Lecture 6: Control Problems and Solutions

CS 344R: Robotics

Benjamin Kuipers

But First, Assignment 1: **Followers**

- A follower is a control law where the robot moves forward while keeping some error term small.
  - Open-space follower
  - Wall follower
  - Coastal navigator
  - Color follower
- Due October 4.

Control Laws Have Conditions

- Each control law includes:
  - A *trigger*: Is this law applicable?
  - The law itself: $\mathbf{u} = H_i(\mathbf{y})$
  - A *termination condition*: Should the law stop?

Open-Space Follower

- Move in the direction of large amounts of open space.
- Wiggle as needed to avoid specular reflections.
- Turn away from obstacles.
- Turn or back out of blind alleys.
- Try to be elegant and robust.

Wall Follower

- Detect and follow right or left wall.
- Implement the PD control law taught in class.
- Respond to step-changes in environment or set-point.
- Tune to avoid large oscillations.
- Terminate on obstacle or wall vanishing.

Coastal Navigator

- Join wall-followers to follow a complex "coastline"
- When a wall-follower terminates, make the appropriate turn, detect a new wall, and continue.
- Inside and outside corners, 90 and 180 deg.
- Orbit a box, a simple room, or the desks!

Color Follower

- Move to keep a desired color centered in the camera image.
- Train a color region from a given image.
- Follow an orange ball on a string, or a brightly-colored T-shirt.
- How quickly can the robot respond?

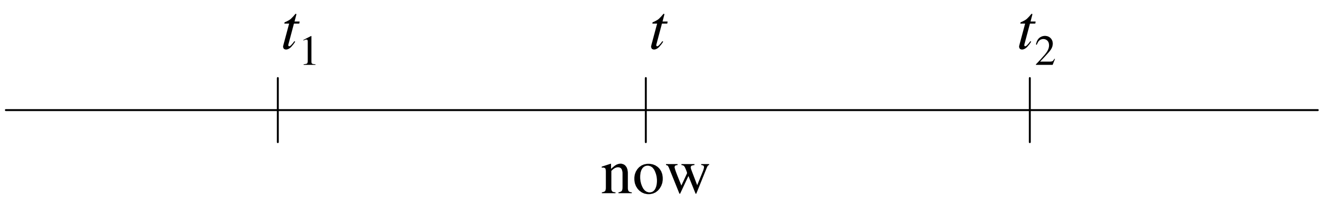
Problems and Solutions

- Time delay
- Static friction
- Pulse-width modulation
- Integrator wind-up
- Chattering
- Saturation, dead-zones, backlash
- Parameter drift

Unmodeled Effects

- Every controller depends on its simplified model of the world.
  - Every model omits almost everything.
- If unmodeled effects become significant, the controller’s model is wrong,
  - so its actions could be seriously wrong.
- Most controllers need special case checks.
  - Sometimes it needs a more sophisticated model.

Time Delay



- At time t,
  - Sensor data tells us about the world at $t_1 < t$.
  - Motor commands take effect at time $t_2 > t$.
  - The lag is $dt = t_2 - t_1$.
- To compensate for lag time,
  - Predict future sensor value at t_2.
  - Specify motor command for time t_2.

Predicting Future Sensor Values

- Later, *observers* will help us make better predictions.
- Now, use a simple prediction method:
  - If sensor s is changing at rate ds/dt,
  - At time t, we get $s(t_1)$, where $t_1 < t$,
  - Estimate $s(t_2) = s(t_1) + ds/dt * (t_2 - t_1)$.
- Use $s(t_2)$ to determine motor signal $u(t)$ that will take effect at t_2.
  - "Smith predictor"

Static Friction ("Stiction")

