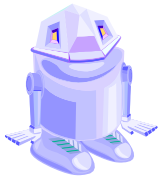


ECE497: Introduction to Mobile Robotics Lecture 2

Dr. Carlotta A. Berry

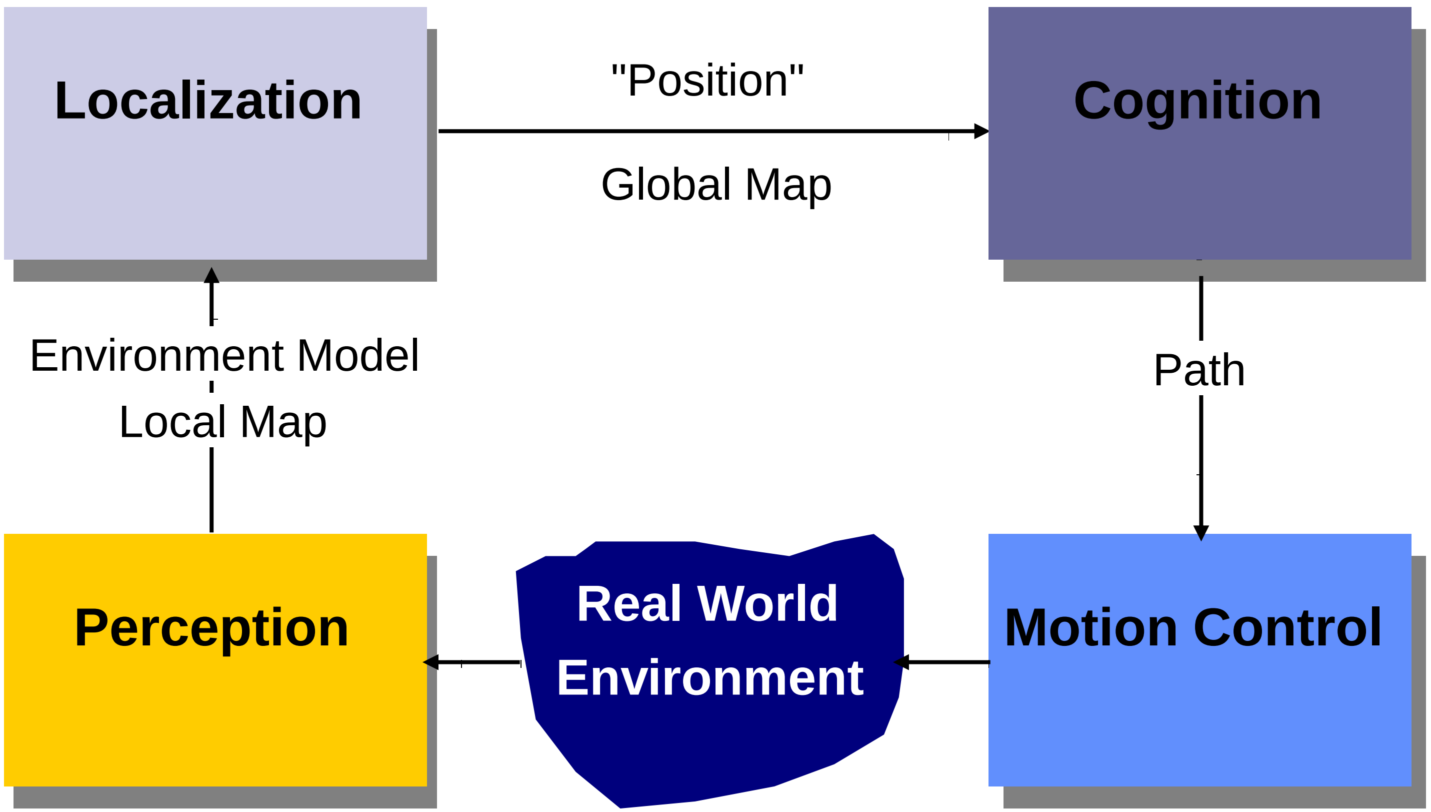
Spring 06 - 07

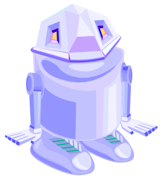
Quote of the Week

"In the fifties, it was predicted that in 5 years robots would be everywhere.
In the sixties, it was predicted that in 10 years robots would be everywhere.
In the seventies, it was predicted that in 20 years robots would be everywhere.
In the eighties, it was predicted that in 40 years robots would be everywhere…"

Marvin Minsky

Locomotion Concepts: Navigation problem





LOCOMOTION

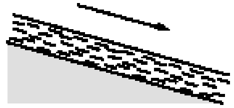
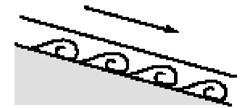
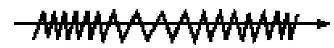
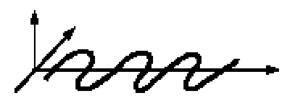
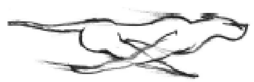
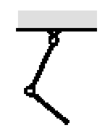
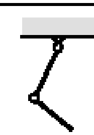
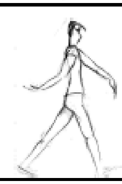
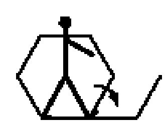
Chapter 2

