MEAM 520
Lecture 23: Actuation

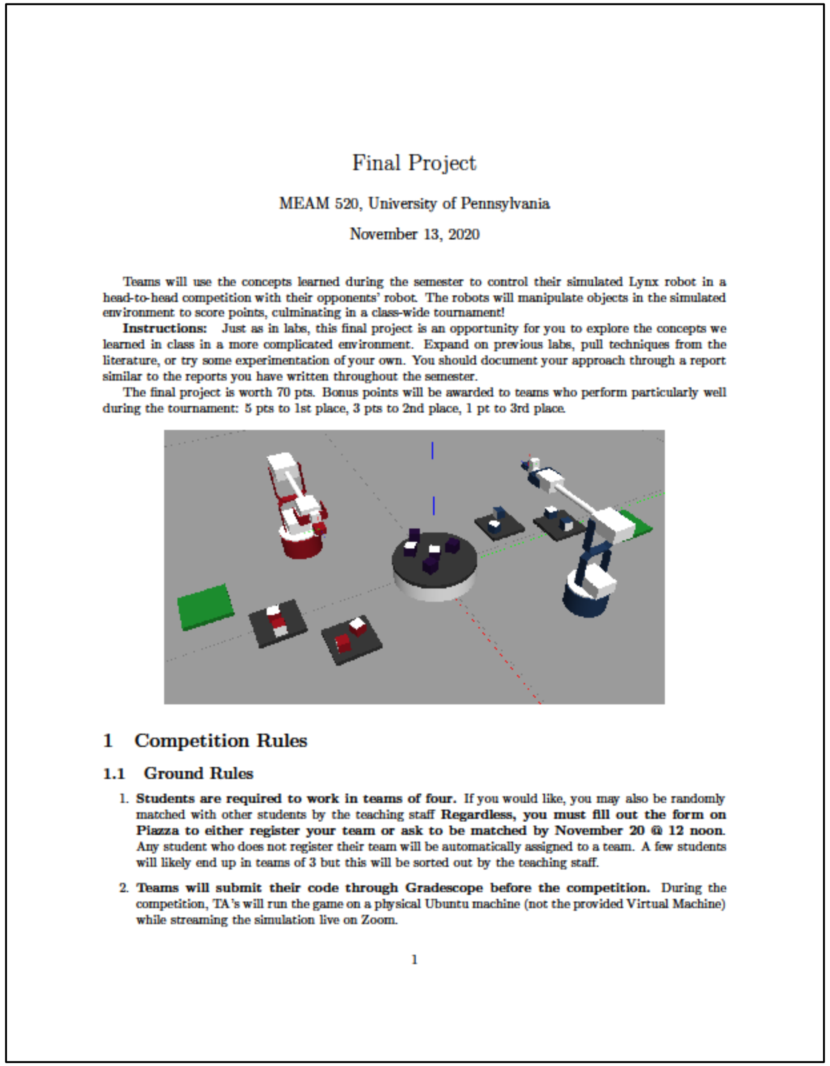
Cynthia Sung, Ph.D.

Mechanical Engineering & Applied Mechanics

University of Pennsylvania

Thanks to Shane for talking
about ROS last time !

Final Projects



Final Project

MEAM 520, University of Pennsylvania

November 13, 2020

Teams will use the concepts learned during the semester to control their simulated Lynx robot in a head-to-head competition with their opponents' robot. The robots will manipulate objects in the simulated environment to score points, culminating in a class-wide tournament!

Instructions: Just as in labs, this final project is an opportunity for you to explore the concepts we learned in class in a more complicated environment. Expand on previous labs, pull techniques from the literature, or try some experimentation of your own. You should document your approach through a report similar to the reports you have written throughout the semester.

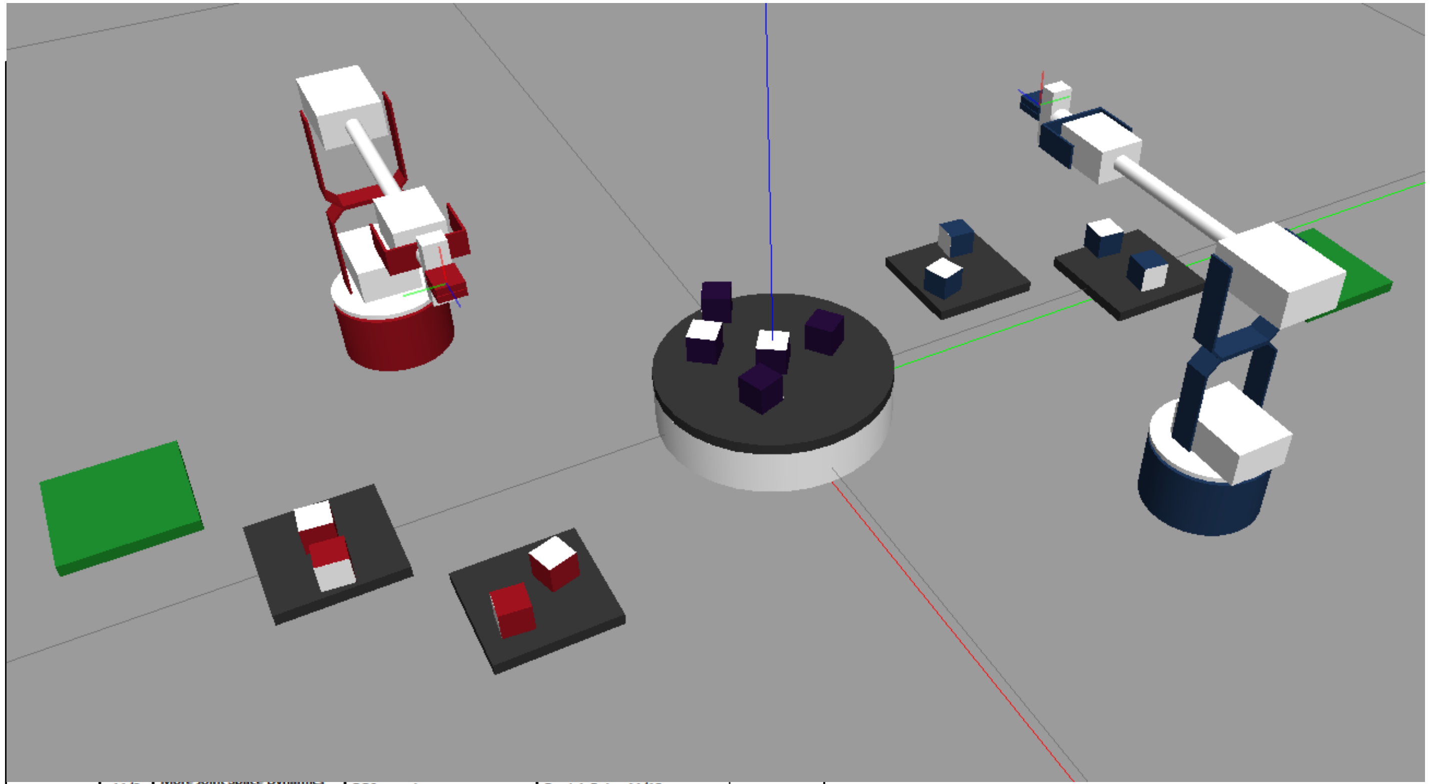
The final project is worth 70 pts. Bonus points will be awarded to teams who perform particularly well during the tournament: 5 pts to 1st place, 3 pts to 2nd place, 1 pt to 3rd place.

1 Competition Rules

1.1 Ground Rules

1. **Students are required to work in teams of four.** If you would like, you may also be randomly matched with other students by the teaching staff **Regardless, you must fill out the form on Piazza to either register your team or ask to be matched by November 20 @ 12 noon.** Any student who does not register their team will be automatically assigned to a team. A few students will likely end up in teams of 3 but this will be sorted out by the teaching staff.
2. **Teams will submit their code through Gradescope before the competition.** During the competition, TA's will run the game on a physical Ubuntu machine (not the provided Virtual Machine) while streaming the simulation live on Zoom.

1



11/9	*Read: SHV 7.4-7.7*	ROS overview	Pre-lab 5 due 11/13
11/16	Actuation and Control *Read: AKKK 7.2-7.3, SHV 6.intro-6.3*	PID Control *Read: SHV 6.3*	Lab 5 due 11/20
11/23	Sensing and State Estimation *Read: AKKK 8.1-8.3*	Thanksgiving – No class	
11/30	Multi-agent Planning *Paper reading*	Final project: Round Robin	Mid-project update due 12/2
12/	Final project: Bracket		Final project due 12/10 (no penalty deadline: 12/14)

The purpose of the tournament is to help you in your evaluation. Identify strengths of limitations throughout the course of the tournament. Feel free to edit your code between this date and the final submission.

Register your team at:
https://forms.gle/wnpzXc44BbVyg1Dt9

Previously: Manipulator Equation

We can write this as a matrix equation

$$\tau = \underbrace{D(q)}_{}\ddot{q} + C(q,\dot{q})\dot{q} + g(q)$$

SHV uses a bit of strange notation.
Most people call this matrix *H* or *M*.

where

$D(q)$ is the *nxn* mass matrix (inertia terms)

$C(q,\dot{q})$ is the *nxn* matrix of centrifugal (square of joint velocities) and Coriolis (product of two different joint velocities) terms

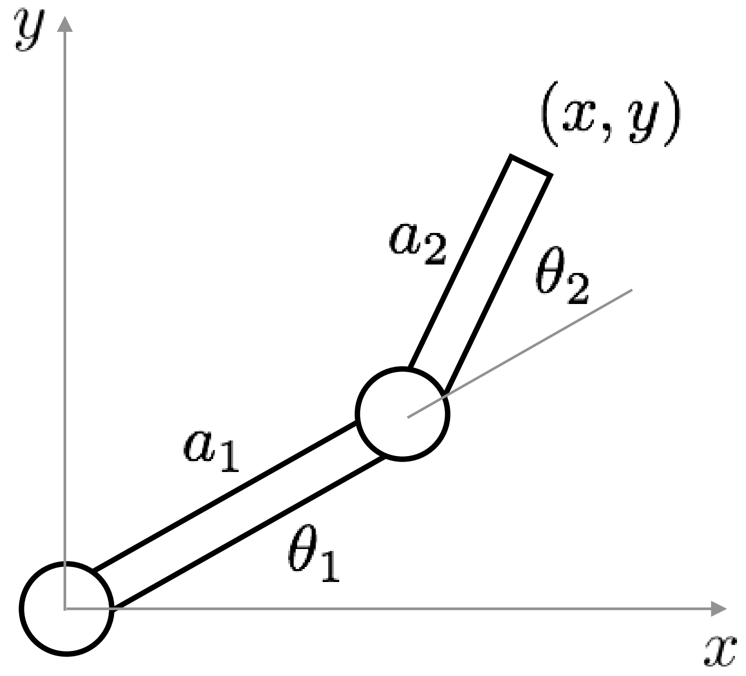
$g(q)$ is a *nx1* vector of gravitational terms

Previously: Euler-Lagrange Method

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q)$$

$$D = \sum_{i=1}^{N} \left(m_i J_{vci}^{\top} J_{vci} + J_{\omega i}^{\top} R_i I_i R_i^{\top} J_{\omega i} \right)$$

$$g = \frac{\partial}{\partial q} \sum_{i=1}^{N} m_i \vec{g} \cdot \vec{r}_i$$



$$(C\dot{q})_k = \sum_{i,j} \frac{1}{2} \left(\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k} \right) \dot{q}_i \dot{q}_j$$

or

$$c_{kj} = \sum_{i} \frac{1}{2} \left(\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k} \right) \dot{q}_i$$

Previously: Newton-Euler (for revolute joints)

Start with $\omega_0 = 0, \alpha_0 = 0, a_{c,0} = 0, a_{e,0} = 0$

Solve kinematic constraints for i from 1 to n

Start with $f_{n+1} = 0, \tau_{n+1} = 0$

Solve force/moments for i from n to 1

Kinematic constraints

i terms on the left

No forces/moments!

$$\omega_i = R_{i-1}^i \omega_{i-1} + z_{i-1}^i \dot{q}_i$$

$$\alpha_i = R_{i-1}^i \alpha_{i-1} + z_{i-1}^i \ddot{q}_i + \omega_i \times z_{i-1}^i \dot{q}_i$$

$$a_{e,i} = R_{i-1}^i a_{e,i-1} + \dot{\omega}_i \times r_{i,i+1} + \omega_i \times (\omega_i \times r_{i,i+1})$$

$$a_{c,i} = R_{i-1}^i a_{e,i-1} + \dot{\omega}_i \times r_{i,c_i} + \omega_i \times (\omega_i \times r_{i,c_i})$$

i-1 terms on the right

Forces/Moments

$$f_i - R_{i+1}^i f_{i+1} + m_i g_i = m_i a_{c,i}$$

$$\tau_i - R_{i+1}^i \tau_{i+1} + f_i \times r_{i,c_i} - (R_{i+1}^i f_{i+1}) \times r_{i+1,c_i} = I_i \dot{\omega}_i + \omega_i \times (I_i \omega_i)$$

Previously: Method Comparisons

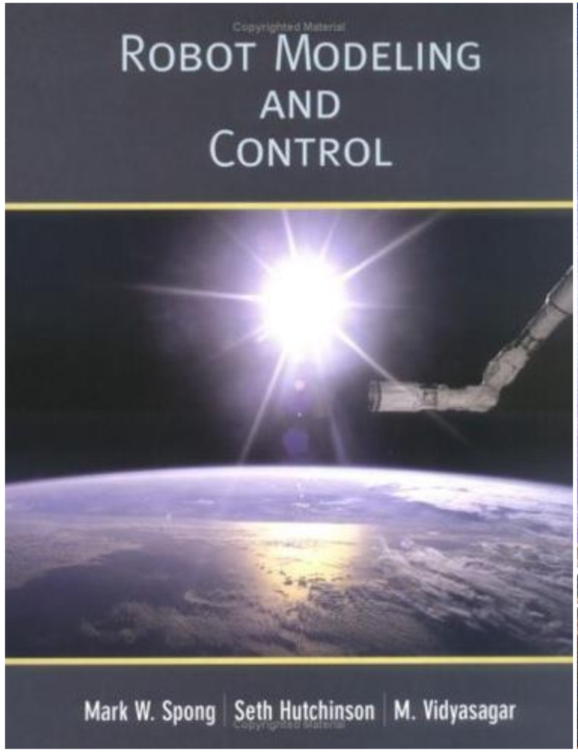
Newton-Euler

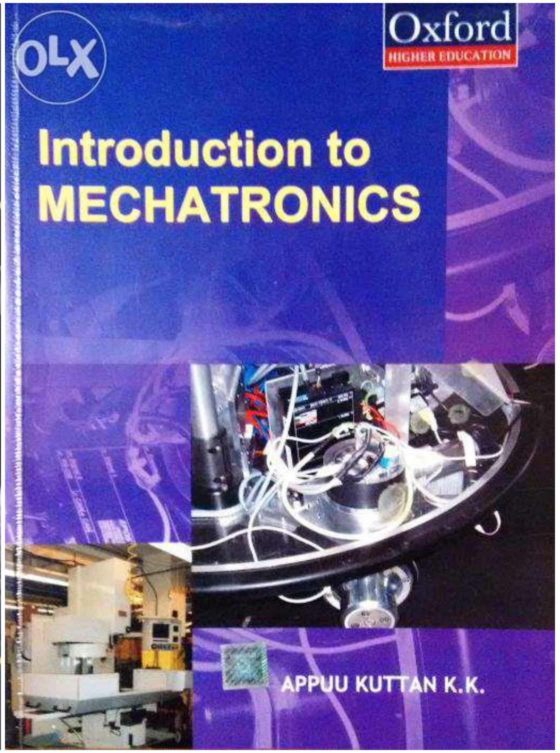
- Complete solution for all forces and kinematic variables
- Inefficient when only a few of the system's forces need to be solved for

Euler-Lagrange

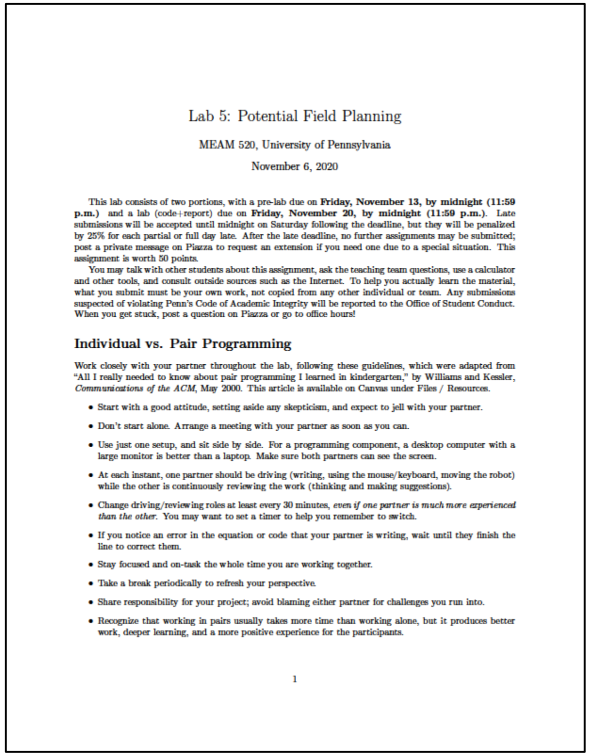
- Disregard all interactive and constraint forces that do not perform work
- Need to differentiate scalar energy functions
- Inefficient for large multi-body systems

Today: Actuation and Control





- Read: SHV 6.intro – 6.3
- AKKK: 7.2 – 7.3



Lab 5: Potential Field Planning

MEAM 520, University of Pennsylvania

November 6, 2020

This lab consists of two portions, with a pre-lab due on **Friday, November 13, by midnight (11:59 p.m.)** and a lab (code+report) due on **Friday, November 20, by midnight (11:59 p.m.)**. Late submissions will be accepted until midnight on Saturday following the deadline, but they will be penalized by 25% for each partial or full day late. After the late deadline, no further assignments may be submitted; post a private message on Piazza to request an extension if you need one due to a special situation. This assignment is worth 50 points.

