

$$z_k = \begin{bmatrix} x_k \\ d_k \end{bmatrix} \tag{7.186}$$

Por lo tanto, el sistema aumentado esta dado por

$$\begin{bmatrix} x_{k+1} \\ d_{k+1} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & B_d \\ 0 & 1 \end{bmatrix}}_{A_z} \begin{bmatrix} x_k \\ d_k \end{bmatrix} + \underbrace{\begin{bmatrix} B_d \\ 0 \end{bmatrix}}_{B_z} u_k \tag{7.187}$$

$$y_k = \underbrace{\begin{bmatrix} C_d & 0 \end{bmatrix}}_{C_z} \begin{bmatrix} x_k \\ d_k \end{bmatrix} \tag{7.188}$$

Realizando los mismos pasos vistos en la sección anterior para el observador básico, se tiene el siguiente sistema de observador para el sistema aumentado

$$\hat{z}_{k+1} = A_z \hat{z}_k + B_z u_k + L_z (y_k - C_z \hat{z}_k) \tag{7.189}$$

en donde los autovalores del observador esta dado por $(A_z - L_z C_z)$.

Nótese, que al implementar un observador aumentado se tiene la estimación de los estados x_k y la perturbación d_k, esta nueva información del sistema nos permite replantear el sistema de control descrito en capítulos anteriores para mejorar su rendimiento.

Habiendo definido como obtener la ganancia del observador para un sistema aumentado, esta estimación se puede representar de la forma de variables estado. Reagrupando la ecuación (7.189) se tiene:

$$\bar{x}_{k+1} = A_{obs} \bar{x}_k + B_{obs} \bar{u} \tag{7.190}$$

$$\bar{y}_k = C_{obs} \bar{x} \tag{7.191}$$

con:

$$A_{obs} = A_z - L_z C_z \quad ; \quad B_{obs} = \begin{bmatrix} \begin{bmatrix} B_z & 0 \end{bmatrix}^T & L \end{bmatrix} \quad ; \quad C_{obs} = I \tag{7.192}$$

Las entradas serian las mediciones $\bar{u} = \begin{bmatrix} u_k & y_k \end{bmatrix}^T$ y las salidas corresponden a las estimaciones $\bar{y} = \begin{bmatrix} \hat{x}_{1_k} & \hat{x}_{2_k} & \hat{d}_k \end{bmatrix}^T$.

7.4. Seguimiento de referencia considerando una perturbación estimada

Al utilizar un observador extendido, se tiene la siguiente representación de estados del sistema:

$$x_{k+1} = A_d x_k + B_d u_k + B_d \hat{d}_k \tag{7.193}$$

$$y_k = C_d x_k \tag{7.194}$$

Repitiendo los pasos descritos para obtener un seguimiento de referencia al realizar MPC, se tiene en estado estacionario se tiene que $y_\infty \approx r$ y se considera la perturbación constante como $\hat{d}_k = \hat{d}$. De esta forma, se tiene que

$$x_\infty = A_d x_\infty + B_d u_\infty + B_d \hat{d} \tag{7.195}$$

$$y_\infty = C_d x_\infty \tag{7.196}$$

Despejando x_∞ y u_∞, se obtiene:

$$\underbrace{\begin{bmatrix} A_d - I & B_d \\ C_d & 0 \end{bmatrix}}_{I} \begin{bmatrix} x_\infty \\ u_\infty \end{bmatrix} = \begin{bmatrix} -B_d \hat{d} \\ r \end{bmatrix} = \begin{bmatrix} -B_d & 0_{2\times 1} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{d} \\ r \end{bmatrix} \tag{7.197}$$

$$\begin{bmatrix} x_\infty \\ u_\infty \end{bmatrix} = \underbrace{I^{-1} \begin{bmatrix} -B_d & 0_{2\times 1} \\ 0 & 1 \end{bmatrix}}_{T} \begin{bmatrix} \hat{d} \\ r \end{bmatrix} \tag{7.198}$$

Desarrollando la matriz T utilizando herramientas computacionales, se obtiene el siguiente resultado:

$$\begin{bmatrix} x_\infty \\ u_\infty \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \hat{d} \\ r \end{bmatrix} \tag{7.199}$$

Finalmente, recordando que el sistema planteado a controlar posee 2 estados y 1 entrada. Para tener un seguimiento de referencia considerando una perturbación de entrada se tiene:

$$\begin{bmatrix} x_{1_\infty} \\ x_{2_\infty} \\ u_\infty \end{bmatrix} = \begin{bmatrix} 0 \\ r \\ -\hat{d} \end{bmatrix} \tag{7.200}$$

Cabe destacar, que al implementar un seguimiento de referencia considerando la zona muerta del motor como una perturbación y realizar una técnica de control en el sistema, se produce una acción integral en este. Se recuerda que la entrada óptima a implementar en el sistema es:

$$\vec{u}^* = \vec{\tilde{u}}^* + \vec{u}_\infty \tag{7.201}$$

donde $\vec{\tilde{u}}^*$ es la entrada generada por el controlador que permite seguir una referencia de forma óptima y $\vec{u}_\infty$ es la entrada definida por el seguimiento de referencia el cual corrige el error en estado estacionario.

De este forma, se concluye que en esta memoria la utilización de MPC con integración como técnica de control a emplear en el Motor DC.

8. Implementación de Método de Control

En este capítulo se describirá el proceso de implementación de Control Predictivo, el cual se realizará a través del software matemático Matlab; específicamente en el entorno de programación visual Simulink. Es por ello que se comenzará por explicar en qué consiste este software, qué herramientas ofrece y cómo interactuar con esta.

Luego se explicará la Toolbox Quarc, el cual corresponde a una extensión en simulink realizada por la empresa Quanser. Este Toolbox nos permitirá enviar y recibir información con la planta real.

Como parte extra de esta memoria, también se realizará un control MPC utilizando el MicroLabBox de la empresa dSpace. Al igual que con la toolbox quarc, dSpace dispone de software capaz de realizar una conexión entre Simulink y la planta real.

De esta forma, se plantearán dos métodos para implementar la técnica de control descrita en capítulos anteriores.

8.1. Matlab/Simulink

Matlab es un software matemático que permite el análisis de datos, diseño de modelos, como también el desarrollo de algoritmos. Todo esto bajo un entorno de desarrollo integrado que cuenta con su propio lenguaje de programación, el que permite expresar las operaciones matemáticas de matrices y de arreglos de manera directa.

Este software es utilizado por un gran número de investigadores y académicos, debido a que sus herramientas permiten implementar una gran variedad de sistemas, como por ejemplo, comunicaciones inalámbricas, robótica, sistemas de control, procesamiento de señales, análisis de datos, entre otras.

En la Figura 8.20 se pueden apreciar algunas secciones del software:

- **Menú principal:** Se encuentra en todo el borde superior. Desde esta se pueden abrir archivos, editores de texto, entre otras.

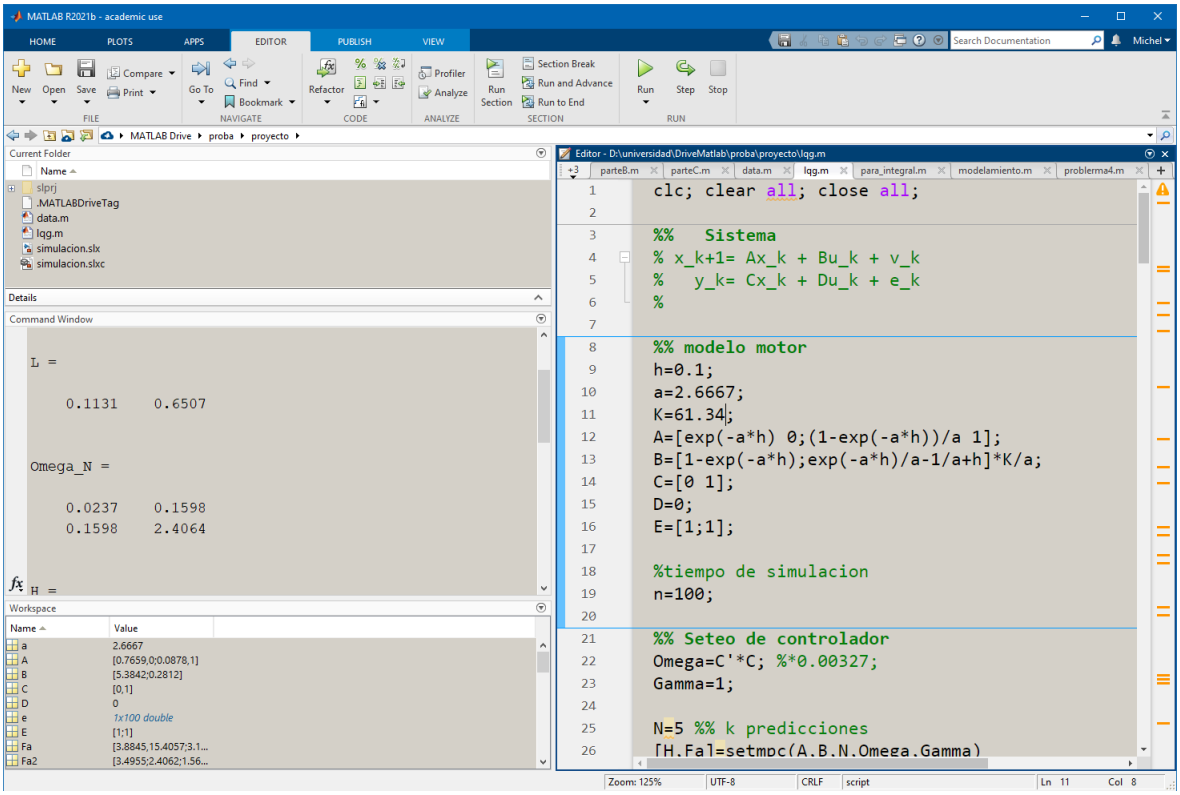


Figura 8.20: Esquema de la plataforma Matlab

- **Ubicación en carpeta:** Se encuentra en la esquina superior izquierda, bajo el menú. Permite identificar la carpeta donde se encuentra guardado el archivo.
- **Workspace:** Se encuentra en la esquina inferior izquierda. Esta ventana muestra las variables que se han compilado.
- **Ventana de comandos:** Se encuentra en la parte media de la izquierda. En esta ventana se escriben los comandos para trabajar directamente.
- **Script:** Se encuentra en el lado derecho. Es similar a un bloc de notas, para no escribir directamente en la ventana de comandos.

Cabe destacar que la ubicación de estas ventanas se puede personalizar.

Simulink es un entorno de programación visual que funciona conjuntamente con Matlab. La programación de los sistema se realiza a través de la unión de bloques disponibles en la biblio-

teca de este, otorgándole a los usuarios un entorno gráfico que permite comprender de mejor manera el funcionamiento de los sistemas de interés.

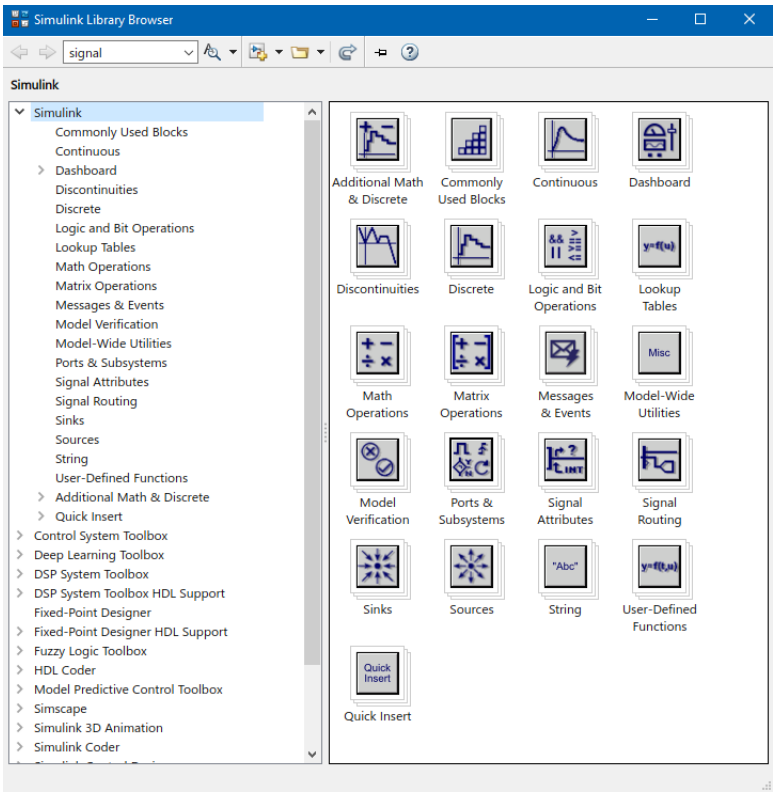


Figura 8.21: Biblioteca de Simulink

Al realizar un esquema, en el menú de Simulink se puede configurar los parámetros con los que el solver trabajara, por ejemplo, trabajar con Euler con un tiempo de sampleo de 1 segundo.