Locomotion (2.1)

- ***<u>Locomotion</u>*** mechanisms enable robot’s to move unbounded in the environment

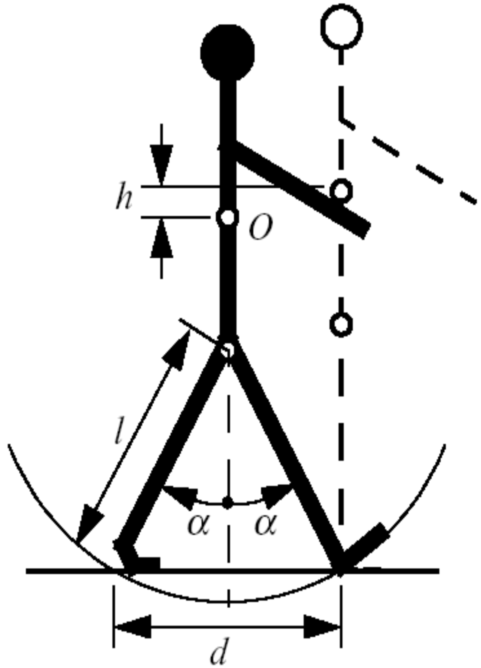
- In research robots, some locomotion mechanisms simulate biological systems (i.e. walk, jump, run, slide, skate, swim, fly, roll)

Biological Inspirations for Locomotion

Type of motion	Resistance to motion	Basic kinematics of motion
Flow in a Channel 	Hydrodynamic forces	Eddies 
Crawl 	Friction forces	 Longitudinal vibration
Sliding 	Friction forces	 Transverse vibration
Running 	Loss of kinetic energy	Oscillatory movement of a multi-link pendulum 
Jumping 	Loss of kinetic energy	Oscillatory movement of a multi-link pendulum 
Walking 	Gravitational forces	Rolling of a polygon (see figure 2.2) 

Locomotion Concepts (2.1)

- Concepts found in nature (biological systems)
  - Move better through a wide variety of harsh environments
  - However these mechanisms are difficult to imitate technically
- Therefore, most technical systems use wheeled mechanisms or a small number of articulated legs
- Rolling is
  - most efficient on hard surfaces
  - simple to implement and well suited to flat ground
- Wheeled motion versus legged motion has
  - A smaller number of actuators
  - Less structural complexity
  - Less control expense
  - More energy efficient on hard surfaces
- The movement of a walking biped is close to rolling
  - Wheeled locomotion is a human invention but bipedal walking can be approximated by a rolling polygon with sides equal in length d to the span of the step (Figure 2.2)
  - The smaller the step gets, the more the polygon tends to a circle (wheel)



Key issues for Locomotion

- **Locomotion** is the complement of manipulation
  - In *locomotion*, the environment is fixed and the robot moves by imparting force to the environment

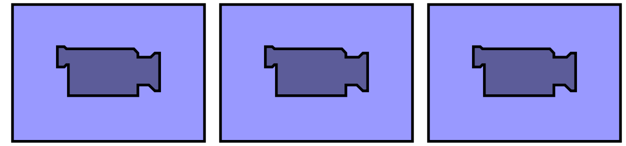
  In *manipulation*, the robot arm is fixed but moves objects by imparting force to the environment

- Actuators that generate forces and mechanisms that implement desired kinematic and dynamic properties are the scientific basis for locomotion and manipulation

Key issues for Locomotion (2.3)

- stability
  - number of contact points
  - center of gravity
  - static/dynamic stabilization
  - inclination of terrain
- characteristics of contact
  - contact point or contact area
  - angle of contact
  - friction
- type of environment
  - structure
  - medium (water, air, soft or hard ground)

Stability (2.3)



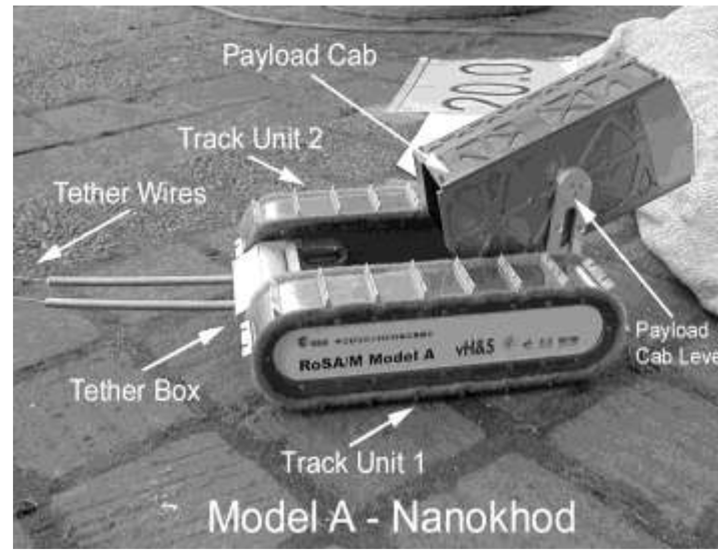
- Robots need to be stable to get their job done
- Stability can be
  - Static: the robot can stand still without falling over
  - Dynamic: the body must actively balance or move to remain stable
- Static stability is achieved through the mechanical design of the robot
- Dynamic stability is achieved through control

Wheeled Mobile Robots (2.3)

