

The Command of iRobot Create Using Visual Basic 6.0 and Visual Basic .NET

Genti PROGRI¹, Perika MARANGO²

MSc. Ing Genti PROGRI

Abstract: *The purpose of this paper is the provision of information on how to command the robot Roomba Model 4400. Through this paper will be shown how to command the robot using serial port commands library built for this purpose. The purpose is to show how the Visual Basic 6.0 and Visual Basic.NET programmers, use the command library to build specific programs that control the robot mentioned before. This library of commands and functions is intended to make easier and more comfortable commanding the robot.*

Keywords: iRobot Create, Visual Basic, .Net, Robot, hardware, connection,

1. Introduction

iRobot Create is a programmable model of the Roomba Model 4400, used for educational and scientific purposes. To command this robot from the PC the use of the serial port (7 pin Mini-DIN connector) or of the 25 pin port (DB 25 pin connector) is needed, where the Bluetooth package is inserted (Fig 1). The built library has a large variety of commands and functions which realize specific tasks. This library will serve as a connecting bridge, which means it will send the proper commands from the user to the robot. In general the commands put the robot into motion while the functions bring to the user specific values which are the parameters of the robot.

2. How Hardwares Realize the Connection between the PC and the Robot

Instructions and commands towards the robot will be send through the serial cable which is connected with the COM port of the PC, while in iRobot Create it is connected with the '7 pin Mini-DIN connector' serial port (Fig1). After the physical joint is made, it is needed the proper configuration of the COM port in the PC. This can be done by the following steps : Control Panel / System / Hardware / Device Manager / Ports (LPT & COM). For the configuration of the serial port firstly it should be chosen which of the ports is connected (COM1, COM2 ...). If a PC is being used it is suggested that the connection is made through COM1 ; however, if a laptop is being used, which means it does not have the COM port physically, the connection is made through a USB converter into COM. The wireless connection with the robot can be realized through the Bluetooth package, also known as Element Direct BAM (Bluetooth Adaptor Module).

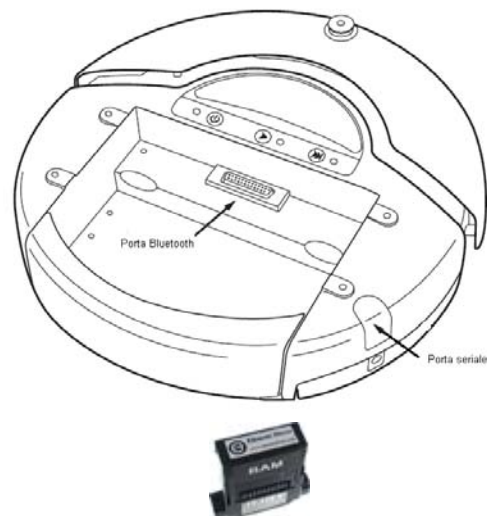


Figure 1: iRobot Create and the BAM element

3. Programs which Help into the Connection of the PC and the Robot

The command of the iRobot Create is realized through a simple program which has a form where buttons with the possible movements of the robot are placed. After a button is clicked, commands and functions in the Visual Basic 6.0 and Visual Basic .NET. libraries is executed. The library for the robot is built using ActiveComport Serial Port. To realize the communication with the iRobot Create through the PC serial port, we could have used Microsoft Comm Control 6.0 (ActiveX Control për Visual Basic 6.0) or the .NET System.IO.Ports library. In both this cases the basic structure of the communications would have been almost the same, with some small changes.

4. Programming with Active Comport Serial Port

The problem of the communication through the serial port, even though it is widely discussed in materials, has it's own specifications. Being that the realization of the communication is complex, a lot of companies built helpful programs which are used as connecting bridges between the serial port and the programmers [4]. One of this programs, also called components COM, is ActiveComport Serial Port.

