ESTUDIO DE ROBUSTEZ FRENTE A RETARDOS Y PÉRDIDA DE DATOS DE UNA ESTRATEGIA DMPC BASADA EN POCOS CICLOS DE COMUNICACIÓN

Francisco Javier Muros Ponce
franmuros@gmail.com

José María Maestre Torreblanca
pepemaestre@cartuja.us.es

Eduardo Fernández Camacho
eduardo@esi.us.es

Departamento de Ingeniería de Sistemas y Automática Universidad de Sevilla
Camino de los Descubrimientos s/n, 41092, Sevilla, Spain

Resumen

Este artículo propone una estrategia de control MPC Distribuido que utiliza conceptos de teoría de juegos para realizar el trasvase de información entre los agentes, reduciendo lo máximo posible los ciclos de comunicación. Se ha estudiado el grado de robustez del algoritmo implementado frente a los errores en el canal de comunicaciones que originan retardos y pérdida de datos. Se han obtenido importantes resultados acerca de la máxima tasa de error en el canal soportada por la estrategia.

Palabras clave: MPC Distribuido, Teoría de Juegos, Retardos, Pérdida de Datos.

1. INTRODUCCIÓN

El Control Predictivo Basado en el Modelo, *(Model (Based) Predictive Control, MPC)*, también llamado control por horizonte deslizante *(Receding Horizon Control, RHC)*, constituye un campo muy amplio de métodos de control que integra diversas disciplinas como control óptimo, control estocástico, control de procesos con tiempos muertos, control multivariable o control con restricciones [2].

El Control Predictivo no es una estrategia de control específica, sino que se trata más bien de un campo muy amplio de métodos de control desarrollados en torno a ciertas ideas comunes. Los distintos algoritmos de MPC difieren entre sí casi exclusivamente en el modelo usado para representar el proceso y los ruidos y en la función de coste a minimizar. Aunque las diferencias puedan parecer pequeñas *a priori*, pueden provocar distintos comportamientos en bucle cerrado, siendo críticas para el éxito de un determinado algoritmo en una aplicación concreta [1].

El Control Predictivo es un tipo de control de naturaleza abierta dentro del cual se han desarrollado muchas implementaciones, encontrando gran aceptación tanto en aplicaciones industriales como en el mundo académico. El buen funcionamiento de estas aplicaciones muestra la capacidad del MPC para conseguir sistemas de control de elevadas prestaciones capaces de operar sin apenas intervención durante largos períodos de tiempo [1].

La teoría de juegos es un área de la matemática aplicada que utiliza modelos para estudiar interacciones en estructuras formalizadas de incentivos (los llamados juegos) y llevar a cabo procesos de decisión. Sus investigadores estudian las estrategias óptimas así como el comportamiento previsto y observado de individuos en juegos. Tipos de interacción aparentemente distintos pueden, en realidad, presentar estructuras de incentivos similares y, por lo tanto, representar

conjuntamente un mismo juego [7]. Últimamente se están aplicando los conceptos de teoría de juegos a problemas MPC.

El MPC Distribuido surgió como alternativa al MPC centralizado debido entre otras cosas a que en éste último los recursos requeridos de capacidad de cálculo y memoria son bastante altos. Por otro lado, en las aplicaciones prácticas el trasvase de información entre distintos módulos MPC no será ideal, sino que se verá afectado por una serie de problemas en la comunicación. Se puede dar el caso de que se pierdan datos en la transmisión. Esta pérdida de datos se verá aumentada evidentemente cuanto menor sea la capacidad del canal de comunicaciones y esto está relacionado con los problemas de retardos cuando dicha capacidad de la red de comunicación no permita a los agentes intercambiar información mientras resuelven sus problemas locales de optimización [4], [8], [15].

En este artículo vamos a proponer una modificación de la estrategia utilizada en [9], para que contemple también los problemas de retardos y pérdida de datos en las comunicaciones. El artículo se organiza de la siguiente manera. En el apartado 2 vamos a definir brevemente unos conceptos básicos de control MPC Distribuido y de teoría de juegos. En el apartado 3 expondremos la estrategia que se propone para contemplar los retardos y la pérdida de datos en la comunicación, detallando el algoritmo implementado en el apartado 4. En el apartado 5 se comentarán los resultados obtenidos mediante simulación. Finalmente, se expondrán las conclusiones alcanzadas con los estudios realizados en el apartado 6.

