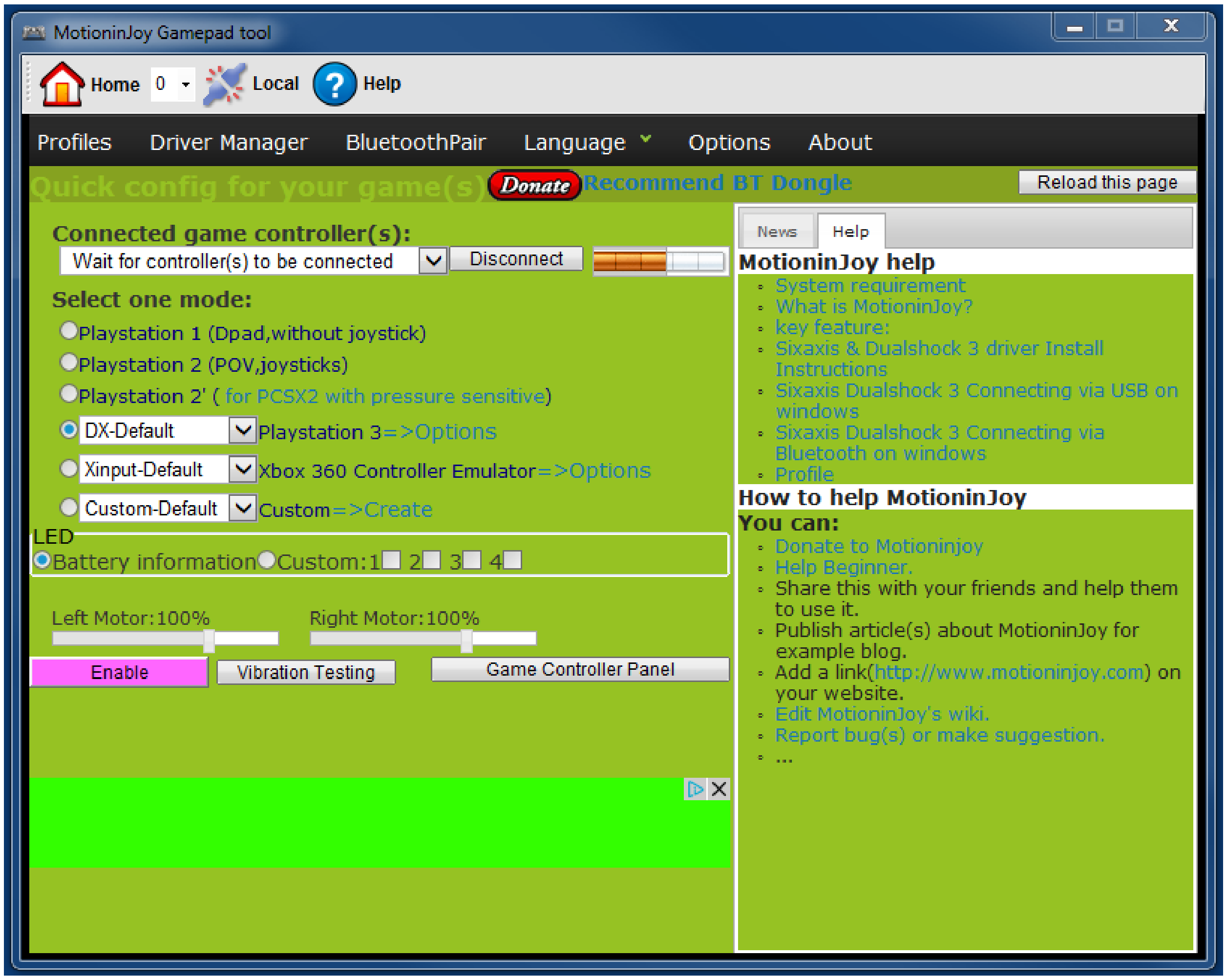
Using a PS3 Controller to control a DFRobot Mobile Platform over WiFi

Written by Justin Ashtiani and Shan Ali
7/17/2013
Georgia Tech Research Institute

Connect your PS3 Controller to your PC using Bluetooth

To connect your controller to your PC, Download and install the Motioninjoy software. (http://bit.ly/9iDG5H).