You may talk with other students about this assignment, ask the teaching team questions, use a calculator and other tools, and consult outside sources such as the Internet. To help you actually learn the material, what you submit must be your own work, not copied from any other individual or team. Any submissions suspected of violating Penn's Code of Academic Integrity will be reported to the Office of Student Conduct. When you get stuck, post a question on Piazza or go to office hours!

Individual vs. Pair Programming

Work closely with your partner throughout the lab, following these guidelines, which were adapted from "All I really needed to know about pair programming I learned in kindergarten," by Williams and Kessler, *Communications of the ACM*, May 2000. This article is available on Canvas under Files / Resources.

- Start with a good attitude, setting aside any skepticism, and expect to jell with your partner.
- Don't start alone. Arrange a meeting with your partner as soon as you can.
- Use just one setup, and sit side by side. For a programming component, a desktop computer with a large monitor is better than a laptop. Make sure both partners can see the screen.
- At each instant, one partner should be driving (writing, using the mouse/keyboard, moving the robot) while the other is continuously reviewing the work (thinking and making suggestions).
- Change driving/reviewing roles at least every 30 minutes, *even if one partner is much more experienced than the other*. You may want to set a timer to help you remember to switch.
- If you notice an error in the equation or code that your partner is writing, wait until they finish the line to correct them.
- Stay focused and on-task the whole time you are working together.
- Take a break periodically to refresh your perspective.
- Share responsibility for your project; avoid blaming either partner for challenges you run into.
- Recognize that working in pairs usually takes more time than working alone, but it produces better work, deeper learning, and a more positive experience for the participants.

1

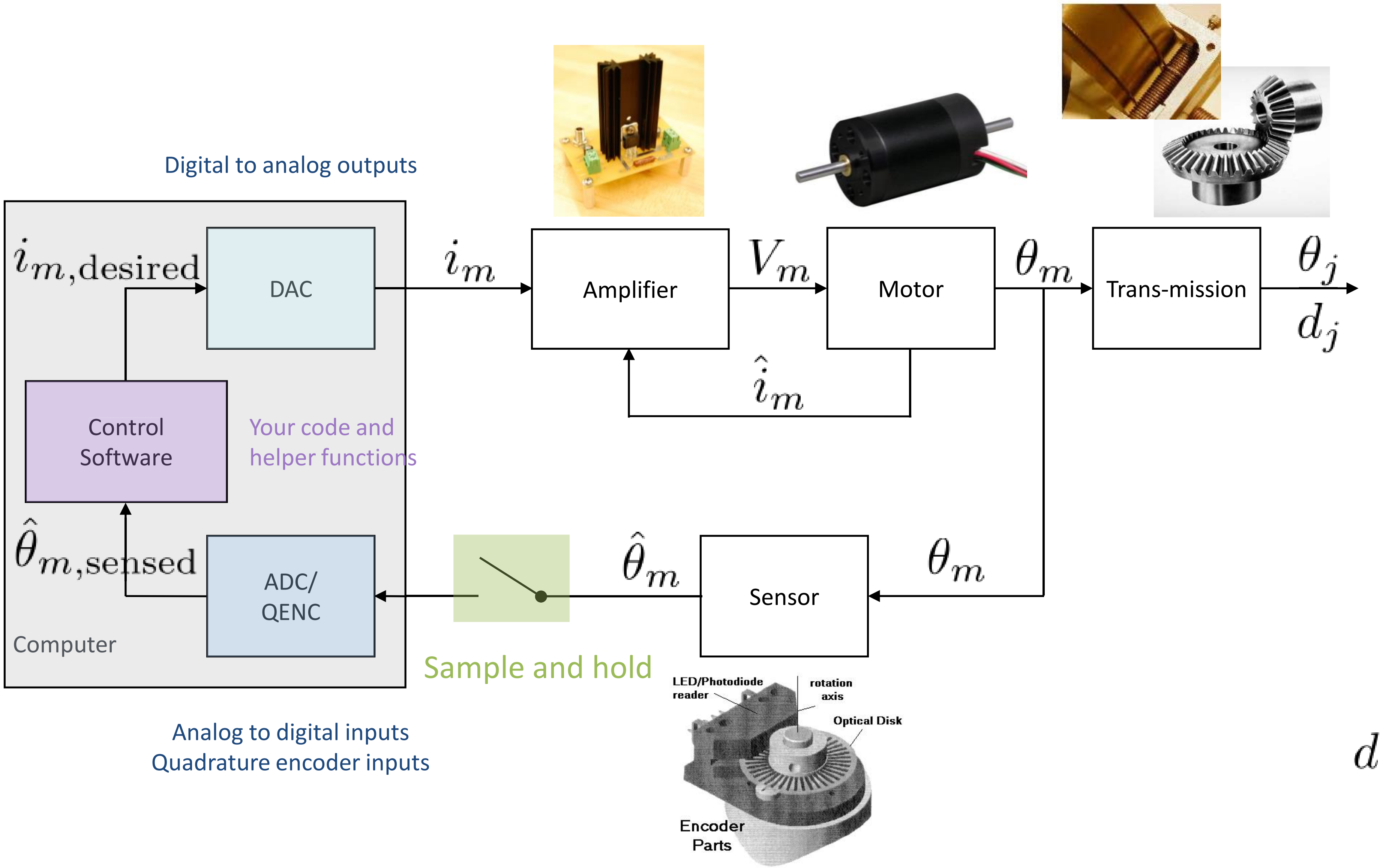
Lab 5: Potential Fields due 11/20

Even if you input the exact torques from these equations, your real robot may not follow the desired trajectory. Why?

- Ignored issues like: friction, vibration (non-rigidity), hysteresis (backlash), etc.
- Calibration: wrong link lengths, offsets, or zeroing of angles
- Control precision: may not be as high as our calculations
- Actuation limits: may not be able to deliver necessary torques
- Noise: on electrical lines, sensor readings
- Computation limits: may not be able to compute fast enough to react

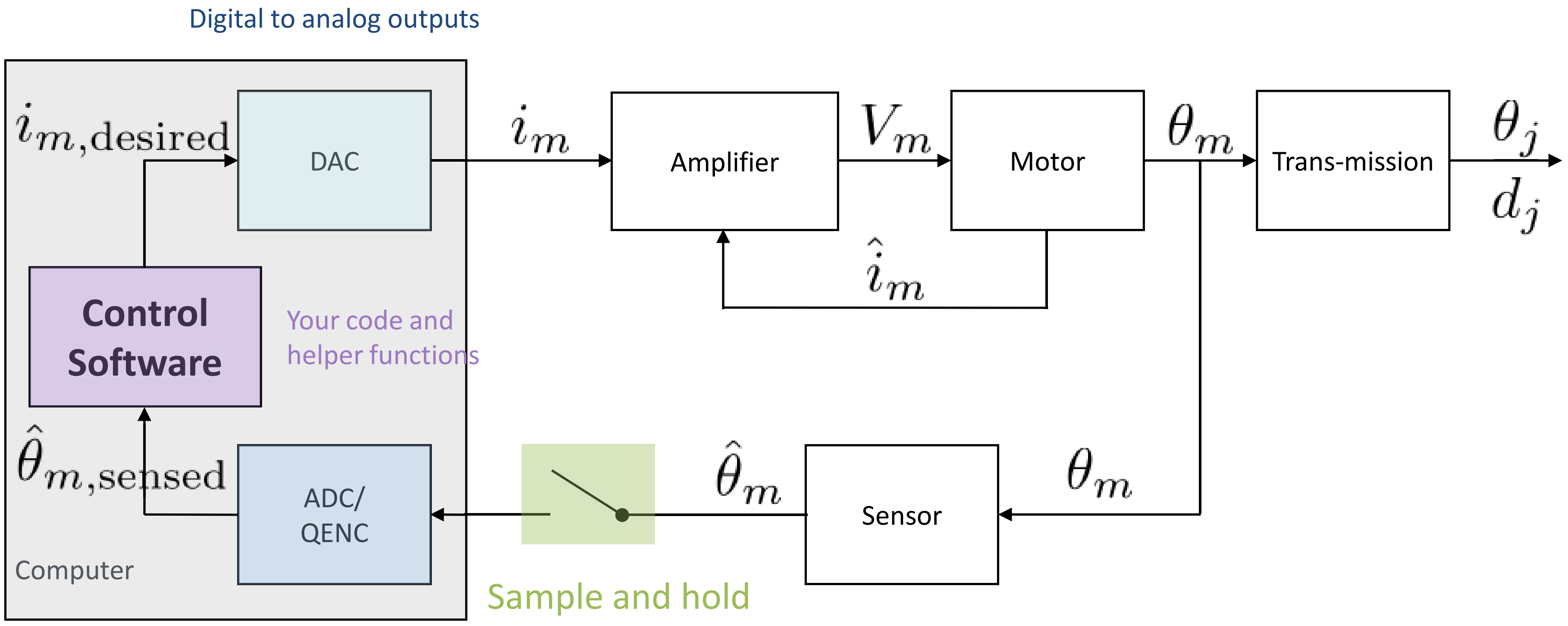
Every robot has non-idealities that make its behavior deviate from predictions.

How most real robots work



i : current
V : voltage
m : motor
j : joint
θ : angle
d : displacement
$\hat{}$: estimate

Control Software



From Lecture 18

sensor data to joint values

$$\vec{q}_k = \vec{a} * \vec{Q}_k + \vec{b}$$

joint values to position

$$\vec{x}_k = \Lambda(\vec{q}_k)$$

controller

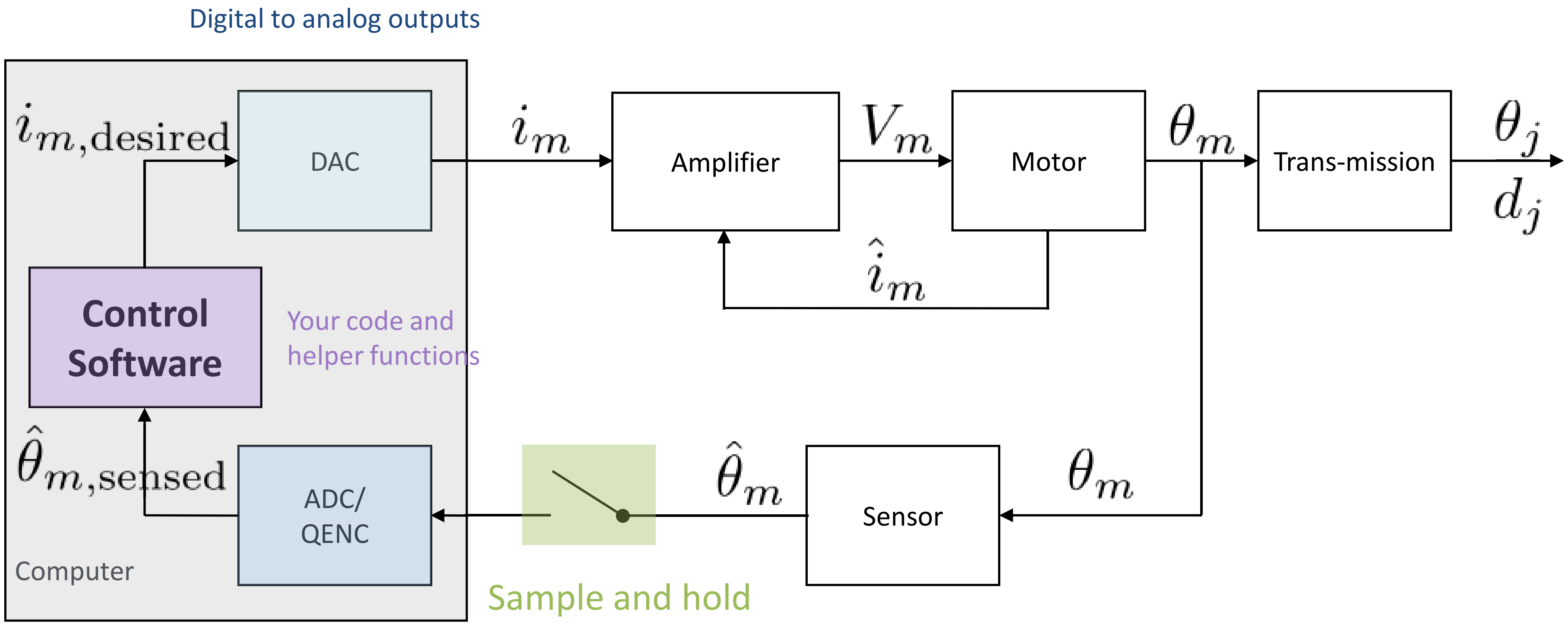
$$\vec{\tau}_k = F_i(\vec{q}_k)$$

joint torques to control outputs

$$\vec{r}_k = \vec{c} * \vec{\tau}_k + \vec{d}$$

Joint Angles to Robot Position

How can we estimate robot pose from $\hat{\theta}_j$? Forward kinematics!



sensor data to joint values

$$\vec{q}_k = \vec{a} * \vec{Q}_k + \vec{b}$$

joint values to position

$$\vec{x}_k = \Lambda(\vec{q}_k)$$

controller

$$\vec{\tau}_k = F_i(\vec{q}_k)$$

joint torques to control outputs

$$\vec{r}_k = \vec{c} * \vec{\tau}_k + \vec{d}$$

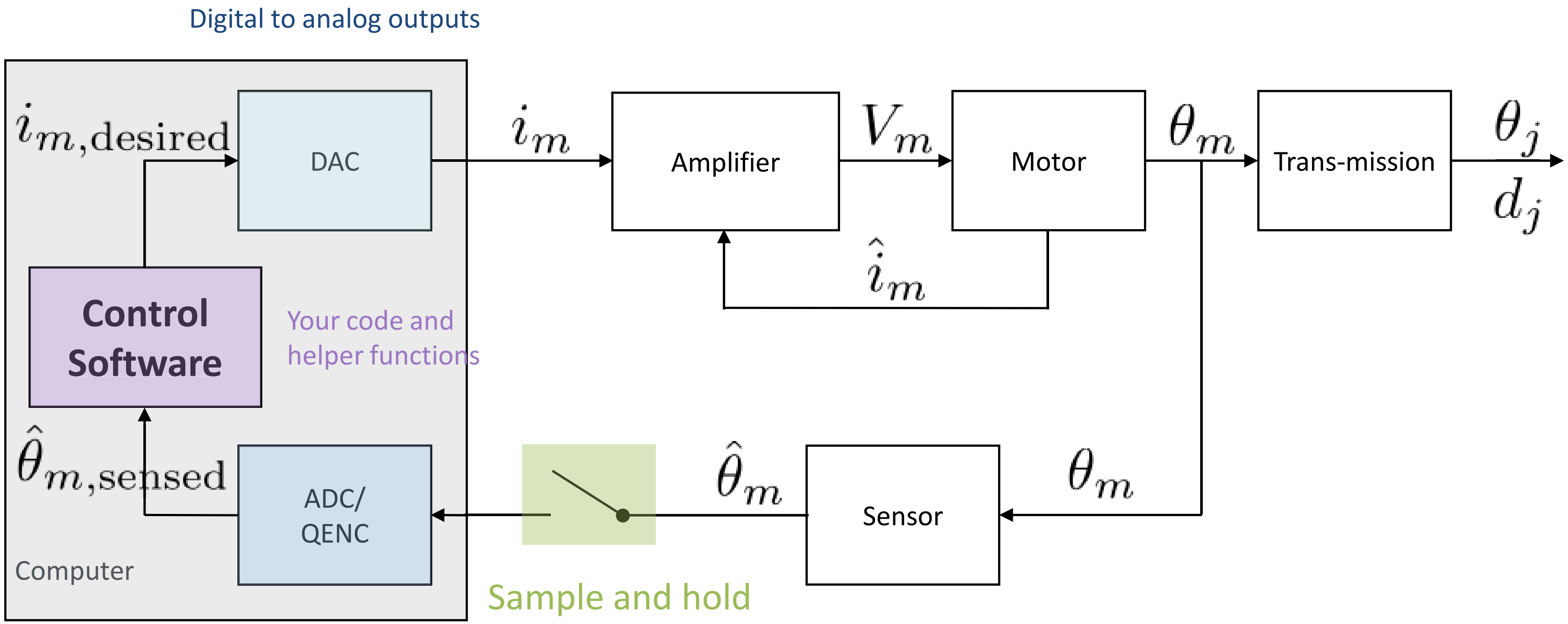
Derive using DH or geometry.