- Stability is not typically an issue in wheeled robot designs because all wheels are on the ground at the same time
- Three wheels are sufficient to guarantee stable balance
- More than three wheels require a suspension system
- Wheeled robot research focuses on
  - Sufficient traction over various surfaces
  - Maintain stability over all desired terrains
  - Maneuverability such as turning sharp enough
  - Control over the robot velocity without overshoot or collision
  - Selection of wheels depend on the application

Tracked Mobile Robots (2.3.2.3)

- Robots that make use of tread have larger ground contact and significantly improved maneuverability and traction
- However, changing the orientation requires a slip/skid turn
- A slip/skid turn is where the robot spins wheels facing the same direction at different speeds or in opposite directions
- The disadvantage is that the slip/skid steering makes it hard to determine the exact center of rotation of the robot. This makes it difficult to predict the exact change in position and orientation depending on ground friction
- Dead reckoning is also highly inaccurate on these robots
- A slip/skid turn on a high friction surface will overcome the toque of the motors

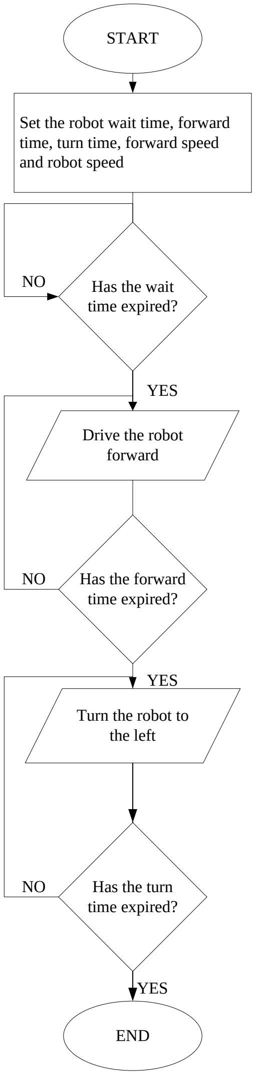


The NANOKHOD II, developed by von Hoerner & Sulger GmbH and Max Planck Institute, Mainz for European Space Agency (ESA) will probably go to Mars

Odometry
Feedback Control

Pre-Lab 1b (Due Thursday, 3/15/07)

Modify the Circle.c and Square.c programs to include feedback from the encoder. Create 2 flowcharts for the modified programs. Generate the figures electronically using Microsoft Visio, Word or some other drawing software. The flowcharts should be detailed enough to help you modify your code and provide the reader with a logical flow of execution.



Odometry

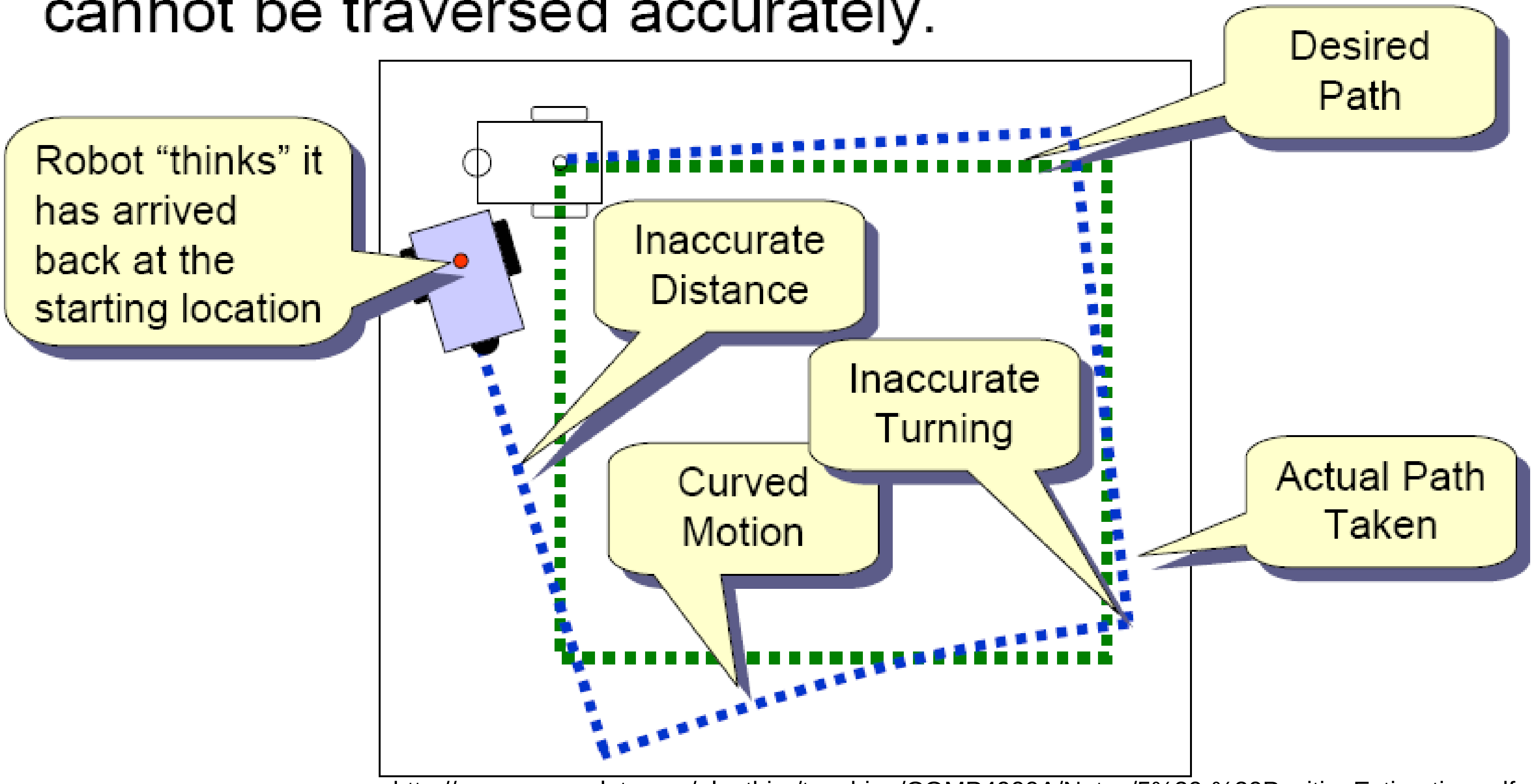
- Odometry is a means of implementing Dead Reckoning
- A way of determining a robot's position based upon previous known position information given a specific course heading and velocity
- Periodically requires error measurement to be 'fixed' or reset
- Meant for short distance measurements

