Práctica 3: Introducción al iRobot Framework
Lectura y calibración de los sensores del iRobot

Robótica, Sensores y Actuadores

2018-2019

Objetivos y materiales

OBJETIVOS

- Aprender a usar el ***iRobot Framework***
- Programar la lectura de los sensores de iRobot Create y su calibración en lenguaje C/C++.

MATERIAL NECESARIO

- **Hardware**
  - iRobot Create
  - Raspberry Pi2
- **Software**
  - NetBeans
  - iRobot Framework
- **Manuales (en eGela)**
  - Guía del usuario: iRobot Create Owner's Guide
  - Manual del lenguaje de comandos: iRobot Create Open Interface
  - Lista de funciones iRobotConnection

Instalación y uso del iRobot_Framework

Para instalar y verificar el iRobot_Framework seguir el procedimiento detallado en

Instalacion iRobot_Framework.pdf

disponible en eGela.

Puesta en marcha

- Instalar el compilador de Raspberry Pi2, versión 4.8
  > sudo apt-get install gcc-4.8
  > sudo apt-get install g++-4.8

- Borrar los enlaces simbólicos del sistema antiguos
  > sudo rm /usr/bin/gcc
  > sudo rm /usr/bin/g++

- Actualizar los enlaces simbólicos del sistema: /usr/bin/gcc y /usr/bin/g++
  > sudo ln –s /usr/bin/gcc-4.8 /usr/bin/gcc
  > sudo ln –s /usr/bin/g++-4.8 /usr/bin/g++

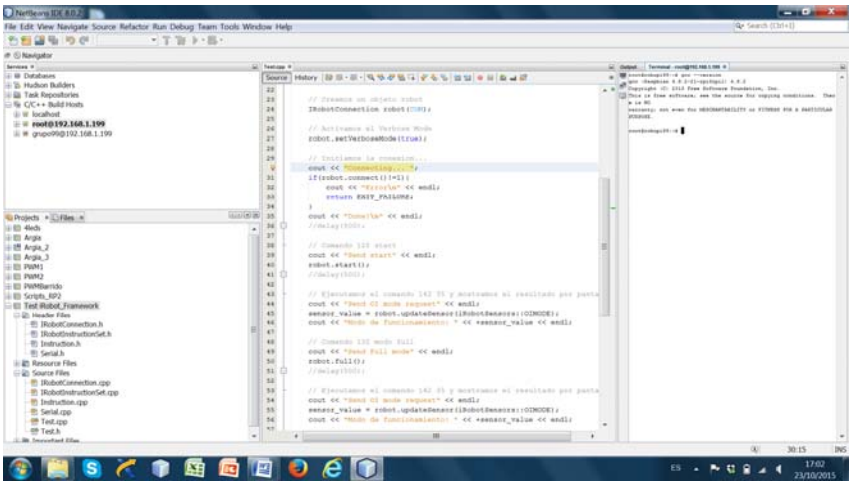
Proceso de instalación

- Abrir NetBeans
- Importar el proyecto frameworktest.zip **(sin descomprimir)**
  - seleccionar File—> Import Project —> From Zip
- frameworktest.zip está en eGela: Práctica 3
  - Contiene los ficheros:

- IRobotConnection.h
- IrobotInstructionSet.h
- Instruction.h
- Serial.h
- Test.h

- IRobotConnection.cpp
- IrobotInstructionSet.cpp
- Instruction.cpp
- Serial.cpp
- Test.cpp

