NATCAR – Background Information

Steering Control-Loop Design
Part I

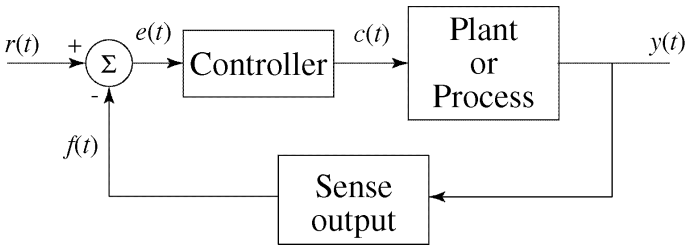
Prof. Richard Spencer

Outline

- The feedback-control problem
- PID Controllers
- Kinematic model of car
- Simulated response

The Feedback Control Problem

- Remember the basic block diagram we had before:
- The "Plant" or "Process" is your car in this project
- The objective in designing the sensing and controller is to make the car follow the course in a well-behaved manner so that it can do so quickly
- You have two loops to design: steering and speed



Controller Design

- There are many types of controllers used to tailor the response of a control system. EEC157A,B discusses this topic at length.
- We will only examine one very general form for the controller; a Proportional + Derivative + Integral (PID) controller.
- The controller output is simply given by the sum of three terms:
  - Proportional; $c(t) = k_p e(t)$
  - Integral; $c(t) = \frac{k_p}{T_i} \int_0^t e(\lambda) d\lambda$
  - Derivative; $c(t) = k_p T_d \left[de(t)/dt \right]$

PID Controller

- The controller transfer function is

$$H_c(s) = \frac{C(s)}{E(s)} = k_p\left(1 + sT_d + \frac{1}{sT_i}\right)$$

- The loop gain is then the product of the transfer functions of the controller, the plant and the sensing system.
- Let's derive a model for our "plant", which is the car. Then we'll return to examine the design of an appropriate controller. We will focus here on the steering control loop – it is BY FAR the most important (the fastest car to date had a fixed speed).

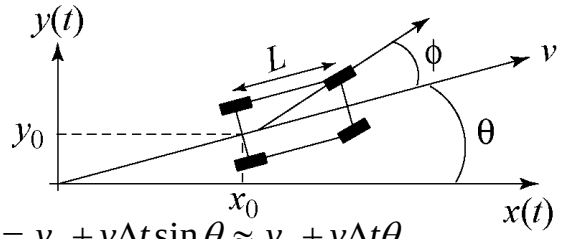
Kinematic Model for the Car

- Because the traction is very good and the tires don't deform much, we can use a kinematic model (i.e., ignoring mass and force) instead of a full dynamic model for the car.
- We will assume a constant velocity, but the results are fine so long as the velocity doesn't change too rapidly.
- When we have the model in the Laplace domain, we can then build a Matlab (or even SPICE) model for the car and simulate the performance of the car with a given controller design.

Kinematic Car Model Continued

- Assume small θ and ϕ and that the car is moving with constant velocity v. Write equations for the position of the center of the rear axle.

This model is sometimes called the bicycle model



- First: $y(\Delta t) = y_0 + v\Delta t \sin\theta \approx y_0 + v\Delta t\theta$

  then, $\dot{y} = dy/dt \approx v\theta$

x_0 and y_0 are the positions at $t = 0$

- Second: $x(\Delta t) = x_0 + v\Delta t \cos\theta \approx x_0 + v\Delta t$

  so, $\dot{x} = dx/dt \approx v$

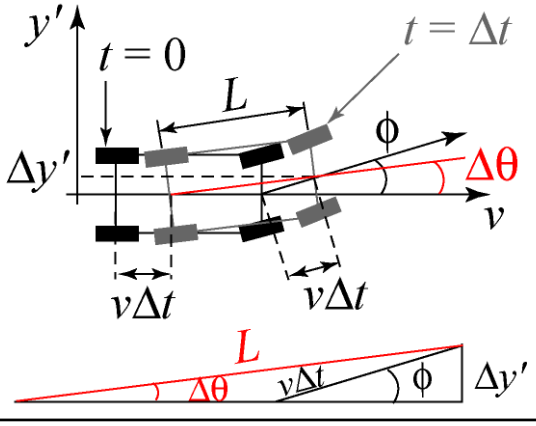
Use New Axes in Line with the Car

Assume both the back and front wheels move in the direction they are pointing (can't be true). We want to find the change in car direction, $\Delta\theta$. Define a new axis, y', perpendicular to v

$$\Delta\theta = \sin^{-1}\left(\frac{\Delta y'}{L}\right) \approx \frac{\Delta y'}{L}$$