Connected game controller(s):
Wait for controller(s) to be connected
Disconnect

Select one mode:
Playstation 1 (Dpad,without joystick)
Playstation 2 (POV,joysticks)
Playstation 2' (for PCSX2 with pressure sensitive)
[✓ DX-Default] Playstation 3=>Options
Xinput-Default Xbox 360 Controller Emulator=>Options
Custom-Default Custom=>Create

LED
[✓ Battery information] Custom:1 2 3 4

Left Motor:100%
Right Motor:100%
Enable
Vibration Testing
Game Controller Panel

MotioninJoy help

- System requirement
- What is MotioninJoy?
- key feature:
- Sixaxis & Dualshock 3 driver Install Instructions
- Sixaxis Dualshock 3 Connecting via USB on windows
- Sixaxis Dualshock 3 Connecting via Bluetooth on windows
- Profile

How to help MotioninJoy

You can:

- Donate to Motioninjoy
- Help Beginner.
- Share this with your friends and help them to use it.
- Publish article(s) about MotioninJoy for example blog.
- Add a link(http://www.motioninjoy.com) on your website.
- Edit MotioninJoy's wiki.
- Report bug(s) or make suggestion.
- ...

Launch the software, plug in your controller and Bluetooth dongle (I used the Abe Bluetooth Dongle), and install the drivers for both devices. Note: make sure that you will not use the Bluetooth dongle for any other purpose, as the drivers will make the Bluetooth dongle proprietary to the Motioninjoy software. Make sure the Bluetooth dongle also supports both *Enhanced Data Rate ACL 2 Mbps mode* and *Enhanced Data Rate ACL 3Mbps mode*. After pairing the device with your controller plugged in, move back to the Profiles tab, and unplug your controller. The connected game controller(s) dropdown box will change to (Bluetooth). You can test the Bluetooth connection by selecting *Vibration Testing*.

Using a PS3 Controller to control a DFRobot Mobile Platform over WiFi

Written by Justin Ashtiani and Shan Ali
7/17/2013
Georgia Tech Research Institute

Retrieve joystick values from the controller using DirectInput and C++

Before you begin, retrieve the 5 files that are included to help you acquire joystick data and send the values over Wi-Fi. You will also need a wireless router if you would like to send the commands over Wi-Fi, and the reference page for the DirectInput (http://bit.ly/12wNtmR). The 5 files that you need are the following:

PS3Controller.cpp- Contains structures and functions essential to the execution of PS3Interface.cpp

Ps3Interface.cpp- This is your main program. This program initializes the controller and the UDP connection, and then updates the controller to receive the current values. It then sends those values over Wi-Fi to an Arduino

PS3Controller.h- Contains structures and classes used by PS3Controller.cpp and Ps3Interface.cpp.

UDPCon.cpp- Contains the implementation of the UDP communication

UDPCon.h- Contains the class declaration of the UDP communication object

Understanding PS3Controller.h

Explanation	***Code***
Creates a structure defining integers such as left_y.	`struct controllerStatus`
This class includes variables, constructors and destructors, and functions to determine the values from the joystick and scale them appropriately for the DFRobot.	`class PS3Controller`

Understanding PS3Controller.cpp

Explanation	***Code***

sets a joystick to null, and initializes the controller	`PS3Controller::PS3Controller(void)//constructor`
	`{`
	`g_pDI = NULL;`
	`g_pJoystick = NULL;`
	`Initialize();`
	`}`

Register with the DirectInput subsystem and get a pointer to an IDirectInput interface we can use, and then create a DInput object.	`if( FAILED( hr = DirectInput8Create( GetModuleHandle( NULL ), DIRECTINPUT_VERSION,`
	
	`IID_IDirectInput8, ( VOID** )&g_pDI, NULL ) ) )`

This code queries the interface, and then configures the joystick. If the joystick connects, “PS3 Controller Connected” will print to the command window.	`IDirectInputJoyConfig8* pJoyConfig = NULL;`
	`    if( FAILED( hr = g_pDI->QueryInterface( IID_IDirectInputJoyConfig8, ( void** )&pJoyConfig ) ) )`
	`        return hr;`
	
