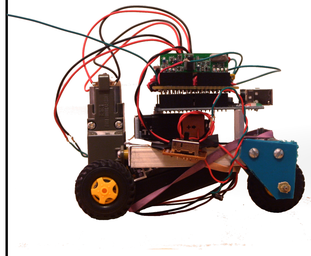
Keegan Mann
San Pasqual High School



Stability Analysis of Control Algorithms

Base Angle Estimation

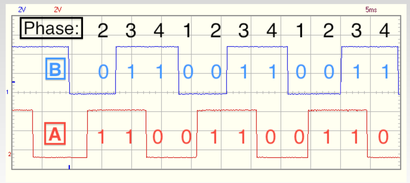
The angular velocity of the base cylinder can be calculated from the sensors on the robot according to the following equation:

$$\omega_b = \omega_g - \omega_w * r_w / r_b$$

(see free body diagram)

This value can be used in an additional PID which sets the setpoint of the primary PID. Using two PIDs in this way is referred to as cascading.

Encoder



A quadrature encoder outputs two square waves 90 degrees out of phase.

		0		1		Prev. A
		0	1	0	1	Prev. B
0	0	0	-	+	E	
	1	+	0	E	-	
1	0	-	E	0	+	
	1	E	+	-	0	
A	B					

By noting the change in state of the two lines, we can detect steps in either direction.

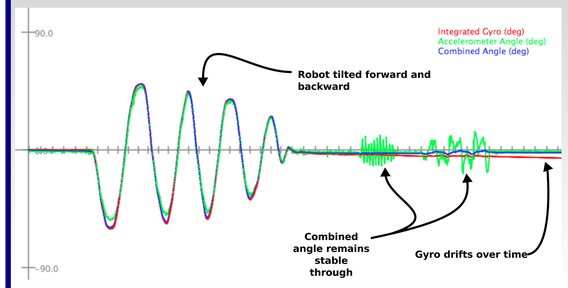
Results

The gyro and accelerometer didn't output acceptable data on their own and must be combined in order to arrive at an accurate angle reading. As expected, the frequency response curve was upward sloping for higher frequencies, but it was interesting to note that the cure initially decreased. Being able to see the output of the system graphically was essential to tuning the gains in the control loop and detecting problems in software. For example, a sign reversal in the D term was not apparent until I graphed the output of the different terms of the PID.

Conclusions

A PID can be used to solve a wide range of control systems, but must be tuned carefully in order to get good results. For more complex systems, a cascaded PID can be used, but the increased complexity makes the system much more difficult to debug and tune. In my case, integral windup was a serious problem. In the future, I may want to investigate methods to mitigate this such as setting limits or using a floating average.

Orientation Calc.



A dual axis accelerometer can be used to get an angle measurement, however the signal tends to be noisy and will interpret lateral accelerations as changes in angle. A rate gyro yields a signal which can be integrated to obtain an angle which is accurate and doesn't respond to lateral accelerations. However, the rate gyro signal has error which accumulates over time causing it to drift. In order to use the strengths of both devices we can use the following equation to combine the gyro and accelerometer outputs into a reliable rotation measurement:

$$\theta_{n+1} = (\theta_n + \omega_g * T) * k + a * (1 - k)$$

Where "θ" is the output angle, "ω_g" is the gyro rate signal, "T" is the time interval between iterations, and "a" is the accelerometer angle. "k" is a proportion which indicates how much of the angle measurement should be added back into the signal. By adding a little bit of the accelerometer angle into the integrated gyro calculation each iteration, drift is corrected.z

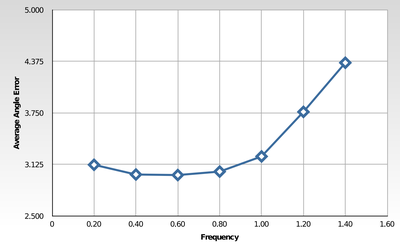
Abstract

Both a robotics project and an experiment comparing the effectiveness and tuning process of different control theory algorithms, my idea is a robot that balances on a rolling cylinder turned on its side. It is also possible to set the robot to a target velocity which it will follow. Control algorithms like the ones I have tested are used in everything from household heaters to high performance aircraft and are becoming ever more important.

Before starting, I wrote a program in Java which simulates the system and the algorithms involved in stabilizing it. This program served as a starting point illustrating that the control theory algorithms work. Building the robot has involved building several sensors and circuits from scratch including an encoder and a motor controller. In addition, I wrote the firmware for the robot myself in C and devised ways for measuring the stability of the system.

The project uses both an accelerometer and a gyro for orientation data and a wheel encoder to get accurate velocity data. These sensors together provide the microcontroller on which the controller code runs with enough data to balance effectively. In my project I will provide statistics on the performance of different control algorithms and tuning methods as they apply to my system.

