Trabajo Fin de Máster
Máster Universitario en Ingeniería Industrial

Desarrollo de algoritmos de control basados en control predictivo explícito

Autor: Ana Sánchez Amores

Tutor: Paula Chanfreut Palacio

José María Maestre Torreblanca

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Trabajo Fin de Máster
Máster Universitario en Ingeniería Industrial

Desarrollo de algoritmos de control basados en control predictivo explícito

Autor:
Ana Sánchez Amores

Tutor:
Paula Chanfreut Palacio
Investigador Predoctoral

José María Maestre Torreblanca
Catedrático de Universidad

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Trabajo Fin de Máster: Desarrollo de algoritmos de control basados en control predictivo explícito

Autor: Ana Sánchez Amores
Tutor: Paula Chanfreut Palacio
José María Maestre Torreblanca

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

A mi hermana, por ser un ejemplo de sacrificio, constancia y esfuerzo.
A mis padres, por su apoyo y amor incondicional.
A Andrés, por acompañarme en este camino y, en ocasiones, creer en mí más que yo misma.
A mis tutores: a Pepe por la confianza que ha depositado en mí, y por las oportunidades que me está brindando, a Paula por su implicación, paciencia y ayuda. Gracias por permitirme aprender tanto de vosotros.

Ana Sánchez Amores
Máster Universitario en Ingeniería Industrial

Sevilla, 2021

Resumen

Dentro del marco del control predictivo basado en modelo (MPC) se encuentra el control MPC explícito; cuya principal ventaja reside en poder calcular el controlador de manera *offline*, reduciendo el coste computacional en la implementación, ya evita resolver un problema de programación cuadrática (QP) para cada instante de muestreo. El objetivo principal de este proyecto es la comprensión de la metodología necesaria para calcular la ley de control predictiva de forma previa a su implementación, aprovechando estos conocimientos para tratar de adaptar la estrategia explícita a algoritmos de control distribuido.

En primer lugar, se presenta el estudio de las condiciones de optimalidad de Karush Kuhn Tucker, las cuales se emplean para la construcción del mapa de regiones que caracteriza al MPC explícito. Tras la comprensión de las mismas, se procede a desarrollar las expresiones que definen los poliedros de las regiones, así como la ley de control de cada una de ellas partiendo de un problema QP. Se han incluido estas expresiones en un código en *Matlab* que permite, dado un sistema en espacios de estados con restricciones en la entrada y el estado, calcular su controlador explícito asociado. Para probar el funcionamiento del código generado, se aplica a varios sistemas de estudio, y se realizan modificaciones en el mismo para tener en cuenta el efecto de posibles perturbaciones, ya que no han sido consideradas inicialmente. Se parametriza el sistema en función de las perturbaciones con el objetivo de poder extender de forma posterior esta idea al control distribuido, considerando el acoplamiento entre subsistemas como la perturbación a un subsistema individual.

Finalmente, se integra la estrategia explícita dentro de tres populares algoritmos de MPC distribuido, adaptando el código a cada metodología. Se estudian dos estrategias cooperativas aplicadas a sistemas con agentes acoplados por la entrada, en las que existe un proceso iterativo entre subsistemas para negociar la acción de control en cada instante. Por último, se plantea un método heurístico basado en los fundamentos de la técnica min-max, en la que el acoplamiento entre subsistemas se tiene en cuenta en el modelo de perturbaciones de cada uno. Tras un arranque centralizado, cada subsistema actuará aplicando una acción de control tal que, considerando que el agente vecino va a hacer lo peor para el subsistema de estudio, minimice el daño en términos de coste.

Abstract

Within the framework of model predictive control (MPC), we find explicit MPC, whose main advantage lies in computing offline the control law, reducing the online computational cost. It avoids solving a quadratic programming (QP) problem for each sampling instant. This project's main objective is to study the methodology required to compute the MPC control law before its implementation, taking advantage of this knowledge to adapt the explicit strategy to distributed control algorithms.

Firstly, the study of the Karush Kuhn Tucker optimality conditions is presented. The optimization problem must satisfy these conditions to build the map of regions that characterize the explicit MPC. After introducing them, we proceed to develop the expressions that define the polyhedron of the regions and the control law of each of them starting from a QP problem. These expressions have been included in a *Matlab* code that calculates the associated explicit controller for a given system in state space with input and state constraints. In order to test the performance of the code, it is applied to several case studies. Some modifications are made to take into account the disturbances model since they have not been initially considered. Disturbances become a new parameter of the explicit controller to extend this idea to distributed control later, considering the coupling between subsystems as the disturbance on each local entity.

Furthermore, three popular distributed control strategies are adapted to the explicit MPC formulation. Two cooperative strategies applied to systems with agents coupled by the input are studied, where there is an iterative process between subsystems to negotiate the control action at each instant. Finally, a heuristic method is proposed based on the min-max technique, where coupling between subsystems is taken into account in the disturbance model of each of them. After a centralized start-up, each subsystem will apply a control action considering the worst disturbance from the neighbour. In other words, each agent will apply an input that attempts to minimize the cost.

Índice

Índice de Figuras

Índice de Tablas

1 Introducción

1.1 Motivación

Este proyecto podría entenderse como una extensión o continuación de mi trabajo fin de grado (TFG) "*Control predictivo de drone comercial*", en el que, tras no poder implementar el controlador que se tenía pensado inicialmente; una ley de control predictiva basada en modelo, se empezó a indagar en la estrategia explícita de la misma. El control predictivo basado en modelo (MPC) resuelve para cada periodo de muestreo un problema de optimización, obteniendo una acción de control óptima para el estado actual en cada instante. Esta estrategia se empezó a implementar hace años en sistemas con dinámicas lentas, por lo que el requerimiento computacional del controlador era menor porque se trabajaba con tiempos de muestreo bastante largos. En el caso de mi TFG, se quiso implementar un controlador de este tipo a un *drone*, que corresponde con un sistema en tiempo real con dinámicas bastante rápidas, ya que para controlar su vuelo el problema de optimización del MPC tiene que resolverse periódicamente cada ciertos milisegundos.

Se compró un pequeño *drone* comercial cuya principal ventaja era que el Instituto Tecnológico de Massachusetts, *MIT*, había desarrollado un *toolbox* en el entorno de *Matlab-Simulink*. A través de este, se puede diseñar el controlador que se va a implementar en el *drone*, para después compilar el código asociado y probar en tiempo real el comportamiento, permitiendo además realizar un análisis posterior del vuelo. La problemática surgió al utilizar una función de *Matlab (quadprog)* encargada de resolver el problema de programación cuadrática, ya que no se podía compilar y generar un código en C al estar utilizando una herramienta propia de este software.

Debido a que el controlador diseñado funcionaba correctamente en simulación en el entorno de *Simulink*, para poder implementarlo en tiempo real se planteó como alternativa emplear la estrategia explícita del MPC, ya que calcula de forma previa la ley de control para todo el espacio de estado. Este tipo de control disminuye la carga computacional en tiempo real, al resolver de forma *offline* el problema de optimización para todo x, y dividiendo el espacio de estados en una serie de regiones definidas por cuatro matrices; dos que delimitan la región y otras dos para describir la ley de control. La implementación en tiempo real se resume en cargar la base de datos con los resultados de la caracterización de regiones, y comprobar en qué región se encuentra el estado, y aplicar la acción de control asociada.

Para el cálculo de regiones se decidió hacer uso de un *toolbox* de *Matlab* específico para MPC explícito, en el que se introducen las matrices que definen al sistema discreto en espacios de estado, así como el horizonte de predicción que quiere utilizarse junto con las restricciones del problema; restringiendo los valores de la acción de control y del estado. Desafortunadamente, la ayuda que proporcionaba *Matlab* para las funciones incluidas en este *toolbox* era poco intuitiva y algo insuficiente para entender su funcionamiento, por lo que era complicado interpretar y sacar provecho de los resultados que devolvía la función. Es por esto por lo que surgió la necesidad de entender los fundamentos teóricos que respaldaban la generación de regiones, teniendo como objetivo generar un código propio que, dado el sistema en espacios de estado y sus restricciones, resolviese el controlador MPC explícito calculando todas sus regiones.

Además de ser capaces de generar un código propio que permita comprender los fundamentos de este tipo de controlador, se quiere aprovechar esta ventaja para ser capaces de estudiar el MPC explícito ante diferentes casuísticas, así como tratar de implementarlo de manera distribuida y aprovechar sus características del cálculo *offline* para reducir la carga computacional.

1.2 Introducción al control predictivo

Los métodos de control predictivo basado en modelo (MPC, *Model Predictive Control*) [1] se caracterizan por emplear un modelo del sistema de estudio, para tratar de predecir el comportamiento futuro del mismo. Esta exitosa estrategia emplea un horizonte de predicción N, para el cual se calculan las acciones de control óptimas minimizando una función objetivo, empleado únicamente la primera componente que corresponde al instante actual. El éxito de la aplicación de esta estrategia reside en la fiabilidad del modelo de la planta, ya que debe considerar en la medida de lo posible, las características dinámicas del sistema para minimizar las discrepancias entre el modelado y la realidad.

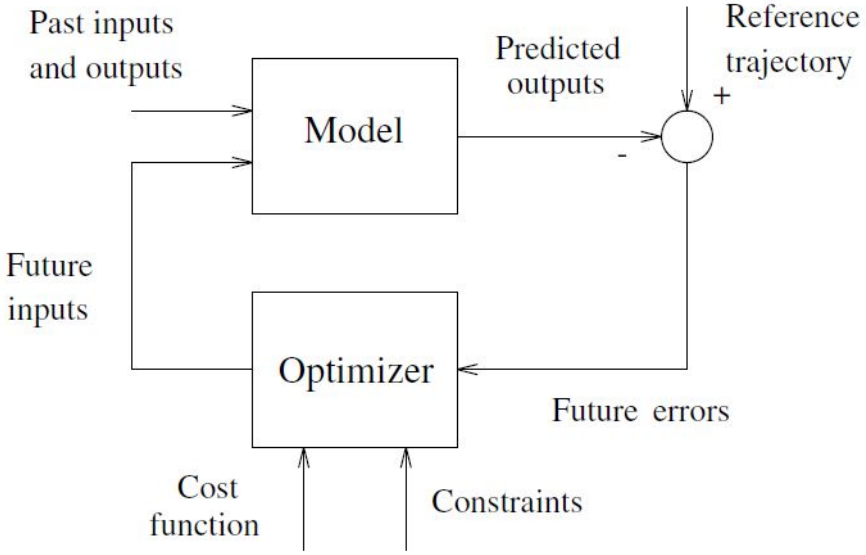


Figura 1.1 Estructura MPC [1].

Una de las características que destacan de este control óptimo, es el elevado número de entradas y salidas con las que es capaz de trabajar, así como la definición de restricciones del problema de control, donde se pueden limitar los valores del estado y la acción de control. Esto presenta una gran ventaja en comparación con otras estrategias de control, ya que a la hora de implementarlo en un sistema real, este tendrá limitaciones de potencia, velocidad, caudal... que se contemplan con estas restricciones. Por ejemplo, para el sistema de estudio del *drone*, si se vuela en interior, es importante establecer restricciones en el vector de estados, ya que el desplazamiento horizontal y vertical estará limitado por las características de la estancia. Además, los motores del multirrotor tienen una velocidad máxima que debe estar limitada.

1.3 MPC explícito

Una de las principales limitaciones del MPC tradicional o implícito, es la elevada carga computacional que supone la resolución de un problema de optimización para cada instante de tiempo, con el objetivo de calcular la actuación óptima a aplicar. Esta limitación se agrava si la planta de estudio se caracteriza por una dinámica rápida, como por ejemplo un *drone*, donde se trabaja con periodos de muestreo muy pequeños, del orden de 5ms. La complejidad de la resolución del problema de optimización de manera *online* aumenta con el tamaño del sistema y horizontes de predicción mayores que uno, ya que las matrices que definen el sistema se extienden para todo N.

La versión explícita del MPC, resuelve la problemática que surge debido a la elevada carga computacional que supone la resolución de un problema de programación cuadrática para cada instante de muestreo, ya que, la principal característica de este planteamiento es el cálculo *offline* de la ley de control para todo el espacio de estados del problema. Este cálculo *offline* resulta en la creación de un mapa del estado, dividiéndolo en distintas particiones, denominadas regiones, formadas por poliedros. Cada una de ellas queda definida mediante una desigualdad matricial, que define la condición de pertenencia a dicha región, y dos matrices que componen su ley de control. A priori, esta característica hace que su implementación en sistemas con una dinámica rápida sea posible, ya que, dado un valor del estado, las operaciones *online* se limitan a comprobar

en qué región se encuentra y aplicar la ley de control correspondiente. Al ser una versión más del MPC, también contempla restricciones en el estado y en la entrada, siendo las primeras las que delimitan el mapa del estado.

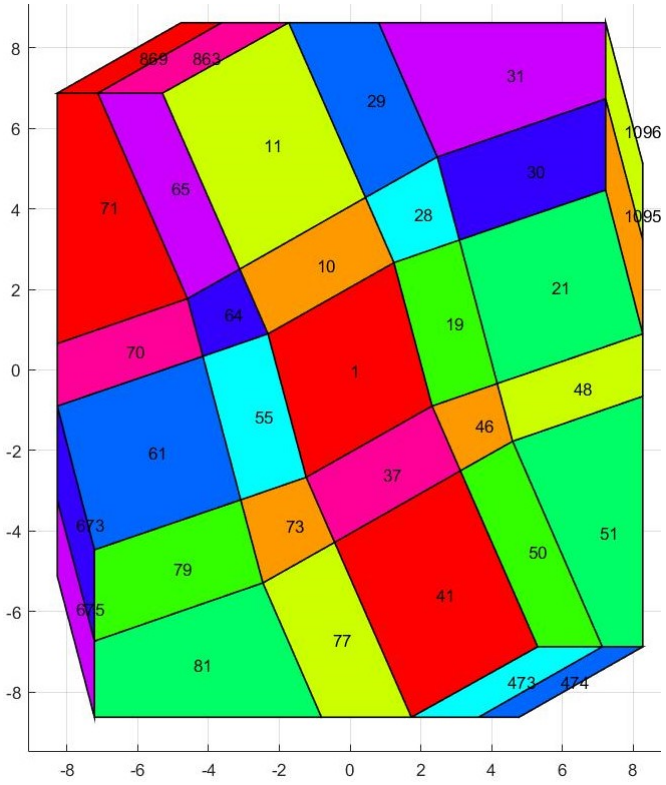


Figura 1.2 División del mapa del estado en regiones.

En la Figura 1.2 se puede apreciar un ejemplo de cómo quedaría el mapa de regiones para un sistema con estado de dimensión 2. Cada región se caracteriza con cuatro matrices, las cuales se nombran con un subíndice i, que expresa la región de estudio.

- A_{ri}, b_{ri}: delimitan el poliedro que define la región.
- K_i, U_{0i}: definen la ley de control de la región i.

Por lo tanto, a la hora de implementar el MPC explícito, lo primero será comprobar la siguiente desigualdad para identificar en qué región nos encontramos.

$$A_{ri}\, x \leq b_{ri} \quad \forall i = 1,2,...,n_{reg} \tag{1.1}$$

Conocida i, se aplica la siguiente acción de control:

$$U_i = K_i\, x + U_{0i} \tag{1.2}$$

Es importante destacar que, al igual que presenta la ventaja de poder calcular la ley de control de manera *offline* y reducir así el tiempo de computación, tiene la limitación de que es poco escalable para sistemas con un vector de estados de elevada dimensión, numerosas restricciones o un horizonte de predicción elevado. Todos estos factores hacen que el número de regiones incremente considerablemente, y aunque el cálculo *offline* pueda no suponer un problema ya que solamente se realiza una vez, si contamos con un elevado número de regiones y un sistema con una dinámica rápida, la comprobación *online* para comprobar en qué región se encuentra el sistema se ralentiza al contar con miles de regiones.

2 Condiciones de optimalidad

El presente capítulo se centra en el estudio y desarrollo de las condiciones necesarias que han de cumplirse a la hora de resolver el problema de optimización que supone la implementación del MPC explícito. Las condiciones de optimalidad de Karush Kuhn Tucker (KKT) suponen una generalización de los multiplicadores de Lagrange para restricciones de desigualdad.

2.1 Multiplicadores de Lagrange

A través de los multiplicadores de Lagrange se consigue encontrar el máximo o el mínimo de un problema de optimización con restricciones, solamente de igualdad, en los valores de entrada de una función.

$$\begin{aligned} &\min_x f(x) \\ &\text{s.a } g(x) = c \ (c = \text{cte}) \end{aligned} \tag{2.1}$$

La solución del problema de optimización anterior consiste en los puntos donde las curvas de nivel de f y g sean tangentes, que corresponde con aquellos puntos donde el gradiente de f y g son paralelos. La función g solamente tendrá una curva de nivel al estar igualada al valor c constante, pero f tendrá tantas curvas como $f(x) = k$ podamos definir, por lo que, probando con distintas k encontraremos el punto de tangencia.

Se conseguirá la condición de que los gradientes sean paralelos a través del gradiente de la Lagrangiana del problema de optimización, introduciendo un multiplicador de Lagrange λ. Como los gradientes son paralelos en el óptimo, basta encontrar la constante que, al multiplicar, nos de el mismo vector.

$$\mathscr{L} = f(x) + \lambda(g(x) - c) \tag{2.2}$$

$$\nabla_x \mathscr{L} = f_x + \lambda g_x = 0 \tag{2.3}$$

Siendo f_x y g_x la derivada parcial de las funciones con respecto a x; la variable a optimizar.

2.2 Condiciones de Karush Kuhn Tucker

Introduciendo restricciones de desigualdad, se considera el siguiente problema de optimización:

$$\begin{aligned} &\min_x f(x) \\ &\text{s.a } g_i(x) \leq 0 \quad \forall i = 1...m \\ &\quad\ h_j(x) = 0 \quad \forall j = 1...k \end{aligned} \tag{2.4}$$

Las condiciones de Karush Kuhn Tucker [5] distinguen entre restricciones activas e inactivas en el óptimo. En el primer caso, las restricciones de desigualdad que estén activas en el óptimo se tratarán como restricciones de igualdad, asignándoles un multiplicador de Lagrange (λ). Si por el contrario la restricción no está activa en el óptimo deberá ignorarse, por lo que su multiplicador λ se anulará. Por lo tanto, para cada restricción, la función que la define o su multiplicador deberá de ser cero según si está activa o inactiva respectivamente.

Para escribir el Lagrangiano del problema se deben expresar todas las restricciones de desigualdad como menor o igual a cero (≤ 0), para posteriormente sumarle a la función objetivo el producto de cada restricción por su multiplicador de Lagrange.

$$\mathscr{L} = f(x) + \sum_{\forall i} \lambda_i g_i(x) + \sum_{\forall j} \mu_j h_j(x) \tag{2.5}$$

Siendo S el conjunto factible donde se cumplen las restricciones, e $I(x)$ las restricciones activas en x:

$$S = \{x \in \mathbb{R}^n \mid g_i(x) \leq 0 \; i = 1...m, \; h_j(x) = 0 \; j = 1...k\} \tag{2.6}$$

$$I(x) := \{i \mid g_i(x) = 0\} \tag{2.7}$$

Se definen las condiciones KKT para un punto $x^* \in S$:

$$\nabla_x f(x^*) + \sum_{\forall i} \lambda_i \nabla_x g_i(x^*) + \sum_{\forall j} \mu_j \nabla_x h_j(x^*) = 0 \tag{2.8}$$

$$\lambda_i g_i(x^*) = 0 \tag{2.9}$$

$$\lambda_i \geq 0 \tag{2.10}$$

De la ecuación (2.9) se extraen las siguientes condiciones:

- Restricciones activas: $i \in I(x) \rightarrow$ La restricción se verifica con estricto signo de igualdad, cumpliendo el multiplicador $\lambda_i > 0$.
- Restricciones inactivas: $i \notin I(x) \rightarrow$ Si la restricción no está activa debe ignorarse, anulándose el multiplicador de Lagrange $\lambda_i = 0$.

Nótese que, para restricciones de igualdad estricto, no existen condiciones de positividad para el multiplicador de Lagrange μ_j asociado a cada restricción.

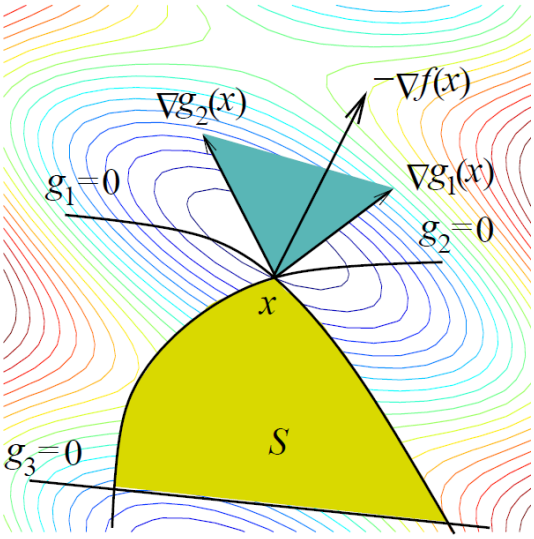


Figura 2.1 *Normal cone* [2].

Se puede concluir por tanto, que el gradiente negativo de la función objetivo $f(x)$ se puede expresar como una combinación lineal positiva de los gradientes de las restricciones activas. Esto se ilustra en la Figura 2.1, en la que se define el concepto "*normal cone* N_S [2], que corresponde con un cono dibujado por los gradientes de las restricciones activas.

Consideraciones:

- $-\nabla_x f(x^*) \in N_S$, por lo que $-\nabla_x f(x^*)$ apunta fuera del conjunto S.
- Si x^* = mínimo, no se podrá decrementar f dentro de S.

2.2.1 Problema de programación cuadrática (QP)

Debido a que la implementación del control predictivo supone la resolución de un problema QP, se aplican las condiciones de optimalidad anteriores a un problema de este tipo, formulado de la siguiente forma:

$$\begin{aligned} &\min_U \frac{1}{2} U^T H U + x^T F^T U \\ &\text{s.a } GU \leq W + Sx \end{aligned} \tag{2.11}$$

Expresando el Lagrangiano del problema QP como:

$$\mathscr{L} = \frac{1}{2} U^T H U + x^T F^T U + \lambda (GU - W - Sx), \tag{2.12}$$

se aplican las condiciones KKT:

$$\nabla_U \mathscr{L} = HU + Fx + G^T \lambda = 0 \tag{2.13}$$

$$\lambda^T (GU - W - Sx) = 0 \tag{2.14}$$

$$\lambda \geq 0 \tag{2.15}$$

2.3 Aplicación a MPC explícito

La estrategia explícita del control predictivo se basa en resolver el problema de optimización QP de manera *offline*, es decir, de forma previa a su implementación, para todo el conjunto de estados. Este estudio divide todo el espacio de estados en un número n_{reg} de regiones, cada una caracterizada por las ecuaciones que delimitan y definen dicha región, representadas por la desigualdad matricial:

$$A_i x_k \leq b_i \quad \forall i = 1...n_{reg} \tag{2.16}$$

Además, para cada región existe una ley de control en función del estado, definida mediante dos matrices:

$$U_i = K_i x_k + U_{0_i} \quad \forall i = 1...n_{reg} \tag{2.17}$$

Por lo que la implementación de este controlador, una vez calculadas todas las regiones y las matrices que las caracterizan, consiste en comprobar en qué región del espacio se encuentra el estado actual, comprobando la desigualdad de pertenencia a la región, y aplicar la ley de control correspondiente.

2.3.1 Desarrollo de las regiones

Las regiones se definen distinguiendo entre las restricciones que se encuentran activas e inactivas en cada región [6, 7], apoyándose en las condiciones de optimalidad KKT estudiadas en este capítulo, ya que establecen distintas condiciones que han de cumplirse según si en el óptimo la restricción se encuentra activa o no.

Formulando de nuevo el problema QP:

$$\begin{aligned} &\min_U \frac{1}{2} U^T H U + x^T F^T U \\ &\text{s.a } GU \leq W + Sx \end{aligned} \tag{2.18}$$

Se va a emplear la siguiente notación:

- $\tilde{G}$, $\tilde{S}$, $\tilde{W}$, $\tilde{\lambda}$ $\rightarrow$ filas de sus respectivas matrices que corresponden con las restricciones que están **activas**.
- $\hat{G}$, $\hat{S}$, $\hat{W}$, $\hat{\lambda}$ $\rightarrow$ filas de sus respectivas matrices que corresponden con las restricciones que están **inactivas**.

Se aplican a continuación las condiciones KKT descritas anteriormente, haciendo distinción entre restricciones activas e inactivas. Combinando y desarrollando estas ecuaciones se obtendrán las desigualdades que definen las regiones, así como su ley de control asociada.

$$\nabla_U \mathscr{L} = HU + Fx + G^T \lambda = 0 \tag{2.19}$$

$$\lambda_i (G^i U - W^i - S^i x) = 0 \tag{2.20}$$

$$\tilde{G}U - \tilde{W} - \tilde{S}x = 0 \tag{2.21}$$

$$\hat{G}U \leq \hat{W} + \hat{S}x \tag{2.22}$$

$$\tilde{\lambda} \geq 0, \quad \hat{\lambda} = 0 \tag{2.23}$$

Suponiendo: $G = \begin{bmatrix} \tilde{G} \\ \hat{G} \end{bmatrix}$ y $\lambda = \begin{bmatrix} \tilde{\lambda} \\ \hat{\lambda} \end{bmatrix}$ con $\hat{\lambda} = 0$, se sustituye en la ecuación 2.19 esta idea:

$$HU + Fx + \begin{bmatrix} \tilde{G} \\ \hat{G} \end{bmatrix}^T \begin{bmatrix} \tilde{\lambda} \\ 0 \end{bmatrix} = 0 \;\rightarrow\; HU + Fx + \tilde{G}^T \tilde{\lambda} = 0 \;\rightarrow\; U = -H^{-1} \left(Fx + \tilde{G}^T \tilde{\lambda} \right) \tag{2.24}$$

Sustituyendo en 2.21 se obtiene la expresión de los multiplicadores de Lagrange de las restricciones activas:

$$\tilde{\lambda}(x) = -\left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1} \left(\tilde{W} + \left(\tilde{S} + \tilde{G}H^{-1}F\right)x\right) \tag{2.25}$$

Por lo que la ley de control de cada región puede expresarse en función de las restricciones activas y el valor del estado x:

$$U(x) = H^{-1} \left[\tilde{G}^T \left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1} \left(\tilde{W} + \left(\tilde{S} + \tilde{G}H^{-1}F\right)x\right) - Fx \right] \tag{2.26}$$

$$U(x) = \underbrace{H^{-1} \left[\tilde{G}^T \left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1} \left(\tilde{S} + \tilde{G}H^{-1}F\right) - F \right]}_{K} x + \underbrace{H^{-1}\tilde{G}^T \left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1} \tilde{W}}_{U_0} \tag{2.27}$$

Se construyen las regiones aplicando las ecuaciones 2.22 y 2.23 a las expresiones que se han obtenido para $\tilde{\lambda}$ y $U(x)$, separando términos dependientes e independientes de x para construir las fronteras que delimitan la región.

