Práctica 7

Práctica Final con simulador TORCS

7.1 Objetivos

El objetivo de esta práctica es realizar un programa de control donde se ilustren las distintas etapas de la construcción de una aplicación de tiempo real, aunque no se exigirá el análisis de los tiempos de respuesta para no complicar la práctica en exceso.

7.2 Introducción

TORCS (The Open Racing Car Simulator) es un simulador de carreras de coches de código abierto en 3D disponible para GNU/Linux, FreeBSD, Mac OS X y Microsoft Windows. TORCS fue creado por Eric Espie y Guionneau Christophe, pero el desarrollo del proyecto está encabezado por Bernhard Wymann. Está escrito en C++ y está disponible bajo la GNU GPL. TORCS está diseñado para permitir IA pre-programadas, pilotos en uno contra el otro, al tiempo que permite al usuario controlar un vehículo usando un teclado, un ratón, o volante.

Fig. 7.1: El simulador de carreras Torcs

En la práctica se ha modificado uno de los coches (Roboto) para enviar y recibir información a través de un socket. El objetivo es conducir el coche en un circuito, lo más rápidamente posible utilizando la información que se envía a través del socket y enviando los comandos de conducción a través del mismo socket. En los ordenadores del laboratorio se ha instalado TORCS y el Roboto para que se puedan probar los controladores que se implementarán en Ada. El alumno únicamente tendrá que programar los algoritmos de control. La parte de lectura/escritura del socket en Ada se entregará a los alumnos.

Para conducir el coche se podrá controlar el volante y los pedales de acelerador y freno.

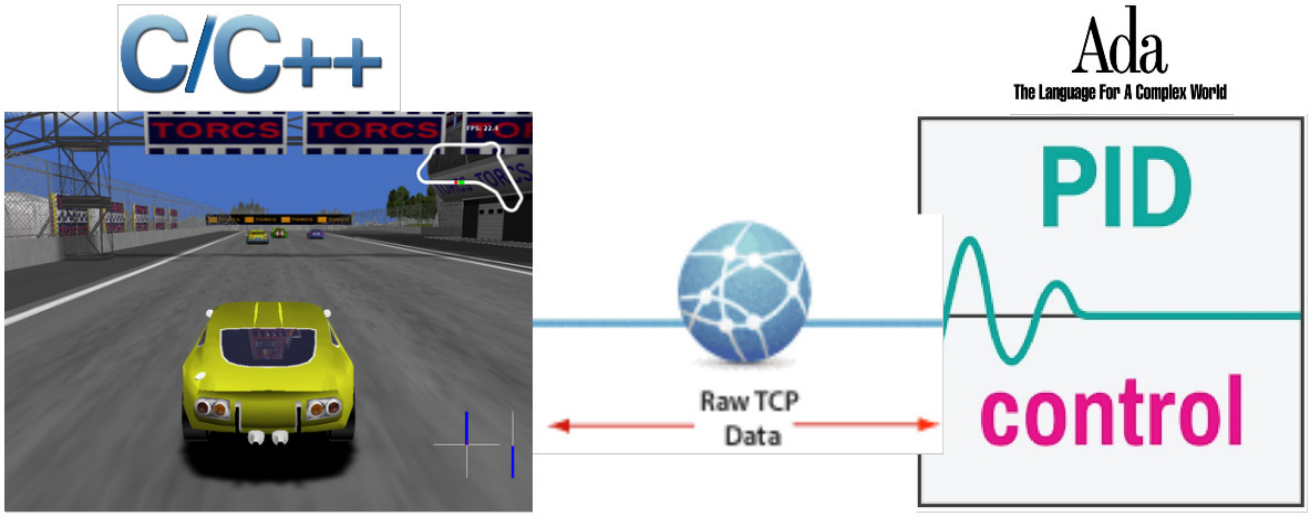

Fig. 7.2: Esquema del funcionamiento de la práctica

7.3 Información enviada y recibida por el socket desde TORCS

Para facilitar la tarea de control del vehículo se han diseñado distintos modos de funcionamiento del Roboto en TORCS. El Roboto se puede configurar para que funcione de forma completamente autónoma o controlado remotamente por comandos. Para configurar cómo va a funcionar el Roboto hay que enviar un comando al comienzo del programa como se indica a continuación.