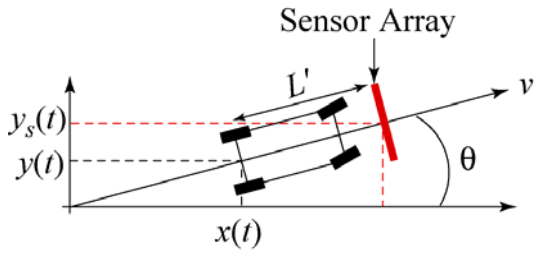
$$\Delta y' = v\Delta t \sin\phi \approx v\Delta t\phi$$

$$\Delta\theta \approx \frac{v\Delta t\phi}{L}$$



Kinematic Car Model Continued

- Now write an equation for the y-component of the sensor position (taken to be at the center of the sensor array)



- Still assuming small θ, we have

$$y_s(\Delta t) = y(\Delta t) + L'\sin\theta$$
$$\approx y(\Delta t) + L'\theta$$

- So, $\dot{y}_s(\Delta t) \approx \dot{y}(\Delta t) + L'\dot{\theta} \approx v\theta + L'\dot{\theta}$

Put the Equations Together

We have $\dot{y}_s \approx v\theta + L'\dot{\theta}$ and $\Delta\theta \approx \dfrac{v\Delta t\phi}{L}$

Therefore, $\dot{\theta} \approx \dfrac{v\phi}{L}$ and, still assuming v is constant,

$$\ddot{y}_s \approx v\dot{\theta} + L'\ddot{\theta} = \frac{v^2\phi}{L} + \frac{vL'\dot{\phi}}{L}$$

so that, in the Laplace domain we have

$$Y_s(s) \approx \left(\frac{v^2}{Ls^2} + \frac{vL'}{Ls}\right)\Phi(s) = \left(\frac{v^2}{L}\right)\left(\frac{1+sL'/v}{s^2}\right)\Phi(s)$$

Model the System

The transfer function for the car is

$$H_{car}(s) = \frac{Y_s(s)}{\Phi(s)} = \left(\frac{v^2}{L}\right)\left(\frac{1+sL'/v}{s^2}\right)$$

Let's assume that the servo is linear and model it by a single-pole LP transfer function

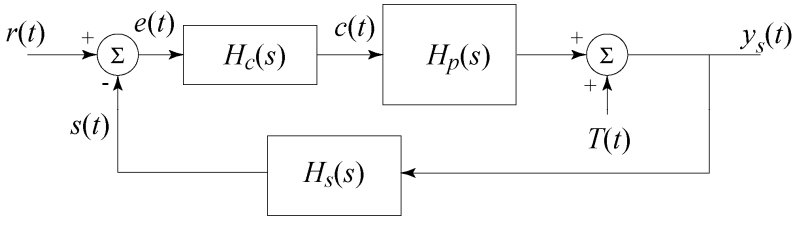
$$H_\phi(s) = \frac{\Phi(s)}{C(s)} = \frac{k_\phi}{1+s/\omega_\phi}$$

The overall model for our plant is then given by

$$H_p(s) = \frac{Y_s(s)}{C(s)} = \frac{k_\phi}{1+s/\omega_\phi}\left(\frac{v^2}{L}\right)\left(\frac{1+sL'/v}{s^2}\right)$$

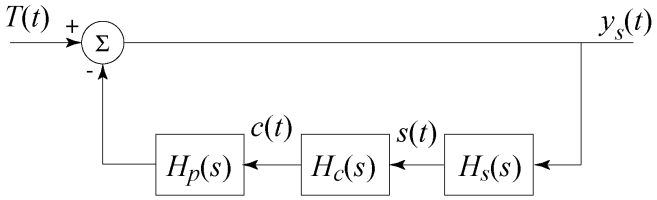
System Model Continued

- Assume the sensors provide an output proportional to the deviation from the track so that $H_s(s) = k_s$
- We now have a complete block diagram model of our control system.
  - $y_s(t)$ is the sensor array deviation from the track (in feet)
  - $r(t)$ is the reference input, which is zero
  - $T(t)$ is the racetrack; it provides deviations that the control loop must respond to.