Components COM is built to help the programmers with the programming languages such as : VBScript, Visual Basic 6.0, Visual Basic .Net and Visual C ++. The usage of ActiveComport Serial Port is really wide, as: it can command different industrial gadgets which can connect with the PC through the serial port; it commands and controls different routers in the web; it controls serial modems, USB, Bluetooth etc.

To create a COM serial object, the following lines are necessary:

VB6:

```
Dim objComport As ACOMPORTLib.ComPort
Set objComport = CreateObject("ActiveXperts.ComPort")
'Creates a Comport instance
```

.NET:

```
Public objComport As ACOMPORTLib.ComPort
Try
objComport = CreateObject("ActiveXperts.ComPort")
objComport.ComTimeout = 1000
Catch
MsgBox("Problem in the creation of ActiveXperts.ComPort
object", vbCritical,
Application.ProductName)
End Try
```

The serial object created should be configured. In this case, for the communication with the iRobot Create, the configuration is as it follows:

VB6:

```
objComport.Device = "COM1" 'Use COM1 port with no
interference from the Windows drivers
objComport.ComTimeout = 100 'Time out of
communication (timeout)
objComport.LogFile = App.Path & "\Errors.Log" 'Log file
where error messages are saved
objComport.BaudRate = 57600 ' BoudRate ne bps (possible
19200 or 57600)
objComport.HardwareFlowControl = 0 '
objComport.SoftwareFlowControl = 0 '
objComport.DataBits = objComport.asDATABITS_8 '
objComport.StopBits = objComport.asSTOPBITS_1 '
objComport.Parity = objComport.asPARITY_NONE '
objComport.Open 'Command for the opening of the port
```

.NET

```
comm = "COM1"
objComport.Device = comm
objComport.ComTimeout = 100
objComport.LogFile =
System.Windows.Forms.Application.ExecutablePath &
"\Errors.Log"
objComport.BaudRate = 57600
objComport.HardwareFlowControl = 0
objComport.SoftwareFlowControl = 0
objComport.DataBits = objComport.asDATABITS_8
objComport.StopBits = objComport.asSTOPBITS_1
objComport.Parity = objComport.asPARITY_NONE
objComport.Open()/To send the data in the port, the
command used is as it follows :
```

```
objComport.WriteString( "any text") ose
objComport.WriteByte wByte
```

The first instruction (objComport.WriteString("any text")) sends one information to the string-arranged port, while the second (objComport.WriteByte wByte) sends to the port 1 byte of information (wByte). In this case, the instruction needed to send the data to the port will be that of a " byte-after-byte".

To realize the process where the data sent from the device on the other side of the cable (in this case iRobot Create) will be taken, the follow instructions are used:

VB6 and vb .NET:

```
objComport.ComTimeout = 100 'The time after it has
TimeOut
```

```
str = objComport.ReadString ' Reads a text information
from the port
```

```
wByte = objComport.ReadByte ' Reads a byte information
from the port
```

```
wByte = objComport.ReadBytes ' Reads a range of byte
information from the port
```

The closure of the communication serial port is done from:

```
objComport.Close 'Command given to close the port
```

According to what was said above, some commands and functions will be listed, to help in the movement of the iRobot Create.

5. Movement Commands and State Functions of the Robot

Some of the most used commands and functions are as follows.

Opening of the serial port for the communication with the robot is done by the function :

RoombaInit "3" where "3" expresses the number of the serial port where the communication will be done.