	`    PreferredJoyCfg.dwSize = sizeof( PreferredJoyCfg );`
	`    if( SUCCEEDED( pJoyConfig->GetConfig( 0,`

	`&PreferredJoyCfg, DIJC_GUIDINSTANCE ) ))`

Calls for each enumerated joystick,	`BOOL CALLBACK`
	`PS3Controller::EnumJoysticksCallback( const`
	`DIDEVICEINSTANCE* pdidInstance,`
	`VOID*`
	`pContext )`

Gets the DirectInput State of joystick, polls the device to read its current state, and defines the directional axis to a joystick axis.	`HRESULT PS3Controller::UpdateInputState( )`
Initializes DirectInput Variables	`VOID PS3Controller::FreeDirectInput()`

Determines the current joystick position and its raw value, and then scales the value within a range of 0 to 255.	`int`
	`PS3Controller::getControllerAnalogPosScaled(int`
	`which)`

Understanding PS3Interface.cpp

This program is what you will run when you need to control the robot. This program is the simplest of all, as it initializes the Wi-Fi and controller portions of the program, and then proceeds into a loop which will run until you end the program.

Explanation	***Code***

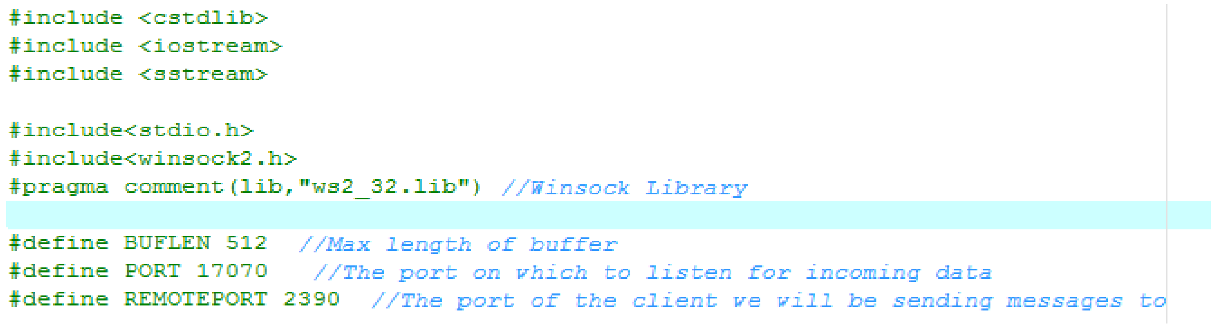
Initializes PS3 controller	`PS3Controller MyController;`
	
	`bool quit = false;`

Initializes UDP	`UDPCon myConnection;`
	`myConnection.Initialize();`
	`myConnection.ConfigRecieveAddr();`

Updates input state, gets, scales, and sends the data for both joysticks over Wi-Fi to the Arduino	`while(!quit)`