2. CONCEPTOS INICIALES

2.1. MPC Distribuido

La idea básica del DMPC *(Distributed MPC)* y la principal diferencia con MPC consiste en que en DMPC se utilizan varios agentes para resolver el problema de control dinámico. Es decir, consiste en descomponer el problema global en un conjunto de m subproblemas que se asignan a cada uno de los m agentes. Estos subproblemas dependerán unos de otros. Se asume implícitamente que cada agente contiene información de los demás agentes para resolver su subproblema. En el caso de que los agentes no tengan acceso a cierta información de otros agentes, es necesario bien que se produzca una comunicación entre agentes o bien que se establezcan algunas formas de predicción de la información necesaria.

Podemos asumir que:

$$J = \sum_i J_i, \tag{1}$$

donde J_i es el problema de optimización del agente i-ésimo. Este agente tendrá que resolver el siguiente problema de control MPC

$$\min_{U_i(k)} J_i(X_i(k), U_i(k), X_j(k), U_j(k)), j \in \eta \tag{2}$$

donde X_i y U_i son el estado y variable de actuación locales, respectivamente, y X_j y U_j representan el estado y la variable de actuación de los agentes que pertenezcan al subconjunto η, que engloba a aquellos agentes que interactúan con el agente i-ésimo.

Se pueden encontrar en la literatura diversas estrategias de MPC Distribuido, tales como DMPC con Restricciones de Estabilidad, *(Stability Constrained MPC, SC-DMPC)* [3], [4], [5], [6], MPC basado en Cooperación Viable *(Feasible Cooperation MPC, FC-MPC)* [13], [14], DMPC con Optimización de Vecindad [15], DMPC para Redes de Transporte [10], [11], [12], o el propio DMPC en Pocos Ciclos de Comunicación [9], analizado en este artículo.

2.2. Teoría de Juegos

Podemos definir Juego G como una interacción entre agentes o jugadores conscientes o lógicos, donde las decisiones de cada jugador pueden influir al comportamiento de los otros [7]. Un juego se define por:

- **Conjunto de agentes** A: Jugadores que toman decisiones para optimizar un servicio o función de coste. Un juego consta de al menos dos jugadores.
- **Espacio de Estrategias** S: Plan completo de acciones o estrategias que un jugador puede tomar durante el juego. Denotamos S_i como el espacio de estrategias del agente i-*ésimo*.
- **Ganancia** J: Representa la medida de la motivación de cada jugador por los posibles resultados o finales del juego. Es lo que cada agente recibe al final del juego, como resultado de haber tomado una estrategia determinada.

Supongamos que la teoría de juegos hace una única predicción sobre las estrategias elegidas por los jugadores. Para que esta predicción sea correcta es necesario que cada jugador esté dispuesto a elegir la estrategia predicha por la teoría. Por ello, es evidente que la estrategia predicha de cada jugador debe ser la mejor respuesta de dicho jugador a las estrategias predichas de los otros jugadores [7]. Llamaremos a tal predicción equilibrio de Nash:

En el juego en forma normal de m jugadores G, las estrategias $\{s_1^*, \ldots, s_m^*\}$ *forman un equilibrio de Nash (en estrategias puras) si, para cada jugador i,* s_i^* *es la mejor respuesta del jugador i (o al menos una de ellas) a las estrategias de los otros* $m-1$ *jugadores* $\{s_1^*, \ldots, s_{i-1}^*, s_{i+1}^*, \ldots, s_m^*\}$:

$$J(s_1^*, \ldots, s_{i-1}^*, s_i^*, s_{i+1}^*, \ldots, s_m^*) \geq J(s_1^*, \ldots, s_{i-1}^*, s_i, s_{i+1}^*, \ldots, s_m^*) \quad (EN) \tag{3}$$

para cada posible estrategia en S_i.

John Forbes Nash demostró que en cualquier juego finito existe al menos un equilibrio de Nash (en estrategias puras o mixtas) [7].

El óptimo de Pareto corresponde a la mejor estrategia conjunta que se podría conseguir, suponiendo que todos trabajan de forma que buscan el mayor bien común. Dicho óptimo de Pareto, puede pero no tiene por qué coincidir con el equilibrio de Nash.

