UNIVERSITY OF ZAGREB

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

DEPARTMENT OF

ELECTRONIC SYSTEMS AND INFORMATION PROCESSING

LABORATORY FOR SYSTEMS AND SIGNALS

A student project within the course

"SOFTWARE DESIGN FOR MEASUREMENT AND PROCESS SYSTEMS"

Telecontrol of a Mindstorms® NXT Robot

Technical documentation

Kristina Bashota
Tibor Čordaš
Dalibor Jelača
Iva Jelenčić
Vedran Koruga
Damir Kušević

Predrag Pale
mentor

Zagreb, February 6, 2008.

Contents

Introduction 3

Description of the comunication and robot functioning 4

Enabling the web interface 5

Appendix 6

Introduction

For the purpose of this project a *Mindstorms® NXT* robot is used.
Accessories that were used for the robot are: an ultrasonic sensor, touch sensor, sound sensor and opening/closing the robot's claws.

Managing the robot is programmed in the C++ language. A free *Bloodshed Dev-C++ compiler* was used for that purpose and was downloaded from the following website http://www.bloodshed.net/devcpp.html. The programming was simplified by downloading already written libraries from the website http://www.norgesgade14.dk/bluetoothlibrary.php.

It is quite easy to communicate with the robot via a Bluetooth connection. After establishing a connection it's necessary to find out which COM port is used by the robot.

Instructions are sent to the robot directly through the C++ files. There are six classes which enable:

- Establishing and disconnecting a Bluetooth connection,
- Control over the motors,
- Reading the data from the sensors,
- Naming the *Brick* and reading the firmware,
- Sound reproduction
- Using the ultrasnonic sensors.

The web interface is designed to enable maneuvering the robot. The user can choose actions for the robot (moving forward / backward, turning left / right, opening / closing claws). Data received from the sensors are clearly displayed on the interface so the user can decide on his next move. Managing the robot from a remote computer through the web interface is enabled by the server computer.

Description of the comunication and robot functioning

Bluetooth connection

The COM port for Bluetooth comunication is opened with the function ***int connect(char *port)***, is closed with the function ***disconnect()***.

Managing the motors

The robot has three motors which are managed by the following functions. ***Motor(int output_port, Serial *cp)*** transfers the port number and the pointer to the serial connection. The function ***int on(int speed)*** starts up a motor with the set velocity. If the negative algebraic sign is added, the motor will rotate in the counter direction. The ***int stop()*** function stops the motor, whilst the ***int reset_rotation()*** resets the motor rotation.

The movement velocity of the robot forward and backward is set at a fixed value of 35 (the velocity can be chosen arbitrarily and the velocity range is from 0 to 100). The velocity of turning left and right is reduced to a quarter of the fixated value for easier steering. The robot's movement instructions given from the web interface are written in a textual file named *upravljanje.txt* which is created by the program itself.

Managing the sensors

Data received from the ultrasonic sensor and the touch sensor has to be forwarded to the computer which will be displayed on the web interface.

The data given from the ultrasonic sensor is read in an infinite *while* loop. By calling the ***sonar.distance()*** function the sensor is asked to measure the distance from an obstacle. That information is sent to the computer via a Bluetooth connection for futher processing. (The given function can return the distance either in cm or inches.) The information is then written in the textual file *udaljenost.txt* which is created by the program and is a link to the web interface.

The condition of the touch sensor is also cheked in the above mentioned infinite loop. The touch sensor is used for detecting low obstacles lying on the ground such as cables, books, pencils etc. By calling the ***touch.read()*** function the sensor is asked to detect obstacles. If the sensor comes across an obstacle, the robot sends a message via a Bluetooth connection to the computer that the sensor has been pressed. When the sensor is released, a message is also sent to the computer. The condition of the sensor is also written in the textual file *tipkalo.txt* which is a link to the web interface.

Managing other accessories

The instruction ***speaker.beep()*** enables playing a sound from the sound sensor.
Opening and closing the robot's claws is enabled by using the ***motorA.reset_rotation()*** and ***motorA.on(10,45)*** (10 is the opening and closing velocity, 45 is the angle) functions.

Enabling the web interface

After designing a web interface in a html editor, it's important to follow the following steps for making the comunication with the robot possible:

- Download and install *apache* web server (version 2.2.8, although the version isn't that important) from the following website

http://ftp.carnet.hr/misc/apache/httpd/binaries/win32/apache_2.2.8-win32-x86-no_ssl.msi

The installation is simple, just choose next.

- Download and install the *php* from the website

http://www.php.net/get/php-5.2.5-win32-installer.msi/from/a/mirror

The installation is simple, just choose next. At one moment it will ask for which program to install the module. Mark the *apache 2.2.x*. Also, it will ask for the path of the configuration file of the *apache*. It's located in the *conf* directory which is in the directory where the *apache* was installed.

- Copy the directory with the web page within the *htdocs* directory which is located in the *apache* directory. Copy the main program for managing the robot to the *mstore/tmp* directory and start the main program.