8.1.1. Bloques útiles para implementar MPC

A continuación, se describirán algunos bloques útiles para realizar la implementación de la técnica de control.

- **Matlab Function:** Este bloque permite crear una función de Matlab, de el mismo entorno de programación que este posee.

  Este bloque se utilizará para implementar el algoritmo de ADMM para MPC.

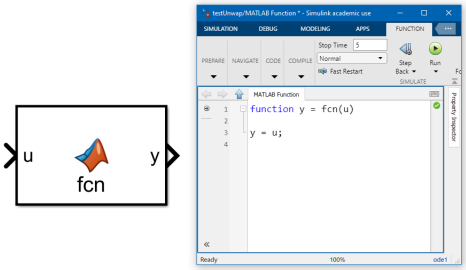


Figura 8.22: Bloque Simulink: Matlab Function, y su entorno de programación.

- **Discrete State-Space:** Este bloque permite implementar un sistema descrito por su representación de estado en tiempo discreto.

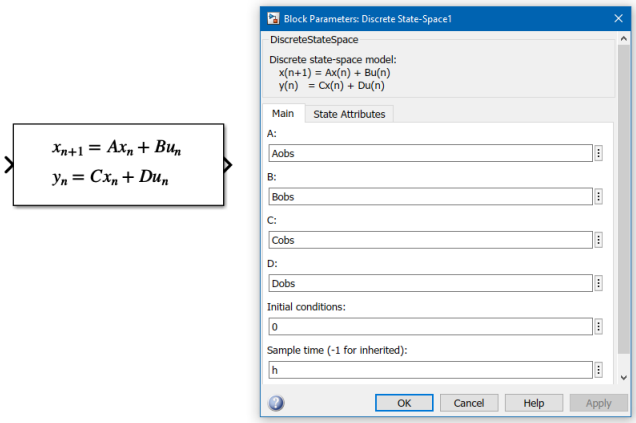


Figura 8.23: Bloque Simulink: Discrete State-Space.

Este bloque se utilizará para implementar el observador de estados del sistema. Este bloque tendrá de entrada $u = [V_m(k)\ \theta(k)]^T$ y salida $y = [x_1(k)\ x_2(k)\ \hat{d}(k)]^T$.

- **Demux y Mux:** Estos bloques permiten separar o juntar datos de un vector respectivamente.

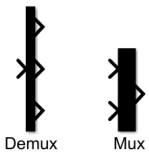


Figura 8.24: Bloque Simulink: Demux y mux

Estos se utilizarán para empaquetar y separar datos del observador.

- **Otros bloques:** Estos corresponden a bloques comunes que permiten manipular u observar las señales.

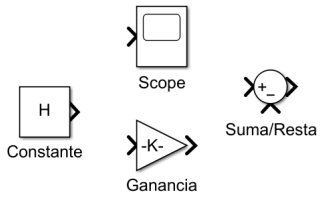


Figura 8.25: Bloque Simulink: Bloques comunes

8.2. Quanser

Quanser es una empresa canadiense reconocida a nivel mundial por sus equipos de laboratorio de ingeniería especializados para la enseñanza y la investigación, centrado mayormente en control, robótica y mecatrónica. De esa manera proporciona un marco moderno para la educación en ingeniería a través de una base física de la teoría matemática, manifestada de una manera que permita a los estudiantes ver y sentir como las matemáticas fluyen de la teoría al mundo físico.

Entre los productos que proporciona, podemos encontrar el software Quarc, el cual permite diseñar, implementar y validar aplicaciones en tiempo real en hardware utilizando Simulink. Tambien podemos encontrar hardware que permite realizar las conexiones entre simulink y la planta real. Cabe destacar que el Servo motor SRV02 también pertenece a los productos de Quanser.

En las siguientes secciones se describirá el hardware y software de Quanser utilizado para implementar MPC.

8.2.1. Hardware Quanser

El hardware corresponde a los elementos físicos o materiales que constituyen a una computadora o un sistema informático. A continuación, se describirán los componentes utilizados.

- **Q8-USB Data Acquisition [22]:** Corresponde a un dispositivo de adquisición de datos de ocho canales que ofrece un rendimiento fiable en el tiempo a través de una

interfaz USB. Este permite conectar y controlar una variedad de dispositivos equipados con sensores analógicos y digitales, incluidos codificadores.

Figura 8.26: Hardware Quanser: Q8-USB Data Acquisition Device

Este dispositivo se encuentra conectado por USB al computador para enviar y recibir datos a partir de este puerto. Además, este dispositivo recibe datos del encoder del servo por los puertos 'Encoders' y envía la señal de control por un puerto análogo a VoltPAQ-X2 Amplifier.

- **VoltPAQ-X2 Amplifier [23]:** Este corresponde a un amplificador lineal de voltaje. Este es ideal para todas las configuraciones de control complejo relacionado con necesidades educativas o de investigación. Este dispositivo recibe la señal de control de

Figura 8.27: Hardware Quanser: VoltPAQ-X2 Amplifier

'Q8-USB Data Acquisition Device', la amplifica y envía al servomotor.

- **Rotatory Servo Base Unit [24]:** Este corresponde a la planta a controlador descrita en capítulos anteriores.

Figura 8.28: Hardware Quanser: Servo SRV02

Este recibe la señal de control amplificada y realiza la dinámica correspondiente de un servo motor. Además, este posee un encoder integrado el cual está conectado a ' Q8-USB Data Acquisition Device', de esta forma es posible medir la velocidad del motor en el computador.

8.2.2. Software Quanser

Este software 'QUARC Real-Time Control Software' genera un código en tiempo real directamente desde los controladores diseñador por Simulink y lo ejecuta en tiempo real en el objetivo de Windows, todo sin procesamiento de señal digital o sin escribir una sola línea de código.

Para usar las herramientas de QUARC en Simulink, se debe primero importar sus librerías, en este caso, bastará con una linea de comando en Matlab, el cual ejecutara el setup e importará todo lo necesario a Simulink. Para acceder a las herramientas, se debe entrar a la biblioteca de Simulink y buscar la sección de Quarc. Se utilizaron los siguientes bloques:

- **HIL Initialize:** Iniciliza una placa HIL y asocia un nombre con esta placa.

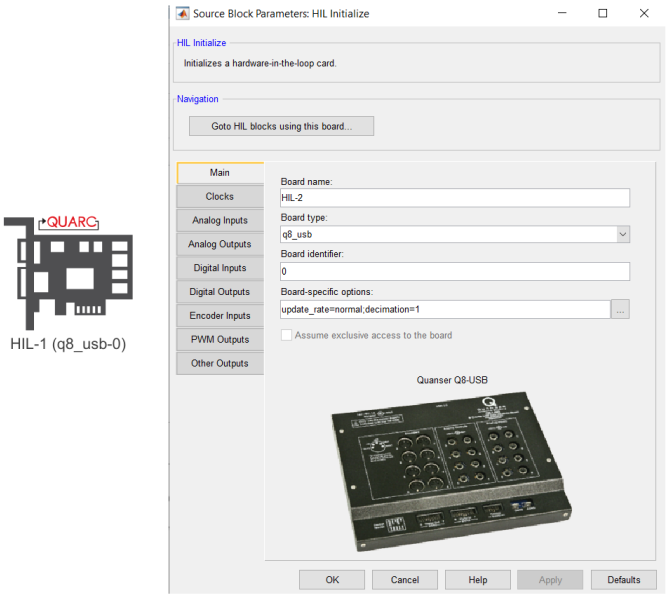


Figura 8.29: Software Quanser: HIL Initialize

Este bloque se utiliza para inicializar 'Q8-USB Data Acquisition Device'.

- **HIL Write Analog:** Este bloque permite enviar una señal análoga inmediatamente en el canal especificado.

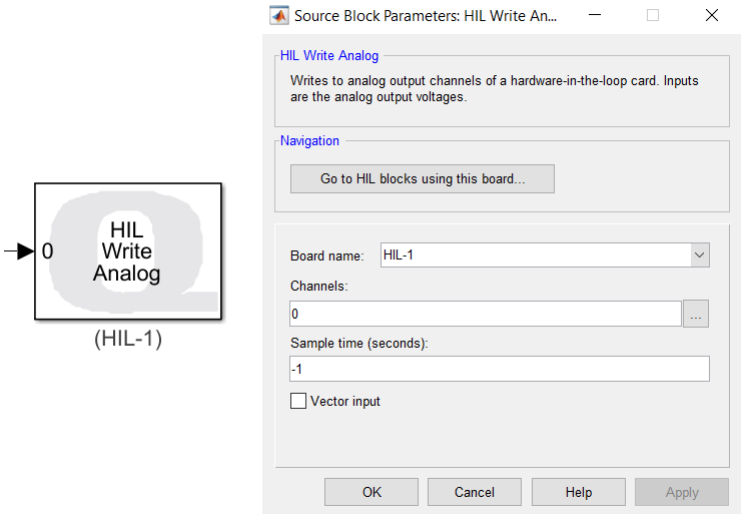


Figura 8.30: Software Quanser: HIL Write Analog

Se utilizó el canal 0 para enviar la señal de control.

- **HIL Read Encoder:** Este bloque lee los canales del encoder especificado.

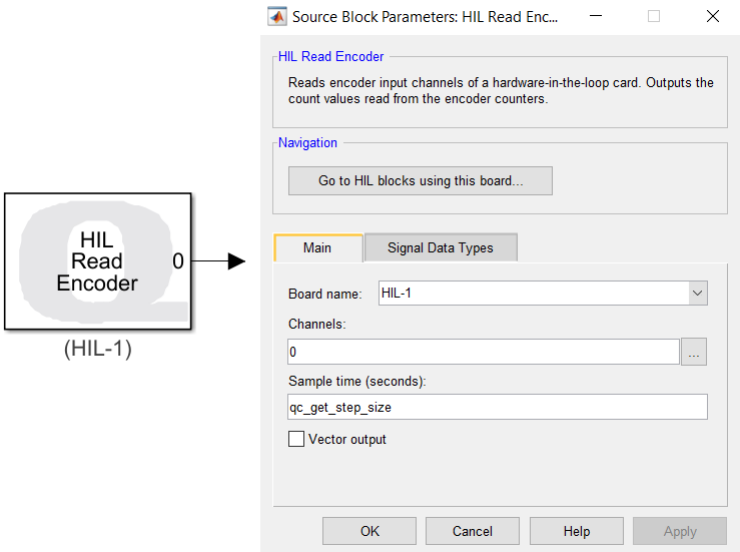


Figura 8.31: Software Quanser: HIL Read Encoder

Se utilizó el canal 0 para recibir la información del encoder del servo motor. Este bloque entrega la medición de la posición angular del motor en radianes.

- **Inverse Modulus:** Este bloque calcula la salida acumulada de una señal con envolvimiento.

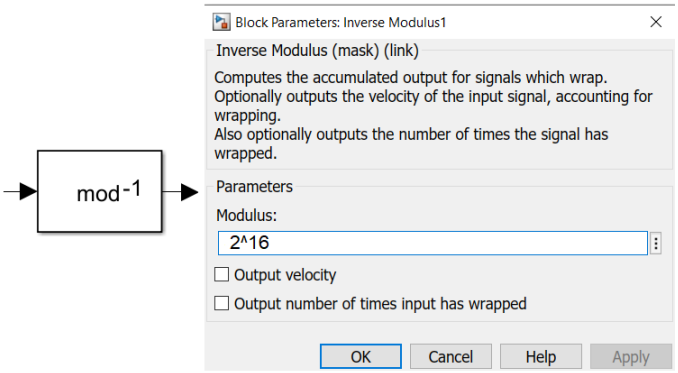


Figura 8.32: Software Quanser: Inverse Modulus

Los encoder suelen tener un máximo de cuentas para medir, al superar este máximo,

se reinicia la cuenta desde 0. Al utilizar este bloque, se evita este 'overflow' en las mediciones del encoder.

- **To host file:** Este bloque permite guardar los datos en archivos .mat.

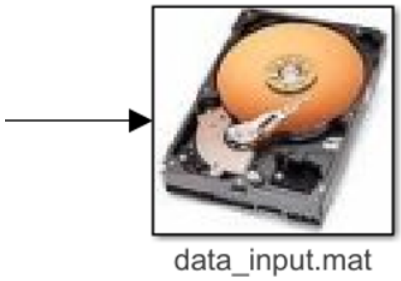


Figura 8.33: Software Quanser: To host file

8.2.3. Acotaciones al usar Quarc en Simulink

A continuación, se realizarán ejemplos de cómo se utilizan estos bloques de Quarc.

En la Figura 8.34 se presenta un ejemplo de conexiones para enviar y tomar datos en el servo motor.