The building of the function is as follows :

```
Public Function RoombaInit(my_COM As String) As Boolean
Dim Key As Integer
Dim s As String
Dim comm As String
```

```
'Output buffer must be big enough to take largest message size
If IsNumeric(my_COM) = False Then
MsgBox "Undetermined communication port.", vbInformation,
App.Title
RoombaInit = False
End If
```

```
'Determines the communication port with the Romba robot
comm = "COM" & my_COM
objComport.Device = comm
objComport.ComTimeout = 1000
objComport.LogFile = App.Path & "\Errors.Log"
```

```
objComport.BaudRate = 57600
objComport.HardwareFlowControl = 0
objComport.SoftwareFlowControl = 0
objComport.DataBits = objComport.asDATABITS_8
objComport.StopBits = objComport.asSTOPBITS_1
objComport.Parity = objComport.asPARITY_NONE
objComport.Open
```

```
confirmation = GetResult
```

```
If objComport.IsOpened = True And objComport.LastError = 0
Then
Key = 128: objComport.WriteByte (Key):DoSleep 0.1
Key = 132: objComport.WriteByte (Key): DoSleep 0.1
```

```
'light LEDs
```

```
Key = 139: objComport.WriteByte (Key)
```

```
Key = 25: objComport.WriteByte (Key)
```

```
Key = 0:objComport.WriteByte (Key)
```

```
Key = 128:objComport.WriteByte (Key)
```

```
' set song
```

```
Key = 140:objComport.WriteByte (Key)
```

```
Key = 1:objComport.WriteByte (Key)
```

```
Key = 1:objComport.WriteByte (Key)
```

```
Key = 48:objComport.WriteByte (Key)
```

```
Key = 20:objComport.WriteByte (Key):DoSleep 0.05
```

```
' sing it
```

```
Key = 141:objComport.WriteByte (Key)
```

```
Key = 1:objComport.WriteByte (Key)
```

```
RoombaInit = True
```

```
Else
```

```
MsgBox " Unable to open the communication port .",
```

```
vbInformation, App.Title
```

```
RoombaInit = False
```

```
End If
```

```
End Function
```

The function brings back the values True or False in case the connection is realized with the robot, or not.

The movements : forwards, backwards, left and right (with linear speed chosen for each wheel) are given from :

Set DriveWheels Create Speed Wheel Left, Speed Wheel Right

Parameters Speed Wheel Left and Speed Wheel Right chose the speed of the left and right wheel of the robot (Fig 2). Linear speed of the robot wheels is between the values (-500 – 500 mm/s) [2].

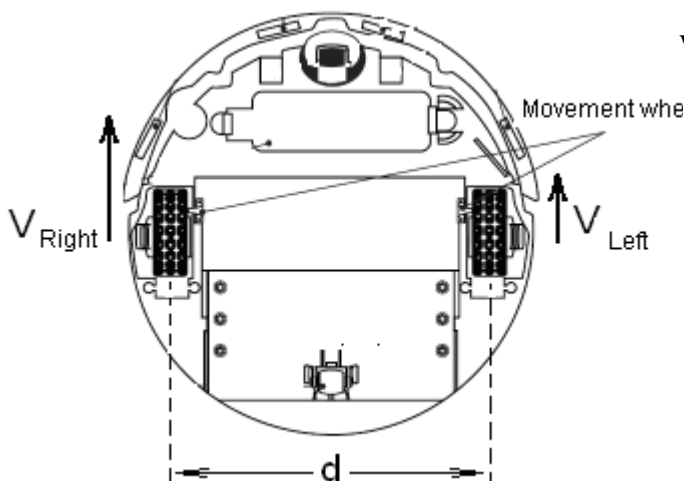


Figure 2: Wheels of the iRobot Create and their linear speeds

When the forward movement of the robot is required, after the speed he will use is chosen, we need the command:

SetDriveWheelsCreate vMovement, vMovement 'Speed of vMovement = [0, 500] mm/s

For the backwards movement it is needed:

SetDriveWheelsCreate vMovement, vMovement 'Speed of vMovement = [-500, 0] mm/s

Rotation 90 degrees left and right :

SetDriveWheelsCreate 0, vMovement 'Speed of vMovement = [0, 500] mm/s

SetDriveWheelsCreate vMovement, 0 'Speed of vMovement = [0, 500] mm/s

Rotation 180 degrees left and right:

SetDriveWheelsCreate - vMovement, vMovement 'Speed of vMovement = [0, 500] mm/s

SetDriveWheelsCreate vMovement, - vMovement 'Speed of vMovement = [0, 500] mm/s

Knowing the distance between the wheels of the robot and the linear speeds we can obtain rotations of different degrees. For this we will use some simple physic formulas. The problem would be : what must the linear speed of the wheels be , when the rotation of the robot with angle α is requested.