- Condición de restricciones activas $\tilde{\lambda}$:

$$\tilde{\lambda}(x) = -\left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1} \left(\tilde{W} + \left(\tilde{S} + \tilde{G}H^{-1}F\right)x\right) \geq 0 \tag{2.28}$$

$$-\left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1} \left(\tilde{S} + \tilde{G}H^{-1}F\right)x \;\geq\; \left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1} \tilde{W} \tag{2.29}$$

$$\underbrace{\left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1} \left(\tilde{S} + \tilde{G}H^{-1}F\right)}_{A_1} x \;\leq\; \underbrace{-\left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1} \tilde{W}}_{b_1} \tag{2.30}$$

- Condición de restricciones inactivas utilizando la expresión 2.27:

$$\hat{G}(Kx + U_0) \leq \hat{W} + \hat{S}x \;\; \rightarrow \;\; \hat{G}(Kx + U_0) - \hat{W} - \hat{S}x \leq 0 \tag{2.31}$$

$$\underbrace{\left(\hat{G}K - \hat{S}\right)}_{A_2} x \;\leq\; \underbrace{\hat{W} - \hat{G}U_0}_{b_2} \tag{2.32}$$

Agrupando, las regiones quedan definidas mediante:

$$\left[\begin{array}{c} A_1 \\ A_2 \end{array}\right] x \leq \left[\begin{array}{c} b_1 \\ b_2 \end{array}\right] \tag{2.33}$$

Habrá tantas restricciones como filas tenga la matriz G, y existirá un número máximo de regiones correspondiente a todas las posibles combinaciones de restricciones. Sea $q = n^{\circ}\ filas(G)$, el número máximo de regiones es igual a 2^q. Se habla de número máximo porque existen combinaciones de restricciones que no pueden darse a la vez, ya que son incompatibles.

2.3.2 Simplificación de regiones

Una de las limitaciones del control predictivo explícito es el gran número de regiones que puede llegar a calcular; creciendo a medida que aumentamos el horizonte de predicción, y con ello la dimensión de las matrices que definen las restricciones del problema. Para reducir la complejidad del problema existen distintos métodos de simplificación de regiones [8], que pueden dividirse en aquellos que no introducen suboptimalidad al problema y en los que sí.

Sin introducir suboptimalidad

- *Optimal Region Merging Method*. Fusiona aquellas regiones que tengan la misma expresión de la acción de control mediante una unión convexa. Es aplicable a estrategias de MPC explícito con independencia de sus propiedades.
- *Clipping*. Aprovecha la continuidad de la ley de control, que está garantizada si el modelo es lineal, para eliminar regiones con la acción de control saturada [9]. La característica de este método es que el espacio que han dejado las regiones que han sido eliminadas se cubren o rellenan extendiendo las regiones vecinas no saturadas.
- *Separation*. Similar al método *clipping* anterior ya que elimina las regiones con la acción de control saturada, pero, a diferencia de este, no cubre después los huecos, por lo que solo permanecen las regiones que no están saturadas. Ambos métodos llegan al mismo resultado o nivel de reducción de regiones, siendo el anterior menos demandante computacionalmente.

Introduciendo suboptimalidad

- *Fitting*. Un método para reducir el número de regiones introduciendo suboptimalidad es reemplazar la ley de control óptima $f_{opt}(x)$ por una función definida a trozos más simple $f_{aprox}(x)$ mientras se minimiza la suboptimalidad, representada por:

$$\int_{\chi} \| f_{opt}(x) - f_{aprox}(x) \| \, dx \tag{2.34}$$

  Una manera de proceder sería, en primer lugar, obtener una partición más sencilla del espacios de estado resolviendo el MPC utilizando un horizonte de predicción inferior, lo que resultará en un número menor de regiones al que teníamos. Denotando esta nueva ley de control como $f_{aprox}(x) = Kx + U_0$, el segundo paso consiste en optimizar los parámetros K y U_0 de la ley de control subóptima para minimizar esta suboptimalidad, representada mediante la integración del error cuadrático de la expresión (2.34).

3 Desarrollo del código para MPC explícito

En este capítulo se detallan los pasos que han sido necesarios para la generación del código en *Matlab* a través de una "clase" que, dado un sistema expresado en espacios de estado junto con sus restricciones, devuelve el mapa dividido en las distintas regiones que caracterizan al MPC explícito. Dicho código puede encontrarse en el *Apéndice A* de esta memoria. El propósito de una "clase" o *class*, es definir un objeto que encapsule datos y operaciones que se realizan sobre estos datos.

3.1 Algoritmo para el modelo en espacio de estados

Inicialmente, se ha considerado un modelo en espacio de estados como el que se muestra a continuación:

$$x_{k+1} = A\,x_k + B\,u_k \tag{3.1}$$

donde:

- x_k representa el vector de estados, que pertenece a $\mathbb{R}^{n_x \text{ x } 1}$, con n_x igual al número de estados.
- A es una matriz que modela la relación entre el estado en el instante actual y el anterior, $\in \mathbb{R}^{n_x \text{ x } n_x}$.
- u_k es un vector que representa la entrada o acciones de control sobre el sistema $u_k \in \mathbb{R}^{n_u \text{ x } 1}$, con n_u igual al número de entradas.
- B es una matriz que modela la relación entre el estado en el instante actual y la entrada anterior, $\in \mathbb{R}^{n_x \text{ x } n_u}$.

El efecto de las perturbaciones w_k en el sistema se estudiará más adelante. Tanto las matrices como los vectores que se acaban de describir deben expresarse para todo el horizonte de predicción N, ya que el control predictivo resuelve el problema de optimización para los N siguientes instantes de tiempo, aplicando únicamente la entrada correspondiente al instante actual. La representación matricial del sistema en espacios de estado se describe mediante:

$$X = G_x\,x + G_u\,U \tag{3.2}$$

Siendo:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ x_N \end{bmatrix} \in \mathbb{R}^{N \cdot n_x \text{ x } 1}, \quad U = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-2} \\ u_{N-1} \end{bmatrix} \in \mathbb{R}^{N \cdot n_u \text{ x } 1} \tag{3.3}$$

$$G_x = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^{N-1} \\ A^N \end{bmatrix} \in \mathbb{R}^{N \cdot n_x \text{ x } n_x}, \quad G_u = \begin{bmatrix} B & 0 & 0 & 0 & 0 \\ AB & B & 0 & 0 & 0 \\ A^2B & AB & B & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \dots & AB & B \end{bmatrix} \in \mathbb{R}^{N \cdot n_x \text{ x } N \cdot n_u} \tag{3.4}$$

Minimizando la siguiente función de coste se obtendrá la acción de control óptima del sistema para cada instante de muestreo.

$$V = \sum_{0}^{N-1} \left(x_{k+1}^T \, Q \, x_{k+1} \, + \, u_k^T \, R \, u_k \right) \tag{3.5}$$

El primer sumando penaliza el error entre el estado predicho y el origen, ponderado por la matriz de pesos Q, y el segundo sumando el esfuerzo de control, ponderado con R. Ambas matrices son matrices diagonales, encargándose la primera del seguimiento de referencias y la segunda de la acción de control. Estas matrices también deben extenderse para todo el horizonte de predicción N para poder expresar el problema al completo en forma matricial:

$$\hat{Q} = \begin{bmatrix} Q & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 \\ 0 & 0 & Q & 0 \\ 0 & 0 & 0 & Q \end{bmatrix} \in \mathbb{R}^{\,N \cdot n_x \text{ x } N \cdot n_x}, \qquad \hat{R} = \begin{bmatrix} R & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 \\ 0 & 0 & R & 0 \\ 0 & 0 & 0 & R \end{bmatrix} \in \mathbb{R}^{\,N \cdot n_u \text{ x } N \cdot n_u} \tag{3.6}$$

Se expresa a continuación la función de costes de la ecuación 3.5 de forma matricial:

$$V = X^T \, \hat{Q} \, X \, + \, U^T \, \hat{R} \, U \tag{3.7}$$

Sustituyendo la expresión de X de la ecuación 3.2 en la ecuación anterior:

$$V = 2 \left((G_x \, x)^T \, \hat{Q} \, G_u \right) U \, + \, U^T \left(G_u^T \, \hat{Q} \, G_u + \hat{R} \right) U \tag{3.8}$$

Agrupando se obtienen las ecuaciones de F y H:

$$F = \left(G_x^T \, \hat{Q} \, G_u \right)^T \tag{3.9}$$

$$H = G_u^T \, \hat{Q} \, G_u + \hat{R} \tag{3.10}$$

Quedando la expresión del problema de programación cuadrática de la siguiente forma:

$$\min_U \frac{1}{2} U^T H U + x^T F^T U \tag{3.11}$$

Una de las ventajas del control predictivo consiste en poder resolver un problema de optimización satisfaciendo restricciones en el estado y en la entrada al sistema, expresadas como:

$$\begin{aligned} x_{min} \leq x_k \leq x_{max} \quad \forall k > 0 \\ u_{min} \leq u_k \leq u_{max} \quad \forall k > 0 \end{aligned} \tag{3.12}$$

Compactando las desigualdades anteriores en forma matricial:

$$\begin{aligned} A_x \, x_k \leq b_x \quad \forall k \in [1, \, N] \\ A_u \, u_k \leq u_x \quad \forall k \in [0, \, N-1] \end{aligned} \tag{3.13}$$

Extendiendo para todo el horizonte de predicción N:

$$\hat{A}_x = \begin{bmatrix} A_x & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 \\ 0 & 0 & A_x & 0 \\ 0 & 0 & 0 & A_x \end{bmatrix}, \quad \hat{b}_x = \begin{bmatrix} b_x \\ \vdots \\ b_x \\ b_x \end{bmatrix}, \quad \hat{A}_u = \begin{bmatrix} A_u & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 \\ 0 & 0 & A_u & 0 \\ 0 & 0 & 0 & A_u \end{bmatrix}, \quad \hat{b}_u = \begin{bmatrix} b_u \\ \vdots \\ b_u \\ b_u \end{bmatrix} \tag{3.14}$$

Se pueden expresar las restricciones en función de los vectores X y U:

$$\begin{aligned} \hat{A}_x X &\leq \hat{b}_x \\ \hat{A}_u U &\leq \hat{b}_u \end{aligned} \tag{3.15}$$

Para expresarlas todas en función de la misma variable, U, se sustituye la ecuación 3.2 en la expresión de las restricciones en el estado:

$$\hat{A}_x X \leq \hat{b}_x \rightarrow \hat{A}_x (G_x x + G_u U) \leq \hat{b}_x \rightarrow \hat{A}_x G_u U \leq \hat{b}_x - \hat{A}_x G_x x \tag{3.16}$$

Debido a que en los problemas QP las restricciones se expresan de la forma $GU \leq W + Sx$ se agruparán todas las restricciones definidas por desigualdades matriciales, y se identificarán los distintos términos:

$$\underbrace{\begin{bmatrix} \hat{A}_x G_u \\ \hat{A}_u \end{bmatrix}}_{G} U \leq \underbrace{\begin{bmatrix} \hat{b}_x \\ \hat{b}_u \end{bmatrix}}_{W} + \underbrace{\begin{bmatrix} -\hat{A}_x G_x \\ 0 \end{bmatrix}}_{S} x \tag{3.17}$$

3.2 Estudio de regiones

Una vez obtenidas las matrices que definen el problema de optimización que ha de resolverse para implementar un controlador MPC en el sistema de espacio de estados dado, se procede al estudio de regiones del problema QP, formulado con la siguiente expresión:

$$\begin{aligned} &\min_U \frac{1}{2} U^T H U + x^T F^T U \\ &\text{s.a } GU \leq W + Sx \end{aligned} \tag{3.18}$$

En el capítulo anterior se estudió cómo afectaba en la definición de regiones la distinción entre restricciones activas e inactivas. Debido a que el mapa de regiones se genera para todo el espacio de estados comprendido dentro de los límites del sistema, se han estudiado todas las posibles combinaciones de restricciones para de esta manera poder distinguir aquellas que se encuentran activas y las que no, y así poder delimitar las regiones. El número máximo de regiones viene determinado por el número de combinaciones entre todas las restricciones del problema, enumerándolas a través del número de filas de la matriz G, y definiendo este valor máximo como $2^{\text{filas}(G)}$. Recordar que, según la nomenclatura empleada, las restricciones activas se distinguen con el superíndice $\tilde{}$, y las inactivas con $\hat{}$.

A la hora de implementar el controlador generado *offline*, se destaca de cada región, que se identifica con el subíndice *reg*, los siguientes campos:

- Matrices que definen el poliedro que delimita la región, A_r y b_r. Si se verifica la desigualdad matricial $A_r x \leq b_r$, se confirma que el estado del sistema x se encuentra dentro de esa región. Una vez identificado el número de región, se procede a aplicar la ley de control pertinente. Tal y como se estudió en el capítulo anterior, estas matrices se definen al aplicar y desarrollar las condiciones de optimalidad de Karush Kuhn Tucker.
- Poliedro P definido por la región. Haciendo uso de la herramienta *Polyhedron* de *Matlab*, se obtiene la representación en el espacio del poliedro formado por las matrices que delimitan la región.
- Ganancia de la acción de control, K. Término que multiplica al estado, definido mediante la siguiente expresión:

$$K_{reg} = H^{-1} \left[\tilde{G}^T \left(\tilde{G} H^{-1} \tilde{G}^T \right)^{-1} \left(\tilde{S} + \tilde{G} H^{-1} F \right) - F \right] \tag{3.19}$$

- Término constante de la acción de control, U_0.

$$U_{0reg} = H^{-1} \tilde{G}^T \left(\tilde{G} H^{-1} \tilde{G}^T \right)^{-1} \tilde{W} \tag{3.20}$$

El objeto creado por la clase devuelve un campo de interés para la visualización de la partición del espacio, denominado *Paux*, que contiene los poliedros P_{reg} de todas las regiones.

3.3 Implementación del código generado

La comprensión del desarrollo incluido en [10] ha sido fundamental para la elaboración del código o *solver* propio para el MPC explícito. Por ello, se ha comparado el ejemplo de un doble integrador que se incluía en las transparencias con el resultado obtenido a través de la clase. En particular, se considera el siguiente sistema:

$$x_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_k \tag{3.21}$$

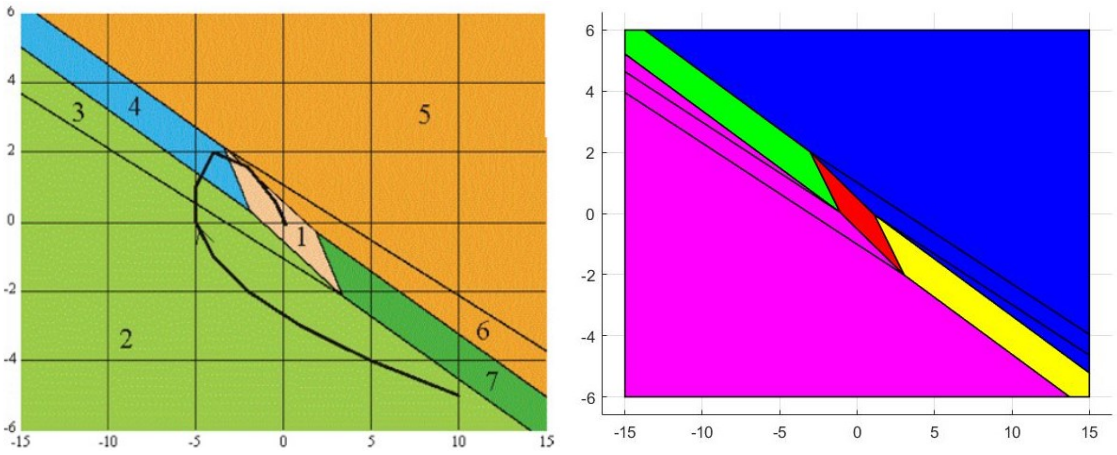
y se asume que la acción de control debe satisfacer$-1 \leq u \leq 1$ para todo instante de tiempo k. Estas restricciones se expresan de forma matricial para poder introducirlas en la clase:

$$\underbrace{\begin{bmatrix} 1 \\ -1 \end{bmatrix}}_{A_u} u \leq \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{b_u} \tag{3.22}$$

Otros parámetros necesarios para la resolución del problema son el horizonte de predicción, con valor $N = 2$, y las matrices de ponderación:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \qquad R = 0.01 \tag{3.23}$$

Introduciendo las matrices que definen el modelo del sistema, las restricciones, matrices de peso y horizonte de predicción, el código generado procede a calcular las distintas regiones en las que se divide el estado, presentando los resultados obtenidos en la Figura 3.1 (b).



(a) Ejemplo diapositivas. **(b)** Resultado clase.

Figura 3.1 Regiones doble integrador.

Se obtienen un total de nueve regiones, pero de los fundamentos del control predictivo se sabe que para cada instante de tiempo se obtiene la siguiente secuencia de la acción de control: $[u_0(x)\ u_1(x)\ \ldots\ u_{N-1}(x)]^T$, aplicando únicamente su primera componente $u_0(x)$, que corresponde a la del instante actual. Como se explicó en el capítulo anterior, viendo las distintas técnicas para simplificar el número de regiones, el método *Optimal Region Merging* une regiones con la misma expresión de la acción de control mediante una unión convexa. Se ha procedido de esta manera, fusionando aquellas regiones con $u_0(x)$ iguales, lo que puede identificarse en la Figura 3.1 (b) mediante regiones separadas por rectas pero representadas con el mismo color. Llegados a este punto, se puede concluir que el código elaborado genera las regiones de forma exitosa, ya que comparándolo con el ejemplo teórico se han obtenido los mismos resultados.

Una vez se ha llevado a cabo el cálculo de regiones de forma *offline*, el segundo paso es la implementación *online* para llevar al estado a la referencia deseada. En este ejemplo se parte de un valor inicial $x_0 = [10\ -5]^T$ con el objetivo de llevar el estado al origen. Para ello, se establece una simulación de 20 instantes de tiempo, evaluando en cada uno en qué región se encuentra el estado y aplicando la ley de control correspondiente.

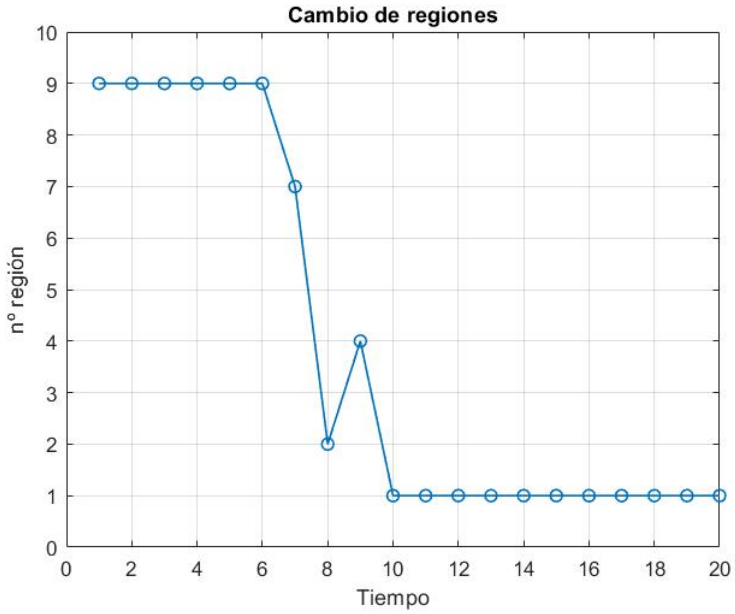


Figura 3.2 Cambio de regiones a lo largo de la simulación.

En la Figura 3.2 se aprecia la evolución del cambio de regiones que experimenta el sistema según dónde se encuentre el estado. En el código generado para este proyecto, se van estudiando todas las distintas combinaciones de restricciones para así definir las regiones, empezando por el caso en el que no hay ninguna restricción activa. Cuando el estado ha alcanzado la referencia deseada, en la que se anula el valor de la entrada porque el sistema está controlado, éste se encontrará en la primera región al no estar la acción de control saturada y no cumplirse ninguna restricción.

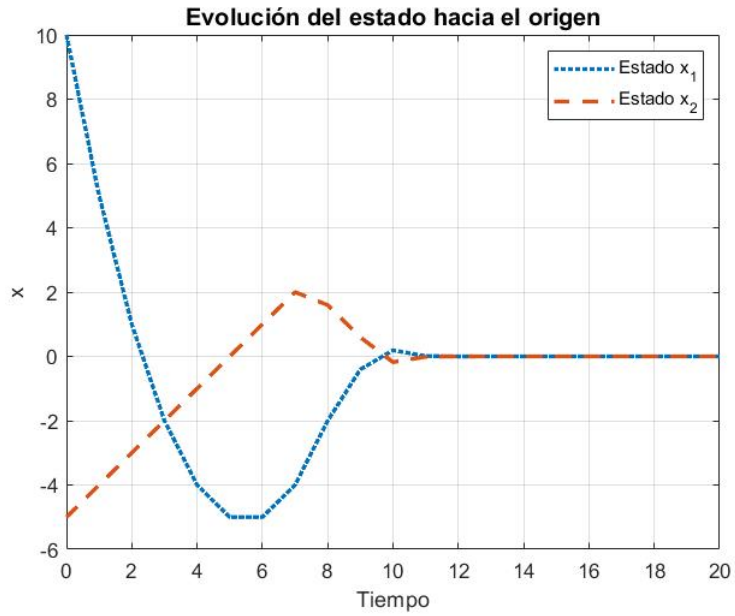


Figura 3.3 Evolución de cada componente del estado.

En la Figura 3.4 se ha representado la evolución de los valores de la acción de control hasta que se consigue llevar al estado a su referencia. Se han representado en línea roja discontinua las bandas superior e inferior que acotan el valor de la acción de control. La actuación se encuentra saturada durante los primeros instantes de tiempo, ya que es el momento en el que el estado se encuentra más alejado de la referencia y la acción a aplicar debe ser mayor.

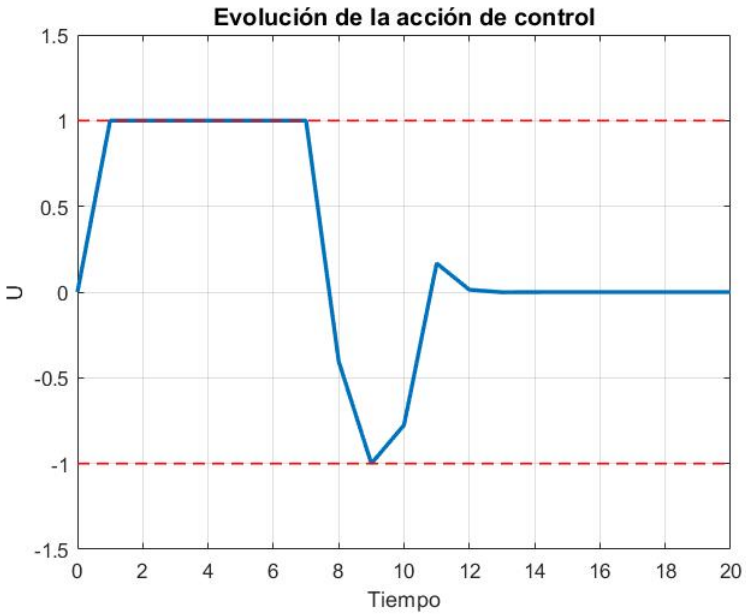


Figura 3.4 Evolución de la entrada.

Por último, en la Figura 3.5 se ha representado la trayectoria que sigue el estado (x_1, x_2) a lo largo de toda la simulación hasta llegar al origen, viendo gráficamente el cambio de regiones que realiza. Al compararla con la trayectoria teórica de la Figura 3.1 (a), se puede comprobar que coinciden. Destacar una particularidad de este ejemplo, y es que al no existir restricciones en el estado, el mapa del explícito se extiende hacia el infinito $\forall x$, por lo tanto todas las regiones excepto la central no estarían acotadas.

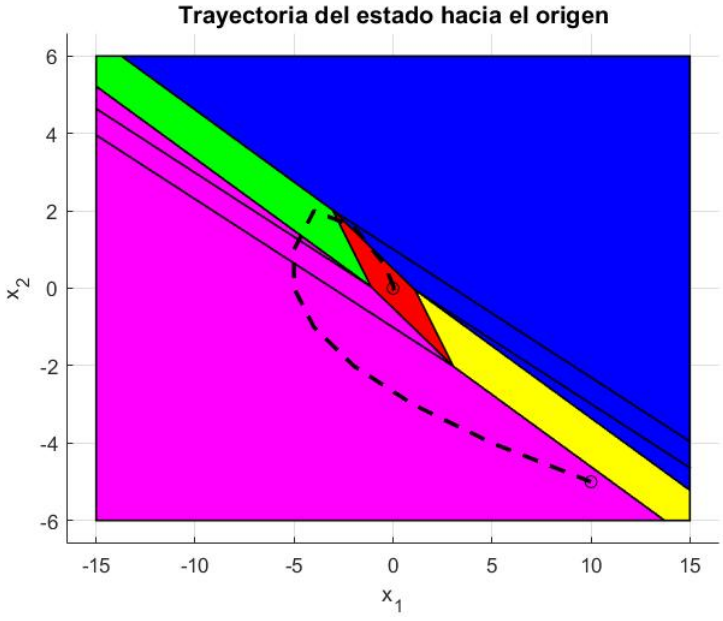


Figura 3.5 Trayectoria seguida por el estado desde x_0 hasta el origen.

4 Aplicaciones MPC explícito

El desarrollo de este capítulo se centra en el estudio de distintas simulaciones que han sido realizadas utilizando el código generado en el capítulo anterior. El objetivo de estas simulaciones, más allá de probar el funcionamiento del código con otros sistemas que tuviesen restricciones en el estado además de en la entrada, ha sido ver cómo varía el mapa del explícito ante distintas situaciones como por ejemplo cambios en el horizonte de predicción. Además de ver el efecto en la partición de regiones, se han podido identificar distintas limitaciones presentes en el código generado al no estar totalmente optimizado.

