Advances in Robotics & Automation
ISSN: 2168-9695

Manolov, Adv Robot Autom 2015, 4:1
DOI: 10.4172/2168-9695.1000128

Research Article Open Access

Remote Control of Educational Mobile Mini-Robot via Wireless Communication

Manolov OB*

Department of Applied Informatics and Computer Technologies, European Polytechnic University, Pernik

Abstract

This work aims to describe another contemporary manner for interaction between human and mechatronic device by Bluetooth communication with a purpose for implementing wireless remote motion control of an educational mobile robot.

Keywords: Wireless communication; Remote motion control; Educational mobile robot; Bluetooth

Introduction

In recent years, there has been a growing interest in mobile robot motion control. Usually to control the movement of a mobile robot we need to control the speed of rotation of his engines or the rotation direction. This could be done with one of the Wi-Fi or Bluetooth interfaces. The mobile robot has to have the engines connected to a control circuit with one of these interfaces. While controlling the mobile robot with Bluetooth we also have to consider that our application will not only control the robot but also will do other tasks for gathering information about the environment, computing the mobile robot's moving direction and therefore the Bluetooth connection should not block these tasks. Herein the three major topics will be discussed:

1. The educational mobile robot "Audrino".
2. The "Wii Remote Plus" as a console for wireless motion control.
3. The communication between robot and console.

Conceptual Configuration of a Distance Control

The educational mobile platform "Audrino"

Arduino robot is a self-contained platform that allows to be developed an interactive machine to explore an environment. It is the result of the collective effort from an international team in collaboration with Complubot, 4-time world champion in robocup junior robotics soccer, looking at how science can be made fun to learn [1].

For our research and experiments an own mobile robot has assembled, shown on Figure 1. It is based on classical two wheeled platform with reversible DC-servo driven wheels and spherical fulcrum on the rear side. The Motor control module drives the motors, and the Control Board reads sensors and decides how to operate.

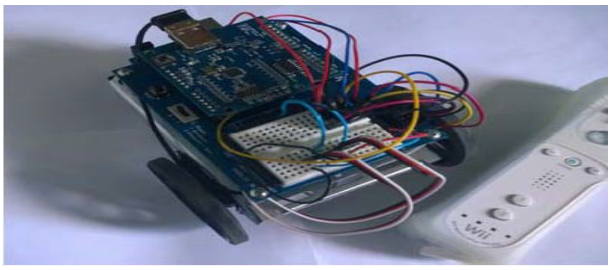

Figure 1: The assembled own "Audrino" type mobile robot.

Each of the boards is a full programmable using the Arduino IDE. The robot's completeness includes the following modules: Arduino Uno Rev3, Parallax Robotics Shield Kit, USB Host Shield 2.0 and Bluetooth communication, which modules are described in detail as follows below. The Arduino Uno Rev3 module, represented on Figure 2, is an open source microcontroller board, based on the Atmel ATmega328 MCU, plus a free software development environment [1]. The module can be used to sense inputs from switches, sensors, and computers, and then to control motors, lights, and other physical outputs.

It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Arduino Uno Rev3 differs from other preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. The microcontroller board can be powered via the USB connection or with an external power supply. The power source is selected automatically. The technical specifications of the module Arduino Uno Rev3 are:

- Microcontroller: ATmega328

Figure 2: The module Arduino Uno Rev3 (front and back).

***Corresponding author:** Manolov OB, Department of Applied Informatics and Computer Technologies, European Polytechnic University, Pernik , Tel: +359 (0) 76 600 773; E-mail: ognyan.manolov@epu.bg

Received January 29, 2015; **Accepted** February 18, 2015; **Published** February 28, 2015

Citation: Manolov OB (2015) Remote Control of Educational Mobile Mini-Robot via Wireless Communication. Adv Robot Autom 4: 128. doi:10.4172/2168-9695.1000128

- Operating Voltage: 5V
- Input Voltage (recommended): 7-12V
- Input Voltage (limits): 6-20V
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- Analog Input Pins: 6
- DC Current per I/O Pin: 40 mA
- DC Current for 3.3V Pin: 50 mA
- Flash Memory: 32 KB (ATmega328) of which 0.5 KB used by boot-loader
- SRAM: 2 KB (ATmega328)
- EEPROM: 1 KB (ATmega328)
- Clock Speed: 16 MHz

The Arduino free programming software is designed for communication with a computer, another Arduino, or other microcontrollers by UART TTL (5V) serial communication and includes a serial monitor which allows simple textual data to be sent to and from the Arduino board, as it is shown on Figure 3. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer.