Odometry errors

- Imprecise measurements
  - Discrepancy with actual speed and turn angles
- Inaccurate control model
  - Wheels are not perfectly aligned or do not make contact at a single point
- Immeasurable physical characteristics
  - Friction
  - Wobbling wheels
  - Surface is not perfectly smooth and hard
  - Sliding

Dead Reckoning

- As a result of these error factors, a simple path cannot be traversed accurately.



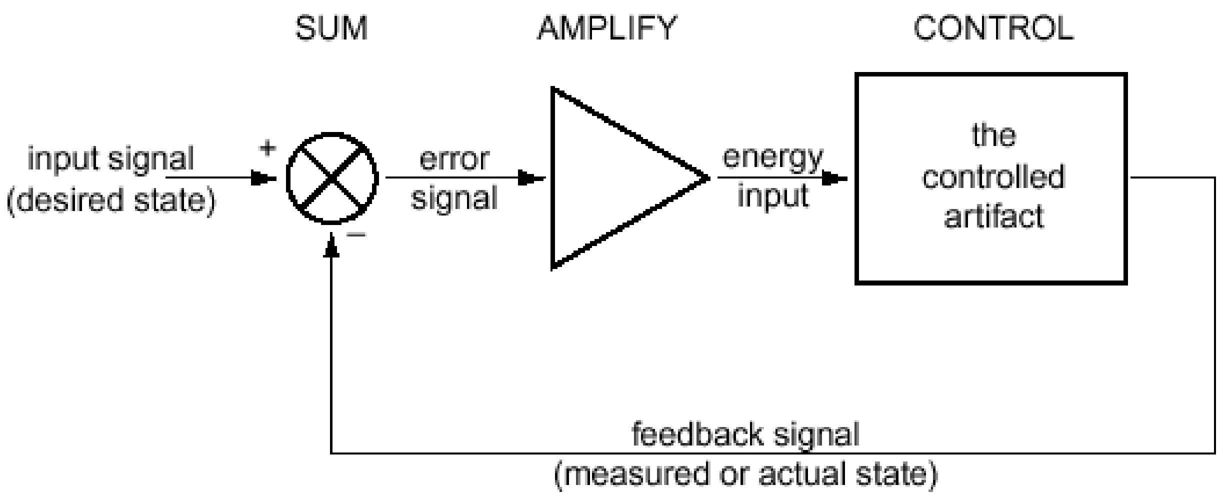
http://www.scs.carleton.ca/~lanthier/teaching/COMP4900A/Notes/5%20-%20PositionEstimation.pdf

Open Loop Control

- *Open Loop Control* does not use sensory feedback, and the robot state is not fed back into the system
- *Feed-forward control*
  - The command signal is a function of some parameters measured in advance
  - i.e., battery strength measurement could be used to "predict" how much time is needed for the turn
  - Still open loop control, but a computation is made to make the control more accurate
- Feed-forward systems are effective only if
  - They are well calibrated
  - The environment is predictable and does not change

Feedback Control

Feedback (closed loop) control is when a system achieves and maintains a *desired state* by continuously comparing its current and desired states, then adjusting the current state to minimize the difference



Error

- *Error* is the the difference in the current state and desired state of the system
  - The controller has to minimize the error at all times
- *Direction of error:*
  - Which way to go to minimize the error
- *Magnitude of error:*
  - The distance to the goal state
- *Zero/non-zero error:*
  - Tells whether there is an error or not
  - The least information we could have
- Control is much easier if we know both magnitude and direction

Feedback Control: Wall following

- Use feedback to design a wall following robot
- What sensors to use, what info will they provide?
  - Tactile: provides contact with the wall and the least amount of information (binary)
  - Infrared, sonar, laser : would provide distance to wall (analog)
- Control algorithm