4.1 Efecto del horizonte de predicción

En primer lugar, se estudia el sistema propuesto en [8], que corresponde con un doble integrador discretizado para un segundo. La peculiaridad que se encuentra en este ejemplo es que tiene todas las características que se pueden introducir como entradas a la clase del explícito para generar sus regiones, ya que cuenta con restricciones en el estado y la entrada, y define la matriz de ponderación P del coste terminal. La dinámica del sistema se modela según la siguiente ecuación es espacios de estados:

$$x_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} u_k \tag{4.1}$$

Las restricciones del sistema son:

$$\begin{aligned} -5 \leq x_k \leq 5 \\ -1 \leq u_k \leq 1 \end{aligned} \tag{4.2}$$

Transformando a forma matricial para poder introducirlas en la clase:

$$\underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}}_{A_x} x \leq \underbrace{\begin{bmatrix} 5 \\ 5 \\ 5 \\ 5 \end{bmatrix}}_{b_x} \qquad \underbrace{\begin{bmatrix} 1 \\ -1 \end{bmatrix}}_{b_u} u \leq \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{b_u} \tag{4.3}$$

Al contar en este ejemplo con la matriz que pondera el coste terminal P, lo cual no se ha visto previamente, se define su influencia en la función de coste:

$$V = \sum_{0}^{N-1} \left(x_{k+1}^T \, Q \, x_{k+1} \, + \, u_k^T \, R \, u_k \right) + x_N^T \, P \, x_N \tag{4.4}$$

El valor de estas matrices de ponderación que se incluyen en la función a optimizar es el siguiente:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1, \quad P = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix} \tag{4.5}$$

La llamada al código de la clase se hace mediante la siguiente expresión, estando todas sus entradas, excepto el horizonte de predicción N, definidas.

$$obj = ExpMPC(A,\ B,\ A_x,\ b_x,\ A_u,\ b_u,\ Q,\ R,\ P,\ N) \tag{4.6}$$

Para este ejemplo, se ha simulado el código generado para el explícito con horizonte de predicción 1 hasta 4, ya que se detuvo la simulación para $N = 5$ al llevar calculadas 41600 regiones y considerar este un número muy elevado. En la Figura 4.1 se han representado los cuatro mapas del estado que se obtienen para los distintos horizontes de predicción simulados. A diferencia del ejemplo expuesto en el capítulo anterior, los mapas tienen todas sus regiones útiles acotadas porque se han introducido restricciones en el estado. Destacar que, aunque las restricciones limiten el valor de x a ± 5, existen regiones que comprenden valores mayores y menores del estado, donde la acción de control correspondiente suele estar saturada para actuar sobre el estado con "más fuerza" y llevarlo hacia una región dónde cumpla las restricciones.

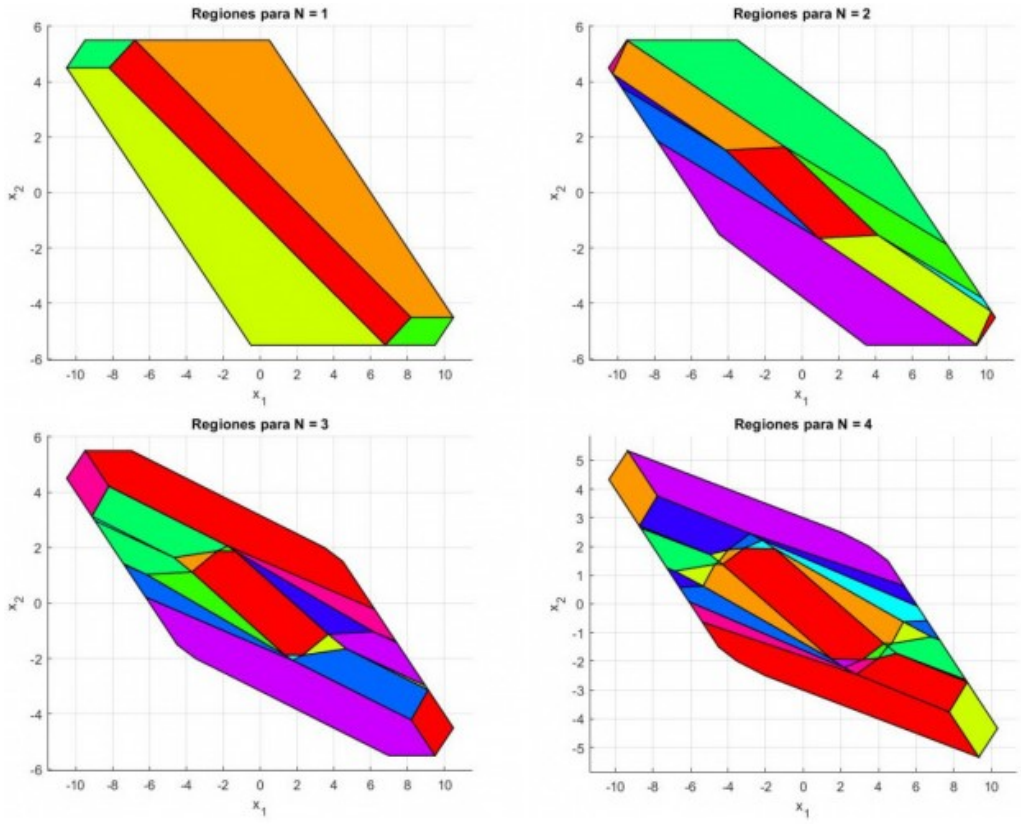


Figura 4.1 Efecto del horizonte de predicción en el mapa del estado.

Las matrices del sistema y las restricciones se extienden para todo N, por lo que a mayor horizonte de control se puede observar en la figura superior el efecto que este tiene sobre el mapa del estado. No solo el mapa se divide en más regiones, sino también se puede observar cómo, a medida que se va aumentando el horizonte, el mapa disminuye su tamaño y se va estrechando. Se han definido las restricciones de la forma: $GU \leq Sx + W$, y se sabe que el número máximo de regiones viene determinado por la combinación de todas las filas de G, por lo que si esta aumenta su tamaño, el número de combinación de posibles regiones a estudiar será mayor.

En la Tabla 4.1 de la siguiente página se ha representado, para cada horizonte de predicción, el número de regiones calculadas por la clase. Como se explicó en el capítulo de generación del código, se van estudiando las distintas combinaciones de restricciones utilizando las filas de la matriz G. Este estudio no está optimizado, ya que se estudian todas las posibles combinaciones aunque estas no se puedan dar en la realidad, por lo que el estudio resulta computacionalmente más pesado de lo que podría ser si se excluyesen aquellas combinaciones de restricciones que no son posibles que estén activas a la vez.

Se emplea el *Multi-Parametric Toolbox* [11] para trabajar con poliedros, facilitando su representación e incluyendo múltiples funciones útiles para el estudio del MPC explícito. Sea un poliedro P, para un estado x dado, con la función *P.contains(x)* se puede obtener directamente si el estado está contenido dentro de ese poliedro, evitando trabajar con desigualdades matriciales para comprobar la pertenencia a una región. Otra función que ha sido de utilidad para construir la tabla de esta página ha sido *P.isEmptySet*, la cual ha permitido calcular el número de regiones vacías que no son útiles para controlar el sistema.

Tabla 4.1 Influencia del horizonte de predicción en el número de regiones.

Horizonte de predicción	Regiones calculadas	Regiones vacías	Regiones útiles
1	7	2	5
2	61	50	11
3	583	562	21
4	5813	5780	33

Se observa por tanto, que la clase calcula un numero de regiones totales muy superior a las que después son útiles para poder controlar el sistema. Aunque esto supone una desventaja por el tiempo de computación empleado *offline*, afecta al control global porque utilizar un horizonte $N = 26$, como se hace en el *paper* de estudio, se hace impensable debido al elevado número de regiones que calcularía, y el tiempo que se emplearía en ello.

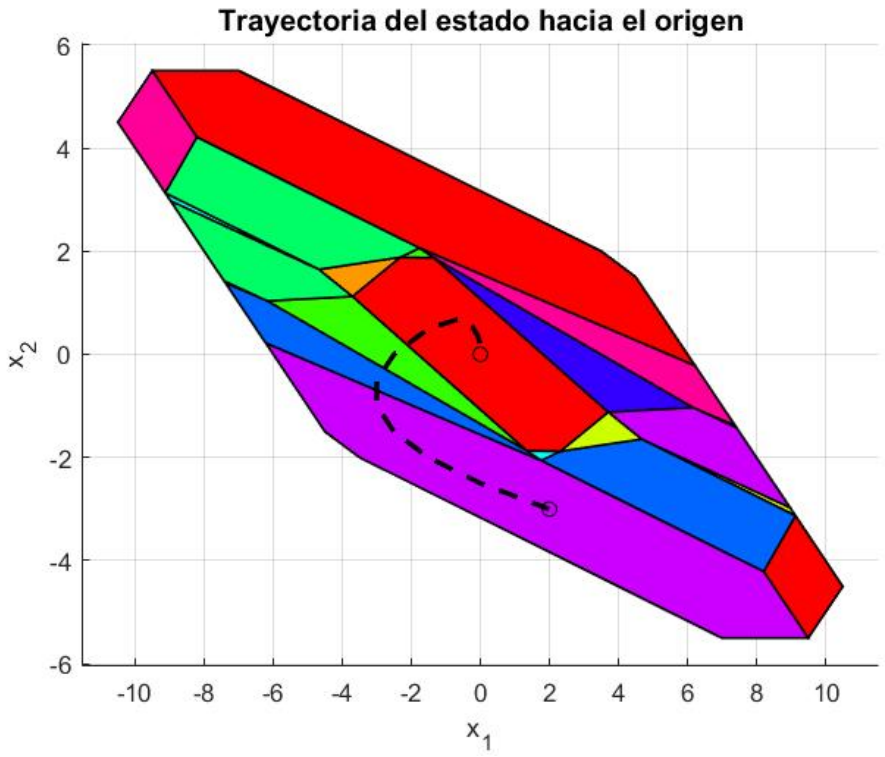


Figura 4.2 Trayectoria del estado partiendo de $[2\ -3]^T$.

Por último, se puede observar en la Figura 4.2 la trayectoria que sigue el estado, partiendo de una posición inicial de $x = [2\ -3]^T$ hasta que llega al origen. La simulación del control de este sistema se ha realizado para el ejemplo de horizonte de predicción $N = 3$, donde existen 21 regiones útiles a las que puede pertenecer el estado.

4.2 Aplicación al modelo del *drone*

4.2.1 Modelado del sistema

En este apartado, se emplea como sistema de estudio el modelo de un *drone*, del que se dispone físicamente para poder realizar pruebas en tiempo real sobre el mismo. Las ecuaciones que gobiernan la dinámica del drone vienen desarrolladas en [12], ya que, como se comentó en la introducción, buena parte de este trabajo viene motivado por la inquietud de poder probar en tiempo real un MPC en este multirrotor comercial. Recordar que, como se explica en el TFG, la ventaja que presenta este modelo comercial es que existe un *software* desarrollado en el entorno de *Matlab-Simulink* por el Instituto Tecnológico de Massachusetts (MIT) que permite diseñar controladores con facilidad, para posteriormente compilar el código y subirlo al *drone* para probar el funcionamiento en tiempo real. Este entorno se ha diseñado de tal forma que permite simular los controladores de forma previa al vuelo, simulando la respuesta de los sensores y toda la dinámica del *drone*.

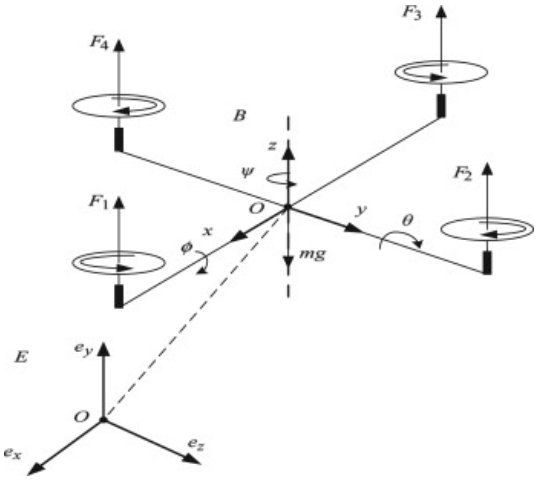


Figura 4.3 Sistema de referencia *drone* [3].

La entrada al sistema tendrá dimensión cuatro, ya que corresponde con el comando que se le entrega a cada motor, determinando el sentido de giro con el signo y velocidad con la magnitud. El vector de estados del sistema tiene doce componentes , que corresponden con las coordenadas en el plano XYZ, los tres ángulos de Euler; roll, pitch y yaw, las velocidades lineales y las velocidades angulares según el plano XYZ, todas las componentes ordenadas como se han nombrado.

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ z \\ \Psi \\ \Theta \\ \Phi \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} \tag{4.7}$$

Una vez definidas las dimensiones del vector de estados y de entradas, es inmediato pensar que, debido a que las matrices del sistema serán de un gran tamaño, el explícito calculará un número elevado de regiones aun cuando se emplee horizonte de predicción igual a uno. Para tratar de simplificar el problema, se tomó la decisión de reducir las restricciones, ya que como solo es de interés el control en posición, no se han introducido restricciones en las velocidades lineales y angulares, quedando las restricciones de la siguiente forma:

- Restricciones en los ejes XYZ

$$-2 \leq \begin{bmatrix} x \\ y \\ z \end{bmatrix} \leq 2 \tag{4.8}$$

- Restricciones en los ángulos de Euler

$$-0.6 \leq \begin{bmatrix} \Psi \\ \Theta \\ \Phi \end{bmatrix} \leq 0.6 \tag{4.9}$$

- Restricciones en la entrada

$$-300 \leq u_i \leq 300 \quad \forall i = 1...4 \tag{4.10}$$

Las matrices de peso del problema de optimización se obtienen mediante la Ley de Bryson, que define estas matrices según una serie de pesos que se le asigna a cada estado, estableciendo así la importancia relativa del mismo a la hora de controlar el sistema. Para el cálculo detallado de Q y R ver [12], donde se detalla el ajuste de los pesos α_i.

$$Q = \begin{bmatrix} \frac{{\alpha_1}^2}{{x_{1\,max}}^2} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{{\alpha_{n_x}}^2}{{x_{n_x\,max}}^2} \end{bmatrix} \quad \text{con } n_x = 12 \tag{4.11}$$

$$R = \begin{bmatrix} \frac{1}{{u_{1\,max}}^2} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{{u_{n_u\,max}}^2} \end{bmatrix} \quad \text{con } n_u = 4 \tag{4.12}$$

Para el modelo del sistema, el *Toolbox* desarrollado por el MIT contiene un diagrama principal en *Simulink*, donde se representa y simula toda la dinámica del sistema, que permite obtener el modelo lineal del sistema. El resultado del análisis lineal proporciona las matrices del sistema continuo:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{4.13}$$

Para poder emplear este controlador, las matrices del sistema continuo han de discretizarse, para lo cual se emplea un periodo de discretización de cuarenta milisegundos. La expresión del modelo en espacios de estados deja de depender del tiempo t, y pasa a depender del instante k de simulación:

$$x_{k+1} = A_d x_k + B_d u_k \tag{4.14}$$

Se presentan a continuación las matrices del modelo discreto del sistema:

$$Ad = \begin{bmatrix} 1 & 0 & 0 & 0 & -0.008 & 0 & 0.04 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0.008 & 0 & 0.04 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0.04 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.04 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0.001 & 0 & 0 & 0 & 0.039 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -0.001 & 0 & 0.038 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0.392 & 0 & 0.997 & 0 & 0 & 0 & -0.002 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.392 & 0 & 0.997 & 0 & 0.002 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.008 & 0 & -0.039 & 0 & 0.916 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0.006 & 0 & 0.029 & 0 & 0 & 0 & 0.937 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.995 \end{bmatrix} \tag{4.15}$$

$$Bd = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0.0002 & -0.0002 & -0.0002 & 0.0002 \\ 0.0003 & 0.0003 & -0.0003 & -0.0003 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -0.0004 & 0.0004 & -0.0004 & 0.0004 \\ 0.0161 & 0.0161 & -0.0161 & -0.0161 \\ 0.0121 & -0.0121 & -0.0121 & 0.0121 \\ -0.0005 & -0.0005 & -0.0005 & -0.0005 \end{bmatrix} \tag{4.16}$$

4.2.2 Control MPC explícito

Al ejecutar el código generado para el MPC explícito introduciendo todas las matrices del sistema y las restricciones anteriores con horizonte de predicción $N = 1$, el resultado es un mapa del estado dividido en 3089 regiones. Como es lógico, el característico mapa que devuelve el explícito tiene la dimensión de su estado, por lo que en este caso en el que se trabaja con 12 dimensiones no es posible representar el mapa. Cuando se implementó el código del explícito en el entorno desarrollado por el MIT para controlar el *drone*, la simulación fue satisfactoria y las regiones generadas permitían seguir la trayectoria de referencia especificada. Destacar que, por intentar simplificar el problema de control, el buen funcionamiento del control en altura (eje Z) fue el objetivo principal.

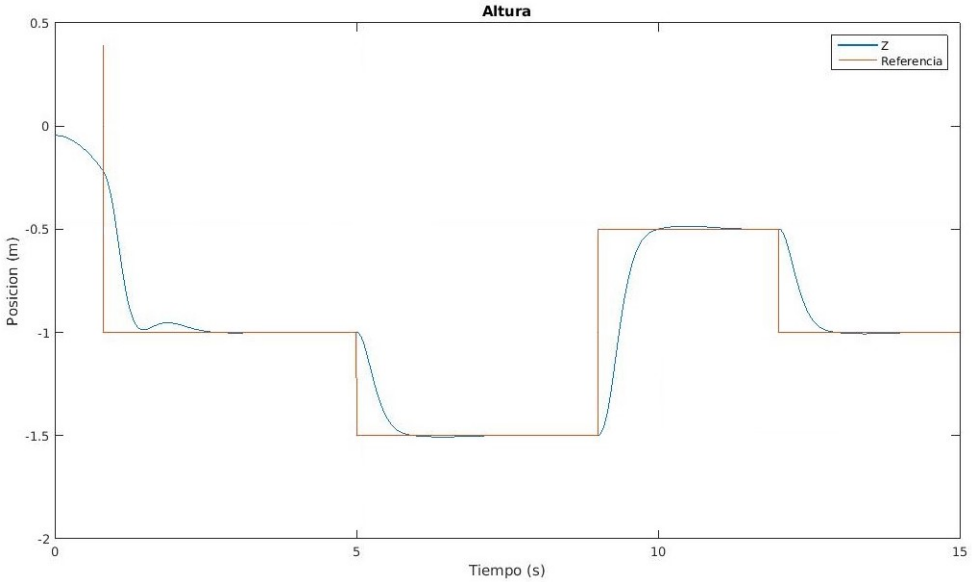


Figura 4.4 Resultado de la simulación del explícito en el *toolbox* del MIT.

Llegados a este punto, el siguiente objetivo era averiguar cómo conseguir subir al *drone* todas las matrices que caracterizaban a las 3089 regiones del explícito para poder implementarlo *online*, ya que en *Matlab* se guardan todos los datos en una estructura y esto no puede compilarse. Los pasos necesarios para este propósito fueron los siguientes:

1. Recorrer la estructura solución, con longitud el número de regiones, guardando en ficheros distintos las matrices que definen la región; A_r y b_r, y las que definen la acción de control; K y U_0. Como resultado de este paso, se tendrán cuatro ficheros, cada uno correspondiente con una matriz característica de la región, con los datos de todas las matrices de las regiones. Resaltar que todas las matrices correspondientes a una misma propiedad no tienen la misma dimensión, por lo que para facilitar la lectura de los ficheros se rellenó con ceros hasta completar la dimensión máxima para cada caso.

2. Conocidas las dimensiones de cada matriz, ya que se han rellenado con ceros para que tengan la dimensión máxima, se genera un código en C que lea los cuatro ficheros y vaya guardando en una estructura las distintas matrices para cada región.

El primer paso fue inmediato, pero al llegar al segundo, el compilador que se estaba empleando no permitía crear estructuras con tantos campos ni tan pesadas, ya que la matriz de mayor dimensión era 20x12. Para intentar atajar esta limitación, se ejecutó una simulación estudiando cuáles eran las regiones del explícito que se empleaban para controlar el sistema, para intentar compilar las matrices de un número de regiones reducido. Como resultado de este estudio, y para nuestra sorpresa, se observó que el sistema estaba siendo controlado únicamente con la primera región. Al estudiar algunas de las otras regiones, se pudo ver que los poliedros no estaban acotados, además de haber muchas vacías, ya que en la simplificación realizada seis componentes del estado no se estaban acotando con las restricciones. La primera región se caracterizaba por tener un $U_0 = 0$, por lo que la acción de control se reducía a aplicar una ganancia, cuyos valores eran similares a los de la ganancia que se aplica al diseñar un controlador LQR para el *drone*.

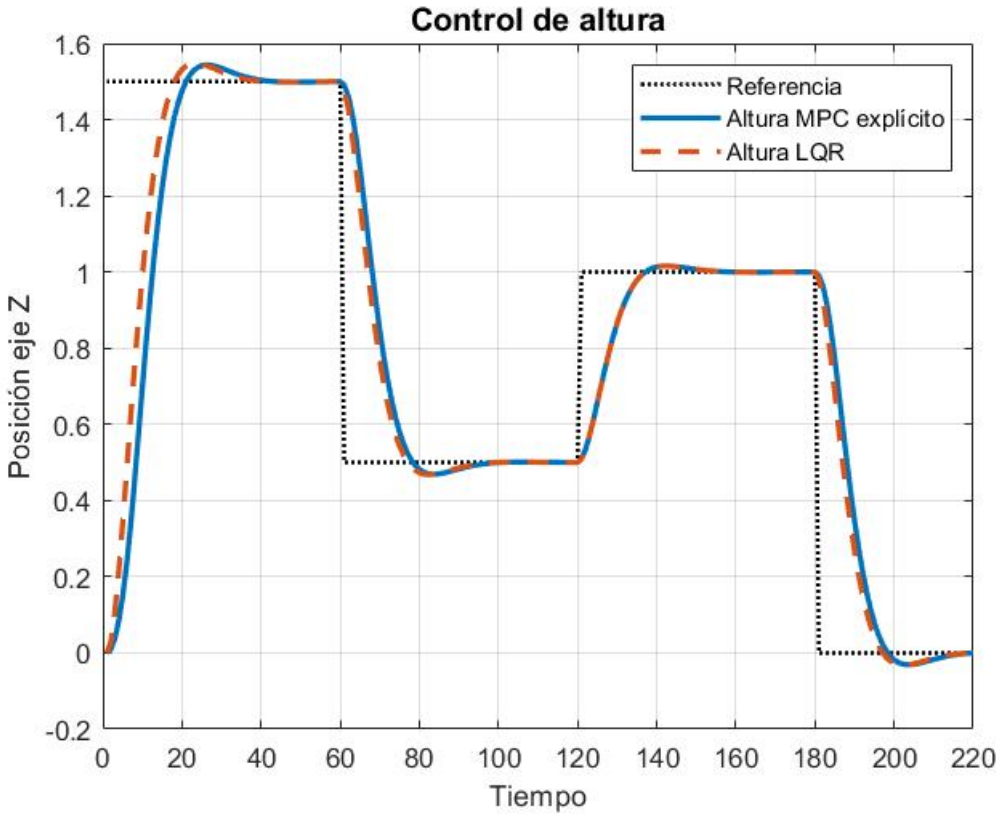


Figura 4.5 Control en altura con MPC explícito y LQR.

Se concluye por tanto, que la simplificación realizada a la hora de reducir las restricciones para tratar de disminuir el número de regiones del explícito da como resultado un mapa del estado aun así con muchas regiones, con el inconveniente de de que la mayoría de ellas se constituyen por poliedros no acotado. Como se ha mencionado anteriormente, la primera región calculada por la clase corresponde con aquella en la que no hay ninguna restricción activa, por lo que el sistema se mueve durante toda la simulación dentro de las restricciones, al ser la única región con la que se controla el sistema. Viendo la Figura 4.5 se confirma que al estar controlando con una única región en la que la ley de control se reduce a aplicar una ganancia sobre el estado, no se consigue ninguna mejora considerable para este ejemplo con respecto al control LQR. Entonces, desafortunadamente, el objetivo de probar este tipo de controlador en tiempo real para solventar las limitaciones encontradas durante el desarrollo de mi TFG no ha podido ser cubierto.

4.3 Estudio de las perturbaciones

Hasta el momento, los sistemas de estudio que se han simulado han sido de la forma $x_{k+1} = Ax_k + Bu_k$, por lo que no incluían la matriz D que modela el efecto de las perturbaciones w_k sobre el estado x_{k+1}:

$$x_{k+1} = Ax_k + Bu_k + Dw_k \tag{4.17}$$

En este apartado se va a modelar el problema del MPC explícito parametrizando el valor de la perturbación en conjunto con el estado, por lo que se construye un vector de estados ampliado:

$$\hat{x}_k = \begin{bmatrix} x_k \\ w_k \end{bmatrix} \tag{4.18}$$

De esta forma, el mapa del estado del explícito aumentaría en la dimensión de w_k, y la condición de pertenencia a la región así como la acción de control a aplicar será en función de $\hat{x}$, vector de estados ampliado o vector de parámetros. El objetivo de esta notación será tratar de extender esta idea al control distribuido, en la que se modelará el acoplamiento entre subsistemas como una perturbación, pero esto se verá con más detalle en el siguiente capítulo.