In this example the *html* page is named *index.html* and is saved in the *mstore* directory with all the files (textual files and the jpeg pictures of icons used for the interface) needed for the interface to work properly. In order to access the website link to *localhost/mstore* from the local computer or to *IP-address/mstore* from a remote computer.

Appendix

The commented code appended to the documentation is as follows.

```
#include <cstdlib>
#include <iostream>
#include <string>
#include <conio.h>
#include <stdlib.h>


#include "serial.hpp"
#include "brick.hpp"
#include "sound.hpp"
#include "motor.hpp"
#include "sonar.hpp"
#include "sensor.hpp"

Serial bluetooth;
Sonar sonar = Sonar(0,&bluetooth);   //ultrasonic sensor on port 1
Sensor touch = Sensor(1,&bluetooth); //touch sensor on port 2
Brick nxt = (&bluetooth);
Sound speaker = (&bluetooth);

Motor motorA = Motor(0,&bluetooth);
Motor motorB = Motor(1,&bluetooth);
Motor motorC = Motor(2,&bluetooth);//Motor C on port 3
Sound zvuk;

using namespace std;
int main(int argc, char *argv[]){
   FILE *fp, *fp1, *fp2;

   int brz; //velocity
   int naredba; //managing the robot
   int j,k,l; //flags for the files
   int i,x,y; //flags for the instructions

   int distance;
   int tipkalo; //variable for touch sensor
   char string [4];

   if(bluetooth.connect("COM6")){//connects to the NXT through COM port 6
      cout << "Veza OK!" << endl;
      nxt.set_name("LEGO"); //Set the brickname to LEGO
      cout << "Firmware verzija: " << nxt.firmware_version() << endl;

      touch.type_and_mode(TOUCH,BOOL_MODE); //initialize the touch sensor
```

```
 sonar.setup(); //initialize the ultrasonic sensor
 //intialize the flags and variables
 brz=35; // velocity
 naredba=13;
 i=12;
 j=0;
 k=0;
 l=0;
 x=1;
 y=1;
 distance=0;
 tipkalo=0;

// cout << "Postavite zeljenu brzinu!" << endl;
// cin >> brz;

 while (naredba != 1){
     Sleep(10);
     //load instruction through file "upravljanje.txt"
     if ((fp = fopen("upravljanje.txt", "r+")) !=NULL){  //create file
        j=1;
     }

     if(j){
       naredba=fgetc(fp);
       fclose(fp);  //close file
       fopen("upravljanje.txt", "w");

       fclose(fp);  //close file
       j=0;
     }

     //write distance to file "udaljenost.txt"
     if ((fp1 = fopen("udaljenost.txt", "w"))!=NULL){//create file
        k=1;

     }

     if(k){
        itoa(distance,string,10);
        fputs(distance,string,fp1); //write information about the distance to the file
        fclose(fp1);  //close file
        k=0;
     }

     //writing the condition of the touch sensor to the file "tipkalo.txt"
     if ((fp1 = fopen("tipkalo.txt", "w")) !=NULL){  //kreiranje datoteke
        l=1;
     }
```

```
        if(l){
           tipkalo=touch.read();

           fputc(tipkalo,fp1); //write condition of the touch sensor to file
           fclose(fp1);  //close file
           l=0;
        }

        // controlling movement and sensors

        distance=sonar.distance(); //measuring the distance

//          if (distance < 35){
//             motorB.stop();
//             motorC.stop();
//          }
        naredba=naredba-48;
        if (naredba == 8 && naredba!=i){ //both motors forward
              motorC.on(brz);
              motorB.on(brz);
              i=naredba;
        }

        if (naredba == 5){ // both motors  stopped
              motorC.stop();
              motorB.stop();
              i=naredba;
        }

        if (naredba == 2){ // both motors backward
                motorC.on(-brz);
                motorB.on(-brz);
                i=naredba;

        }

        if (naredba == 4){ //move left
                motorC.on(-brz/4);
                motorB.on(brz/4);
                i=naredba;
        }

        if (naredba == 6){ //move right
                motorC.on(brz/4);
                motorB.on(-brz/4);
                i=naredba;
        }

        if (naredba == 0){ //sound
                speaker.beep(200);
```

```
                i=naredba;

            }

            if (naredba == 7){ //open claws
                    motorA.reset_rotation();
                    motorA.on(10,45);
                    x=0;
                    y=1;
            }

            if (naredba == 9){ //close claws
                    motorA.reset_rotation();
                    motorA.on(-10,45);
                    y=0;
                    x=1;
            }

            if (naredba == 17){ //moving right
                    motorC.on(brz/2,30);
                    motorB.on(-brz/2,30);

            }
            if (naredba == 18 ){ //moving left
                    motorC.on(-brz/2,30);
                    motorB.on(brz/2,30);

            }
            if (naredba == 19 ){ //moving forward
                    motorC.on(brz/2,135);
                    motorB.on(brz/2,135);

            }


        }

        bluetooth.disconnect();//disconnect Bluetooth
    }
    else{
        cout << "Bluetooth veza nije uspostavljena!" << endl;
    }
    Sleep(200);
    exit(1);
}
```