If distance-to-wall is correct, **then** keep moving

If distance-to-wall is too large

  then turn toward the wall

else if distance-to-wall is too small

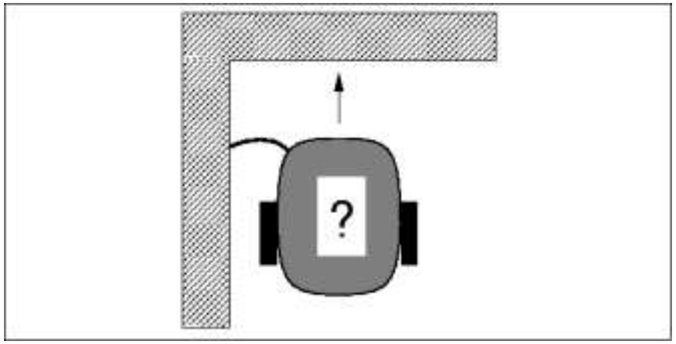
  then turn away from the wall

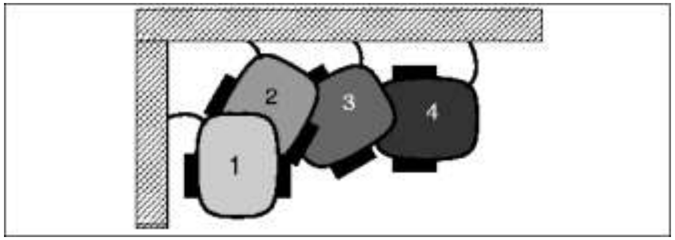
Feedback Control: Overshoot and Oscillations

- *Overshoot* is when the robot goes beyond its set point (position) or changes direction before stabilizing on it
- The robot *oscillates* around the optimal distance from the wall, getting either too close or too far
- In general, the behavior of a feedback system oscillates around the desired set point (position)
- To decreasing oscillations:
  - Adjust the turn angle
  - Use a range instead of a fixed distance as the goal state

Wall Following

- Negotiating a corner
  - Make little turns, drive straight ahead, detect the wall, back up, repeat
  - The disadvantage is that this method is time consuming and produces jerky movements
- Alternative:
  - Execute a turn command that was timed to accomplish a ninety degree rotation (open loop control)
  - Works reliably only when the robot is very predictable (battery strength, traction on the surface, and friction)





http://www.cse.unr.edu/~monica/Courses/CPE470-670/cpe470-670-sp05.html

Types of Feedback Control

There are 3 types of basic feedback controllers

- P: proportional control
- PD: proportional derivative control
- PID: proportional integral derivative control

Feedback Control: Proportional Control

- The response of the system is proportional to the amount of the error
- The output of the controller is proportional to the input i (speed or velocity error):

$$\text{output} = K_p * \text{error input}$$

- K_p is a proportionality constant (gain)
- The controller generates a stronger response the farther away the system is from the goal state
- The robot will turn sharply toward the wall if it is really far from it
- The robot will turn gently toward the wall if it is slightly away from it

Feedback Control; Determining Controller Gains

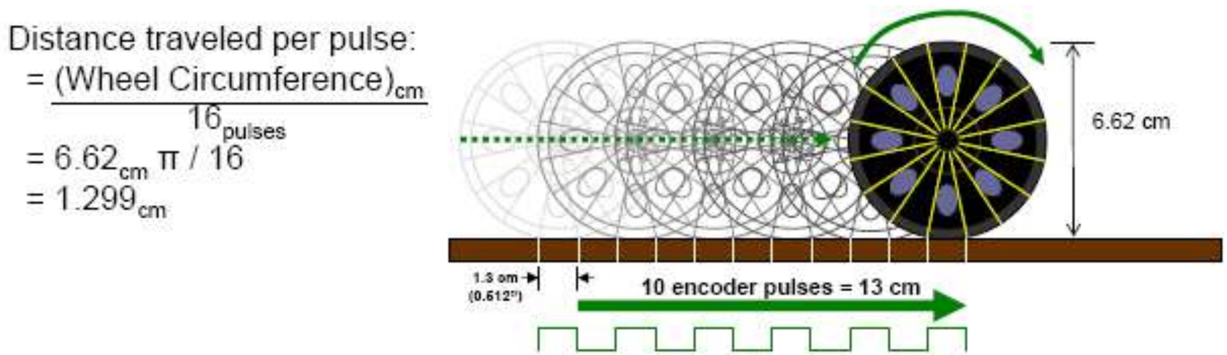
- How do we determine the controller gain?
- Empirically (trial and error):
  - require that the system be tested extensively
- Analytically (mathematics):
  - require that the system be well understood and characterized mathematically
- Automatically
  - by trying different values at run-time

Feedback Control: Gains and Oscillations

- Incorrect controller gains will cause the system to undershoot or overshoot the desired state and cause oscillations
- Gain values determine if:
  - The system will keep oscillating or increase in oscillations
  - The system will stabilize
- Damping: process of systematically decreasing oscillations
- A system is **properly damped** if it does not oscillate out of control or has no oscillations at all

Encoders: Measuring wheel distance

- Relative position estimation is dependent on the measurement of the robot's velocity
- Optical encoders can be used on each wheel to measure angular position and or velocity
- If the robot moves straight ahead, you can count encoder pulses to determine its new location
- If 10 encoder pulses = 13 cm then distance can have an error up to 1.3 cm
  - This can be serious for short distances
  - Accumulated error can cause problems