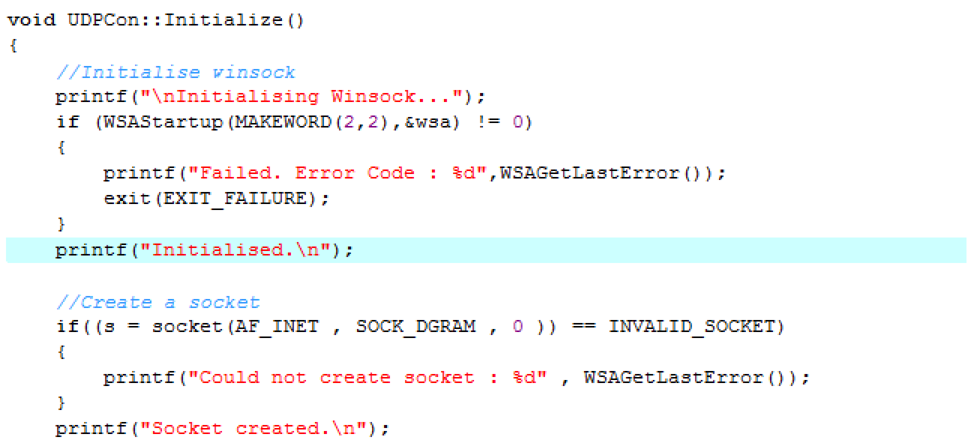
UDPCon – UDP Communication Code

The class in this particular program is named UDPCon. In this class, there are four major functions, which are Initialize, SendData, ConfigRecieveAddr, and SendJoystickData. These four main functions each play critical roles in the successful compilation of this program. The list below shows the required includes for operation of the code. Also defined is a **PORT** and **REMOTEPORT**. *You must enter the remoteport that the Arduino code is configured to listen on!*



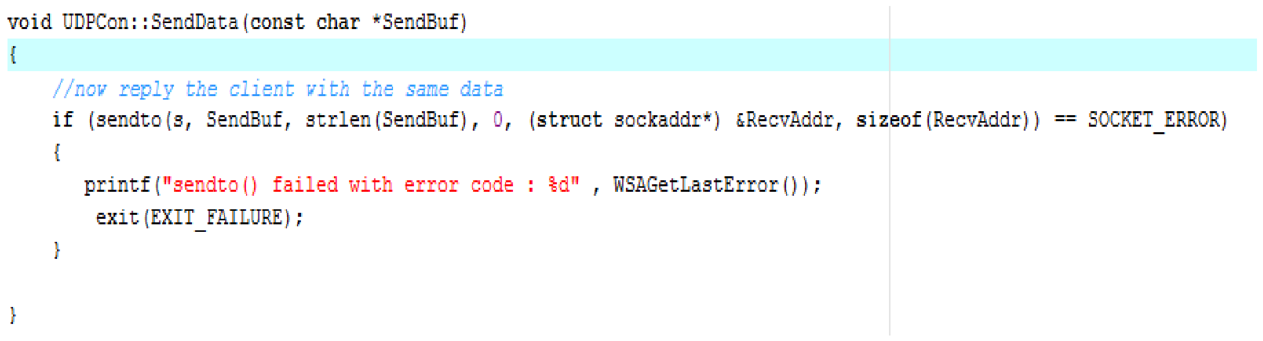
```
#include <cstdlib>
#include <iostream>
#include <sstream>

#include<stdio.h>
#include<winsock2.h>
#pragma comment(lib,"ws2_32.lib") //Winsock Library

#define BUFLEN 512  //Max length of buffer
#define PORT 17070   //The port on which to listen for incoming data
#define REMOTEPORT 2390  //The port of the client we will be sending messages to
```

The first function is the Initialize function. Before you can start opening up sockets, it is necessary to initialize the Winsock. The program will print that it has initialized the Winsock once it has performed the task successfully. After you write the code for initializing the Winsock as shown, you must create the socket, which will let you send data over them. Once the socket is created, that program will print that it has been created successfully.



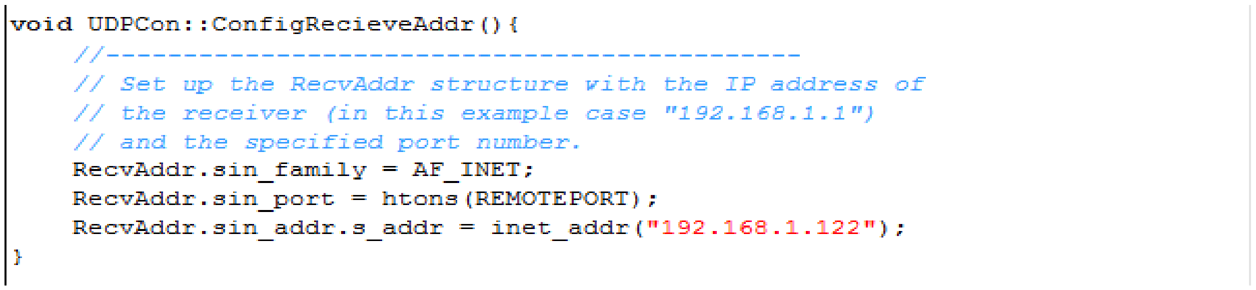
```
void UDPCon::Initialize()
{
    //Initialise winsock
    printf("\nInitialising Winsock...");
    if (WSAStartup(MAKEWORD(2,2),&wsa) != 0)
    {
        printf("Failed. Error Code : %d",WSAGetLastError());
        exit(EXIT_FAILURE);
    }
    printf("Initialised.\n");

    //Create a socket
    if((s = socket(AF_INET , SOCK_DGRAM , 0 )) == INVALID_SOCKET)
    {
        printf("Could not create socket : %d" , WSAGetLastError());
    }
    printf("Socket created.\n");
```