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the boot-loader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

The Parallax Robotics Shield Kit module includes a Board of Education Shield (BOE), represented on Figure 4, which makes it easy to build circuits and connect servo motors to the Arduino Uno Rev3 module [2]. The BOE Shield mounts on metal chassis with motors and wheels.

With this kit and Arduino module is able to activate over 40 hands-on activities in robotics, such as:

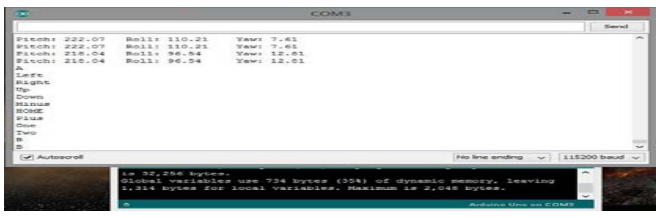

Figure 3: A textual data sent to and from the Arduino board.

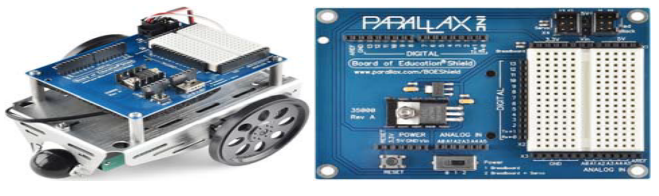

Figure 4: The Parallax Robotics Shield Kit and BOE.

Figure 5: "Wii Remote" one-handed console.

- Learning to program the robot's Arduino Brain
- Calibrating the robot's continuous rotation servo motors
- Using lights and speakers for status indicators
- Assembling the robot
- Preprogrammed navigation
- Using touch-switches to navigate by contact with objects
- Using phototransistors to navigate by light
- Using non-contact infrared sensors to measure distance and avoid or follow objects
- Remote motion control by wireless communication.

The "Wii Remote Plus" as a console for wireless motion control

Motion controllers are used to achieve some desired benefits which can include:

- increased the accuracy of position and speed;
- higher speeds;
- faster time of reaction;
- increased productivity;
- smoother movements;
- integration with other automation;
- integration with other processes;

The "Wii Remote" handle is able to communicate wirelessly with the controller via short-range Bluetooth radio, which permits to operate as far as 10 meters away from the console with up to four controllers. However, to utilize the pointer functionality, the "Wii Remote" must be used within 5 meters [3].

More over, the controller's symmetrical design allows it to be used in either hand and also to use two "Wii Remote" handles in each hand.

The "Wii Remote" handle, shown on Figure 5, represents a one-handed, remote-control based design console, instead of the traditional gamepad controllers.

The handle has the ability to sense acceleration along three axes through the use of an ADXL330 accelerometer (Figure 6). It also features a PixArt optical sensor, allowing it to determine where the console is pointing.

The built into the console BCM2042 microcontroller, shown on Figure 7, includes a large 108 Kb on-chip ROM section for storing firmware. The "Wii Remote" contains a 16 KB EEPROM chip from which a section of 6 kilobytes can be freely read and written by the host.

The "Wii Remote" is a wireless input device, using standard Bluetooth technology and HID protocol to communicate with the host,

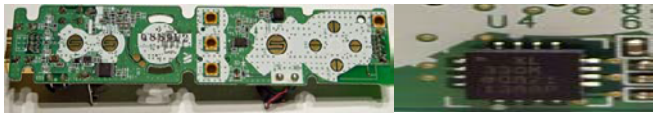

Figure 6: The ADXL330 accelerometer in "Wii Remote".

Figure 7: Wii Remote" with the BCM2042 microcontroller.

Figure 8: "Wii Remote" with the expansion device "Wii MotionPlus".