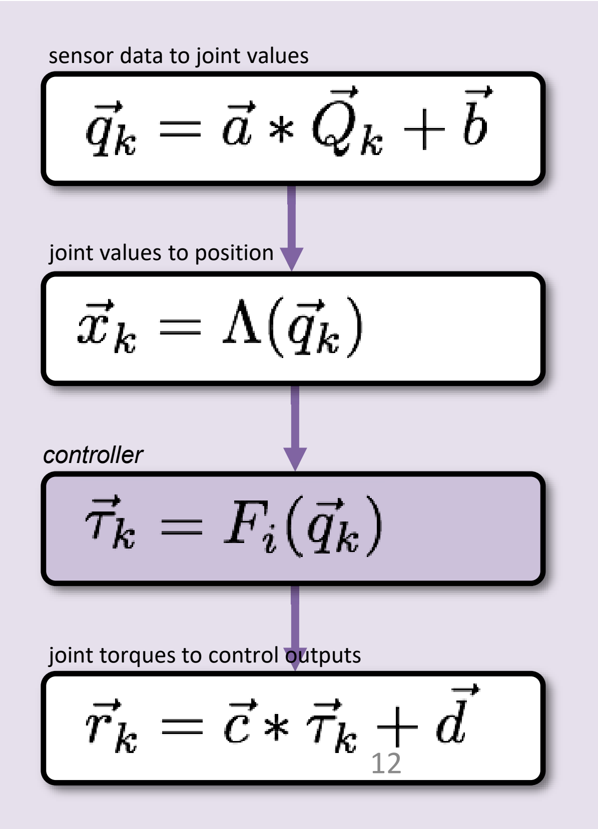
zero pose, link lengths, and link offsets matter!

Robot Positions to Forces to Torques

How can we calculate joint torques from desired forces?

Jacobian transpose!





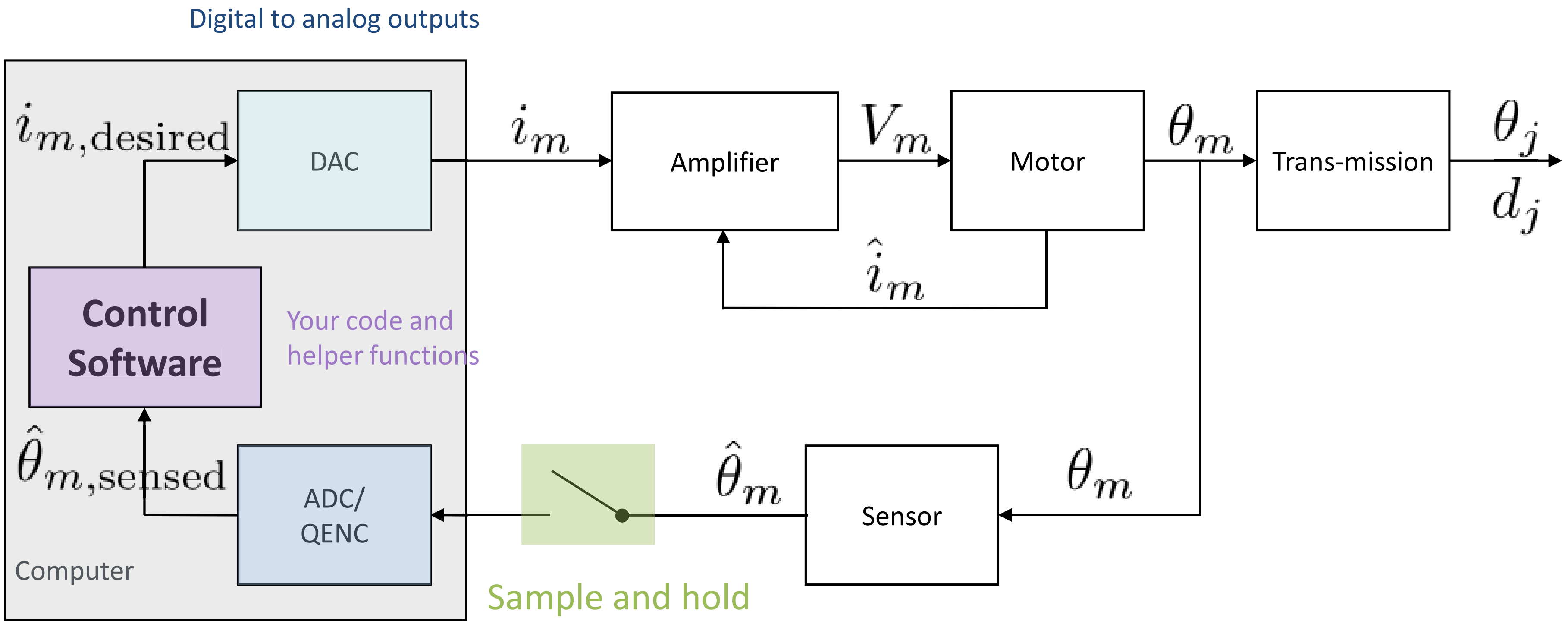
*Calculate the Jacobian transpose for the **robot's current pose***

$$\vec{\tau} = J_v^\top \vec{F}$$

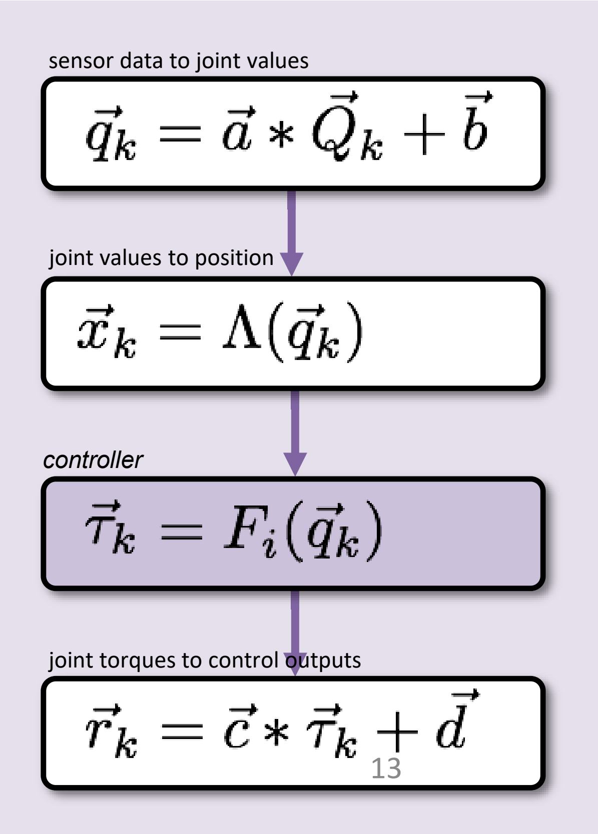
Alternatively, Robot Positions to Forces to Torques

How can we calculate joint torques from desired forces?

Manipulator equation!



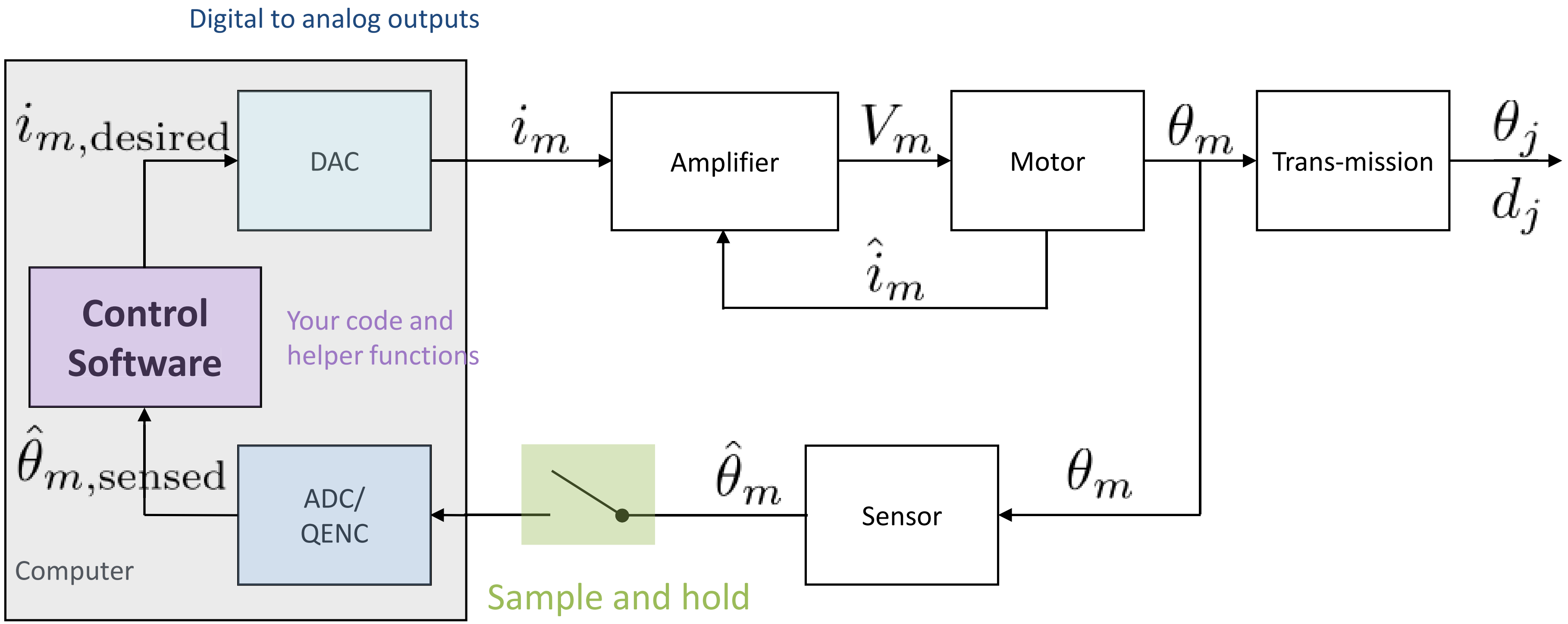
$$\tau = D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q)$$



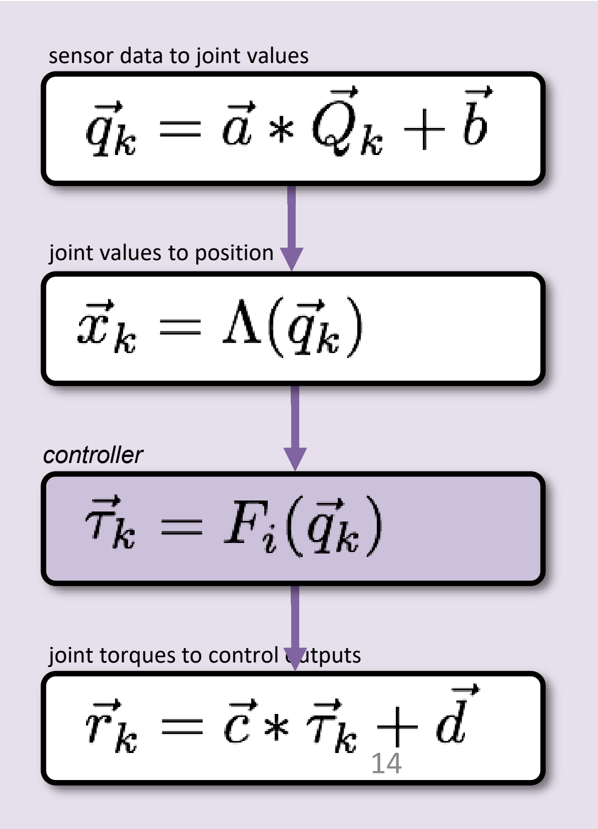
Alternatively, Robot Positions to Torques

How can we calculate joint torques from positions? PID control!

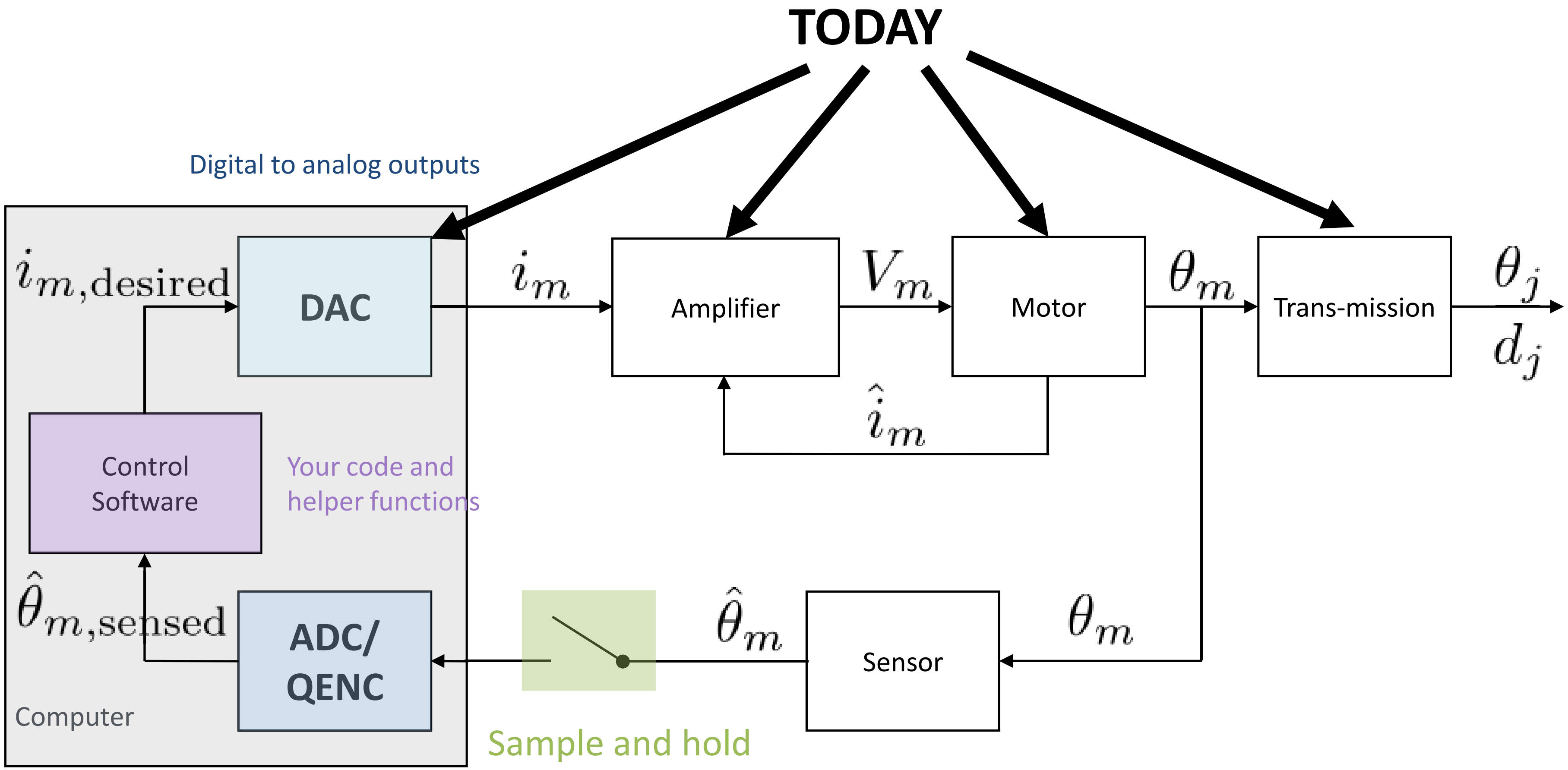
next time

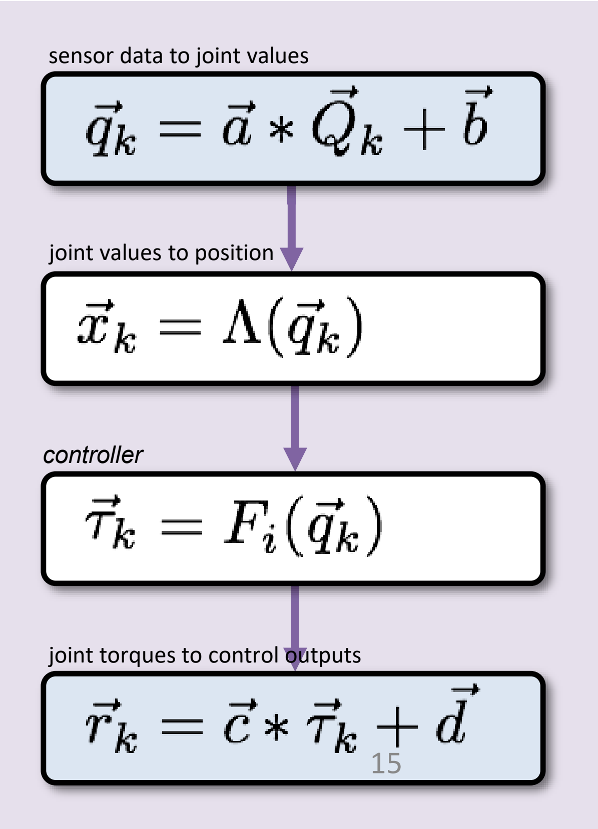


$$\tau_i = k_p(\theta_{i,des} - \theta_i) + k_d(\omega_{i,des} - \omega_i) + k_i \int \theta_{i,des} - \theta_i$$

