



UNIVERSIDAD POLITÉCNICA DE CATALUÑA  
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial

# ANÁLISIS DEL RENDIMIENTO DE UN SISTEMA ROBÓTICO DE CONTROL EN RED BASADO EN SIMULACIÓN

---

PROYECTO PRESENTADO PARA OPTAR AL TÍTULO DE

MÁSTER EN AUTOMÁTICA Y ROBÓTICA

POR

DENNYS PAILLACHO CHILUIZA

DIRECTORA: ALÍCIA CASALS GELPÍ

BARCELONA, JUNIO DE 2010

**PROYECTO FIN DE MASTER**

Autor:

Dennys Paillacho Chiluza

Directora:

Alícia Casals Gelpí

Tribunal de evaluación:

[NOMBRE DEL PRESIDENTE]

[NOMBRE DEL VOCAL]

[NOMBRE DEL SECRETARIO]

## Resumen

Debido a la dificultad de evaluar el rendimiento global de un sistema robótico de control en red de manera analítica, el presente trabajo trata el análisis del comportamiento de un sistema robótico de control en red, tanto temporal como espacial, basado en simulación.

El entorno experimental de sistema robótico de control en red está constituido por un robot manipulador planar con articulaciones angulares, donde el control de movimiento del robot está distribuido en varios controladores remotos, uno para cada articulación, y un controlador principal, los mismos que se comunican por medio de una red de control.

Para simplificar el análisis del modelo del sistema robótico, se ha elegido un modelado por capas, el mismo que permite definir más fácilmente las interacciones entre los componentes del sistema (controlador principal, red de control y controladores remotos).

El modelo del sistema robótico es llevado a un entorno de simulación y puesto a prueba con distintas configuraciones (como el tipo de red de control, tasa de datos, tamaño de datos y número de controladores remotos). De los resultados obtenidos se hace un análisis tanto temporal como espacial del ciclo de comunicación y del error de seguimiento de trayectoria, respectivamente.

## Contenido

Resumen .....	3
Contenido .....	4
1 Introducción y objetivos .....	6
1.1 Introducción .....	6
1.2 Justificación .....	7
1.3 Objetivos.....	7
Objetivo principal .....	7
Objetivos específicos .....	7
1.4 Descripción del proyecto.....	8
1.5 Metodología.....	9
1.6 Cronograma de actividades .....	9
2 Sistemas de control en red .....	11
2.1 Clasificación de los NCS.....	11
2.2 Componentes de un NCS .....	13
2.2.1 Red de control .....	13
2.2.2 Controladores .....	19
2.2.3 Sensores y actuadores .....	19
2.3 Tiempos de retardo en un NCS .....	19
Tiempo de preprocesamiento ( $T_{pre}$ ) y posprocesamiento ( $T_{pos}$ ) .....	20
Tiempo de transmisión en la red ( $T_{red}$ ) .....	20
Tiempo de bloqueo .....	21
2.4 Control de movimiento.....	21
2.4.1 Sistema de control de movimiento en red (motion NCS).....	21
2.4.2 Control de movimiento de un robot manipulador.....	23
2.4.3 Control de articulaciones independiente .....	23
2.5 Evaluación del rendimiento .....	25
2.5.1 Calidad de servicio de la red.....	25
2.5.2 Calidad de rendimiento del control.....	26
3 Modelo del sistema propuesto .....	28
3.1 Formulación del problema .....	29
3.2 Ciclo de comunicación.....	30
3.3 Cinemática del robot.....	32
4 Simulación .....	34

4.1	Parámetros de simulación .....	34
4.1.1	Caso a: análisis temporal del ciclo de comunicación .....	34
4.1.2	Caso b: análisis espacial del seguimiento de la trayectoria deseada .....	36
4.2	Generador de trayectorias deseadas .....	36
5	Análisis de resultados .....	39
5.1	Ciclos de comunicación .....	39
5.1.1	Diferentes tamaños de datos (TD). R=1, CR=6 .....	39
5.1.2	Diferentes tasas de datos (R). CR=6, TD=12.....	40
5.1.3	Diferentes números de controladores remotos (CR). R=1, TD=12.....	41
5.2	Seguimiento de trayectoria .....	42
5.2.1	Diferentes tamaños de datos (TD). R=1, CR=6 .....	42
5.2.2	Diferentes tasas de datos (R). CR=6, TD=12.....	43
5.2.3	Diferentes números de controladores remotos (CR). R=1, TD=12.....	43
6	Conclusiones y futuros trabajos .....	44
7	Bibliografía.....	45

# 1 Introducción y objetivos

## 1.1 Introducción

Los Sistemas de Control en Red (*Networked Control Systems - NCS*) forman parte de un área emergente de los Sistemas de Control Distribuido (*Distributed Control Systems - DCS*), dada la gran integración y convergencia de las comunicaciones, la informática y el control automático. En general, un NCS está compuesto por controladores, actuadores y sensores cuyas operaciones están distribuidas en diferentes ubicaciones geográficas y coordinadas a través del intercambio de mensajes sobre una red de comunicación [1]. Los NCS son ampliamente utilizados en la industria, automoción, aviación, transporte eléctrico, robots, etc.

Los estudios de los Sistemas de Control en Red - NCS involucran el diseño del controlador, la planificación del CPU y la planificación de la red de comunicación. En el pasado, el diseño de los controladores y la planificación (tanto del CPU y de la red) normalmente se han estudiado por separado [2] [3]. Esta separación ha permitido a la comunidad de control enfocarse en su propio dominio del problema sin preocuparse sobre cómo la planificación es realizada, y también ha liberado a la comunidad de planificación de la necesidad de entender el impacto que la planificación tiene en la estabilidad y rendimiento de la planta bajo control. Sin embargo, cuando los dos diseños están combinados en un NCS, algunas hipótesis no son verdaderas [3], lo que genera un impacto en la medida del rendimiento del NCS resultante. Una interacción más estrecha entre estos campos es necesaria para las aplicaciones de control que requieren un alto grado de flexibilidad, especialmente cuando los recursos de cómputo (por ej.: el tiempo del CPU y el ancho de banda de la red) son limitados. Adicionalmente, la interacción entre el sistema de control y la red de comunicación deben ser considerados para un uso más efectivo de los recursos de comunicación y así minimizar los retardos en el proceso de diseño [4]. Es necesario un apropiado protocolo de envío de mensajes para garantizar la calidad del servicio de la red QoS (por sus siglas en inglés - Quality of Service), además es deseable el diseño de un controlador avanzado para garantizar la calidad del rendimiento QoP (Quality of Performance). Dada la estrecha relación entre el control y la red de comunicación en un NCS, es importante considerar ambos parámetros simultáneamente para asegurar tanto la QoS y la QoP.

Aunque el análisis y modelamiento de sistemas con tiempos de retardo ha sido un área progresiva de investigación [5], los impactos de la planificación y comunicación en el rendimiento del sistema son a menudo muy difíciles, sino imposibles de estudiarlos analíticamente [6]. El presente trabajo utiliza una aproximación basada en simulación para el análisis del rendimiento de un tipo de Sistema de Control en Red, al cual se lo ha denominado Sistema Robótico de Control en Red.

El trabajo está organizado de la siguiente manera. En el Capítulo 2 se hace una descripción de los Sistemas de Control en Red y sus componentes. El Capítulo 3 trata el diseño del modelo de un Sistema Robótico de Control en Red. Los parámetros de simulación y las diferentes configuraciones son presentados en el Capítulo 4. Basados en los resultados de las simulaciones, dos métricas para evaluar el rendimiento del sistema son analizadas y discutidas en el Capítulo 5. Conclusiones y futuros trabajos se presentan finalmente en el Capítulo 6.

## 1.2 Justificación

A medida que existen más y más aplicaciones de Sistemas de Control en Red - NCS en la industria, el transporte, comunicaciones, aviación, robótica, vuelos espaciales, etc, más atención ha sido puesta en el diseño y análisis de los NCS [7].

En comparación con un sistema de control convencional con interconexión punto-a-punto, las principales ventajas de un NCS son: diseño modular y flexible, rápida implementación, fácil diagnóstico y mantenimiento, y aumento en la agilidad del sistema [8].

El cambio en la arquitectura de comunicación punto-a-punto al de una red, sin embargo, introduce diversos tiempos de retardo entre sensores, actuadores y controladores. Estos tiempos de retardo provienen del tiempo compartido del medio de comunicación así como del tiempo de cómputo requerido para procesar los datos. Las características de los tiempos de retardo pueden ser constantes, delimitadas o incluso aleatorias. Es muy conocido que en los NCS, los tiempos de retardo pueden degradar el rendimiento del sistema e inclusive causar inestabilidades. El análisis sobre los efectos de los tiempos de retardo en el rendimiento del sistema puede ser utilizado para el desarrollo de metodologías apropiadas de control en red [9] [10].

Generalmente, hay tres métodos de estudio de los NCS: Tipo 1 – métodos de planificación que garanticen la calidad del servicio (QoS) [11]; Tipo 2 – métodos de control que garanticen la calidad del rendimiento (QoP - quality of performance) [12]; y Tipo 3 – métodos integrados de planificación y control que consideren tanto la QoS y la QoP [13]. En el diseño y análisis integral de un NCS (del Tipo 3), dada la complejidad del sistema global, suelen utilizarse herramientas de simulación [6] [14] [15] [16] [17].

El análisis del comportamiento de un sistema robótico de control en red servirá de base para verificar la factibilidad de un determinado diseño, como paso previo a su implementación. Esto, a futuro, permitirá la construcción de sistemas robóticos mucho más flexibles, escalables y robustos.

## 1.3 Objetivos

### Objetivo principal

Realizar un estudio temporal y espacial del rendimiento de un sistema robótico de control en red considerando diferentes parámetros de configuración del modelo del sistema.

### Objetivos específicos

- Hacer un estudio general de los Sistemas de Control en red.
- Modelar un Sistema Robótico de Control en Red flexible y escalable.
- Recrear el modelo en base a distintos parámetros de configuración por medio de simulación.
- Analizar los resultados para medir el comportamiento global del sistema tanto a nivel temporal (ciclo de comunicación) como a nivel espacial (error de seguimiento).
- Redactar un reporte técnico sobre el proyecto.

## 1.4 Descripción del proyecto

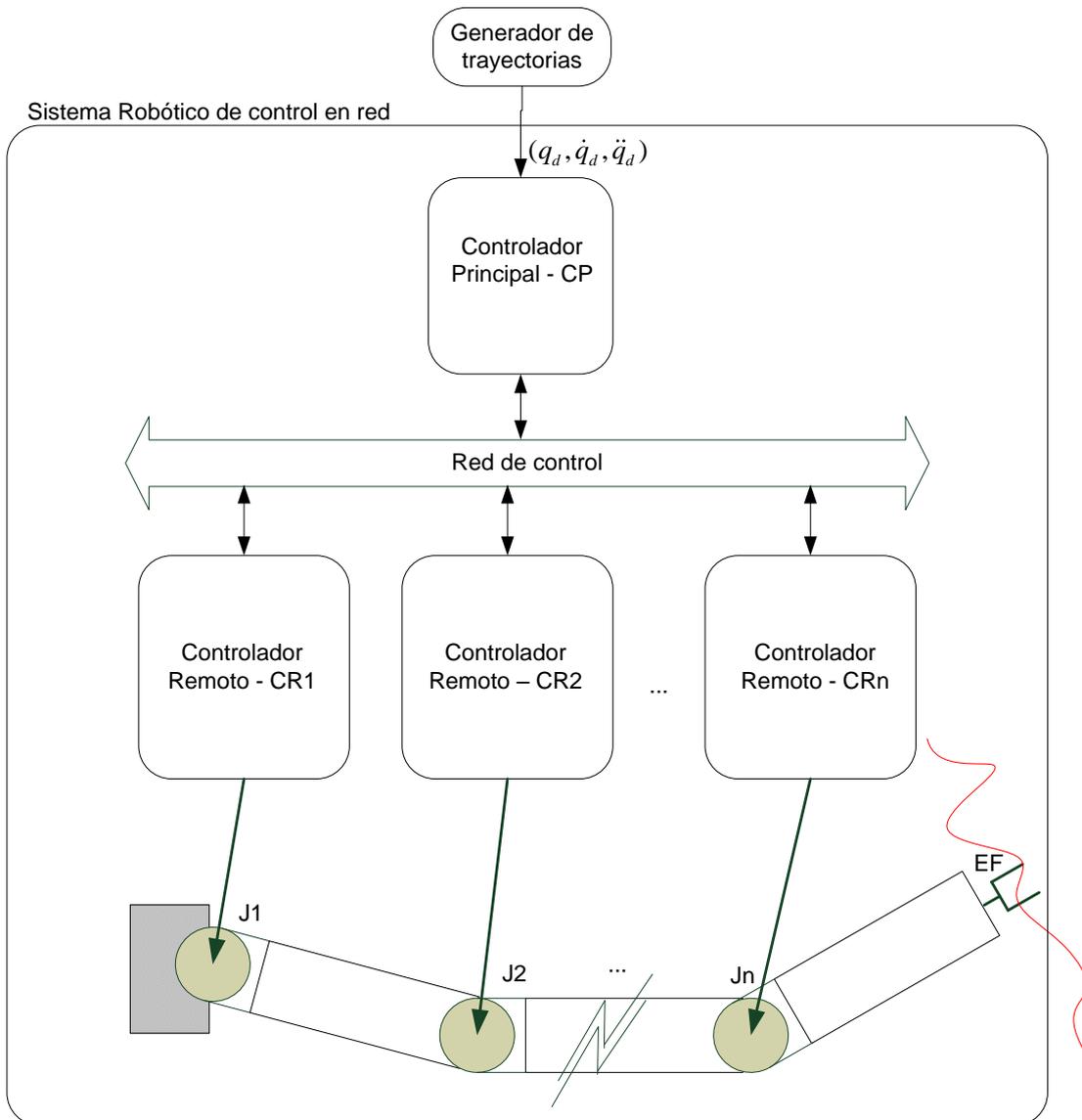


Figura 1.1 Esquema general de un Sistema Robótico de Control en Red.

En la Fig.1.1 podemos observar los principales componentes del sistema robótico de control en red. El controlador principal (CP) será el encargado de enviar las consignas o señales de referencia a los controladores remotos de movimiento; la red de comunicación es el medio por el cual se comunican los dispositivos. Los controladores remotos de movimiento son los encargados de realizar el lazo de control de movimiento de las consignas enviadas por el controlador principal. Cabe recalcar que fuera del sistema está el bloque Generador de trayectorias, el mismo que genera las trayectorias de posición, velocidad y aceleración en base a una función de tiempo conocida.

La primera parte del proyecto hace un breve repaso sobre los tipos de sistemas de control en red, las características de los tiempos de retardo, métricas utilizadas para medir el rendimiento del sistema.

Se especifica luego el diseño del modelo del sistema en base a un modelado por capas, que sirve para definir las interrelaciones entre los componentes del sistema.

En base a una herramienta de simulación, y definiendo ciertos parámetros y restricciones de simulación, varias configuraciones, que definen el tipo de red, tasa de datos, tamaño de datos y número de controladores, son recreadas en el modelo del sistema propuesto.

Finalmente se analizan los resultados de la simulación, desde dos enfoques distintos, el primero desde el punto de vista temporal, es decir, sobre los tiempos de retardo del sistema; y desde el punto de vista espacial, es decir, analizando el error generado entre la trayectoria deseada y la generada por el sistema.

## 1.5 Metodología

Para cumplir con los objetivos trazados, el presente trabajo se ha dividido en 6 fases: I) Revisión bibliográfica, II) Modelo del Sistema, III) Simulación, IV) Análisis de resultados, V) Conclusiones y futuros trabajos y VI) Publicación de resultados.

I) Revisión bibliográfica: búsqueda de información bibliotecaria o a través de medios digitales relacionada con el tema.

II) Modelo del sistema: utilizando un tipo de modelado de sistemas, detallar las características de los componentes y sus relaciones.

III) Simulación: aprender el uso de una herramienta de software que permita la simulación del sistema.

IV) Análisis de resultados: Uso herramientas estadísticas para mostrar los resultados de una manera comprensible.

V) Conclusiones y futuros trabajos: contrastar los objetivos deseados con los resultados obtenidos y proponer futuras líneas de trabajo.

VI) Publicación de resultados: Redactar un reporte técnico mostrando los resultados obtenidos.

## 1.6 Cronograma de actividades

	Mes1				Mes2				Mes3				Mes4			
	s1	s2	s3	s4												
Fase I) Revisión bibliográfica	x	x	x	x												
Fase II) Diseño del modelo del sistema				x	x	x	x									
Fase III) Simulación						x	x	x	x							
Fase IV) Análisis de Resultados									x	x	x	x				

Fase V) Conclusiones y futuros trabajos											x	x	x			
Fase VI) Publicación de resultados												x	x	x	x	x

## 2 Sistemas de control en red

Los Sistemas de Control en Red – NCS surgieron principalmente de la necesidad de la industria por contar con sistemas de cableado más simples y para tratar de reducir costos tanto a nivel de hardware como de mantenimiento.

Así por ejemplo, en 1983 la compañía alemana Robert Bosch desarrolló el protocolo CAN (*Controller Area Network*) para uso en sistemas de automoción, siendo ahora utilizada en muchas otras aplicaciones de control industrial. Otra red industrial es Profibus, desarrollada por algunas empresas e institutos alemanes en 1987. Profibus es un bus de campo que opera como un sistema maestro/esclavo. Otros protocolos para redes de control industrial tales como Foundation Fieldbus y DeviceNet se desarrollaron en similar periodo de tiempo. La mayoría de estos protocolos son confiables y robustos para el control en tiempo real.