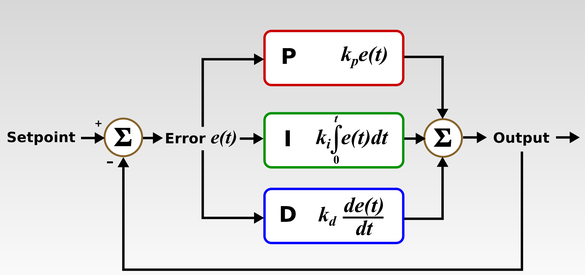
Freq. Response

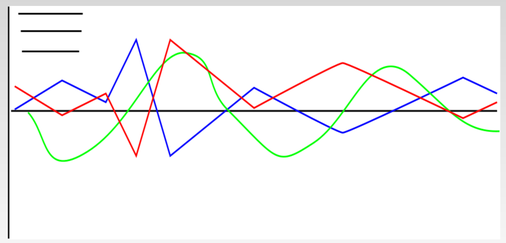


The robot attempts to maintain it's orientation as the wheel is moved back and forth at a set frequency.

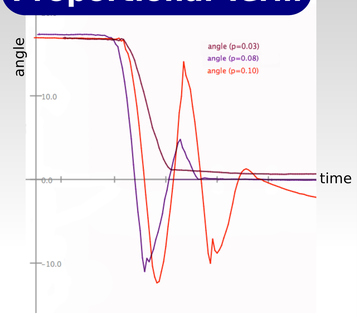
When exposed to large external oscillations, the robot is unable to compensate quickly leading to a higher average angle error. At very low frequencies, the robot does not completely reach the setpoint because it is moving so slowly.

PID Controller



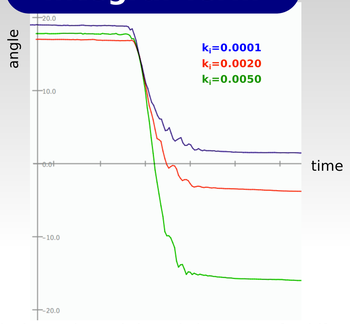


Proportional Term



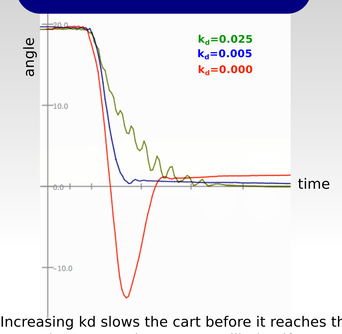
As you increase kp, the system begins to oscillate.

Integral Term



The integral term is intended to correct for drift, but if set too high it can cause the machine to drastically overshoot. This is called integral windup.

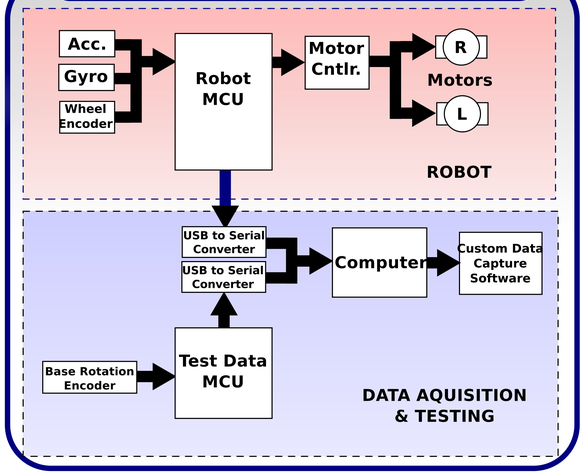
Derivative Term



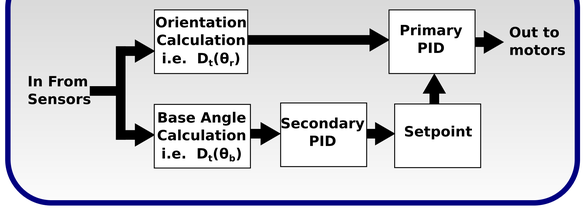
Increasing kd slows the cart before it reaches the setpoint. It can also cause oscillation if set too high.

For the above tests, the robot was offset from level on top of the wheel held stationary then turned on.

Hardware Block Diagram

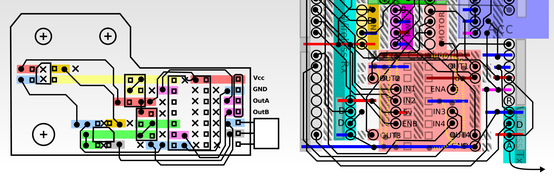


Software Block Diagram



PCB Layout

See Appendix B for Schematics



Free Body Diagram