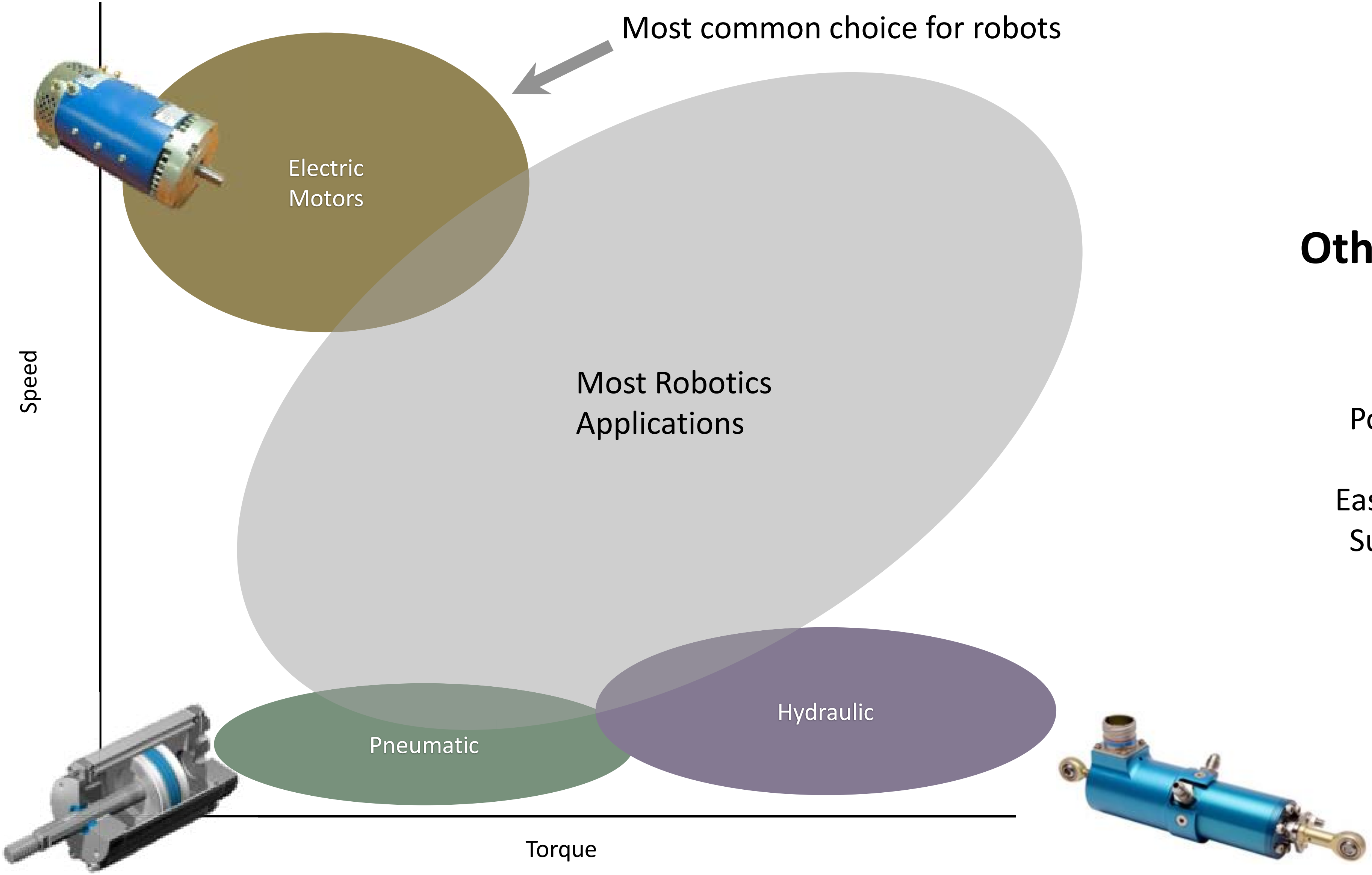


What goes in here?





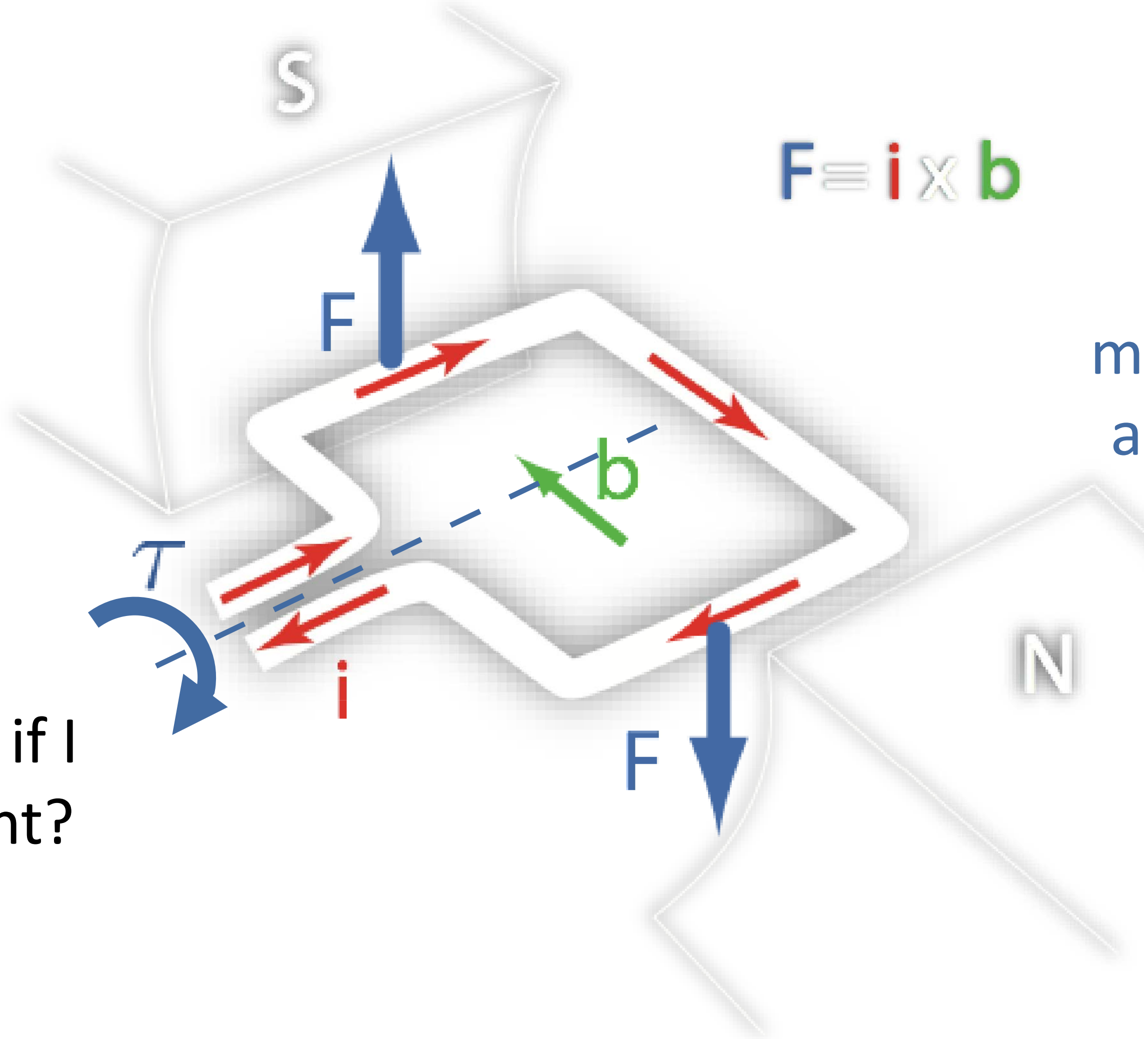
Types of Actuators



Other concerns?

Cost
Availability
Size
Power to Weight
Stiffness
Ease of Integration
Support Systems

Electric Motors

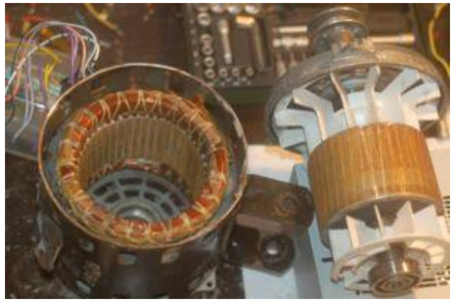


Pair of offset equal-magnitude forces causes a **torque** (a.k.a. couple, moment) around the axis of rotation

What would happen if I turned off the current?

What would happen if I flipped the sign of the current?

What if I kept the current but rotated coil to another position?



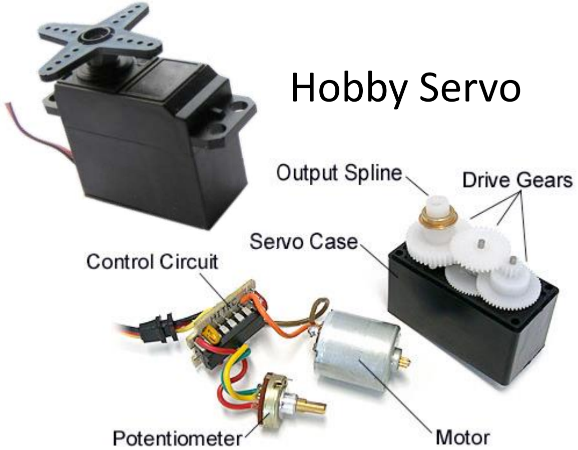
AC

Magnetic Rotor
Coil Stator
Output speed is a sub-multiple of voltage supply frequency

DC Brushed

Most common!

Coil Rotor
Magnetic Stator
Brushes carry current to the rotor



Hobby Servo

DC brushed motor with integrated sensor and controller

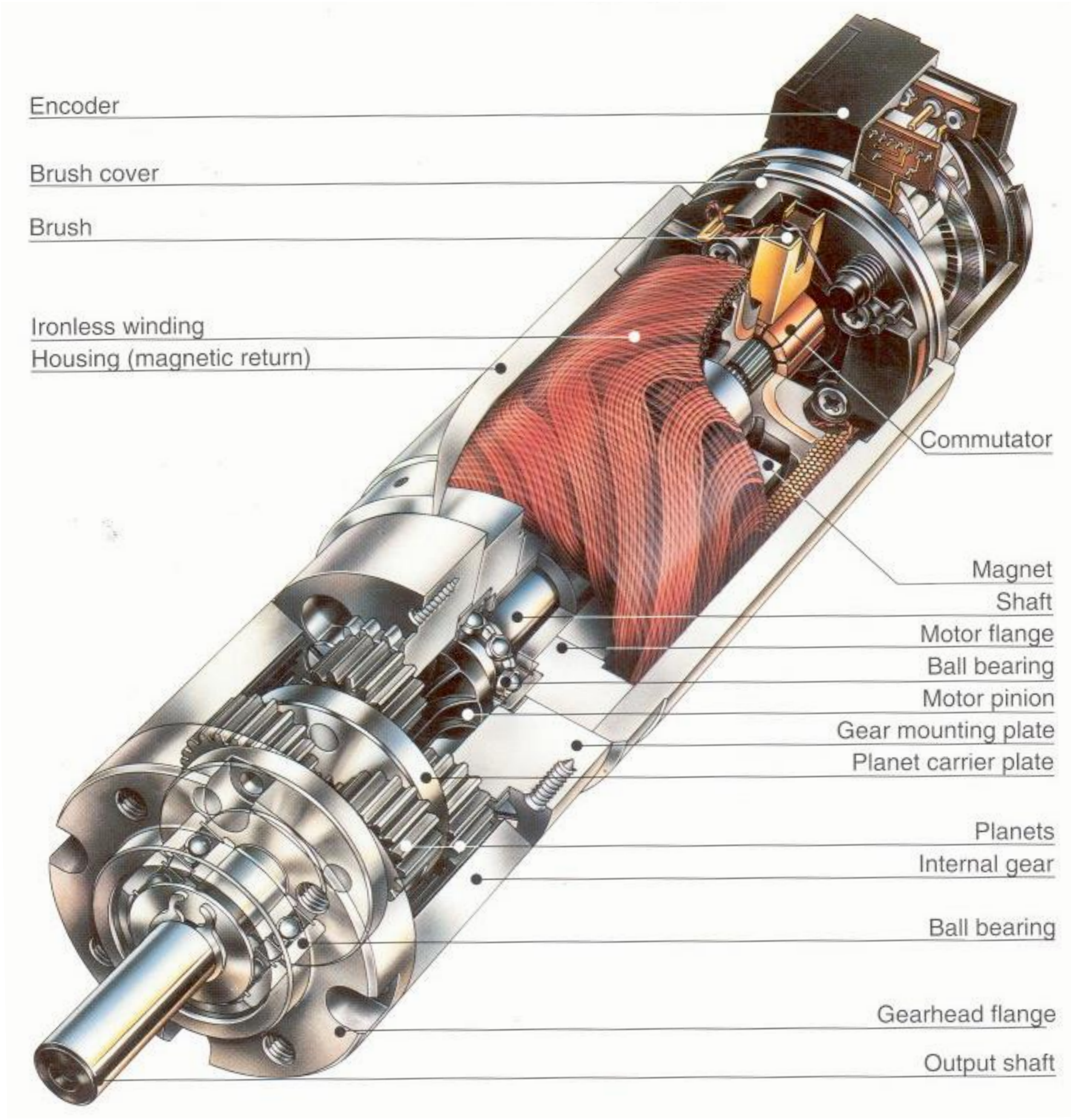
DC Brushless

Magnetic Rotor
Coil Stator
Similar in construction to AC, but electrically commutated
Requires a position sensor (commonly built in)

Stepper

Toothed Magnetic Rotor
Multi-Coil Stator
Capable of open-loop positioning
Requires a controller