Por otra parte, hoy en día las tecnologías de redes de datos como Ethernet, que han progresado enormemente en la disminución de sus precios, incremento en su velocidad de transmisión, amplios usos, gran oferta de software y aplicaciones así como una infraestructura bien establecida, han venido a ser los grandes competidores a las redes de control para aplicaciones industriales [18]. Además, la popularidad del internet ha permitido el uso de las redes entre varias organizaciones. Así, las aplicaciones de control pueden utilizar estas redes para conectarse al Internet para realizar un control remoto de larga distancia sin invertir en mucha infraestructura. A pesar de que las redes industriales han sido mejoradas para su conectividad con el Internet, los precios más bajos y los amplios usos de las redes generales siguen siendo atractivos para su uso en las aplicaciones de control.

A continuación se presentarán diferentes tipos de clasificación de los NCS, así como una descripción de sus componentes. Finalmente, se estudiará los factores que ocasionan los retardos de tiempo en un NCS.

### 2.1 Clasificación de los NCS

Dado el surgimiento de un sinnúmero de aplicaciones y tipos de sistemas de control en red, existe la necesidad de clasificarlos. Hay dos tipos de clasificación de especial interés para el presente proyecto: por el tipo de enfoque en el diseño y por el tipo de aplicaciones [1].

Por el **tipo de enfoque** para diseñar un NCS [19]:

- 1) Estructura jerárquica.- Como se muestra en la Figura 2.1, la estructura jerárquica está compuesta por varios subsistemas. Cada subsistema contiene uno o varios sensores, un actuador, y un controlador en sí mismo. Estos componentes del sistema están conectados a la misma planta. En este caso, el controlador del subsistema recibe una o varias señales de referencia (por ej.: de posición, velocidad y aceleración) del Controlador Principal (Cp). El subsistema entonces trata de satisfacer estas señales por sí mismo. La medición de los sensores, o unas señales de estado, son devueltas al Cp. En lo que resta del presente trabajo se hará referencia al subsistema (compuesto por el controlador, el o los sensor/es y actuador) como controlador remoto de movimiento o simplemente **controlador remoto**.

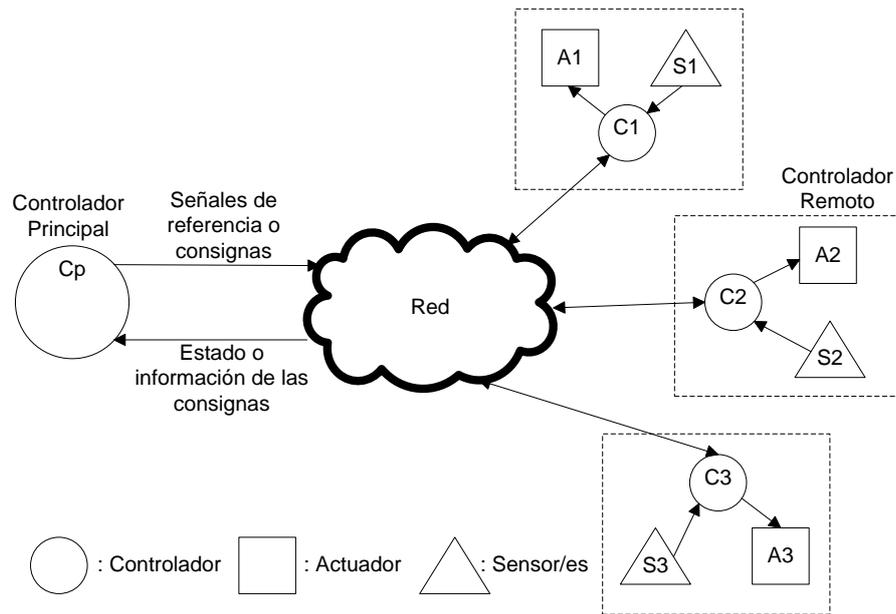


Figura 2.1 NCS en estructura jerárquica

- 2) Estructura directa.- Como se puede observar en la Figura 2.2, lo que caracteriza a este tipo de estructura es que el controlador realiza toda la tarea de control, donde este recibe la información de la medida del sensor, ejecuta la tarea de control, para luego enviar la señal de control hacia el actuador. Los inconvenientes con este tipo de estructuras es que al existir más actuadores y sensores en el sistema, se requiere de mayor capacidad de procesamiento del controlador para ejecutar las tareas de control, además de haber un mayor tráfico en la red.

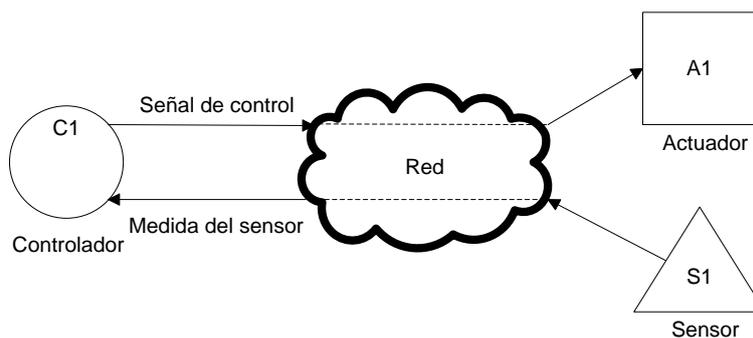


Figura 2.2 NCS en estructura directa

Por el **tipo de aplicaciones**, los NCS pueden estar divididos en dos categorías:

- 1) Aplicaciones sensibles a los tiempos de retardo.- Son aplicaciones críticas en el tiempo, es decir, si el tiempo de retardo excede un tiempo límite tolerable, puede deteriorarse el funcionamiento de la planta o disminuir su rendimiento. Ejemplos de aplicaciones sensibles al tiempo pueden ser la teleoperación para tareas de rescate, operaciones submarinas, conducción automática por carreteras, telecirugía, etc.
- 2) Aplicaciones insensibles a los tiempos de retardo.- En este tipo de aplicaciones las tareas se ejecutan en tiempo real, pero sus plazos no son críticos. Por ejemplo: el control de riego en una plantación, el control de la temperatura en un edificio, etc.

## 2.2 Componentes de un NCS

Todo sistema de control en red tiene una serie de elementos básicos primordiales, tales como: la red de control, los controladores, los sensores y los actuadores. A continuación se hará una descripción de los componentes.

### 2.2.1 Red de control

La red de comunicación es la columna vertebral de un NCS, ya que es el medio que permite la comunicación entre los diferentes dispositivos (controladores, sensores y actuadores). Existen varios factores que afectan la disponibilidad de la red: tasas de muestreo en las cuales los diferentes dispositivos envían información, el número de nodos que intervienen en la red, el método de sincronización entre los dispositivos, tamaño de los mensajes, factores físicos de la red y el tipo de Control de Acceso al Medio (MAC) que controla la transmisión de la información [20].

El protocolo de subcapa de Control de Acceso al Medio, que describe el protocolo para obtener el acceso a la red, es el responsable de satisfacer los requisitos de respuesta en tiempo-real o tiempo-crítico sobre la red, y además de la calidad y fiabilidad de la comunicación entre los nodos [21]. Existen tres tipos de MAC:

- De acceso aleatorio con retransmisión cuando hay colisiones (ej.: Ethernet),
- De acceso aleatorio con prioridad para evadir colisiones (ej.: CAN),
- De multiplexación por división de tiempo utilizando paso-de-testigo o maestro-esclavo.

Dentro de cada uno de estos tipos de acceso al medio existen un sinnúmero de protocolos que han sido definidos y utilizados en diversos tipos de aplicaciones. Para cada tipo de protocolo, [22] estudia los principales parámetros de la red cuando se utilizan en un sistema de control, incluyendo la utilización de la red, la magnitud del tiempo de retardo esperado y las características de los retardos.

#### Tipos de conexión de mensajes

Para la operación de una red de control, el tipo de conexión del mensaje, que es la manera como se comunican los dispositivos, debe ser especificado [22]. En los NCS, hay tres tipos de conexión de mensajes: *strobe*, *poll* y cambio de estado (COS)/cíclica.

En una conexión tipo ***strobe***, ver Figura 2.3, el dispositivo maestro emite un mensaje tipo *strobe* a todos o a un grupo de dispositivos y estos responden con su actual condición (ya sea con un valor de error, de alguna lectura o de su estado). En este caso, se considera que todos los dispositivos hacen el muestreo al mismo tiempo.

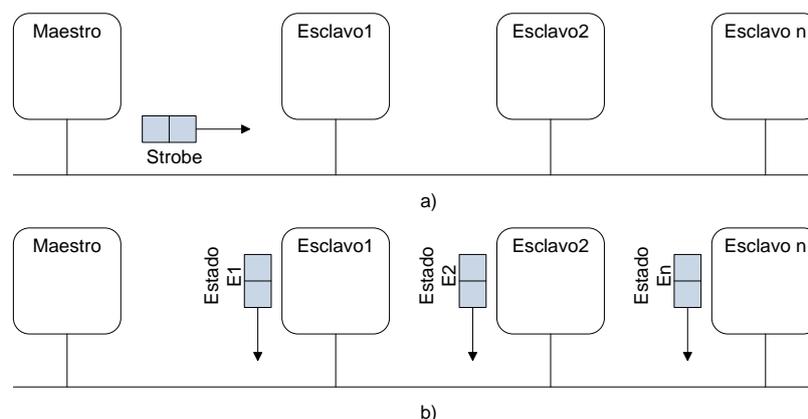
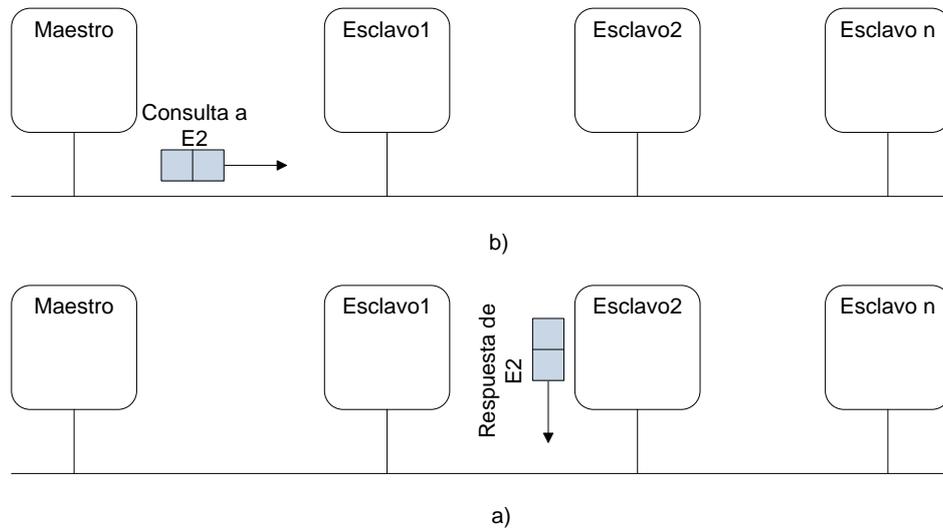


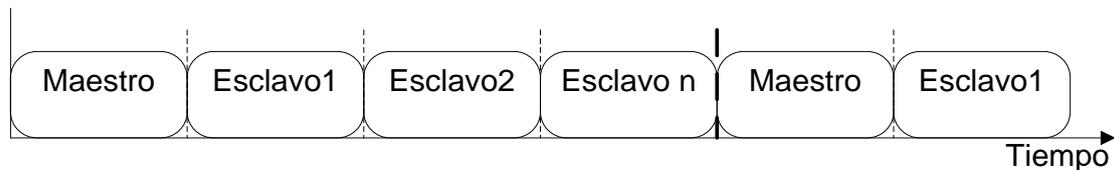
Figura 2.3 Conexión de mensajes tipo *strobe*. a) el nodo maestro envía la señal de *strobe* a los nodos esclavos. b) los nodos esclavos responden con una señal de estado al nodo maestro.

En una conexión tipo **poll** o encuesta, como lo muestra la Figura 2.4, el dispositivo maestro envía mensajes personalizados a los dispositivos encuestados y solicita información actualizada de ellos. Los dispositivos solo responden a nuevas señales luego de que hayan recibido un mensaje de encuesta.



**Figura 2.4** Conexión de mensajes tipo **poll** o encuesta. a) el nodo maestro realiza una consulta al nodo esclavo 2  
b) el nodo esclavo 2 responde al nodo maestro

En una conexión tipo **COS/cíclico**, ver Figura 2.5, los dispositivos envían mensajes bien sea cuando estos hayan cambiado su estado (COS) o de manera periódica (cíclica). A pesar de que la conexión COS/cíclica parezca ser la más apropiada desde el punto de vista tradicional de los sistemas de control, las conexiones *strobe* y *poll* son comúnmente utilizadas en redes de control industrial [22].



**Figura 2.5** Conexión de mensajes tipo **COS/cíclico**.

### Redes Ethernet (CSMA)

Las redes Ethernet generalmente utilizan el método de acceso al medio CSMA/CD (acceso múltiple por detección de portadora con detección de colisiones) como lo especifica el estándar IEEE 802.3 [23] [24] [25], con la finalidad de resolver la contención en el medio de comunicación. Hay dos tipos de Ethernet con cable: 1) Ethernet basado en concentrador (*hub-based*), ampliamente utilizado en ambientes de oficina, y 2) Ethernet conmutado (*switched*), utilizado con frecuencia en la manufactura y control de procesos.

#### Ethernet basado en concentrador (CSMA/CD)

Este tipo de Ethernet utiliza un concentrador para interconectar los dispositivos en una red. Cuando un mensaje llega al concentrador, este simplemente hace un *broadcast* de dicho mensaje al resto de dispositivos conectados al concentrador. Por lo tanto, todos los dispositivos reciben el mismo paquete simultáneamente, y es posible que existan colisiones de mensajes si varios dispositivos intentan responder al mismo tiempo.

### Formato de la trama

Como lo muestra la Figura 2.6, una trama o *frame* Ethernet está compuesta por **26 bytes de overhead** (correspondientes a 7 Bytes de de preámbulo, 1 Byte de inicio de delimitador, 6 Bytes de dirección destino, 6 Bytes de dirección origen, 2 Bytes de longitud y 4 Bytes de checksum) y de 46 a 1500 bytes de datos.

Bytes	7	1	6	6	2	46-1500	4
	Preámbulo	Inicio del delimitador	Dirección Destino	Dirección Origen	Longitud	Datos	Checksum

Figura 2.6 Formato de la trama Ethernet

Un requisito del estándar es que el tamaño de la trama, contando desde la Dirección Destino hasta el Checksum, no puede ser menor de 64 Bytes; por lo tanto, el campo Datos no puede ser nunca menor a 46 Bytes. Si el tamaño del dato fuera menor a 46 Bytes, entonces se utilizan unos Bytes de relleno (*padding*) para cumplir con los mínimos requisitos.

Para redes de control, el protocolo Ethernet comúnmente utiliza una tasa de datos estándar de **10 Mbps** (ej. Modbus/TCP). El Ethernet de alta velocidad (100Mbps o 1Gbps) es utilizado principalmente en las redes de datos [25].

### Control de acceso al medio (MAC)

El MAC en una red Ethernet opera de la siguiente manera: cuando un nodo quiere transmitir, este escucha la red. Si la red está ocupada, el nodo espera hasta que la red esté libre; caso contrario, este transmite inmediatamente. Si dos o más nodos detectan que la red está libre y deciden transmitir simultáneamente, los mensajes de estos nodos colisionarán y se corromperán. Mientras está transmitiendo, un nodo debe también “escuchar” la red para que pueda detectar una colisión. Cuando se detecta una colisión entre dos o más mensajes, los nodos que están transmitiendo detienen su transmisión y esperan un tiempo aleatorio para intentar su retransmisión. Este tiempo aleatorio es determinado por el estándar *Binary Exponential Backoff* (BEB).

El algoritmo BEB es el siguiente: el tiempo de retransmisión es seleccionado aleatoriamente entre 0 y  $(2^i - 1)$  ranuras de tiempo, donde  $i$  corresponde al  $i$ -ésimo evento de colisión detectado por el nodo, y una ranura de tiempo es el mínimo tiempo necesario para una transmisión de ida y vuelta (*round-trip*). Sin embargo, luego de que se hayan alcanzado 10 intentos de transmisión, el intervalo se fija a un máximo de 1023 ranuras. Luego de 16 colisiones, el nodo detiene su intento de transmitir y notifica un error al procesador. Es posible que se implementen algoritmos de recuperación de mensajes en las capas superiores del modelo OSI [25].

### Tiempo de bloqueo

El tiempo de bloqueo en Ethernet incluye el tiempo de las colisiones con otros mensajes y su posterior tiempo de espera a ser retransmitido. El algoritmo BEB explicado anteriormente muestra un tiempo de espera probabilístico. Un análisis exacto del tiempo de bloqueo esperado para Ethernet es muy difícil. A un alto nivel, el tiempo de bloqueo esperado puede ser descrito por la siguiente ecuación:

$$E\{T_{\text{bloqueo}}\} = T_{\text{resid}} + \sum_{k=1}^{16} E\{T_k\} \quad (2.1)$$

Donde  $T_{\text{resid}}$  denota el tiempo residual hasta que la red esté libre, y  $E\{T_k\}$  es el tiempo esperado para la  $k$ -ésima colisión. Para la colisión número 16, el nodo descarta el mensaje y reporta un mensaje de error a las unidades de procesamiento de mayor nivel [25]. Puede verse que el tiempo de bloqueo no es determinístico y hasta puede ser ilimitado debido a los descartes de los mensajes.

#### Ethernet conmutado o *switched ethernet* (CSMA/CA)

Este utiliza conmutadores para subdividir la arquitectura de la red, de esta manera evita colisiones, incrementa la eficiencia de la red, y mejora su determinismo. Esta red es ampliamente utilizada en aplicaciones de manufactura. La principal diferencia entre la red basada en un concentrador con la basada en conmutador es la “inteligencia” en la transmisión de los paquetes. Los concentradores simplemente pasan el tráfico entrante de cualquier puerto a todos los demás puertos, mientras que los conmutadores aprenden la topología de la red y envían los paquetes solo a los nodos destinatarios. En un esquema de red de tipo estrella, cada nodo está conectado con un solo cable al conmutador, y por tanto no existirán colisiones en la red.