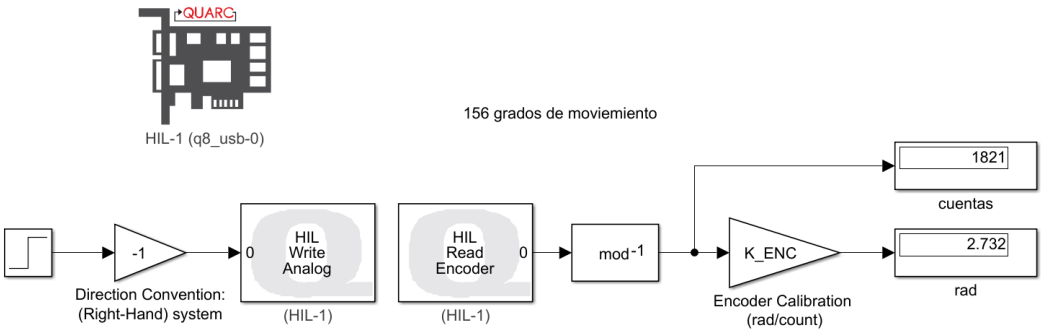


Figura 8.34: Ejemplo de envío y recepción de señales en Servo utilizando Quarc

De este ejemplo se extrae la siguiente información:

- Se inicializa 'Q8-USB Data Acquisition Device' utilizando el bloque 'HIL-1'. Este toma de nombre 'q8_usb_0' al setear esta interfaz USB.

- Utilizando un bloque escalón, se envía una señal de $1[V]$ hacia el motor utilizando el bloque 'HIL Write Analog'. Cabe destacar que las señales de control se deben multiplicar por -1, debido a transformaciones internas realizadas por el hardware 'Q8-USB Data Acquisition Device' y 'VoltPAQ-X2 Amplifier'.

- Debido a que se envía una señal de control al servo, se comienza a realizar un movimiento. Utilizando el bloque 'HIL Read Encoder' se toma la medición de las cuentas realizas por el encoder.

8.2.3.1. Encoder El encoder utilizado es del tipo óptico incremental, este corresponde a un sensor de desplazamiento angular relativo que mide desplazamiento relativo a una posición previamente conocida. A diferencia de un codificador absoluto, un codificador incremental no conservar su información de posición en caso de pérdida de energía. Un codificador incremental emite una serie de pulsos que se correlacionan al cambio relativo en la posición angular.

Los encoders de cuadratura permiten medir la posición y dirección de rotación a partir de la información de dos canales –comúnmente llamados Canal A y Canal B. El Canal A se encuentra desfasado 90° con respecto al Canal B (ver Figura 8.35). Para determinar la dirección de rotación se monitorea la transición de un canal con respecto al otro canal. Esta información debe ser decodificada en un circuito de acondicionamiento normalmente llamado decodificador de cuadratura. El circuito decodificador de cuadratura se basa en un contador de alta velocidad y resolución que incrementa o disminuye su valor de acuerdo con la diferencia de fase entre los canales A y B, la cual depende de la dirección de rotación del eje del motor [25].

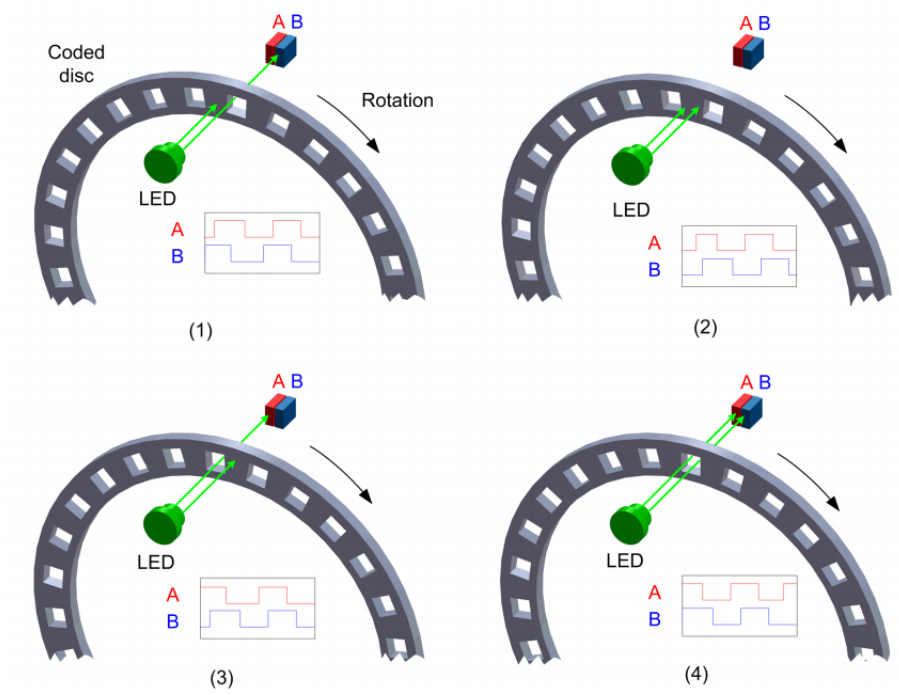


Figura 8.35: Salida de un encoder incremental cuando existe una rotación [3]

El encoder utilizado posee una resolución en cuadratura de 4096[cuentas/rev], es decir, por cada vuelta que realice el servo motor, el bloque entregará un valor de 4096[cuentas].

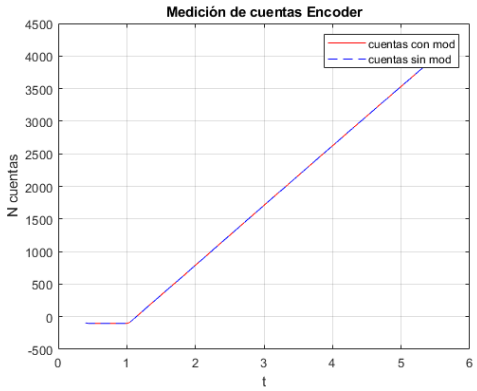
- Se utiliza el bloque 'Inverse Modulus' o mod^{-1}, para evitar una perdida en la medición en caso de que los bits del dispositivo de adquisición de datos se llene y ocurra un overflow (el DAQ utilizado posee una resolución de 16 bits, 65536 en cuentas).

- Se utiliza un bloque de ganancia K_{ENC} para transformar las cuentas en desplazamiento en angular (en radianes). Esta constante se obtiene de:

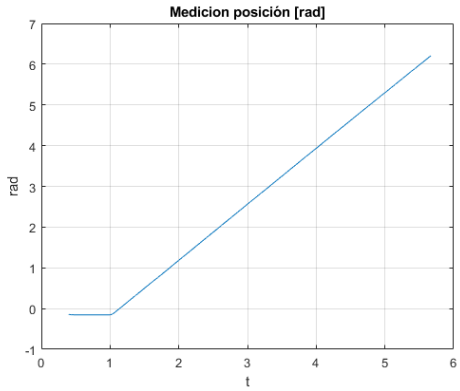
$$K_{\text{ENC}} = \frac{2\pi}{N \cdot PPR} = \frac{2\pi}{4 \cdot 1024} = \frac{2\pi}{4096} = 0.0015 \frac{[\text{rad}]}{[\text{cuentas}]} \tag{8.202}$$

donde N corresponde a los algoritmos utilizados para decodificar los datos, al usar cuadratura este es $N = 4$. PPR se refiere al número de incrementos existentes en un disco de encoder incremental, el utilizado posee 1024.

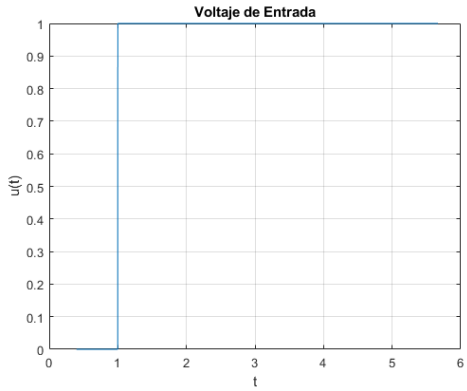
Los datos obtenidos en el ejemplo de la Figura 8.34 son los siguientes:



(a) Gráfico: Medición de cuentas de encoder



(b) Gráfico: Medición de posición angular de encoder (rad)



(c) Gráfico: Señal de entrada utilizada

Figura 8.36: Gráficos obtenidos para primer ejemplo de uso de Quarc

Otro ejemplo interesante de comentar es sobre el uso del bloque 'Inverse Modulus'. Aparentemente en el ejemplo anterior este bloque está demás, debido a que la medición sin utilizar este bloque es igual.

A continuación, se presentan los siguientes ejemplos sobre el uso de este bloque, para estos tres casos se ocupó la misma señal de entrada, la cual corresponde a una señal diente de cierra con pendiente 16 y periodo 1[s] que simula la medición de un encoder con resolución de 4 bits.

- **Caso Modulus 16:** Este corresponde a la Figura 8.37 y Figura 8.38. En el gráfico obtenido, se puede apreciar que la salida obtenida al utilizar mod sigue contando correctamente la medición del encoder cuando se reinicia la cuenta de este.

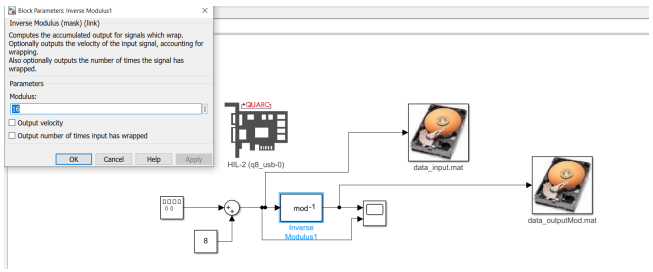


Figura 8.37: Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 16.

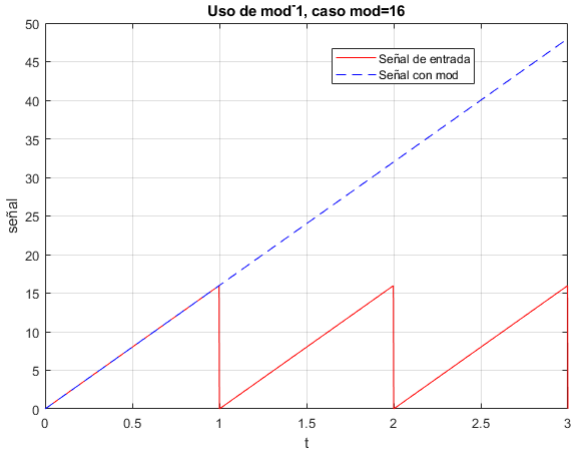


Figura 8.38: Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 16.

- **Caso Modulus 20:** Este corresponde a la Figura 8.39 y Figura 8.40. En el gráfico obtenido, se puede apreciar que la salida obtenida al utilizar mod no sigue correctamente las cuentas del encoder. Al momento del reinicio de la cuenta del encoder, la medición del bloque mod salta hasta el múltiplo de 20 más cercano hacia arriba.

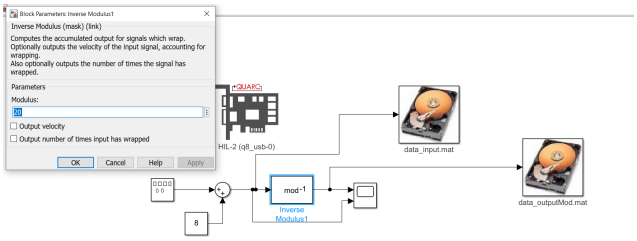


Figura 8.39: Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 20.

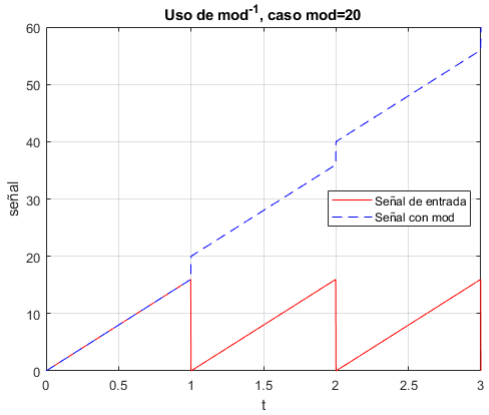


Figura 8.40: Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 20.

- **Caso Modulus 12:** Este corresponde a la Figura 8.41 y Figura 8.42. En el gráfico obtenido, se puede apreciar que la salida obtenida al utilizar mod no sigue correctamente las cuentas del encoder. Al momento del reinicio de la cuenta del encoder, la medición del bloque mod salta hasta el múltiplo de 12 más cercano hacia abajo.

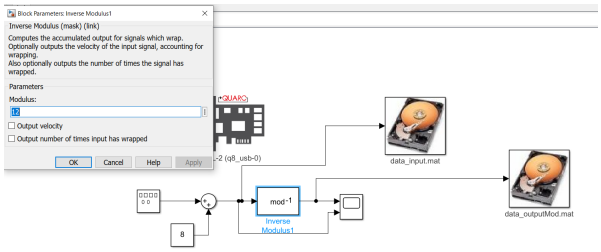


Figura 8.41: Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 12.

Figura 8.42: Ejemplo 2: uso de bloque 'Inverse Modulus' con modulus 12.