Un conjunto de estrategias s está dominada en el sentido de Pareto por s' si y solo si

$$J_i(s_1', \ldots, s_m') \geq J_i(s_1, \ldots, s_m), \forall i \tag{4}$$

Es decir, un conjunto de estrategias s' es un óptimo de Pareto si y solo si no está dominada por ningún otro conjunto en ese sentido [7].

Evidentemente, lo ideal en un esquema cooperativo sería alcanzar el óptimo de Pareto, pero en muchas ocasiones no es posible y debemos conformarnos con llegar al equilibrio de Nash, pues para alcanzar el óptimo de Pareto todos los agentes deben trabajar de forma completamente altruista.

En este artículo se propone un algoritmo que intenta alcanzar el óptimo de Pareto, intentando minimizar lo máximo posible el número de intercambios de información entre los agentes en cada tiempo de muestreo.

3. PROBLEMAS EN LAS COMUNICACIONES

En este apartado vamos a proponer una modificación de la estrategia utilizada en [9] para que contemple también los problemas de retardos y pérdida de datos en las comunicaciones.

La clave del algoritmo de [9] es que el agente i-ésimo y sus vecinos eligen entre tres opciones para la señal de control (U_i^0, U_i^*, U_i^{wished}), escogiéndose aquella combinación de las mismas que minimice de forma global J. En la figura 1 se observan las 9 posibilidades de J_i para el caso de 2 jugadores.

A1\A2	U_2^0	U_2^*	U_2^{wished}
U_1^0	$J_1(U_1^0,U_2^0)$ $J_2(U_1^0,U_2^0)$	$J_1(U_1^0,U_2^*)$ $J_2(U_1^0,U_2^*)$	$J_1(U_1^0,U_2^w)$ $J_2(U_1^0,U_2^w)$
U_1^*	$J_1(U_1^*,U_2^0)$ $J_2(U_1^*,U_2^0)$	$J_1(U_1^*,U_2^*)$ $J_2(U_1^*,U_2^*)$	$J_1(U_1^*,U_2^w)$ $J_2(U_1^*,U_2^w)$
U_1^{wished}	$J_1(U_1^w,U_2^0)$ $J_2(U_1^w,U_2^0)$	$J_1(U_1^w,U_2^*)$ $J_2(U_1^w,U_2^*)$	$J_1(U_1^w,U_2^w)$ $J_2(U_1^w,U_2^w)$

Figura 1: Posibles J_i para el caso de 2 jugadores

Vamos a suponer por simplicidad que disponemos de un sistema con dos agentes o jugadores. En cada tiempo de muestreo k cada agente resuelve dos problemas de optimización para obtener U_i^* y U_{nei}^{wished}, esto es, su acción de control óptima y la acción de control que éste desearía que tuviera el vecino. Para resolver dichos problemas de optimización, el único dato que recibe cada agente de su vecino es U_{nei}^0, la acción de control inicial del vecino. Nosotros proponemos introducir errores en las comunicaciones para alterar ese dato que cada agente recibe del vecino en cada tiempo de muestreo k.

En [9] se utilizan dos parámetros que dan una medida de la operatividad del algoritmo propuesto en comparación con control MPC centralizado. Se trata de

- λ: Indica el radio de convergencia de la función de coste global. En otras palabras mide el grado de mejora de la función de coste final con respecto a la inicial. Si $\lambda < 1$ significa que los agentes son más felices al final que al principio, y serán más felices cuanto menor sea este valor de λ. Se define λ como

$$J_{final} = J_{inicial} \cdot \lambda^k, \lambda > 0 \tag{5}$$

Siempre que obtengamos $\lambda > 1$, significa que nuestro sistema es inestable.

- J_v: Indica el número de tiempos de muestreo k que se requieren para conseguir que el error relativo medio de cada subsistema sea menor al 5 %, definiendo el error relativo como

$$E_{ri} = \left| \frac{ref_i - y_i}{ref_i} \right| \cdot 100 \tag{6}$$

Se adopta el siguiente convenio para dos agentes

$$J_v \rightarrow E_{r1} + E_{r2} < 10\,\% \tag{7}$$

Evidentemente, si $J_v = k_{max}$ no hemos conseguido un error por debajo de la cota propuesta con el número de tiempos de muestreo empleado. Esto no significa que el sistema no converja. Simplemente quiere decir que para obtener más información habría que aumentar el número de tiempos de muestreo.

