Software Engineering 6GA3 – Real-time Systems

Real-time Re-positioning of a Motorised Platform to Match a Reference Platform Using Accelerometers

Graduate Student Course Project

Ron Harwood – 0871419

Objective

Produce a three degrees of freedom motorised platform that automatically changes position to match that of a reference platform.

Method

The motorised platform are suspended by three motorised sliding potentiometers, while the reference platform are suspended by three non-motorised units. Both platforms carry a three-axis accelerometer to measure a position relative to the direction of gravity.

The motors and accelerometer are controlled and monitored by a single AVR atmega328p microcontroller, clocked at 8MHz (using the internal oscillator) and operating with Arduino firmware. The reference accelerometer and the position of the motorised sliding potentiometers are monitored by two additional atmega328p microcontrollers configured identically to the first.

Driving of the motors is achieved with a combination of a Texas Instruments TLC5940 16 channel pulse-width modulation (PWM) driver, connected to two ST Microelectronics L293B motor drivers. The combination of PWM with the motor drivers allows for digitally controlling the speed and direction of the motors at higher than TTL voltage (12V rather than 5).

The microcontrollers are connected together via an i2c network for command and data communication with a Raspberry Pi (RPi) computer. The RPi connects to the i2c network with a bidirectional logic level converter - the RPi operates at 3.3v while the rest of the logic circuitry is operating at 5v. Commands are sent from the RPi to the microcontrollers to request data collected or update data and settings.

Three PID controllers are implemented on the microcontroller that controls and monitors the motorised platform. The accelerometer data from the reference platform is retrieved by the RPi and then sent to the motorised platform microcontroller - this data then acts as the set points for the the PID controllers.

The data retrieved from the sliding potentiometers is averaged by the RPi and if the platform is too high or low, a correction command is sent to the motorised platform microcontroller.

The RRpi is running a real-time linux extension called Xenomai. Four separate real-time tasks are run via Xenomai. The first communicates setpoint data to the motorised platform microcontroller, while the second task retrieves data from the reference accelerometer. These two tasks share the highest priority. The third task retrieves data from the sliding potentiometer monitoring accelerometer, averages the values and sends correction data, if necessary, to the motorised platform controller. The third task runs at a slightly lower priority than the first two. The fourth task simply prints out task stats for user review, and runs at a much lower priority and frequency than the first three tasks.

While i2c has arbitration measures built into it, they only arbitrate between devices communicating on the network - individual processes on a device must have their use of the network regulated, or failures occur. A resource lock, known as a mutex (for MUTual EXclusion) in Xenomai, is used to ensure only one process at a time communicates on the network. Xenomai implements a priority inheritance protocol algorithm, to prevent priority inversion from occurring, when a mutex is in use.

Observations and Results

The motorised platform does correct its position to one matching the values of the reference platform accelerometers, however, tuning of the PID and settling time between corrections and measurements are needed.

The correction of the platform position alters the reading for its accelerometers, but the motion of the correction temporarily affects the accelerometer readings as well. Allowing the platform to move for a correction, stop and then rest for a short period gives the accelerometer a chance to stabilize its measurements. Experimentation with the correction time and rest period showed that the system becomes increasingly unstable as the total period decreased and/or the ratio of of correction time to rest time increased.

The PID tuning was done through manual experimentation, and could still be improved upon. When Kp values were set too low, the motors had trouble overcoming static friction and, for upward movement, gravity. Of course downward movement was less hampered at the same Kp value, so dampening of the output settings for downward movement has been attempted (50% output seems to work adequately). Increasing of Kp, for small corrections, generally did not destabilize operations, but could cause significant ringing for any gross movement of the reference platform - even accidental bumping of the table that the platforms were sitting on could cause sudden and wildly erratic behaviour. Setting of Ki and Kp to fairly small values seemed to add some benefit - but any significant increase immediately destabilized the system.

Problems Encountered and Lessons Learnt

The i2c implementation on the RPi has a hardware flaw that does not tolerate "clock stretching" - a technique used when a "slave" device needs extra time to respond to a command. The command will succeed on the slave device, but the RPi will report errors. The bitrate of the the i2c network was reduces to 10kbit/s, from the normal 100kbit/s - mitigates this issue and does not impact system performance in any noticeable way, as only two or three bytes are sent in any command or response.

Stability of a PID system, that is dependant on accelerometers, needs "gentle" correction and a "patient" algorithm. The measurements of position still need to be done before additional corrections are made to platform orientation, but settling time of the accelerometers needs to be taken into consideration. Continuous driving of the motors to correct position will only exacerbate the error of the PID system. As a consequence, this leads to a sluggish response and long times to achieve correction.

Possible Improvements

Additional sensors could be used in determining the position of the reference platform. For example, a combination of gyroscope and accelerometer would provide both orientation and a measurement of the forces exerted on the reference platform.

A possibility for overcoming the static friction issue might be to base the output of the PID on time rather than voltage to the motors. The time for settling would probably not have be directly related to the time of correction, and it may be possible to use a static value for the time.