8.2.4. Implementación de MPC en QUARC

Recapitulando capítulos anteriores, se describió como formular MPC con acción integral, en donde se detalló como utilizar ADMM para resolver el problema de optimización QP. Además, se modeló la planta a controlar, la cual corresponde a un motor DC llamado 'srv02'. Este modelo se utiliza en MPC para plantear las matrices que predicen N instantes futuros. También, este modelo se utiliza para estimar los estados y una perturbación del sistema a través del observador definido.

Para implementar MPC, en general este se divide en dos partes. La primera corresponde a una etapa de cálculo previo de las matrices del sistema y el problema QP, esto se puede realizar debido a que son constantes en el tiempo. Estas son:

- Se define h de muestreo, a partir de este parámetro se calcula las matrices del sistema A_d, B_d y C_d.
- Se define las matrices de peso del problema de optimización. Se usó $Q = C_d^T \cdot C_d \cdot 0.4$ y $R = 1$
- Se calcula las matrices H, h, $\hat{M}$, $\hat{c}$ del problema QP.
- Se calcula la ganancia L_z del observador. A partir de esta constante, se construyen las matrices A_{obs}, B_{obs}, C_{obs} y D_{obs} para implementar el observador en un bloque de representación de estados.

La segunda parte corresponde a la implementación en tiempo real. Para esto, se construye el diagrama en simulink. Este se puede apreciar en la Figura 8.43.

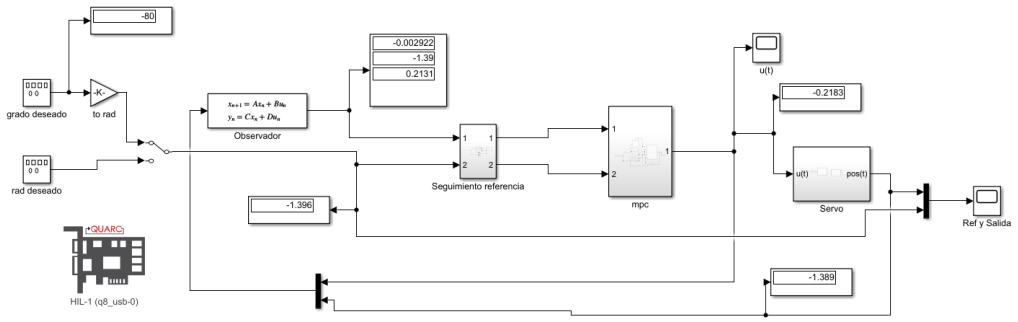


Figura 8.43: Implementación de MPC en simulink con Quarc

El bloque 'Seguimiento referencia' corresponde al subsistema detallado en la Figura 8.44. La entrada 1 corresponde a las estimaciones del observador y la entrada 2 corresponde a la referencia. La salida 1 corresponde a las estimaciones de los estados y la salida 2 corresponde a la estimación de la perturbación.

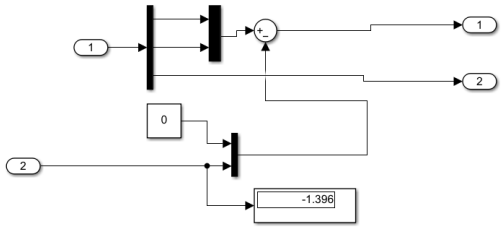


Figura 8.44: Implementación de MPC en simulink con Quarc. Subsistema de seguimiento de referencia

El bloque 'mpc' corresponde al subsistema detallado en la Figura 8.45. Este bloque recibe las estimaciones y mediciones del sistema, plantea las matrices del problema QP y las ingresa a un bloque 'Matlab Function', el cual posee el algoritmo de ADMM. Este subsistema entrega la señal de control óptima.

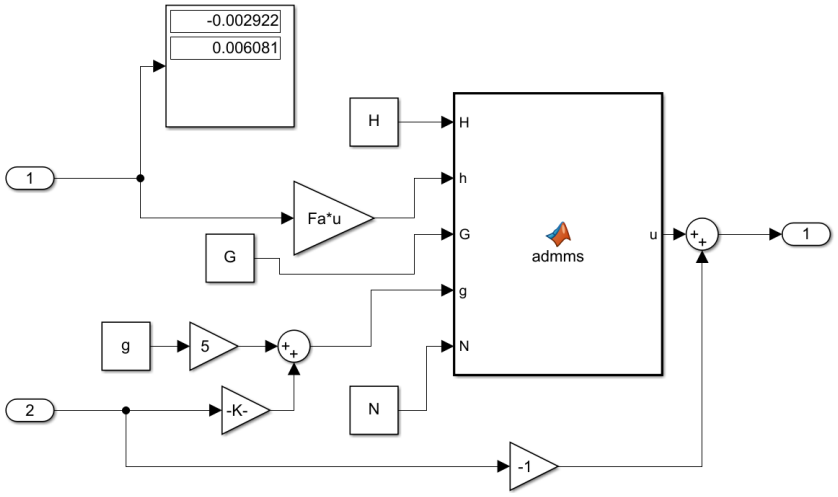


Figura 8.45: Implementación de MPC en simulink con Quarc. Subsistema de MPC

El bloque 'servo' corresponde al subsistema detallado en la Figura 8.46. Este bloque envía la señal de control óptimo al servo motor. Además, recibe las mediciones del encoder en radianes.

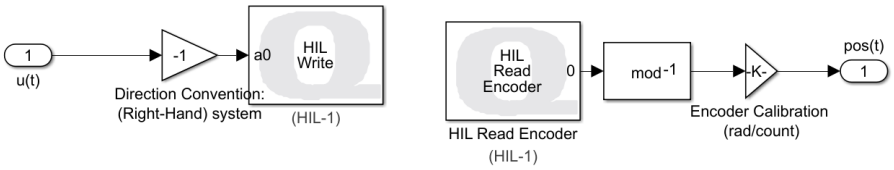


Figura 8.46: Implementación de MPC en simulink con Quarc. Subsistema de conexiones con Servo motor real

8.2.5. Resultados obtenidos en Quarc

A partir de la implementación detallada en la sección anterior, se realizó un control en la planta de interés. Se realizaron varias pruebas de su funcionamiento variando el tiempo de muestreo h y el horizonte de predicción N. A continuación, se presentaran los resultados gráficos de 2 casos.

- **h=0.1[s] y N=5:**

  La Figura 8.47, Figura 8.48, Figura 8.49 y Figura 8.50 corresponden a los datos obtenidos respecto a las salidas, entrada, estados y tiempo de resolución de MPC respectivamente.

  Para este caso, se comprueba que la técnica de control utilizada sigue la referencia de forma aceptable respetando las restricciones designadas. Se aprecia que intenta corregir el error, pero no lo hace de forma eficiente debido al tiempo de muestreo. Respecto a los tiempos de resolución del algoritmo de MPC, se obtuvo un tiempo máximo de 0.0054$[s]$ y una media de 0.0022$[s]$.

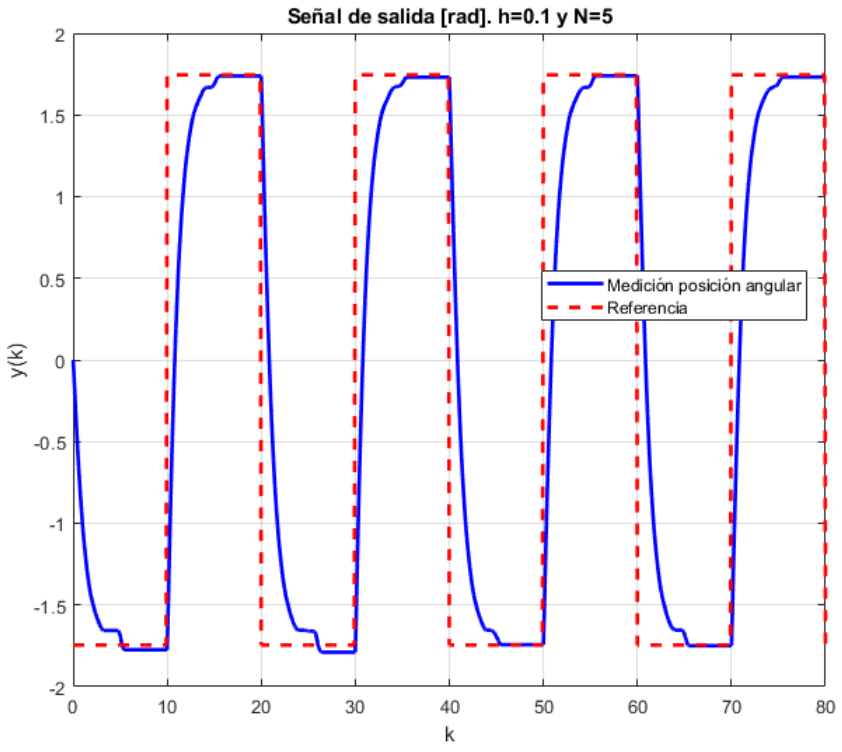


Figura 8.47: Gráfico de salida obtenida con h=0.1[s] y N=5 en Quarc.

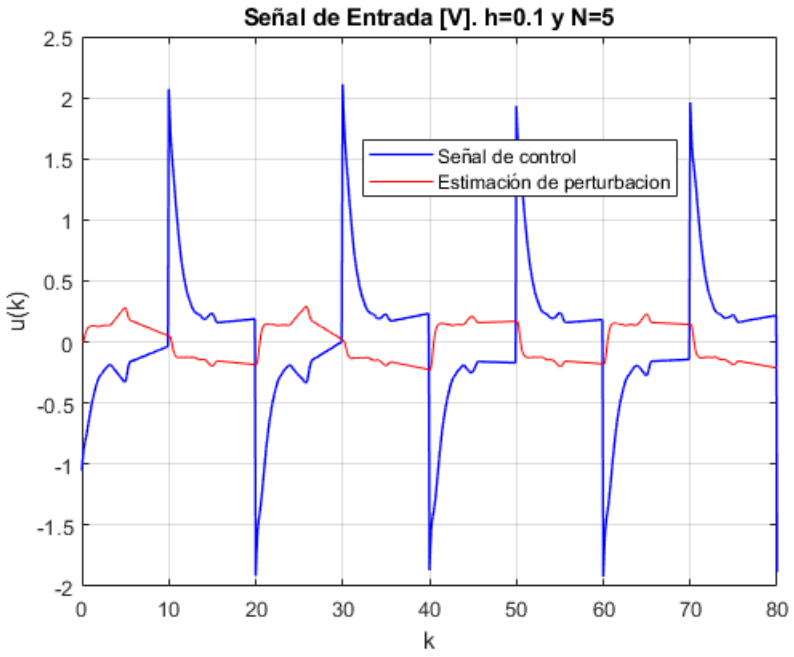


Figura 8.48: Gráfico de señal de entrada con h=0.1[s] y N=5 en Quarc.

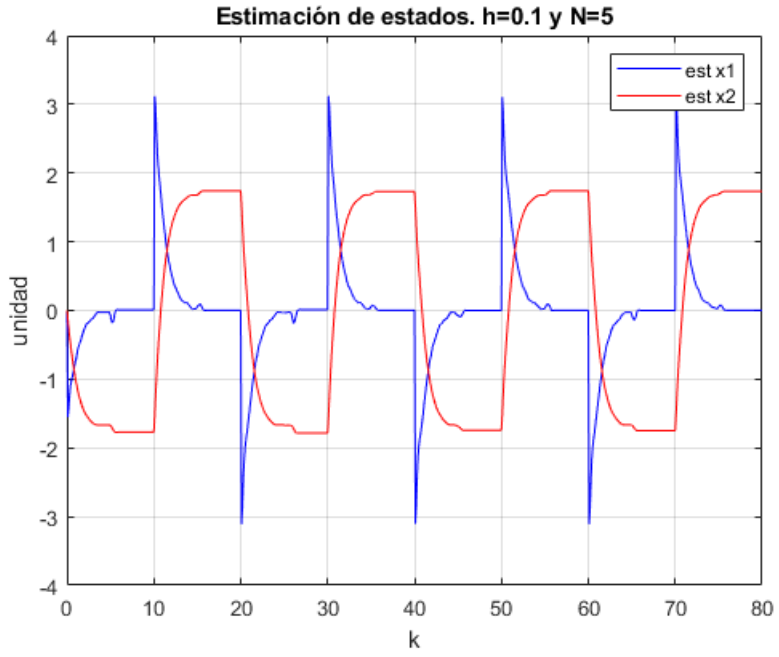


Figura 8.49: Gráfico de señal de entrada con h=0.1[s] y N=5 en Quarc.