DC Motor



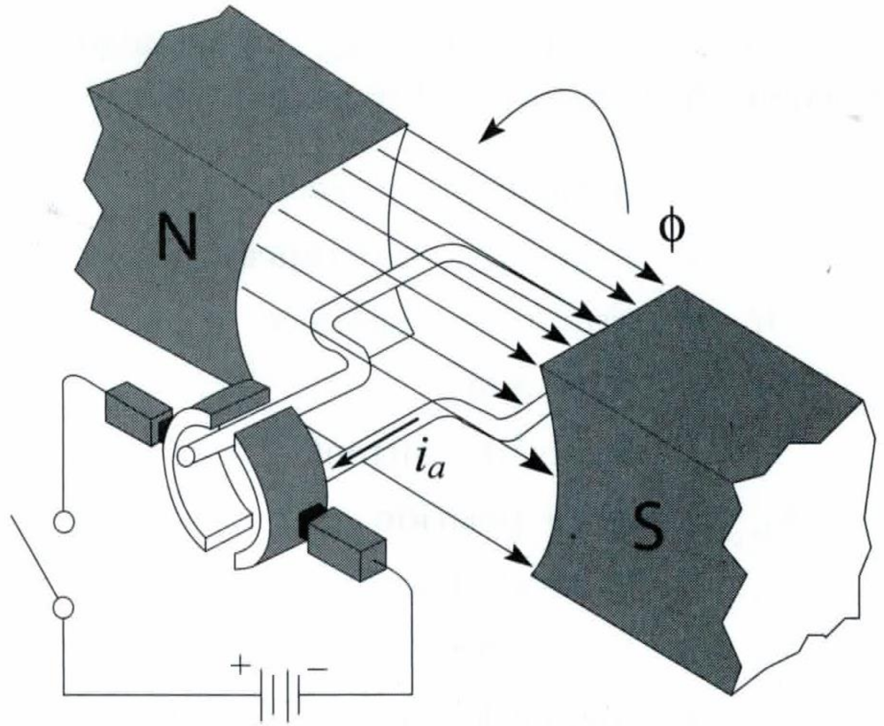
Encoder
Brush cover
Brush
Ironless winding
Housing (magnetic return)
Commutator
Magnet
Shaft
Motor flange
Ball bearing
Motor pinion
Gear mounting plate
Planet carrier plate
Planets
Internal gear
Ball bearing
Gearhead flange
Output shaft

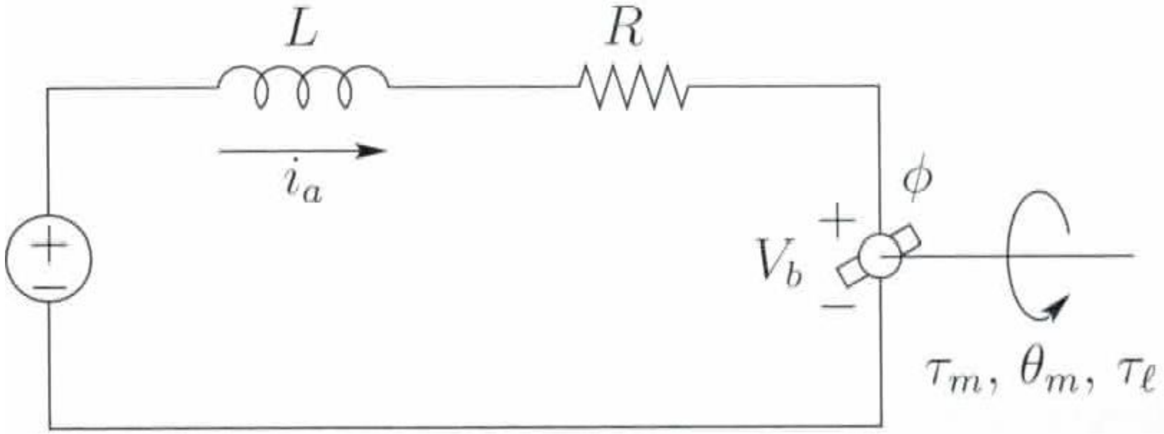
DC Motor

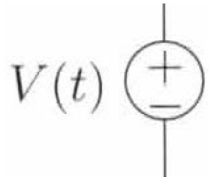
Commutation ensures that the coil with the best mechanical advantage is receiving the current.

(SHV Section 6.1)

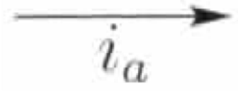
DC Motor



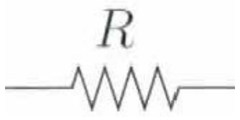




Time-varying **voltage** supply.
Compare **voltage** to water pressure.

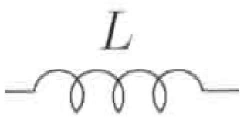


Current through the motor armature.
Compare **current** to water flow.



Electrical **resistance** of the armature.
Compare **resistance** to water flow resistance in pipes (small diameter)
Follows Ohm's Law.

$$V = iR$$

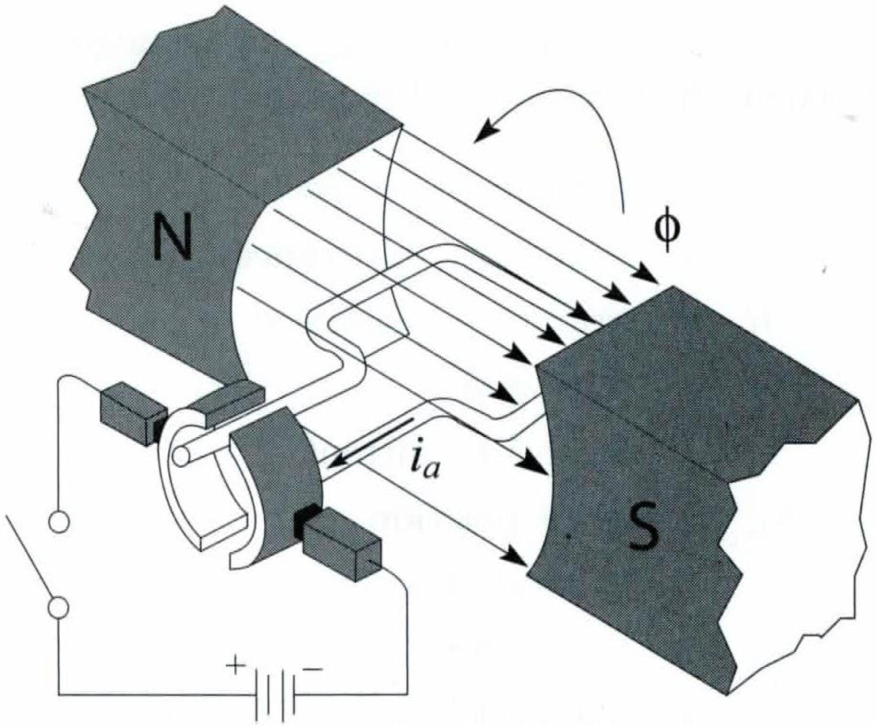


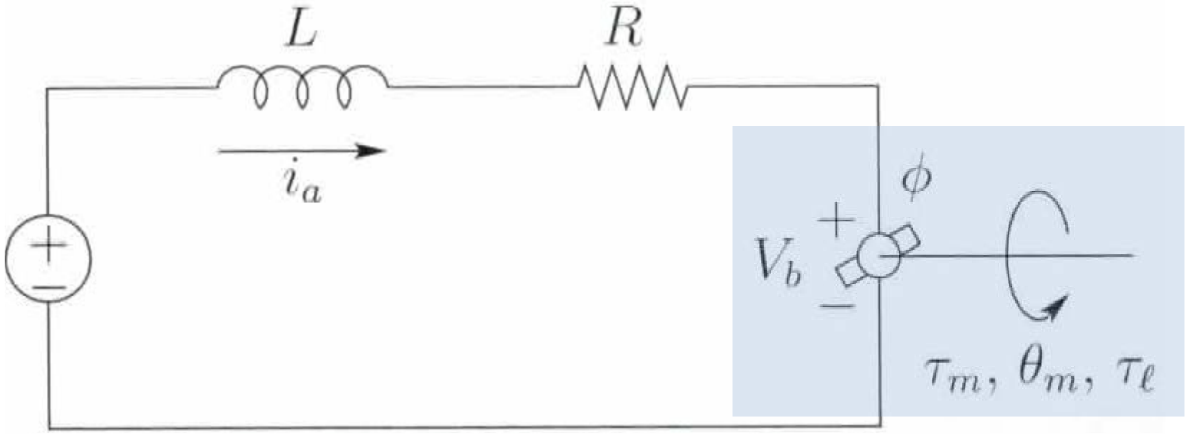
Electrical **inductance** of armature.
Compare **inductance** to the momentum of the water flowing in a pipe.
Follows constitutive equation.

$$V = L\frac{di}{dt}$$

Circuit representation for a DC motor driven by a time-varying voltage.

DC Motor





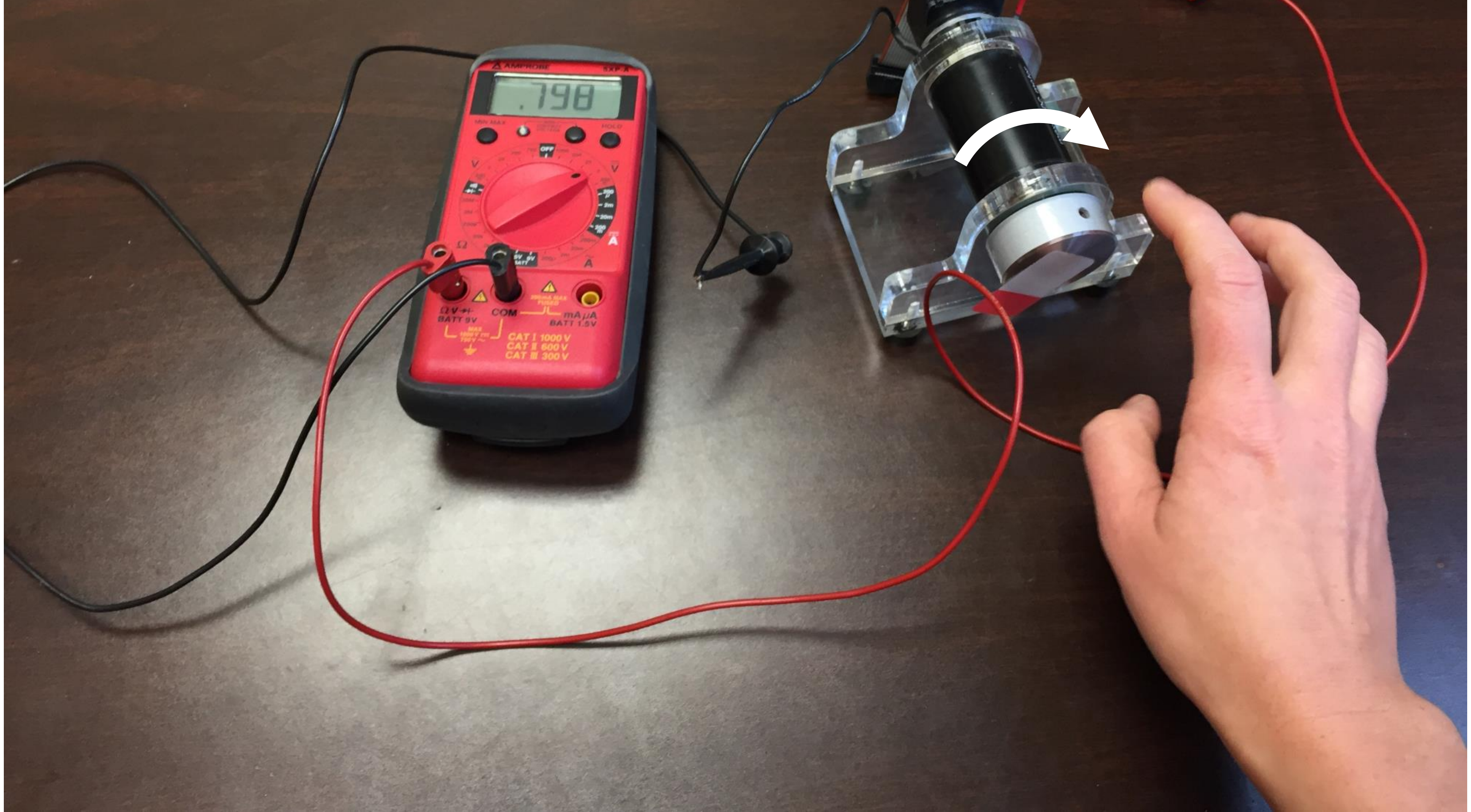
Circuit representation for a DC motor driven by a time-varying voltage.

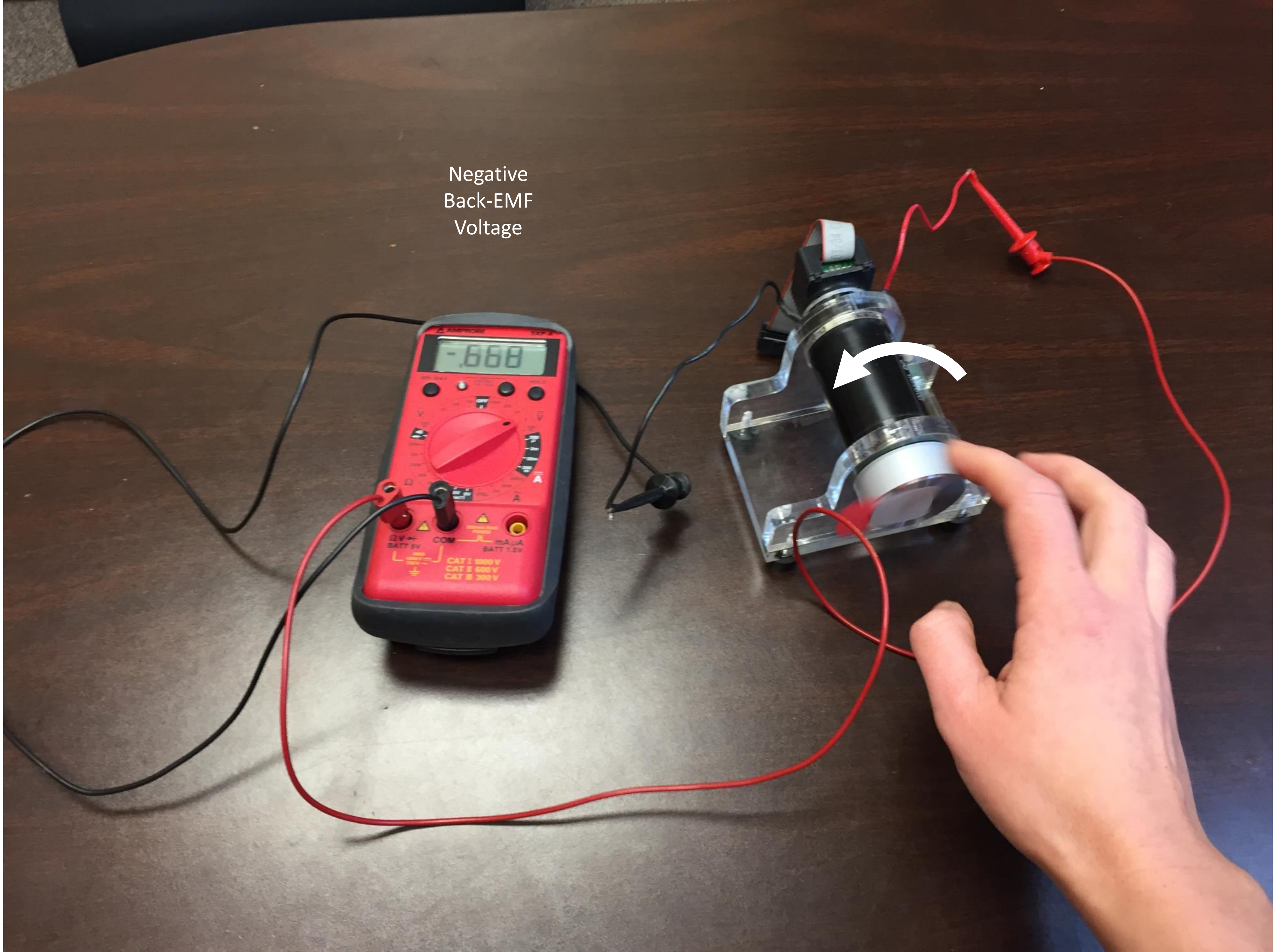
Time-varying voltage caused by the back electro-motive force (**back-emf**).

