Application notes for using the blimp gondola in the LITEC room.

Hardware:

1) Gondola
    a. Power switch: location the side of the gondola.
    b. Battery pack will last for more than an hour at full power. There is a charger in the TA corner room. (TAs will take care of batteries.)
    c. The down load is done the same as with the car, connect the USB cable and JTAG adapter to the gondola
    d. There is an RF link for the RS-232 serial connection. printf and getchar() can function without a hardwire connection. Response data should be done on a gondola with an RF link.
    e. There is an ultrasonic ranger and electronic compass on the bottom of the gondola, so in the LITEC room we will flip the gondola. This way the ranger looks up and will "see" more than 6". It means that the compass is flipped compared to blimp flying. So the compass will read backwards – north and south are the same, but east and west are backwards. The compass will have to be calibrated for the operating condition. But the tail fan for the turntable in the class room is wired reversed compared to the fan on the blimp, so your code won't see a difference – the compass is reversed and the fan motor is reversed – the effects cancel.
    f. The fans all have greater thrust in the forward direction than in the reverse. This will be apparent as you run the system. It is a feature of the system, not an error or problem. A proportional plus derivative controller will function well.
    g. 8 switches are mounted on the blimp. They are on all 8 pins of port 3 and can be used as they were for lab 3. Or they can be used to start and stop the code, or for other team defined options.
2) RF link
    a. The RF link connects directly to a USB port of your laptop. It requires a driver available at http://www.ftdichip.com/Drivers/VCP.htm Try downloading and running the "installation executable" which links to: http://www.ftdichip.com/Drivers/CDM/Win2000/CDM_Setup.exe
    It that doesn't work, download the zip file and the instructions.
    b. The RF link runs at 38400 baudrate, which is the same as what we have been using.
    c. The link sends data in packets. It ignores any new communication while it is sending a packet. So printf statements must:
        i. Be shorter than 92 bytes
        ii. Be spaced out in time. Only printing during each new range reading should work.
    d. The RF links use channel numbers, so don't try to move them from gondola to gondola. It won't work.

Software:

1) Capture compare modules: P0.4 is connected to the speed controller for the rudder, P0.5 is connected to the thrust angle servo, and P0.6 is connected to the thrust power speed controller on the left side and P0.7 is connected to the thrust power speed controller on the right side. It is recommended that your code set XBR0=0x25; in which case:
    a. CCM0 (cex0) will control the rudder fan
    b. CCM1 (cex1) will control the thrust angle
    c. CCM2 (cex2) will control the thrust power (this is different than what some teams used on the car. But it is consistent with the detailed requirements of Lab 3.)
    d. CCM3 (cex3) can be set to control the right thrust fan (a cable option inside the gondola.)
2) Pulsewidths
    a. Both the rudder fans and the thrust fans use speed controllers. These are identical to the ones used for speed control on the Smart Car. The heading pulsewidths must be adjusted to these values, PW_min=2000, PW_neutral=2750, PW_max=3500.
    b. The thrust angle is controlled by a servo. The team must have a code that allows for adjustment of the angle. The pulsewidth needed for vertical thrust will vary from gondola to gondola, and may even vary with use on the same gondola. Don't hard code in a pulsewidth, allow for it to be adjusted each run. This code is very similar to the steering calibration code created by the steering pair in lab 3. Modify and use that code.
    c. Once PD control is implemented, relatively large proportional gains are useable. The result is that the calculation of the temp pulse width may go negative and/or exceed the range of an int. You need to type cast terms on the right hand side of the pulse width equation as long int. For example, if Kp is the proportional gain and Kd is the differential gain, then the following might appear in your code:

```
error = desired_heading – heading;
tmp_pw=(long)Kp*error+(long)Kd*(error-previous_error)+RUDDER_CENTER;
if (tmp_pw > (long)RUDDER_LEFT)  tmp_pw = RUDDER_LEFT;
if (tmp_pw< (long)RUDDER_RIGHT) tmp_pw = RUDDER_RIGHT;
rudder_pw = tmp_pw;
previous_error = error;
```

    tmp_pw should be a long to handle both large positive or negative calculations

    d. The rate of change of the heading is relatively slow, usually only one or two degrees per heading reading. One may want to consider some smoothing of the control code, perhaps using several previous readings.

3) Gain – The rudder fan on the turn table is designed for an inverted gondola. The gondola should always be upside down when on the turn tables. If this is done, then the sign of the gains used in the LITEC room are the same as those on the blimp.

4) Compass - The compass needs to be recalibrated every time the gondola is switched from inverted operation to normal operation.

5) Battery Voltage: There is a voltage divider across the blimp battery with the output connected to pin 5 of port 1. You are to do an A/D conversion on this pin, which of course means that this pin must be configured for A/D. The divider has R1=22kΩ and R2=6.8kΩ, so the voltage read by the A/D is 0.236*V(battery).