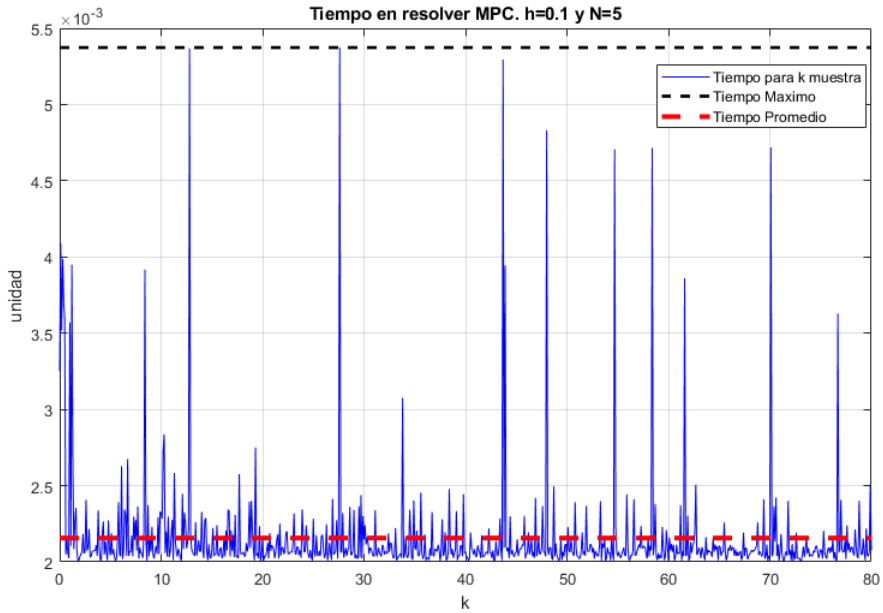


Figura 8.50: Gráfico de tiempos en resolver MPC con h=0.1[s] y N=5 en Quarc.

- **h=0.01[s] y N=5:**

  La Figura 8.51, Figura 8.52, Figura 8.53 y Figura 8.54 corresponden a los datos obtenidos respecto a las salidas, entrada, estados y tiempo de resolución de MPC respectivamente.

  Para este caso, se comprueba que la técnica de control utilizada sigue la referencia de forma correcta respetando las restricciones designadas. Respecto a los tiempos de resolución del algoritmo de MPC, se obtuvo un tiempo máximo de 0.0055[s] y una media de 0.0023[s].

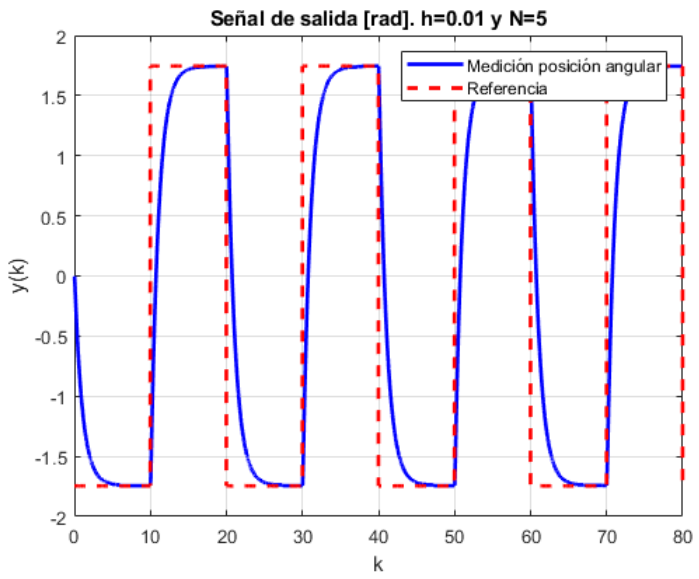


Figura 8.51: Gráfico de salida obtenida con h=0.01[s] y N=5 en Quarc.

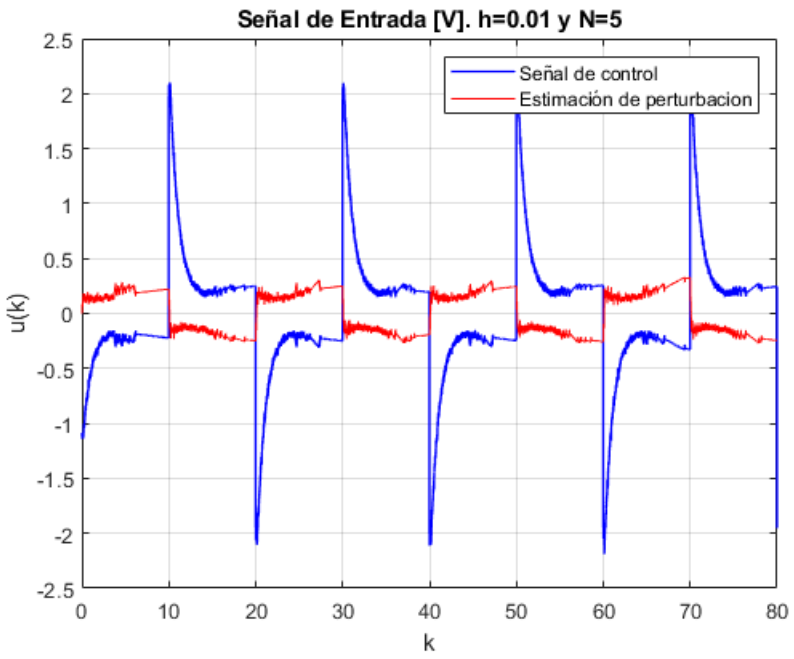


Figura 8.52: Gráfico de señal de entrada con h=0.01[s] y N=5 en Quarc.

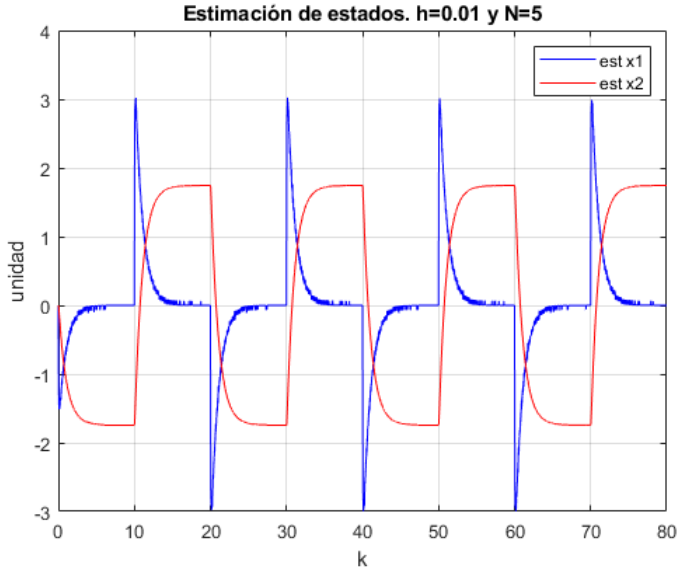


Figura 8.53: Gráfico de señal de entrada con h=0.01[s] y N=5 en Quarc.

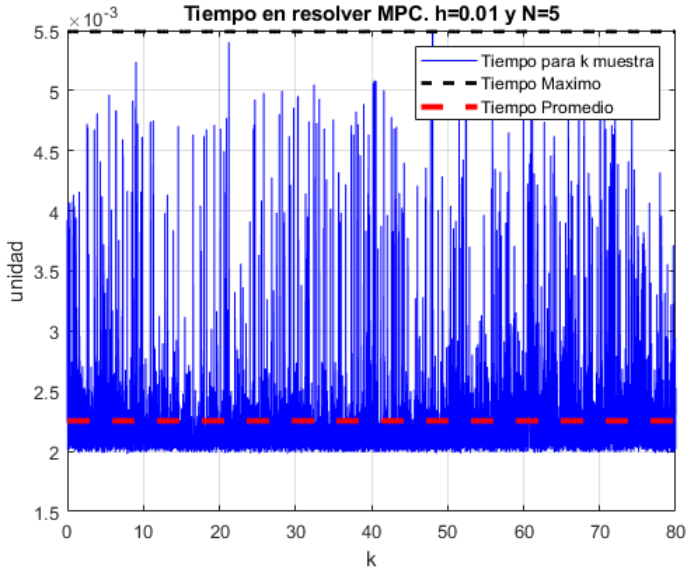


Figura 8.54: Gráfico de tiempos en resolver MPC con h=0.01[s] y N=5 en Quarc.

El resto de las pruebas realizadas, se compilaron en la siguiente tabla. En donde 1 representa un funcionamiento correcto de MPC y 0 representa un fallo en el funcionamiento en MPC para este caso.

h[ms]\N	2	3	4	5	6	10	15
100	*1*	*1*	*1*	*1*	*1*	*1*	*1*
10	*1*	*1*	*1*	*1*	*1*	*1*	*0*
5	*1*	*1*	*1*	*1*	*1*	*0*	*0*
3	*1*	*1*	*1*	*0*	*0*	*0*	*0*
2	*1*	*1*	*0*	*0*	*0*	*0*	*0*
1.5	*1*	*0*	*0*	*0*	*0*	*0*	*0*
1	*0*	*0*	*0*	*0*	*0*	*0*	*0*

Tabla 8.2: Resultados obtenidos sobre la implementación de MPC en Quarc. 1=Éxito, 0=Fallo

Luego de realizar estas pruebas, se intentó optimizar el funcionamiento de MPC en simulink. En ADMM hay algunos cálculos de matrices constantes que se realizan en cada iteración del algoritmo, para mejorar el rendimiento de MPC, se extrajo ese cálculo del bloque 'Matlab Function' de modo que estas matrices se calculen sólo una vez. Al realizar este cambio, se obtuvieron mejores resultados. Estos se pueden apreciar en la siguiente tabla:

h[ms]\N	2	3	4	5	6	10	15
100	*1*	*1*	*1*	*1*	*1*	*1*	*1*
10	*1*	*1*	*1*	*1*	*1*	*1*	*1*
5	*1*	*1*	*1*	*1*	*1*	*1*	*0*
3	*1*	*1*	*1*	*1*	*1*	*0*	*0*
2	*1*	*1*	*1*	*0*	*0*	*0*	*0*
1.5	*1*	*1*	*0*	*0*	*0*	*0*	*0*
1	*0*	*0*	*0*	*0*	*0*	*0*	*0*

Tabla 8.3: Resultados obtenidos sobre la implementación de MPC en Quarc (optimizado). 1=Éxito, 0=Fallo

Cabe destacar que para un $h \leq 1.5[ms]$, la parte mecánica del servo motor no reacciona correctamente a cambios tan bruscos debido al bajo tiempo de muestreo, debido a que el

controlador no funciona correctamente. Además, el desempeño del controlador en las tablas anteriores para un resultado no es igual, de modo que un 1 no representa la misma señal de control generada al comparar las tablas presentadas.

8.3. MicroLabBox/ dSpace

dSpace corresponde a una empresa que posee una gran variedad de productos. Esta empresa apoya constantemente en la transformación dinámica de la industria automotriz con soluciones para la conducción autónoma, la electromovilidad y la digitalización. Ofrece soluciones para gestión de energía, automatización industrial e ingeniería médica, así como para desafíos en aplicaciones aeroespaciales, comerciales y todoterreno, y aplicaciones marinas.

MicroLabBox es un producto de este empresa, este corresponde a un todo en uno para el laboratorio, que combina el tamaño compacto y los bajos costos del sistema con un alto rendimiento y versatilidad.

MicroLabBox le permite configurar sus aplicaciones de control, prueba o medición de forma rápida y sencilla. Este posee más de 100 canales de E/S de diferentes tipos hacen de la MicroLabBox un sistema versátil que se puede utilizar en áreas de investigación y desarrollo como robótica, mecatrónica, ingeniería médica, control de accionamientos eléctricos, energía renovable, ingeniería de vehículos o aeroespacial.

Al igual que con Quanser, se explicará como se implementó MPC utilizando el MicroLabBox:

8.4. Hardware dSpace

- **MicroLabBox [26]:** es un sistema de desarrollo compacto para el laboratorio. Este posee una alta potencia de computo, proporcionando un gran rendimiento en implementaciones en tiempo real. Además, posee una FPGA programable brindando un alto grado de flexibilidad y posibilidades para realizar implementaciones. Este posee una gran variedad de canales análogos y digitales de entrada y salida.

Figura 8.55: Hardware dSpace: MicroLabBox

El microlabbox se utilizará como un microcontrolador, el cual tomará las mediciones digitales del encoder y enviará la señal análoga de control. Además, este posee un puerto ethernet el cual permite comunicarse con un computador.

- **Cables fabricados:** Para poder utilizar el MicroLabBox en el servo motor, se tuvieron que fabricar cables para poder enviar y recibir información de este.

Se utilizó un cable BNC-RCA para enviar la señal de control. El puerto BNC se conectó a una salida análoga del MicroLabBox y el puerto RCA se conectó a la entrada análoga del 'VoltPAQ-X2 Amplifier'. De este modo, se envía la señal de control usando dSpace, esta pasa por el amplificador en donde se acondiciona la señal para posteriormente ser enviada al servomotor.