4.3.1 Formulación del problema de optimización

Al cambiar la manera de formular el modelo de espacio de estados del sistema, va a variar la forma en la que está expresado el problema QP. Extendiendo la ecuación 4.17 para todo el horizonte de predicción N, se puede expresar mediante la notación matricial que se introdujo en el capítulo anterior, pero añadiendo los términos que corresponden a la perturbación.

$$X = G_x x + G_u U + G_w W \tag{4.19}$$

Se definen las nuevas matrices que aparecen en la formulación:

$$W = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{N-2} \\ w_{N-1} \end{bmatrix} \in \mathbb{R}^{N \cdot n_w \text{ x } 1} \quad G_w = \begin{bmatrix} D & 0 & 0 & 0 & 0 \\ AD & D & 0 & 0 & 0 \\ A^2D & AD & D & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A^{N-1}D & A^{N-2}D & \dots & AD & D \end{bmatrix} \in \mathbb{R}^{N \cdot n_x \text{ x } N \cdot n_w} \tag{4.20}$$

Extendiendo también la función objetivo $\forall N$: $V = X^T \hat{Q} X + U^T \hat{R} U$, y sustituyendo la expresión de X, se desarollo hasta llegar a:

$$\begin{aligned} V &= 2\left((G_x x)^T \hat{Q} G_u + (G_w W)^T \hat{Q} G_u\right) U + U^T \left(G_u^T \hat{Q} G_u + \hat{R}\right) U \\ V &= 2\left(x^T \underbrace{G_x^T \hat{Q} G_u}_{F^T} + W^T \underbrace{G_w^T \hat{Q} G_u}_{T^T}\right) U + U^T \left(\underbrace{G_u^T \hat{Q} G_u + \hat{R}}_{H}\right) U \end{aligned} \tag{4.21}$$

Viendo la estructura de la función de costes, es inmediato pensar que la formulación del problema QP cambia, ya que se ha identificado una nueva matriz T que multiplica al efecto de las perturbaciones. Este efecto también es importante tenerlo en cuenta a la hora de formular las restricciones, ya que, aunque no se introduzcan restricciones en W como tal, W afecta a X y por lo tanto la expresión $GU \leq Sx + W$ va a tener un término que dependa de W. Una vez se tienen las restricciones expresadas de forma matricial, se pueden extender para todo N utilizando los vectores X y U como se vio en capítulos anteriores.

$$\begin{aligned} \hat{A}_x X &\leq \hat{b}_x \\ \hat{A}_u U &\leq \hat{b}_u \end{aligned} \tag{4.22}$$

Para expresar estas ecuaciones en función de la entrada, se sustituye la expresión de X en las restricciones del estado.

$$\hat{A}_x X \leq \hat{b}_x \rightarrow \hat{A}_x (G_x x + G_u U + G_w W) \leq \hat{b}_x \rightarrow \hat{A}_x G_u U \leq \hat{b}_x - \hat{A}_x G_x x - \hat{A}_x G_w W \tag{4.23}$$

Separando y agrupando aquellos términos que dependen de U, x_0 y W, se obtiene la siguiente desigualdad matricial que define las restricciones del problema de programación cuadrática:

$$\underbrace{\begin{bmatrix} \hat{A}_x G_u \\ \hat{A}_u \end{bmatrix}}_{G} U \leq \underbrace{\begin{bmatrix} \hat{b}_x \\ \hat{b}_u \end{bmatrix}}_{W} + \underbrace{\begin{bmatrix} -\hat{A}_x G_x \\ 0 \end{bmatrix}}_{S} x + \underbrace{\begin{bmatrix} -\hat{A}_x G_w \\ 0 \end{bmatrix}}_{J} W \tag{4.24}$$

Finalmente, se puede formular el problema QP como:

$$\begin{aligned} &\min_{U} \ \frac{1}{2} U^T H U + \left(x^T F^T + W^T T^T\right) U \\ &\text{s.a } GU \leq W + [S\ J] \begin{bmatrix} x \\ W \end{bmatrix} \end{aligned} \tag{4.25}$$

4.3.2 Desarrollo de las condiciones KKT

Como la formulación del problema QP ha cambiado, hay realizar de nuevo el desarrollo de las condiciones de optimalidad para encontrar la desigualdad matricial que define las regiones, así como la ley de control de cada una de ellas, en función del estado x y del valor de la perturbación w. A continuación, partiendo de la ecuación 4.25 se desarrollan las condiciones KKT:

$$\mathscr{L} = \frac{1}{2} U^T H U + \left(x^T F^T + U^T T^T\right) U + \mu^T \left(GU - W - [S\ J] \begin{bmatrix} x \\ W \end{bmatrix} \right) \tag{4.26}$$

$$\nabla_U \mathscr{L} = HU + Fx + TW + G^T \mu = 0 \tag{4.27}$$

$$\tilde{G}_i U - \tilde{W}_i - \tilde{S}_i x - \tilde{J}_i W = 0 \tag{4.28}$$

$$\hat{G}_i U \leq \hat{W}_i + \hat{S}_i x + \hat{J}_i W \tag{4.29}$$

$$\begin{cases} \tilde{\mu}_i \geq 0 \text{ para las restricciones activas} \\ \hat{\mu}_i = 0 \text{ para las restricciones inactivas} \end{cases} \tag{4.30}$$

De la condición del gradiente del lagrangiano igual a cero se puede despejar:

$$U(x,U) = -H_1^{-1} \left(Fx + TW + \tilde{G}^T \tilde{\mu}\right) \tag{4.31}$$

Expresión de $\tilde{\mu}$, multiplicador de Lagrange, utilizando $\tilde{P} = \left(\tilde{G} H^{-1} \tilde{G}^T\right)^{-1}$

$$\tilde{\mu}(x,W) = -\tilde{P} \left(\tilde{W} + \tilde{G} H^{-1} F x + \tilde{S} x + \tilde{G} H^{-1} T W + \tilde{J} W\right) \tag{4.32}$$

$$\tilde{\mu}(x,W) = -\tilde{P}\tilde{W} - \begin{bmatrix} \tilde{P}\left(\tilde{G} H^{-1} F + \tilde{S}\right) & \tilde{P}\left(\tilde{G} H^{-1} T + \tilde{J}\right) \end{bmatrix} \begin{bmatrix} x \\ W \end{bmatrix} \tag{4.33}$$

Sustituyendo la expresión de $\tilde{\mu}$ en la primera definición de $U(x,U)$ se obtiene la expresión que define la ley de control de cada región en función del estado y el valor de la perturbación. Empleando $\tilde{M} = H^{-1} \tilde{G}^T \left(\tilde{G} H^{-1} \tilde{G}^T\right)^{-1}$ se obtiene:

$$U(x,W) = \tilde{M}\tilde{W} + \tilde{M}\tilde{S} x + \left(\tilde{M}\tilde{G} H^{-1} - H^{-1}\right) F x + \tilde{M}\tilde{J} W + \left(\tilde{M}\tilde{G} H^{-1} - H^{-1}\right) T W \tag{4.34}$$

Puede reescribirse como:

$$U(x,W) = \tilde{M}\tilde{W} + \begin{bmatrix} \tilde{M}\tilde{S} + \left(\tilde{M}\tilde{G}H^{-1} - H^{-1}\right)F & \tilde{M}\tilde{J} + \left(\tilde{M}\tilde{G}H^{-1} - H^{-1}\right)T \end{bmatrix} \begin{bmatrix} x \\ W \end{bmatrix} \tag{4.35}$$

Cálculo de regiones:

- Condición de restricciones inactivas: $\hat{G}U \leq \hat{W} + \hat{S}x + \hat{J}U$ Sustituyendo la expresión 4.35 de U

$$\underbrace{\begin{bmatrix} \hat{G}\tilde{M}\tilde{S} + \hat{G}\left(\tilde{M}\tilde{G}H^{-1} - H^{-1}\right)F - \hat{S} & \hat{G}\tilde{M}\tilde{J} + \hat{G}\left(\tilde{M}\tilde{G}H^{-1} - H^{-1}\right)T - \hat{J} \end{bmatrix}}_{G_1} \begin{bmatrix} x \\ W \end{bmatrix} \leq \underbrace{\hat{W} - \hat{G}\tilde{M}\tilde{W}}_{g_1} \tag{4.36}$$

- Condición de positividad multiplicadores de Lagrange en de las restricciones activas: $\tilde{\mu} \geq 0$

$$\underbrace{\begin{bmatrix} \tilde{P}\tilde{S} + \tilde{P}\tilde{G}H^{-1}F & \tilde{P}\tilde{G}H^{-1}T + \tilde{P}\hat{J} \end{bmatrix}}_{G_2} \begin{bmatrix} x \\ W \end{bmatrix} \leq \underbrace{-\tilde{P}\tilde{W}}_{g_2} \tag{4.37}$$

Agrupando, las regiones quedan definidas mediante:

$$\begin{bmatrix} G_1 \\ G_2 \end{bmatrix} \begin{bmatrix} x \\ W \end{bmatrix} \leq \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \tag{4.38}$$

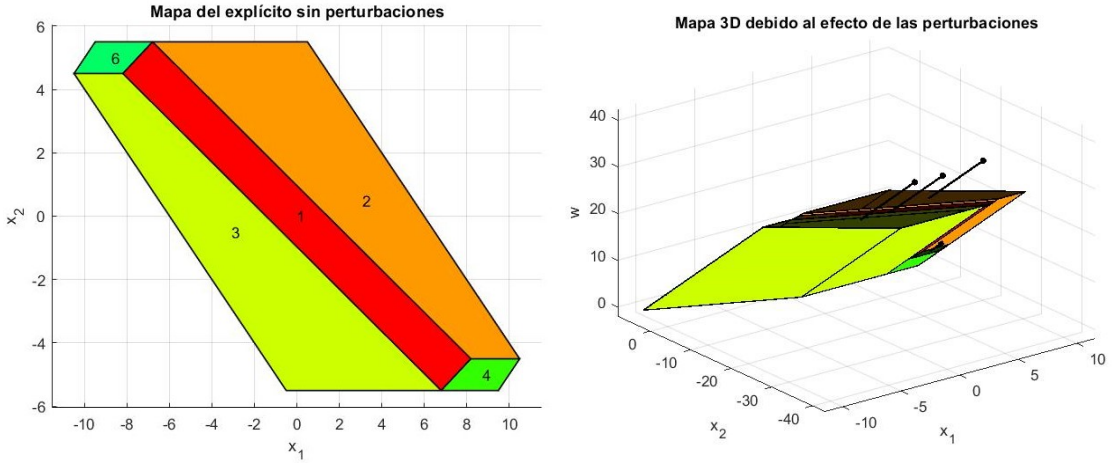
4.3.3 Efecto de W en el mapa del explícito

Para estudiar el efecto de las perturbaciones, se va a utilizar el ejemplo del apartado 4.1 de este capítulo, basado en [8], introduciendo una matriz D compuesta de unos para poder ver el efecto que tendría sobre el mapa del estado. Formulando el sistema como:

$$x_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} u_k + \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{D} w_k \,, \tag{4.39}$$

cuyas restricciones en el estado y la entrada son: $-5 \leq x_k \leq 5$ y $-1 \leq u_k \leq 1$, y con matrices de peso:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1, \quad P = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix} \tag{4.40}$$



(a) Sin considerar perturbaciones. **(b)** Considerando perturbaciones.

Figura 4.6 Mapa del estado para $N = 1$.

Se ha modificado el código generado en el apartado anterior para tener en cuenta el efecto de la matriz D, que se traduce en una nueva formulación del problema QP como se ha visto anteriormente. Como se ha parametrizado el controlador para el estado y las perturbaciones, el mapa aumenta tantas dimensiones como las que tengan las perturbaciones. La dimensión de la perturbación es la misma que la de la acción de control, por lo que en este caso solo aumentará el mapa en una dimensión. Para este ejemplo se puede representar el resultado obtenido en un mapa 3D, ya que el estado pertenece a $\mathbb{R}^2$ y se ha simulado para horizonte de predicción 1. En el caso de haber utilizado un horizonte mayor, la dimensión del vector de perturbaciones se extendería $\forall N$, por lo que no se podría representar en el espacio.

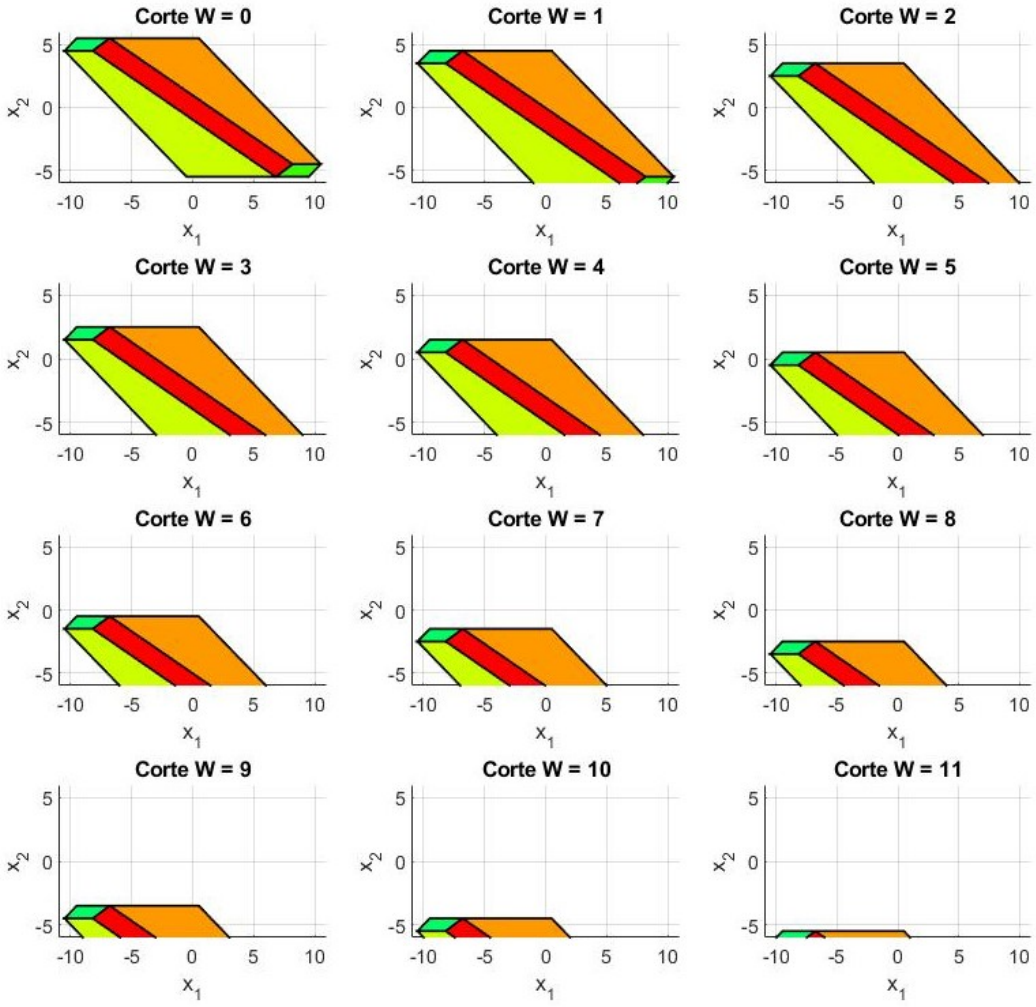


Figura 4.7 Cortes del mapa 3D para distintos valores de la perturbación.

Para poder estudiar el resultado con mayor claridad se han realizado doce cortes del mapa de la Figura 4.6 (b) para distintos valores del eje Z, que corresponde con el valor de la perturbación. Empezando con una perturbación nula, es lógico pensar que el resultado obtenido tiene que ser el mismo que con el código original en el que no se tienen en cuenta el efecto de las perturbaciones. Se puede ver en la Figura 4.6 (b) que el mapa 2D no se extiende de manera vertical en el eje Z, sino que tiene una inclinación determinada. Como el valor de W no se encuentra acotado, parte de cero hasta el infinito, pero no cualquier valor de la perturbación es admisible para el sistema. En la Figura 4.7 se han representado los distintos cortes pero manteniendo fijos los intervalos del estado, ya que ambos están acotados entre ± 5, aunque se ha representado un intervalo mayor que corresponde al que devuelve el mapa original. A medida que la perturbación crece, la región del estado donde se puede controlar se va haciendo mayor, y para valores de $W > 12$ sería imposible controlarlo. Cuanto mayor es el valor de la perturbación más afecta al mapa del explícito, ya que la combinación de dicha perturbación con algunos valores del estado se vuelve inadmisible, y no existen regiones que contengan esta combinación, por lo que no habrá una acción de control asociada para poder controlar el sistema en ese punto.

5 Algoritmos MPC explícito distribuido

5.1 MPC distribuido

Uno de los principales inconvenientes que se presentan cuando se quiere implementar un controlador de tipo MPC de forma centralizada [13], es que no puede ser aplicado a grandes sistemas como redes de potencia, agua o problemas de tráfico. Existen limitaciones en cuanto a la gran demanda computacional que supone, y a la obtención de un modelo fiable del sistema completo, que dificultan su aplicación en sistemas de gran escala. Se conoce como subsistema a una parte del sistema físico, y "agente" a la entidad lo controla.

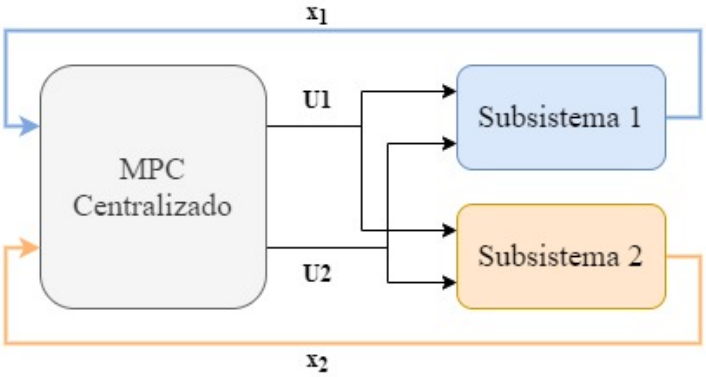


Figura 5.1 Arquitectura MPC centralizado.

Para sistemas basados en redes o de gran escala, el problema puede descomponerse en distintos subsistemas cuyos "agentes" no se comunican entre sí, controlados por tanto de forma descentralizada. Esto supone una mejora en cuanto a la demanda computacional, pero el desempeño global del sistema no es óptimo al no estar compartiendo información e ir de forma independiente.

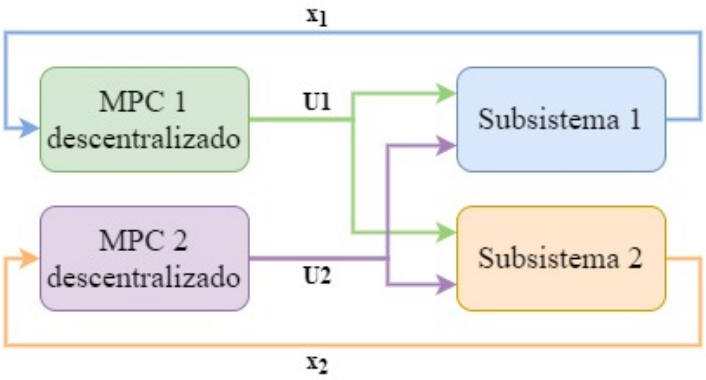


Figura 5.2 Arquitectura MPC descentralizado.

Del objetivo de mejorar el rendimiento global del sistema surge el enfoque distribuido, en el que el sistema global se divide en distintos subsistemas acoplados entre sí, que a su vez están controlados por agentes locales que pueden intercambiar información. Este desempeño dependerá en la cantidad de interacciones que existan entre los agentes, ya que se ha de limitar a un mínimo la cantidad de información que comparten para que no existan problemas de carga computacional, pero también es necesaria la coordinación para obtener un buen rendimiento. La estrategia MPC distribuida divide el problema global en "subproblemas" reducidos del tipo MPC, que se encuentran acoplados entre ellos.

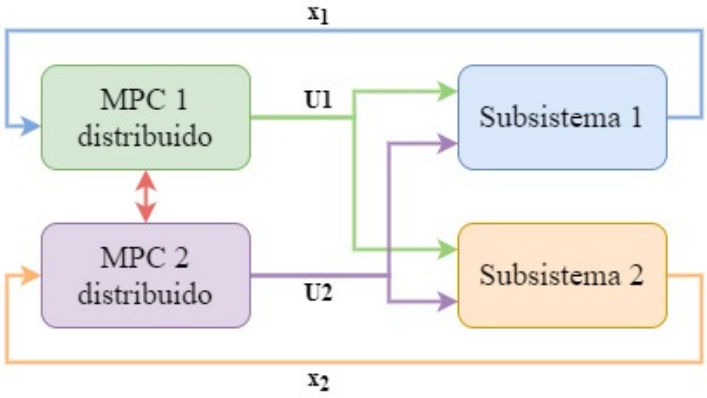


Figura 5.3 Arquitectura MPC distribuido.

Para observar las diferencias entre la implementación de los distintos esquemas, se propone un sistema con número de entradas $n_u = 2$ y dimensión del estado $n_x = 2$:

$$x(k+1) = \underbrace{\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}}_{A} x(k) + \underbrace{\begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}}_{B} u(k) \tag{5.1}$$

- **Esquema centralizado**

  En este caso se cuenta con un sistema pequeño, por lo que no habría problema en resolverlo de manera centralizada, que consiste en implementar un control MPC para el modelo de la ecuación 5.1, con las matrices A y B señaladas.

- **Esquema descentralizado**

  Como se ha mencionado, el enfoque descentralizado descompone el sistema global en distintos subsistemas pero sin que exista interacción entre los controladores locales. Para este caso, se construyen dos controladores MPC que obtengan las acciones óptimas U_1 y U_2 anulando el acoplamiento que se recoge en los términos A_{ij}, B_{ij} para $i \neq j$. El controlador asociado a U_i calculará la acción óptima del subsistema local:

$$x_i(k+1) \; = \; A_{ii}x_i(k) \; + \; B_{ii}u_i(k), \qquad \forall i = 1,2 \tag{5.2}$$

- **Esquema distribuido**

  Para el caso anterior, como cada subsistema decide y actúa por cuenta propia sin tener en cuenta al resto, para casos con mucho acoplamiento, puede resultar en que no se consigue controlar el sistema global entorno a la referencia deseada, ya que se hace necesaria la interacción entre subsistemas. Por último, para el MPC distribuido o DMPC (del inglés *Distributed Model Predictive Control*) se divide en subsistemas teniendo en cuenta el acoplamiento, y existirá un proceso de negociación entre los agentes para llegar a la acción de control óptima.

$$x_i(k+1) \; = \; A_{ii}x_i(k) \; + \; A_{ij}x_j(k) \; + \; B_{ii}u_i(k) \; + \; B_{ij}u_j(k), \qquad \forall i,j = 1,2 \text{ con } j \neq i \tag{5.3}$$

A continuación, se proponen varias metodologías dentro del enfoque de MPC distribuido en las que se ha tratado de adaptar la formulación del MPC explícito a estas casuísticas, tratando de aprovechar su característica *offline* con objetivos como obtener el mismo resultado que para un MPC tradicional o implícito con un menor coste computacional *online*. Se proponen tres métodos distintos, para los cuales se han tenido que reformular las condiciones KKT, ya que los distintos enfoques modifican la expresión del problema de optimización. Es importante modificar la formulación del problema para expresarlo como en [10], ya que es la forma en la que debe estar el sistema para desarrollar las condiciones de optimalidad y obtener la expresión explícita de las regiones y la acción de control.

5.2 Dual decomposition

Dentro de las estrategias de control distribuido, se pueden encontrar enfoques no cooperativos y cooperativos, en este segundo grupo se encuentra el algoritmo de descomposición dual [14, 15, 16], que se estudiará en este apartado. Con este método distribuido, se puede obtener la solución del MPC centralizado a través de un proceso iterativo que se ayuda de unas variables auxiliares conocidas como multiplicadores de Lagrange; empleadas para satisfacer las restricciones del problema. Dado un sistema global que puede ser dividido en distintos subsistemas o agentes, asumimos que cada agente i optimiza su vector u_i pero también las entradas de los subsistemas vecinos que afecten a la dinámica del agente de estudio i. Al contar con subsistemas acoplados, se tiene como consecuencia un problema de optimización con variables compartidas, donde el proceso iterativo utiliza los multiplicadores de Lagrange para coordinar estas variables.

Se procede a estudiar el caso de un sistema con dos agentes acoplados por la entrada, en el que ambos optimizan la variable $U = [u_1\ u_2]^T$. La función objetivo global se puede expresar como:

$$V(U) = V_1(U) + V_2(U) \rightarrow \min_U V(U) \tag{5.4}$$

El procedimiento consiste en crear dos versiones locales de la variable compartida U, una por cada agente: U_1 y U_2, forzando a que ambas variables locales tengan el mismo valor $U_1 = U_2$. Es importante no confundir, por ejemplo, U_1 con la componente u_1 de U, ya que las nuevas versiones de la variable compartida son variables de optimización de cada subsistema, que no coinciden con las respectivas entradas a cada uno. Al estar acoplados por la entrada, cada variable de optimización contiene las acciones de control de ambos agentes. El objetivo de crear estas variables locales es tratar de desacoplar los problemas de optimización, y que cada agente optimice una función objetivo propia atendiendo a la restricción que fuerza que U_1 y U_2 sean iguales.

$$\begin{aligned} &\min_{U_1,\,U_2} \ V_1(U_1) + V_2(U_2) \\ &\text{s.a} \quad U_1 = U_2 \end{aligned} \tag{5.5}$$

Para conseguir que ambos agentes den el mismo resultado se introduce un multiplicador de Lagrange λ para caracterizar esa restricción, quedando el problema de optimización como:

$$\max_{\lambda} \min_{U_1,U_2} \ V_1(U_1) + V_2(U_2) + \lambda^T (U_1 - U_2) \tag{5.6}$$

Por lo tanto, se consigue desacoplar el problema para tratarlo de forma distribuida, donde cada subsistema optimiza una función de coste local, definidas mediante:

$$\text{Subsistema 1: } \min_{U_1} V_1(U_1) + \lambda U_1 \qquad \text{Subsistema 2: } \min_{U_2} V_2(U_2) - \lambda U_2 \tag{5.7}$$

Una vez se resuelven los problemas locales, se compara el valor de ambas variables de optimización, ya que el objetivo es que ambos resultados sean iguales. Se calculará la norma de la diferencia entre los valores de U_1 y U_2, estableciendo un umbral mínimo η de la misma, a partir del cual se considera que los resultados son suficientemente iguales y la negociación interna se detiene. En cada paso de esta negociación iterativa, el valor del multiplicador de Lagrange que se emplea para imponer la restricción de igualdad ha de actualizarse, en función de un paso γ e incrementando su valor:

$$\lambda(k+1) = \lambda(k) + \gamma(U_1 - U_2) \tag{5.8}$$

5.2.1 Descomposición dual adaptada a MPC explícito

Siguiendo con la notación anterior, se plantea a continuación la formulación del problema para poder adaptar la clase del explícito al algoritmo de descomposición dual. Para ello, dado un problema global que puede descomponerse en $\mathscr{N}$ subsistemas acoplados por la entrada, la función de optimización local de cada subsistema puede tratarse como el siguiente problema de optimización cuadrática:

$$\begin{aligned} &\min_{U_i} \frac{1}{2} U_i{}^T H U_i + x_i{}^T F^T U_i + \lambda^T U_i \\ &\text{s.a } G U_i \leq W + S x_i \end{aligned} \tag{5.9}$$

Siendo U la variable compartida entre subsistemas, U_i la variable local de optimización, y x_i el estado correspondiente al agente i. Introducir el multiplicador de Lagrange en la función de optimización se traduce en $Fx \leftarrow Fx + \lambda$. De esta manera, el controlador MPC explícito estará parametrizado en función del estado y el multiplicador de Lagrange λ; que caracteriza la restricción que fuerza a que las versiones locales de la variable compartida coincidan. Se desarrollan a continuación las condiciones de optimalidad KKT para el problema 5.9.

$$\mathscr{L} = \frac{1}{2} U_i{}^T H U_i + x_i{}^T F^T U_i + \lambda^T U_i + \mu^T (G U_i - W - S x_i) \tag{5.10}$$

$$\nabla_U \mathscr{L} = 0 \rightarrow H U_i + F x_i + \lambda + G^T \mu = 0 \tag{5.11}$$

$$\tilde{G}_n U_i - \tilde{W}_n - \tilde{S}_n x_i = 0 \tag{5.12}$$

$$\hat{G}_n U_i \leq \hat{W}_n + \hat{S}_n x_i \tag{5.13}$$

$$\mu_n (G_n U_i - W_n - S_n x_i) = 0 \quad \begin{cases} \tilde{\mu}_n \geq 0 \text{ para las restricciones activas} \\ \hat{\mu}_n = 0 \text{ para las restricciones inactivas} \end{cases} \tag{5.14}$$

Se comienza despejando la expresión de $\tilde{\mu}$, multiplicador de Lagrange de las restricciones activas, utilizando $\tilde{P} = \left(\tilde{G} H^{-1} \tilde{G}^T\right)^{-1}$. No hay que confundir μ con λ, ya que el primero corresponde al multiplicador que se utiliza para caracterizar las restricciones del problema QP en el lagrangiano.

$$\tilde{\mu}(x) = -\tilde{P}\tilde{W} - \begin{bmatrix} \tilde{P}\tilde{S} + \tilde{P}\tilde{G}H^{-1}F & \tilde{P}\tilde{G}H^{-1} \end{bmatrix} \begin{bmatrix} x_i \\ \lambda \end{bmatrix} \tag{5.15}$$

Utilizando la expresión de μ, se despeja U_i utilizando $\tilde{M} = H^{-1}\tilde{G}^T \left(\tilde{G}H^{-1}\tilde{G}^T\right)^{-1}$

$$U_i(x_i, \lambda) = \tilde{M}\tilde{W} + \begin{bmatrix} \tilde{M}\tilde{S} + \left(\tilde{M}\tilde{G}H^{-1} - H^{-1}\right)F & \left(\tilde{M}\tilde{G}H^{-1} - H^{-1}\right) \end{bmatrix} \begin{bmatrix} x_i \\ \lambda \end{bmatrix} \tag{5.16}$$

A continuación, se imponen las dos condiciones necesarias para calcular la expresión de las regiones.