We use popular formulas which connect angular speed with linear speed, and linear speed with time:

$$\omega = \frac{v}{r} = \frac{2 \cdot v}{d} \quad (1)$$

$$\omega = \frac{\alpha}{t} \quad (2)$$

From 1 and 2 we can calculate the linear speed of the wheel for a rotation of α degrees through time t .

$$\alpha = \frac{2 \cdot v \cdot t}{d} \quad (3)$$

Where d is distance between the wheels of the robot Fig 3.

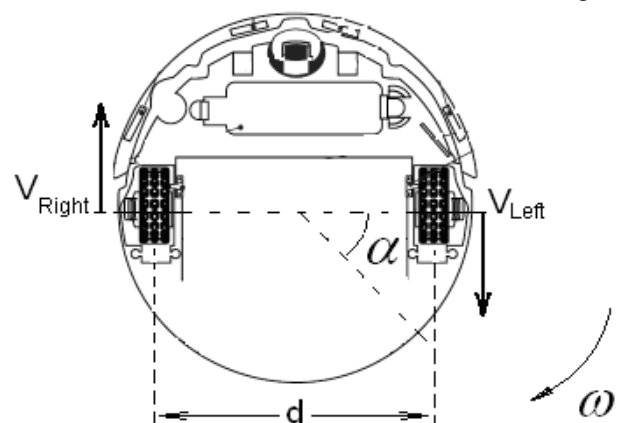


Figure 3: Rotation of iRobot Create

The command for a composed movement in the form of an arc is:

SetFwdVelRadiusRoomba vMovement, rRay

Where vMovement is the linear speed of the robot and rRay is the ray of the path that the robot follows (Fig 4).

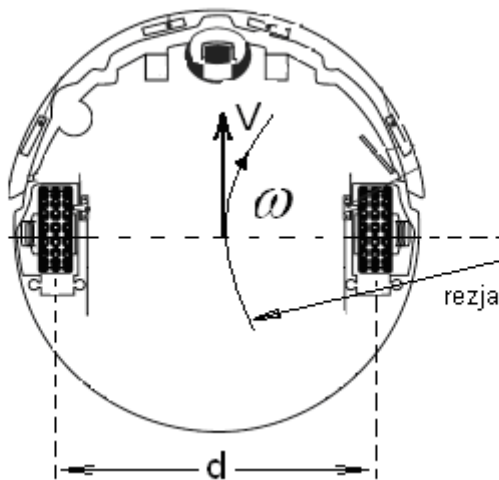


Figure 4. Composed movement of iRobot Create

To stop the robot it is acted as it follows :

SetDriveWheelsCreate 0, 0 or SetFwdVelRadiusRoomba 0,1

For both of the commands shown above, values vMovement and vRay are numbers from 16 bit to 4 byte in total. In the case speed will be -200 mm/s and the ray 500 mm, the information which will be sent is:

[137] [255] [56] [1] [244]

Command [137] puts the wheels of the robot under control for an arc movement.

The range above is taken from the calculations that follow [7] :

vMovement = -200 = hex FF38 = [hex FF] [hex 38] = [255] [56]

rRay = 500 = hex 01F4 = [hex 01] [hex F4] = [1] [244]

To get the above calculations in Visual Basic 6.0 and Visual Basic .NET, functions HIBYTE (intParameter) and LOBYTE(intParameter) must be used :

Public Function HIBYTE(ByVal Intg As Integer) As Byte
HIBYTE = (Intg And &HFF00) / 256
End Function

Public Function LOBYTE(ByVal Intg As Integer) As Byte
LOBYTE = Intg And &HFF
End Function

To choose the distance that the robot moves it is used the *DistanceSensorRoomba()* function which turns backwards the meters done. To put the robot into the beep state (beep sound) it is executed the following command : BeepRoombaVB6VBNet ().