Los conmutadores emplean las técnicas de “cortar-seguir” *cut-through* o “guardar y avanzar” para enviar paquetes de un puerto a otro, usando varios buffers por cada puerto para que los paquetes esperen a ser enviados a dicho puerto. Los conmutadores con la técnica de “cortar-seguir” primero leen la dirección MAC y envían el paquete dependiendo de la dirección destino y de una tabla de envío en el conmutador. Con la técnica de “guardar y avanzar” se examina todo el paquete primero. Usando el código de chequeo de redundancia cíclica (CRC), el conmutador verifica primero que la trama ha sido correctamente transmitida antes de continuar con el envío hacia el nodo destino. Si existe un error, el paquete es descartado. Los conmutadores “guardar y avanzar” son más lentos, pero evitan el envío de algún paquete dañado.

A pesar de que no existan colisiones en una red Ethernet conmutada, puede existir congestionamiento dentro del conmutador cuando un puerto repentinamente recibe un gran número de paquetes de los otros puertos, en cuyo caso se aplica un principio de colas dentro del conmutador [22].

#### Redes tipo CAN: *DeviceNet*

CAN (Controller Area Network) es un protocolo de comunicación serial, que nació como respuesta a la necesidad que tenía la industria de la automoción de disponer de una comunicación fiable, eficaz y rápida para los vehículos producidos [26]. CAN también es capaz de ofrecer un buen rendimiento en otras aplicaciones de tiempo-crítico.

Debido a sus grandes prestaciones, la ISO (*International Standard Organization*) adoptó dos estándares del CAN:

- ISO 11519-2: Estándar para aplicaciones de baja velocidad con una tasa de transmisión de 125Kbit/s (LS\_CAN).
- ISO 11898: Estándar para aplicaciones de alta velocidad con una tasa de transmisión de 1 Mbit/s (HS\_CAN) [27].

#### Formato de la trama

CAN tiene dos formatos de trama:

- El formato estándar con un identificador de mensaje (ID) de 11 bits; y,
- El formato extendido con ID de 29 bits [28].

La especificación del bus CAN está definida solamente para la capa de enlace de datos ('*link-data*') del modelo OSI, indicando solamente las características de la trama del mensaje, el arbitraje y la verificación de errores.

DeviceNet es un protocolo con una tecnología basada en la especificación CAN y ha recibido una considerable aceptación en aplicaciones de manufactura a nivel de dispositivos. Este protocolo está basado en el formato estándar (Id de 11 bits) de CAN con especificaciones para las capas física y de aplicación [29] [30].

En la Figura 2.7 se muestra el formato de la trama del DeviceNet [29]. El overhead total es de 47 bits, que incluye el inicio de trama (SOF), ID, control, CRC, reconocimiento (ACK), fin de trama (EOF), y campos de intermisión (INT). El tamaño de los datos está entre 0 y 8 bytes (ó 64 bits). El protocolo usa el campo ID para proveer direccionamiento fuente y destino así como la priorización de los mensajes.



**Figura 2.7 Formato de trama para DeviceNet (CAN)**

### Control de acceso al medio (MAC)

El protocolo CAN está optimizado para mensajes cortos y utiliza el método de acceso al medio CSMA/AMP (acceso múltiple por detección de portadora con arbitraje en la prioridad del mensaje). Cada mensaje tiene una prioridad específica que es utilizada para el arbitraje del acceso al bus en caso de transmisiones simultáneas. El flujo de bits de una transmisión está sincronizado por el bit de inicio, y el arbitraje es realizado en el identificador del mensaje (ID), en el cual un 0 lógico es dominante sobre un 1 lógico. Un nodo que quiera transmitir un mensaje espera que el bus esté libre y luego inicia enviando el identificador (ID) del mensaje bit a bit. Los conflictos por el acceso al bus son resueltos durante la transmisión por un proceso de arbitraje a nivel de bits del campo arbitraje, que es la parte inicial de la trama. Aquí, si dos dispositivos quieren emitir un mensaje al mismo tiempo, ellos primero inician enviando una trama para luego escuchar el bus. Si uno de ellos recibe un bit diferente al enviado, este pierde el derecho a continuar enviando su mensaje, y el otro gana el arbitraje. Con este método, una transmisión en curso no es afectada.

En una red tipo CAN, los datos son transmitidos y recibidos usando tramas de mensajes que llevan datos desde un nodo transmisor a uno o varios nodos receptores. Los datos transmitidos no necesariamente llevan las direcciones del nodo fuente y destino. En vez de eso, cada mensaje está etiquetado por un identificador que es único en toda la red. Todos los otros nodos en la red reciben el mensaje y aceptan o rechazan este, dependiendo de la configuración de los filtros de máscara para el identificador. Este modo de operación es conocido como multicast.

### Tiempo de bloqueo

El tiempo de bloqueo,  $T_{bloqueo}$ , en DeviceNet puede ser descrito por la siguiente ecuación [31]:

$$T_{bloqueo}^{(k)} = T_{resid} + \sum_{\forall j \in N_{hp}} \left\lceil \frac{T_{bloqueo}^{(k-1)} + T_{bit}}{T_{peri}^{(j)}} \right\rceil T_{tx}^{(j)} \quad (2.2)$$

Donde  $k$  es el índice de iteración para obtener el estado estable de  $T_{bloqueo}$ ,  $T_{resid}$  es el tiempo residual que requiere el nodo actual para finalizar la transmisión,  $N_{hp}$  es el conjunto de nodos con mayor prioridad que el nodo de espera,  $T_{peri}^{(j)}$  es el periodo del  $j$ -ésimo nodo, y  $\lceil x \rceil$  denota el entero menor que es mayor o igual a  $x$ . La sumatoria indica el tiempo necesario para enviar todos los mensajes con mayor prioridad que el nodo actual. Mientras un nodo de menor prioridad espera la disponibilidad del canal, es posible que los nodos con mayor prioridad entren a la cola de mensajes, en cuyo caso los nodos de menor prioridad pierden el arbitraje nuevamente. Esta situación acumula el tiempo de bloqueo. El  $T_{resid}$  en el peor de los casos, sobre una red con baja carga de tráfico, es igual a:

$$T_{resid} = \max_{\forall j \in N_{nodo}} T_{tx}^{(j)} \quad (2.3)$$

Donde  $N_{nodo}$  es el conjunto de nodos en la red. Sin embargo, dado el mecanismo de arbitraje de prioridad en el mensaje, la transmisión de un mensaje de baja prioridad puede no ser determinista o delimitada si la carga en la red es alta [22].

### Redes con multiplexación por división de tiempo (TDM)

Existen dos tipos de multiplexación por división de tiempo:

- Tipo maestro-esclavo.- En este un nodo maestro consulta a varios esclavos. Los esclavos solo pueden enviar datos sobre la red cuando son consultados por el maestro. De esta manera no hay colisiones, mientras que las transmisiones son cuidadosamente planificadas por el nodo maestro.
- Tipo token-passing.- Esta red tiene múltiples maestros, o “compañeros”, y se comunican entre sí mediante el paso de un testigo o token. Al nodo que actualmente tiene el token se le permite enviar datos. Cuando el nodo ha terminado de enviar los datos, o en su defecto se haya alcanzado su máximo tiempo de posesión del token, este lo cede al siguiente nodo lógico de la red. Si un nodo no tiene ningún mensaje que enviar, este deja pasar el token al nodo sucesor. La ubicación física del nodo no importa debido a que el token es enviado a su vecino lógico. Aquí no existen colisiones de mensajes, porque solo un nodo transmite en cada instante de tiempo.

Ejemplos de redes de control de tipo maestro-esclavo son: ASI, Bitbus e Interbus-S; mientras que Profibus y ControlNet son redes de tipo token-passing. Cada nodo “compañero” en una red Profibus puede comportarse como un maestro y comunicarse con un conjunto de esclavos durante el tiempo que este conserve el token o testigo. Estas son redes determinísticas porque el máximo tiempo de espera antes de enviar una trama puede ser caracterizado por el tiempo de rotación del testigo. Los nodos en una red tipo testigo están organizados lógicamente en un anillo, y, en el caso por ejemplo de ControlNet, cada nodo conoce la dirección de su predecesor y sucesor. Durante la operación de la red, el nodo con el testigo hace uso de la red transmitiendo información hasta que se acabe la trama de datos o hasta cuando se agote el tiempo de mantenimiento del testigo. El nodo entonces regenera el testigo y lo transmite a su sucesor lógico de la red.

### 2.2.2 Controladores

Una de las mayores ventajas de un sistema controlado en una red es su escalabilidad. Así como se habla de agregar varios sensores en ubicaciones distintas en una red, también es posible tener uno o más actuadores conectados a uno o más controladores a través de la red. Desde hace algunos años acá, los investigadores han dado estrategias de control precisas y óptimas que provienen de la teoría de control clásica, desde el control PID, control óptimo, control adaptivo, control robusto, control inteligente y algunas otras formas avanzadas de estos algoritmos de control. Sin embargo el aplicar estas estrategias de control sobre una red no es una tarea fácil.

En los controladores también es importante considerar la planificación de los recursos del CPU, principalmente para aplicaciones de tiempo-crítico, debido a las restricciones de tiempo de las tareas a llevar a cabo.

#### Planificación del CPU

Un sistema en tiempo real debe estar en la capacidad de asignar los recursos de procesamiento a aquellas tareas que lo necesiten. Para ello se requiere un algoritmo de planificación de tareas del procesador (o CPU).

Existen varias técnicas de planificación del CPU, que se pueden agrupar en técnicas clásicas de lazo abierto y técnicas de planificación de lazo cerrado [32] [33]. Algunas de estas técnicas son:

Rate Monotonic (RM).- es un algoritmo de planificación estático en lazo abierto, que asigna las prioridades a las tareas (periódicas) de acuerdo con las tasas solicitadas.

Earliest Deadline First (EDF).- es un algoritmo dinámico en lazo abierto, que asigna prioridades a las tareas en relación de sus plazos absolutos.

LEF (Large Error First).- es un algoritmo de planificación dinámico en lazo cerrado, que ajusta el planificador basado en la retroalimentación continua de cada lazo de control.

### 2.2.3 Sensores y actuadores

Los sensores y actuadores permiten la interacción del robot desde y hacia el mundo exterior, respectivamente. Los avances en cuanto a eficiencia, miniaturización y capacidad de cómputo han permitido el desarrollo de sensores y actuadores con mejores prestaciones en el mercado.

## 2.3 Tiempos de retardo en un NCS

Es muy conocido que los tiempos de retardo en un NCS, ya sean ocasionados por los controladores, sensores, actuadores o por la red de comunicación, pueden afectar seriamente el rendimiento y la estabilidad del sistema, por lo que es importante estudiar su naturaleza.

Los tiempos de retardo que se toman en cuenta para el análisis de un NCS son los tiempos de retardo de extremo a extremo controlador-actuador y sensor-controlador [22] [19]. El tiempo de retardo de transmisión de un mensaje ( $T_{retardo}$ ) puede ser dividido en dos partes: el tiempo del dispositivo ( $T_{disp}$ ) y el tiempo de la red ( $T_{red}$ ).

$$T_{retardo} = T_{disp} + T_{red} \quad (2.4)$$

En el tiempo del dispositivo se incluyen los retardos en los nodos fuente y destino. El tiempo de retardo en el nodo fuente incluye un tiempo de preprocesamiento ( $T_{pre}$ ), y el tiempo de espera ( $T_{espera}$ ). El retardo en el nodo destino incluye solamente el tiempo de posprocesamiento ( $T_{pos}$ ), así:

$$T_{disp} = T_{pre} + T_{espera} + T_{pos} \quad (2.5)$$

Entonces, como lo muestra la Figura 2.8, el tiempo total de retardo se puede expresar de la siguiente manera:

$$T_{retardo} = T_{pre} + T_{espera} + T_{red} + T_{pos} \quad (2.6)$$

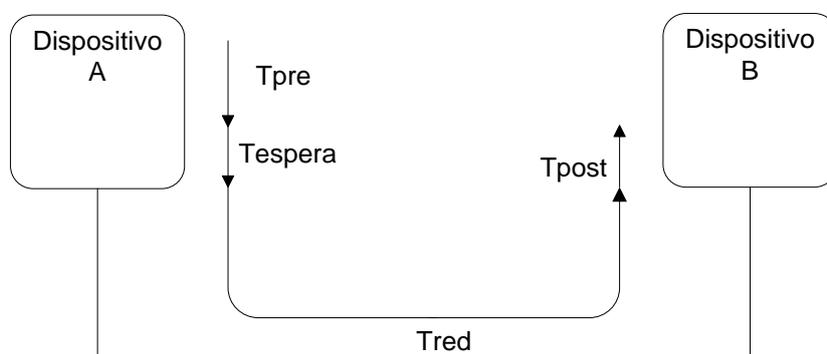


Figura 2.8 Tiempo de retardo en un sistema de control en red

### Tiempo de preprocesamiento ( $T_{pre}$ ) y posprocesamiento ( $T_{pos}$ )

El tiempo de preprocesamiento en el nodo fuente es el tiempo necesario para adquirir los datos del exterior y codificarlos en un formato de datos de red apropiados. Puede haber un solo procesador cumpliendo ambas funciones, o múltiples procesadores. Se define el tiempo total transcurrido requerido como tiempo de pre- o posprocesamiento. Este tiempo depende de las características del hardware y software del controlador. En algunos casos, se puede asumir que el tiempo de preprocesamiento es constante o despreciable.

El tiempo de posprocesamiento en el nodo destino es el tiempo requerido para decodificar los datos de la red en un formato específico para su posterior análisis.

### Tiempo de transmisión en la red ( $T_{red}$ )

El tiempo de transmisión en la red es el parámetro más determinístico en una red debido a que este depende únicamente de la tasa de datos, el tamaño del mensaje y la distancia entre dos nodos. La fórmula para el tiempo de transmisión puede describirse como:

$$T_{red} = T_{trama} + T_{prop} \quad (2.7)$$

Donde  $T_{trama}$  es el tiempo requerido para enviar un paquete de datos a través de la red, y  $T_{prop}$  es el tiempo de propagación entre dos dispositivos.

Dado que la velocidad típica de transmisión en un medio de comunicación es de  $2 \times 10^8$  m/s, el tiempo de propagación  $T_{prop}$  se lo considera insignificante a pequeña escala. Además, el retardo de propagación no puede ser caracterizado fácilmente debido a que las distancias entre la fuente y los nodos de destino por lo general no siempre son constantes para diferentes transmisiones.

El tiempo de la trama,  $T_{trama}$ , depende del tamaño del dato, el overhead, algún bit de relleno y la tasa de datos o el tiempo de bit ( $T_{bit}$ ). Si  $N_{data}$  es el tamaño de datos en términos de bytes,  $N_{ovhd}$  el número de bytes usados como overhead,  $N_{pad}$  es el número de bytes utilizados para rellenar una parte restante de la trama con el fin de satisfacer los mínimos requerimientos de tamaño de

trama, y  $T_{stuff}$  es el número de bytes usados en un mecanismo de relleno (en ciertos protocolos).  $T_{trama}$  se puede expresar de la siguiente forma:

$$T_{trama} = (N_{data} + N_{ovhd} + N_{pad} + T_{stuff}) \cdot 8 \cdot T_{bit} \quad (2.8)$$

Los valores de  $N_{data}$ ,  $N_{ovhd}$ ,  $N_{pad}$  y  $T_{stuff}$  son descritos en detalle para los protocolos Ethernet, ControlNet y DeviceNet en [11].

### Tiempo de bloqueo

Un mensaje podría estar en un estado de espera en una cola de un buffer del nodo remitente y puede ser bloqueado para su transmisión por otros mensajes en la red. Dependiendo de la cantidad de datos que el nodo fuente deba enviar y del tráfico en la red, el tiempo de espera puede ser significativo. Los principales factores que afectan el tiempo de espera son el protocolo de red, el tipo de conexión del mensaje y la carga de tráfico de la red.

El tiempo de bloqueo, que es el tiempo que un mensaje debe esperar toda vez que el nodo está listo para su envío, depende del protocolo de red y es el principal factor en el determinismo y en el rendimiento de una red de control. Este tiempo incluye el tiempo de espera mientras los demás nodos están enviando mensajes y el tiempo necesario para retransmitir el mensaje en caso de colisión. Los protocolos de red de capas superiores como TCP pueden requerir una retransmisión si ha ocurrido algún error en el paquete, o si un switch o router descartó algún paquete. Este incidente puede ser compensado en un NCS. A pesar de que algún tipo de control o de señales sensoriales pueden perderse por las transmisiones en la red, algunos NCS pueden funcionar aceptablemente. En este caso, una retransmisión puede afectar gravemente el funcionamiento de un NCS debido a los mayores retrasos producto de una retransmisión.

## 2.4 Control de movimiento

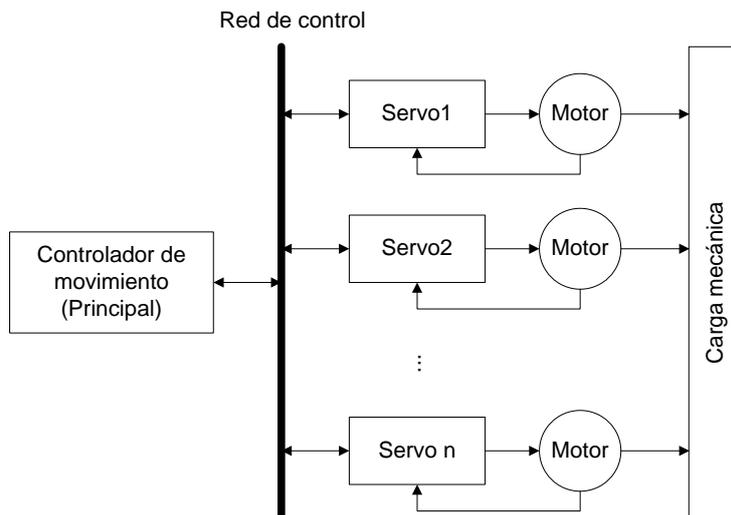
El control de movimiento tiene por objetivo poder realizar un seguimiento de las trayectorias deseadas así como el de mantener el equilibrio ante algún tipo de perturbación, sea esta de tipo interna o externa.