7.3.1 Fase de configuración

Como se comentó anteriormente, el Roboto tiene 2 parámetros a controlar (la posición lateral y la velocidad) y 3 modos de funcionamiento para cada uno de ellos. Estos modos dependen de un comando que se recibe al arranque por el socket. Este comando está formado por un número de dos cifras, la primera indica el modo de control lateral que se desea, el segundo el modo de control longitudinal.

Comando	11	12	13	21	22	23	31	32	33
lateral	auto	auto	auto	radio curvatura	radio curvatura	radio curvatura	posición volante	posición volante	posición volante
longitudinal	auto	velocidad m/s	freno acelerador	auto	velocidad m/s	freno acelerador	auto	velocidad m/s	freno acelerador

Este comando se envía una sola vez al comienzo de la carrera.

- Modo lateral
  1. Si se envía un 1 el control lateral es automáticamente realizado por el coche.
  2. Si se envía un 2 el robot recibirá por el socket el radio de curvatura que se quiere que describa el coche.
  3. Si se envía un 3 el robot recibirá por el socket la posición del volante entre [-1 1] donde 1 es todo a la izquierda y -1 todo a la derecha.
- Modo longitudinal (velocidad)
  1. Si se envía un 1 el control longitudinal es automáticamente realizado por el coche.
  2. Si se envía un 2 el robot recibirá por el socket la velocidad en m/s que se quiere que adquiera el coche.

3. Si se envía un 3 el robot recibirá por el socket la posición del acelerador y el freno entre [-1 1] donde -1 es freno a fondo y 1 acelerador a fondo.

Como ejemplo, si tras conectar con el socket mandamos un 12 quiere decir que el volante funcionará automáticamente y la velocidad se controlará enviándole la velocidad en m/s por el socket.

7.3.2 Fase de conducción

Una vez enviado el modo de funcionamiento se le pueden mandar comandos continuamente al robot. Los comandos de control que se envían al robot por el socket están compuestos siempre de 2 números float:

1. **Comandos de control lateral:** en función del modo en el que hayamos configurado el robot puede ignorar este comando (cuando esté en auto) o considerarlo como el radio de curvatura o la posición del volante.

2. **Comandos de control longitudinal:** en función del modo en el que hayamos configurado el robot puede ignorar este comando o considerarlo como la velocidad en m/s o la posición los pedales.

Por ejemplo si enviamos "-0.85;30.0" puede significar pon el volante al 85 % hacia la derecha y la velocidad a 30 m/s si configuramos el modo anteriormente como 32.

El TORCS enviará y recibirá la información cada 20ms. Si no se le envía nada seguirá ejecutando la última orden recibida.

7.3.3 Fase de lectura de datos

TORCS es un simulador muy completo en el que se dispone de gran cantidad de información tanto del circuito como del coche. Para la realización de la práctica se ha elegido transmitir los siguientes 9 datos:

1. **Error angular:** Diferencia en radianes entre la orientación del coche y el eje de la carretera. Positivo → Derecha. Negativo → Izquierda.