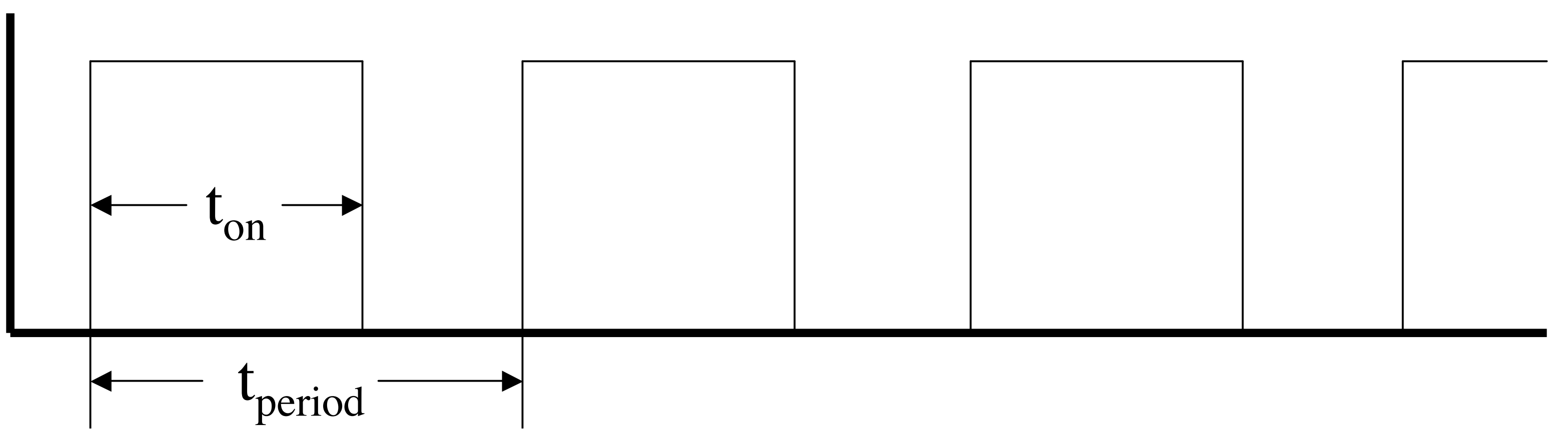
- Friction forces oppose the direction of motion.
- We've seen damping friction: $F_d = -f(v)$
- Coulomb ("sliding") friction is a constant F_c depending on force against the surface.
  - When there is motion, $F_c = \eta$
  - When there is no motion, $F_c = \eta + \varepsilon$
- Extra force is needed to unstick an object and get motion started.

Why is Stiction Bad?

- Non-zero steady-state error.
  - (runaway pendulum story)
- Stalled motors draw high current.
  - Running motor converts current to motion.
  - Stalled motor converts *more* current to heat.
- Whining from pulse-width modulation.
  - Mechanical parts bending at pulse frequency.