En los estudios realizados en [9], se realiza una batería de simulaciones tomando $k = 40$ tiempos de muestreo. Para el caso ideal de control MPC centralizado resulta $\lambda = 0.77$ y $J_v = 13$, mientras que para el algoritmo DMPC propuesto se obtiene $\lambda = 0.90$ y $J_v = 40$. Se observa que, aunque lógicamente se empeora en el segundo caso con respecto al control centralizado, se consiguen resultados muy satisfactorios con la técnica propuesta.

Nosotros, como hemos comentado anteriormente, vamos a introducir errores en la comunicación del dato que se recibe del vecino en cada tiempo de muestreo: U^0_{nei}.

Se introduce el siguiente parámetro en el algoritmo

- $fiabilidad$: mide la probabilidad de que se reciba el dato correctamente en un tiempo de muestreo k concreto. Su rango de variación será $fiabilidad \in [0, 1]$, significando un 0 errores en todas las transmisiones y un 1 ninguna transmisión con errores.

En caso de que tengamos error de transmisión en un cierto tiempo de muestreo k, cada agente no utiliza U^0_{nei} calculado por el vecino en dicho tiempo de muestreo, sino el dato proporcionado por el vecino en el último tiempo de muestreo en el que hubo transmisión efectiva, y esto equivale a recibir el dato con un cierto retraso variable. Por otro lado, es importante tener en cuenta que, en cada iteración con error de comunicación, ambos agentes van a actuar de forma egoísta, debido a que no van a tener en cuenta la información proveniente del otro. Es decir, de los 9 posibles valores de J_i según la tabla de la figura 1, van a escoger la opción 5, en la que ambos calculan su control de forma independiente.

4. ALGORITMO

El algoritmo implementado, que se detalla a continuación, usa una estrategia paralelo-síncrona. Se supone que los agentes tienen una visión incompleta del sistema global. Se conoce cómo influyen en las variables de salida y también se supone que los agentes conocen cómo los agentes vecinos influencian en sus propias variables.

1. Recibir el dato de los vecinos
   - *a*) Si no hay error de transmisión, se recibe $U^{0^{(k)}}_{nei}$.
   - *b*) Si hay error de transmisión, se utiliza la última transmisión sin errores $U^{0^{(k-delay)}}_{nei}$.
2. El agente i-ésimo resuelve el siguiente problema

$$\begin{array}{l} \min_{U_i(k)} J_i \left(X_i(k), U_i(k), X_{nei}(k), U^0_{nei} = \left\{ \begin{array}{c} U^{0^{(k)}}_{nei} \\ U^{0^{(k-delay)}}_{nei} \end{array} \right. \right) \\ s.a. \end{array}$$

$$\begin{array}{c} x_i(k+j+1|k) = F_i(x_i(k+j|k), u_i(k+j|k) \\ x_{nei}(k+j|k), u^0_{nei}) \\ G_i(X_i(k), U_i(k), X_{nei}(k), U^0_{nei}) \leq 0 \end{array}$$

   De esta manera, el agente i-ésimo obtiene U^*_i.
3. El agente i-ésimo resuelve el siguiente problema

$$\begin{array}{l} \min_{U_{nei}(k)} J_i(X_i(k), U^*_i, X_{nei}(k), U_{nei}(k)) \\ s.a. \end{array}$$

$$\begin{array}{c} x_i(k+j+1|k) = F_i(x_i(k+j|k), u^*_i(k+j|k) \\ x_{nei}(k+j|k), u_{nei}(k+j|k)) \\ G_i(X_i(k), U^*_i, X_{nei}(k), U_{nei}(k)) \leq 0 \end{array}$$

   Notar que, actuando de esta forma, el agente i-ésimo obtiene U^{wished}_{nei}, esto es, la actuación deseada para sus vecinos.
4. El agente i-ésimo comunica los datos U^*_i y U^{wished}_{nei} a sus vecinos. Aquí se debe entender a los vecinos como aquellos agentes para los cuales estos datos son relevantes. También comunica el coste asociado desde su punto de vista a dichos valores.