Back-emf voltage is proportional to the rotational speed of the motor and opposes the voltage that would drive the motor in the direction in which it is rotating.

$$V_b = k_v\, \omega_m$$

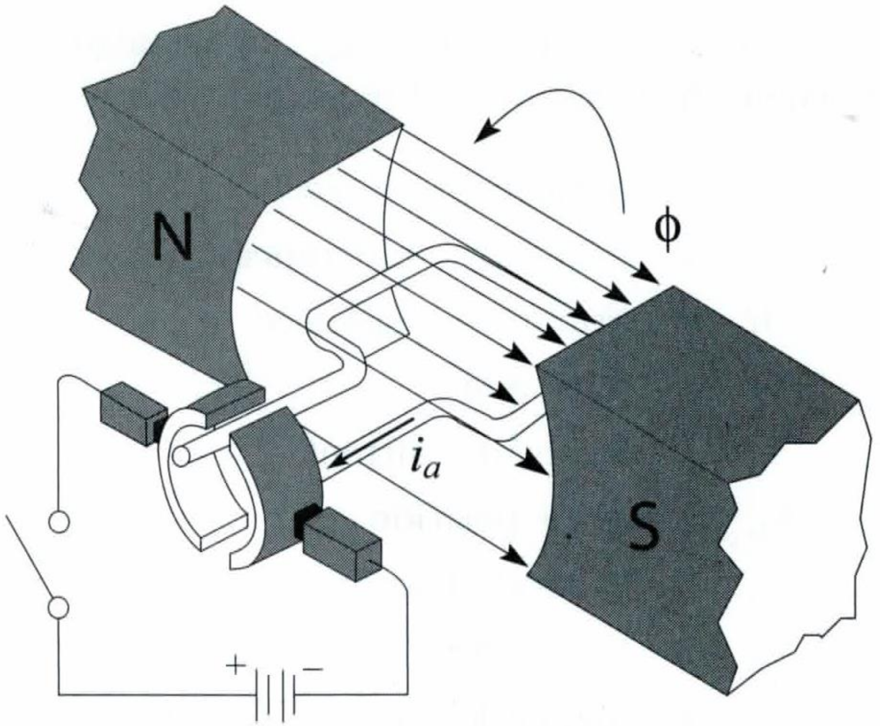
Positive
Back-EMF
Voltage

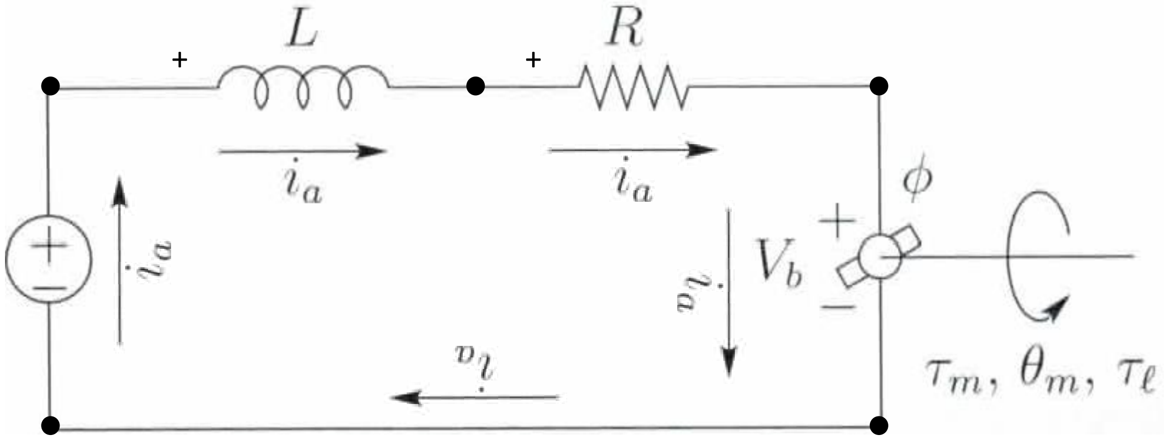



Negative
Back-EMF
Voltage
-.668


Turning one motor creates back-emf (a voltage across that motor's terminals).
If two motors are connected terminal to terminal, turning the first motor makes the other motor turn.
maxon DC motor

DC Motor





How do we analyze this circuit?

Kirchoff's Current Law (KCL)

The sum of the currents flowing into (positive) and out of (negative) a node in the circuit is zero.

The current flowing through all elements of our circuit is the same.

Kirchoff's Voltage Law (KVL)

The sum of the voltage drops around any loop in the circuit is zero.

Draw a + sign where current enters each element other than voltage sources.

$$V(t) - V_L - V_R - V_b = 0$$

$$V = L\frac{di}{dt}$$

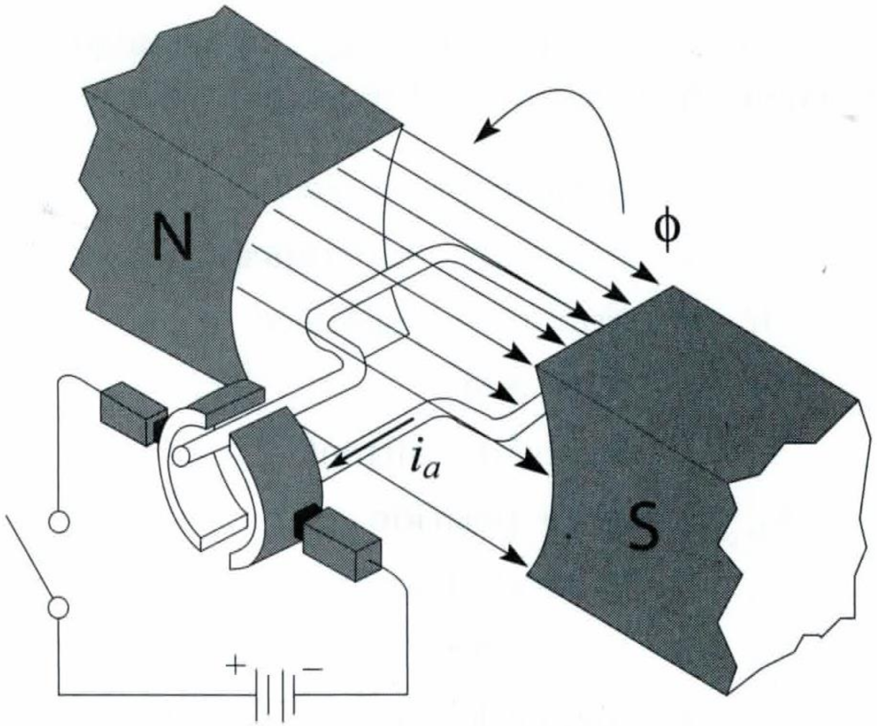
$$V = iR$$

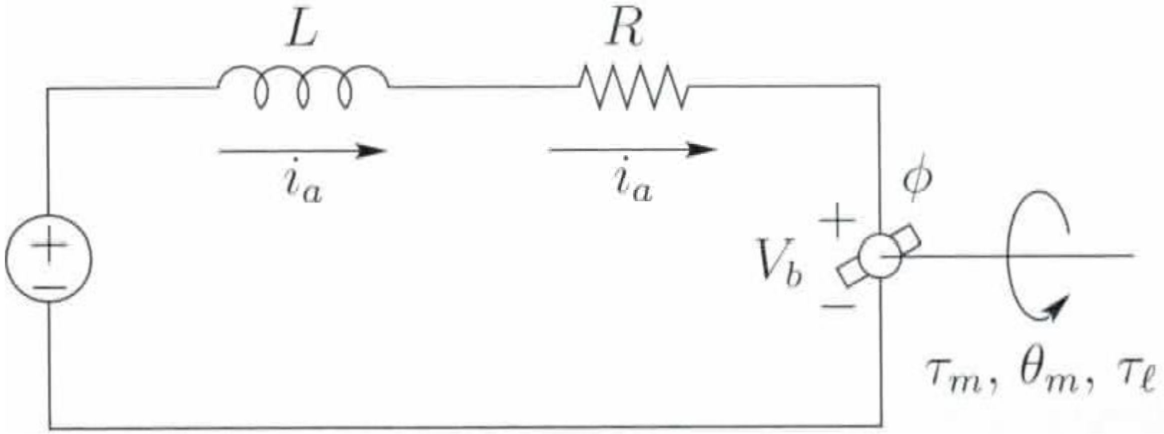
$$V_b = k_v\,\omega_m$$

$$\boxed{V(t) = L\frac{di_a}{d_t} + R\,i_a + k_v\,\omega_m}$$

DC Motor

(SHV Section 6.1)





magnetic flux (webers)

torque constant (N•m/A)

$$\tau_m = K_1\,\phi\,i_a = k_t\,i_a$$

generated torque (N•m)

physical constant

armature current (A)

armature current (A)

$$k_t = k_v$$

if using meters, kilograms and seconds

back emf (V)

magnetic flux (webers)

back-emf constant (V•s)

$$V_b = K_2\,\phi\,\omega_m = k_v\,\omega_m$$

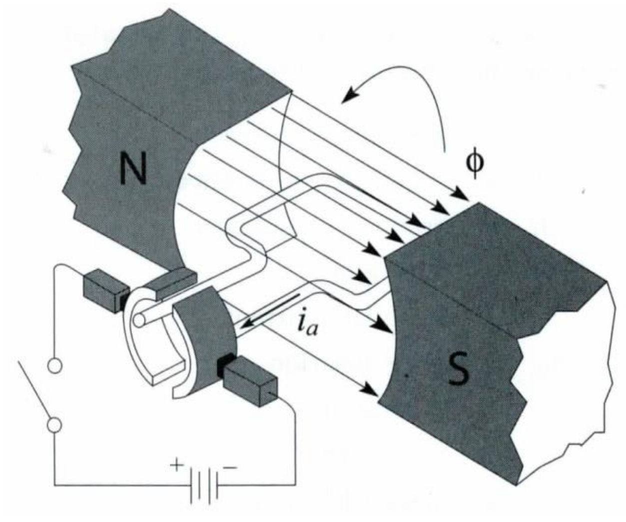
physical constant

motor velocity (rad/s)

motor velocity (rad/s)

(SHV Section 6.1)

DC Motor



Electrical Dynamics

$$V(t) = L\frac{di_a}{dt} + Ri_a + k_v\frac{d\theta_m}{dt}$$

Physical Dynamics

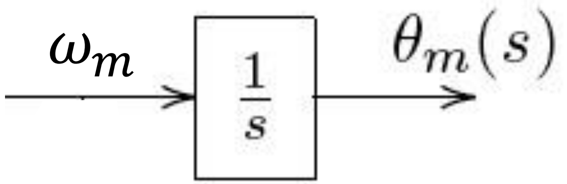
SHV shows the load torque in the wrong direction and confusingly calls gear ratio "r"

$$J_m\frac{d^2\theta_m}{dt^2} + B_m\frac{d\theta_m}{dt} = \tau_m + \tau_{ext} = k_t i_a + \tau_{ext}$$

external disturbances from connections

input

torque constant



electrical dynamics

motor torque

motor physics

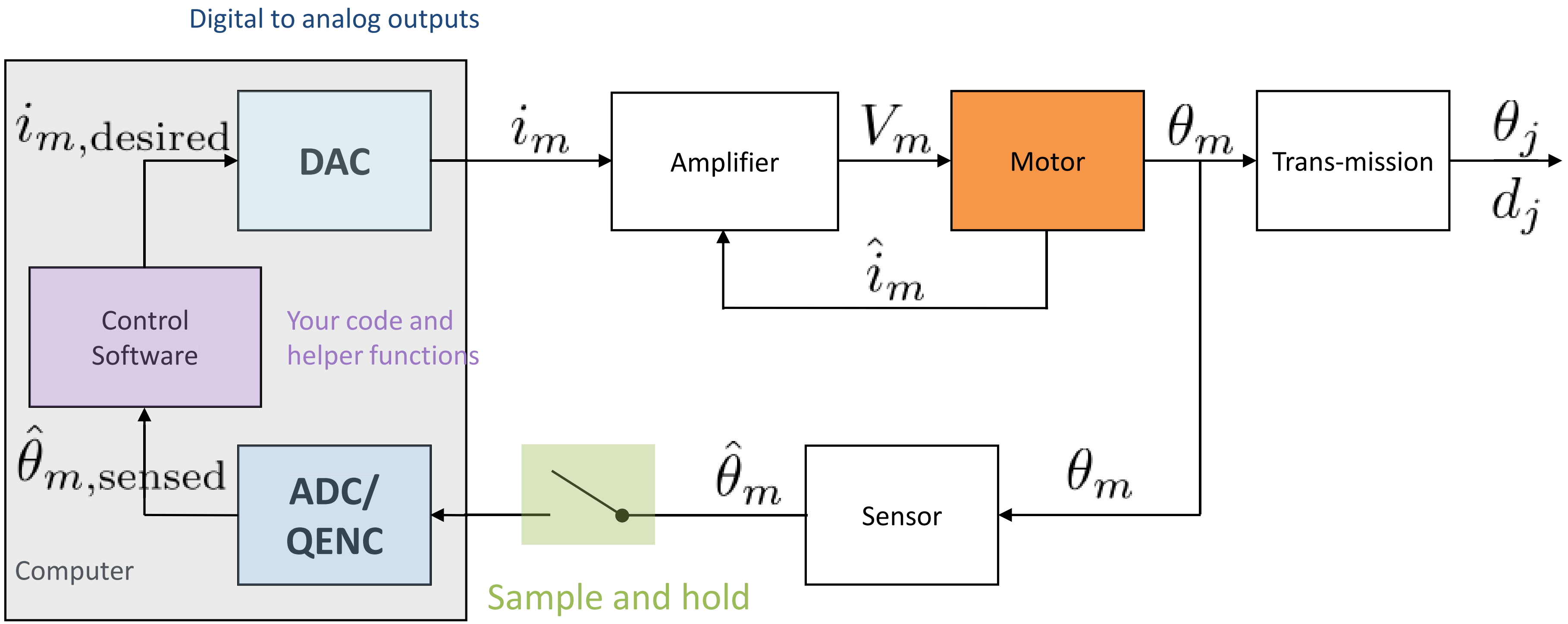
back emf constant

DC Motors

The best brushed DC motors are made by Maxon. They are rather expensive, but they work quite well.

- **Smooth torque output**, independent of motor angle. In other words, very low cogging and torque ripple.
- **Low friction**, both at low and high speeds, due to high quality bearings and low eddy currents.
- **Relatively high stall torque**, which is the torque the motor can deliver when it is not rotating.
- **Larger motors** create higher torques, but they also have higher inertia, higher friction, and higher cost.

How do we send signals to the motors?



From Lecture 18

sensor data to joint values

$$\vec{q}_k = \vec{a} * \vec{Q}_k + \vec{b}$$

joint values to position

$$\vec{x}_k = \Lambda(\vec{q}_k)$$

controller

$$\vec{\tau}_k = F_i(\vec{q}_k)$$

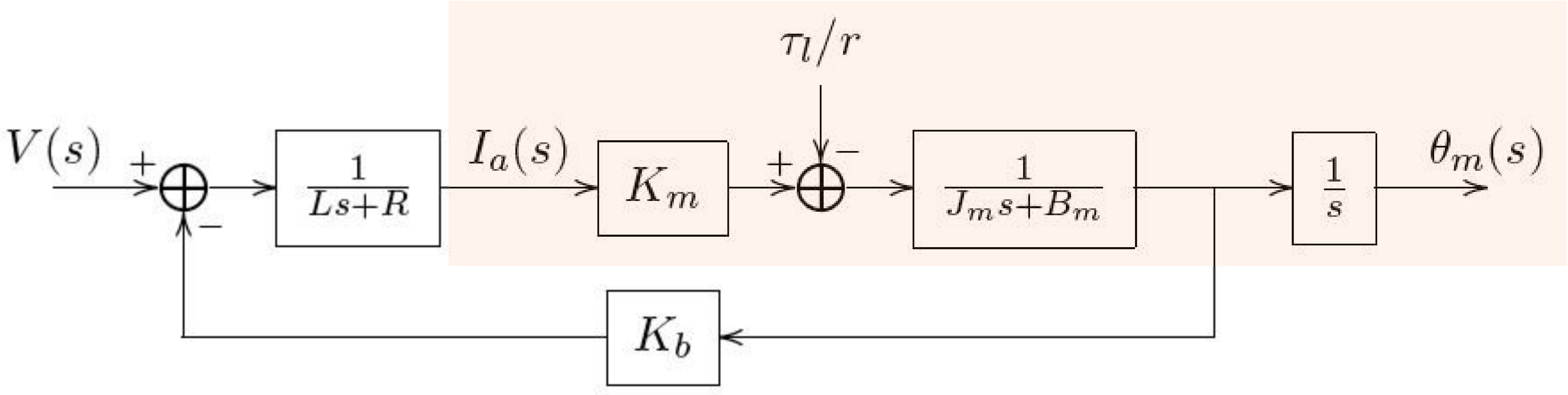
joint torques to control outputs

$$\vec{r}_k = \vec{c} * \vec{\tau}_k + \vec{d}$$

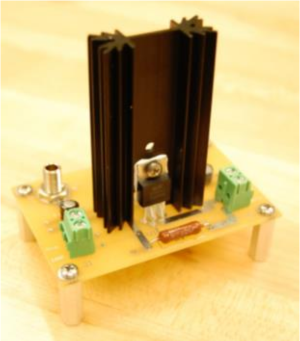
Current Amplifiers

$$\tau_m = k_t \, i_a$$

Motor torque is proportional to current, so if we can control current regardless of speed, we can ignore the motor's electrical dynamics (L, R, V_b).



