

Programmers Notes 1/23/2006

Kent, Ashley

Accomplished

- Programmed bot to use opposing wheel motion to align bot bearing to camera pan. Nearly useless with old 1020 bot with two wheel drive.
- Recalibrated the Victor 884 motor controllers to allow full range (0-254) with 127 as off. This allows much finer voltage control.
- Added a +/- 2 PWM-value dead band in software during aligning to keep the drive motors from hunting. This works reasonably with the 8-bit resolution of the PWMs and recalibrated controllers, but doesn't allow very fine control.
- Began moving all PhyXTGears code to team1720.c and team1720.h for easy maintenance. Our functions will then be called from user_routines.c and user_routines_fast.c as appropriate.

Problems discovered

- Default camera code had a couple of constants defined incorrectly. Specifically, the pixel centers for the camera X/Y axes were switched.
- Default code also needed to be adjusted to more accurately reflect the PWM center for our pan servo. This changed the default value of 124 to 103. Slight variations in the pan linkage probably accounts for this, as mechanical advance is used to change the servo's 90-deg swing to ~160 deg camera pan.
- The correct tilt center will have to be adjusted in the same manner.
- Default scan code is not very intelligent. It scans in one direction only instead of using a more efficient 'S' search pattern. This needs work.
- Changing the default tilt range affects scan functionality. The scan code resets the camera to the lowest tilt value, then skips the low row entirely. Causes the camera to briefly see the target then shift and lose it. Shooting this low is probably not an issue for competition, but makes testing inconvenient.

Recommendations for next session(s)

- Attempt to make use of the pixel error values returned by the camera. Since the camera search code has its own (adjustable) PWM dead band, this will be necessary for fine pointing.
- Test pointing accuracy and repeatability with some sort of boresight (laser pointer preferred).
- Add timing counter to allow the bot to determine if it has actually moved in response to a tracking command or if power needs to be boosted to get motion.
- Need a failsafe timeout if the above is used.
- Determine the settle time needed for the camera to actually acquire the target, and speed up the search pattern if possible.
- Discuss what happens if bot fails to acquire the target during autonomous mode. Should it attempt to rotate (potentially 360 degrees) and bring the camera to bear on the target?

- COMPREHENSIVE, WRITTEN CALIBRATION SCHEME TO CORRECT FOR ROUGH SHIPPING!!!

Wish list

- Joystick button up/down click-to-adjust pan center while running instead of recompiling each time
 - Acceleration/deceleration function when tracking instead of simple-minded on/off.
 - 'S' pattern for camera search, or better yet, outward spiral from neutral position.
 - Timing calculations during fine tracking (pixel-level) to time required for pixel error value to change. This can be used for sub-pixel interpolation, increasing accuracy.
 - A low-backlash turret to take advantage of the above.
 - Fries and a Coke. j/k
-