5. Para los datos recibidos, el agente i-ésimo calcula los posibles valores de su función de coste J_i y comunica estos valores al resto de los agentes del sistema.

6. El agente i-ésimo recibe la información y calcula la función de coste completa para todas sus opciones, esto es, las consecuencias globales de su acción medidas en términos de utilidad. El agente i-ésimo elige entre U_i^0, U_i^* y U_i^{wished} la opción que minimiza J. Notar que podría haber más de un valor para U_i^{wished}. En el caso de error de comunicación, se tendrá J mínima para el supuesto de 2 agentes con la opción de agentes egoístas (opción 5 de la tabla de la figura 1).

7. Aplicar la señal de control $U(k)$.

8. Mover el horizonte al siguiente tiempo de muestreo, $k+1 \rightarrow k$, y volver al paso 1.

Como se observa, este algoritmo no requiere un proceso iterativo para cumplir una condición terminal en cada tiempo de muestreo, a diferencia de las otras propuestas analizadas.

Se puede hacer un análisis cualitativo para las soluciones propuestas en cada paso por el agente i-ésimo:

- U_i^0**: Opción estable**. El agente tiene la posibilidad de mantener su actuación. Esta puede ser una buena opción porque $U_j^*(j \neq i)$ se calcula suponiendo que $U_i = U_0$. Esto también permite al sistema permanecer globalmente estable una vez que se ha alcanzado un mínimo global en J.

- U_i^***: Opción egoísta**. Esta opción ofrece una mejora en J_i si el resto de las variables manipuladas del sistema permanecen sin cambiar. Esta puede ser una buena elección para reducir J si el resto de los agentes han alcanzado una actuación de estado estable.

- U_i^{wished}**: Opción altruista**. Esta opción ofrece la máxima mejora para el agente j-ésimo cuyo deseo es $U_i = U_i^{wished}$ cuando alcance U_j^*. El agente i-ésimo sacrifica su propio bienestar para satisfacer al agente j-ésimo.

Suponiendo que hay N agentes en el sistema, hay que recordar que en el peor caso cada agente debería construir una matriz N-dimensional. De todas formas esto puede simplificarse porque no todas las variables son relevantes para todos los agentes. Además no todas las posibles opciones de cooperación se emplean con la misma frecuencia, así que podrían dejarse de calcular las opciones poco frecuentes para simplificar los cálculos.

Por otro lado, se debe garantizar la convergencia para el algoritmo. Como se puede ver en [8], [15] la condición de convergencia en el caso de control DMPC lineal es:

$$|\rho(D_0)| < 1, \tag{8}$$

donde D_0 denota el incremento de control entre dos pasos del algoritmo de convergencia $\Delta u^{l+1}(k) = D_0 \Delta u^l(k)$, y $\rho(.)$ indica el radio espectral de la matriz.

5. SIMULACIONES

Para las simulaciones se ha escogido el mismo sistema empleado en [9], proveniente a su vez de un ejemplo encontrado en [2]. Se trata de un sistema MIMO 2x2 que modela un reactor de tanque con agitación, descrito por la siguiente matriz de transferencia

$$\begin{bmatrix} Y_1(s) \\ Y_2(s) \end{bmatrix} = \begin{bmatrix} \frac{1}{1+0.7s} & \frac{5}{1+0.3s} \\ \frac{1}{1+0.5s} & \frac{2}{1+0.5s} \end{bmatrix} \cdot \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix}, \tag{9}$$

donde las variables manipuladas o entradas son respectivamente el flujo de alimentación (*feed*) y el flujo del refrigerante de la sobrecubierta (*coolant*). Las variables controladas o salidas son respectivamente la concentración del vertido (*effluent*) y la temperatura del reactor. Se puede observar un gráfico ilustrativo del sistema en la figura 2.