Figura 8.56: Hardware dSpace: Cable BNC-RCA

Para recibir la señal del encoder, se tuvo que fabricar un cable. Este corresponde a un cable '5-pin DIN', en donde el puerto macho se dejó intacto, pero se remplazo el puerto hembra. Este extremo posee 5 cables. Los cables relacionados a los pines Vcc y GND se soldaron a conectores banana. El resto de pines se soldaron a un conector macho DB50 (50 pines). Este cable se puede apreciar en la Figura 8.57.

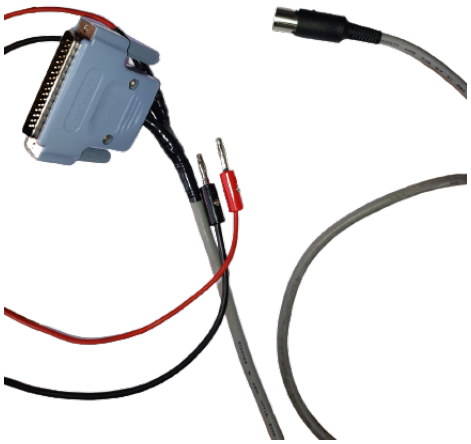


Figura 8.57: Hardware dSpace: Cable BNC-RCA

De esta forma, la señal del encoder es enviada a los puertos digitales del MicroLabBox.

Además, el encoder se energiza con $5[V]$ a través de puertos de alimentación de sensores del MicroLabBox, utilizando los cables banana.

Cabe destacar, la utilización del amplificador 'VoltPAQ-X2 Amplifier' para energizar de forma correcta el servo motor.

8.5. Software para dSpace

Se utilizaron los siguientes software para la implementación de MPC usando dSpace:

- **Vitis Model Composer:** corresponde a una toolbox de Xilinx para Matlab y Simulink. Al abrir este software, se abre matlab incluyendo la librería 'Real-Time Interface (RTI).
- **Real-Time Interface (RTI):** este corresponde a un software de la empresa dSpace. Permite realizar la conexión entre Matlab/Simulink y el MicroLabBox. Además, nos dispone de bloques en Simulink para enviar y recibir datos.

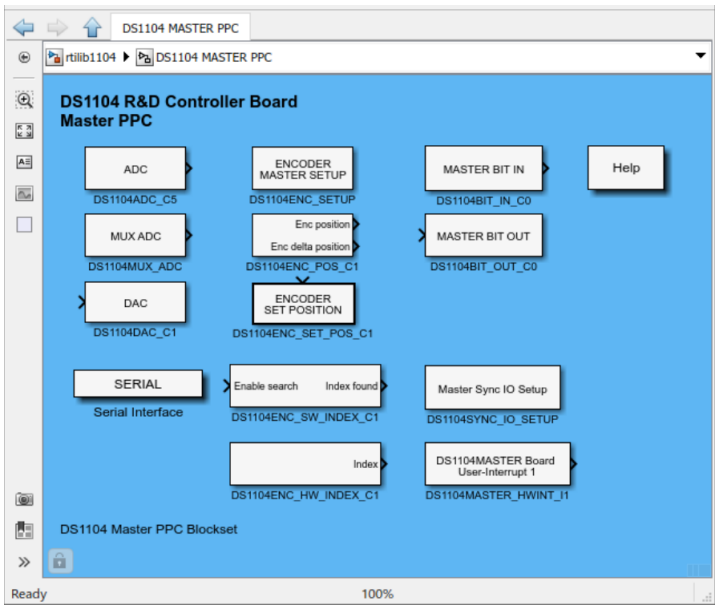


Figura 8.58: Software dSpace: Real-Time Interface (RTI)

Se utilizaron los bloques RTI DS1202, los cuales son compatibles con el modelo de MicroLabBox disponible (se detallarán más adelantes los bloques utilizados).

- **Control Desk:** es el software de experimentación de dSpace para el desarrollo continuo de ECU (unidad de control de motor). Realiza todas las tareas necesarias y le brinda un entorno de trabajo único, desde el comienzo de la experimentación hasta el final. Este software es similar a Matlab/Simulink o Labview.

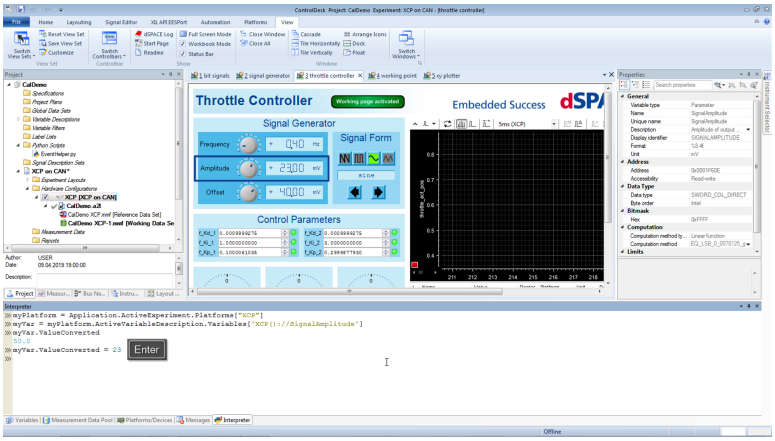


Figura 8.59: Software dSpace: Control Desk

Este software se utiliza para ejecutar los archivos compilados de simulink de formato .sdf en el MicroLabBox. Además, permite realizar un muestreo de las señales, manipularlas y guardarlas en archivos .m de matlab.

Luego de describir los programas utilizados para dSpace, se comentarán los bloques de Simulink suministrados por RTI de dSpace:

- **RTI Data:** Este bloque permite compilar el archivo utilizando la toolbox RTI. Este bloque viene incluido al abrir un nuevo archivo Simulink enfocado en dSpace.

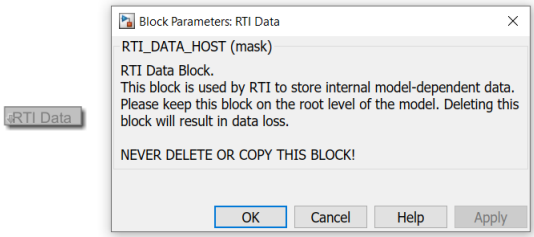


Figura 8.60: Bloque dSpace: RTI Data

- **Sensor Supply:** Este bloque permite setear el voltaje de la fuente en el MicroLabBox para alimentar con energía sensores.

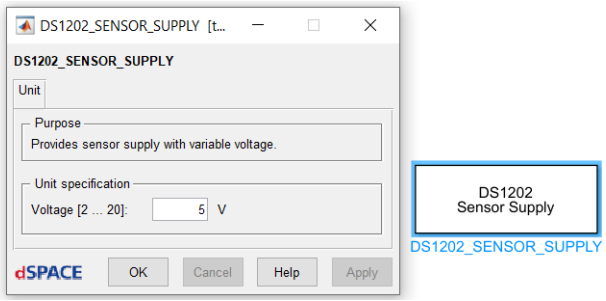


Figura 8.61: Bloque dSpace: Sensor Supply

Se setea en $5[V]$ para energizar el encoder del servo motor.

- **DAC_CLASS1_BL1:** Este bloque permite enviar una señal análoga a través de un puerto análogo de salida del MicroLabBox.

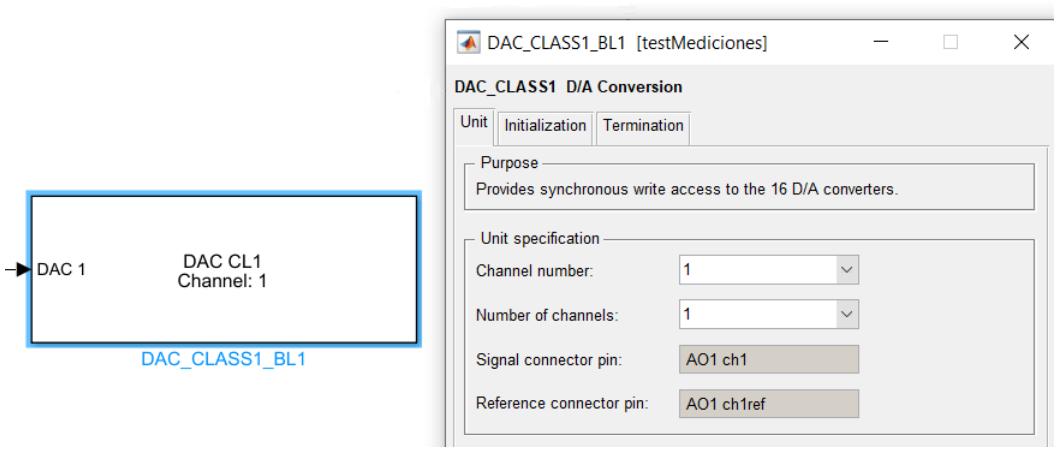


Figura 8.62: Bloque dSpace: DAC CLASS1

Este utilizó el puerto análogo 1 para enviar la señal de control.

- **EMC_Encoder_BL1:** Este bloque permite recibir la señal digital del encoder y decodificarla. De este modo, el bloque entrega directamente la posición angular registrada por el encoder.

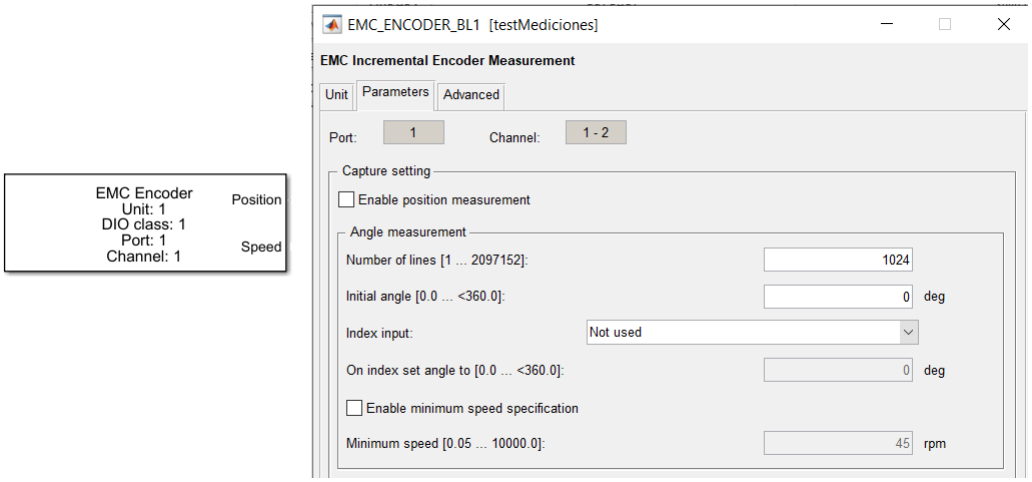


Figura 8.63: Bloque dSpace: EMC Encoder

Se utilizó el puerto digital 1 y canales 1-2. Además, se ingresó el numero de incrementos que posee el encoder. Cabe destacar que este bloque sólo registra mediciones dentro del rango de 0 a 360 grados. En caso de superarlo, tanto por arriba o por abajo, sigue contando desde la otra cota del rango de medición. Es decir, si se mide 360º y el motor sigue moviéndose, este seguirá contando desde 0, en caso contrario, si se miden 0º y el motor sigue moviéndose en sentido contrario, este seguirá contando desde 360º.

8.5.1. Acotaciones al usar MicroLabBox con Simulink

Los pasos para cargar un archivo desde el PC, usando dSpace son los siguientes:

1. Conectar MicroLabBox con PC a través del puerto Ethernet,
2. Encender switch de MicroLabBox.
3. Teniendo cerrado el software Matlab, se abre el programa "Vitis Model Compose". Al realizar esto, se abrirá matlab con las librerías RTI (puede tomar un tiempo en cargar las librerías).
4. Abrir el archivo simulink deseado teniendo en cuenta que este archivo se encuentre en la ubicación actual del Workspace. Teniendo listo el archivo, se debe compilar para generar el archivo .sdf, este corresponde al archivo ejecutable para 'Control Desk'. Se puede utilizar Ctrl+B para facilitar el proceso de compilar.

5. Teniendo el archivo ejecutable .sdf, se debe abrir 'ControlDesk'. En el se debe crear un nuevo proyecto y cargar el archivo .sdf.

6. Finalmente, en 'Control Desk' se ejecuta el proyecto iniciando la implementación en tiempo real. En este programa se puede realizar un muestreo de las señales y guardar datos.

Se realizó el mismo ejemplo realizado sobre el envío y recepción de señales con el servo motor, pero en dSpace.