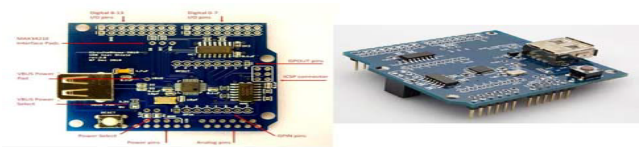

Figure 9: The USB Host Shield 2.0 module.

which is directly based upon the USB HID standard.

It is built around a Broadcom BCM2042 Bluetooth System-on-a-chip, and contains multiple peripherals that provide data to it, as well as an expansion port for external add-ons. If the EEPROM chip really contains code for the BCM2042 then this was probably done to make firmware updates possible, so there might be a way of accessing the other parts of the EEPROM via Bluetooth as well.

The "Wii MotionPlus" is an expansion device for the Wii Remote (Figure 8) that allows it to capture complex motion more accurately, as a remote design is fitted perfectly for pointing, and in part to help the console appeal to a broader audience that includes non-gamers.

It incorporates a dual-axis tuning fork gyroscope, and a single-axis gyroscope which can determine rotational motion [4]. The information captured by the angular rate sensor can then be used to distinguish true linear motion from the accelerometer readings.

This allows for the capture of more complex movements than possible with the "Wii Remote" alone. More over, it gives the ability the "Wii Remote" also to be turned horizontally and used like a steering wheel.

The Bluetooth communication between robot and console

The USB Host Shield 2.0 module, shown on Figure 9, is universal connection tool and currently supports the following device classes [5]

- HID devices, such as keyboards, mice, joysticks, etc.
- Game controllers - Sony PS3, Nintendo Wii, Xbox360
- USB to serial converters - FTDI, PL-2303, ACM, as well as certain cell phones and GPS receivers
- ADK-capable Android phones and tables
- Digital cameras - Canon EOS, Powershot, Nikon DSLRs and P&S, as well as generic PTP
- Mass storage devices, such as USB sticks, memory card readers, external hard drives
- Bluetooth dongles.

In order to connect the educational robot Arduino with a wireless controller it was necessary to equip the mobile robot with a wireless communication. Thus, the USB Host Shield 2.0 module with a Bluetooth dongle is used as add-on board for Arduino development platform.

The USB Host Shield 2.0 provides USB Host interface, allowing full and low-speed communication. The board contains Maxim MAX3421E USB host controller, 12MHz crystal, level shifters, resistors, capacitors, Reset button and USB A-type connector. There are also a number of solder pads and jumpers, which are marked with red arrows. The board layout is:

- Power Select
- Power pins
- Analog pins
- GPIN pins
- ICSP connector
- GPOUT pins
- Digital I/O pins 0-7
- Digital I/O pins 8-13
- MAX3421E interface pad
- VBUS power pad

MAX3421E interface pads are used to make shield modifications easier. Pads for SS and INT signals are routed to Arduino pins 10 and 9 via solder jumpers. In case pin is taken by other shield an re-routing is necessary, a trace is cut and corresponding pad is connected with another suitable Arduino I/O ping with a wire. To undo the operation, a wire is removed and jumper is closed. GPX pin is not used and is available on a separate pad to facilitate further expansion. It can be used as a second interrupt pin of MAX3421E.

For activation of USB Host Shield 2.0 module with Bluetooth has to be written communication software and uploaded part of which it is shown on Figure 10. For activation of "Wii Remote" one-handed console with the expansion device "Wii MotionPlus" we need to provide a software initialization procedure where after its fulfillment the "Wii Remote" reached to the return states, represented on Table 1.