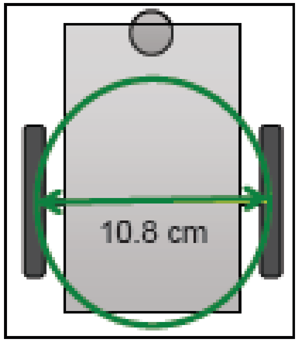


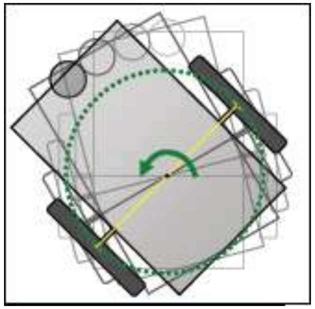
http://www.scs.carleton.ca/~lanthier/teaching/COMP4900A/Notes/5%20-%20PositionEstimation.pdf

Encoders: Measuring spin angle

- Assuming the robot turns around it’s center ($v_1 = -v_2$)
- If the robot has a diameter of 10.3 cm, the circumference is $\pi d = \pi(10.8) = 33.93$ cm
- If each encoder pulse indicates a travel distance of 1.3 cm

$$(1.3 \text{ cm/pulse})/(33.93 \text{ cm})(360°) = 13.79°/\text{pulse}$$



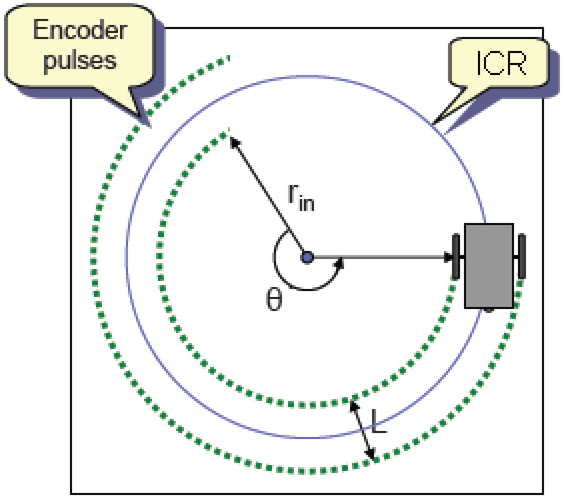


http://www.scs.carleton.ca/~lanthier/teaching/COMP4900A/Notes/5%20-%20PositionEstimation.pdf

Encoders; Measuring turn angle

When the wheel velocities are not equal

- Each wheel traces out a circle with a different circumference
- The diameter of the robot L = 10.8 cm
- Length of the inner arch (left wheel) → $s_2 = r_{in}\,\theta$
- Length of the outer arch (right wheel) → $s_1 = (r_{in} + L)\theta = s_2 + L\,\theta$
- So $\theta = (s_1 - s_2)/L$
- The length of the arcs can be described in terms of the encoder pulses for the right (p_1) and left (p_2) wheels.



Encoders: Measuring turn angle, cont.

- the difference in the encoder pulses, $p_{\Delta} = p_1 - p_2$
- Using this to find the change in the angle

$$\theta\Delta = (p_{\Delta})(1.3 \text{ cm/pulse})/10.8 \text{ cm}$$

$$= 0.12037\, p_{\Delta} \text{ radians} = (6.897)\, p_{\Delta}{}^{\circ}$$

- Note that when the pulses of both wheels are equal (i.e. same velocity), $\theta\Delta = 6.897\, p_{\Delta}{}^{\circ} = (6.897)(2p_1) = 13.79\, p_1{}^{\circ}$
- To find the (x,y) position of the robot substitute encoder pulses for velocities in the forward kinematic equations

Proportional Control for Rotation Position of Wheel

- In order to control rotational position of the Traxster wheel adjust the motor speed
  - If the desired position = 100 encoder pulses
  - The controller will vary power to the motor based upon the error
- The encoder will keep track of the shaft position

Proportional Control for Rotation Position of Wheel

- Simple Proportional controller:
  - Command = 100 – encoder-counts
- Initially, the error is 100
  - The motor turns on full speed
  - As it starts going, the error becomes progressively smaller
- Halfway, at position 50, the error is only 50
  - at that point the motor goes at 50% of full power
- When it arrives at the intended position of 100, the error is zero
  - the motor is off

Proportional Control for Rotation Position of Wheel

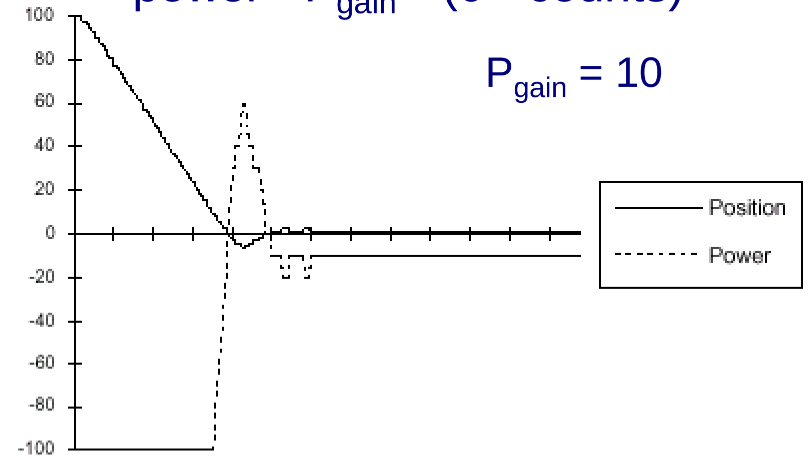
- Proportional gain:

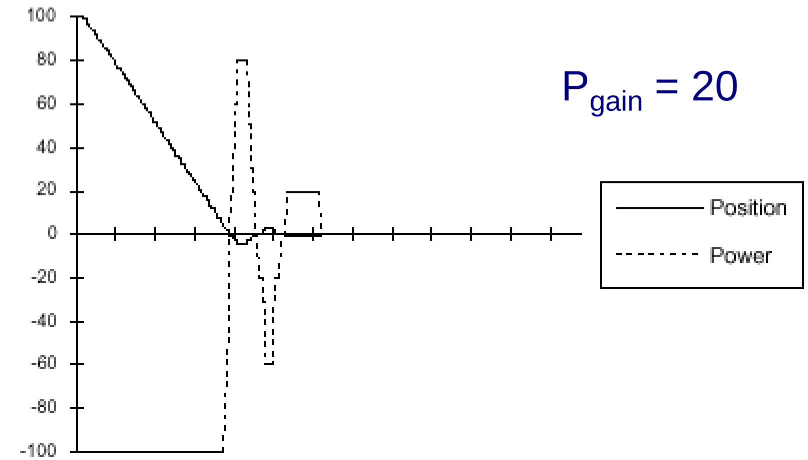
  Command = ***5*** *(100 – encoder-counts)
- The response will be tighter
- There will be a more aggressive resistance to being turned away from the set point
- The wheel reaches the set point position faster
- The system is *critically damped*

Proportional Control

- P_{gain}=10:
  - System overshot the zero point, and had to turn around
  - Offset Error: System did not stabilize at the goal
  - Power command too small to activate the motor
- P_{gain}=20: should ameliorate the offset problem
  - Offset error is solved
  - Oscillation: the system overshoots three times

power= P_{gain} * (0 - counts)





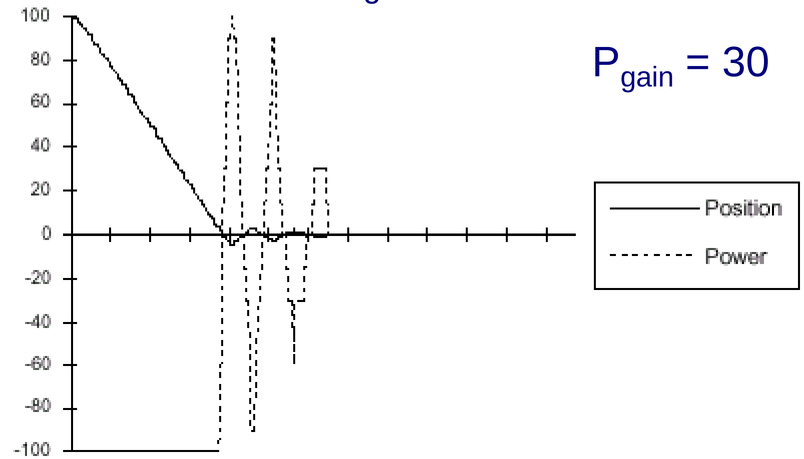
http://www.cse.unr.edu/~monica/Courses/CPE470-670/cpe470-670-sp05.html

Proportional Control

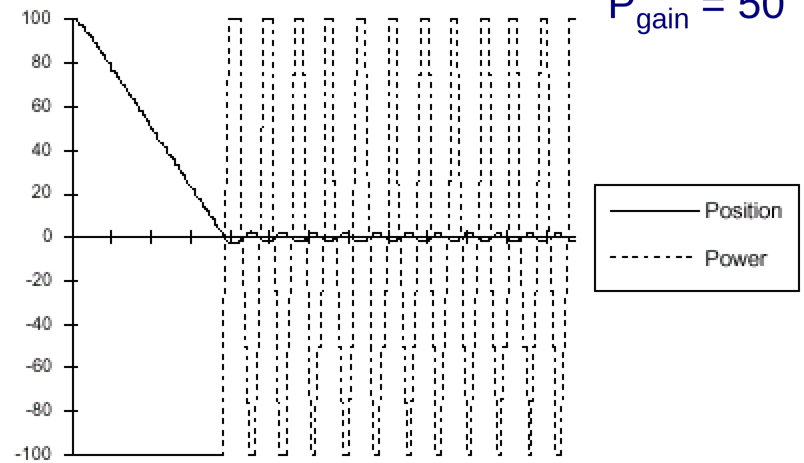
- P_{gain}=30:
  - Oscillation problem is more pronounced;
  - The response is *underdamped*
- P_{gain}=50:
  - Oscillation behavior has taken over
  - System cannot stabilize at the set point
  - A small error generates a power command that moves the system across the set point
  - The system is *undamped*

power= P_{gain} * (0 - counts)