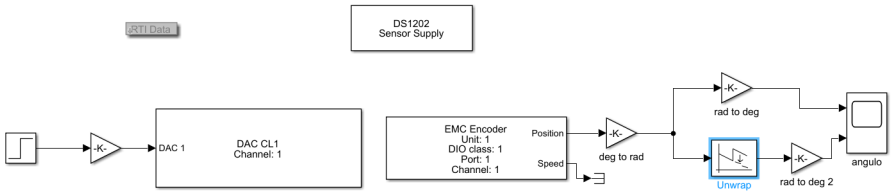


Figura 8.64: Ejemplo de envío y recepción de señales en Servo utilizando dSpace

Al utilizar 'Control Desk', se guardaron las mediciones realizadas por el encoder en archivos '.m'. Al graficar las mediciones del encoder, se obtiene:

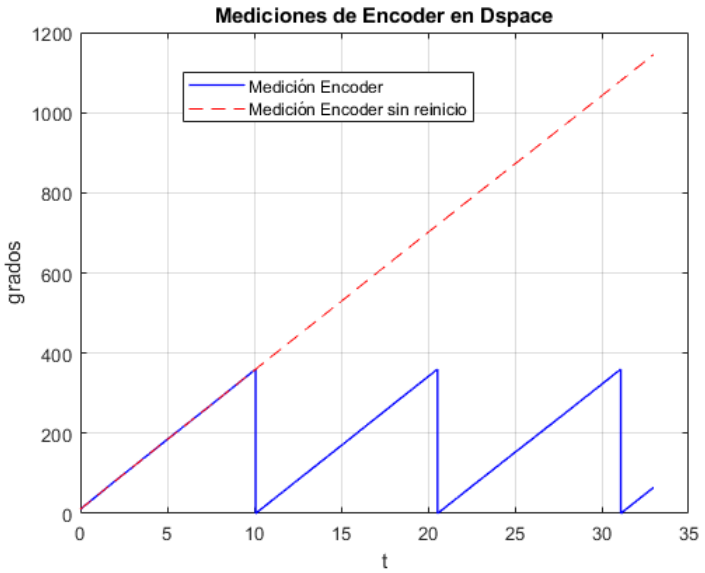


Figura 8.65: Gráfico de mediciones de ejemplo dSpace

Al implementar el ejemplo presentado en la Figura 8.64, se extraen las siguientes acotaciones sobre los bloques RTI:

- El rango de voltaje que soporta MicroLabBox respecto a las señales análogas es de $10[V]$.
- Al enviar una señal análoga de x magnitud, MicroLabBox amplifica la señal 10 veces. Es por esto que se utiliza un bloque de ganancia 1/10 para mantener la señal de control intacta.
- Al recibir un señal análoga desde el MicroLabBox, la señal en Simulink es reducida 10 veces a la señal original. En caso de que se requiera medir una señal análoga (no es el caso), esta se debe amplificar por 10.
- Para realizar mediciones del encoder, se debe utilizar el bloque 'Sensor Supply' para energizar el encoder.
- El bloque 'EMC Encoder' entrega mediciones en grado en un rango de $0^{\underline{o}}$ a $360^{\underline{o}}$.
- Se utilizó el bloque 'Unwrap' para evitar este reset en la cuenta del encoder. Se configuró el 'unwrap' con 2π. Este bloque pertenece a la toolbox de DSP. Se podria decir que es un equivalente al bloque 'Inverse Modulus' de Quarc.

8.5.2. Implementación de MPC con dSpace

Para esta implementación, se utilizaron de base los archivos realizados para Quarc, pero se cambiaron los bloques Quarc por los de dSpace. Además se agrego el bloque que permite energizar el sensor.

Después de realizar los cálculos previos para implementar MPC, se compiló el archivo Simulink. Este diagrama se puede apreciar en la Figura 8.66.

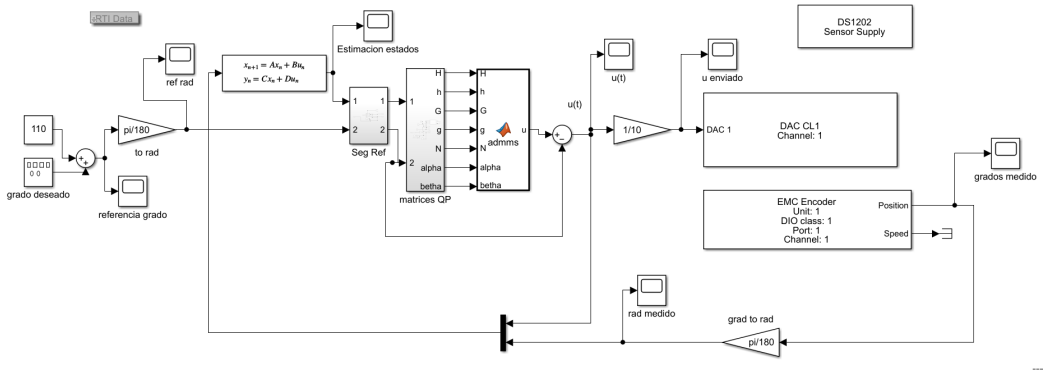


Figura 8.66: Implementación de MPC en Simulink con dSpace

Se descartó el uso del bloque 'Unwrap', debido a que entrega mediciones incorrectas al implementar MPC.

Utilizando 'Control Desk', se observó el funcionamiento de MPC. La figura 8.67 corresponde a una captura del proyecto utilizado en este programa. Cabe destacar que no se logró tomar la medición del tiempo en resolver el problema de optimización.

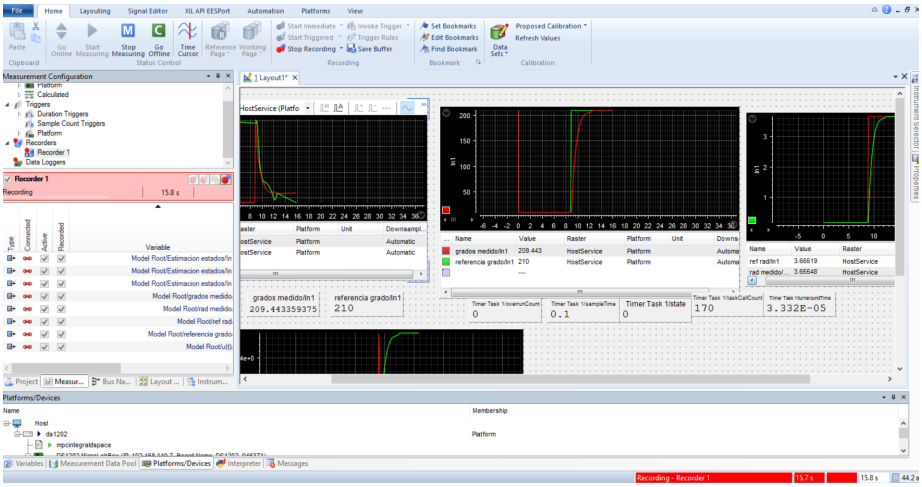


Figura 8.67: Captura de Datos en Control Desk

8.5.3. Resultados obtenidos con dSpace

A partir de la implementación detallada en la sección anterior, se realizó un control en la planta de interés. Se realizaron varias pruebas de su funcionamiento variando el tiempo de muestreo h y el horizonte de predicción N. A continuación, se presentaran los resultados gráficos de 2 casos.

- **h=0.1[s] y N=5:**

  La Figura 8.68, Figura 8.69 y Figura 8.70 corresponden a los datos obtenidos respecto a las salidas, entrada y estados respectivamente.

  Para este caso, se comprueba que la técnica de control utilizada sigue la referencia de forma aceptable respetando las restricciones designadas. Se aprecia que intenta corregir el error, pero no lo hace de forma eficiente debido al tiempo de muestreo.

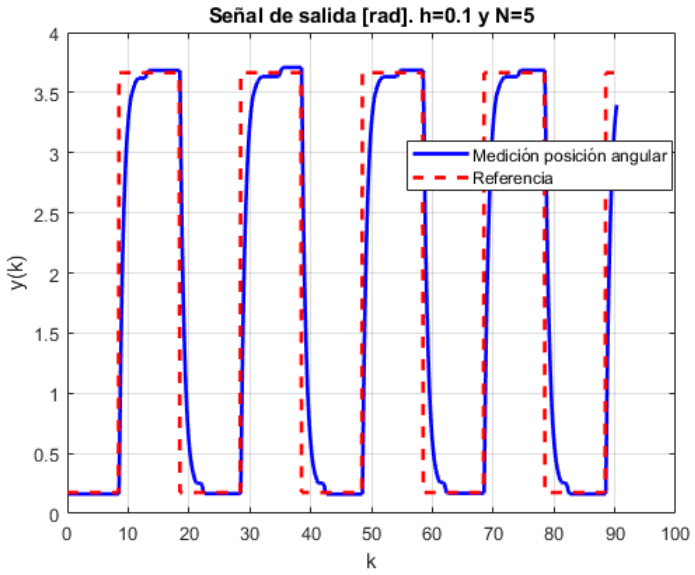


Figura 8.68: Gráfico de salida obtenida con h=0.1[s] y N=5 en dSpace.

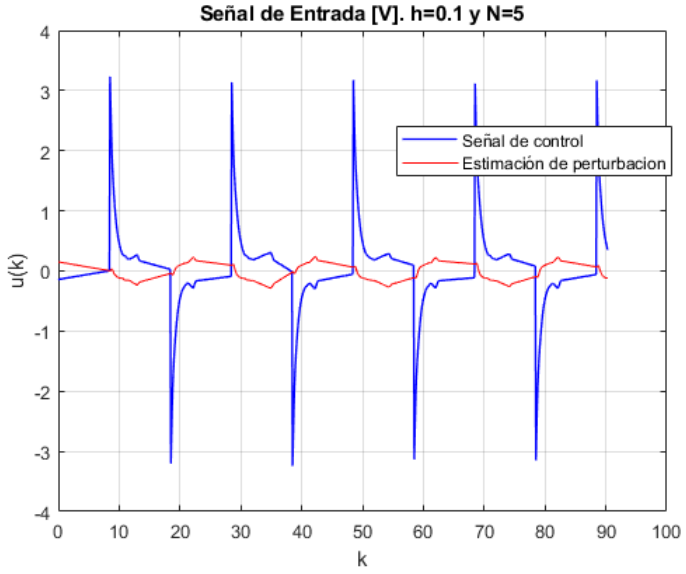


Figura 8.69: Gráfico de señal de entrada con h=0.1[s] y N=5 en dSpace.

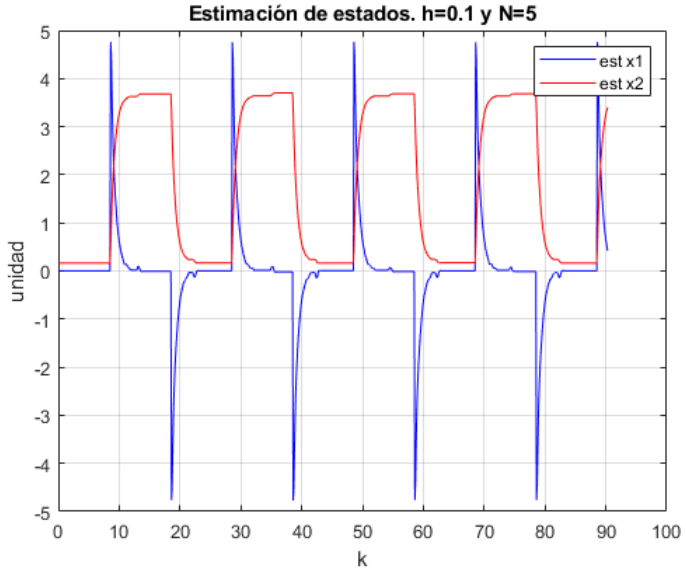


Figura 8.70: Gráfico de estimaciones de estados con h=0.1[s] y N=5 en dSpace.

- **h=0.01[s] y N=5:**

  La Figura 8.71, Figura 8.72 y Figura 8.73 corresponden a los datos obtenidos respecto a las salidas, entrada y estados respectivamente.

Para este caso, se comprueba que la técnica de control utilizada sigue la referencia de forma correcta respetando las restricciones designadas.

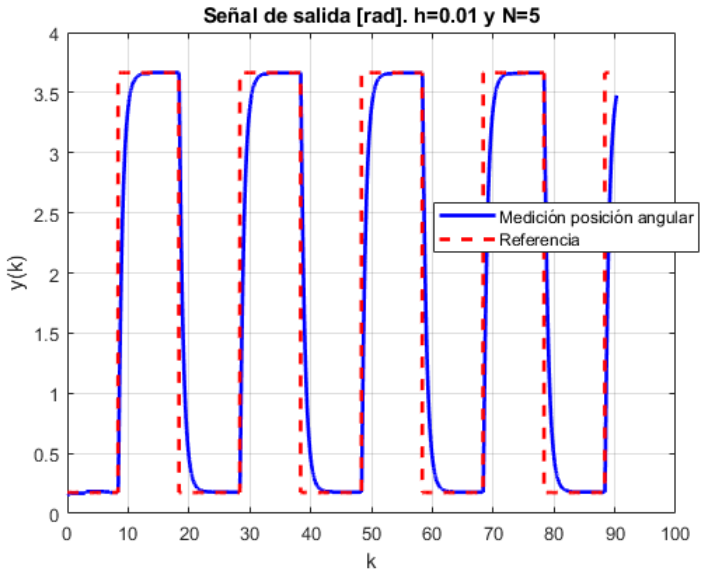


Figura 8.71: Gráfico de salida obtenida con h=0.01[s] y N=5 en dSpace.