- Condición de restricciones inactivas: $\hat{G}U_i \leq \hat{W} + \hat{S}x_i$

$$\underbrace{\begin{bmatrix} \hat{G}\tilde{M}\tilde{S} + \hat{G}\left(\tilde{M}\tilde{G}H^{-1} - H^{-1}\right)F - \hat{S} & \hat{G}\left(\tilde{M}\tilde{G}H^{-1} - H^{-1}\right) \end{bmatrix}}_{A_1} \begin{bmatrix} x_i \\ \lambda \end{bmatrix} \leq \underbrace{\hat{W} - \hat{G}\tilde{M}\tilde{W}}_{b_1} \tag{5.17}$$

- Condición de positividad multiplicadores de Lagrange en de las restricciones activas: $\tilde{\mu} \geq 0$

$$\underbrace{\begin{bmatrix} \tilde{P}\tilde{S} + \tilde{P}\tilde{G}H^{-1}F & \tilde{P}\tilde{G}H^{-1} \end{bmatrix}}_{A_2} \begin{bmatrix} x_i \\ \lambda \end{bmatrix} \leq \underbrace{-\tilde{P}\tilde{W}}_{b_2} \tag{5.18}$$

Agrupando, las regiones quedan definidas mediante:

$$\begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \begin{bmatrix} x_i \\ \lambda \end{bmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{5.19}$$

Efecto del parámetro λ en el mapa del explícito

A continuación, se propone un sistema sencillo, al que se le va a aplicar el nuevo código de la clase del explícito parametrizado en función del estado y el multiplicador de Lagrange, para observar el efecto que tiene sobre el mapa de regiones. Dado el sistema en espacios de estado:

$$\begin{aligned} & x(k+1) = x(k) - \frac{1}{2}u(k) \\ & \text{s.a } -5 \leq x \leq 5 \qquad -2.5 \leq u \leq 2.5 \end{aligned} \tag{5.20}$$

El estado del sistema tiene dimensión 1, igual que su acción de control, por lo que al parametrizar en función de λ esta tendrá la misma dimensión que la entrada. Las regiones del explícito original describen una línea sobre un eje, por lo que si se emplea un horizonte $N = 1$ el multiplicador de Lagrange aumentará en una dimensión el mapa de regiones. Para $N > 1$ la dimensión de los poliedros descritos por las regiones irá aumentando en una unidad con el horizonte, es por ello por lo que para ilustrar de la mejor forma posible este ejemplo, se utiliza horizonte uno para verlo en el plano $\mathbb{R}^2$.

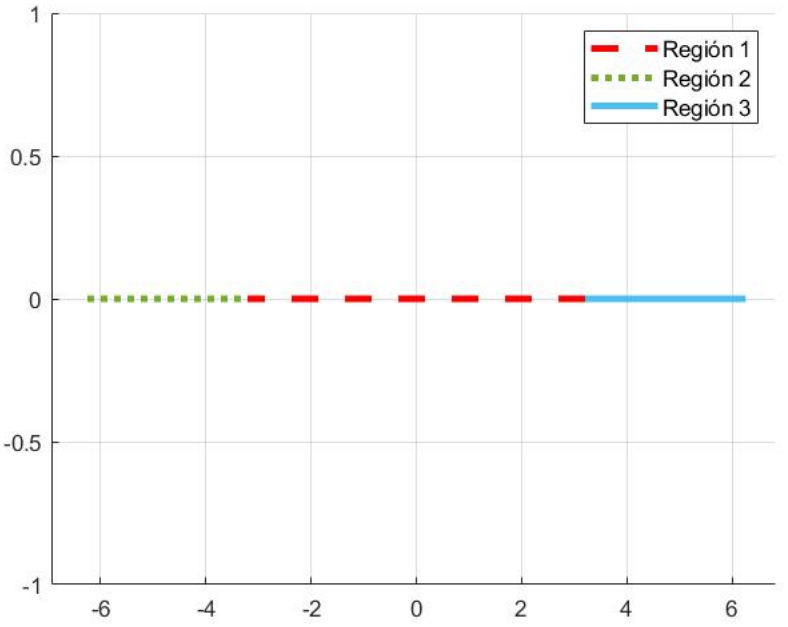


Figura 5.4 Regiones MPC explícito.

En la Figura 5.4 se pueden observar las tres regiones que componen el controlador MPC explícito del sistema de la ecuación 5.20 para un horizonte de predicción $N = 1$ y matrices de peso $Q = 1$ y $R = 1$. Se detallan a continuación, las condiciones de pertenencia de cada región así como la acción de control asociada:

- **Región 1:** línea discontinua roja $\rightarrow$ $-3.202 \leq x \leq 3.202$
  Acción de control: $U_{reg_1} = 0.7808\, x$
- **Región 2:** línea punteada verde $\rightarrow$ $-6.250 \leq x \leq -3.202$
  Acción de control: $U_{reg_2} = -2.500\, x$
- **Región 3:** línea continua azul $\rightarrow$ $3.202 \leq x \leq 6.250$
  Acción de control: $U_{reg_3} = 2.500\, x$

Utilizando el nuevo código para el mismo horizonte de predicción, se obtiene un mapa de regiones de dos dimensiones, ya que se ha parametrizado todo el problema en función del multiplicador de Lagrange. Se obtienen 5 regiones, las cuales pueden verse en el gráfico de la izquierda de la Figura 5.5, y se encuentran definidas por:

- **Región 1** (roja):

$$\begin{bmatrix} 0.800 & 0.400 \\ -0.800 & -0.400 \\ 0.400 & -0.800 \\ -0.400 & 0.800 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} \leq \begin{bmatrix} 5.000 \\ 5.000 \\ 2.500 \\ 2.500 \end{bmatrix} \tag{5.21}$$

  Ley de control:

$$U_{reg_1} = [0.4000 - 0.800] \begin{bmatrix} x \\ \lambda \end{bmatrix} \tag{5.22}$$

- **Región 2** (verde):

$$\begin{bmatrix} 1.000 & 0.000 \\ -1.000 & 0.000 \\ 0.500 & -1.000 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} \leq \begin{bmatrix} 3.750 \\ 6.250 \\ -3.125 \end{bmatrix} \tag{5.23}$$

  Ley de control:

$$U_{reg_2} = -2.500 \tag{5.24}$$

- **Región 3** (amarilla):

$$\begin{bmatrix} 1.000 & 0.000 \\ -1.000 & 0.000 \\ -0.500 & 1.000 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} \leq \begin{bmatrix} 6.250 \\ 3.750 \\ -3.125 \end{bmatrix} \tag{5.25}$$

  Ley de control:

$$U_{reg_3} = 2.500 \tag{5.26}$$

- **Región 4** (azul):

$$\begin{bmatrix} 2.000 & 0.000 \\ -2.000 & 0.000 \\ 4.000 & 2.000 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} \leq \begin{bmatrix} -7.500 \\ 12.500 \\ -25.000 \end{bmatrix} \tag{5.27}$$

  Ley de control:

$$U_{reg_4} = [2.000\ 0.000] \begin{bmatrix} x \\ \lambda \end{bmatrix} + 10.000 \tag{5.28}$$

- **Región 5** (fucsia):

$$\begin{bmatrix} 2.000 & 0.000 \\ -2.000 & 0.000 \\ -4.000 & -2.000 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} \leq \begin{bmatrix} 12.500 \\ -7.500 \\ -25.000 \end{bmatrix} \tag{5.29}$$

  Ley de control:

$$U_{reg_5} = [2.000\ 0.000] \begin{bmatrix} x \\ \lambda \end{bmatrix} - 10.000 \tag{5.30}$$

Lo primero que se observa es que, el explícito en función únicamente del estado contaba solamente con tres regiones útiles, ya que existían dos más pero con poliedros vacíos por lo que no definían ninguna región con la que se pudiese calcular. Cuando se extiende el problema introduciendo λ como parámetro, se obtienen 5 regiones útiles en las que se puede encontrar el vector de estados ampliado del sistema; compuesto por $[x \ \ \lambda]^T$. Como es natural, al contar con un mapa en $\mathbb{R}^2$ las condiciones de pertenencia a los poliedros definidos por las regiones pasan a depender de λ para todos los casos. Además, puede observarse que únicamente en la primera región, en la que no está activa ninguna restricción del problema, la acción de control depende de x y de λ, pero en las otras regiones esto no ocurre. En la segunda y tercera región, la ganancia de la acción de control es nula, y se cuenta únicamente con el término constante U_0 de la ley de control; resultando en una entrada saturada en estas regiones. Al ser la matriz K nula, significa que el valor de la acción de control

en estas regiones es independiente tanto de x como de λ. Además, la ley de control de las regiones 4 y 5 dependen únicamente del estado, ya que los términos de la columna de K que multiplican a λ son nulos.

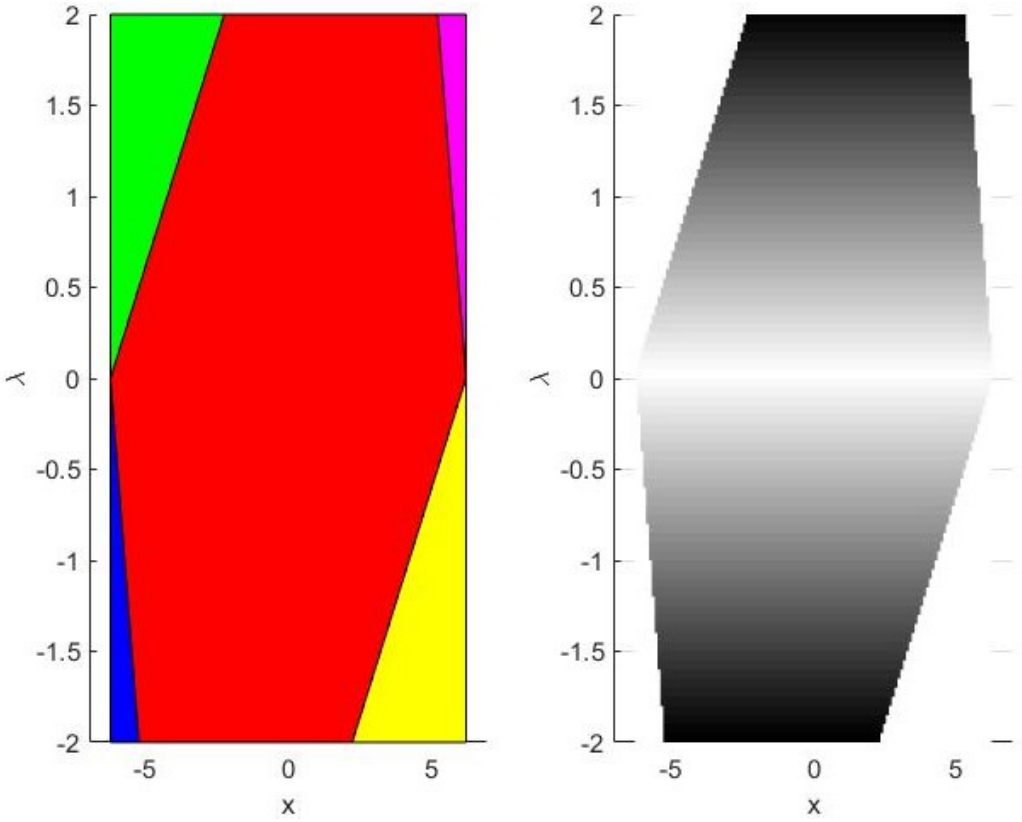


Figura 5.5 Efecto de λ en la acción de control de las regiones.

Suponiendo que $-2 \leq \lambda \leq 2$, se compara punto a punto la acción de control cuando se estudia el problema con y sin el parámetro λ, es decir, considerando $\lambda = 0$. Los resultados se muestran en la Figura 5.5. La escala de grises indica que en las zonas más claras que tienden al color blanco, la acción de control es prácticamente la misma en ambos casos, difiriendo en mayor medida en las zonas más oscuras del mapa. Existen cuatro zonas blancas que corresponden a las regiones 2-5, lo que cuadra con el análisis previo en el que se ha explicado que en estas zonas la acción de control es independiente del multiplicador de Lagrange. Por lo tanto, tiene mayor interés el estudio de la escala de grises de la primera región, ya que la acción de control sí depende de este parámetro. A medida que el parámetro se aleja más del origen el valor de las acciones de control difieren en mayor medida, viéndose representadas por el color negro.

5.2.2 Implementación del algoritmo

Para probar el funcionamiento de este esquema, se emplea un sistema sencillo de dos estados y dos entradas, pudiendo dividir el problema global en dos subsistemas acoplados por la entrada. Se define el modelo centralizado como:

$$x(k+1) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} u_1(k) \\ u_2(k) \end{bmatrix}, \tag{5.31}$$

sujeto a las siguientes restricciones:

$$\begin{aligned} -5 \leq x(k) \leq 10 \\ -1 \leq u(k) \leq 1 \end{aligned} \tag{5.32}$$

Se identifican los dos subistemas:

$$\begin{aligned} x_1(k+1) &= x_1(k) + u_1(k) - u_2(k) \\ x_2(k+1) &= x_2(k) - u_1(k) + u_2(k), \end{aligned} \tag{5.33}$$

y se resuelve el problema de optimización con horizonte de predicción $N = 2$, utilizando las siguientes matrices de peso globales:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{5.34}$$

El sistema parte de la posición inicial x_0, y su estado de referencia es x_{ref}:

$$x_0 = \begin{bmatrix} 5 \\ -2 \end{bmatrix} \quad x_{ref} = \begin{bmatrix} 1.25 \\ 0.75 \end{bmatrix} \tag{5.35}$$

En primer lugar, se calcula de forma *offline* un controlador MPC explícito por cada subsistema, entregándoles las matrices que definen el modelo de cada uno, sus restricciones, matrices de peso, y horizonte de predicción. Se generarán dos controladores explícitos cuyas regiones estarán parametrizadas en función del estado local y el multiplicador de Lagrange λ, que se encargará de que ambos agentes, tras iterar, lleguen al mismo valor de la acción de control. Destacar que, el mapa del estado local no puede representarse en esta ocasión, ya que se emplea un horizonte de predicción tal que hace que los poliedros de las regiones estén en $\mathbb{R}^5$; una dimensión por el estado y otras cuatro porque el multiplicador λ tiene dos componentes, y al extenderlo $\forall N$ multiplica por dos su dimensión.

Una vez se han calculado los controladores, se procede a implementar el bucle de simulación, en el que dentro de cada paso k se ejecuta un proceso iterativo que corresponde a la negociación entre agentes. Dentro de este proceso, el valor del multiplicador se va actualizando (con paso $\gamma = 0.01$) hasta que se consiga que ambos agentes calculen el mismo valor de la acción de control tras resolver su problema de optimización local. El proceso iterativo calcula la norma de la diferencia entre las acciones de control locales, y la compara con un umbral $\eta = 1e-4$. Al alcanzar este valor mínimo, se considera que los resultados son suficientemente parecidos, y se detiene la negociación para ese instante de simulación k, aplicando última acción de control obtenida. El número de iteraciones necesarias para que el problema converja depende sensiblemente del valor del paso γ, ya que si es muy pequeño tardará mucho en converger mientras que si es muy grande puede que nunca lo haga.

Resultados de la simulación

Se analizan a continuación los resultados obtenidos aplicando el esquema de descomposición dual al sistema anterior, comparándolos con la respuesta obtenida tras resolver el problema de optimización global con un MPC centralizado. Para todas las variables, se representan los resultados del centralizado en línea discontinua. En primer lugar, se presenta la evolución de las acciones de control del sistema en la Figura 5.6, comprobando que ambas trayectorias son prácticamente iguales. Este algoritmo distribuido converge al mismo comportamiento que el centralizado, pero en este caso se establece un umbral a partir del cual se detiene la negociación entre agentes por lo que existe una leve discrepancia en las trayectorias. Este umbral η afecta a que la trayectoria distribuida se aproxime en mayor medida a la centralizada.

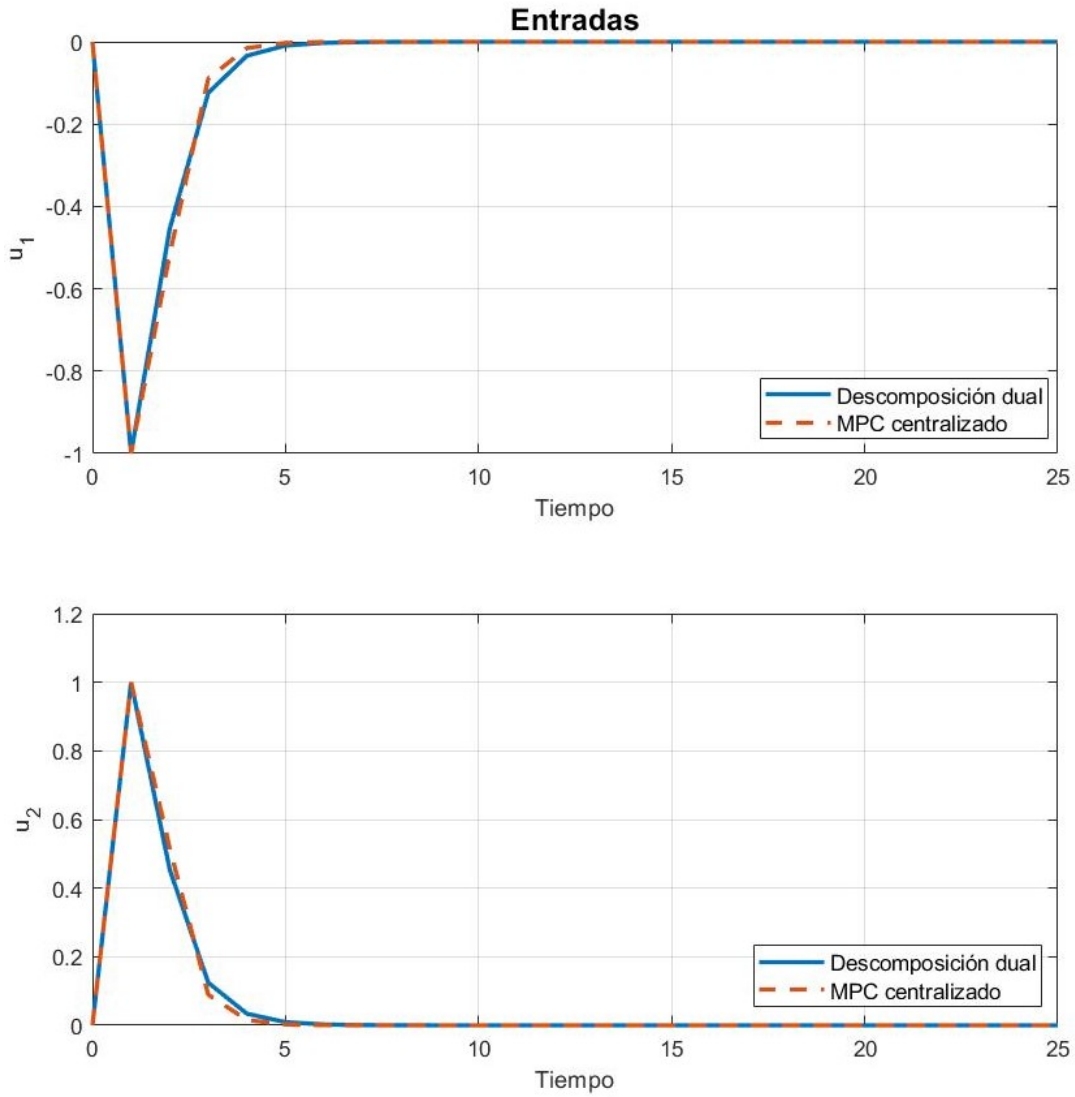


Figura 5.6 Evolución de la entrada.

En la próxima página, en las Figura 5.7, se puede observar la evolución de los estados de ambos subsistemas, buscando que x_1 tome el valor 1.25, y x_2 alcance el valor 0.75. Cada controlador local busca optimizar su función objetivo propia, tratando de llevar su estado hacia la referencia deseada con la actuación de la acción de control, pero al estar acoplados por la entrada deben llevar a un acuerdo utilizando el multiplicador de Lagrange λ. Por ello, el estado se aproxima a su referencia y se estabiliza entorno a un valor muy cercano pero nunca llega a alcanzarlo, ya que comparten la misma u y en la negociación iterativa llegan a un consenso para tratar de permanecer ambos cercanos a su referencia. De nuevo, se puede comparar la evolución del esquema distribuido en línea continua en comparación con el centralizado en discontinua. Por último, en la Figura 5.8 se ha representado el coste acumulado obtenido durante los 25 instantes de simulación, el mismo valor en ambos casos, lo cual comprueba que con la estrategia distribuida de descomposición dual se consigue el mismo desempeño que el MPC centralizado.

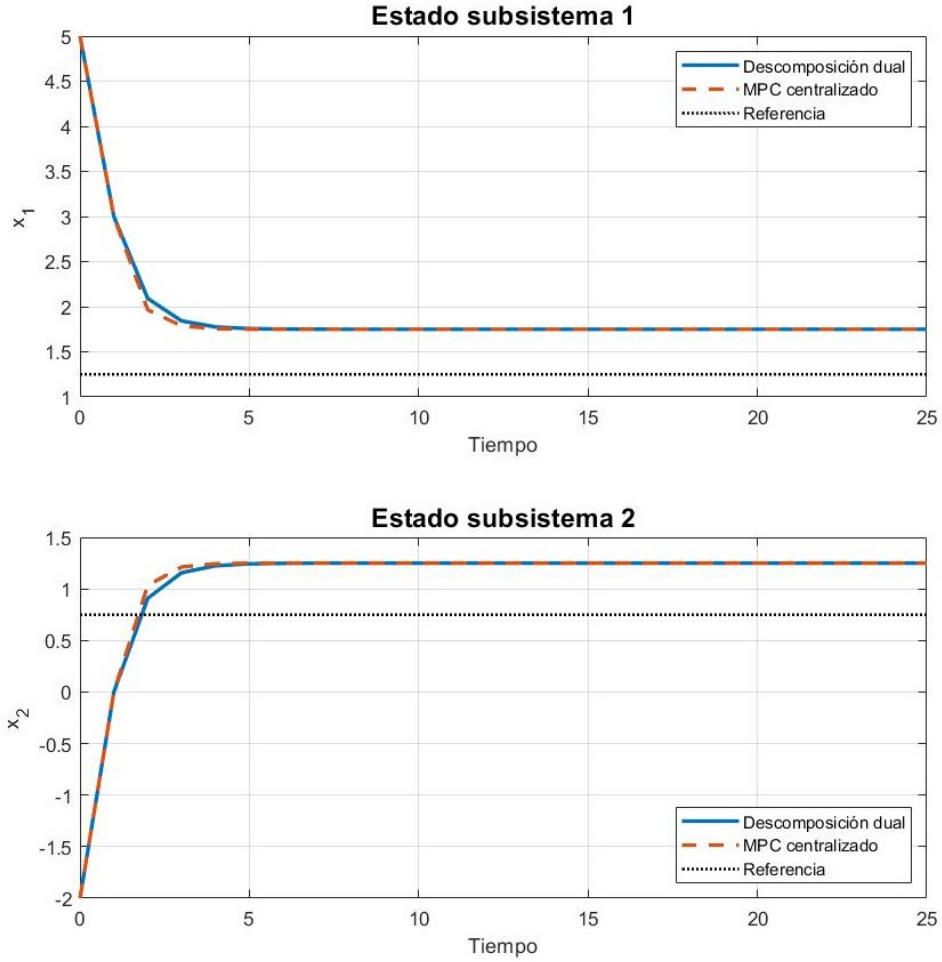


Figura 5.7 Evolución del estado del sistema.

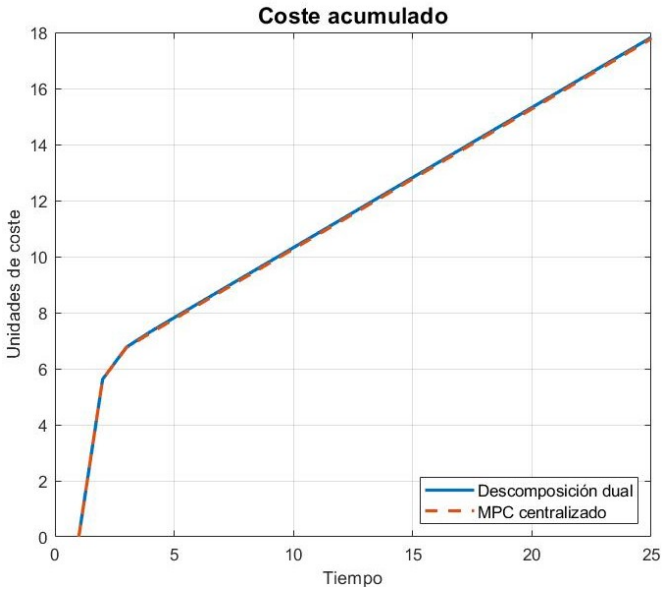


Figura 5.8 Coste acumulado de ambas estrategias.

5.3 Feasible cooperation-based MPC (FC-MPC)

En [17] se exponen varios métodos para abordar problemas de MPC distribuido, donde los agentes interactúan entre sí. En primer lugar, se presenta un MPC basado en comunicación entre subsistemas, donde cada agente optimiza su función objetivo sin tener información a cerca de las funciones de coste de los otros subsistemas, por lo que las acciones de control de los otros agentes no se ven alteradas. El principal inconveniente que se encuentra, es que el objetivo de cada agente entra en conflicto con los objetivos de los otros, por lo que puede no converger.

Para solventar los inconvenientes del MPC basado en comunicación, los autores proponen un algoritmo óptimo para el sistema completo, y no solo por subsistemas, que converge al comportamiento del MPC centralizado. Esta estrategia, denominada como MPC basado en cooperación, modifica la función objetivo de cada subsistema para tener en cuenta a otros agentes, evaluando el desempeño global del sistema y asegurando que se satisfacen las restricciones en el estado y la entrada. Para ello, descomponiendo el sistema en M subsistemas, la función de costes global será la suma ponderada de las funciones objetivos de cada subsistema, siendo la suma de los pesos ρ_i igual a la unidad. Se utilizarán los mismos pesos que en [17], que corresponden con $\rho_i = {}^1/_M$, $\forall i = 1...M$.

$$\Phi = \sum_i \rho_i \Phi_i \;\; \text{con} \;\; \rho_i \geq 0, \;\; \text{y} \;\; \sum_i \rho_i = 1 \quad \forall i = 1...M \tag{5.36}$$

Cualquiera de estos dos esquemas distribuidos se implementan a través de un proceso iterativo, en el que los subsistemas intercambian sus variables. El número de iteraciones puede limitarse, ya que puede que no nos convenga continuar con el bucle hasta converger, por lo que se contabilizará este número y se valora con un umbral si es preciso seguir iterando o no. Este umbral evalúa la diferencia entre el valor anterior y actual del estado y de la entrada, si se encuentra por encima del umbral ε se sigue con el proceso de intercambio y negociación, en caso contrario, se detiene el bucle y se implementa y recogen los valores de u y x.