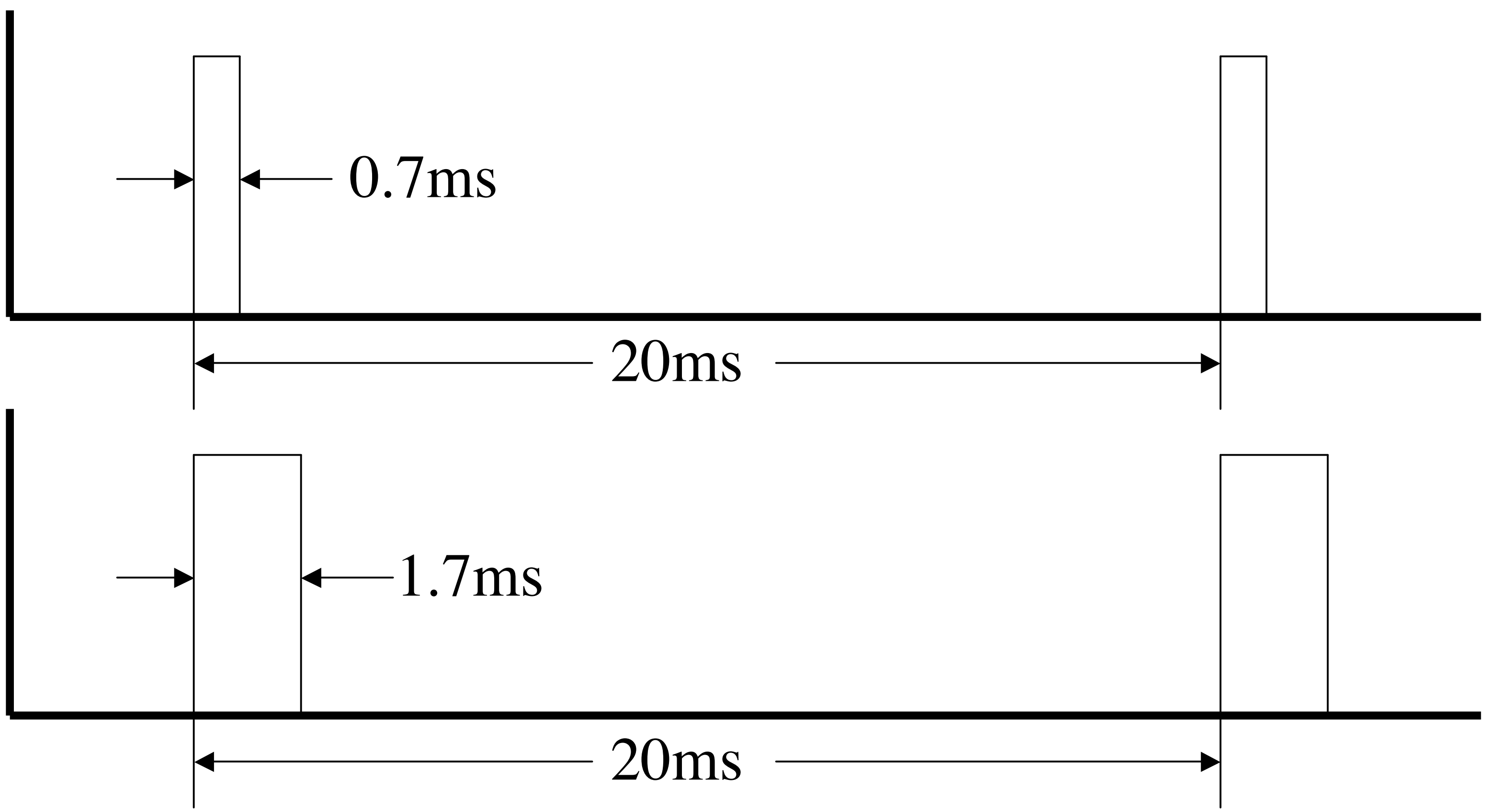
Pulse-Width Modulation

- A digital system works at 0 and 5 volts.
  - Analog systems want to output control signals over a continuous range.
  - How can we do it?
- Switch very fast between 0 and 5 volts.
  - Control the average voltage over time.
- Pulse-width ratio $= t_{on}/t_{period}$. (30-50 μsec)



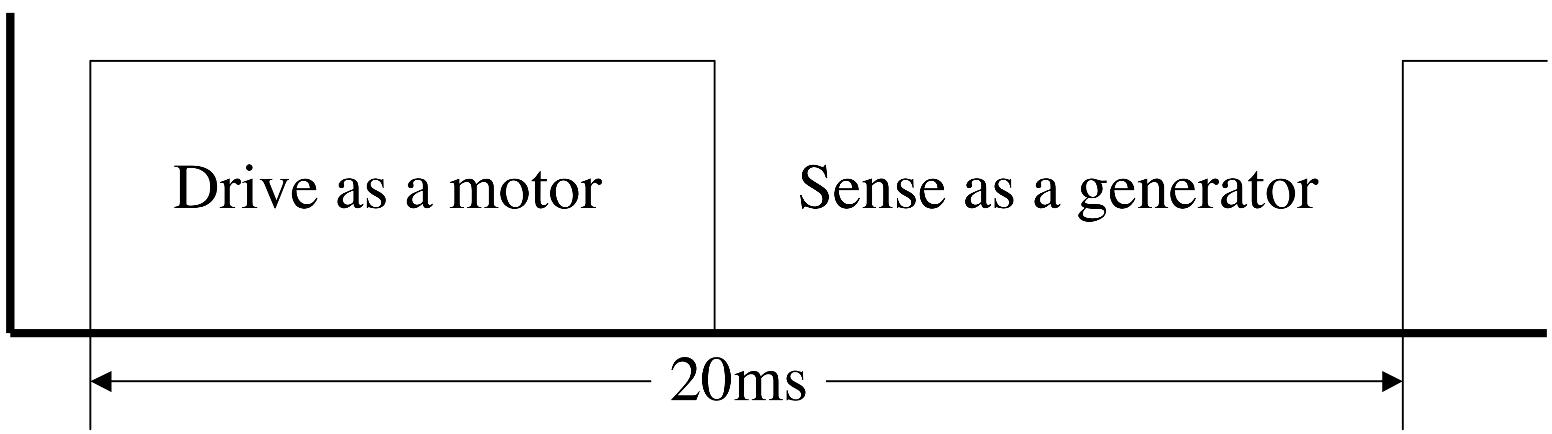
Pulse-Code Modulated Signal

- Some devices are controlled by the length of a pulse-code signal.
  - Position servo-motors, for example.