System Model Continued

- Since $r(t) = 0$, we can redraw the loop with the track as the input.



- Using the equations we've derived, we have

$$L(s) = H_s(s)H_c(s)H_p(s)$$

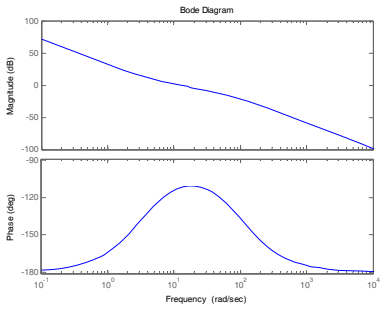
$$= k_s k_p \left(1 + sT_d + \frac{1}{sT_i}\right) \frac{k_\phi}{1 + s/\omega_\phi} \left(\frac{v^2}{L}\right) \left(\frac{1 + sL'/v}{s^2}\right)$$

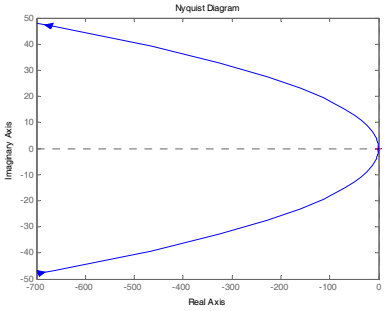
System Model; Example Numbers

- Assume the following:
  - The sensor gain is one (V/ft)
  - The servo gain is $\pi/2 = 1.57$ (rad/V)
  - The servo bandwidth is 100 rad/sec
  - The wheelbase is $L = 1$ foot and $L' = 1.5$ feet
  - The velocity is 5 ft/sec
  - The controller is proportional only with a gain of one (V/V)
- Let's use Matlab to generate the Bode plots, a Nyquist plot of L, and the step response for the closed-loop system with a proportional-only controller

Bode and Nyquist Plots; Proportional Control

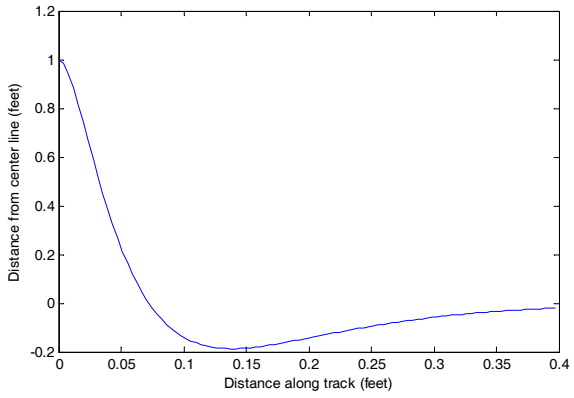
The Bode and Nyquist plots are shown below. The system is obviously stable (the Nyquist plot encircles -1+*j*0, but in the CCW direction), which can also be seen in the step response on the next slide.





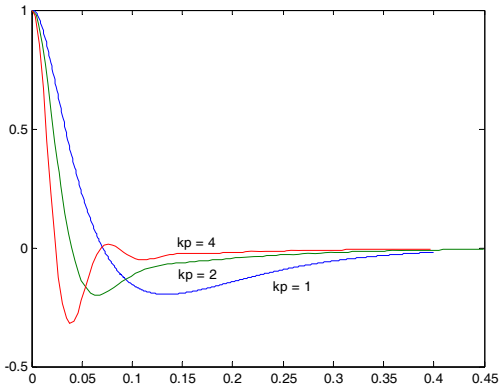
Closed-Loop Step Response; Proportional Control

The closed-loop step response shows the loop is mathematically stable, but note that your car probably can't turn this fast, which might cause you to lose the track



Closed-Loop Step Response; Proportional Control

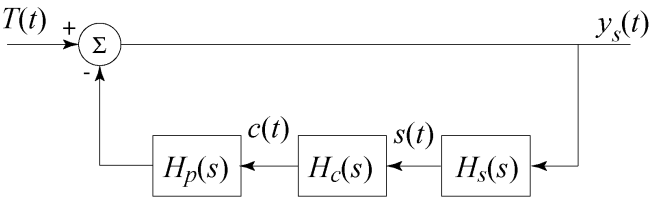
Also, notice that different proportional control-loop gains have a significant effect on the response:



Obviously – if the loop gain is too high, the system is practically unstable

Re-Consider System Model

- Let's reconsider our system model and see what we are ignoring
  - The servo has a slew-rate limitation