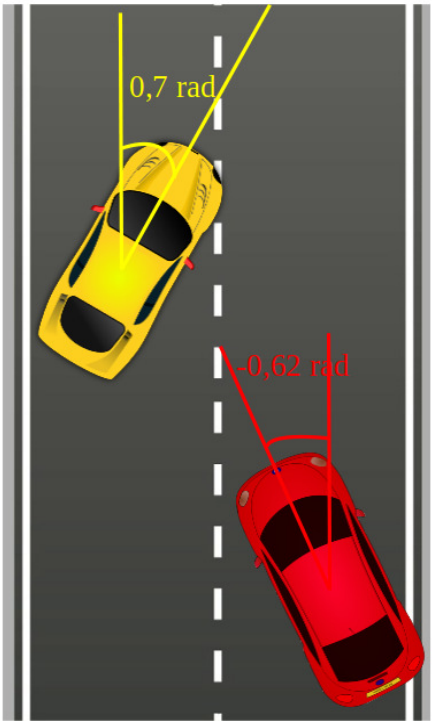

Fig. 7.3: Error angular

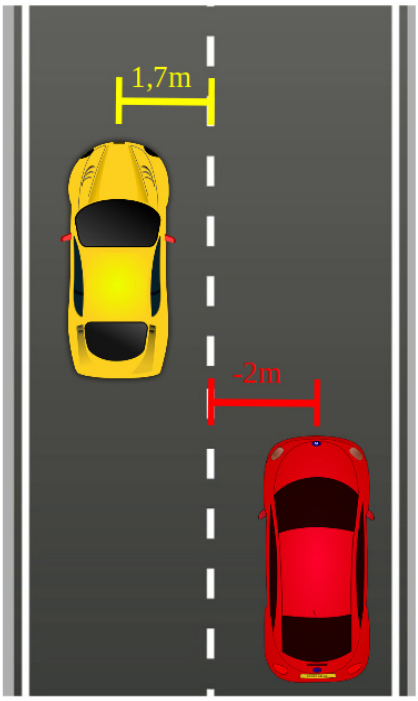

Fig. 7.4: Error lateral

2. **Error Lateral:** Distancia en metros del centro del coche al centro de la carretera. Positivo → Izquierda. Negativo → Derecha.
3. **Tipo de segmento:** 1 → curva derecha, 2 → curva izquierda, 3 → recto.
4. **Radio del segmento actual:** es el radio en metros de la curva
5. **Distancia al siguiente segmento:** distancia en metros al siguiente segmento que sea de distinto tipo. Puede ser o distancia a la siguiente curva o distancia a la siguiente recta según el segmento en el que nos encontremos.
6. **Tipo de segmento siguiente:** Igual que la distancia al siguiente segmento, pero con el tipo.
7. **Curvatura del segmento siguiente:** Igual que la distancia al siguiente segmento, pero con la curvatura.
8. **Longitud del siguiente segmento:** Igual que la distancia al siguiente segmento, pero con lalongitud.
9. **Velocidad:** velocidad del coche en m/s.

7.4 Utilización de TORCS

Todos los ordenadores del laboratorio tienen instalado el TORCS con un coche modificado para que reciba los comandos a través de un socket, como se ha explicado en el punto 2. Para utilizar TORCS únicamente hay que teclear en el terminal torcs y el programa arrancará. Se seleccionará "quick race" y en "Configure race" se elegirán los participantes (dejar únicamente el Robot0) y circuito (se recomienda usar myOval y wideOval). Cuando le demos a iniciar la carrera el coche quedará esperando la conexión por el socket para recibir la configuración y comenzar la carrera. Hasta que no se conecte el socket no comenzará la carrera.

7.5 Controlador en Ada

Hasta que no se conecte el socket no comenzará la carrera. Se proporciona el programa en Ada que se llama vcv.adb. Este fichero deberá ser modificado por el alumno para lograr el control del coche. Cuando el TORCS esté esperando para iniciar la carrera se deberá lanzar el programa con el comando ./vcv

7.6 Instalación de TORCS

Sigue estos pasos para instalar TORCS:

1. Abre un terminal y extrae el contenido de los archivos con:

   ```
   tar -xvzf torcs-1.3.7_modSTR.tar.gz
   ```

2. Haz que el script 'configure' sea ejecutable con:

   ```
   sudo chmod a+x configure
   ```

3. Ejecuta el script 'configure' con:

   ```
   ./configure
   ```

4. Compila TORCS con el siguiente comando (puede tardar un poco):

   ```
   make
   ```

5. Instala el programa básico de TORCS con:

   ```
   sudo make install
   ```

6. Instala los vehículos (incluyendo nuestro Robot0) y pistas (como myOval y wideOval) con:

   ```
   sudo make datainstall
   ```

7. Ejecuta TORCS con:

   ```
   torcs
   ```

7.7 Desinstalación

Para eliminar TORCS de tu sistema, ejecuta los siguientes comandos:

1. Eliminar el ejecutable de TORCS:

   ```
   sudo rm -rf /usr/local/bin/torcs
   ```

2. Eliminar las bibliotecas de TORCS:

```
sudo rm -rf /usr/local/lib/torcs
```

3. Eliminar los archivos de juego de TORCS:

```
sudo rm -rf /usr/local/share/games/torcs
```