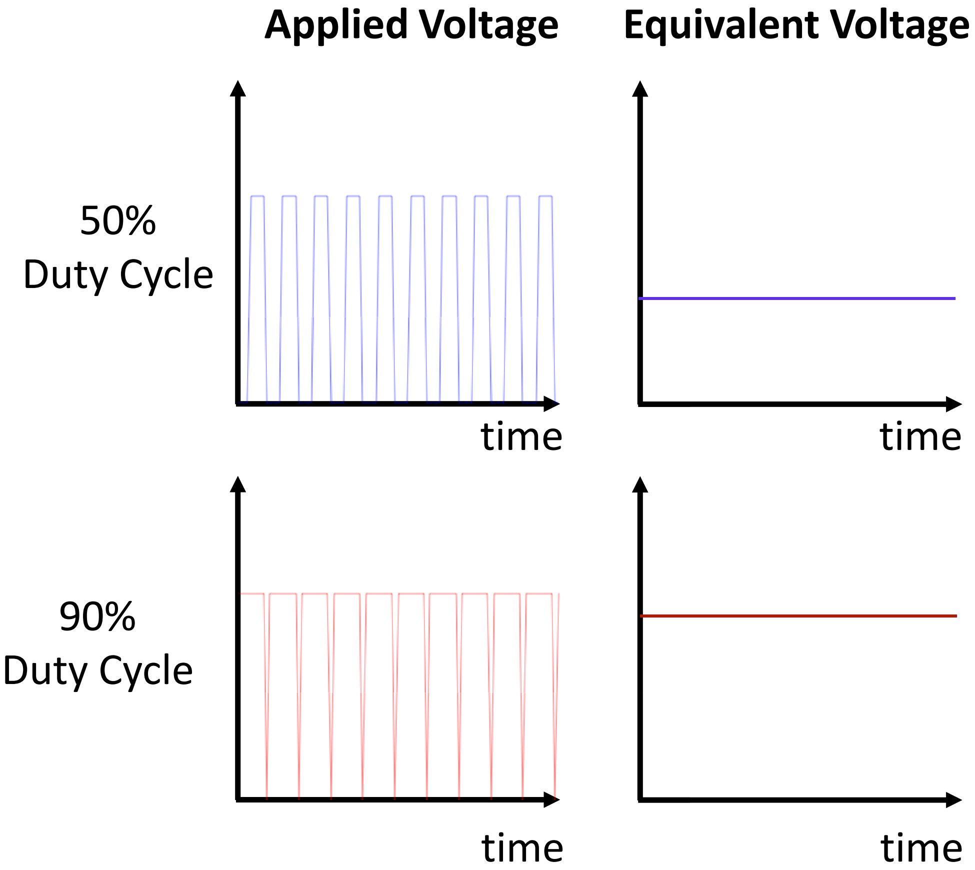
Current Amplifier



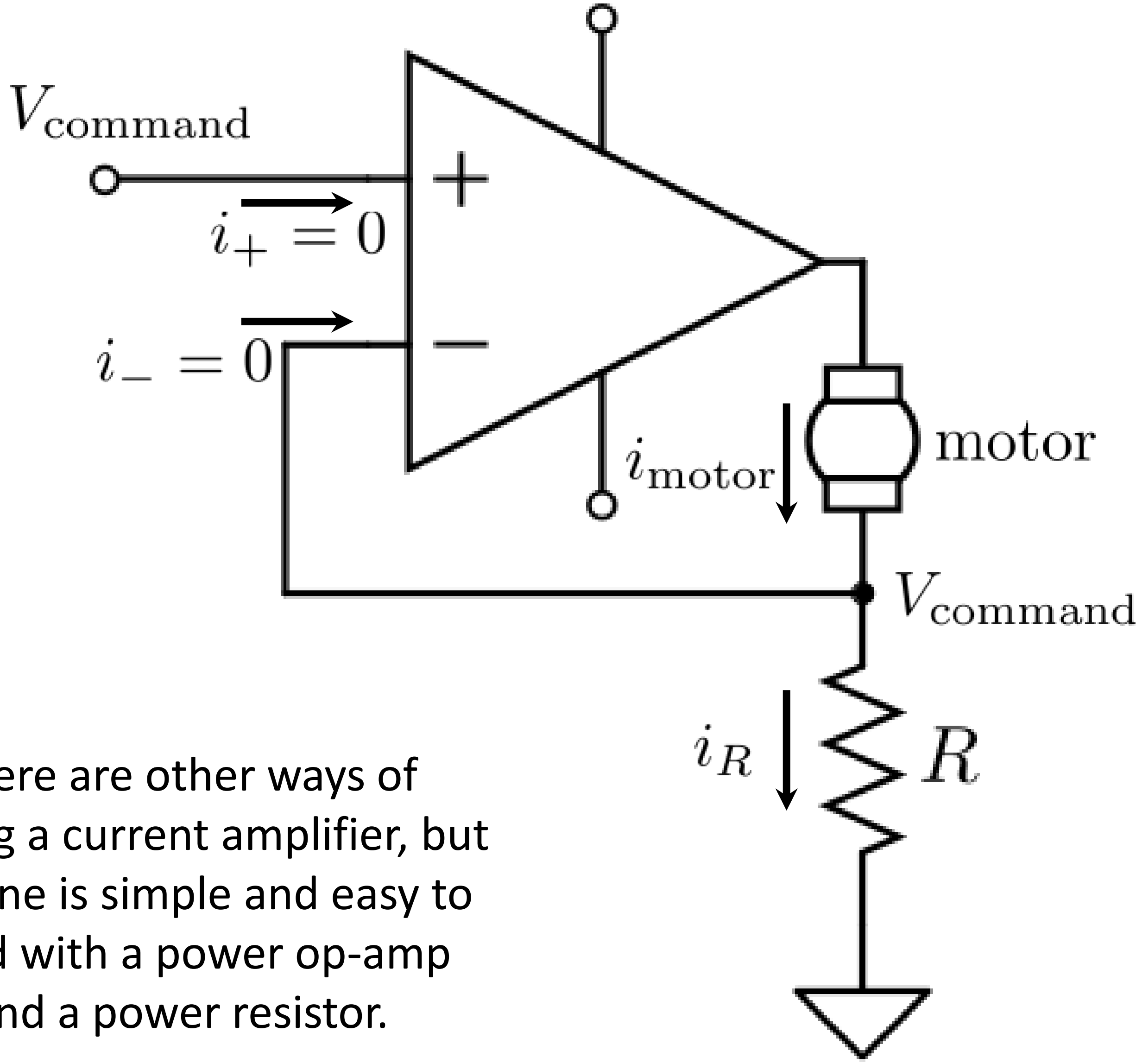
$$\tau_m = k_t \, i_a$$

Takes an information signal (usually an analog voltage) from the computer and drives the requested amount of current through the actuator.

Two common types are Pulse Width Modulation (PWM) and Linear. Linear amplifiers often have higher bandwidth and cause less electrical noise.



Current Amplifier Circuit



There are other ways of making a current amplifier, but this one is simple and easy to build with a power op-amp and a power resistor.

Operational amplifier in negative feedback, so it follows the two golden rules of op-amps:

No current enters or leaves the inverting or non-inverting inputs.

The voltages at the inverting and non-inverting inputs are equal.

KCL: $i_{\text{motor}} = i_R$

Ohm's Law: $i_R = \dfrac{V_{\text{command}}}{R}$

$$\boxed{i_{\text{motor}} = \frac{V_{\text{command}}}{R}}$$

Transmission

Most DC motors are designed for high-speed low-torque output.

In order to create robots that can bear load, we need a transmission system.

Common Transmissions

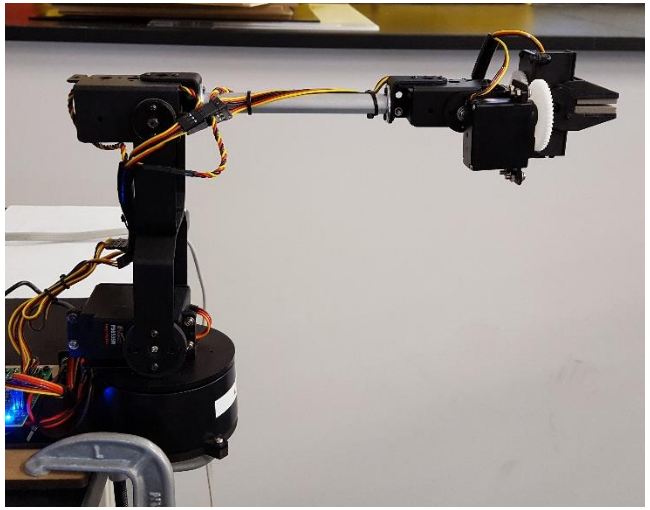
Direct Drive: simplest implementation

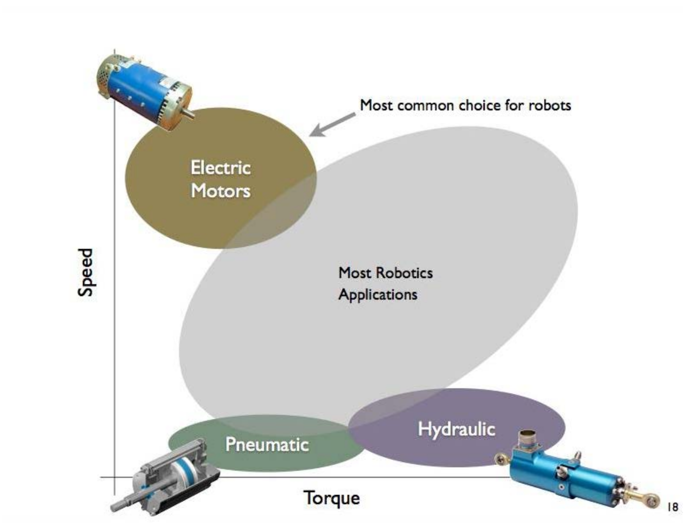
Band/Belt Drive:

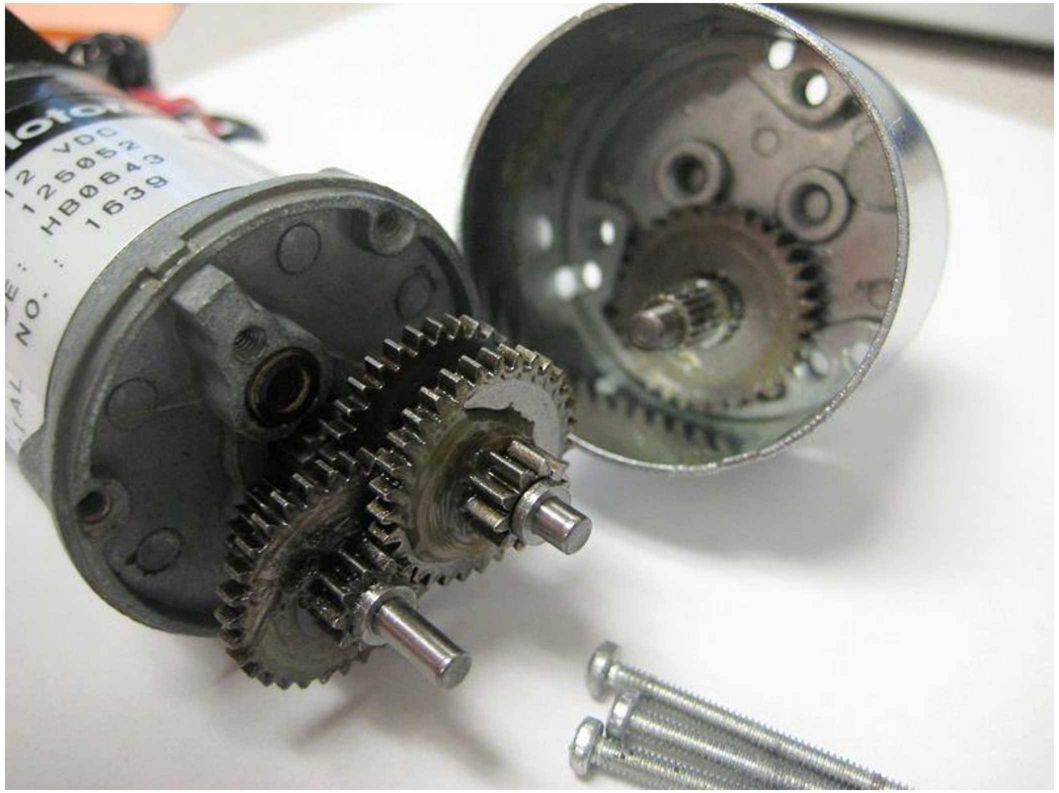
- move actuator mass away from joint
- smoothest drive but unstable when belts are long

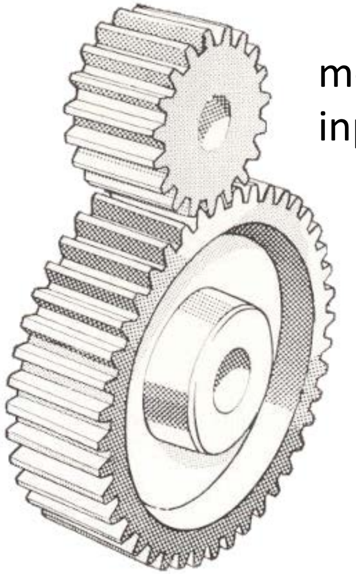
Gear Drive:

- high torque low speed
- backlash









motor input

load output

gear ratio

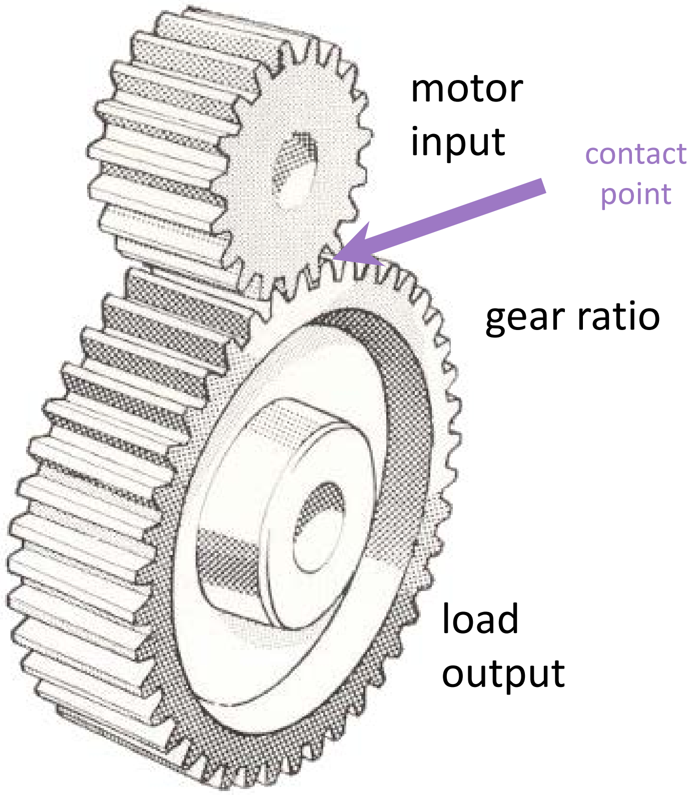
$$N = \frac{n_{\text{out}}}{n_{\text{in}}} = \frac{r_{\text{out}}}{r_{\text{in}}} = \frac{\omega_{\text{in}}}{\omega_{\text{out}}} = \frac{\tau_{\text{out}}}{\tau_{\text{in}}}$$

Teeth Radius Speed Torque

Same equations apply for belts, pulleys, and friction drive

$$N = \frac{n_{\text{out}}}{n_{\text{in}}} = \frac{r_{\text{out}}}{r_{\text{in}}} = \frac{\omega_{\text{in}}}{\omega_{\text{out}}} = \frac{\tau_{\text{out}}}{\tau_{\text{in}}}$$

Teeth Radius Speed Torque



Imagine the small gear rotates an angle equivalent to 5 gear teeth.

How far does the large gear rotate?

Imagine the you apply a torque of 1 Nm to the small gear while holding the large gear still.

What torque do you feel on the other gear?