Compilar y ejecutar Test.cpp

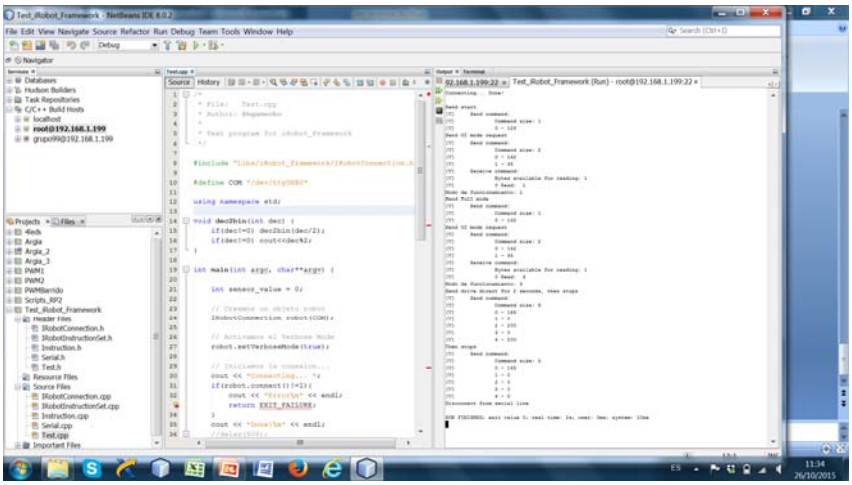


Verificaciones

- Si se trabaja con la cuenta root
  - Verificar que está compilando para vuestra dirección IP (root@192.168.1.1XX)
- Si se trabaja con la cuenta GrupoXX
  - Verificar que está compilando para vuestra dirección IP (grupoXX@192.168.1.1XX)
  - Para dar al usuario GrupoXY permiso de ejecución en el adaptador línea-serie conectado al iRobot Create, ejecutar en el terminal remoto de Raspberry Pi2 :

    > sudo usermod -aG dialout grupoXY

La ejecución producirá



- A partir de este momento ya se pueden escribir programjas en C/C++ usando las instrucciones de iRobot proporcionadas por iRobot_Framework

Programación de sensores del iRobot Create mediante el iRobotFramework

- Para probar y calibrar los sensores del iRobot Create disponemos de la instrucción

  Int IRobotConnection::updateSensor (char sensorId)".

- Los códigos sensorId se encuentran en el fichero iRobotInstructionSet.h dentro del namespace iRobotSensors.

Ejemplo de programación mediante el iRobotFramework

- Si queremos consultar el modo de funcionamiento en el que se encuentra iCreate, usaremos la siguientes instrucciónes:

  int modo = robot.updateSensor(iRobotSensors::OIMODE);

  cout << modo << endl;
- Este código nos devolverá los siguientes valores:
  - 1 si estamos en el modo Passive
  - 2 si estamos en el modo Safe
  - 3 si estamos en el modo Full
- Tenemos que tener en cuenta que el método updateSensor siempre devuelve un int (entero con signo de 16 bits)
- donde se guarda el valor que recibe del robot iCreate (a través de Open Interface), dónde puede tener otro formato.
- Por ejemplo, si se leen los sensores *Bumps & WheelDrops* hay que interpretar bit a bit, mediante una máscara de bits, el entero de 16 bits que devuelve el framework.

Codificaciones Sensores iCreate

- Desde updateSensor(char code); siempre obtenemos un Int (con signo), en cada caso habrá que hacer un casting al tipo de datos adecuado para poder tratar la información de los sensores correctamente.
- Ejemplo:

  Char value = updateSensor(iRobotSensors::BUMPERS_AND_WHEELDROPS);

  printf("Front Caster WheelDrop: %d \n",value & 0x10);
- Hay más información sobre los sensores disponibles en las páginas 24 y 25 del manual Create_Open_Interface_v2 (disponible en eGela)

Sensor	Codificación
Bumps & WheelDrops	Máscara de bits
Wall, Cliffs x 4, Virtual Wall	1 bit value (El menos significativo)
Infrared	1 Byte [0,255]
Distance, Angle, Requested Radious	Signed 16 bit value [-32768 - 32768]
Wall Signal, Cliffs x 4,	**Unsigned 16 bit [0 – 4095]**
Requested Velocity x3	Signed 16 bit value [-500, 500]

Cómo consultar sensores

Int value = updateSensor(char code);

