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

Hardware:

1) Gondola
   a. The power switch is located on the side of the gondola. There are 4 other switches on the sides: 3 that disable the fan motors (for safety or to save battery) and 1 that enables the gondola to respond to the RF remote controller (**not** the RF link below). The 3 switches labeled "TAIL, "LEFT", and "RIGHT" must be ON to enable the respective fans. The switches disable the motors by disconnecting the battery from the controllers. **The switch labeled "Remote" must be OFF, otherwise the motors cannot be controlled by the 8051.**
   b. A 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 handing out and charging batteries.)
   c. Downloading your code 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(), putchar() and getchar() can function without a hardwire connection. Response data must be collected on a gondola through 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 a flying blimp. So the compass will read backwards – north and south are the same, but east and west are backwards. There are two implications of this: 1) compass angular heading will be reversed when compared to the right-side-up blimp and 2) the compass will have to be calibrated for the operating condition.
   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 DIP 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. We recommend plugging the RF link into your USB port and seeing if the operating system finds the driver. If that doesn't work, the driver is available at: http://www.ftdichip.com/Drivers/VCP.htm. Try downloading and running the "setup executable" that links to: http://www.ftdichip.com/Drivers/CDM/CDM%20v2.12.00%20WHQL%20Certified.exe.
   If that doesn't work, download the zip file and the instructions. You may also go to the LITEC LMS page **Course Resources > Software & Drivers** for the appropriate FTDI driver.
   b. The RF link runs at 38400-baud rate, 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, (the Keil compiler already cuts off at 40 bytes, so this isn't a real limitation.)
      ii. Be spaced out in time. Only printing during every other range reading (or 2 to 3 times a second) should work.
   d. The RF links use channel numbers assigned to specific gondolas, so don't try to move them from gondola to gondola. They won't work.
   e. The receiver unit should not be too close to the gondola. **If it is less than 3 feet away the signal may be over-modulated and be corrupted.** The characters on the terminal will be garbled or turn to graphical symbols. If this happens SecureCRT must be restarted.
   f. The receiver unit for a gondola must remain on the table with the gondola. Do not place it in the box with the receivers for the car RF units.

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 left thrust power fan (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) will control the right thrust power fan (a cable option inside the gondola.)

   NOTE: Left and Right are relative to the gondola hanging from a blimp with the ranger pointing down and the tail fan on the backend (flipped over when compared to mounted on the turntable). CEX3 is on the same side of the gondola as the PWR toggle switch and CEX2 is on the opposite side.
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=2028, PW_neutral=2765, PW_max=3502.
   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. See Worksheet 11. Once PD control is implemented, relatively large proportional gains are desirable. The result is that the calculation of the temporary 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 (rudder_pw is declared as an unsigned int):

```
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;
else if (tmp_pw < (long)RUDDER_RIGHT) tmp_pw = RUDDER_RIGHT;
rudder_pw = (unsigned int)tmp_pw;
previous_error = error;
```

   tmp_pw must 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 turntable is designed for an inverted gondola. The gondola should always be upside down when on the turntables. 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 3 of port 1. You are to do an A/D conversion on this pin, which of course means that this pin must be configured as analog. The divider has R1=22kΩ and R2=6.8kΩ, so the voltage read by the ADC is $0.236 \cdot V_{battery}$ (in other words, the actual battery voltage is 4.235·ADC voltage).