En el inicio se explicará el control de movimiento en red, así como sus configuraciones existentes. Luego se hará un enfoque del problema del control de movimiento aplicado a un robot manipulador. Finalmente se expondrán varios tipos de control de movimiento desde el punto de vista del control de las articulaciones independiente.

### 2.4.1 Sistema de control de movimiento en red (motion NCS)

Los sistemas de control de movimiento sincronizado de múltiples ejes pueden encontrarse en varios procesos de manufactura, por ej., máquinas de impresión, robótica, máquinas de montaje superficial, líneas de ensamblaje, máquinas CNC, etc. Dada la alta demanda de sistemas de automatización más flexibles, los sistemas de control de movimiento en red tienden a reemplazar las soluciones tradicionales. Como se mencionó anteriormente, existen varias ventajas de los sistemas de control en red sobre los convencionales, como la reducción en el cableado, facilidad de diagnóstico y mantenimiento, agilidad en el sistema, etc.

Un sistema de control de movimiento en red, ver Figura 2.9, está compuesto de servocontroladores con interfaz de red, motores, un controlador de movimiento y una red de control. El movimiento en red requiere de una red de control que soporte altas velocidades de transmisión de comandos de control e información de retroalimentación [34].



**Figura 2.9 Esquema de un Sistema de control de movimiento en red (motion NCS).**

Para construir un sistema de control de movimiento en red, se tiene que realizar una buena configuración del sistema. De acuerdo con el método de control empleado para cada eje, existen tres tipos de configuraciones generales:

1) La configuración independiente: en este tipo de configuración todos los lazos de las consignas (de posición, velocidad y aceleración) de cada eje se encuentran dentro del controlador servo. Los lazos de control de cada eje son independientes uno del otro. En cada ciclo de comunicación el controlador de movimiento envía las consignas deseadas a todos los servocontroladores por la red de control. Esta configuración provee una buena modularidad y reduce la carga de la red. Sin embargo, una pobre precisión de sincronización es la mayor desventaja de esta configuración. Mientras que los lazos de control y perturbación de carga son diferentes para cada eje, existe un error de sincronización considerable.

2) Configuración maestro-esclavo: en esta configuración los lazos de control de cada eje también residen dentro de cada servo controlador. En cada ciclo de comunicación, el eje maestro recibe una consigna desde el controlador de movimiento a través de la red de control y luego utiliza esta información para realizar el lazo de control localmente y envía un mensaje de la medida de posición del eje maestro a todos los ejes esclavos mediante la red. Los servocontroladores de cada eje esclavo reciben tanto la medida de posición del maestro como su consigna para el lazo de control. Esta configuración puede reducir el error de sincronización sin una alta carga en la red, pero se requiere tener una comunicación esclavo-esclavo.

3) Configuración centralizada: en esta configuración los lazos de control están contenidos dentro del controlador principal. En cada ciclo de comunicación, el controlador principal recibe los mensajes de las mediciones de cada eje vía red de control, realiza el cálculo del control y envía los comandos a todos los controladores por la red. Usando el método de acoplamiento cruzado, el controlador de movimiento puede compensar la diferencia de la dinámica y la perturbación de la carga de cada eje y reducir significativamente el error de sincronización. Sin embargo, la mayor desventaja en esta configuración es la necesidad de un gran ancho de banda.

### 2.4.2 Control de movimiento de un robot manipulador

El problema de controlar un robot manipulador puede ser formulado como el determinar la evolución temporal de las fuerzas generalizadas de los actuadores de las articulaciones de manera tal que garanticen la ejecución de los comandos deseados mientras satisfacen un transiente dado y requerimientos de estado estable. La tarea puede considerarse o bien la ejecución de movimientos específicos para un manipulador operando en el espacio libre o bien la ejecución de movimientos especificados y fuerzas de contacto para un manipulador cuyo efector final es limitado por su entorno. Dada la complejidad del problema, se suele tratar estos aspectos de manera separada; así, el control de movimiento en el espacio libre, y luego el control de interacción en espacio restringido.

Las técnicas de control en los manipuladores se pueden clasificar en esquemas de *control descentralizados*, donde una articulación del manipulador es controlada independientemente del resto (control de articulaciones independiente), y esquemas de control centralizado, donde los efectos de interacción dinámica entre las articulaciones son tomados en cuenta.

El presente proyecto considera el control de movimiento en el espacio libre con un esquema de control descentralizado.

### 2.4.3 Control de articulaciones independiente

La más simple estrategia de control que se pueda pensar es el de considerar al manipulador formado por  $n$  sistemas independientes ( $n$  articulaciones) y controlar cada eje de la articulación como un *sistema de una entrada/una salida*. Los efectos de acoplamiento entre las articulaciones debido a la variación de configuraciones durante el movimiento son tratados como perturbaciones de entrada. Asumiendo que el actuador es un motor DC rotacional, el esquema de bloques de la articulación  $i$  puede ser representada en el dominio de la variable compleja  $s$  como en la Figura 2.10. En este esquema  $\theta$  es la variable angular del motor,  $I$  es el promedio de la inercia reportada en el eje del motor,  $R_a$  es la Resistencia de la armadura, y  $K_t$  y  $K_v$  son las constantes de torque y del motor respectivamente. Además,  $G_v$  denota la ganancia del voltaje del amplificador de potencia, por lo que la entrada de referencia no es el voltaje de la armadura  $V_a$  sino el voltaje de entrada  $V_c$  del amplificador. En el esquema se asume que el coeficiente de fricción mecánica es mucho menor que el coeficiente de fricción eléctrica, por lo que no ha sido tomado en cuenta [35].

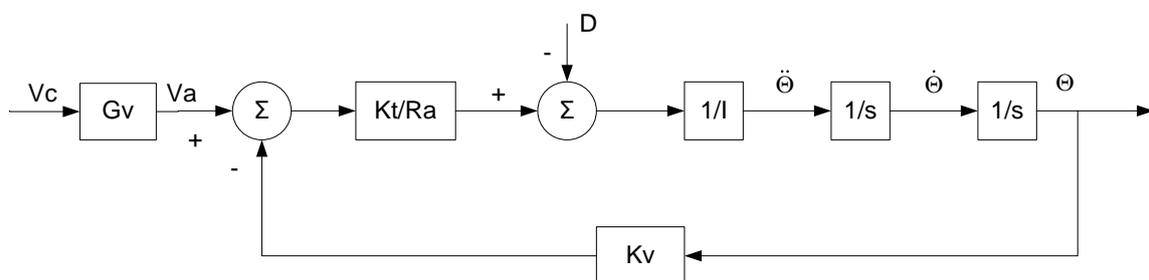


Figura 2.10 Esquema de un sistema de transmisión de una articulación

El principal objetivo de un sistema de control de movimiento en un robot es el de realizar las tareas de movimiento de sus articulaciones. El movimiento es realizado por un actuador (normalmente un motor eléctrico). Un actuador es un dispositivo que utiliza la energía eléctrica para producir energía mecánica. Esta energía produce un movimiento mecánico que consiste en el movimiento de una articulación del robot. Para el movimiento estable de un robot, el movimiento

de la articulación debe ser preciso y seguro. Por tanto, todo sistema de control de movimiento en su nivel más bajo es provisto por el servo control. La necesidad de mejorar los tiempos de respuesta transitoria, reducir los errores en estado estacionario y reducir la sensibilidad de una articulación a parámetros de carga es la razón básica para usar servo sistemas en vez de sistemas de lazo abierto.

Mejorar el tiempo de respuesta transitoria generalmente significa incrementar el ancho de banda del sistema. Tiempos de respuesta más rápidos significa un establecimiento más rápido permitiendo una mayor productividad del sistema. Reducir los errores en estado estacionario se relaciona con la precisión del servo sistema. Finalmente, reducir la sensibilidad a los parámetros de carga significa que el servo sistema puede tolerar fluctuaciones tanto en sus parámetros de entrada como de salida.

El servo control en general se puede dividir en dos clases de problemas. El primero trata con el seguimiento de consignas; abordando la pregunta de qué tan bien el movimiento actual sigue lo que se esté mandando. Los típicos comandos en el control de movimiento de rotación son posición, velocidad, aceleración y par. Para el movimiento lineal, la fuerza es utilizada en vez del par. La parte del servo control que directamente trata con este a menudo es referido como control "Feedforward". Este puede ser pensado en términos de que comandos internos son necesarios para que los comandos de movimiento sean seguidos sin errores. El control Feedforward asume un buen conocimiento del modelo del motor y de la carga.

La segunda clase general del servo control se refiere a las características de rechazo de perturbaciones del sistema. Las perturbaciones en los sistemas de control de movimiento pueden ser de variados tipos. Pueden ser perturbaciones de par en el eje del motor o estimaciones incorrectas en los parámetros del motor usados en el control feedforward. El controlador PID (Proporcional, Integral y Derivativo) es utilizado muy a menudo para combatir este tipo de problemas. En contraste con el control feedforward, que predice los comandos internos necesarios para un error de seguimiento nulo, el control de rechazo de perturbaciones reacciona a perturbaciones desconocidas y modelado de errores. Un sistema de servo control completo combina estos dos tipos de servo control para proveer el mejor rendimiento general [36].

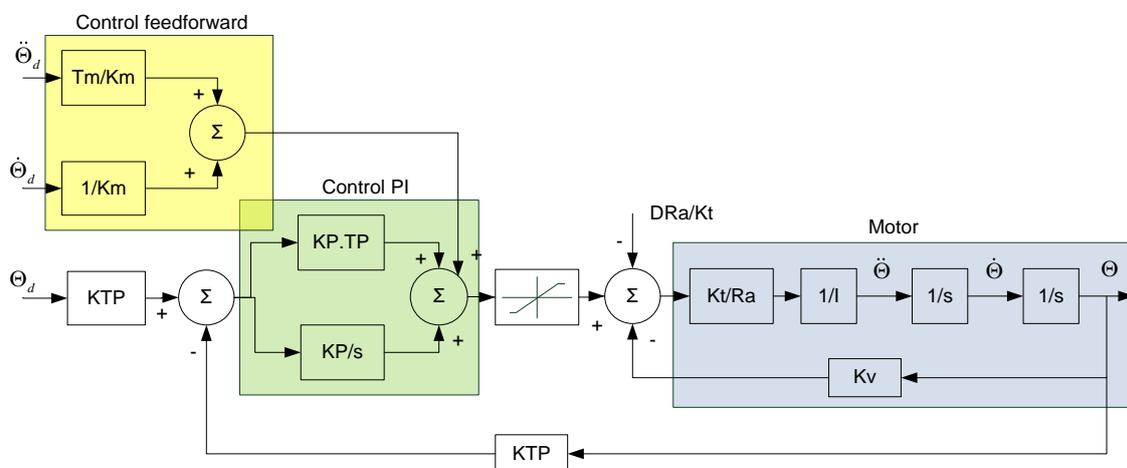


Figura 2.11 Esquema de control tipo PI con retroalimentación de posición y compensación descentralizada feedforward.

Cuando los servo controladores de las articulaciones son requeridos para seguir trayectorias de de referencia con elevados valores de velocidad y aceleración, la adopción de una *compensación descentralizada mediante feedforward* reduce el error de seguimiento. Un requisito básico para el

control feedforward es la disponibilidad de los comandos de velocidad y aceleración sincronizados con los comandos de posición. Un ejemplo de cómo el control feedforward es utilizado junto con un control de rechazo de perturbaciones es mostrado en la Figura 2.11

Donde

$$k_m = \frac{1}{k_v} \quad T_m = \frac{R_a I}{k_v k_t}$$

Son la ganancia velocidad-voltaje y la constante de tiempo del motor. Además  $KTP$  es la constante del transductor de posición,  $D$  es el torque de perturbación y  $KP$  y  $TP$  son los parámetros del controlador PI.

## 2.5 Evaluación del rendimiento

En un NCS, las redes son utilizadas para transmitir datos entre los dispositivos del sistema de control. Por lo tanto, la calidad de servicio de la red (QoS) debe ser analizada antes de implementar un sistema de control con una arquitectura de red. Por otra parte, la calidad de rendimiento del control (QoP) debe ser especificado para ayudar a evaluar el rendimiento del sistema de control. En esta sección, se expondrán algunas medidas utilizadas para determinar la calidad de servicio de la red y la calidad de rendimiento del control [4].

### 2.5.1 Calidad de servicio de la red

Existen algunas medidas de la QoS de la red como: estadísticas del retardo, eficiencia de la red, utilización de la red, y el número de mensajes perdidos o no enviados. Estas medidas son utilizadas para determinar la capacidad de la red, así como para proveer información que ayude a especificar parámetros de control como el período de muestreo.

#### Tiempo de retardo total y promedio

Para un tiempo de simulación,  $T_{max}$ , se puede calcular la suma y el promedio de los tiempos de retardo desde cada nodo.

$$T_{delay}^{sum} = \sum_{i \in N_{node}} \sum_{j=1}^{M^{(i)}} T_{delay}^{(i,j)} \quad (2.9)$$

$$T_{delay}^{avg} = \frac{1}{N} \sum_{i \in N_{node}} \left[ \frac{\sum_{j=1}^{M^{(i)}} T_{delay}^{(i,j)}}{M^{(i)}} \right] \quad (2.10)$$

Donde  $N$  es el número total de nodos,  $N_{node}$  es el conjunto de nodos, y  $M^{(i)}$  el número de mensajes solicitados al nodo  $i$ . Si se asume que todos los mensajes son periódicos, entonces el número total de mensajes es igual al tiempo total de ejecución dividido por el período de los mensajes. El retardo promedio puede ser calculado para toda la red, como lo muestra la ecuación (2.10), o para el  $i$ -ésimo nodo.

#### Eficiencia de la red

Se define a la eficiencia de la red,  $P_{eff}$ , como la razón entre el tiempo de transmisión de información con el tiempo total utilizado para enviar mensajes, esto es,

$$P_{eff} = \frac{\sum_{i \in N_{node}} \sum_{j=1}^{M(i)} T_{tx}^{(i,j)}}{T_{delay}^{sum}} \quad (2.11)$$

Así,  $P_{eff} \rightarrow 1$  denota que todo el tiempo de retardo es debido a la transmisión de la información y el rendimiento de la red es bueno. De otro modo,  $P_{eff} > 0$  significaría que la mayor parte del retardo se debe a la contención o a la colisión de mensajes.

### Utilización de la red

La utilización de la red,  $P_{util}$ , es definida como la razón entre el tiempo total utilizado para transmitir información y el tiempo total de ejecución, esto es,

$$P_{util} = \frac{\sum_{i \in N_{node}} \sum_{j=1}^{M(i)} T_{tx}^{(i,j)}}{T_{max}} \quad (2.12)$$

$P_{util}$  describe el porcentaje del ancho de banda de la red utilizado por los nodos.  $P_{util} > 1$  significa que la red está saturada, y se tendrá que rediseñar la arquitectura del sistema. Si  $P_{util} < 0$ , entonces significa que hay suficiente ancho de banda libre en la red para ser utilizado.

### Número de mensajes no enviados

Los sistemas de control necesitan que la información requerida sea transmitida exitosamente y de manera inmediata. Si la transmisión de la información en una red induce pérdida de mensajes, se podría deteriorar el rendimiento del sistema y hacerlo inestable.

#### 2.5.2 Calidad de rendimiento del control

Dos criterios, el IAE y el ITAE, son generalmente utilizados para evaluar el diseño y el rendimiento de un sistema de control. El IAE es la integral del valor absoluto del error, y el ITAE es la integral del tiempo multiplicada por el valor absoluto del error [37], así:

$$IAE = \int_{t_0}^{t_f} |y_{des}(t) - y_{act}(t)| dt \quad (2.13)$$

$$ITAE = \int_{t_0}^{t_f} t |y_{des}(t) - y_{act}(t)| dt \quad (2.14)$$

Donde  $t_0$  y  $t_f$  son los tiempos inicial y final del periodo de evaluación,  $y_{des}$  es la trayectoria deseada y  $y_{act}$  es la trayectoria actual. El ITAE da más peso a los errores más tardíos y descuenta la respuesta transitoria, mientras el IAE pesa a todos los errores por igual.

Además, debemos mencionar que en los sistemas de control de movimiento existen dos tipos de errores: el error de seguimiento y el error de contorno [38]. Como vemos en la Figura 2.12, para un instante de tiempo cualquiera, R corresponderá a la posición deseada, P la posición actual, e será el error de seguimiento y E el error de contorno. Nuestro estudio evaluará la calidad de control de seguimiento de una trayectoria deseada como una métrica de la calidad de rendimiento del control.