Types of Gears

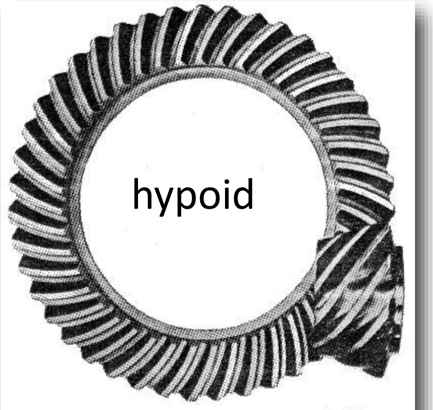
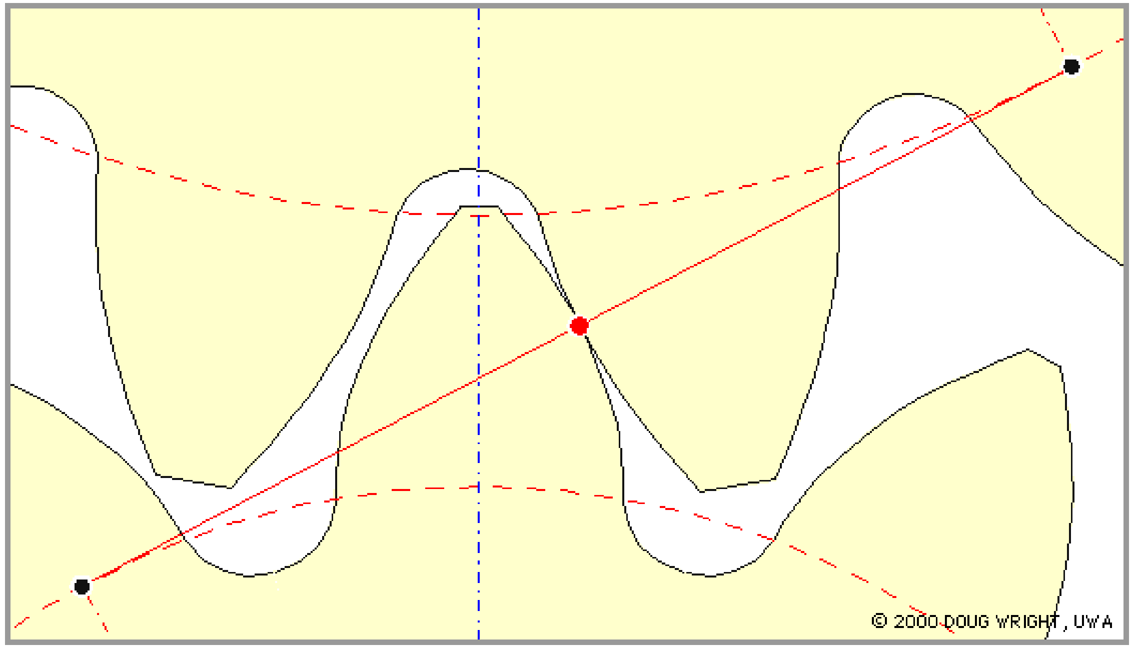
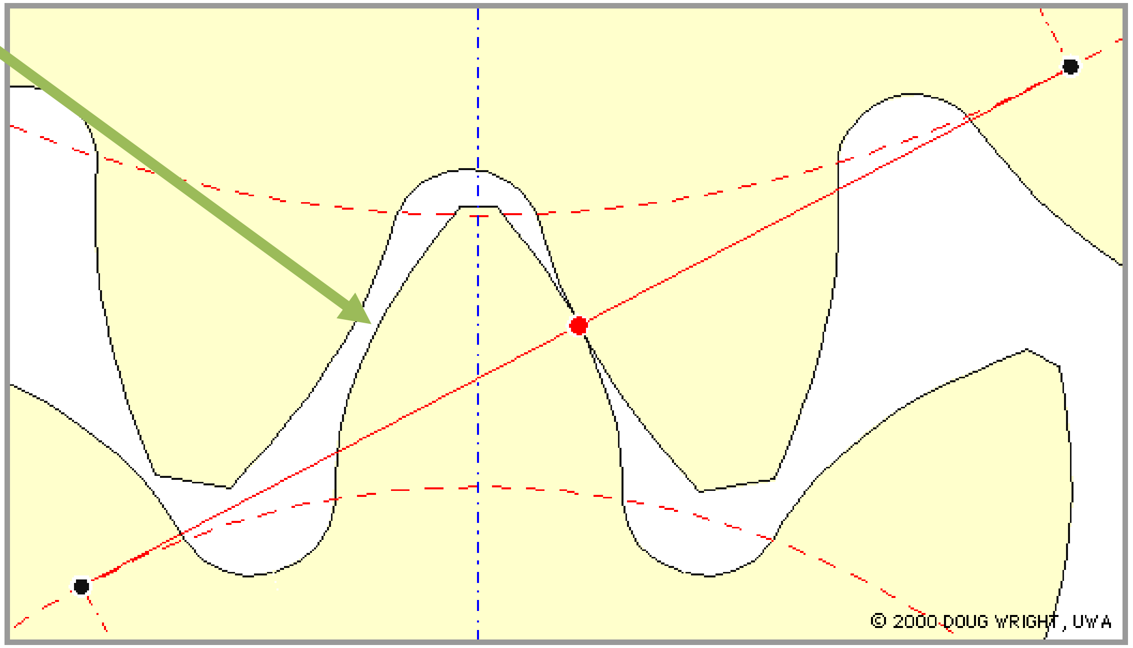


FIGURE 6.14 Hypoid gears. (Courtesy of Gleason Works.)

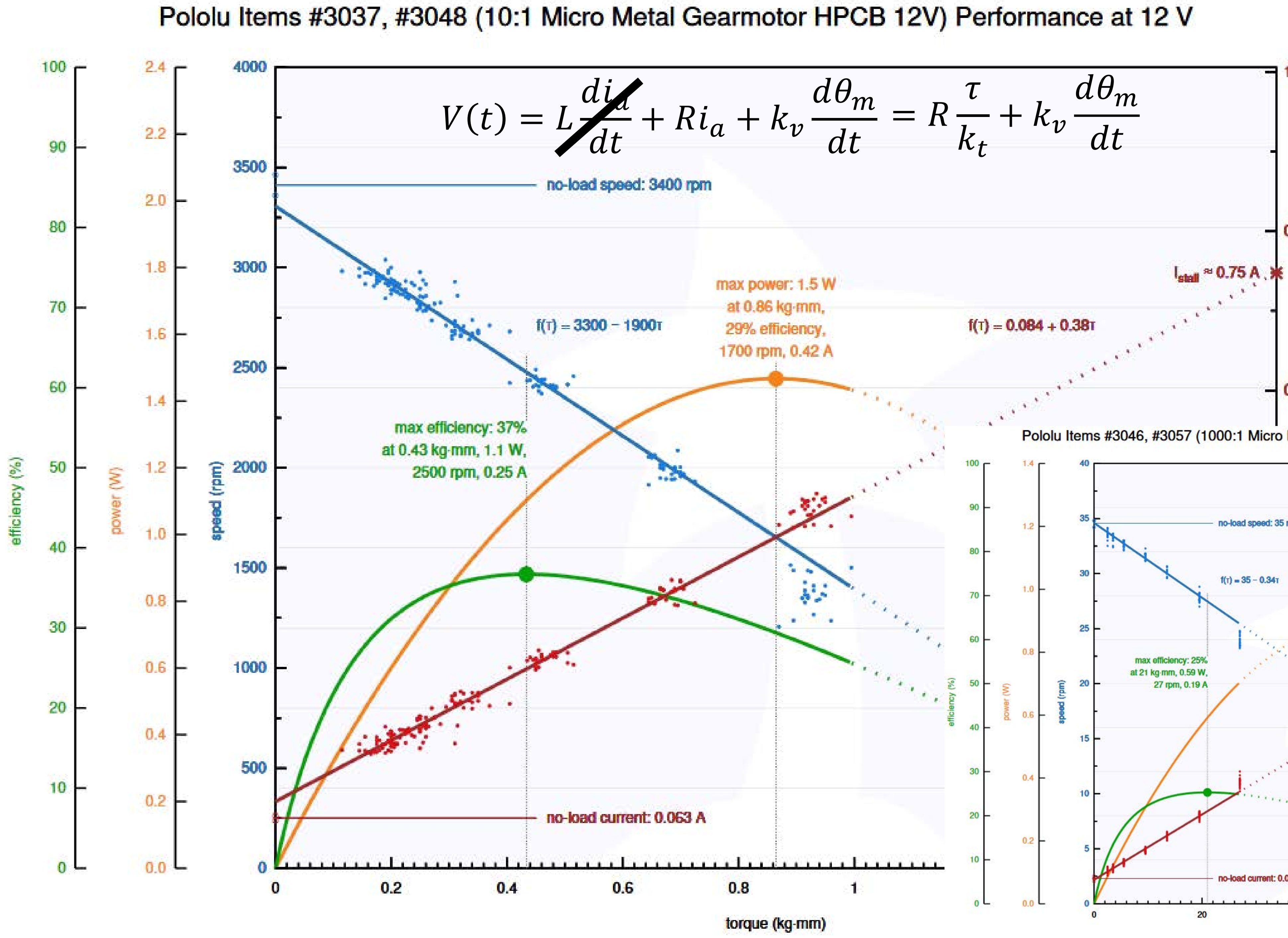
Backlash

Gap between teeth means that if the rotation changes direction, one gear can move a small amount without making the other move.

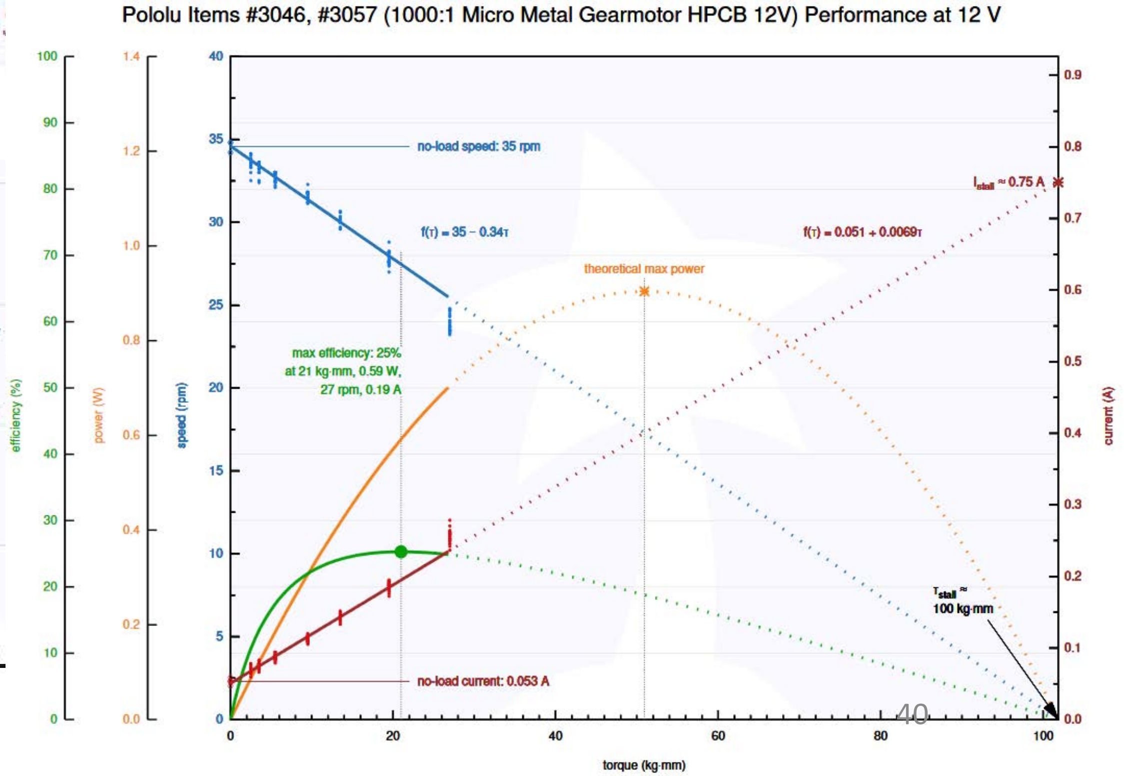




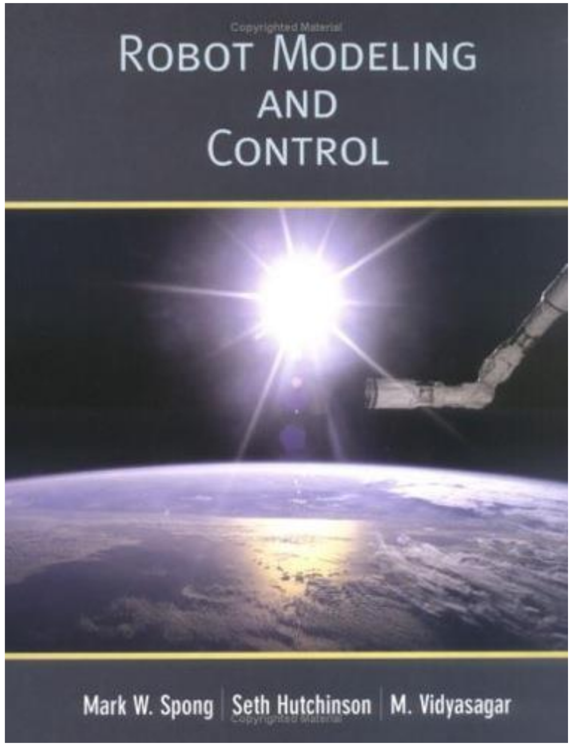
$$V(t) = L\cancel{\frac{di_a}{dt}} + Ri_a + k_v\frac{d\theta_m}{dt} = R\frac{\tau}{k_t} + k_v\frac{d\theta_m}{dt}$$



www.pololu.com

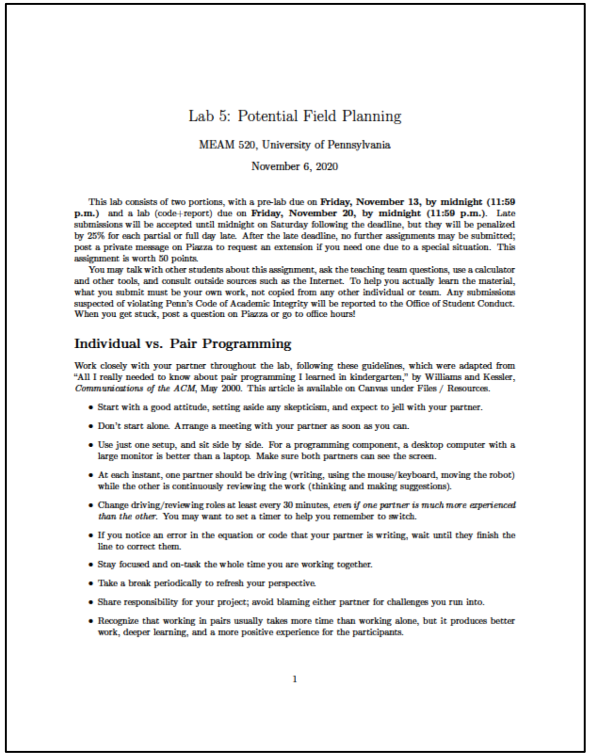


Upcoming: Control



Chapter 6: Joint Control

- Read 6.3



Lab 5: Potential Field Planning

MEAM 520, University of Pennsylvania

November 6, 2020

This lab consists of two portions, with a pre-lab due on **Friday, November 13, by midnight (11:59 p.m.)** and a lab (code+report) due on **Friday, November 20, by midnight (11:59 p.m.)**. Late submissions will be accepted until midnight on Saturday following the deadline, but they will be penalized by 25% for each partial or full day late. After the late deadline, no further assignments may be submitted; post a private message on Piazza to request an extension if you need one due to a special situation. This assignment is worth 50 points.

You may talk with other students about this assignment, ask the teaching team questions, use a calculator and other tools, and consult outside sources such as the Internet. To help you actually learn the material, what you submit must be your own work, not copied from any other individual or team. Any submissions suspected of violating Penn's Code of Academic Integrity will be reported to the Office of Student Conduct. When you get stuck, post a question on Piazza or go to office hours!

Individual vs. Pair Programming

Work closely with your partner throughout the lab, following these guidelines, which were adapted from "All I really needed to know about pair programming I learned in kindergarten," by Williams and Kessler, *Communications of the ACM*, May 2000. This article is available on Canvas under Files / Resources.

- Start with a good attitude, setting aside any skepticism, and expect to jell with your partner.
- Don't start alone. Arrange a meeting with your partner as soon as you can.
- Use just one setup, and sit side by side. For a programming component, a desktop computer with a large monitor is better than a laptop. Make sure both partners can see the screen.
- At each instant, one partner should be driving (writing, using the mouse/keyboard, moving the robot) while the other is continuously reviewing the work (thinking and making suggestions).
- Change driving/reviewing roles at least every 30 minutes, *even if one partner is much more experienced than the other*. You may want to set a timer to help you remember to switch.
- If you notice an error in the equation or code that your partner is writing, wait until they finish the line to correct them.
- Stay focused and on-task the whole time you are working together.
- Take a break periodically to refresh your perspective.
- Share responsibility for your project; avoid blaming either partner for challenges you run into.
- Recognize that working in pairs usually takes more time than working alone, but it produces better work, deeper learning, and a more positive experience for the participants.

1

Lab 5: Potential Fields due 11/20