The next function is the SendData function. Since the UDP socket is not connected to the receiver, you would have to tell the Winsock the address of the receiver in another way, by using the Sendto function. In this function, the program is replying the client with the same data. This makes it easier to send several strings over the socket.



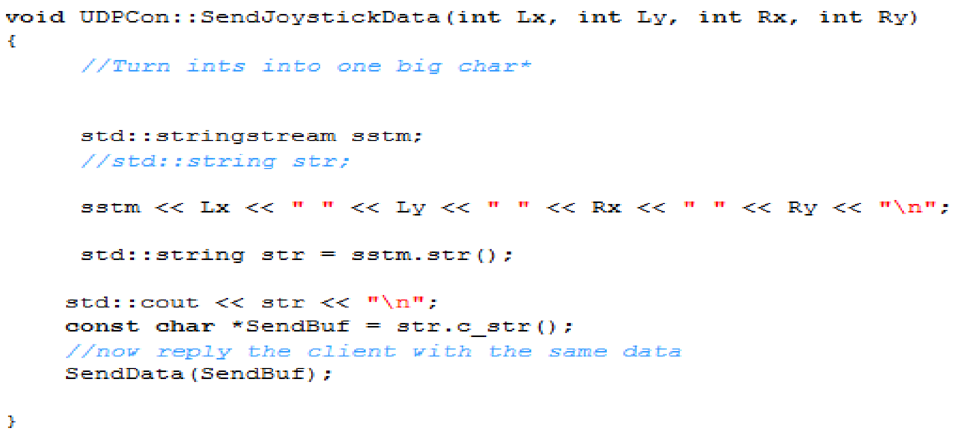
```
void UDPCon::SendData(const char *SendBuf)
{
    //now reply the client with the same data
    if (sendto(s, SendBuf, strlen(SendBuf), 0, (struct sockaddr*) &RecvAddr, sizeof(RecvAddr)) == SOCKET_ERROR)
    {
      printf("sendto() failed with error code : %d" , WSAGetLastError());
       exit(EXIT_FAILURE);
    }

}
```

The third function in the class, that was created, is the ConfigRecieveAddr function. This function is designed to implement the IP address of the Arduino board and the specified port number of the receiver. It's necessary to input the IP address into the last line where the red text is located. The specified remote port number is also required to be inputted. This value should be placed in the second to last line where "RemotePort" is located.