5.3.1 Formulación del problema de optimización

Sea un sistema de dos entradas $U = [U_1 \; U_2]^T$, el esquema MPC basado en cooperación, FC-MPC, puede implementarse haciendo el siguiente cambio de variables: $U = M_1 U_1 + M_2 U_2$ con $M_1 = [1 \; 0]^T$ y $M_2 = [0 \; 1]^T$. Para el estudio de este algoritmo, se emplea el siguiente modelo en espacios de estado de un sistema con dos entradas acopladas entre ellas; $B_{12}, B_{21} \neq 0$.

$$x(k+1) = \underbrace{\begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}}_{A} x(k) + \underbrace{\begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}}_{B} u(k) \tag{5.37}$$

Formulando el problema QP del sistema centralizado:

$$\begin{aligned} &\min_U \frac{1}{2} U^T H U + x^T F^T U \\ &\text{s.a } GU \leq W + Sx \end{aligned} \tag{5.38}$$

Aplicando el cambio de variables propuesto $U = M_1 U_1 + M_2 U_2$, se pueden identificar los siguientes términos:

$$\begin{aligned} V = {} & \frac{1}{2} \left(M_1 U_1 + M_2 U_2\right)^T H \left(M_1 U_1 + M_2 U_2\right) \; + \; x^T F^T \left(M_1 U_1 + M_2 U_2\right) = \\ & \frac{1}{2} U_1^T \underbrace{M_1^T H M_1}_{H_1} U_1 \; + \; \frac{1}{2} U_1^T \underbrace{M_1^T H M_2}_{T_2{}^T} U_2 \; + \; \frac{1}{2} U_2^T \underbrace{M_2^T H M_1}_{T_1{}^T} U_1 \; + \; \frac{1}{2} U_2^T \underbrace{M_2^T H M_2}_{H_2} U_2 \; + \\ & x^T \underbrace{F^T M_1}_{F_1{}^T} U_1 \; + \; x^T \underbrace{F^T M_2}_{F_2{}^T} U_2 \end{aligned} \tag{5.39}$$

Se pueden identificar los siguientes problemas de optimización en función de las variables U_1 y U_2.

- **Subsistema 1**

$$\begin{aligned} &\min_{U_1} \frac{1}{2} U_1^T H_1 U_1 + \left(x^T F_1^T + U_2^T T_1^T\right) U_1 \\ &\text{s.a } GM_1 U_1 \leq W + Sx - GM_2 U_2 \end{aligned} \tag{5.40}$$

  Siendo $H_1 = M_1^T H M_1$, $F_1^T = F^T M_1$ y $T_1^T = M_2^T H M_1$.

- **Subsistema 2**

$$\begin{aligned} &\min_{U_2} \frac{1}{2} U_2^T H_2 U_2 + \left(x^T F_2^T + U_1^T T_2^T\right) U_2 \\ &\text{s.a } GM_2 U_2 \leq W + Sx - GM_1 U_1 \end{aligned} \tag{5.41}$$

  Siendo $H_2 = M_2^T H M_2$, $F_2^T = F^T M_2$ y $T_2^T = M_1^T H M_2$.

Es importante destacar que, en este enfoque cooperativo, cada agente tiene en cuenta al otro en su función objetivo y restricciones gracias al cambio de variable realizado. Con esta nueva formulación de los problemas de optimización, se modifica el código de la clase inicial para incluir esta notación según el esquema MPC cooperativo.

5.3.2 Desarrollo de las condiciones KKT

Se realiza el desarrollo de las condiciones de optimalidad de Karush Kuhn Tucker para el subsistema i, identificando el otro agente con el subíndice j, resaltar que este desarrollo habrá que realizarlo para cada subsistema, ya que cada uno tiene un controlador MPC explícito propio. En la notación, se emplea la variable M_k, que consiste en un vector columna compuesto por ceros excepto el valor que ocupa la posición k que será un uno.

$$\mathscr{L} = \frac{1}{2} U_i^T H_i U_i + \left(x^T F_i^T + U_j^T T_i^T\right) U_i + \mu^T \left(GM_i U_i - W - Sx + GM_j U_j\right) \tag{5.42}$$

$$\nabla_{U_i} \mathscr{L} = 0 \rightarrow H_i U_i + F_i x + T_i U_j + M_i^T G^T \mu = 0 \tag{5.43}$$

Se emplea el subíndice n para identificar una restricción, que será una fila de la desigualdad generada por las restricciones $GM_i U_i \leq W + Sx - GM_j U_j$.

$$\tilde{G}_n M_i U_i - \tilde{W}_n - \tilde{S}_n x + \tilde{G}_n M_j U_j = 0 \tag{5.44}$$

$$\hat{G}_n M_i U_i \leq \hat{W}_n + \hat{S}_n x - \hat{G}_n M_j U_j \tag{5.45}$$

$$\begin{cases} \tilde{\mu}_n \geq 0 \text{ para las restricciones activas} \\ \hat{\mu}_n = 0 \text{ para las restricciones inactivas} \end{cases} \tag{5.46}$$

Se despeja U_i de la condición del gradiente del Lagrangiano:

$$U_i(x, U_j) = -H_i^{-1} \left(F_i x + T_i U_j + M_i^T \tilde{G}^T \tilde{\mu}\right) \tag{5.47}$$

Expresión de $\tilde{\mu}$ utilizando $\tilde{P} = \left(\tilde{G} M_i H_i^{-1} M_i^T \tilde{G}^T\right)^{-1}$ y $\tilde{Y} = \tilde{G} M_i H_i^{-1}$.

$$\tilde{\mu}(x, U_j) = -\tilde{P}\tilde{W} - \begin{bmatrix} \tilde{P}\left(\tilde{Y} F_i + \tilde{S}\right) & \tilde{P}\left(\tilde{Y} T_i - \tilde{G} M_j\right) \end{bmatrix} \begin{bmatrix} x \\ U_j \end{bmatrix} \tag{5.48}$$

Sustituyendo en la expresión de U_i y utilizando $\tilde{L} = H_i^{-1} M_i^T \tilde{G}^T \tilde{P}$:

$$U_i(x, U_j) = \tilde{L}\tilde{W} + \begin{bmatrix} \tilde{L}\tilde{S} + \left(\tilde{L}\tilde{Y} - H_i^{-1}\right) F_i & \left(\tilde{L}\tilde{Y} - H_i^{-1}\right) T_i - \tilde{L}\tilde{G} M_j \end{bmatrix} \begin{bmatrix} x \\ U_j \end{bmatrix} \tag{5.49}$$

Cálculo de regiones:

- Condición de restricciones inactivas:

$$\hat{G}M_i U_i \leq \hat{W} + \hat{S}x - \hat{G}M_j U_j \tag{5.50}$$

Sustituyendo la expresión 5.49 de U_i

$$J_1 \begin{bmatrix} x \\ U_j \end{bmatrix} \leq j_1 \tag{5.51}$$

$$J_1 = \begin{bmatrix} \hat{G}M_i \left(\tilde{L}\tilde{S} + \left(\tilde{L}\tilde{Y} - H_i^{-1}\right) F_i\right) - \hat{S} & \hat{G}M_i \left(\left(\tilde{L}\tilde{Y} - H_i^{-1}\right) T_i - \tilde{L}\tilde{G}M_j\right) + \hat{G}M_j \end{bmatrix} \tag{5.52}$$

$$j_1 = \hat{W} - \hat{G}M_i \tilde{L}\tilde{W} \tag{5.53}$$

- Condición de positividad multiplicadores de Lagrange en de las restricciones activas:

$$\tilde{\mu} \geq 0 \tag{5.54}$$

$$\underbrace{\begin{bmatrix} \tilde{P}\left(\tilde{Y}F_i + \tilde{S}\right) & \tilde{P}\left(\tilde{Y}T_i - \tilde{G}M_j\right) \end{bmatrix}}_{J_2} \begin{bmatrix} x \\ U_j \end{bmatrix} \leq \underbrace{-\tilde{P}\tilde{W}}_{j_2} \tag{5.55}$$

Agrupando, las regiones para un subsistema i quedan definidas mediante:

$$\begin{bmatrix} J_1 \\ J_2 \end{bmatrix} \begin{bmatrix} x \\ U_2 \end{bmatrix} \leq \begin{bmatrix} j_1 \\ j_2 \end{bmatrix} \tag{5.56}$$

5.3.3 Implementación del algoritmo

Para probar el funcionamiento del algoritmo FC-MPC aplicado a MPC explícito, se va a utilizar el modelo de una planta de cuatro tanques [4] como la de la Figura 5.9. Los tanques superiores, 3 y 4, descargan en los inferiores, 1 y 2 respectivamente. Existe un depósito en el inferior del cual se suministrará agua a los distintos tanques a través de dos bombas, q_A y q_B en la imagen.

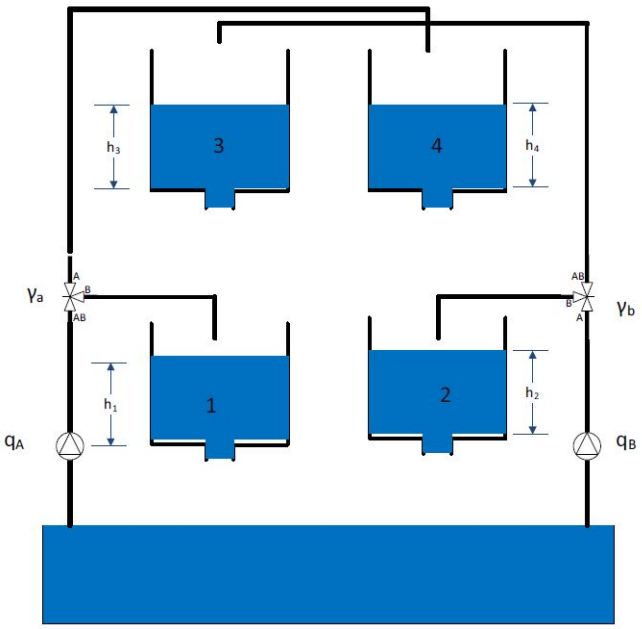


Figura 5.9 Sistema de cuatro tanques [4].

El flujo a través de las bombas lo denotaremos como u_1 y u_2 respectivamente, y se controla a través de dos válvulas, este flujo corresponde con las entradas al sistema. En cuanto al estado, se distingue uno por cada tanque, y se puede dividir el sistema en dos subsistemas, identificando en uno los tanques de la derecha; 4 y 2, y en otro los respectivos de la izquierda; 3 y 1. La división en subsistemas se realiza de esta manera porque, al descargar un tanque sobre otro, esos tanques se encuentran acoplados por el estado. El estado mide la altura del nivel del agua de cada tanque, que se encuentra limitado por: $0.2 \leq h_i \leq 2$, medido en $[m]$, mientras que el caudal de la bomba A se mide con u_1, que tiene como restricciones $0 \leq q_A \leq 3.26$, medido en $[m^3/h]$. Por último, u_2, que mide el caudal de la bomba B, se encuentra limitada por $0 \leq q_B \leq 4$, también en unidades $[m^3/h]$. Es preciso trabajar con un modelo lineal del sistema, por lo que se debe tener en cuenta el punto de operación o equilibrio del mismo, que distinguiremos con el superíndice "$\circ$":

$$h_i{}^{\circ} = 0.65\ [m]\ \forall i = 1...4, \quad q_A{}^{\circ} = 1.63\ [m^3/h], \quad q_B{}^{\circ} = 2\ [m^3/h] \tag{5.57}$$

Se definen a continuación las variables con las que se va a trabajar, teniendo en cuenta el punto de equilibrio del sistema:

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} h_1 \\ h_3 \\ h_2 \\ h_4 \end{bmatrix} - \begin{bmatrix} h^{\circ} \\ h^{\circ} \\ h^{\circ} \\ h^{\circ} \end{bmatrix}, \ \text{con} \ -0.45\ [m] \leq x_i \leq 0.71\ [m] \ \ \forall i = 1,2 \tag{5.58}$$

$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} q_A \\ q_B \end{bmatrix} - \begin{bmatrix} q_A{}^{\circ} \\ q_B{}^{\circ} \end{bmatrix}, \ \text{con} \ \begin{bmatrix} -1.63 \\ -2 \end{bmatrix} [m^3/h] \leq \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \leq \begin{bmatrix} 1.63 \\ 2 \end{bmatrix} [m^3/h] \tag{5.59}$$

Sujeto a las restricciones anteriores, se presenta el modelo del sistema centralizado:

$$x(k+1) = \begin{bmatrix} 0.9705 & 0.0205 & 0 & 0 \\ 0 & 0.9792 & 0 & 0 \\ 0 & 0 & 0.9661 & 0.0195 \\ 0 & 0 & 0 & 0.9802 \end{bmatrix} x(k) + \begin{bmatrix} 0.0068 & 0.0011 \\ 0 & 0.0137 \\ 0.0002 & 0.0091 \\ 0.0160 & 0 \end{bmatrix} u(k) \tag{5.60}$$

Identificando los dos subsistemas:

- Subsistema 1

$$x_1(k+1) = \begin{bmatrix} 0.9705 & 0.0205 \\ 0 & 0.9792 \end{bmatrix} x_1(k) + \begin{bmatrix} 0.0068 & 0.0011 \\ 0 & 0.0137 \end{bmatrix} u(k) \tag{5.61}$$

- Subsistema 2

$$x_2(k+1) = \begin{bmatrix} 0.9661 & 0.0195 \\ 0 & 0.9802 \end{bmatrix} x_2(k) + \begin{bmatrix} 0.0002 & 0.0091 \\ 0.0160 & 0 \end{bmatrix} u(k) \tag{5.62}$$

Se definen las matrices de peso: $Q = I \in \mathbb{R}^{4x4}$ y $R = 0.01I \in \mathbb{R}^{2x2}$.

Al tener dos problemas de optimización, uno por cada subsistema, se ha de calcular para cada uno las respectivas regiones resultantes del MPC explícito con esta estrategia incorporada, por lo que habrá que simular el código en dos ocasiones. La llamada a la nueva clase, *ExpVenkat*, difiere de los códigos anteriores, recibiendo los siguientes argumentos: $obj = ExpVenkat(H,\ F,\ G,\ W,\ S,\ M_i,\ M_j)$. A diferencia del explícito normal y aquel que tenía el efecto de las perturbaciones incorporadas, este código no recibe las matrices que definen el sistema ni las restricciones del mismo, sino que recibe las matrices H y F de la función objetivo del problema QP centralizado, y las matrices que definen las restricciones del problema global $GU \leq W + Sx$. Por lo que, en este caso, se han de calcular previamente las matrices que definen el problema QP centralizado, y, para la generación de las regiones del explícito, será necesario introducir todas estas matrices junto con los vectores columna M_i y M_j. El orden en el que se introducen estos vectores columna es importante, ya que en primer lugar irá aquel que corresponde con el sistema que se está estudiando. Un inconveniente que se presenta es que todo se ha generado tomando $N = 1$ como horizonte de predicción, por lo que el comportamiento predictivo del control MPC no se estaría aprovechando.

A continuación se muestra el algoritmo que se ha de implementar para cada instante del bucle de simulación, habiendo calculado previamente los controladores MPC explícitos con la estructura de los problemas de optimización que se han obtenido tras hacer el cambio de variables.

Algoritmo 1 Procedimiento FC-MPC

Se establece un umbral ε para el cual se deja de iterar. Se procede para cada instante de muestreo k:

1: Inicializo número de iteraciones en cero: $p \leftarrow 0$
2: Inicializo la norma en un valor mayor al del umbral: $dist = 100$
3: **while** $p < p_{max}$ y $max(dist) < \varepsilon$ **do**
4: Calculo el valor de $u_i(k)$ según la clase del explícito para cada subsistema: $\forall i \in 1,\ M$.
5: Actualizo: $u_i \leftarrow \frac{1}{M}u_i(k) + \left(1 - \frac{1}{M}\right) u_i(k-1),\ \forall i \in \{1,\ M\}$.
6: Actualizo: $x_i \leftarrow \frac{1}{M}x_i(k) + \left(1 - \frac{1}{M}\right) x_i(k-1),\ \forall i \in \{1,\ M\}$.
7: Calculo el valor con el que se va a comparar el umbral: $dist \leftarrow \| (x_i(k), u_i(k)) - (x_i(k-1), u_i(k-1)) \|$
8: Actualizo: $p \leftarrow p+1$
9: **end while**
10: Calculo el valor del estado: $x_i(k+1) = A_{ii}x_i(k) + B_{ii}u_i(k) + B_{i,j}u_j(k),\ \ \forall i, j = 1...M$ y $j \neq i$

Resultados de la simulación

En primer lugar, se compara el comportamiento de un MPC centralizado optimizando el problema QP del sistema global con la función *quadprog* de *Matlab*, con el resultado que se obtiene de precalcular las regiones del explícito de manera *offline* con este esquema implementado para una simulación con distintos cambios de referencias.

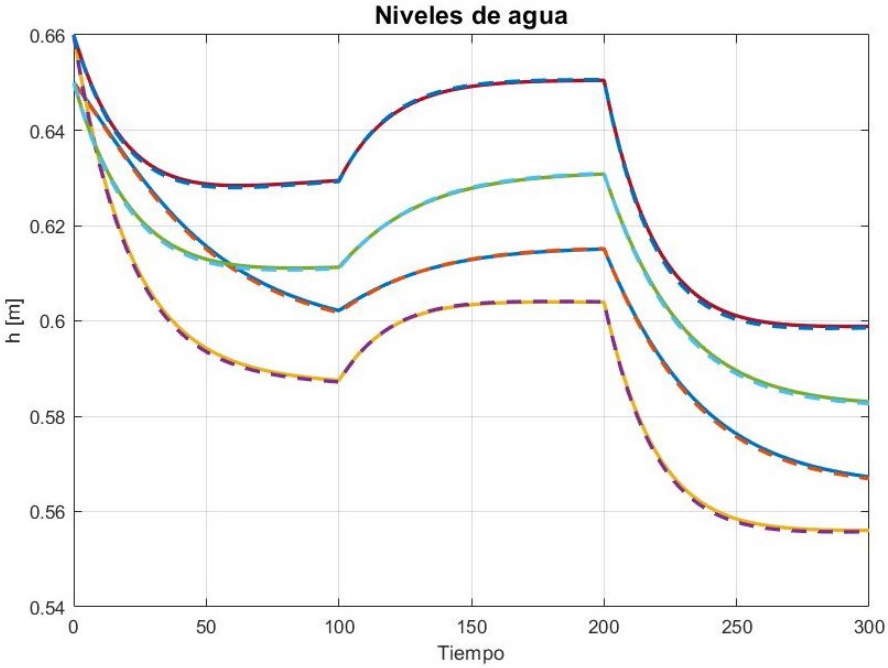


Figura 5.10 Comparación entrada del FC-MPC explícito con MPC centralizado.

En la Figura 5.10, se representa en línea continua la evolución de los cuatro niveles de agua (estado + punto de operación) en el seguimiento de referencias utilizando el esquema FC-MPC adaptado al explícito, mientras que las líneas discontinuas corresponden con la respuesta del MPC centralizado. Se puede observar que la respuesta es prácticamente igual que la centralizada, al igual que la evolución de los caudales (actuación + punto de operación) necesarios para alcanzar las referencias, pudiendo observar esta evolución en la Figura 5.11 de la próxima página.

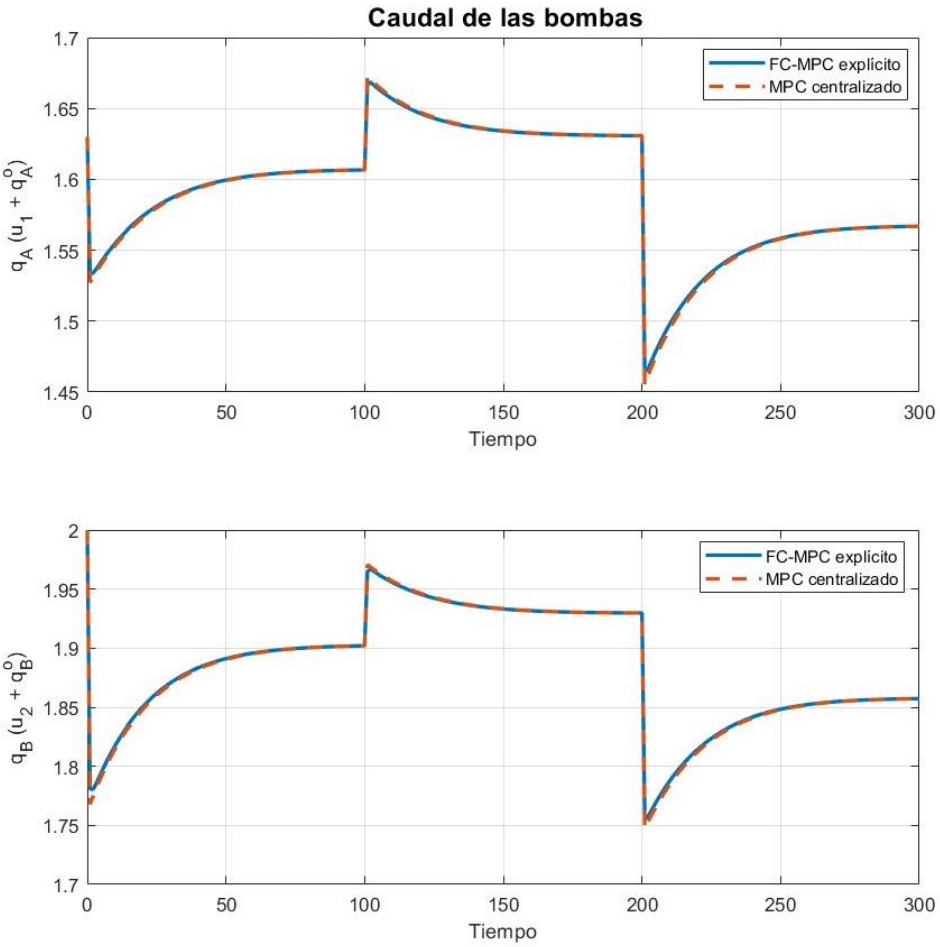


Figura 5.11 Comparación entrada del FC-MPC explícito con MPC centralizado.

En cuanto al coste acumulado, calculado como $V = \sum_{\forall k} \; {x_{k+1}}^T Q x_{k+1} + {u_k}^T R u_k$, se puede observar en la Figura 5.12 que el enfoque cooperativo resulta en el mismo desempeño que el centralizado.

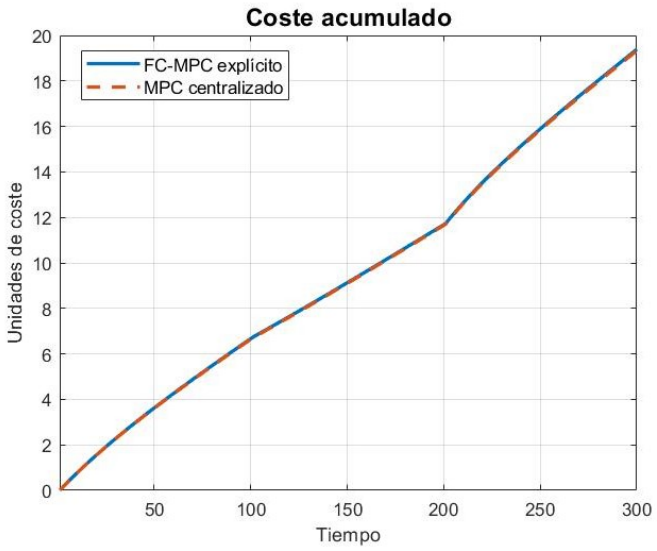


Figura 5.12 Comparación coste del FC-MPC explícito con MPC centralizado.

En segundo lugar, se ha decidido comparar el resultado obtenido implementando la estrategia MPC cooperativo utilizando la función *quadprog* de *Matlab* para resolver el problema QP y obtener el valor de la acción de control óptima, con el comportamiento que resulta de aplicar la clase del explícito para este esquema FC-MPC. Se representa con línea punteada el enfoque utilizando el explícito, y con línea discontinua la estrategia tradicional utilizando *quadprog*.

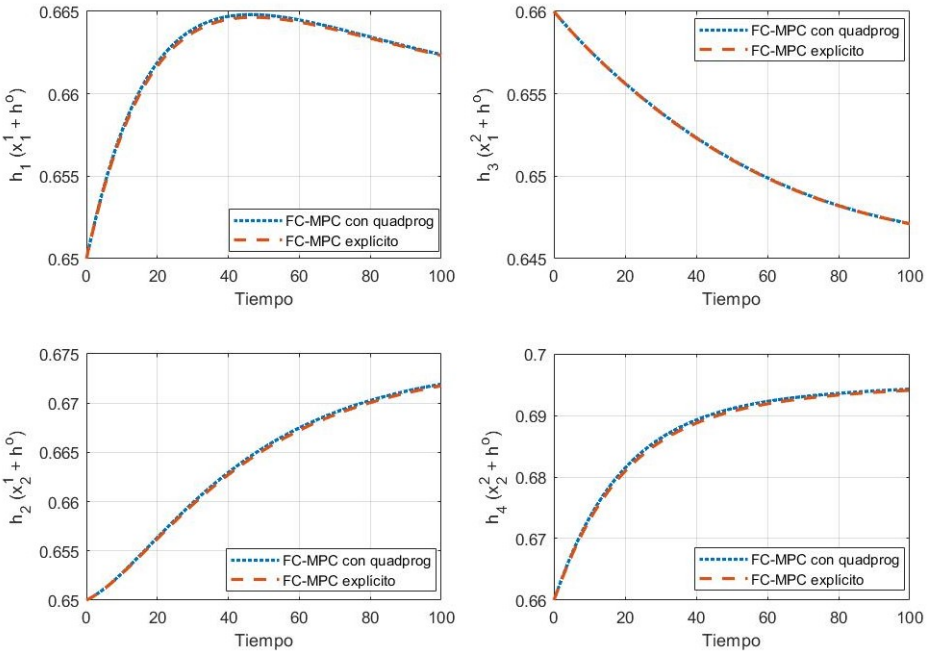


Figura 5.13 Evolución del estado comparando estrategias de control distribuido.