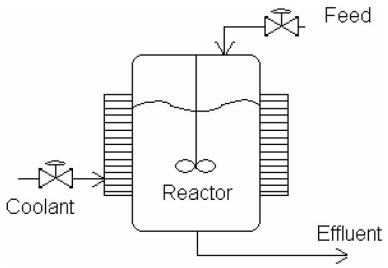

Figura 2: Reactor de tanque con agitación

Cada agente tiene una visión incompleta del sistema. De hecho cada uno sólo conoce una fila del modelo global, esto es, cómo ellos influencian en sus variables de salida y cómo éstas son perturbadas por el otro agente. Por tanto, los modelos del agente 1 y 2 son, respectivamente

$$[Y_1(s)] = \begin{bmatrix} \frac{1}{1+0.7s} & \frac{5}{1+0.3s} \end{bmatrix} \cdot \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix} \tag{10}$$

$$[Y_2(s)] = \begin{bmatrix} \frac{1}{1+0.5s} & \frac{2}{1+0.5s} \end{bmatrix} \cdot \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix} \quad (11)$$

En nuestro estudio vamos a escoger un número de tiempos de muestreo superior al utilizado en [9], pues lógicamente al introducir los errores de comunicación en el sistema, la convergencia del mismo será más lenta. Estudiaremos los casos $k = 100$ y $k = 300$.

Primeramente se ha realizado una batería de simulaciones para conocer qué valores toman los parámetros λ y J_v para el caso ideal sin errores. Se han obtenido los valores medios que aparecen en la tabla 1 con el parámetro $fiabilidad = 1$.

Tabla 1: λ y J_v para $fiabilidad = 1$

$k = 100$	$J_v = 41.0125$	$\lambda = 0.8858$
$k = 300$	$J_v = 41.1458$	$\lambda = 0.9578$

Con estos valores, el sistema alcanza la evolución de entradas, salidas y función de coste que se observa en la figura 3.

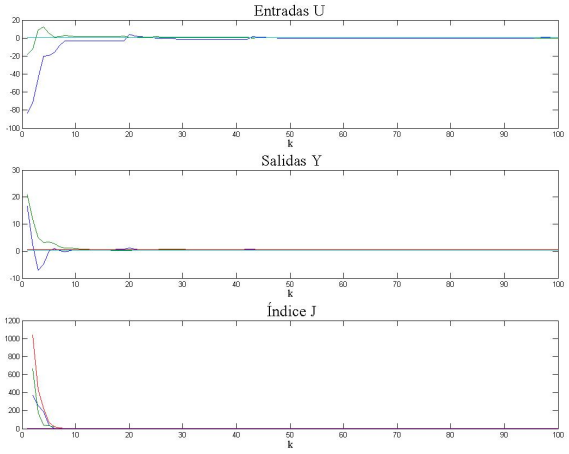

Figura 3: Evolución del sistema sin pérdida de datos

Si vamos degradando la comunicación bajando gradualmente el valor del parámetro $fiabilidad$, obtenemos los valores de las tablas 2 y 3.

Tabla 2: λ y J_v para $fiabilidad \in [0.5, 0.9]$

	$fiabilidad$				
	0.9	0.8	0.7	0.6	0.5
$\lambda_{k=100}$	0.8929	0.8976	0.9098	0.9315	0.9902
$J_{v\,k=100}$	44.6375	48.025	64.1375	76.575	84.15
$\lambda_{k=300}$	0.9598	0.9599	0.9625	0.969	0.984
$J_{v\,k=300}$	45.375	52.8125	71.3125	90.9375	129.625

Tabla 3: λ y J_v para $fiabilidad \in [0.1, 0.4]$

	$fiabilidad$			
	0.4	0.3	0.2	0.1
$\lambda_{k=100}$	1.1767	1.3917	1.6086	1.9318
$J_{v\,k=100}$	98.8625	100	100	100
$\lambda_{k=300}$	1.1604	1.3818	1.6041	1.9235
$J_{v\,k=300}$	291.0313	300	300	300

Usando los valores anteriores se consigue la gráficas de la figura 4 donde se representa la degradación de los parámetros J_v y λ con el aumento del parámetro $fiabilidad$.