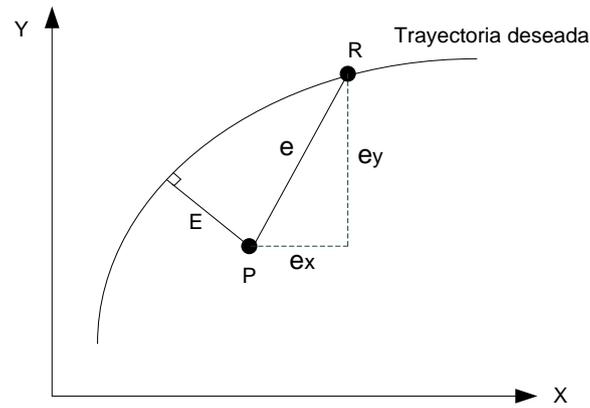


Figura 2.12 Error de seguimiento (e) y error de contorno (E) [38]

Si aplicamos la fórmula del IAE en el seguimiento de la trayectoria deseada podemos expresar lo siguiente:

$$IAE_{tracking} = \int_{t_0}^{t_f} \sqrt{e_x^2(t) + e_y^2(t)} dt \quad (2.15)$$

Donde  $e_x$  corresponde al error entre R y P en el eje X, mientras que  $e_y$  corresponde al error entre R y P en el componente Y.

### 3 Modelo del sistema propuesto

La Figura 3.1 muestra el esquema general del sistema robótico de control en red propuesto. Una manera de modelarlo es a través del desarrollo de modelos individuales para cada componente. Otra manera es modelando todo el sistema como un único módulo tomando en cuenta las relaciones de entrada y salida entre sus componentes. Para obtener un modelo más simplificado se ha elegido el segundo tipo de modelado [39].

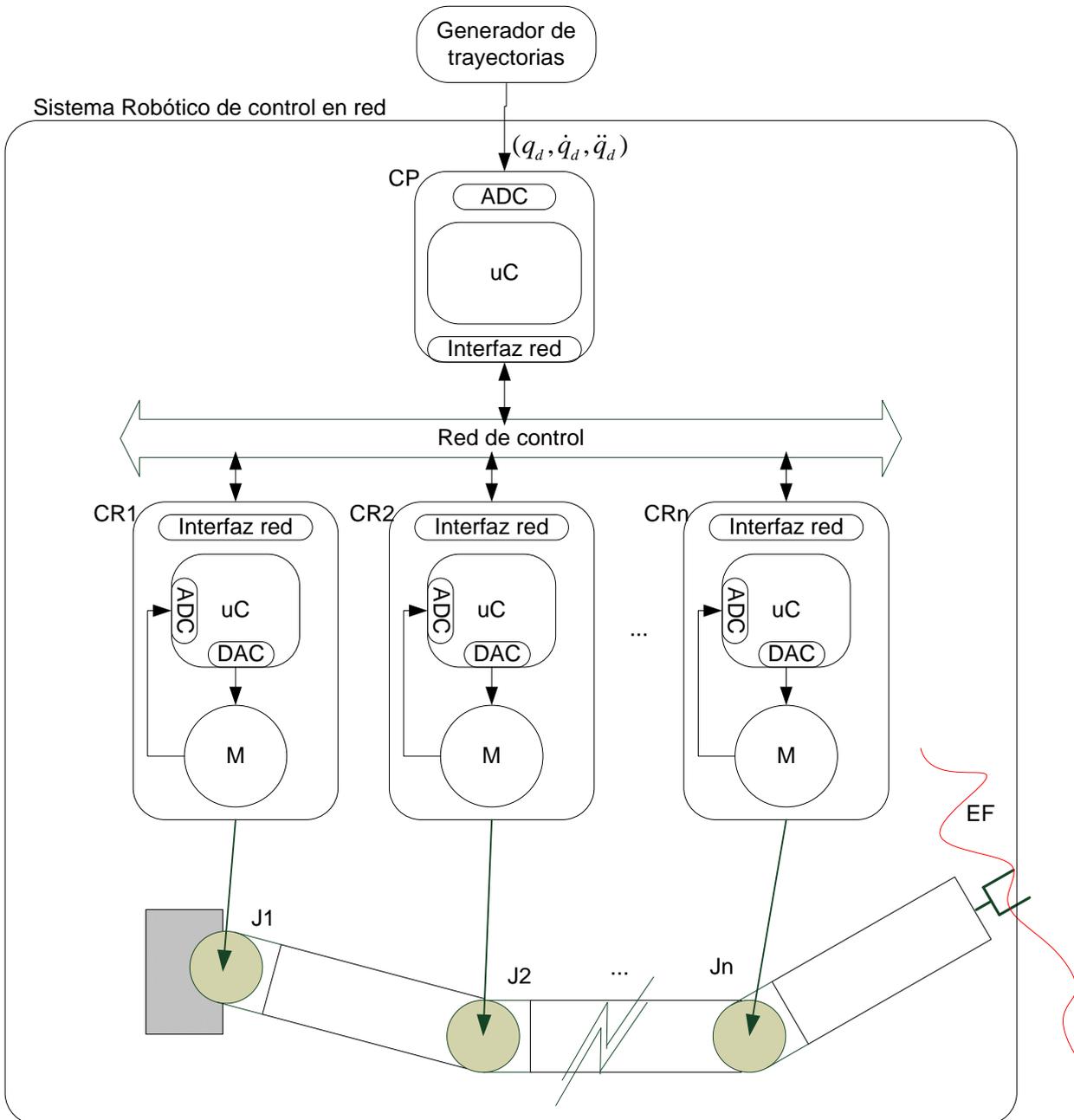


Figura 3.1 Estructura general de un Sistema Robótico de Control en Red

Para tener un sistema que sea flexible y escalable se propone una arquitectura con estructura jerárquica, con el fin de que las tareas no estén todas centralizadas en un solo controlador (ej. Controlador Principal - CP), sino distribuidas también en sus controladores remotos, aliviando la carga de procesamiento en un solo CPU, además de disminuir la congestión en la red.

El sistema está compuesto por tres módulos: el Controlador Principal (CP), la red de control (Red) y los Controladores Remotos (CR's). La idea general consiste en que el CP envía las consignas deseadas (posición - P, velocidad - V y aceleración - A) a los CR's, y, una vez que la salida del motor M coincide con la consigna deseada, cada CR responde al CP con un mensaje de estado. Una vez que el controlador principal - CP haya recibido todos los mensajes de estado de los  $n$  controladores remotos se cierra el ciclo de comunicación; y el CP vuelve a enviar un nuevo conjunto de consignas deseadas a los CR's iniciando un nuevo ciclo de comunicación.

Para probar la calidad del seguimiento de una trayectoria, existe un generador de trayectorias que define un conjunto de consignas de posición, velocidad y aceleración deseadas en el espacio articular,  $q_d(t)$ ,  $\dot{q}_d(t)$  y  $\ddot{q}_d(t)$  respectivamente.

La salida de cada actuador se corresponde con una articulación de un brazo manipulador, cuyo efector final describirá una trayectoria en el espacio operacional. Para facilitar el estudio del seguimiento entre la trayectoria deseada y la generada, se ha elegido un brazo robótico planar de  $n$  articulaciones moviéndose en el plano XY.

Tanto el comportamiento de los tiempos de ciclo de comunicación como el error entre la trayectoria deseada y generada nos servirá de medidas para evaluar el rendimiento del sistema robótico para las distintas configuraciones.

### 3.1 Formulación del problema

Para simplificar el análisis, todo el sistema es visto en su conjunto como un modelo de capas (modelo de cebolla) [40]. Las capas del modelo, ver la Figura 3.2 son:

Capa 0.- El controlador principal lee y coordina el envío de las consignas de las trayectorias a los controladores remotos por la red de control. El controlador principal también gestiona la llegada de los mensajes de estado de los  $n$  controladores remotos.

Capa 1.- Red para mapear las consignas (posición - P, velocidad - V y aceleración - A) desde el CP hacia los CR's y los mensajes de estado entre los CR's y el CP.

Capa 2.- El controlador remoto realiza el interfaz con la red de control para recibir las consignas (P, V, A) y enviar los mensajes de estado del mismo, y ejecuta el lazo de control de movimiento del actuador.

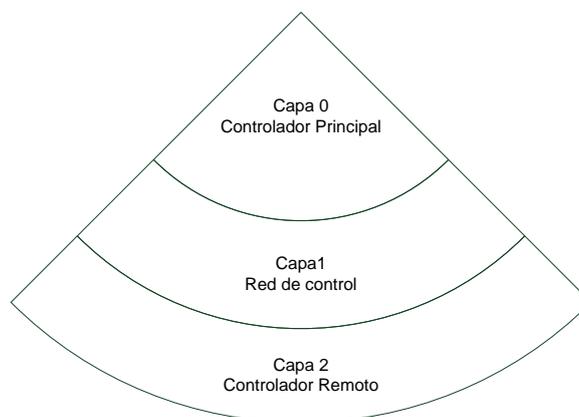


Figura 3.2 Modelo en capas del sistema robótico

Cada capa tiene su conjunto de tareas a realizar, con sus respectivos plazos o deadlines. La secuencia cronológica de las tareas normalmente es como sigue:

En la Capa 0, el orden lógico de las tareas es el siguiente:

Tarea<sub>00</sub>.- tomar lectura de las consignas deseadas del Generador de trayectorias,

Tarea<sub>01</sub>.- encapsular las consignas en paquetes y enviarlas a través de la red de control.

Tarea<sub>02</sub>.- gestionar la llegada de los mensajes de estado de los  $n$  controladores remotos. Una vez que todos los mensajes hayan sido recibidos, el CP dispara la Tarea<sub>00</sub>.

Las tareas son disparadas por eventos y el tiempo de liberación dependerá del algoritmo de planificación del CPU utilizado. El tiempo total de ejecución de la capa 0 dependerá en parte del número de controladores remotos del sistema.

Las tareas en la Capa 1 son:

Tarea<sub>10</sub>.- mapear las consignas deseadas (P, V, A) desde el Controlador Principal a cada Controlador Remoto.

Tarea<sub>11</sub>.- mapear los mensajes de estado desde los controladores remotos al CP.

El tiempo de mapeo en esta capa depende de varios factores como: la topología de la red, ancho de banda, número de nodos, etc.

En la capa 2, el orden lógico de las tareas para un Controlador Remoto es el siguiente:

Tarea<sub>20</sub>.- extraer las consignas deseadas (P, V, A) de la trama recibida en la red de control.

Tarea<sub>21</sub>.- realizar el lazo de control. Si la diferencia entre la consigna deseada y la salida actual del motor es menor a cierto porcentaje de error, el CR dispara la Tarea<sub>22</sub>.

Tarea<sub>22</sub>.- encapsular el mensaje de estado del controlador remoto en paquetes y enviarlos por la red.

### 3.2 Ciclo de comunicación

Como vimos en el Capítulo 2, los tiempos de retardo que se dan en un sistema de control en red afectan su rendimiento e incluso pueden generar inestabilidades, por ende es muy importante analizar su naturaleza.

Utilizando la expresión del tiempo de retardo de la ecuación (2.3), podemos agrupar los tiempos de retardo en todo el sistema en: el tiempo de retardo desde el controlador principal al controlador remoto y viceversa, así:

$$T_{CP \rightarrow CR} = T_{CPpre} + T_{CPespera} + T_{CP \rightarrow CRred} + T_{CPpos} \quad (3.1)$$

$$T_{CR \rightarrow CP} = T_{CRpre} + T_{CRespera} + T_{CR \rightarrow CPred} + T_{CRpos} \quad (3.2)$$

Donde:

$T_{CP \rightarrow CR}$  es el tiempo de retardo del controlador principal al controlador remoto

$T_{CPpre}$  es el tiempo de preprocesamiento en el controlador principal

$T_{CPespera}$  es el tiempo de espera en el CP

$T_{CP \rightarrow CRred}$  es el tiempo de la red cuando el CP envía información al CR

$T_{CPpos}$  es el tiempo de posprocesamiento del CP, una vez haya llegado la información al CR

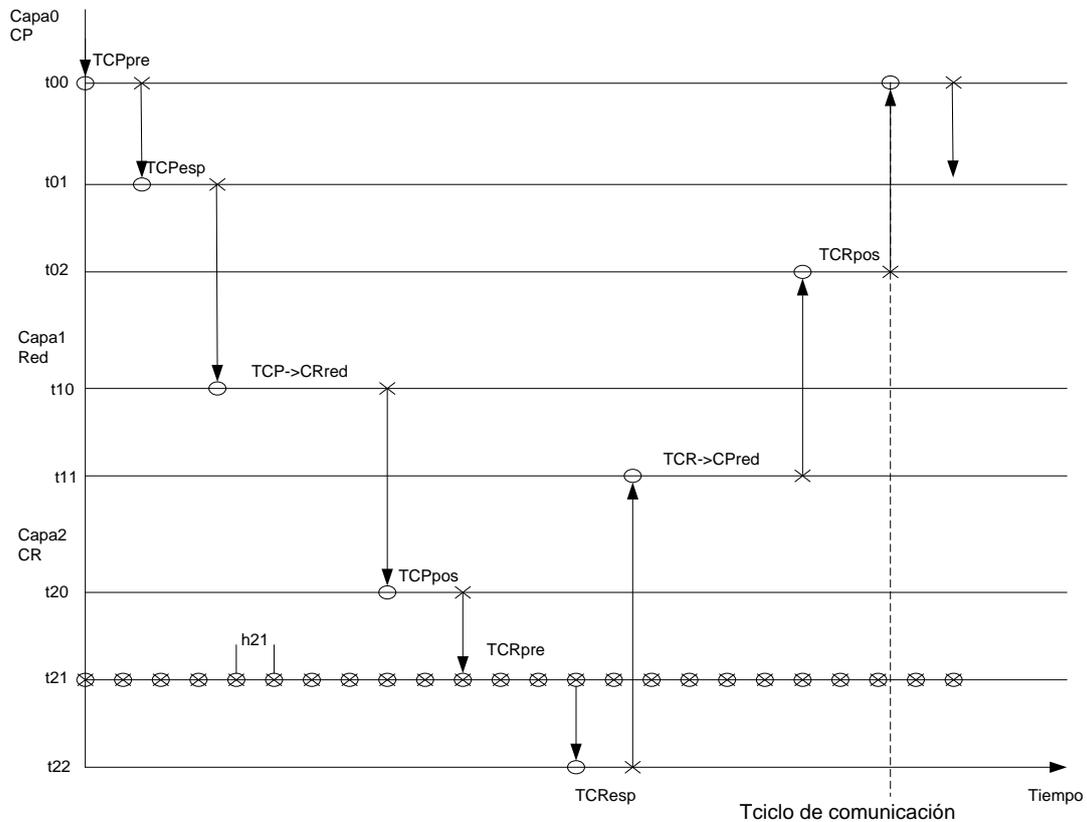


Figura 3.3 Diagrama de tiempos de un ciclo de comunicación del modelo

En la Figura 3.3 se puede observar el diagrama de los tiempos de retardos y su relación con las tareas de las distintas capas de nuestro modelo. Los círculos indican el tiempo de entrada a una tarea, mientras que una cruz indica el tiempo de una salida. El **ciclo de comunicación (CC)** se define como el tiempo comprendido entre el envío de la primera consigna por parte del controlador principal y la llegada del último mensaje de respuesta enviado por el controlador remoto, así tenemos que:

$$CC = T_{CP \rightarrow CR} + T_{CR \rightarrow CP} \quad (3.3)$$

Cabe resaltar que la única tarea periódica es efectuada por  $t21$  en la capa2 del controlador remoto, y tiene un periodo  $h21$ .

Debido a que casi la mayoría de las tareas en el sistema son manejadas por eventos, la posible variabilidad de los tiempos de retardo en ciertas tareas (por. ej. en el tiempo de espera del

controlador remoto) hacen que varíe también su ciclo de comunicación. Esta tesis pretende analizar la variabilidad del ciclo de comunicación cuando se aplican diferentes tipos de configuraciones (tipo de red, tamaño de datos, número de controladores remotos, tasa de datos) al modelo.

### 3.3 Cinemática del robot

Para facilitar el estudio del rendimiento global del sistema robótico con control en red, el tipo de robot propuesto es un brazo manipulador planar con  $n$  articulaciones angulares (*revolute joints*) que se mueve sobre el plano XY, como lo muestra la Figura 3.4. Como se conoce las consignas de posición deseadas en el espacio articular, se aplicará la cinemática directa para conocer la posición deseada del efector final en el espacio operacional.

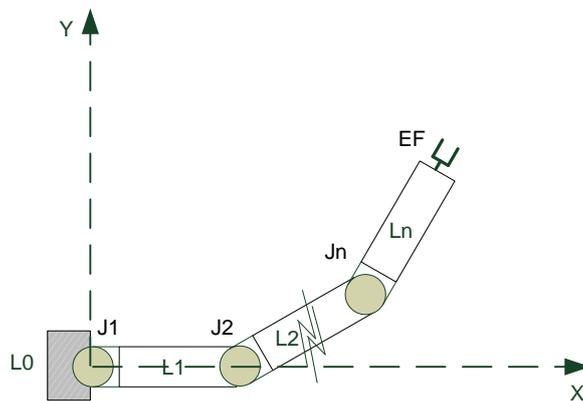


Figura 3.4 Robot manipulador planar con  $n$  articulaciones de giro

De acuerdo a la convención de Denavit-Hartenberg [35], los parámetros iniciales de los eslabones para el brazo robótico son:

Link	$a_i$	$\alpha_i$	$d_i$	$\theta_i$
1	L1	0	0	0
2	L2	0	0	0
	...			
i	$L_i$	0	0	0
	...			
n	$L_n$	0	0	0

Donde

$a_i$  es la longitud del eslabón.

$\alpha_i$  Es el ángulo de separación del eje  $Z_{i-1}$  y el eje  $Z_i$ . Inicialmente  $\alpha_i$  es igual a cero para todas las articulaciones, por lo que el brazo robótico sobre el eje  $X$ , con las coordenadas del efector final en  $[1,0]$ .

$d_i$  es un parámetro para las articulaciones prismáticas por lo que no se toma en cuenta.

$\theta_i$  Es el ángulo que forman los ejes  $X_{i-1}$  y  $X_i$  medido en un plano perpendicular al eje  $Z_{i-1}$ . Debido a que el robot es de tipo planar,  $\theta_i$  es 0.

