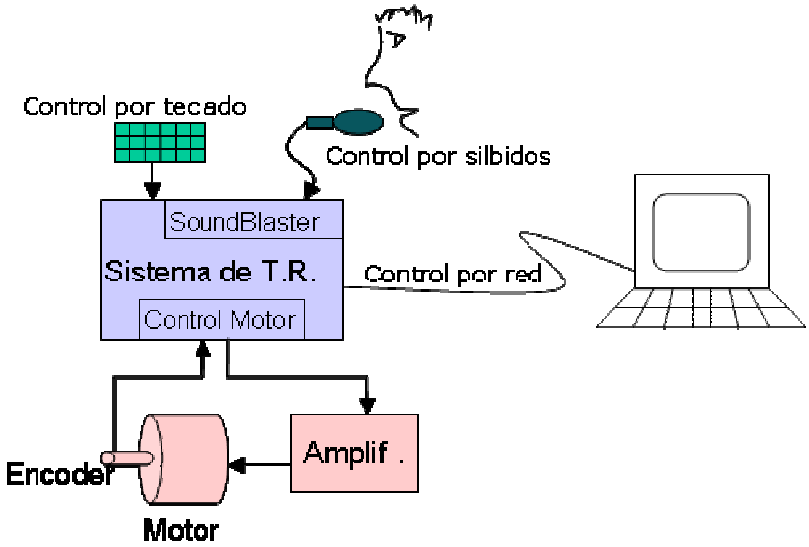
TRABAJO CURSO 2011/2012

1. Introducción

Realizar un programa de control en tiempo real de un motor DC que responda a estímulos sonoros.



El programa será realizado en modo consola, utilizando Microsoft Visual Studio 2008 bajo Sistema Operativo Windows XP/Windows 7.

2. Planteamiento básico

Se trata de desarrollar el software necesario para controlar la posición y velocidad de un motor de corriente continua.

El funcionamiento es el siguiente:

- Mediante teclado o conexión por red, el usuario indica la posición angular que desea para la salida del motor.
- A través de secuencias de silbidos, captadas mediante micrófono, el usuario puede realizar realizar las acciones de arranque (silbido continuo en 2 fases con la 2ª más aguda) y parada (id. con la 2ª fase más grave).
- Cuando se recibe una secuencia de silbidos de arranque, se comienza el control en lazo cerrado del motor DC (simulado mediante una aplicación Windows).
- Cuando se recibe una secuencia de silbidos de parada, o se ha alcanzado la posición deseada, se finaliza el control hasta recibir una nueva orden.

3. Estructura del sistema

El sistema desarrollado constará de 4 hilos:

- Hilo principal, de baja prioridad, cuya tarea será el control de la interfaz con el usuario, con al menos las siguientes opciones:
  - Salir de la aplicación.
  - Indicar la posición final deseada, mediante teclado o por medio de potenciómetro.
  - Cambiar los parámetros del algoritmo de control.
  - Arrancar o parar el motor.
  - Añadir o eliminar una dirección IP de las admitidas para control remoto (ver hilo 4).
  - Visualizar la posición actual y la objetivo.

- El segundo hilo estará encargado del procesamiento en tiempo real de datos de sonido mediante una función callback, según el algoritmo que se adjunta más adelante. Añadirá los datos de frecuencia a una tabla temporal, y procesará esta tabla para determinar si se produce una secuencia de control (frecuencia alta→baja para parada, y baja→alta para arranque).
- El tercer hilo será de tiempo real y realizará el accionamiento del motor a través de un lazo de control, en función del estado de arranque o parada.
- Por último, el cuarto hilo estará encargado de las comunicaciones por red, mediante las cuales recibirá comandos de texto que realizarán acciones similares a los recibidos por el teclado. Se deberá comprobar la identidad del llamante (dirección IP) en un archivo de texto que contenga los equipos admitidos.