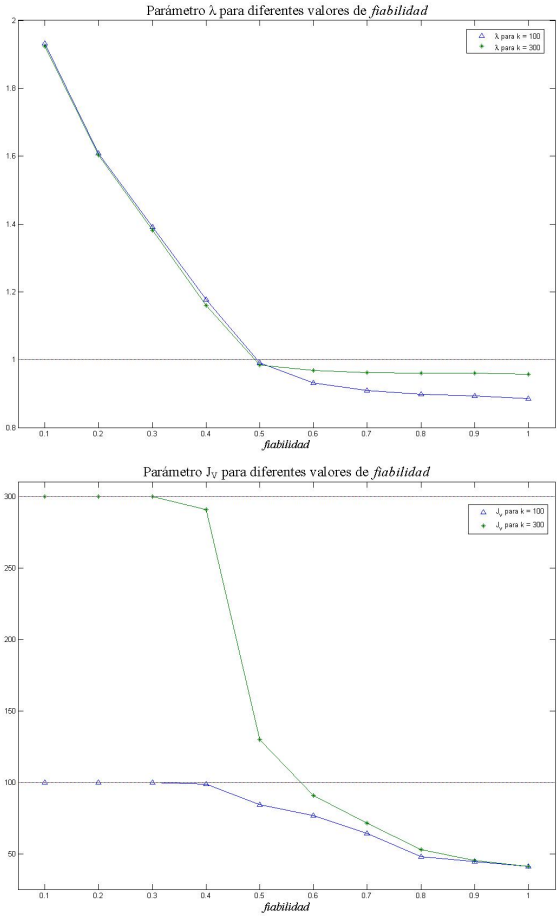

Figura 4: Degradación de λ y J_v con el parámetro $fiabilidad$

Podemos apreciar un aumento de λ al aumentar k, cuando el sistema es estable ($\lambda < 1$). Esto se debe al hecho de que si un sistema ha alcanzado prácticamente la convergencia en k_1 tiempos de muestreo, la evolución en los k_2 tiempos de muestreo siguientes no va a ser significativa, por tanto el parámetro λ_{k_1} que mide el grado de eficiencia o mejora del sistema en k_1 tiempos de muestreo va a ser mejor (más pequeño) que el parámetro global $\lambda_{k_1+k_2}$. Para valores de $\lambda > 1$ (sistema inestable)

no se aprecian diferencias significativas al variar k.

Por otro lado, también se observa un aumento de J_v con k. Esto es lógico, pues en $J_{v_{media}}$ influirán muestras que convergen y que no, y para éstas últimas, $J_v = k_{max}$, pues nunca se alcanza el error mínimo deseado. Evidentemente, a mayor fiabilidad, más porcentaje de muestras convergerán.

Para valores de $fiabilidad$ que hagan $\lambda < 1$, el sistema es capaz de controlar como vemos en la gráfica de la figura 5, donde $fiabilidad = 0.5$.

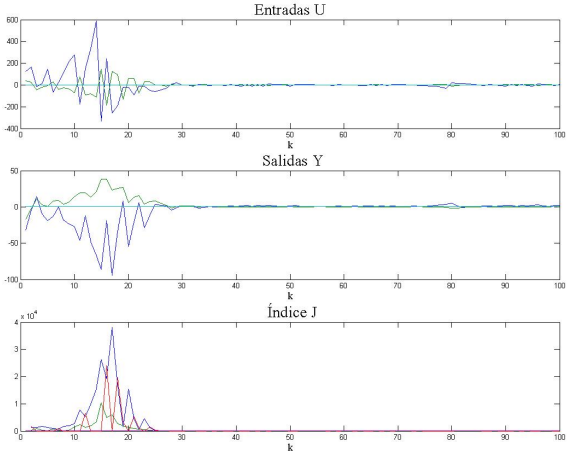

Figura 5: Evolución del sistema con $fiabilidad = 0.5$

Sin embargo para valores de $fiabilidad$ tales que $\lambda > 1$ el sistema se inestabiliza, tal y como se aprecia en la figura 6, en donde se ha tomado $fiabilidad = 0.2$.

Se han obtenido *valores medios* de $\lambda > 1$ para $fiabilidad < 0.5$. Esto no implica que ninguna muestra converja para $fiabilidad < 0.5$, o que siempre se logre convergencia para $fiabilidad \geq 0.5$, pues hay que recordar que hablamos de resultados medios. Por tanto, y a la vista de los resultados obtenidos, podemos concluir que el sistema es capaz de soportar una tasa media de pérdida de datos del 50 %, y aún así continúa alcanzando la estabilidad.