## 4 Simulación

En este apartado se definirán dos casos de estudio, que son ejecutados mediante simulación. Estos son: a) análisis temporal del ciclo de comunicación y, b) análisis espacial del seguimiento de la trayectoria deseada. En ambos casos se tomarán en cuenta parámetros de simulación como el tipo de red, tasa de datos, tamaño de datos y número de controladores remotos.

### 4.1 Parámetros de simulación

Los parámetros de simulación definen las características iniciales del modelo así como sus restricciones.

#### 4.1.1 Caso a: análisis temporal del ciclo de comunicación

Debido a que un ciclo de comunicación puede variar de una configuración a otra, la simulación para su análisis será realizada en base al número de mensajes que envíe el controlador principal, que será de 1000 mensajes para cada configuración.

Se tomarán en cuenta varias configuraciones como:

- Tipo de red y tasa de datos (Ethernet, Switched Ethernet y CAN).- las tasas de datos de Ethernet y Switched Ethernet son de 1, 5 y 10 Mbps, mientras que CAN tiene una tasa de datos máxima de 1 Mbps.
- Tamaño de datos (3, 6, 9 y 12 bytes).- tomando en cuenta que el controlador principal envía tres consignas (posición, velocidad y aceleración) a cada controlador remoto y que el tamaño de cada consigna varía entre 1 y 4 bytes (rango suficiente para representar un valor de consigna). Para los tamaños de datos de 9 y 12 bytes existirá fragmentación de los mensajes solamente en el caso de la red CAN, que permite un máximo tamaño de paquete de 8 bytes.
- El número de controladores remotos será de 2, 6 y 10 unidades, suficientes para representar los grados de libertad de un brazo robótico.

Como dispositivo controlador, ya sea principal o remoto, se propone para la simulación el uso de un microprocesador de bajo costo (microcontrolador) con un kernel multitarea en tiempo-real [41]. Si tomamos como referencia una frecuencia del oscilador de 4 MHz, el tiempo de instrucción del CPU será de 1us. La política de planificación del CPU es el Earliest deadline first (EDF).

La Tabla 4.1 muestra las características de las tareas del controlador principal y remoto. El número de instrucciones está relacionado con el número de líneas de código de cada tarea.

Tabla 4.1 Parámetros de las tareas del controlador principal y controlador remoto.

Capa	Tareas	Tipo	Deadline	Período	Nro. de Instrucciones
0 Controlador Principal					
	Tarea <sub>00</sub>	Aperiódica	0,001	0	10
	Tarea <sub>01</sub>	Aperiódica	0	0	25
	Tarea <sub>02</sub>	Aperiódica	0,002	0	10
2 Controlador Remoto					
	Tarea <sub>20</sub>	Aperiódica	0,01	0	4
	Tarea <sub>21</sub>	Periódica	0	0,001	45
	Tarea <sub>22</sub>	Aperiódica	0,001	0	5

Para poder realizar las comparaciones lo más uniformes posibles, se ha especificado que la longitud del robot sea fija; y además las longitudes de los eslabones sean todos iguales, por lo que el tamaño de cada eslabón será igual a:

$$L_i = L_{robot} / n; \forall i = 1, 2, \dots, n \quad (4.1)$$

En la Tabla 4.2 podemos observar los parámetros del robot, del actuador (motor DC) y del control seleccionados para la simulación.

**Tabla 4.2 Características del robot, actuador y parámetros del control.**

<b>Característica</b>	<b>Valor</b>
Longitud del robot	1m
<u>Parámetros del actuador</u>	
Marca	DC Faulhaber serie 2342-018CR
Constante de torque (Kt)	2.07E-2 N*m/A
Constante del motor (Kv)	2.07E-2 V*s/rad
Resistencia de la armadura (Ra)	4.10 Ohm
Inercia del eje del motor (Im)	6.2E-7 Kg*m <sup>2</sup>
<u>Parámetros de control</u>	
voltaje máximo del controlador (Umax)	12 V
Porcentaje de error admitido	2.0 %
Período de muestreo (h)	0.001 s
Tipo de control	P,I con retroalimentación de posición y compensación descentralizada feedforward
Constante del transductor de posición (KTP)	1
KP	0.001
TP	1100

Debido a que el presente trabajo representa una primera aproximación para estudiar el rendimiento de un sistema robótico de control en red, el estudio del sistema se lo hará en condiciones ideales, es decir: pérdidas de paquetes, tiempos de propagación e interferencias en la red de control no serán tomados en cuenta, el tamaño del mensaje entre el controlador principal y un controlador remoto tendrán la misma longitud en cualquier dirección de la comunicación. De la misma manera en los controladores remotos no se consideran las perturbaciones generadas por las características dinámicas de cada eslabón.

En la Figura 4.1 se puede observar algunos bloques utilizados para llevar a cabo la simulación del sistema robótico de control en red, mismos que han sido desarrollados en Matlab utilizando librerías como TrueTime 2.0 y Robotics Toolbox.

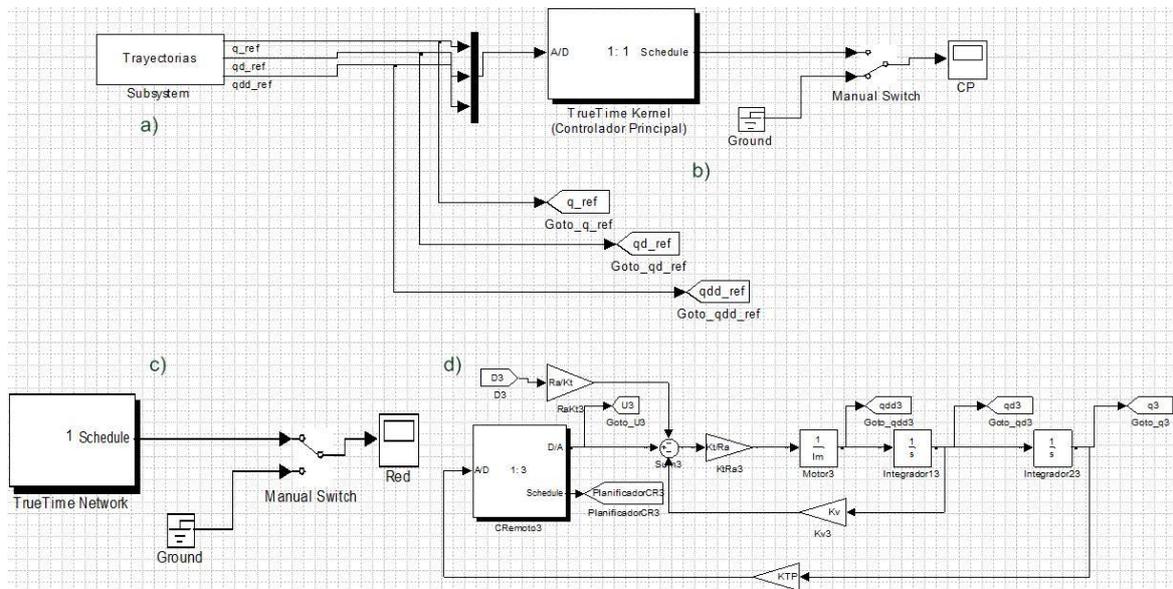


Figura 4.1 Bloques de simulación del sistema robótico en red. a) Generador de trayectorias, b) Controlador principal, c) Red de control y d) un controlador remoto

#### 4.1.2 Caso b: análisis espacial del seguimiento de la trayectoria deseada

Debido a que la calidad del control en el seguimiento de la trayectoria será evaluada por el IAE, se requiere que para este caso, la simulación sea realizada por tiempo, que en nuestro caso será de 1 segundo.

Las configuraciones y demás parámetros de simulación son similares a las del caso a.

### 4.2 Generador de trayectorias deseadas

Se ha elegido la siguiente función de una posición deseada en el espacio articular con respecto al tiempo definida como [42]:

$$q_d(t) = b(1 - e^{-2t^3}) + c(1 - e^{-2t^3}) \sin(\omega t) \quad (4.2)$$

Para  $b = \pi/4$  [rad],  $c = \pi/9$  [rad] y  $\omega = 10$  [rad/s].

Esta función de trayectoria de posición ha sido elegida por las siguientes características:

- La trayectoria contiene un término sinusoidal que permite evaluar el rendimiento del controlador siguiendo movimientos periódicos relativamente rápidos.
- La trayectoria también contiene un término de incremento lento que permite llevar al robot a su punto de operación sin llevar a los actuadores a un estado de saturación no deseado.

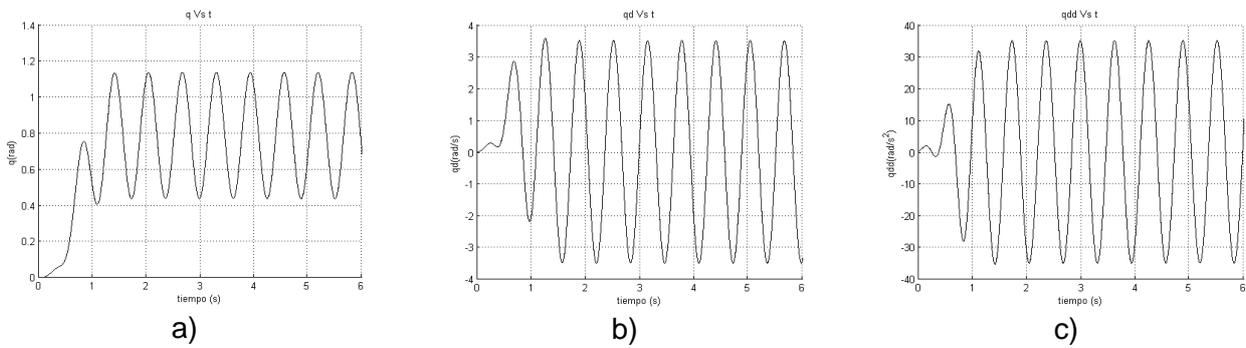
De la expresión de las trayectorias de posición deseadas, se obtiene analíticamente las expresiones para la velocidad y aceleración mediante la derivada y doble derivada, obteniendo:

$$\dot{q}_d(t) = 6bt^2 e^{-2t^3} + 6ct^2 e^{-2t^3} \sin(\omega t) + (c - ce^{-2t^3}) \cos(\omega t) \omega \quad (4.3)$$

$$\ddot{q}_d(t) = 12bte^{-2t^3} - 36bt^4 e^{-2t^3} + 12cte^{-2t^3} \sin(\omega t) - 36ct^4 e^{-2t^3} \sin(\omega t) + \quad (4.4)$$

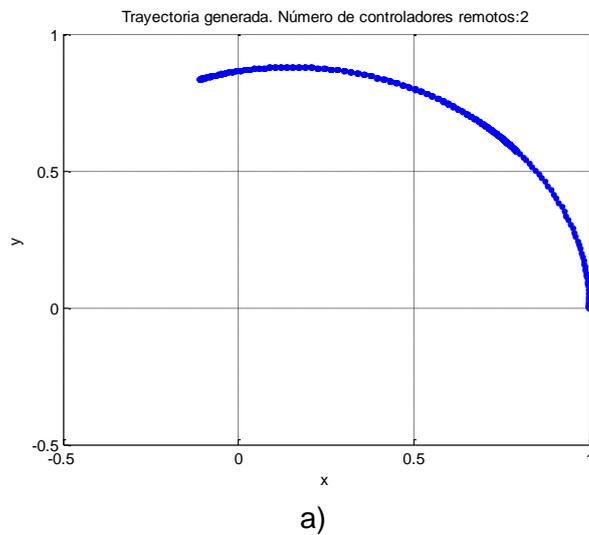
$$+ 12ct^2 e^{-2t^3} \cos(\omega t) \omega - (c - ce^{-2t^3}) \sin(\omega t) \omega^2$$

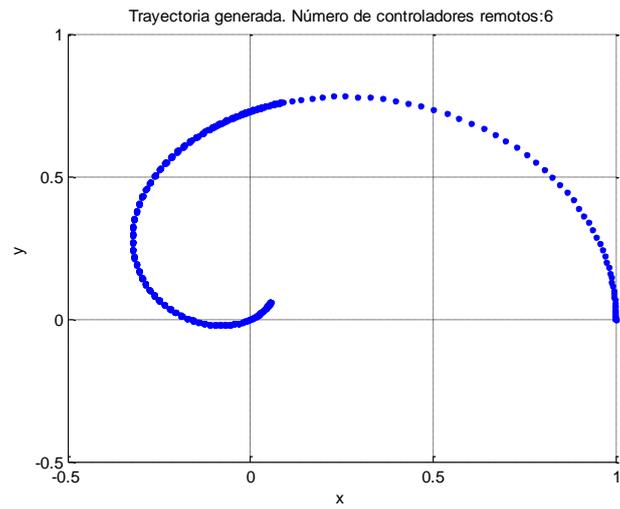
La Figura 4.2 muestra las trayectorias de referencia de posición, velocidad y aceleración deseadas en el espacio articular.



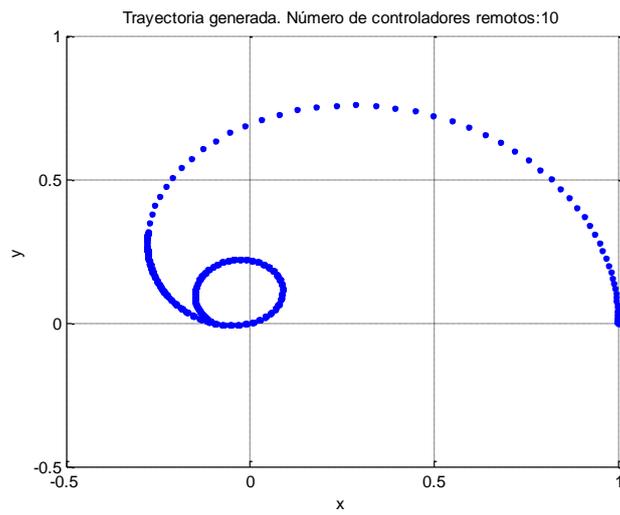
**Figura 4.2** Trayectorias de referencia deseadas en el espacio articular. a) Posición Vs t. b) Velocidad Vs t. c) Aceleración Vs t.

En la Figura 4.3 se observan las trayectorias generadas por el movimiento de  $n$  articulaciones. Si consideramos una longitud del robot constante igual a 1m para cualquier número de grados de libertad (DOF), entonces se obtienen diferentes trayectorias deseadas en el espacio operacional, como lo muestra





b)



c)

Figura 4.3 Trayectorias deseadas generadas en el espacio operacional (plano XY). a) 2 grados de libertad. b) 6 grados de libertad. c) 10 grados de libertad

## 5 Análisis de resultados

En esta sección haremos el análisis de los resultados de la simulación, tomando como punto de partida los dos casos establecidos en el capítulo anterior: estudio temporal de los ciclos de comunicación y estudio espacial del seguimiento de trayectoria.

### 5.1 Ciclos de comunicación

El ciclo de comunicación corresponde al tiempo comprendido entre el envío de la primera consigna por parte del controlador principal y del último mensaje respondido por el controlador remoto. Ya que estos tiempos pueden ser variables, es necesario analizar sus características.

#### 5.1.1 Diferentes tamaños de datos (TD). R=1, CR=6

En la Figura 5.1 se observa dos tipos de gráficas, la distribución de frecuencias del ciclo de comunicación (PDF) y la distribución de frecuencias acumulada del ciclo de comunicación (CDF). Se puede notar que los tamaños de los datos propuestos afectan el tiempo de ciclo en una red CAN. Ya que el estándar CAN no considera un mínimo tamaño de paquete de datos, el tamaño de la trama varía en relación al TD, variando el tiempo de transmisión por la red y afectando finalmente el ciclo de comunicación (CC).

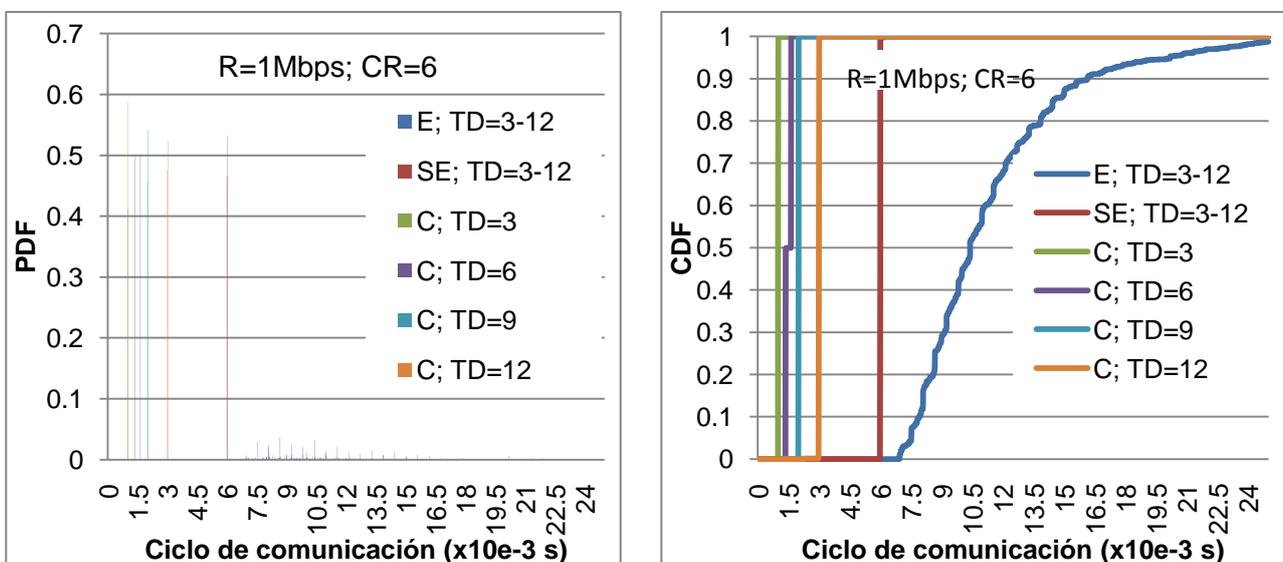


Figura 5.1 Ciclos de comunicación con TD=3, 6, 9, 12 bytes

Por el contrario, para las redes Ethernet y Switched Ethernet los CC no varían para los diferentes valores propuestos de TD, por ser menores que el mínimo tamaño de paquete de datos (46 bytes) establecido por el estándar para Ethernet.