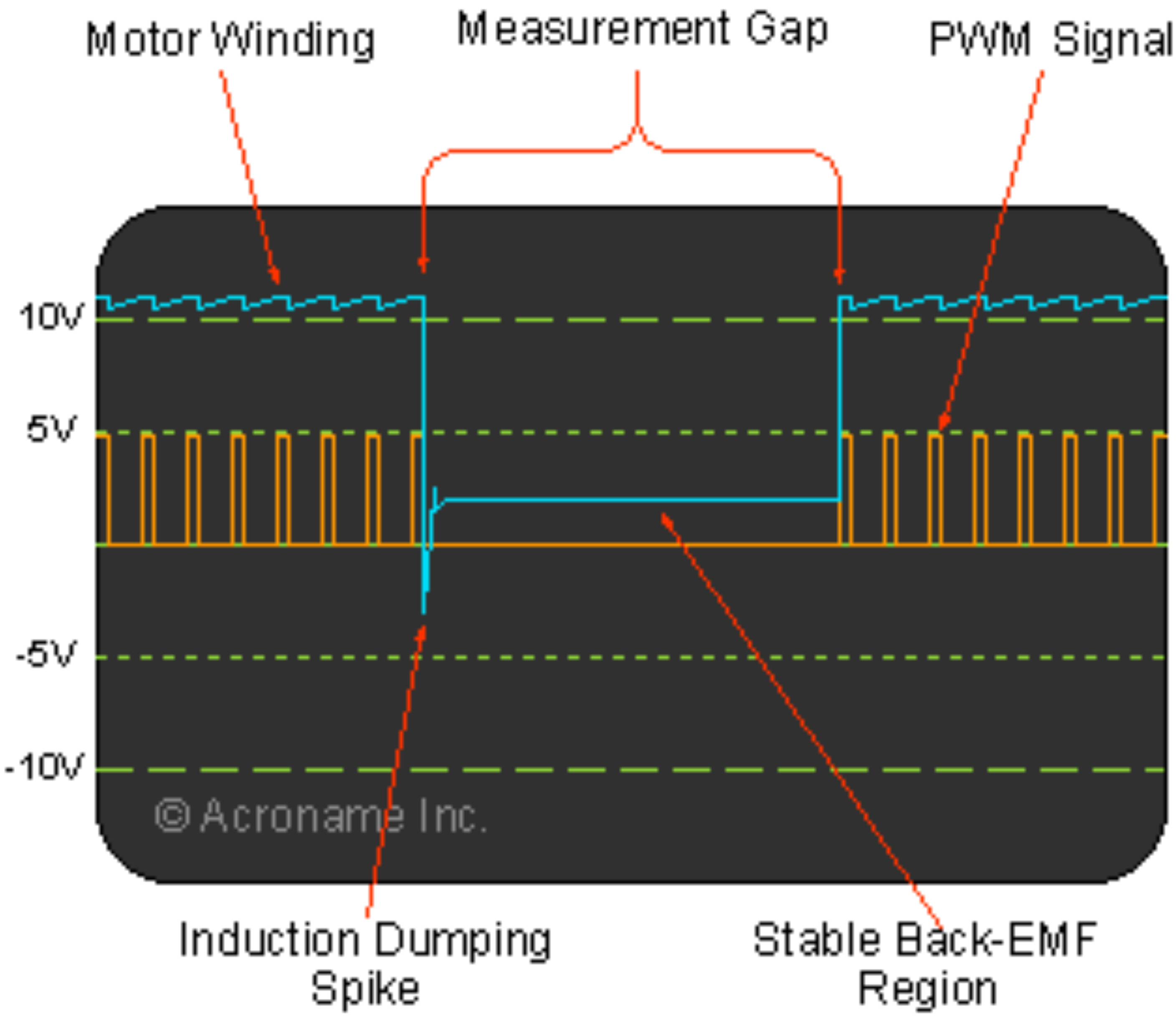


Back EMF Motor Control

- Motor torque is proportional to current.
- Generator voltage is proportional to velocity.
- The same physical device can be either a motor or a generator.
- Switch back and forth quickly, as in PWM.



Back EMF Motor Control



Integrator Wind-Up

- Suppose we have a PI controller

$$u(t) = -k_P\, e(t) - k_I \int_0^t e\, dt + u_b$$

- Motion might be blocked, but the integral is winding up more and more control action.

$$u(t) = -k_P\, e(t) + u_b$$

$$\dot{u}_b(t) = -k_I\, e(t)$$

- Reset the integrator on significant events.

Chattering

- Changing modes rapidly and continually.
  - Bang-Bang controller with thresholds set too close to each other.
  - Integrator wind-up due to stiction near the setpoint, causing jerk, overshoot, and repeat.

Dead Zone

- A region where controller output does not affect the state of the system.
  - A system caught by static friction.
  - Cart-pole system when the pendulum is horizontal.
  - Cruise control when the car is stopped.

- Integral control and dead zones can combine to cause integrator wind-up problems.

Saturation

- Control actions cannot grow indefinitely.
  - There is a maximum possible output.
  - Physical systems are necessarily nonlinear.
- It might be nice to have bounded error by having infinite response.
  - But it doesn't happen in the real world.

Backlash

- Real gears are not perfect connections.
  - There is space between the teeth.

- On reversing direction, there is a short time when the input gear is turning, but the output gear is not.

Parameter Drift

- Hidden parameters can change the behavior of the robot, for no obvious reason.
  - Performance depends on battery voltage.
  - Repeated discharge/charge cycles age the battery.
- A controller may compensate for small parameter drift until it passes a threshold.
  - Then a problem suddenly appears.
  - Controlled systems make problems harder to find

Unmodeled Effects

- Every controller depends on its simplified model of the world.
  - Every model omits almost everything.
- If unmodeled effects become significant, the controller's model is wrong,
  - so its actions could be seriously wrong.
- Most controllers need special case checks.
  - Sometimes it needs a more sophisticated model.