6. CONCLUSIONES

En este artículo se ha diseñado una modificación de una estrategia DMPC basada en pocos ciclos de comunicación para incluir en su

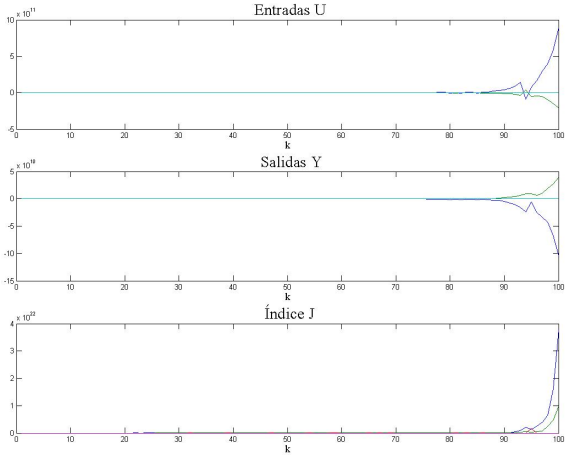

Figura 6: Evolución del sistema con $fiabilidad = 0.2$

algoritmo problemas de retardos y pérdida de datos en el canal. Se ha analizado la robustez del sistema degradando poco a poco las prestaciones del mismo mediante un aumento en la pérdida de datos, lo que conlleva la recepción del dato de los vecinos con un retardo variable. Para el sistema bajo estudio, la tasa de error del 50 % es el caso límite en media para el cual la estrategia propuesta deja de controlar al sistema.

Referencias

[1] Bordóns, C., (2000) **Control Predictivo: metodología, tecnología y nuevas perspectivas**. *Technical report. I Curso de Especialización en Automática. Aguadulce, Almería.*

[2] Camacho, E. F. y Bordóns, C., (1995) **Model Predictive Control in the Process Industry**. *Ed. Springer-Verlag, Berlin, Germany.*

[3] Camponogara, E., (2001) **Controlling Networks with Collaborative Nets**. *PhD Thesis.*

[4] Camponogara, E., Jia, D., Krogh, B. H. y Talukdar, S., (2002) **Distributed Model Predictive Control**. *IEEE Control Systems Magazine.*

[5] Cheng, X. y Krogh, B. H., (1997) **Stability Constrained Model Predictive Control**

for Nonlinear Systems. *Proc. of the 36th IEEE CDC San Diego, Ca.*

[6] Cheng, X. y Krogh, B. H., (2001) **Stability-Constrained Model Predictive Control**. *IEEE transactions on Automatic Control, Vol 46, Nº 11.*

[7] Gibbons, R. Traducido por Calvo, P. y Vilà, X., (1997) **Un Primer Curso de Teoría de Juegos**. *Ed. Antoni Bosch. Barcelona.*

[8] Li, S., Zhang, Y. y Zhu, Q., (2005) **Nash-optimization enhanced distributed model predictive control applied to the Shell benchmark problem**. *Elsevier. Information Sciences 170, 329-349.*

[9] Maestre, J. M. y Camacho, E. F., (2008) **Distributed Model Predictive Control in Few Communication Cycles**. *Enviado a CDC.*

[10] Negenborn, R. R., De Schutter, B. y Hellendoorn, J., (2006) **Multi-agent model predictive control for transportation networks with continuous and discrete elements**. *Proceedings of the 11th IFAC Symposium on Control in Transportation Systems, Delft, The Netherlands, pp. 609-614.*

[11] Negenborn, R. R., (2007) **Multi-Agent Model Predictive Control with Applications to Power Networks**. *PhD Thesis.*

[12] Negenborn, R. R., De Schutter, B. y Hellendoorn, J., (2007) **Multi-agent model predictive control for transportation networks: Serial versus parallel schemes**. *Elsevier. Engineering Applications of Artificial Intelligence 21, 353-366.*

[13] Venkat, A. N., Rawlings, J. B. y Wright., S. J., (2005) **Stability and optimality of distributed model predictive control**. *44th IEEE Conference on Decision and Control, and the European Control Conference. Seville, Spain.*

[14] Venkat, A. N., (2006) **Distributed Model Predictive Control: Theory and Applications**. *PhD Thesis.*

[15] Zhang, Y. y Li, S., (2007) **Networked model predictive control based on neighbourhood optimization for serially connected large-scale processes**. *Elsevier. Journal of Process Control 17, 37-50.*