Los CC de una red Ethernet son mayores que los de una Switched Ethernet, debido a que el control de acceso al medio en la Ethernet permite colisiones en la red. Cuando varios nodos tratan de acceder simultáneamente a la red, las colisiones que se pueden dar generan retardos. La dispersión en Ethernet se debe a la aleatoriedad en la ventana de contención cuando ocurren estas colisiones.

La utilización de buffers permiten conexiones punto a punto entre los nodos y la red Switched Ethernet, evitando colisiones en la red, por ende la dispersión vista en la CDF es nula, ya que no existe ventana de contención. En CAN la baja dispersión se debe al mecanismo de control de

acceso al medio por prioridad de los mensajes, evitando de esta manera colisiones y mejorando su comportamiento.

Finalmente, de la Figura 5.1 también podemos observar que la probabilidad de que los CC sean menores o iguales a 25 ms es del 100%, sin importar que tipo de red de control se esté utilizando.

### 5.1.2 Diferentes tasas de datos (R). CR=6, TD=12

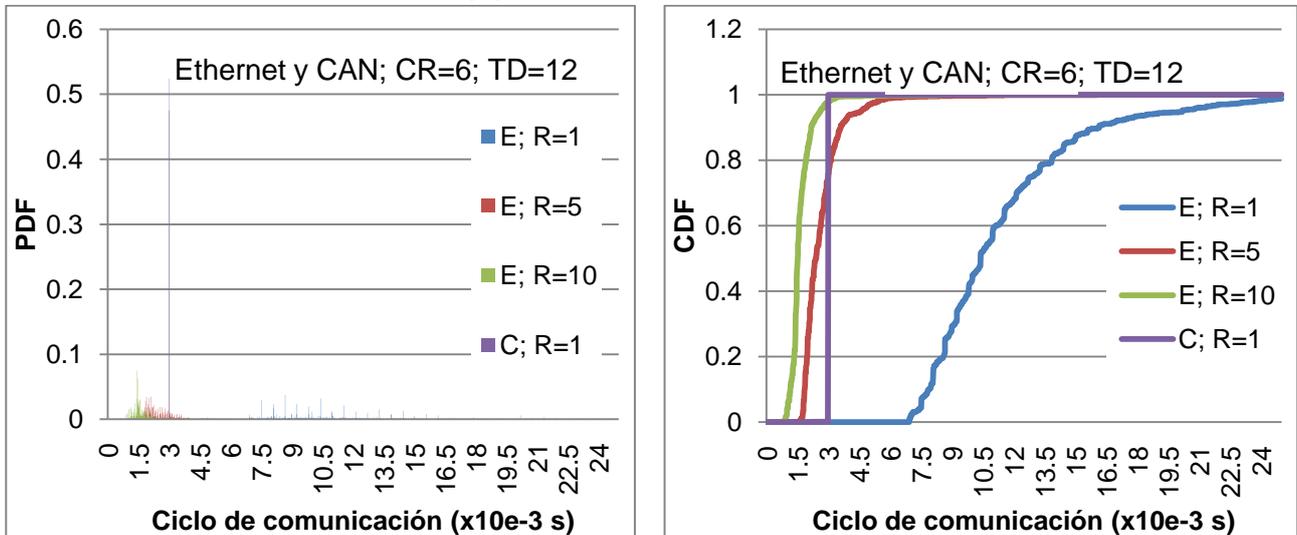


Figura 5.2 Ciclos de comunicación con R=1, 5, 10 Mbps para red Ethernet y CAN

El incremento de la tasa de datos en la red Ethernet hace que disminuyan los valores de los CC y su dispersión

De la Figura 5.2 observamos que la probabilidad para CAN a 1Mbps de que el CC sea menor a 3ms es prácticamente del 100%, del 95% para una red Ethernet a 10Mbps y se reduce al 70% para una red Ethernet a 5Mbps.

Si se requiere tener una probabilidad del 100% para una red de 5Mbps el valor del CC aumenta a 10ms aproximadamente.

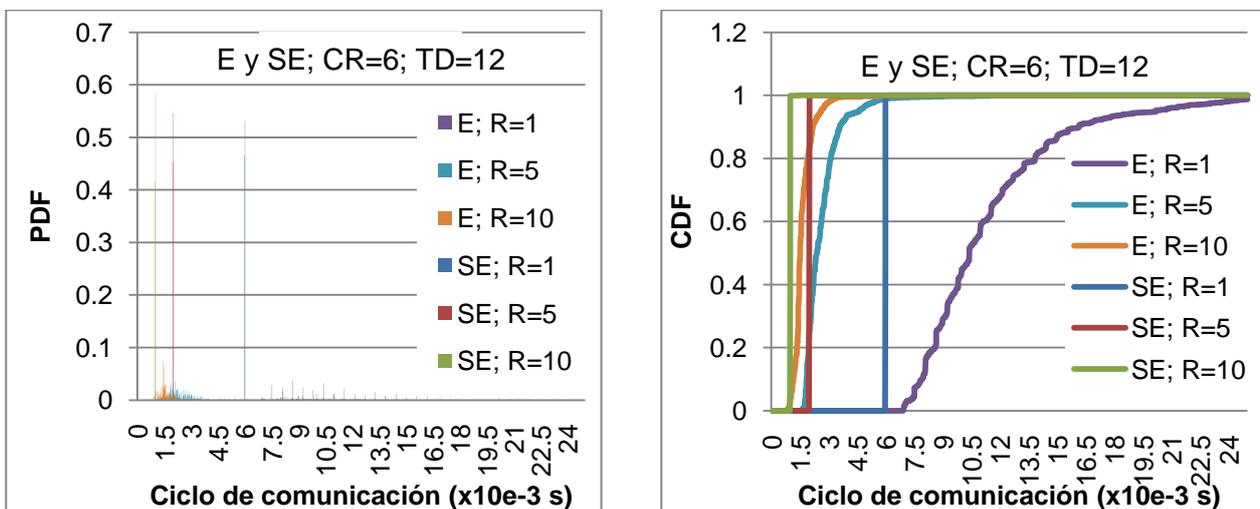


Figura 5.3 Ciclos de comunicación con R=1, 5, 10 Mbps para red E y SE

En la grafica Figura 5.3 vemos la disminuci3n de los CC a medida que aumenta la tasa de datos de las redes. Tambi3n podemos advertir la naturaleza aleatoria en la red Ethernet comparada con la Switched Ethernet.

### 5.1.3 Diferentes n3meros de controladores remotos (CR). R=1, TD=12

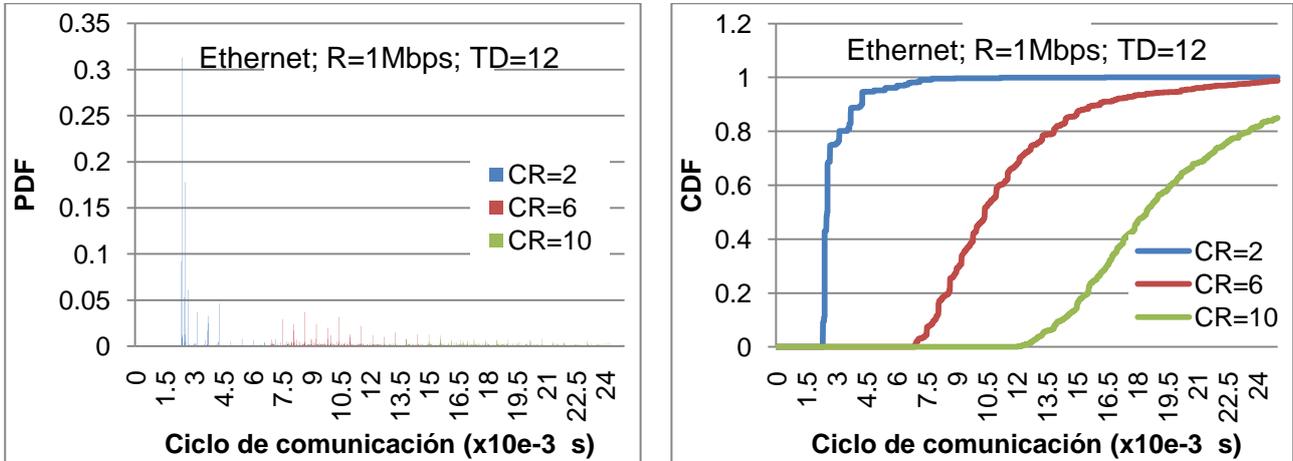
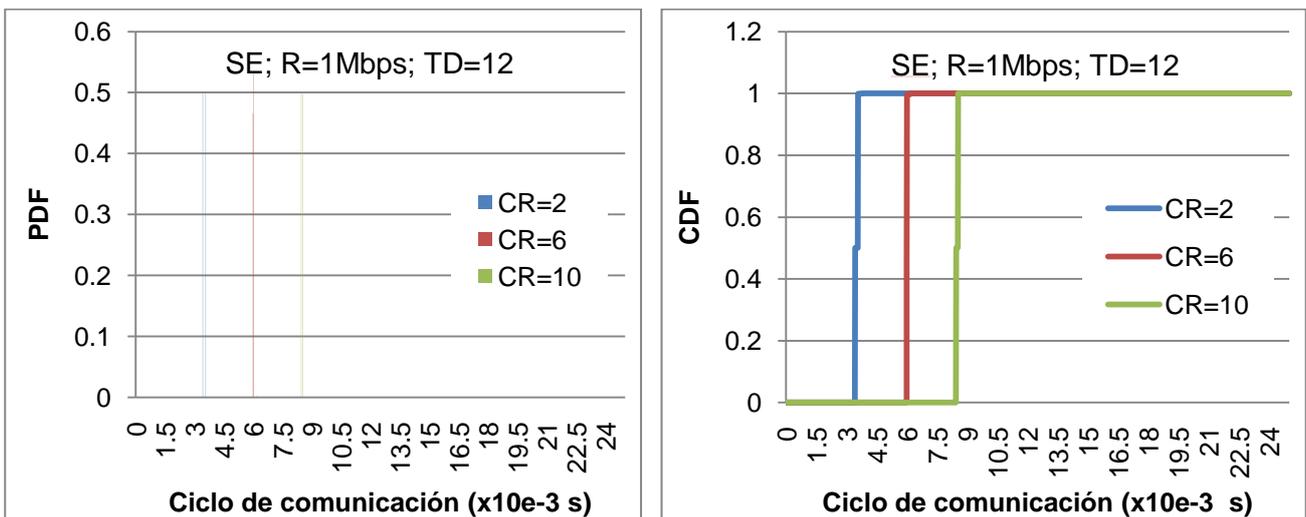


Figura 5.4 Ciclos de comunicaci3n con CR=2, 6, 10 para red E

Al aumentar el n3mero de CR se incrementan los valores promedios de los CC y tambi3n su dispersi3n, por las colisiones en la red.



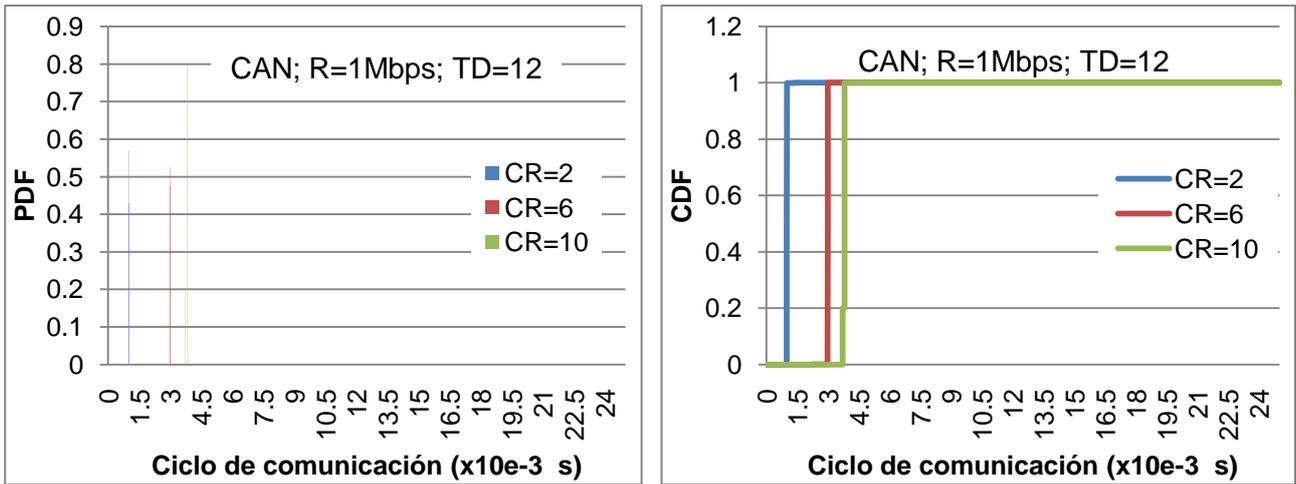


Figura 5.5 Ciclos de comunicación con CR=2, 6, 10 para redes CAN y SE

En la Figura 5.4 puede ver que el incremento del número de controladores remotos hace que también aumente el ciclo de comunicación promedio mas no su dispersión, debido a las características determinísticas de las redes SE y CAN.

## 5.2 Seguimiento de trayectoria

### 5.2.1 Diferentes tamaños de datos (TD). R=1, CR=6

En la Figura 5.6, a modo de referencia hemos graficado las trayectorias de la posición, velocidad y aceleración en el espacio articular para el tiempo de simulación de 1 s, con la finalidad de poder evaluar el comportamiento del error de tracking en el tiempo.

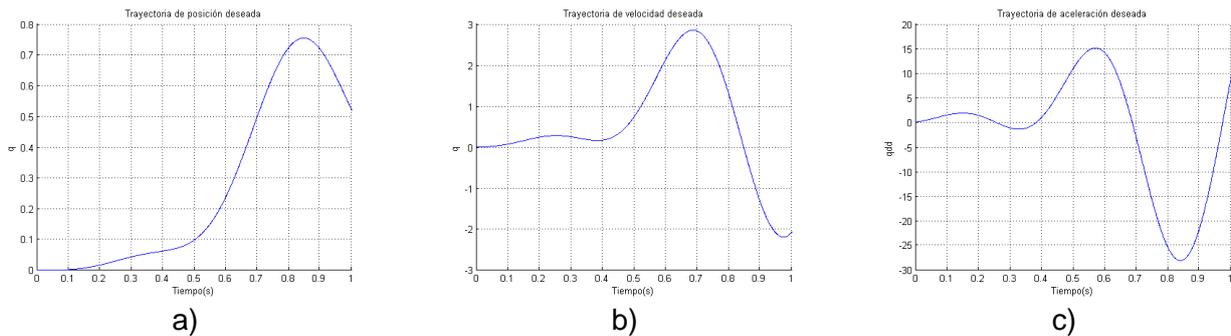


Figura 5.6 Trayectorias deseadas en el espacio articular para un tiempo de simulación de 1s. a) Posición, b) velocidad y c) aceleración.

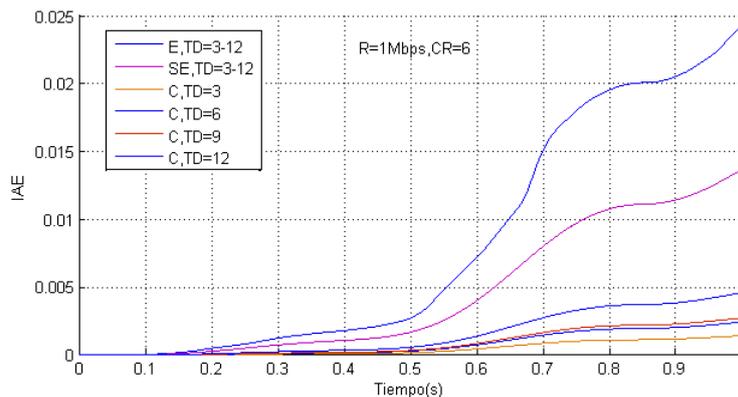


Figura 5.7 IAE del error de tracking para TD=3, 6, 9 y 12

De la Figura 5.7 podemos indicar que en la red CAN la variación del tamaño de los datos si influye en el error de seguimiento, siendo el TD=3 el de menor valor. Por el contrario, la red que más acumula error es la Ethernet. Podemos notar además que la curva tiene un punto donde el error se mantiene constante ( $t = 0.85s$ ), que es el punto donde la velocidad en las articulaciones es igual a 0.