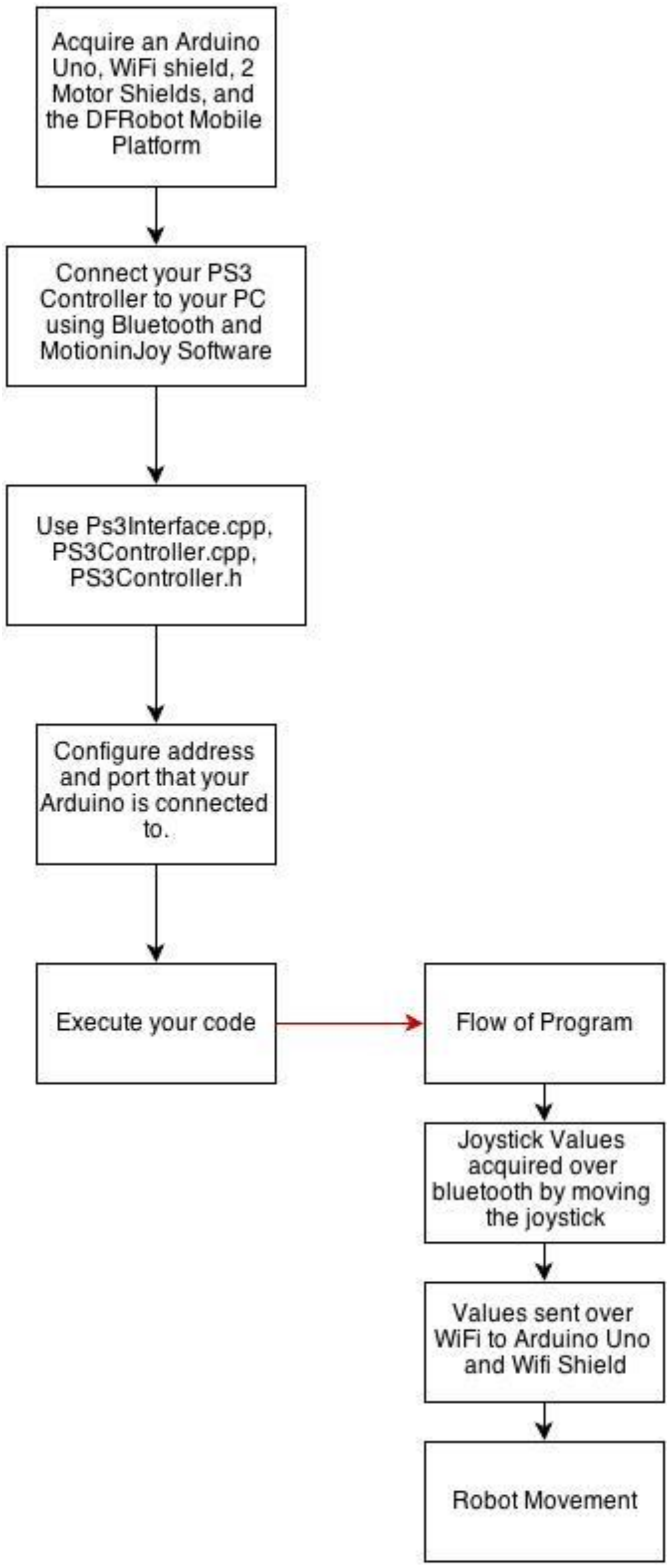


```
void UDPCon::ConfigRecieveAddr(){
    //-----------------------------------------------
    // Set up the RecvAddr structure with the IP address of
    // the receiver (in this example case "192.168.1.1")
    // and the specified port number.
    RecvAddr.sin_family = AF_INET;
    RecvAddr.sin_port = htons(REMOTEPORT);
    RecvAddr.sin_addr.s_addr = inet_addr("192.168.1.122");
}
```

The final function in this program is known as the SendJoystickData function. This essential function starts off with four integers which represent Lx, Ly, Rx, and Ry. This function takes the four integers and turns them into one big string. After these integers are put into a single string, the function then sends this data out over the socket to the receiver



```
void UDPCon::SendJoystickData(int Lx, int Ly, int Rx, int Ry)
{
    //Turn ints into one big char*


    std::stringstream sstm;
    //std::string str;

    sstm << Lx << " " << Ly << " " << Rx << " " << Ry << "\n";

    std::string str = sstm.str();

   std::cout << str << "\n";
   const char *SendBuf = str.c_str();
   //now reply the client with the same data
   SendData(SendBuf);

}
```


Acquire an Arduino Uno, WiFi shield, 2 Motor Shields, and the DFRobot Mobile Platform
Connect your PS3 Controller to your PC using Bluetooth and MotioninJoy Software
Use Ps3Interface.cpp, PS3Controller.cpp, PS3Controller.h
Configure address and port that your Arduino is connected to.
Execute your code
Flow of Program
Joystick Values acquired over bluetooth by moving the joystick
Values sent over WiFi to Arduino Uno and Wifi Shield
Robot Movement