3.1. Hilo principal: interfaz de usuario

La interfaz de usuario se realizará en una ventana de comandos, utilizando la librería ncurses para la realización de ventanas y el uso de colores. Se presentará el interfaz en 4 ventanas:

- Ventana de comandos: permitirá al operador introducir los siguientes comandos de texto:
  - POS=grados → cambia la posición objetivo a la deseada.
  - POS=POT → la posición objetivo es fijada por potenciómetro.
  - ARRANCAR → arranque del motor.
  - PARAR → parada del motor.
  - ADDIP=direc. IP → añadir dirección IP a archivo de admitidas.
  - DELIP=direc. IP → eliminar dirección IP de archivo de admitidas.
  - LISTIP → presentar contenido de archivo de IPs admitidas.
  - RZ=[b_0 … b_m]/[1 a_1 … a_n] → cambia el regulador del lazo de control a:

$$R(z) = \frac{b_0 + b_1.z^{-1} + ... + b_m.z^{-m}}{1 + a_1.z^{-1} + ... + a_n.z^{-n}}$$

- Ventana de motor: se mostrará la posición actual y velocidad del motor, su estado de activación (arranque/paro), y la posición objetivo fijada por el operador.
- Ventana de sonido: se mostrará la amplitud y frecuencia actual detectada por el hilo de sonido, y los eventos de activación del motor (arranque/paro) en función del procesamiento de las frecuencias.
- Ventana de comunicaciones TCP/IP: se mostrará el estado de comunicación con posibles clientes, y en caso de que haya alguno conectado, su dirección IP, puerto, y último comando recibido.

3.2. Hilo de procesamiento de sonido

En este hilo se realizará la inicialización de la tarjeta de sonido, y la captura del mismo a través de una función callback. Cada vez que se capture un conjunto de datos de sonido, se procederá a la búsqueda de la frecuencia fundamental del mismo (supuesta senoidal) mediante el siguiente algoritmo:

- Detección de amplitud, a partir de la desviación típica. Se considera que hay señal válida cuando en el conjunto de muestras (ej. 256) la desviación típica supera un umbral determinado.

$$\sigma = \sqrt{\frac{\sum (y_k - \bar{y})^2}{n-1}}$$

- En caso de superar el umbral, detección de la frecuencia fundamental de la señal senoidal, a través de la identificación de un modelo AR discreto sencillo de 2 polos:

$$y_k = a_1.y_{k-1} + a_2.y_{k-2}$$

- Puesto en forma matricial para todo *k*, el algoritmo de identificación será el siguiente, para un conjunto de *n* datos $y_0 \ldots y_{n-1}$:

$$M_a.A = M_b \qquad M_a = \begin{bmatrix} y_1 & y_0 \\ y_2 & y_1 \\ \ldots & \ldots \\ y_{n-2} & y_{n-3} \end{bmatrix} \quad M_b = \begin{bmatrix} y_2 \\ y_3 \\ \ldots \\ y_{n-1} \end{bmatrix} \quad A = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$$

- Con solución por mínimos cuadrados:

$$A = \left(M_a^t.M_a\right)^{-1}.M_a^t.M_b$$

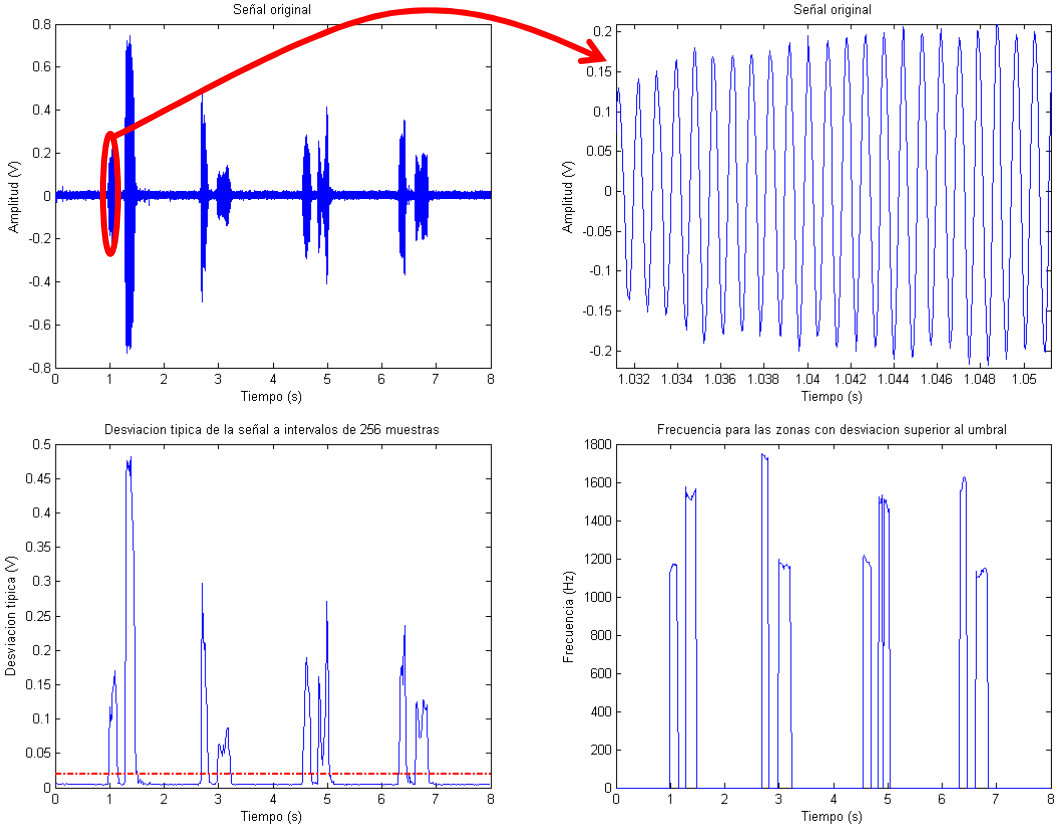
- Puesto que $M_a^t.M_a$ es de orden 2 para este problema, se resuelve sencillamente de la forma siguiente:

$$M^{-1} = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix}^{-1} = \begin{bmatrix} m_{22} & -m_{12} \\ -m_{21} & m_{11} \end{bmatrix} \Big/ \left(m_{11}.m_{22} - m_{12}.m_{21}\right)$$

- Se puede estimar la frecuencia mediante el ángulo que forman las raíces complejas de: $p(z) = z^2 - a_1.z - a_2$

$$r + c.j = \frac{a_1 \pm \sqrt{a_1^2 + 4.a_2}}{2} \qquad \theta = \arctan\left(c/r\right) \qquad f = \frac{\theta}{\pi} \cdot \frac{f_m}{2}$$

Ejemplo emitiendo 4 órdenes: marcha, paro, marcha, paro según se indicó anteriormente, y troceando la señal en tramos de 256 muestras:



- Las últimas frecuencias obtenidas se almacenarán en una tabla de valores temporales (f_k), la cual se procesará para detectar tonos (varias frecuencias consecutivas similares) y secuencias de 2 tonos (2 secuencias de frecuencias similares, cercanas entre sí, pero con medias diferentes).

3.3. Hilo de control del motor

El sistema físico a controlar será un motor DC. Se simulará el funcionamiento del motor mediante la librería adjunta SimulaMotor, que dispone de su documentación correspondiente.
El hilo de control se ejecutará periódicamente (Tm=200 ms), y en cada periodo de muestreo realizará las siguientes tareas:

- Lectura de la referencia (de comando de usuario o de potenciómetro en el simulador) en grados.
- Lectura de la posición actual (de salida correspondiente del simulador) en grados.
- Desplazamiento de tablas temporales: e_k y u_k.
- Cálculo del error actual en grados e introducción en tabla e_k.
- Cálculo de salida actual u_k (en voltios) a través de un algoritmo que implemente la ecuación en diferencias correspondiente a $R(z) = \frac{b_0 + b_1.z^{-1} + ... + b_m.z^{-m}}{1 + a_1.z^{-1} + ... + a_n.z^{-n}}$ (se pueden utilizar los siguientes valores por defecto para los parámetros de R(z): m=1, b_0=0.1265, b_1=-0.1035, n=0).
- Cálculo y generación de señal PWM para cambiar la entrada del motor a partir del valor u_k actual.