P_{gain} = 30



P_{gain} = 50



http://www.cse.unr.edu/~monica/Courses/CPE470-670/cpe470-670-sp05.html

Derivative Control

- Setting gains is difficult, and simply increasing the *proportional gain* does not remove oscillations
- The system needs to be controlled differently when it is close to the desired state and when it is far from it
- The momentum of the correction carries the system beyond the desired state, and causes oscillations

$$\text{Momentum} = \text{mass} * \text{velocity}$$

- Solution: correct the momentum as the system approaches the desired state

Controlling Velocity

- Momentum and velocity are directly proportional therefore we can control the momentum by controlling velocity
- As the system nears the desired state, we subtract an amount proportional to the velocity:

  - (gain ∗ velocity)

  Derivative term (velocity is the derivative of position)
- A controller that has a derivative term is called a *derivative (D) controller*

Derivative Control

- A derivative controller has an output proportional to the derivative of its error = r input i:

$$\text{output} = K_d * d\,(\text{error input})/dt$$

- K_d is a proportionality constant
- The intuition behind derivative control:
  - Controller corrects for the momentum as it approaches the desired state
- Slow down the robot and decrease the turning angle while getting closer to the desired state
- Decrease the motor power while getting closer to the desired state

Proportional-Derivative (PD) Control

- Proportional-derivative control
  - Combination (sum) of proportional and derivative terms

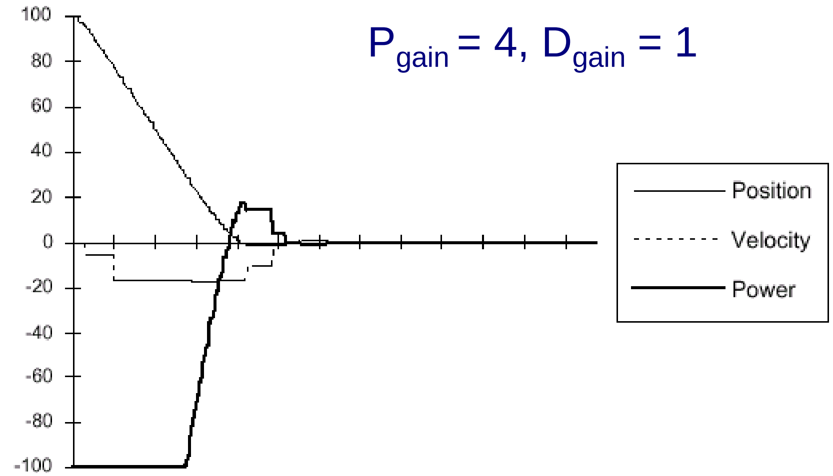
$$\text{output} = K_p * (\text{error input}) + K_d * d(\text{error input})/dt$$

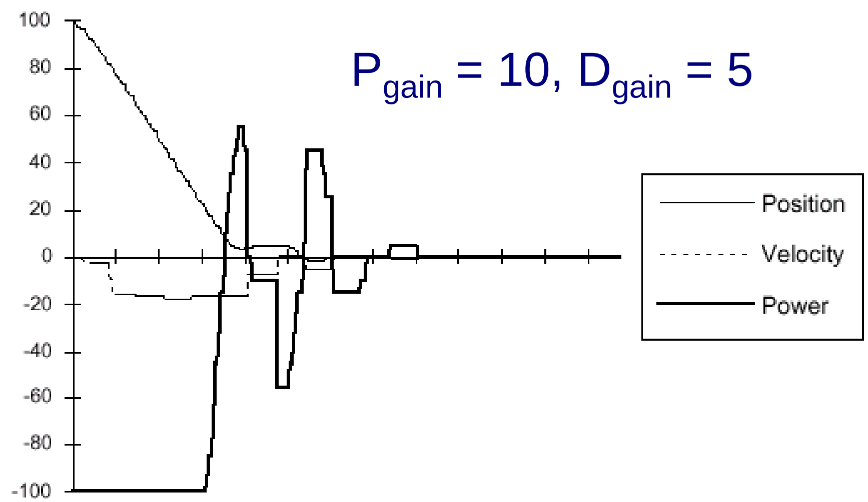
- *PD Control* is used extensively in industrial process control
  - Combination of varying the power input when the system is far away from the set point, and correcting for the momentum of the system as it approaches the set point is quite effective

Proportional-Derivative Control

power= P_{gain} * (0 - counts) - D_{gain} * velocity

- P_{gain}=4, D_{gain}= 1:
  - Overshoot is minimized, no oscillatory behavior at all
- P_{gain}=10, D_{gain}= 5:
  - Unstable; D_{gain} is too large
  - Position graph: controller "puts on the brakes" too hard and the system stops moving before the destination setpoint (between the 0.8 and 1.0 second mark)
  - When the velocity hits zero, the proportional gain kicks in again and the system corrects





http://www.cse.unr.edu/~monica/Courses/CPE470-670/cpe470-670-sp05.html

Integral Control

- errors accumulate as the robot is running a longer time
- How can we solve the problem?
  - Sum up the errors and compensate for them when they become significantly large
- The control system can be improved by introducing an *integral term*

$$output = K_i * \int(error\ input)dt$$

- K_i = proportionality constant
- Intuition:
  - System keeps track of its repeatable, steady state errors
  - These errors are integrated (summed up) over time
  - When they reach a threshold, the system compensates for them

Proportional integral derivative (PID) Control

This is a combination (sum) of the

proportional, derivative and integral terms

output = $K_p * i + K_d * di/dt + K_f * \int i(t)dt$,

where i = error input