To obtain the information of the batteries of the robot , the following command is used : BatteryChargeReaderRoombaVB6VBNet () or BatteryVoltageRoombaVB6VBNet() which bring the tension value of the source package.

6. The Experimental Realization of the Communication Between the PC and the iRobot Create

To realize the command of the robot, the program built in Visual Basic 6.0 is composed from a form (window) shown in Fig 5 and from 12 modules, where each contains commands, functions connecting with the communication of the robot and other helpful functions. The program is easy. On the window of the communication port (Fig 5) it is chosen the number of the serial port. In case when the PC is connected with the robot through a serial cable, in the communication port is chosen either COM1 or COM2. When we have a wireless connection, in the communication port we chose COM3 – COM8 (This choice is obtained after the installation of Bluetooth)

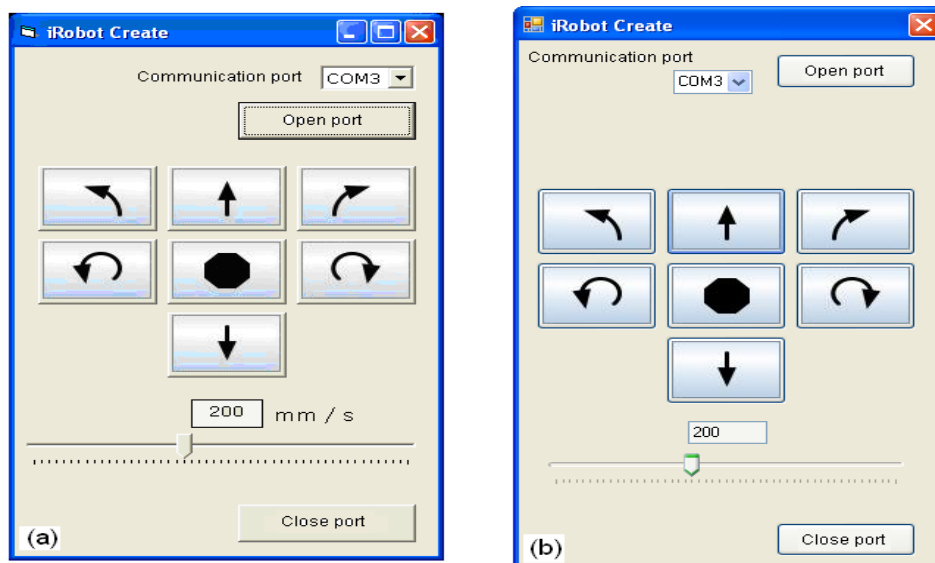


Figure 5. The program to command the robot (a)-VB6, (b)-VB .NET

7. Conclusions

- Through this material it will be shown how can be realized the command of the robot from a serial port using the library built for this purpose.
- The program shows how can Active Comport Serial Port be used in Visual Basic 6.0 and Visual Basic .NET
- Through the program built in Visual Basic 6.0 and Visual Basic .NET it is shown how we can obtain commands to put the robot into linear motion and rotational or in a controlled movement.

References

- [1] Connection Guide For The Irobot Create.
[Http://Portal.Seas.Columbia.Edu/Matlab/](http://Portal.Seas.Columbia.Edu/Matlab/)
- [2] Irobot Command Module Owner's Manual. ©2007 Irobot Corporation.
- [3] Pardue, J., Virtual Serial Port Cookbook.
- [4] Activecomport Serial Port Toolkit Manual Pages 1999-2005 - Activexperts Software
- [5] Axelson J, Access Serial Ports With Visual Basic.NET, Nuts & Volts, April 2008, P60.
- [6] Jan Axelson's Web Site:
[Http://Www.Lvr.Com/Serport.Htm](http://Www.Lvr.Com/Serport.Htm)
- [7] Irobot Create Open Interface (OI) Specification And Virtual Wall. ©2006 Irobot Corporation.