### 5.2.2 Diferentes tasas de datos (R). CR=6, TD=12

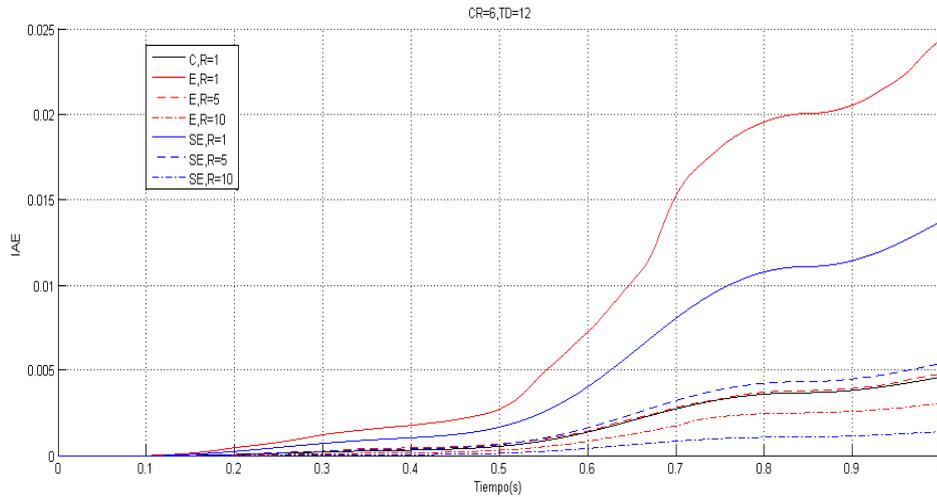
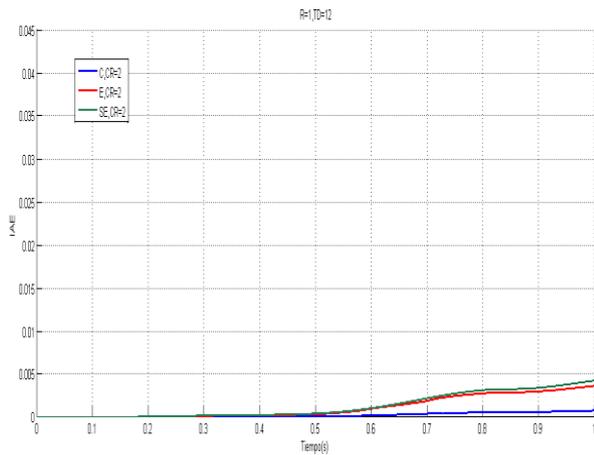


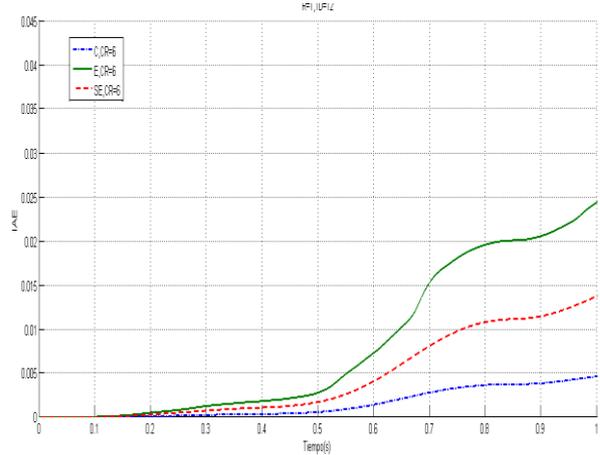
Figura 5.8 IAE del error de tracking para TD=3, 6, 9 y 12

En la Figura 5.8 vemos que la red que tiene el menor error acumulado es la red Switched Ethernet, para 10Mbps, mientras que la red Ethernet el mayor error acumulado. A pesar de que la red CAN no puede ir más allá de 1Mbps, vemos que el error acumulado está al nivel de una red Ethernet a 5Mbps.

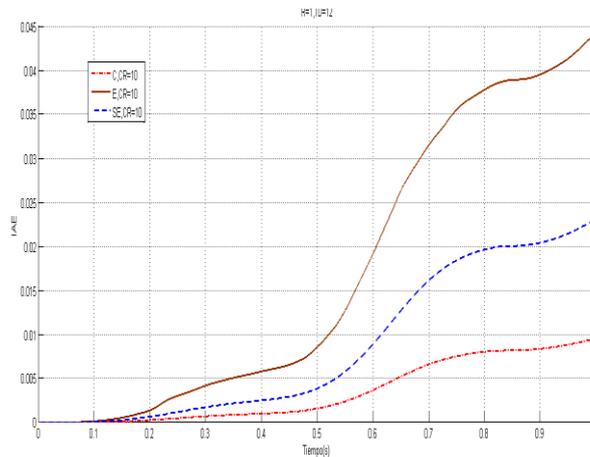
### 5.2.3 Diferentes números de controladores remotos (CR). R=1, TD=12



a)



b)



c)

Figura 5.9 IAE del error de tracking para CR=2, 6 y 10

Dado que en este trabajo se ha fijado una trayectoria deseada a nivel de articulaciones, de acuerdo a la ecuación (4.3), la trayectoria realizada por el efector final es diferente para cada conjunto de actuadores, razón por la cual en la Figura 5.9 se ha considerado el error espacial del efector final por un conjunto de motores para diferentes tipos de red.

## 6 Conclusiones y futuros trabajos

Se realizó el modelo de un sistema robótico de control en red, el mismo que fue recreado con una herramienta de simulación.

La simulación del sistema abarca el diseño de las tareas llevadas a cabo por el controlador principal y los controladores remotos, el diseño del control de movimiento de cada articulación, la cinemática directa del robot para convertir el movimiento en el espacio articular al espacio operacional.

En condiciones ideales, sin interferencias en la red ni perturbaciones en los actuadores, el comportamiento de los ciclos de comunicación tiene una baja dispersión tanto para la red Switched Ethernet como CAN, no así en la red Ethernet, que a pesar de trabajar a una velocidad de transmisión superior a la red CAN, no mejora significativamente su rendimiento para las condiciones propuestas de tamaño de dato y número de controladores remotos.

Se proponen varios trabajos futuros como:

- Validar los resultados obtenidos con la experimentación de un sistema robótico de control en red.
- Añadir al modelo del sistema las interferencias en la red, así como las perturbaciones en el eje de los motores.
- Añadir más configuraciones al modelo del sistema, con otros tipos de redes (ej.: FlexRay, Profinet, etc.)
- Recrear el modelo del sistema con redes inalámbricas (ej.: WLAN, zigbee, etc)

## 7 Bibliografía

1. **Wang, Fei Yue y Derong, Liu.** *Networked Control Systems Theory and Applications*. Londres : Springer-Verlag, 2008. ISBN 978-1-84800-214-2.
2. *Integrated Control and Scheduling*. **K-E Arzen, B. Bembardsson, J.Eker, A. Cervin, K. Nilsson, P. Persson, L.Sha.** Dept. Automatic Control, Lund Institute of Technology : Research report, 1999. ISSN 0820-5316.
3. *Control Scheduling Codesign of RealTime Systems: The Control Server Approach*. **Anton Cervin, Johan Eker.** s.l. : Journal of Embedded Computing, 2004.
4. **Lian, Feng-Li.** *Analysis, Design, Modeling, and Control of Networked Control Systems*. Michigan : Ph.D. dissertation, University of Michigan, 2001.
5. **Nilsson, J.** *Real-time control systems with delays*. Lund : Ph.D.dissertation, Dept. Automatic Control, Lund Institute of Technology, 1998.
6. *Simulation Based Performance Analysis of Networked Control Systems with Resource Constraints*. **Xia, Feng.** Busan : The 30th Annual Conference of the IEEE Industrial Electronics Society, 2004.
7. *Networked control design for linear Systems*. **PV, Zhivoglyadov y RH, Middleton.** 743-750, s.l. : Automatica, 2003, Vol. 39.
8. *Stability of networked control systems*. **Wei Zhang, Michael S. Branicky.** 1, s.l. : IEEE Control System Magazine, Febrero 2001, Vol. 21.
9. *Network-based controller adaptation based on QoS negotiation and deterioration*. **Tipsuwan, Y. y Chow, M.-Y.** Denver, CO : The 27th annual conference of the IEEE industrial electronics society (IECON01), 2001. Vol. 3 (pp. 1794–1799).
10. *A design methodology for distributed control systems to optimize performance in the presence of time delays*. **Yook, J.K., Tilbury, D.M. y Soparkar, N.R.** Chicago, IL : Proceedings of the 2000 American control conference, 2000. Vol. 3 (pp. 1959–1964).
11. *Performance evaluation of control networks: Ethernet, ControlNet, and DeviceNet*. **Feng-Li, Lian, Moyne, J.R. y Tilbury, D.M.** 1, s.l. : Control Systems Magazine, IEEE, 2001, Vol. 21. 0272-1708.
12. *Control methodologies in networked control systems*. **Tipsuwan, Yodyium y Chow, Mo-Yuen.** Pages 1099-1111, s.l. : Control Engineering Practice, October 2003, Vols. Volume 11, Issue 10. ISSN 0967-0661.
13. **Liu, Guo-Ping.** *Analysis and Design of Networked Predictive Control Systems*. [aut. libro] Fei-Yue Wang y Derong Liu. *Networked Control Systems Theory and Applications*. London : Springer, 2008.
14. *How does control timing affect performance? Analysis and simulation of timing using Jitterbug and TrueTime*. **Cervin, A., y otros.** no.3, pp. 16-30, s.l. : Control Systems Magazine, IEEE, 2003, Vol. 23. URL: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1200240&isnumber=27020>.

15. *Feasibility analysis for networked control systems by simulation in Modelica*. **Liu, Liu y Georg, Frey**. s.l. : ETFA 2008, 2008.
16. *Networked control system co-simulation for co-design*. **Branicky, M.S., Liberatore, V. y Phillips, S.M.** s.l. : American Control Conference, 2003. Proceedings of the 2003 , 2003. vol.4, pp. 3341-3346.
17. *Simulation-based performance analysis of distributed systems*. **Schwarz, P. y Donath, U.** s.l. : Parallel and Distributed Real-Time Systems, 1997. Proceedings of the Joint Workshop on , 1-3 Apr 1997. pp.244-249.
18. *Ethernet's winning ways*. **Kaplan, G.** 113-115, s.l. : IEEE Spectrum, 2001, Vol. 38(1).
19. *Control methodologies in networked control systems*. **Yodyium Tipsuwan, Mo-Yuen Chow.** 1099–1111, s.l. : Control Engineering Practice , 2003, Vol. 11.
20. *Real-time open control architectures and system performance*. **Y., Koren, y otros.** s.l. : CIRP Annals-Manufacturing Technology, 1996, Vols. 45(1):377-380.
21. *Modern fieldbus communication architectures for real-time industrial applications*. **S.A., Koubias y Papadopoulos, G.D.** s.l. : Computers in Industry, August, 1995, Vols. 26:243-252. ISSN: 0166-3615.
22. **F.-L., Lian, J.R., Moyne y D.M., Tilbury.** Network Protocols for Networked Control Systems. [aut. libro] Dimitrios Hristu-Varsakelis y William S. Levine. *Handbook of Networked and Embedded Control Systems, 1.* s.l. : SpringerLink, 2005.
23. **D., Bertsekas y R., Gallager.** *Data Networks*. s.l. : Prentice-Hall, Englewood Cliffs, NJ, 1992.
24. *Implementing Ethernet in the industrial environment*. **B. J., Casey.** pages 1469-1477, Seattle, WA. : Proceedings of IEEE Industry Applications Society Annual Meeting, October, 1990, Vol. 2.
25. **Tanenbaum, A. S.** *Computer Networks*. Upper Saddle River, NJ : Prentice-Hall. 3ra ed., 1996.
26. *Field Communication for Process Engineering*. **P. Berrie, M. Hennemann, J. Schulze, T. Schworer.** s.l. : Endress+Hauser, 1995.
27. *CAN Specification. Version 2.0*. **GmbH, Robert Bosch.** 1991.
28. **Pazul, Keith.** *AN713 Controller Area Network (CAN) Basics*. 1999.
29. *DeviceNet specifications*. 1997.
30. *Building a better fieldbus*. **G., Paula.** pag. 90-92, s.l. : Mechanical Engineering, June 1997.
31. *Calculating controller area network (CAN) message response time*. **K., Tindell, A., Burns y A. J., Wellings.** s.l. : Control Engineering Practices, Agosto, 1995, Vols. 3(8):1163-1169.
32. *The Large Error First (LEF) Scheduling Policy for Real-Time Control Systems*. **J., Yopez, P., Marti y Fuertes, J.M.** Cancun, Mexico : In Work-in-Progress Session of the 24th IEEE Real-Time Systems Symposium (RTSS03), December, 2003.

33. *LEF closed-loop scheduling policy for real-time control systems*. **Yopez, J., y otros.** s.l. : 10th IEEE Conference on Emerging Technologies and Factory Automation, 2005. ETFA 2005, 19-22 Sept. 2005. vol.1, no., pp.4 pp.-280.
34. *Control methodologies in networked motion control systems*. **Bing, Chen, y otros.** Guangzhou : Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, 18-21 August 2005.
35. **L., Sciavicco y B., Siciliano.** *Modelling and Control of Robot Manipulators*. London : Springer-Verlag, 2005.
36. **Kaiser, D.** *Fundamentals of Servo Motion Control*. s.l. : Parker Compumotor whitepaper, 2001.
37. *Network Design Consideration for Distributed Control Systems*. **Feng-Li, Lian, James, Moyne y Dawn, Tilbury.** 2, s.l. : IEEE Transactions on Control Systems Technology, March 2002, Vol. 10.
38. *Contour Control for Direct Drive XY Table*. **Wang, Limei, Jin, Fuying y Sun, Yibiao.** Changchun, China : Proceedings of the 2009 IEEE International Conference on Mechatronics and Automation, August 9 - 12, 2009.
39. *Speed Controller Strategies for Distributed Motion Control via Ethernet*. **Lilantha, Samaranayake, Sanath, Alahakoon y Kirthi, Walgama.** Houston, Texas : Proceedings of the 2003 IEEE International Symposium on Intelligent Control, October 5-8.2003.
40. *Analysis of Time Delays in Synchronous and Asynchronous Control Loops*. **Bjorn, Wittenmark, Ben, Bastian y Johan, Nilsson.** Tampa, Florida USA : Proceedings of the 37th IEEE Conference on Decision & Control, December 1998.
41. **Marau, Ricardo, y otros.** *Implementing Flexible Embedded Control on a Simple Real-Time Multitasking Kernel* . Barcelona, Spain : Research report ESAII-RR-07-10, Automatic Control Department, Technical University of Catalonia,, April, 2007.
42. **R., Kelly, V., Santibáñez y A., Loría.** *Control of Robot Manipulators in Joint Space*. London : Springer-Verlag, 2005. ISSN 1439-2232.
43. *An Internet-based real-time control engineering laboratory*. **Overstreet, J. W., & Tzes, A.** 19–34, s.l. : IEEE Control Systems Magazine, 1999, Vol. 19(5).
44. *Gain adaptation of networked mobile robot to compensate QoS deterioration*. **Tipsuwan, Y., & Chow, M.-Y.** Sevilla, Spain : The 28th annual conference of the IEEE industrial electronics society (IECON 02) 4 (pp. 3146–3151), 2002.
45. *Planning and control of internet-based teleoperation*. **Tarn, T.-J., & Xi, N.** Boston, MA. : Proceedings of SPIE: Telemanipulator and telepresence technologies V, Vol. 3524 (pp. 189–193). , 1998.
46. **Bekey, George.** *Robotics: State of the art and future challenges*. Londres : Imperial College Press, 2008. 978-1-84816-006-4.
47. *An application of sensor Networks for syndromic surveillance*. **Chang, C-K, Overhage, JM y Huang, J.** Tuscon, AZ : Proceedings of IEEE Networking, Sensing and Control, 2005. 191–196.

48. *Data fusion in sensor networks*. **Durrant-Whyte, H.** Sydney, NSW, Australia : Proceedings of the IEEE International Conference on Video and Signal Based Surveillance, 2006. 39–39.
49. *Embedded sensor networked operating system*. **Park, S, y otros.** Gyeongju, Korea : Proceedings of Ninth IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing, 2006. 117–124.
50. *Design of energy-efficient wireless sensor networks with censoring, on-off, and censoring and on-off sensors based on mutual information*. **Yamasaki, K y Ohtsuki, T.** Stockholm, Sweden : Proceedings of the IEEE 61st Vehicular Technology Conference, 2005. 2:1312–1316.
51. *Vision-based localization of leader-follower formations*. **GL, Mariottini, y otros.** Sevilla, España : Proceedings of the 44th IEEE Conference on Decision and Control, and European Control Conference, 2005. 635–640.
52. *Real time, dynamic target tracking using image motion*. **IAK, Saeed y NV, Afzulpurkar.** Taipei, Taiwan : Proceedings IEEE International Conference on Mechatronics, 2005. 241–246.
53. *A controller to avoid both occlusions and obstacles during a vision-based navigation task in a cluttered environment*. **D, Folio y V, Cadenat.** Sevilla, España : Proceedings of the 44th IEEE Conference on Decision and Control, and 2005 European Control Conference, 2005. 3898–3903.
54. *Vision based following of locally linear structures using an unmanned aerial vehicle*. **S, Rathinam, y otros.** Sevilla, España : Proceedings of the 44th IEEE Conference on Decision and Control, and European Control Conference, 2005. 6085–6090.
55. **TB, Sheridan.** *Telerobotics, automation, and human supervisory control*. Cambridge, MA : The MIT Press, 1992.
56. *Fieldbus-conflicting "standards" emerge, but interoperability is still elusive*. **J., Pinto.** UK : Design Engineering magazine, Oct.99.
57. *The OSI reference model*. **Day, J.D. y Zimmermann, H.** no.12, pp. 1334-1340, s.l. : Proceedings of the IEEE, Dec. 1983, Vol. vol.71.
58. *Process control communications: Token bus, CSMA/CD, or token ring?* **J. D., Wheelis.** s.l. : ISA Transactions, July, 1993, Vols. 32(2):193-198.
59. *ControlNet specifications*. 1998.
60. **Feng-Li, Lian, James R., Moyne y Dawn M., Tilbury.** *Performance Evaluation of Control Networks: Ethernet, ControlNet and DeviceNet*. Michigan : Technical Report UM-MEAM-99-02, 1999.
61. **Katsuhiko, Ogata.** *Dinámica de sistemas*. Mexico : Prentice-Hall, 1987. pag.536.
62. **Richard, Barnett, Larry, O'Cull y Sarah, Cox.** *Embedded C Programming and the Microchip PIC*. NY : Thomson Delmar Learning, 2004. ISBN: 1401837484. Pag. 460.