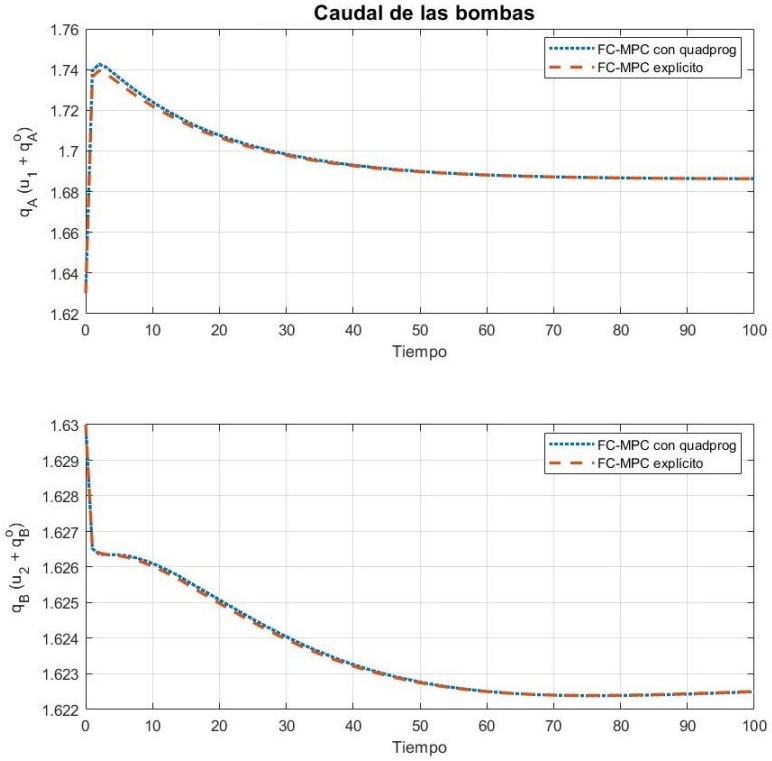


Figura 5.14 Evolución de la entrada comparando estrategias de control distribuido.

La diferencia más notoria entre las estrategias es el tiempo de computación que se emplea, lo cual saca a la luz las ventajas de emplear un control MPC explícito, al calcular *offline* el resultado del problema de optimización, y simplemente *online* hacer una comprobación de en qué región se encuentra el sistema para así aplicar la acción de control pertinente. En este caso, el proceso de negociación o el algoritmo iterativo, naturalmente presente en una estrategia de control distribuido, no cuenta con muchas iteraciones hasta que converge según el umbral que hemos establecido, por lo que puede que no sea el mejor ejemplo para ilustrar lo beneficioso que es para el tiempo de computación utilizar una estrategia explícita.

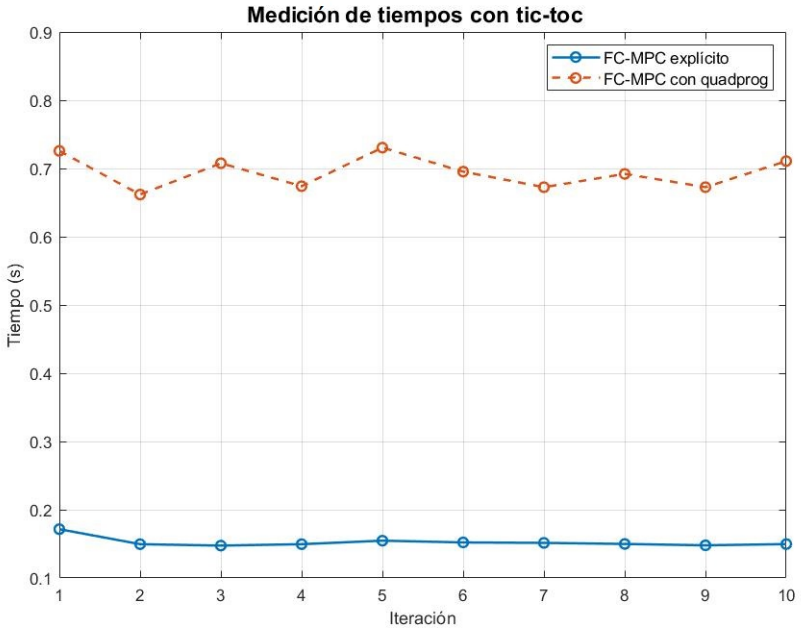


Figura 5.15 Comparación del tiempo de computación de estrategias distribuidas.

Para la medición de tiempos se ha empleado la herramienta *tic-toc* de *Matlab*, también existe la función *CPUtime* pero revisando la ayuda de este *software* la primera herramienta proporciona un resultado más fiable que la segunda. La forma de proceder es la siguiente, se escribe "tic" al inicio y "toc" al final de la parte del código que quiera cronometrarse, el tiempo determinado por "toc" se almacena en una variable. Para el ejemplo actual, se ha cronometrado únicamente el bucle de control de ambas estrategias, eliminando en la de *quadprog* que imprima por pantalla los resultados de la optimización, ya que eso ralentizaría su tiempo y se quiere ser lo más justos posibles con ambos enfoques para obtener una medición fiable. Se puede observar en la Figura 5.15 que el tiempo de computación con el algoritmo FC-MPC explícito es aproximadamente 3.5 veces menor que en el caso de utilizar *quadprog* para la resolución del problema QP.

5.4 Min-max

Un enfoque utilizado en control MPC robusto cuando se tienen en cuenta las perturbaciones en el modelo, es minimizar la función objetivo para el peor caso posible de la incertidumbre; esta estrategia se conoce como control MPC min-max (MMMPC) [18, 19, 20]. Se expresa el sistema discreto en la ecuación 5.63, donde $w(k) \in \mathbb{R}^{\dim w}$ modela las incertidumbres, las cuales se suponen acotadas $\|w(k)\| \leq \varepsilon$.

$$x(k+1) = Ax(k) + Bu(k) + Dw(k) \tag{5.63}$$

Se define la función objetivo para el MPC, esta vez dependiente del valor de las perturbaciones w, además del estado x y la entrada u del sistema.

$$V(x, \mathbf{u}, \mathbf{w}) = \sum_{0}^{N-1} \left(x_{k+1}^T \, Q \, x_{k+1} \; + \; u_k^T \, R \, u_k \right) \tag{5.64}$$

Considerando $\mathbf{u} = [u_0, u_1, \ldots, u_{N-1}]^T$ la secuencia de valores de la señal de control, y $\mathbf{w} = [w_0, w_1, \ldots, w_{N-1}]^T$ una posible secuencia de valores futuros de las perturbaciones del sistema durante el horizonte de predicción. El control min-max se basa en encontrar la secuencia de control $\mathbf{u}$ que minimiza la función objetivo $V(x, \mathbf{u}, \mathbf{w})$ para la peor realización posible de la incertidumbre.

$$V^*(x) = \min_{\mathbf{u}} \max_{\mathbf{w} \in W_N} V(x, \mathbf{u}, \mathbf{w}) \tag{5.65}$$

donde W_N denota el conjunto de todas las posibles secuencias de las perturbaciones. Debido a que la variable $\mathbf{w}$ es convexa, se sabe que el máximo de la función objetivo para el peor caso de la incertidumbre se alcanza al menos en uno o más vértices del politopo W_N, que define los límites de la incertidumbre.

5.4.1 Planteamiento propuesto

En este apartado se propone un planteamiento para control MPC explícito distribuido, basándose en los fundamentos de la estrategia min-max. Para ello, se estudia un sistema global que puede dividirse en subsistemas acoplados por la entrada, expresados según la ecuación 5.63. Se trata el acoplamiento entre subsistemas como la perturbación de cada agente, es decir, dados dos agentes i y j, se puede definir:

$$D_i = B_{ij}, \;\; D_j = B_{ji}, \;\; w_i = u_j \;\; \text{y} \;\; w_j = u_i \tag{5.66}$$

La idea consiste en que cada agente se compromete a cambiar como máximo su acción de control en una cantidad $\pm\delta$ respecto al instante anterior, conociendo por lo tanto los posibles valores de las perturbaciones máximos o mínimos que puede generar el agente vecino en el otro subsistema. Estos valores máximos y mínimos corresponden a los vértices de W_N en ese instante k, donde se evaluará la función objetivo y se calculará una acción de control para cada vértice de las incertidumbres, ya que se conoce que en esos puntos se producen los máximos del problema min-max. Para el primer instante, este esquema se apoya en la trayectoria resultante tras aplicar un MPC centralizado; será el arranque para esta estrategia, comprometiéndose por tanto en el siguiente instante a no desviarse cada subsistema en un $\pm\delta$ de la entrada del centralizado.

Para ello, se ha de formular la ecuación 5.63 de tal forma que la notación de la acción de control sea incremental respecto al instante anterior, necesario para poder introducir restricciones del valor de este incremento. Se parametriza el MPC explícito en función del estado, la acción de control del subsistema en el instante anterior, y las perturbaciones, calculando una ley de control que corresponde a Δu_k, a la que luego habrá que sumarle la entrada en el instante $k-1$. Se parte de la clase generada en el subcapítulo 4.3, *"Estudio de las perturbaciones"*, en el que el MPC explícito se parametriza en función de un vector de estados ampliado que contiene el estado y las perturbaciones. Se introducen restricciones en el estado, la entrada global y el incremento de la acción de control; para poder acotarlo respecto al instante previo.

Fijando un valor de δ para todo el problema, teniendo dos subsistemas y estudiando un horizonte de predicción de valor $N = 1$, para cada instante k de muestreo se conoce el valor de los vértices de las perturbaciones a cada subsistema i, definidos como:

$$w_i^n(k) = u_j(k-1) + \delta, \qquad w_i^m(k) = u_j(k-1) - \delta \qquad \text{para } j \neq i \tag{5.67}$$

Para cada vértice de la perturbación ${w_i}^n(k)$, utilizando la clase del explícito parametrizada de forma incremental, se calcula el correspondiente incremento de la entrada $\Delta {u_i}^n(k)$, y como consecuencia la entrada óptima asociada a ese vértice: ${u_i}^n(k) = u_i(k-1) + \Delta {u_i}^n(k)$. Se propone un planteamiento heurístico a través de una tabla, desde un punto de vista similar al de la teoría de juegos propuesto en [13]. Se construye la tabla 5.1 poniendo en la diagonal principal el coste de implementar para el valor de la perturbación ${w_i}^n$ su acción de control óptima asociada ${u_i}^n$, y en la antidiagonal el coste resultante de aplicar una acción de control no óptima ${u_i}^n$ para un valor de la perturbación ${w_i}^m$ con $n \neq m$.

Tabla 5.1 Distintas posibilidades de función objetivo.

	${w_i}^n$	${w_i}^m$
${u_i}^n$	$V(x_i,\ {u_i}^n,\ {w_i}^n)$	$V(x_i,\ {u_i}^n,\ {w_i}^m)$
${u_i}^m$	$V(x_i,\ {u_i}^m,\ {w_i}^n)$	$V(x_i,\ {u_i}^m,\ {w_i}^m)$

A priori, los agentes no saben el valor de la perturbación que se aplicará en el instante k, pero a través de las opciones resultantes al evaluar la entrada en los vértices de las perturbaciones, cada agente puede tomar una decisión. Suponiendo que el agente vecino va a implementar una perturbación tal que al otro agente le hace el mayor daño posible, este aplicará aquella u_i que más le proteja desde un punto de vista del coste. Por lo tanto, una vez calculada la tabla para el agente i en el instante k, se evaluarán los elementos de la antidiagonal; correspondientes a la combinación de perturbaciones con acciones de control no óptimas, y se aplicará aquella ${u_i}^{ii}$ que de un menor coste para ${w_i}^{jj}$ para $ii \neq jj$.

5.4.2 Formulación del problema de optimización

A continuación, se presenta la formulación que ha de emplearse para generar un código de MPC explícito que tenga en cuenta el valor de la acción de control en el instante anterior, además de las perturbaciones, calculando con la ley de control un valor Δu. Partiendo del siguiente modelo del sistema global en espacios de estado:

$$x(k+1) = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} x(k) + \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} u(k) \tag{5.68}$$

Considerando que los subsistemas se encuentran únicamente acoplados por la entrada; $A_{12} = 0 = A_{21}$, se puede realizar la siguiente división en subsistemas, según la ecuación 5.63.

$$\begin{aligned} x_1(k+1) &= A_{11}x_1(k) + B_{11}u_1(k) + B_{12}u_2(k) \\ x_2(k+1) &= A_{22}x_2(k) + B_{22}u_2(k) + B_{21}u_1(k) \end{aligned} \tag{5.69}$$

Dado un subsistema i, se considera el término de acoplamiento B_{ij} con $j \neq i$ como la matriz D_i de perturbaciones, y la acción de control del otro subsistema u_j como vector de perturbaciones w_i. Estudiando el subsistema i, se expresa la entrada en el instante k como la suma de la acción de control en el instante anterior más un incremento:

$$\begin{aligned} & u_i(k) = u_i(k-1) + \Delta u_i(k) \\ & s.a \ \ u_{min} \leq u_i(k) \leq u_{max} \\ & \qquad -\delta \leq \Delta u_i(k) \leq \delta \end{aligned} \tag{5.70}$$

Se busca parametrizar el MPC explícito en función del estado, la acción de control del subsistema de estudio en el instante anterior, y la perturbación; que sería la acción de control del otro agente. Para ello, utilizando la notación incremental, se crea un vector de estados ampliado $\hat{x}$, que contiene el estado en el instante k, y la acción de control en $k-1$.

$$\underbrace{\begin{bmatrix} x_i(k+1) \\ u_i(k) \end{bmatrix}}_{\hat{x}_i(k+1)} = \underbrace{\begin{bmatrix} A_{ii} & B_{ii} \\ 0 & I \end{bmatrix}}_{\hat{A}} \underbrace{\begin{bmatrix} x_i(k) \\ u_i(k-1) \end{bmatrix}}_{\hat{x}_i(k)} + \underbrace{\begin{bmatrix} B_{ii} \\ I \end{bmatrix}}_{\hat{B}} \Delta u_i(k) + \underbrace{\begin{bmatrix} B_{ij} \\ 0 \end{bmatrix}}_{\hat{D}} u_j(k) \qquad \text{con } j \neq i \tag{5.71}$$

$$\hat{x}_i(k+1) = \hat{A}\hat{x}_i(k) + \hat{B}\Delta u_i(k) + \hat{D}u_j(k) \tag{5.72}$$

Para generar el código, se modifica la clase del explícito que tiene en cuenta el efecto de las perturbaciones de tal manera que reciba el modelo del sistema y las restricciones en el estado x, entrada u e incremento de la entrada Δu. Lo primero que ha de realizarse, es transformar el sistema a la forma incremental contemplada en 5.71, pero a parte de la nueva formulación de las matrices del sistema también se debe tener en cuenta que las regiones se construirán con las restricciones del estado ampliado y del incremento de u, por lo que hay que expresar las restricciones en función de estos parámetros.

- Restricciones en $x_i(k)$: $x_{min} \leq x_i(k) \leq x_{max}$

$$\underbrace{\begin{bmatrix} 1 \\ -1 \end{bmatrix}}_{A_x} x_i(k) \leq \underbrace{\begin{bmatrix} x_{max} \\ -x_{min} \end{bmatrix}}_{b_x} \tag{5.73}$$

- Restricciones en $u_i(k)$: $u_{min} \leq u_i(k) \leq u_{max}$

$$\underbrace{\begin{bmatrix} 1 \\ -1 \end{bmatrix}}_{A_u} u_i(k) \leq \underbrace{\begin{bmatrix} u_{max} \\ -u_{min} \end{bmatrix}}_{b_u} \tag{5.74}$$

  Con $u_i(k) = u_i(k-1) + \Delta u_i(k)$.

- Restricciones en $\Delta u_i(k)$: $-\delta \leq \Delta u_i(k) \leq \delta$

$$\underbrace{\begin{bmatrix} 1 \\ -1 \end{bmatrix}}_{A_{lim}} \Delta u_i(k) \leq \underbrace{\begin{bmatrix} \delta \\ \delta \end{bmatrix}}_{b_{lim}} \tag{5.75}$$

Como el vector de estados ampliados $\hat{x}_i(k)$ se construye como $[x_i(k) \;\; u_i(k-1)]^T$, las restricciones de este parámetro deben incluir las del estado y las de la entrada en el instante anterior, por lo que:

$$\underbrace{\begin{bmatrix} A_x & 0 \\ 0 & A_u \end{bmatrix}}_{A_{\hat{x}}} \hat{x}_i(k) \leq \underbrace{\begin{bmatrix} b_x \\ b_u \end{bmatrix}}_{b_{\hat{x}_i}} \tag{5.76}$$

Además, la matriz de pesos Q de la función objetivo, debe contemplar la modificación de la formulación del vector de estados:

$$\hat{Q} = \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix} \tag{5.77}$$

Las restricciones de la entrada serán las de Δu, ponderando a esta variable en la función objetivo con la matriz R sin realizar ninguna modificación. De esta forma, ya se tienen las restricciones y las matrices del sistema formuladas de manera correcta para implementar el mismo código de MPC explícito que en el subcapítulo 4.3, con la salvedad de que la ley de control calcula un incremento, y el controlador explícito se parametriza en función de $x_i(k)$, $u_i(k-1)$ y $u_j(k)$; correspondiendo esta última variable a la acción de control del otro subsistema que se considerará como perturbación al agente de estudio.

5.4.3 Implementación del algoritmo

Para probar el funcionamiento del planteamiento propuesto, se utilizará la planta de cuatro tanques de la Figura 5.9 [4], que se desarrolló en el subcapítulo 5.3 *"FC-MPC"*. A continuación, se recuerdan las ecuaciones del sistema global y se divide en subsistemas según las ecuaciones 5.63 y 5.69.

$$x(k+1) = \begin{bmatrix} 0.9705 & 0.0205 & 0 & 0 \\ 0 & 0.9792 & 0 & 0 \\ 0 & 0 & 0.9661 & 0.0195 \\ 0 & 0 & 0 & 0.9802 \end{bmatrix} x(k) + \begin{bmatrix} 0.0068 & 0.0011 \\ 0 & 0.0137 \\ 0.0002 & 0.0091 \\ 0.0160 & 0 \end{bmatrix} u(k) \tag{5.78}$$

Identificando los dos subsistemas:

- Subsistema 1

$$x_1(k+1) = \begin{bmatrix} 0.9705 & 0.0205 \\ 0 & 0.9792 \end{bmatrix} x_1(k) + \begin{bmatrix} 0.0068 \\ 0 \end{bmatrix} u_1(k) + \begin{bmatrix} 0.0011 \\ 0.0137 \end{bmatrix} u_2(k) \tag{5.79}$$

$$\hat{x}_1(k+1) = \begin{bmatrix} 0.9705 & 0.0205 & 0.0068 \\ 0 & 0.9792 & 0 \\ 0 & 0 & 1 \end{bmatrix} \hat{x}_1(k) + \begin{bmatrix} 0.0068 \\ 0 \\ 1 \end{bmatrix} \Delta u_1(k) + \begin{bmatrix} 0.0011 \\ 0.0137 \\ 0 \end{bmatrix} u_2(k) \tag{5.80}$$

  Con $\hat{x}_1(k) = [x_1(k) \;\; u_1(k-1)]^T$.

- Subsistema 2

$$x_2(k+1) = \begin{bmatrix} 0.9661 & 0.0195 \\ 0 & 0.9802 \end{bmatrix} x_2(k) + \begin{bmatrix} 0.0091 \\ 0 \end{bmatrix} u_2(k) + \begin{bmatrix} 0.0002 \\ 0.0160 \end{bmatrix} u_1(k) \tag{5.81}$$

$$\hat{x}_2(k+1) = \begin{bmatrix} 0.9661 & 0.0195 & 0.0091 \\ 0 & 0.9802 & 0 \\ 0 & 0 & 1 \end{bmatrix} \hat{x}_2(k) + \begin{bmatrix} 0.0091 \\ 0 \\ 1 \end{bmatrix} \Delta u_2(k) + \begin{bmatrix} 0.0002 \\ 0.0160 \\ 0 \end{bmatrix} u_1(k) \tag{5.82}$$

  Con $\hat{x}_2(k) = [x_2(k) \;\; u_2(k-1)]^T$

Se definen las matrices de peso: $Q = I \in \mathbb{R}^{4x4}$ y $R = 0.01I \in \mathbb{R}^{2x2}$, y las restricciones:

$$\begin{aligned} &-1.63 \leq u_1 \leq 1.63 \qquad -2 \leq u_2 \leq 2 \\ &-0.45 \leq x \leq 0.71 \qquad -\delta \leq \Delta u_i \leq \delta, \; \forall i = 1,2 \end{aligned} \tag{5.83}$$

Resultados de la simulación

Se estudiará el valor de δ como un parámetro del problema, ya que la respuesta del sistema está ligada a la magnitud de este valor. Además, como se comentó previamente, el arranque de esta estrategia distribuida sería con la trayectoria resultante del control MPC centralizado, que se simulará con un horizonte de predicción elevado. El esquema distribuido calculando un controlador MPC explícito para cada subsistema se estudia para un horizonte de predicción $N = 1$, ya que se ha tratado de simular para un horizonte mayor pero escala muy mal al haber parametrizado el explícito en función de tantas variables. Se comparará la respuesta centralizada con el método propuesto, siendo la centralizada siempre mejor al poder utilizar un horizonte de predicción mayor que uno. El valor óptimo de δ depende también del horizonte de predicción que se emplee, así como del estado inicial del sistema, por lo que va a ser un parámetro que haya que ajustar para cada caso.

El sistema parte del estado inicial: $x_{10} = [-0.03 \;\; -0.04]^T$ y $x_{20} = [-0.03 \;\; -0.04]^T$, con el objetivo de llevar a ambos subsistemas al origen de x, que corresponde con el punto de operación de la variable h, ya que $x = h - h^\circ$. Los resultados se presentan en las variables del sistema, es decir, se presentan los niveles de agua h de los cuatro tanques, correspondientes al estado x más el punto de operación:

$$\begin{aligned} \begin{bmatrix} h_1 \\ h_3 \end{bmatrix} &= x_1 + \begin{bmatrix} h^\circ \\ h^\circ \end{bmatrix} \text{ con } \; h^\circ = 0.65 \; [m] \\ \begin{bmatrix} h_2 \\ h_4 \end{bmatrix} &= x_2 + \begin{bmatrix} h^\circ \\ h^\circ \end{bmatrix} \text{ con } \; h^\circ = 0.65 \; [m] \end{aligned} \tag{5.84}$$

De igual manera, en vez de representar el valor de la entrada u_i, se representarán los caudales q_A y q_B, por lo que se tiene en cuenta el punto de operación.

$$\begin{aligned} q_A &= u_1 + {q_A}^\circ, \text{ con } \; {q_A}^\circ = 1.63 \; [m^3/h] \\ q_B &= u_2 + {q_B}^\circ, \text{ con } \; {q_B}^\circ = 2 \; [m^3/h] \end{aligned} \tag{5.85}$$

Se emplea un horizonte para el MPC centralizado de $N_c = 20$, habiendo ajustado el valor del incremento máximo al orden de $1e-2$. El desempeño del sistema centralizado mejora con respecto al distribuido a medida que se aumenta el horizonte de predicción, ya que en el esquema distribuido se ha simulado con un horizonte unitario. No ha sido posible generar los controladores explícitos para un horizonte mayor a causa de las operaciones matriciales que se llevan a cabo para poder generar las regiones, que incrementan su dimensión con el horizonte y aumentan la carga computacional. A continuación, se comparan las trayectorias de los caudales y los niveles de agua, así como el coste acumulado, para distintos valores de δ del mismo orden de magnitud para ver la sensibilidad en la respuesta. A su vez, se plasman los resultados del MPC centralizado de horizonte 20 en línea continua, ya que supone la respuesta óptima.

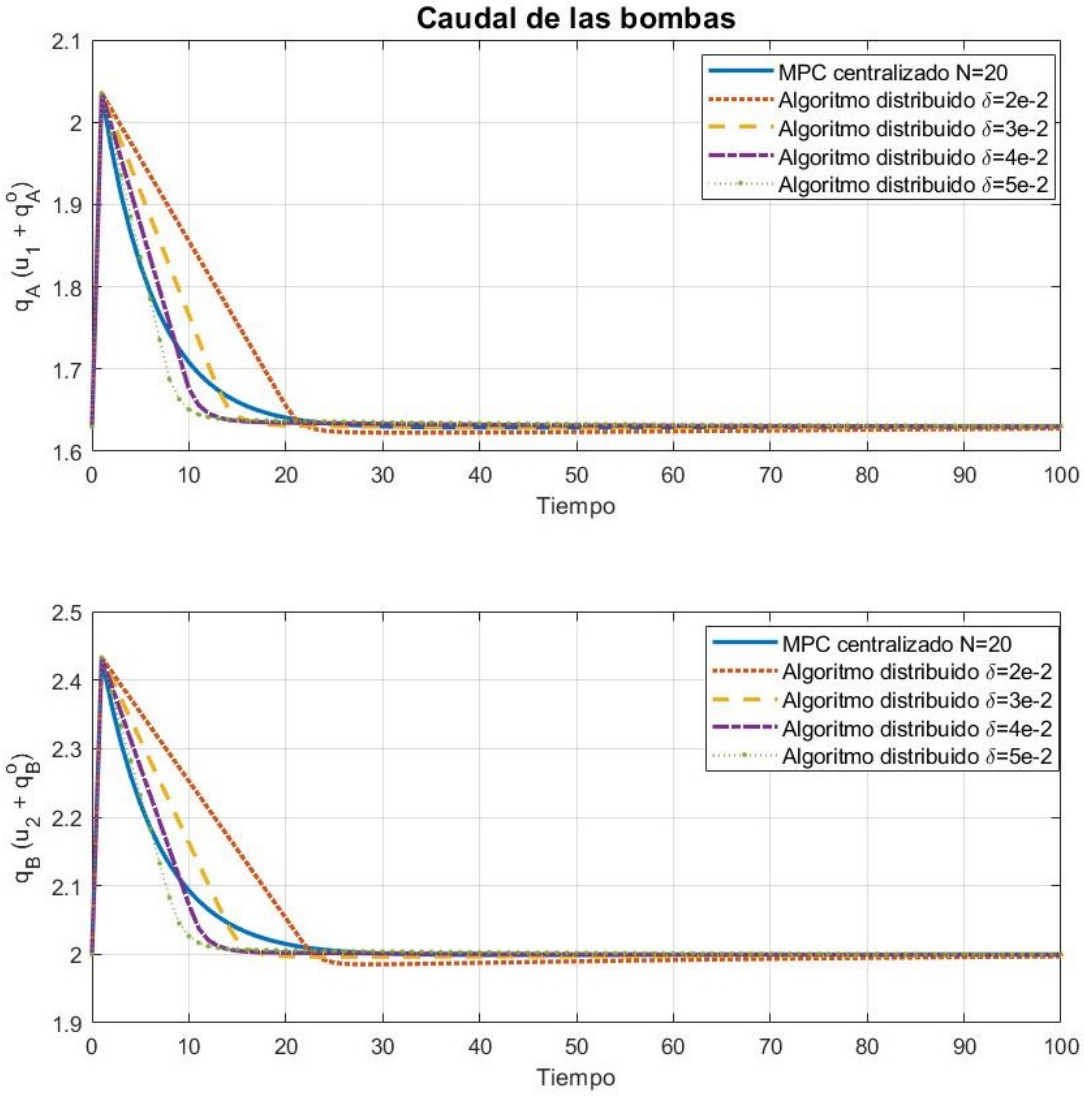


Figura 5.16 Comparación de la evolución de la entrada en función del valor de δ.