3.4. Hilo de comunicaciones TCP/IP

El hilo de comunicaciones funcionará como servidor TCP. Quedará a la espera de solicitudes de conexión TCP por parte de clientes.
Una vez aceptada una solicitud de conexión, se comprobará si la dirección IP del cliente se encuentra en la lista de IPs admitidas, almacenada en un archivo de texto.
Si el cliente no procede de una IP admitida, se cerrará la conexión y se quedará a la espera de una nueva.
Si el cliente procede de una IP admitida, el hilo servidor leerá cadenas de texto procedentes del mismo (longitud máxima 200 caracteres), y las procesará con el mismo criterio que las entradas de teclado.
Cuando el cliente se desconecte o envíe la cadena “EXIT”, se cerrará la conexión y se quedará a la espera de una nueva.
Se deberá realizar un programa simple para gestión del cliente, que solicite conexión, espere por teclado por cadenas de comando, y las envíe por el socket conectado.

3.5. Comunicaciones entre hilos

La comunicación entre los diversos hilos y funciones asíncronas podrá realizarse por variables globales compartidas, reguladas por mútex cuando sea necesario, o bien por mensajes síncronos. El acceso al archivo de llamantes admitidos por TCP también debe estar regulado por algún mecanismo de sincronización.

4. Bibliotecas externas

Se dispone en la página web del trabajo de todas las librerías necesarias para el desarrollo del trabajo, a saber:

- Biblioteca de hilos, semáforos y mútex POSIX para Windows “pthreads-win32”.
- Biblioteca de temporizadores (síncronos y asíncronos) y mensajes (sólo síncronos) estilo POSIX para Windows “posixlib”.
- Biblioteca de simulador del motor “SimulaMotor”.
- Librería “ncurses” para la programación de ventanas en consolas de texto.
- Librería de acceso a la tarjeta de sonido de Windows “winmm.lib” y librería de sockets para comunicación por red “ws2_32.lib” disponibles directamente bajo Visual Studio 2008.

5. Requisitos

El programa deberá realizarse de acuerdo con los siguientes requisitos:

- Realización de funciones para cada funcionalidad básica, agrupadas en archivos de código fuente y encabezados según su temática.
- Utilización de asignación dinámica de memoria para tablas y cadenas de caracteres cuando sea posible.
- Utilización de estructuras para el manejo de matrices.
- Elaboración de una pequeña memoria (2 a 3 páginas) explicando brevemente los contenidos del programa (hilos, archivos utilizados, funciones básicas, sincronización de hilos).
- Entrega del directorio de la solución y la memoria en archivo comprimido por e-mail a ialvarez@isa.uniovi.es en las fechas que se especifiquen para cada convocatoria.

6. Mejoras

Se proponen las siguientes mejoras sobre el funcionamiento básico para aumentar la calificación:

- Programación de un interfaz basado en Windows, que presente la evolución de la frecuencia detectada y de la posición del motor en la pantalla, y permita el control mediante opciones de menú.
- Extensión del algoritmo de detección de sílbidos a otros sonidos (ejemplo vocales), mediante el uso de FFTs.
- Búsqueda de parámetros de control optimizados para el motor (ver G(s) del motor en la documentación del simulador).
- Creación de un nuevo hilo, que genere un sonido de salida basado en una senoidal (incluir algunos armónicos para evitar la sensación de pitido agudo, ver http://www.xtec.es/centres/a8019411/caixa/ondas.htm), cuya frecuencia indica la lejanía del objetivo de posición: frecuencia equivalente a nota “SI” para posición 0, frecuencia equivalente a nota “DO” para llegada a posición final (ver frecuencias a generar para obtener cada una de las notas de la escala en http://www.musicagospel.com.ar/de_donde_notas.htm) .