For the accelerometer in the expansion device "Wii MotionPlus" we need to provide a software initialization procedure as follows:

```
accXwiiM = ((l2capinbuf[12] << 2) | (l2capinbuf[10] & 0x60 >> 5)) - 500;
accYwiiM = ((l2capinbuf[13] << 2) | (l2capinbuf[11] & 0x20 >> 4)) - 500;
accZwiiM = ((l2capinbuf[14] << 2) | (l2capinbuf[11] & 0x40 >> 5)) - 500;
and for the gyroscopes :
gyroYawRaw = ((l2capinbuf[15] | ((l2capinbuf[18] & 0xFC) << 6)) - gyroYawZero);
gyroRollRaw = ((l2capinbuf[16] | ((l2capinbuf[19] & 0xFC) << 6)) - gyroRollZero);
```

```
…….
} /* fill in setup packet */
 setup_pkt.ReqType_u.bmRequestType = bmReqType;
 setup_pkt.bRequest = bRequest;
 setup_pkt.wVal_u.wValueLo = wValLo;
 setup_pkt.wVal_u.wValueHi = wValHi;
 setup_pkt.wIndex = wInd;
 setup_pkt.wLength = nbytes;
 rcode = dispatchPkt(tokSETUP, ep, nak_limit); //dispatch packet
 //Serial.println("Setup packet"); //DEBUG
 if (rcode) { //return HRSLT if not zero
  Serial.print("Setup packet error: ");
  Serial.print(rcode, HEX);
  return(rcode);
{/* Control transfer with status stage and no data stage */
{ byte rcode;
 if (direction)
     { //GET  rcode = dispatchPkt(tokOUTHS, ep, nak_limit);
 } else
{  rcode = dispatchPkt(tokINHS, ep, nak_limit);
 } return(rcode);
}
/* Control transfer with data Stages 2 and 3 of control transfer. */
{ byte rcode;
 if (direction) { //IN transfer
  devtable[addr].epinfo[ep].rcvToggle = bmRCVTOG1;
  rcode = inTransfer(addr, ep, nbytes, dataptr, nak_limit);
  return(rcode);
} else { //OUT transfer
  devtable[addr].epinfo[ep].sndToggle = bmSNDTOG1;
  rcode = outTransfer(addr, ep, nbytes, dataptr, nak_limit);
  return(rcode);
{ .....
```

Figure 10: Example of communication software for activation of USB Host Shield 2.0.

For LEDs	For buttons
0x00, // OFF	0x00008, // UP
0x10, // LED1	0x00002, // RIGHT
0x20, // LED2	0x00004, // DOWN
0x40, // LED3	0x00001, // LEFT
0x80, // LED4	0, // Skip
0x90, // LED5	0x00010, // PLUS
0xA0, // LED6	0x00100, // TWO
0xC0, // LED7	0x00200, // ONE
0xD0, // LED8	0x01000, // MINUS
0xE0, // LED9	0x08000, // HOME
0xF0, // LED10	0x00400, // B
	0x00800, // A

Table 1: Software initialization of “Wii Remote” (return states).

```
gyroPitchRaw = ((l2capinbuf[17] | ((l2capinbuf[20] & 0xFC) << 6)) - gyroPitchZero);
yawGyroSpeed = (double)gyroYawRaw / ((double)gyroYawZero / yawGyroScale);
rollGyroSpeed = -(double)gyroRollRaw / ((double)gyroRollZero / rollGyroScale);
// We invert these values so they will fit the acc values
pitchGyroSpeed = (double)gyroPitchRaw / ((double)gyroPitchZero / pitchGyroScale);
if (!(l2capinbuf[18] & 0x02)) // Check if fast mode is used
yawGyroSpeed *= 4.545;
if (!(l2capinbuf[18] & 0x01)) // Check if fast mode is used
pitchGyroSpeed *= 4.545;
if (!(l2capinbuf[19] & 0x02)) // Check if fast mode is used
rollGyroSpeed *= 4.545;
compPitch = (0.93 * (compPitch + (pitchGyroSpeed * (micros() - timer) / 1000000))) + (0.07 * getWiimotePitch());
```

Conclusion

This working project, nevertheless is still under development and realization, represents the approach and manner for wireless communication by using of Bluetooth connection between human and the educational mobile robot “Audrino” via innovative device - the handle-console “Wii Remote”, which enable to provide the so called ”master-slave” mode of a distant motion control.

References

1. http://arduino.cc/
2. http://learn.parallax.com/BOEShield
3. http://nintendo.wikia.com/wiki/Wii_Remote
4. http://nintendo.wikia.com/wiki/Wii_MotionPlus
5. http://www.circuitsathome.com