Ejemplo:

```
Char value = updateSensor(iRobotSensors::BUMPERS_AND_WHEELDROPS);
printf("BUMPERS AND WHEELDROPS %s \n",value);
```

Codes:

- iRobotSensors::CLIFFLEFT
- iRobotSensors::CLIFFFRONTLEFT
- iRobotSensors::CLIFFFRONTRIGHT
- iRobotSensors::CLIFFRIGHT
- iRobotSensors::BUMPERS_AND_WHEELDROPS
- iRobotSensors::WALL
- iRobotSensors::DISTANCE
- iRobotSensors::ANGLE
- …

Práctica: Calibración de sensores

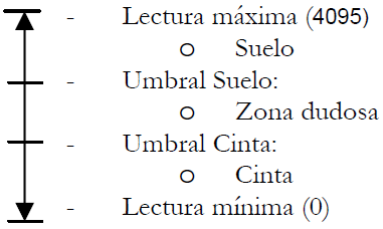
3.1 Verificación de sensores on/off

- Diseñar un programa que lea los valores de cada uno de los sensores siguientes en la situaciones de activado y desactivado.
- Para cada sensor dar una señal acústica cuando cambia de estado. Además, debe encender un LED cuando está desactivado y otro LED distinto cuando está activado.
- Este programa debe verificar el funcionamiento de los sensores:
  - Bumps (izquierdo y derecho)
  - Wheel drops (izquierdo, derecho y caster)
  - Cliff (derecho, frontal derecho, frontal izquierdo e izquierdo)
  - Botones Play y Advance
  - Virtual wall (sólo grupos 1 y 5)
  - Low Side Drivers y Wheel Overcurrents (grupos sólo 2 y 6)
  - Home Base e Internal Charger (sólo grupos 3 y 7)
  - Wall (sólo grupos 4 y 8)

Práctica: Calibración de sensores

3.1 Calibración de sensores de barranco (Cliff Signal):

- Diseñar un programa capaz de recoger 5 muestras de cada sensor de barranco sobre la cinta aislante negra y sobre el suelo del laboratorio.
  - Para cada sensor calcular los valores medios de lectura de cinta y suelo.
  - Definir un umbral de suelo y un umbral de cinta que permitan diferenciar el suelo y la cinta aislante.



Práctica: Calibración de sensores

3.3 Calibración de sensores de distancia:

- Diseñar un programa que haga avanzar el robot un recorrido conocido en línea recta usando el comando DRIVE DIRECT
  - Ejecutarlo 4 veces (al menos) y medir las distancias recorridas realmente.
  - Haced una tabla con los valores medidos, los errores absolutos y relativos y calcular en error absoluto medio y el error relativo medio.
- Diseñar un programa que haga avanzar el robot un recorrido conocido en línea recta usando el comando DRIVE y espera por distancia
  - Ejecutarlo 4 veces (al menos) y medir las distancias recorridas realmente.
  - Haced una tabla con los valores medidos, los errores absolutos y relativos y calcular en error absoluto medio y el error relativo medio.
- Comparar los resultados obtenidos con ambos programas
  - ¿Qué ventajas e inconvenientes presenta cada uno de ellos?
  - ¿Cuál es más conveniente?

Práctica: Calibración de sensores

3.5 Calibración del sensor de distancia a la pared (Wall Signal):

- Recoger 10 muestras del sensor Wall Signal a distintas distancias de la pared.
  - Hacer una tabla con los resultados.
  - Dibujar la recta de regresión y obtener su ecuación mediante el método de mínimos cuadrados.

Informe

- Comprimir todos los ficheros en un único zip con el siguiente nombre:
- Practica3_GrupoXY_Apellido1_Apellido2
  - Resultados, tablas
  - Respuestas a las preguntas
  - Códigos fuente
  - comentarios
- En todos los programas se valorará positivamente la cantidad y calidad de la información que devuelva el programa sobre los valores de los sensores que va leyendo.