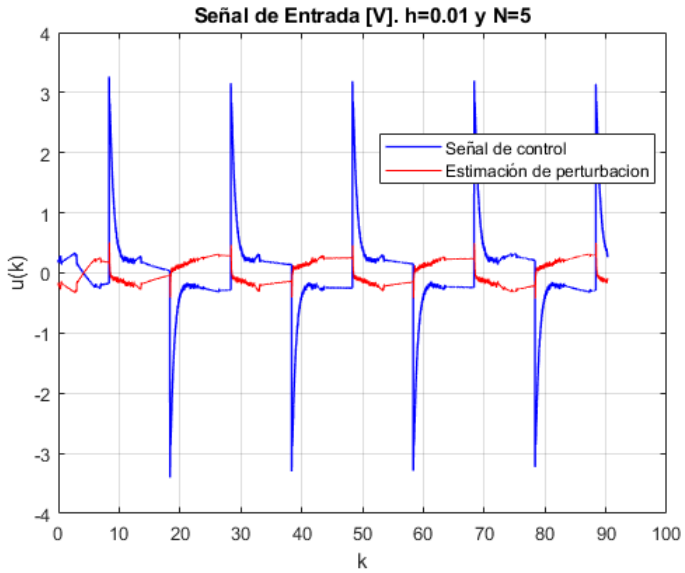


Figura 8.72: Gráfico de señal de entrada con h=0.01[s] y N=5 en dSpace.

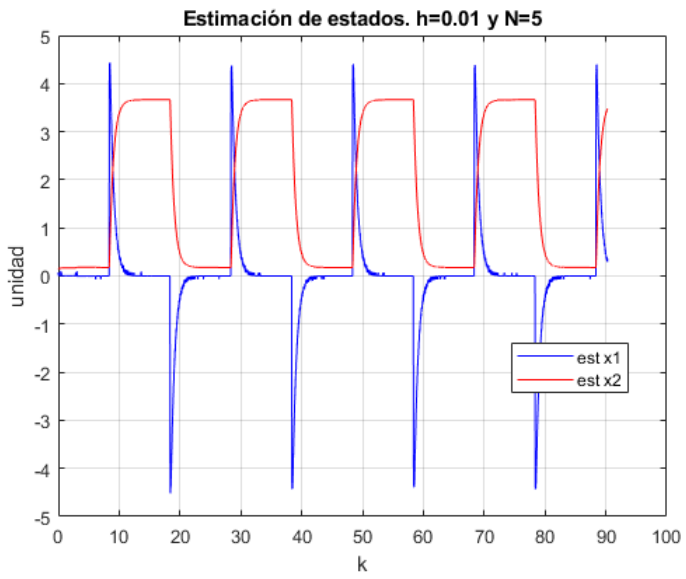


Figura 8.73: Gráfico de estimación de estados con h=0.01[s] y N=5 en dSpace.

El resto de las pruebas realizadas, se compilaron en la siguiente tabla. En donde 1 representa un funcionamiento correcto de MPC y 0 representa un fallo en el funcionamiento en MPC.

h[ms]\N	**2**	**3**	**4**	**5**	**6**	**10**	**15**
100	*1*	*1*	*1*	*1*	*1*	*1*	*1*
10	*1*	*1*	*1*	*1*	*1*	*1*	*0*
5	*1*	*1*	*1*	*1*	*1*	*1*	*0*
3	*1*	*1*	*1*	*1*	*1*	*0*	*0*
2	*1*	*1*	*1*	*1*	*0*	*0*	*0*
1.5	*1*	*1*	*0*	*0*	*0*	*0*	*0*
1	*0*	*0*	*0*	*0*	*0*	*0*	*0*

Tabla 8.4: Resultados obtenidos sobre la implementación de MPC en dSpace. 1=Éxito, 0=Fallo

Se observa que al realizar implementaciones bajo 1.2$[ms]$, el servo motor se comporta inestable. Esto debido a que existen cambios muy bruscos en la señal de control, impidiendo que el servo pueda girar correctamente.

9. Conclusiones

Este trabajo ha presentado una implementación funcional de un control por modelo predictivo con integración integral en un motor DC. Esta implementación se validó utilizando Quarc de Quanser y el MicroLabBox de dSpace. Los resultados mostraron un desempeño similar para ambos casos, alcanzando a controlar con un tiempo de muestreo de 1.2$[ms]$ con un horizonte de predicción $N = 2$.

Respecto a ADMM, se validó su uso en implementaciones de MPC. A partir del algoritmo planteado por Dang [18], se logró minimizar el problema de optimización de forma QP en forma Densa, obteniendo la señal óptima de control.

Para implementar una acción integral, se realizó a través de la estimación de una perturbación de entrada. A partir de la estimación de los estados y la perturbación, se realizó un seguimiento de referencia, el cual mueve el punto de operación del sistema de control, logrando corregir el error entre la salida y la referencia, obteniendo un error en estado estacionario 0.

Finalmente, respecto a las plataformas donde se implementó MPC, se documentó sobre el uso de Quarc y dSpace. Respecto a Quarc, se utilizaron los equipos disponibles en la universidad, de modo que sólo fue necesario explicar el uso de este software para implementaciones de control. En dSpace, debido al ser un equipo 'nuevo', se desarrolló una forma de conectar el MicroLabBox con el servo motor. Al igual que con Quarc, se documentó cómo realizar una acción de control con dSpace.

Como trabajo futuro, se espera mejorar el rendimiento al utilizar MPC usando el MicroLab-Box, para así, realizar implementaciones con tiempos de muestreo menores a 1[ms]. Algunos ejemplos sobre posibles mejoras o variaciones son: utilizar el bloque C-script de Matlab para implementar ADMM, estimar la perturbación de salida en vez de entrada o implementar otra forma de MPC con acción integral.

Referencias

[1] G. F. Franklin, J. D. Powell, A. Emami-Naeini, and J. D. Powell, *Feedback control of dynamic systems*. Prentice hall Upper Saddle River, 2002, vol. 4.

[2] K. Ogata, *Ingeniería de control moderna*. Pearson Educación, 2003.

[3] H. G. Jacob Apkarian, Michel Levis, *Rotatory Servo Integration - Student Workbook*. Quanser, 2019.

[4] E. Izaguirre Castellanos, *Sistemas de automatización*. Feijóo, 2012.

[5] L. Wang, *Model predictive control system design and implementation using MATLAB®*. Springer Science & Business Media, 2009.

[6] G. Goodwin, M. M. Seron, and J. A. De Doná, *Constrained control and estimation: an optimisation approach*. Springer Science & Business Media, 2006.

[7] K.-V. Ling, B. F. Wu, and J. Maciejowski, "Embedded model predictive control (mpc) using a fpga," *IFAC Proceedings Volumes*, vol. 41, no. 2, 2008.

[8] M. Ruf, "Model predictive control of a quanser 3dof helicopter," Master's thesis, 2013.

[9] G. C. Goodwin, H. Haimovich, D. E. Quevedo, and J. S. Welsh, "A moving horizon approach to networked control system design," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1427–1445, 2004.

[10] M. Badoni, A. Singh, B. Singh, and H. Saxena, "Real-time implementation of active shunt compensator with adaptive srlmmn control technique for power quality improvement in the distribution system," *IET Generation, Transmission & Distribution*, vol. 14, no. 8, pp. 1598–1606, 2020.

[11] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*. Nob Hill Publishing Madison, 2017, vol. 2.

[12] G. Goodwin., *Communications and Control Engineering Cap 4*, 2014.

[13] G. Espinosa-Paredes and A. V. Rodríguez, *Aplicaciones de programación no lineal*. OmniaScience, 2016.

[14] M. Escolano, I. Conde, M. Laspalas, M. Lizaranzu, J. Rodríguez, J. Alfonso, J. Orús, A. Chiminelli, J. S. de Aja, and F. M. de la Escalera, "Modelado y control predictivo de un molde de rtm," *Materiales Compuestos*, vol. 2, no. 3, pp. 56–62, 2018.

[15] E. M. Carreño, E. M. T. Ocampo, and A. Escobar, "Optimización de sistemas lineales usando métodos de punto interior," *Scientia et technica*, vol. 10, no. 24, pp. 43–48, 2004.

[16] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[17] A. Rumpf, "Overview of dual ascent," 2019.

[18] T. V. Dang, K. V. Ling, and J. M. Maciejowski, "Embedded admm-based qp solver for mpc with polytopic constraints," in *2015 European Control Conference (ECC)*. IEEE, 2015, pp. 3446–3451.

[19] J. L. Roca, *Introducción a la Electromecánica*. biWy, 2017.

[20] S. J. Chapman *et al.*, "Máquinas eléctricas," 2012.

[21] R. Mantz, "Observadores de estados," 2003.

[22] Quanser. (2020) Q8 usb data acquisition device. [Online]. Available: https://www.quanser.com/products/q8-usb-data-acquisition-device/

[23] ——. (2020) Voltpaq-x2 amplifier. [Online]. Available: https://www.quanser.com/products/voltpaq-x2-amplifier/

[24] ——. (2020) Rotary servo base unit. [Online]. Available: https://www.quanser.com/products/rotary-servo-base-unit/

[25] Y. Gorbounov, "Accurate quadrature encoder decoding using programmable logic," *Journal of Engineering Research and Technology*, vol. 2, no. 4, 2016.

[26] dSpace. (2020) Microlabbox. [Online]. Available: https://www.dspace.com/en/pub/home/products/hw/microlabbox.cfm

10. Anexos

10.1. Códigos *MATLAB*

- **Codigo preparación Matrices MPC**

```
%%para servo
clear all; clc;
K_ENC=0.0015; %para resolucion encoder srv02
h=0.01; %tiempo de meustreo
rho=0.1;

%ctes planta
a=29.4797;
K=45.9744;

%ve discreto
Ad=[exp(-a*h) 0;(1-exp(-a*h))/a 1];
Bd=[1-exp(-a*h);exp(-a*h)/a-1/a+h]*K/a;
C=[0 1];
D=0;
%Matrices de peso
Omega=(C)' *C * 0.4;
Gamma=1;
%MPC parameters
N=20;
[H,Fa]=setmpc(Ad,Bd,N,Omega,Gamma)
LB=-1*ones(N,1);
UB=-LB;
G=[-eye(N);eye(N)];
g=[-LB;UB];

% Diseno de Observador
Ax=[Ad Bd;0 0 1];
Cx=[0 1 0];
L=acker(Ax',Cx',[0.5 0.5 0.5]);
```

```
L=L'
%Matrices en forma V.E. de observador
Aobs=Ax-L*Cx;
Bobs=[[Bd;0] L];
Cobs=eye(3);
Dobs=zeros(3,2);

function [H,Fa]=setmpc(A,B,N,Omega,Gamma)
[L,Omega_N] = dlqr(A,B,Omega,Gamma)
n = length(A);
dimA = size(A,1);
dimB = size(B,2);
Oa = zeros(dimA*N,dimB*N);
 Aa =zeros(dimA*N,dimA);

    for k= 1:N
        Oa(n*k-(n-1):n*k,1) =(A^(k-1))*B;
     end

     for j = 2:N
        for k = j:N
            Oa(n*k-(n-1):n*k,j)=Oa(n*(k-1)-(n-1):n*(k-1),j-1);
        end
     end

     for k = 1:N
        Aa(n*k-(n-1):n*k,:) = A^k;
     end

     H = 2*(kron(eye(N),Gamma)...
        +Oa'*blkdiag(kron(eye(N-1),Omega),Omega_N)*Oa);
 H=(H+H')/2;
     Fa = ((2*Aa'*blkdiag(kron(eye(N-1),Omega),Omega_N)*Oa)');
end
```

- **Código ADMM**

```
function [u] = admms(H,h,G,g,N,xalpha,xbetha)
%==========================================================
% Quadratic programing (QP) problem solution:
%                    min 0.5 x'*H*x + h'*x
%                    s.t     F*x  = f
%                            G*x <= g
%   x       : Optimal solution of qp problem
%==========================================================
    n=size(H,1);  %Se preparan tamano de matrices
    q=size(G,1);
    x=1*ones(n,1);
    z=0.5*ones(q,1);
    tau=0.5*ones(q,1);

    A=[G]; %se asigan matrices de ADMM
    B=[eye(q)];
    c=[g];

    ep=1e-3;  ed=1e-3;  %se definen criterios de parada
    rho=0.1;
    rk=A*x+B*z-c;
    sk=rho*A'*B*(z);
    k=0;
    while norm(rk)>=ep || norm(sk)>=ed
        k=k+1;
        x=-xalpha-xbetha*(B*z+tau-c); %se realizan iteraciones ADMM
        z=max(0,-G*x-tau+g);
        tau=tau+A*x+B*z-c;
        if k==5000; break; end %% Criterio extra de parada
    end
    u=x(1); %se utiliza primera entrada
    %time=toc; %opcional para medicion de tiempos
```