Finalmente, se puede concluir de la evolución de los caudales y niveles de agua de la Figuras 5.16 y 5.17 que el valor óptimo de δ es $4e-2$, ya que sus trayectorias son las que se encuentran más próximas al centralizado para todos los valores de δ representados. Para valores $\delta > 4e-2$ la evolución es más lenta, pero más controlada que para valores $\delta < 4e-2$. También puede observarse en la Figura 5.18 que para este valor de δ, el coste es el más próximo al MPC centralizado con horizonte 20, aunque siempre superior a este. Puede decirse que para el valor óptimo seleccionado, sin haber un proceso iterativo de negociación entre los agentes, teniendo en cuenta la interacción entre subsistemas como perturbaciones, la respuesta es bastante correcta.

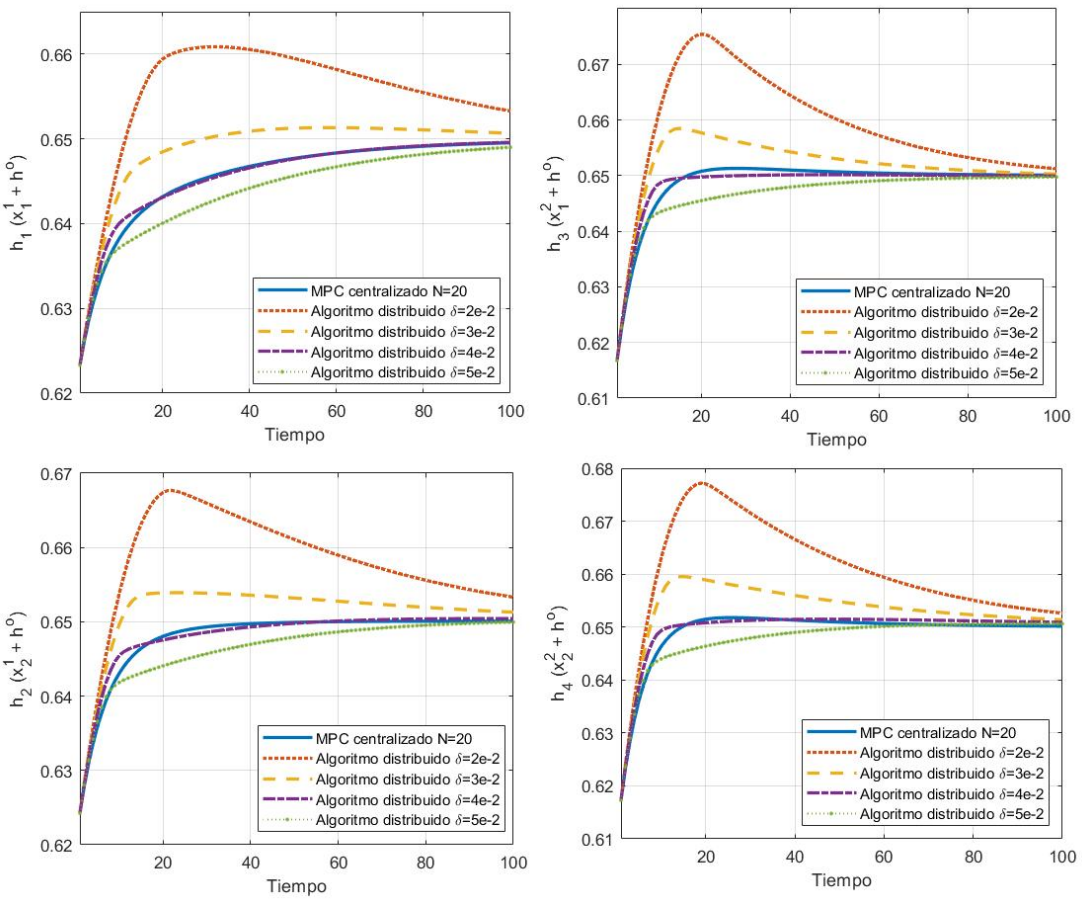


Figura 5.17 Comparación de la evolución del estado en función del valor de δ.

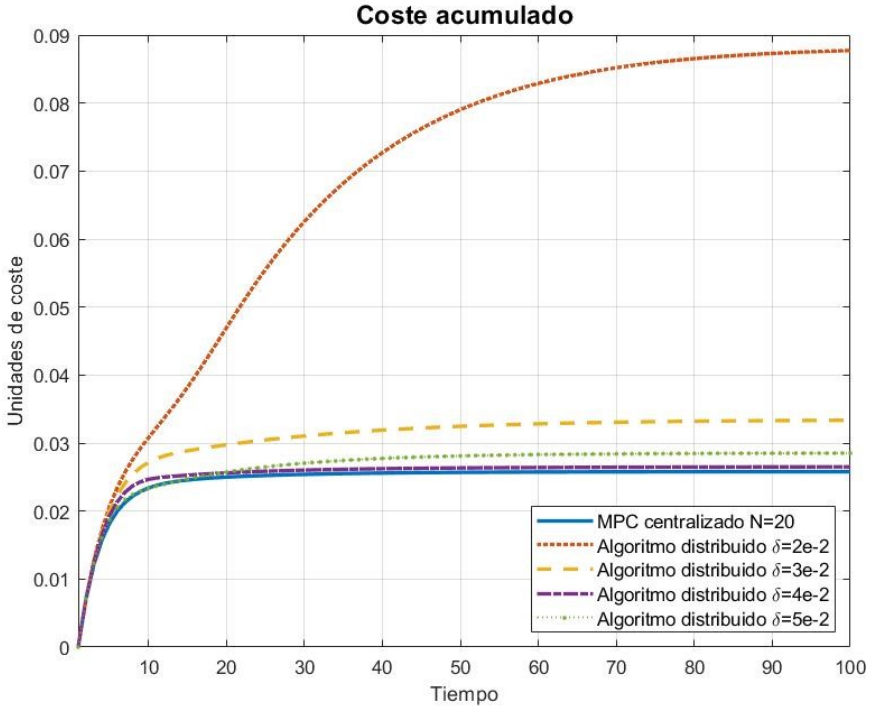


Figura 5.18 Coste acumulado según el valor de δ, en comparación con el centralizado.

6 Conclusiones y líneas futuras

6.1 Conclusiones

Uno de los principales objetivos de este proyecto era la comprensión de los fundamentos teóricos en los que se apoya la estrategia del MPC explícito, para conocer cómo se generan y caracterizan las regiones del mapa del estado. Puede decirse que este punto ha sido cubierto de forma satisfactoria, ya que la inquietud por este tema surgió de la necesidad de comprender lo que otros *solvers* disponibles en la literatura proporcionaban como resultado, para poder tener una mayor comprensión y capacidad de análisis de estas regiones. Las condiciones KKT determinan que la división del mapa del estado que caracteriza al explícito se genera a través de la combinación de restricciones activas, generando así las distintas regiones. En el código generado para este proyecto se estudian todas las posibles combinaciones de restricciones, y como consecuencia algunas de ellas no son factibles al ser restricciones que no pueden estar activas de forma simultánea. La carga computacional, aunque sea *offline*, se incrementa al estudiar este tipo de combinaciones, generando regiones vacías que no se descartan.

Un inconveniente de la aplicación de las condiciones KKT y del código propuesto, es la cantidad de operaciones matriciales que se realizan, involucrando el cálculo de muchas inversas para despejar las expresiones de la ley de control o las desigualdades matriciales que construyen cada región. A medida que se incrementa el horizonte de predicción, estas operaciones matriciales se extienden para todo N, por lo que complica el cálculo y aumenta la carga computacional ya que las restricciones aumentan exponencialmente, y con ello el número de posibles regiones. Para sistemas simples de dos estados se ha llegado a simular hasta un horizonte de $N = 4$ para el esquema inicial del explícito, esto supone aun así un horizonte de predicción muy pequeño, por lo que la capacidad predictiva del MPC no se estaría aprovechando. Esto supone un gran inconveniente, ya que los sistemas reales como redes de tráfico, de agua o de potencia como las populares *smart-grids* involucran a muchos subsistemas interactuando entre sí, por lo que la aplicación a este tipo de dinámicas aun con un horizonte unitario sería complicado.

Como se comentó al inicio del trabajo, la motivación de querer comprender cómo se generaban las regiones del explícito surgió cuando se trató de implementar un MPC a un *drone* comercial durante el desarrollo de mi TFG. Al contar con matrices del sistema de gran dimensión, ya que el vector de estados tenía 12 componentes, aun utilizando un horizonte $N = 1$ y sin introducir restricciones en todos los estados, las regiones calculadas superaban las 3000. No se introdujeron restricciones en todos los estados porque solamente interesaban, desde el punto de vista del vuelo del *drone*, el control de orientación y posición, y no el de velocidades. El mapa del estado por lo tanto contenía regiones no acotadas, ya que se sabe que los poliedros de las regiones se trazan como combinación de las restricciones activas, y al menos 6 componentes del estado no habían sido restringidas. Las referencias de vuelo que se emplearon en simulación se encontraban dentro de las restricciones del problema, ya que se simulaba el vuelo del *drone* en una habitación, ya que se emplea un robot comercial pequeño que si se vuela en exteriores se desestabiliza mucho si hay viento. Por lo tanto, el sistema estaba siendo controlado con únicamente una región de las 3089 totales, teniendo prácticamente la misma respuesta que con un LQR sin presentar ninguna ventaja significativa, ya que con horizonte de predicción unitario no se aprovecha la capacidad predictiva de esta estrategia de control.

En las estrategias distribuidas de descomposición dual y *FC-MPC* se ha podido comprobar que logran el mismo desempeño que el MPC centralizado. Se observa una ligera discrepancia en las trayectorias, sobretodo en el caso de la descomposición dual, que puede ser debida al umbral que se establece para detener el

algoritmo de negociación entre agentes, por lo que se para la simulación antes de que el algoritmo converja. Otro motivo puede ser por las numerosas inversas que se calculan a la hora de definir las regiones del explícito, ya que se ha aplicado este algoritmo a otros sistemas de estudio y empeoraba el comportamiento si, por ejemplo, en las matrices del sistema existían números decimales. Un gran inconveniente al parametrizar el explícito en función de otras variables que no fuesen el estado, como se ha hecho en el capítulo 5, es lo poco que se puede aumentar el horizonte de predicción, utilizando $N = 2$ para la descomposición dual, pero utilizando un horizonte unitario en los demás casos. En este caso, se pudo aumentar el horizonte porque el sistema de estudio era sencillo, pero en el caso de haber empleado un sistema real más complejo aumentar el horizonte utilizando la estrategia explícita puede que no fuese una opción, ya que el código generado es poco escalable como se ha comentado. En el esquema cooperativo *FC-MPC* se pudo observar la ventaja principal del explícito, que reside en poder precalcular la ley de control y ahorrar carga computacional en la implementación, ya que presentaba una ventaja significativa en tiempo de simulación con respecto a la herramienta *quadprog* de *Matlab*. Por último, la limitación del último planteamiento que se presenta en el capítulo 5 de este proyecto es que para cambios de referencia distintos del punto de operación el algoritmo no proporciona una buena respuesta que se aproxime al comportamiento centralizado, además de tener que ajustar el parámetro δ de forma iterativa para cada horizonte de predicción.

6.2 Futuras líneas de trabajo

A continuación, tras haber nombrado algunas limitaciones que se han podido extraer como conclusión de este trabajo, se proponen una seria de posibles líneas futuras de trabajo con las que poder mejorar o solventar estos inconvenientes, involucrando otras líneas de investigación.

- Optimizar el código generado para evitar el análisis de combinación de restricciones que no son factibles. Esto reduce el número de regiones de estudio y también de regiones generadas, ya que el código existente genera regiones vacías con las que no se puede controlar el sistema, como se observó para el sistema que se estudió con distintos horizontes de predicción. Esto es sencillo de detectar si se tratan de restricciones, por ejemplo, de cotas superiores e inferiores en un estado, ya que no es posible que el estado active la restricción de su valor mínimo y máximo a la vez.
- Un posible planteamiento podría ser emplear técnicas de *machine learning* que utilicen redes neuronales para aproximar las regiones del explícito. Esta popular técnica permite trabajar con grandes cantidades de datos, lo que permitiría poder estudiar el problema para un horizonte de predicción suficientemente grande para mejorar el desempeño del sistema. Además de poder ampliar el horizonte, permitiría escalar la estrategia a sistemas más complejos con un mayor número de agentes. En la literatura pueden encontrarse metodologías que emplean *machine learning* y redes neuronales en conjunto con MPC [21], además artículos recientes como [22, 23] particularizan para la estrategia explícita del MPC. Esto podría aprovecharse para mejorar el comportamiento del seguimiento de referencias del planteamiento heurístico basado en min-max, ya que teniendo una secuencia de las posibles perturbaciones del subsistema vecino para todo el horizonte de predicción con $N > 1$ podría mejorar el desempeño de esta estrategia.
- El desempeño de la estrategia distribuida basada en min-max puede verse mejorado también si se emplean técnicas de *machine learning*. En vez de probar de forma iterativa qué *delta* es la mejor para cada sistema en función del horizonte de predicción del arranque centralizado, se pueden emplear redes neuronales que se entrenen para crear una base de datos en las que se estudie según los distintos horizontes de predicción y estados iniciales el valor de *delta* óptimo a emplear en cada caso. Esta idea es similar a la de un *paper* que ha sido aceptado por el *ECC - European Control Conference*, en el que pude colaborar junto con mis tutores de este trabajo; Paula Chanfreut y José María Maestre, y con Eduardo Fernández Camacho. La idea propuesta consiste en aplicar redes neuronales para acelerar la convergencia del algoritmo de descomposición dual, creando una base de datos que estima los multiplicadores de Lagrange óptimos para realizar el arranque de la estrategia distribuida con ellos.

Por último, el pasado septiembre inicié mis estudios como alumna de doctorado en Ingeniería de Sistemas y Automática de la Universidad de Sevilla, teniendo como directores de tesis a José María Maestre y Eduardo Fernández Camacho. Ambos proponen una técnica novedosa dentro del control distribuido, conocida como control coalicional [24, 25], que surge para solventar la posible saturación que puede darse en la red por la comunicación que existe entre subsistemas para coordinarse. La estrategia consiste en agrupar de forma

dinámica aquellos controladores locales entre los que exista un fuerte acoplamiento en lo que se conoce como *clusters*; estas agrupaciones o coaliciones se formarán para mejorar el rendimiento del sistema. Me gustaría enfocar mi tesis dentro de este ámbito, y viendo cómo se puede aprovechar las técnicas de *machine learning*, sobre las que me interesaría aprender, para poder combinarlas con esta estrategia de control, viendo si la formulación del explícito pudiese adaptarse a este enfoque.

Apéndice A
MPC explícito

Código A.1 Código clase MPC explícito.

```
% Class for the definition of explicit MPC by JM Maestre pepemaestre@us.es, A S
    ánchez-Amores asamores@us.es, P Chanfreut pchanfreut@us.es

classdef ExpMPC
    properties
        % Overall system matrices
        Ac=[]; % State transition matrix
        Bc=[]; % Input to state matrix
        Qc=[]; % State penalty
        Rc=[]; % Control action penalty
        xc=[]; % Current state value

        % Stage Constraints
        Ax=[];
        bx=[];
        Px=[];
        Au=[];
        bu=[];
        Pu=[];

        % Quadratic function
        H=[];
        F=[];

        % Quadprog constraints
        AU=[];
        bU=[];
        bUx=[];

        % Prediction horizon
        N=[];

        % Useful values
        nx=[];
        nu=[];

        % Regions
```

```
        Region=[];

        % Polyhedron
        Paux=[];

        % Controller matrices
        K=[];  % Overall feedback matrix
        P=[];  % Overall lyapunov function/cost-to-go

        % Control Signal
        Uc=[];

        % Historical data
        Xhist=[]; % State history
        Uhist=[]; % Input history
        coste=[]; % Cost history

        % Reference
        refx=[];

        % Representing options
        fontsize=[];
        linewidth=[];
    end


    methods
        function obj = ExpMPC(A, B, Ax, bx, Au, bu, Q, R, P, N)
            obj.Ac=A;
            obj.Bc=B;
            obj.Ax=Ax;
            obj.bx=bx;
            obj.Px=Polyhedron(Ax,bx);
            obj.Pu=Polyhedron(Au,bu);
            obj.Au=Au;
            obj.bu=bu;
            obj.Qc=Q;
            obj.Rc=R;
            if isempty(P)
                [K,P] = dlqr(A,B,Q,R,0*B);
                obj.K=-K;
                obj.P=P;
            end
            obj.N=N;
            obj.nx=length(A);
            obj.nu=size(B,2);
            obj.fontsize=16;
            obj.linewidth=2;

            % basicmpc
            nx=size(A,2); % number of states
            nu=size(B,2); % number of inputs

            % Future states can be written as X=Gx*x(0)+Gu*U
            for i=1:N
               Gx((i-1)*nx+1:i*nx,:)=A^i;
            end
```

```
    Gu=zeros(N*nx,N*nu);
    for i=1:N
       for j=1:i
          Gu((i-1)*nx+1:(i)*nx,(j-1)*nu+1:j*nu)=A^(i-j)*B;
       end
    end

    R_hat = kron(eye(N),R);

    aux=eye(N);
    aux(end,end)=0;
    aux2=zeros(N);
    aux2(end,end)=1;
    Q_hat=kron(aux,Q)+kron(aux2,P);

    % Build cost function
    obj.H=Gu'*Q_hat*Gu+R_hat;
    obj.F=(Gx'*Q_hat*Gu)';

    %  Aggregated U constraints
    if ~isempty(Ax)
        Ax_hat=kron(eye(N),Ax);
        bx_hat=kron(ones(N,1),bx);
        obj.AU=[obj.AU; Ax_hat*Gu];
        obj.bU=[obj.bU; bx_hat];
        obj.bUx=[obj.bUx;-Ax_hat*Gx];
    end

    if ~isempty(Au)
        Au_hat=kron(eye(N),Au);
        bu_hat=kron(ones(N,1),bu);
        obj.AU=[obj.AU; Au_hat];
        obj.bU=[obj.bU; bu_hat];
        if ~isempty(Ax)
            obj.bUx=[obj.bUx; zeros(length(bu_hat),size(Ax_hat*Gx,2))];
        else
            obj.bUx=[obj.bUx; 0*bu_hat];
        end
    end
    obj=obj.generate();
    obj=obj.plot();
end

function obj=generate(obj)
    G=obj.AU;
    S=obj.bUx;
    W=obj.bU;
    H=obj.H;
    F=obj.F;

    nr=size(G,1); % nr=num restricciones
    fprintf('MPC problem with %d constraints and a maximum of %d regions
        :\n\n', nr, 2^nr);
    warning('off','all')

    nreg=0; % region number
```

```
        for i=1:2^nr-1
            filaactiva=find(dec2bin(i-1,nr)-'0');
            filainact=setdiff(1:nr,filaactiva);

            % Activas
            Ga=G(filaactiva,:);
            Wa=W(filaactiva,:);
            Sa=S(filaactiva,:);

            if size(Ga,1) > 0 && rank(Ga) < size(Ga,1)
                continue
            else
                nreg=nreg+1;
                obj.Region(nreg).code=dec2bin(i-1,nr);
                fprintf('Generating Region %d (%s)...\n', nreg, dec2bin(i-1,
                    nr));
            end

            % Inactivas
            Gi=G(filainact,:);
            Wi=W(filainact,:);
            Si=S(filainact,:);

            auxinvHGa=H\Ga';
            auxinvHF=H\F;
            obj.Region(nreg).lambda0=-(Ga*auxinvHGa)\Wa;
            obj.Region(nreg).lambdax=-(Ga*auxinvHGa)\(Ga*auxinvHF+Sa);

            obj.Region(nreg).U0=H\(Ga'*((Ga*auxinvHGa)\Wa));
            obj.Region(nreg).U0(abs(obj.Region(nreg).U0)<(1e-10)) = 0;
            obj.Region(nreg).K=H\(Ga'*((Ga*auxinvHGa)\(Ga*auxinvHF+Sa))-F);
            obj.Region(nreg).K(abs(obj.Region(nreg).K)<(1e-10)) = 0;

            obj.Region(nreg).l0=obj.Region(nreg).lambda0;
            obj.Region(nreg).lx=-obj.Region(nreg).lambdax;

            obj.Region(nreg).r0=Wi-Gi*obj.Region(nreg).U0;
            obj.Region(nreg).rx=Gi*obj.Region(nreg).K-Si;

            obj.Region(nreg).Ar=[obj.Region(nreg).lx; obj.Region(nreg).rx];
            obj.Region(nreg).br=[obj.Region(nreg).l0; obj.Region(nreg).r0];
            obj.Region(nreg).P=Polyhedron(obj.Region(nreg).Ar,obj.Region(
                nreg).br);
            if any(any(isnan(obj.Region(nreg).P.H)))
                fprintf('Warning: Region with NaN in polytope\n');
            end
        end
        warning('on','all')
    end

    function obj=plot(obj,rango)
        if nargin==1
            rango=[];
            for i=1:length(obj.Region);
                if ~any(any(isnan(obj.Region(i).P.H))) && ~isempty(obj.Region
                    (i).P.V)
                    rango=[rango i];
```

```
                    end
                end
            end
            Paux=[];
            for i=rango
                Paux=[Paux obj.Region(i).P];
            end
            plot(Paux); obj.Paux=Paux;

         for i=rango
                try
                    coord = sum(obj.Region(i).P.V) / size(obj.Region(i).P.V,1);
                    text(coord(1),coord(2),sprintf('%d',i));
                catch
                end
            end
        end

        function HelpMethods(obj)
            disp('Still pending...');
        end

    end
end
```

Bibliografía

[1] E. F. Camacho and C. Bordons, *Model predictive control*. Springer Science & Business Media, 2013.

[2] E. Gustavsson. (2013, November) Lecture 5–6 — optimality conditions. TMA947 / MMG621 — Nonlinear optimisation. [Online]. Available: http://www.math.chalmers.se/Math/Grundutb/CTH/tma947/1314/lectures/lecture5.pdf

[3] J.-J. Xiong and E.-H. Zheng, "Position and attitude tracking control for a quadrotor uav," *ISA Transactions*, vol. 53, no. 3, pp. 725–731, 2014.

[4] W. Chamorro, P. Velarde, and G. Mosquera, "On the performance comparison between pi-lqr and soft-constrained mpc applied to a 4-tanks system," in *2017 International Conference on Information Systems and Computer Science (INCISCOS)*, 2017, pp. 115–120.

[5] M. French, *General Conditions for Solving Optimization Problems: Karush-Kuhn-Tucker Conditions*. Cham: Springer International Publishing, 2018, pp. 143–157.

[6] A. Bemporad, "Explicit model predictive control," in *Encyclopedia of Systems and Control*. Springer London, 2019, pp. 1–7.

[7] A. Alessio and A. Bemporad, "A survey on explicit model predictive control," in *Nonlinear Model Predictive Control: Towards New Challenging Applications*, L. Magni, D. M. Raimondo, and F. Allgöwer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 345–369.

[8] M. Kvasnica, J. Holaza, B. Takács, and D. Ingole, "Design and verification of low-complexity explicit mpc controllers in mpt3," in *2015 European Control Conference (ECC)*, 2015, pp. 2595–2600.

[9] M. Kvasnica and M. Fikar, "Clipping-based complexity reduction in explicit mpc," *IEEE Transactions on Automatic Control*, vol. 57, no. 7, pp. 1878–1883, 2012.

[10] A. Bemporad. (2009) Explicit model predictive control. Controllo di Processo e dei Sistemi di Produzione - A.a. 2008/09. [Online]. Available: https://pdfs.semanticscholar.org/7088/d63cacaf08f766023378914bf73207ccca1a.pdf

[11] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-parametric toolbox 3.0," in *Proc. of the European Control Conference*, Zürich, Switzerland, July 17–19 2013, pp. 502–510, http://control.ee.ethz.ch/~mpt.

[12] A. Sánchez Amores. (2018, December) Control predictivo de drone comercial. Universidad de Sevilla. [Online]. Available: https://idus.us.es/bitstream/handle/11441/85917/TFG-2171-SANCHEZ.pdf?sequence=1&isAllowed=y

[13] J. M. Maestre, D. Muñoz de la Peña, and E. F. Camacho, "Distributed model predictive control based on a cooperative game," *Optimal Control Applications and Methods*, vol. 32, no. 2, pp. 153–176, 2011.

[14] J. M. Maestre, "Distributed model predictive control based on game theory," Ph.D. dissertation, Universidad de Sevilla, Departamento de Sistemas y Automática, 10 2010.

[15] F. Farokhi, I. Shames, and K. H. Johansson, "Distributed MPC via dual decomposition and alternative direction method of multipliers," in *Distributed model predictive control made easy*. Springer, 2014, pp. 115–131.

[16] P. Giselsson, M. D. Doan, T. Keviczky, B. D. Schutter, and A. Rantzer, "Accelerated gradient methods and dual decomposition in distributed model predictive control," *Automatica*, vol. 49, no. 3, pp. 829–833, 2013.

[17] A. N. Venkat, J. B. Rawlings, and S. J. Wright, "Stability and optimality of distributed model predictive control," in *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005, pp. 6680–6685.

[18] D. Muñoz de la Peña, D. R. Ramírez, E. F. Camacho, and T. Alamo, "Application of an explicit min-max mpc to a scaled laboratory process," *Control Engineering Practice*, vol. 13, no. 12, pp. 1463–1471, 2005, special Section on Power Plants and Power Systems Control.

[19] D. Muñoz de la Peña, D. R. Ramírez, and T. Alamo, "Explicit solution of min–max mpc with additive uncertainties and quadratic criterion," *Systems & Control Letters*, vol. 55, p. 266–274, 04 2006.

[20] D. R. Ramirez and E. F. Camacho, "Piecewise affinity of min–max mpc with bounded additive uncertainties and a quadratic criterion," *Automatica*, vol. 42, no. 2, pp. 295–302, 2006.

[21] B. M. Åkesson and H. T. Toivonen, "A neural network model predictive controller," *Journal of Process Control*, vol. 16, no. 9, pp. 937–946, 2006.

[22] J. Katz, I. Pappas, S. Avraamidou, and E. N. Pistikopoulos, "Integrating deep learning and explicit mpc for advanced process control," in *2020 American Control Conference (ACC)*, 2020, pp. 3559–3564.

[23] J. Chen, Y. Chen, L. Tong, L. Peng, and Y. Kang, "A backpropagation neural network-based explicit model predictive control for dc–dc converters with high switching frequency," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 8, no. 3, pp. 2124–2142, 2020.

[24] J. Maestre, D. Muñoz de la Peña, A. Jiménez-Losada, and E. Algaba, "A coalitional control scheme with applications to cooperative game theory," *Optimal Control Applications and Methods*, vol. 35, 09 2014.

[25] F. Fele, J. Maestre, and E. Camacho, "Coalitional control: Cooperative game theory and control," *IEEE Control Systems Magazine*, vol. 37, no. 1, pp. 53–69, 2017.