COS 495 – Autonomous Robot Navigation
Fall 2011

Lab 1 Manual

1.1 Goals
The programming interface goals of this lab are as follows:

- Introduce students to the Iver2 login system and communication between processors.
- Introduce students to the C# API that allows control of the Iver2 actuators

The robot control theory goals of this lab are as follows:

- Introduce students to feed forward control
- Introduce students to feedback control with a Proportional control system

The programming application goal of this lab is to:

- Use P Control for point tracking

1.2 Get the base code
From the COS 495 website, download the lab base code folder. Unzip the folder. You can open it up and look around the files if you like. Note your code changes to this solution will be in the file "Form1.cs". Take care to go through how similar this is to previous labs. There is a second thread that starts when the start button is clicked. The stop button aborts the thread.

Look at the RunThread function. This is where you will edit the code. Within the while loop there is a section of code that sets your control variables (motorSpeed, yawFin, pitchFin). Figure out where in the code the values of these variables get sent to the main processor.

Note that during the lab,

1.3 Running your first C# Form Application on the Iver2
This portion of the lab should be done with your group. You will compile and do preliminary tests on your computer. Then, you will copy the executable file iceFinder.exe to the iver2 through a wireless network called IVER.

Step 0: Power up the Iver2 !
Make sure the Iver2 is powered on. There should be green and red lights lit up on the antenna. If not, use the remote car lock buttons to turn it on. Do not cycle the power unnecessarily, this will waste time.

Step 1: Connect to the IVER network
Make sure the computer you are using is connected to the IVER wireless network (not the puwireless network). The password is 5086780550, check the card on the wireless router box (yellow) if you need to.

Step 2: Remote Desktop to the Iver2 main processor
Start the program "Remote Desktop Connection". Select the computer "iver2-132" to login to. The username is "iver" and the password is "i". Be patient, it can take a while to login.

Step 3: Run UVC Software Application
From the *Desktop* of the iver2, select "Underwater Vehicle Console". You can click the "Instruments" button to ensure all sensors are running. The UVC needs to be running on the primary processor the entire time since it will communicate with your code on the secondary processor. However, make sure the Manual Control pop-up window is not running since it may send signals the actuators that conflict with your own.

Step 4: Remote Desktop to the Iver2 secondary processor
From within the Iver2 main processor remote login, select the *start =>Programs=>Accessories => Communications* menu item called "Remote Desktop Connection". Connect to "Iver2-02" or "192.168.2.2". The login should be done for you.

Step 5: Transfer your code to the secondary processor
On the laptop computer you are using, open the folder labeled "Transfer". To do this, select *start => Computer*. When the window pops up, click "Network" in the left column. There you should seed an icon for the IVER2-132, the main computer on the Iver2. Double click this icon and you should see the Transfer folder.

Copy the iceFinder application's executable file "iceFinder.exe" from your computer into the Transfer folder. This file will be found in your C# solution's "Debug" or "Release" folder. For example, in *iceFinder => bin => Debug*, there is a file called iceFinder.exe. The ".exe" extension informs you it is an executable file.

On the secondary processor remote login, there is an icon on the desktop labeled My Network Places. Double click this icon. A window will pop up with the "Transfer" on Iver2-132. Open this "Transfer" folder and find your executable file. Drag the iceFinder.exe file to the desktop.

You can now run the file by double clicking the iceFinder.exe file. The applications form window should pop-up. It won't do much now except write simple messages to the log box when the two buttons are pressed. Simply close the form application, close both of your remote connections and return to your computer to continue on to the next step of the lab.

1.4 Creating a repetitive control motion on the Iver2

You can write your code and compile it on your own computer. Once done, ask to connect wirelessly to the iver2 for testing. Unfortunately we all have to shark the one AUV.

Step 1: Add a counter condition

Within the while loop of the Run_Thread method (in Form1.cs), you must create a conditional statement based on a counter.

First, outside of the while loop, create a variable named "count" of type int and initialize it to 0.

```
// Setup a counter
int count = 0;
```

Now, within the loop, after the Thread.Sleep, add an "if" statement that checks if the count is less than 5. If it is less than 5, the fins and motor speed will be adjusted accordingly.

```
if (count < 5)
{
    motorSpeed = 138;
    yawFin = 200;
    pitchFin = 200;
}
```

However, we will also add an else statement in case count is between 5 and 10.

```
else if (count < 10)
{
    motorSpeed = 128;
    yawFin = 45;
    pitchFin = 45;
}
```

The last possible situation we will check for, is if count is greater than 10. If it is, we will set it to zero.

```
else
{
    count = 0;
}
```

Finally, lets log the current control signal for yawFin and increment the count for the next iteration of the while loop.

```
LogBox.Items.Add("Count: "+count + ", Yaw Fin: "+yawFin);
count += 1;
```

So, your while loop should include:

```
// Setup a loop to determine when to swich control modes
if (count < 5)
{
    motorSpeed = 138;
    yawFin = 200;
    pitchFin = 200;
}
else if (count < 10)
{
    motorSpeed = 128;
    yawFin = 45;
    pitchFin = 45;
}
else
{
    count = 0;
}
LogBox.Items.Add("Count: "+count + ", Yaw Fin: "+yawFin);
count += 1;
```

* NOTE: Be sure to delete the actuator setting lines currently in the basecode like "`yawFin=128`". They are in the basecode to make sure they are set to 128 during initial debugging. However they must be deleted for your code to work properly.

Step 2: Compile and run
Make sure this compiles. When you get a chance, try out this code on the iver2. Watch the behavior of the iver2. Note the timing of the system and how the actuators correspond with the counter.

1.5 Creating a P Controller for Yaw on the Iver2

You and your group should create your code base and compile it on your own computer. Once done, copy your iceFinder.exe file to the iver2.

Your code should track 1 degree (is this due North?). To do this, code up a proportional control system that actuates the yawFin to steer the vehicle towards 1 degree. We can test this with dry land tests.

You will probably want to make use of the compState.getHeading() method which will return the current compass yaw measurement in degrees.

Make sure you write a message to the log box that reports the current yaw measurement. This will help with debugging.

1.6 Deliverables

The student pair must demo closed-loop P control of the Yaw fin to the instructor or grader by the end of Friday's class, on September 23rd.