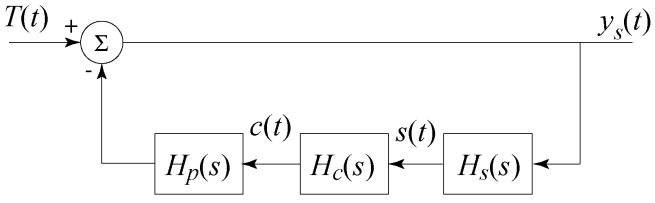


Servo Slew-Rate Limitation

- Slewing is a nonlinear phenomenon, so we can't rigorously model it in the Laplace domain
- Nevertheless, servo slewing delays the steering response, and adding delay into the loop makes it less stable
- In addition, the slewing prevents the car from turning as quickly as our simulations sometimes show, which means that your car may run off the track and lose the signal entirely even if the control loop is theoretically stable

Re-Consider System Model

- Let's reconsider our system model and see what we are ignoring
  - The servo has a slew-rate limitation
  - The sensing transfer function may be nonlinear

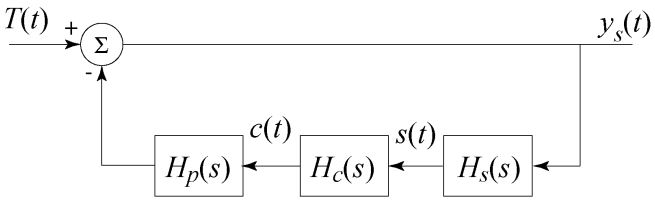


Nonlinear Sensing

- If the sensing transfer function is monotonic and reasonably smooth, then nonlinearity does not significantly affect stability
  - if it had steps, that might cause problems because it would cause rapid changes in the steering and would drive the servo into a slewing mode more often
  - if it was non-monotonic, then negative feedback would become positive feedback wherever the slope of the sensing transfer function changed sign and the loop could enter a local limit cycle

Re-Consider System Model

- Let's reconsider our system model and see what we are ignoring
  - The servo has a slew-rate limitation
  - The sensing transfer function may be nonlinear
  - The steering servo is only controlled at discrete times, in other words, the steering angle is actually sampled and held (about every 3 ms)

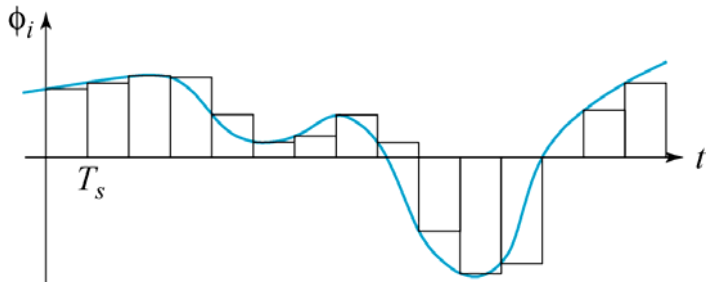


Sampling

- The control signal generated by your circuits, $c(t)$, is continuous-time, but it controls a PW modulator and the value is only transferred to the steering servo once every period (about once every 3ms)
- Therefore, the servo functions as a sample-and-hold circuit; that is, it samples $c(t)$ once a period and then holds that value until the next sample arrives
- Let's consider this intuitively for a moment …

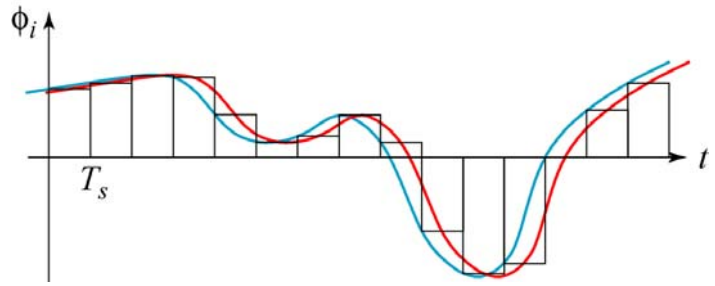
Sampling Continued

- If the time samples are held, you get a waveform that looks like this



Sampling Continued

- If the time samples are held, you get a waveform that looks like this
- Notice that if you reconstruct the original from the sampled-and-held version, it appears delayed!



Sampling Continued

- Adding delay into the loop is equivalent to adding in extra phase, so it tends to make the loop less stable