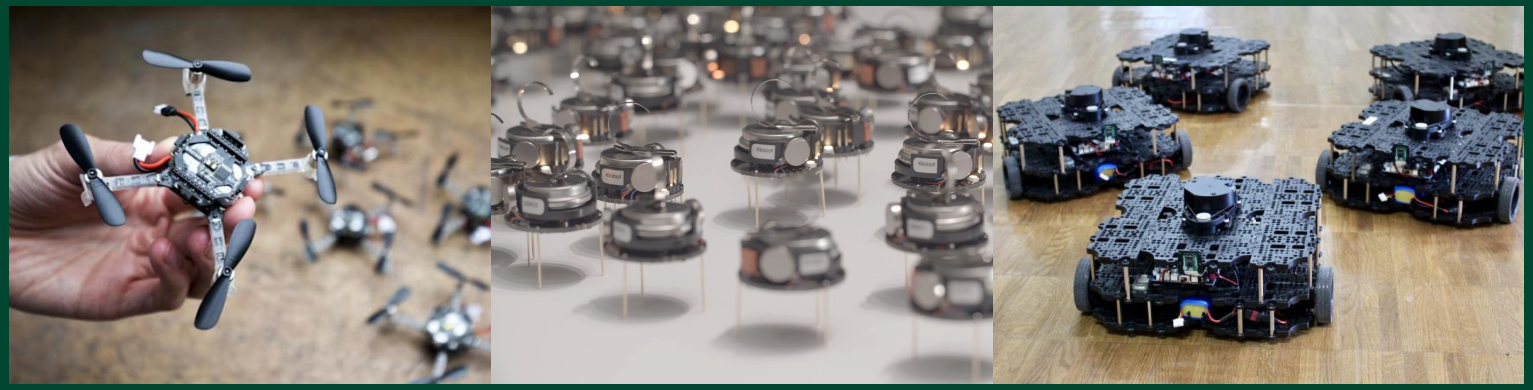
ECE693H, Spring 2025:
Multi-robot System Design

“Action 1”

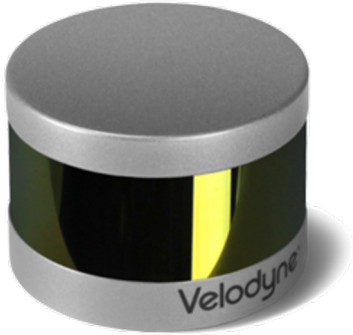


Dr. Daniel Drew

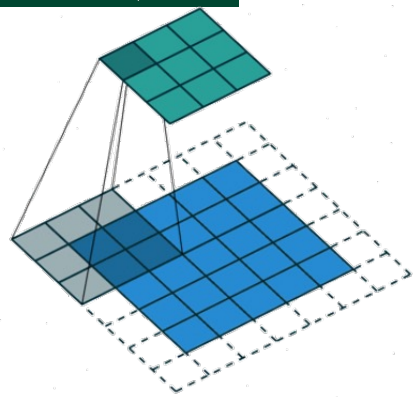
COLLEGE OF ENGINEERING
UNIVERSITY of HAWAI‘I at MĀNOA

Week 1 Recap

Action
(not just moving)

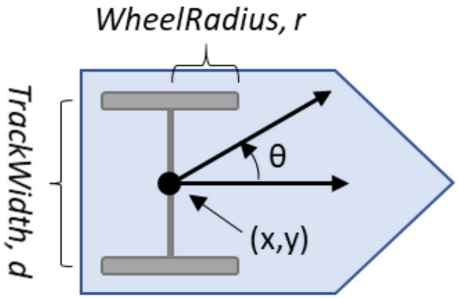


Perception
(not just sensing)



Intelligence
(not just classification)

Off-the-shelf research platforms let you work at high levels of the stack while abstracting out things like:



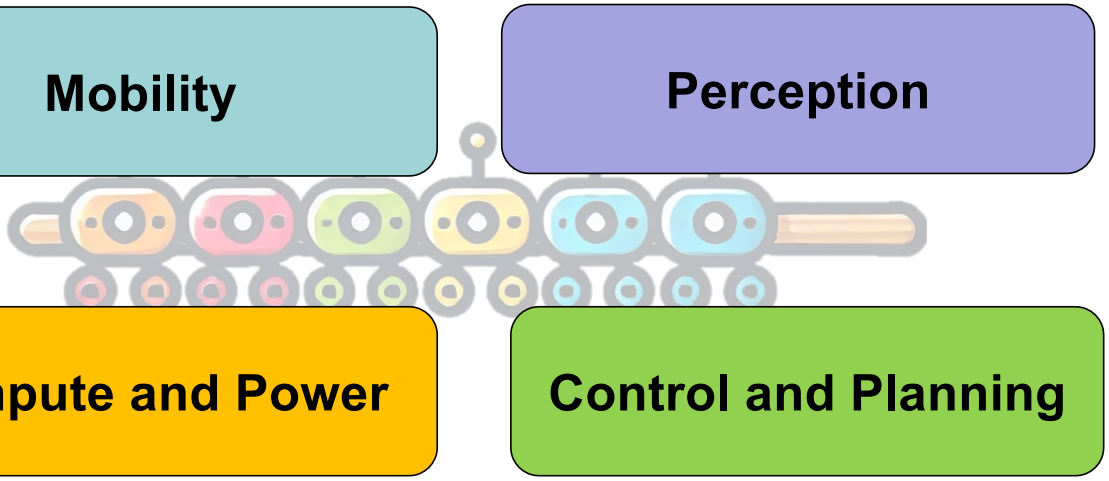
Swarm-on-a-Stick

We will create an open-source swarm robotics kit for education, from scratch.

693H Robotic Subsystems

Motors, drive electronics, encoders, PID controllers, top-down tracking

Sensor suite (IMU, prox.), calibration code, general IO (buttons and LEDs)



Battery and charging, wireless comms, GPIO and power distribution routing

microROS, ROS2, Gazebo simulation, enclosure design

Mechanics / Prototyping

Coding

PCB Design

Questions?

Welcoming New Students

Please share:

- Your name and grade level *(e.g., ECE senior)*
- How you heard about this course
- What you're excited about in this course
- What you're worried about with this course *(e.g., technical content, not enough programming experience)*

- Are neither, one, or both of these examples of robots?
- What makes something a robot?

Class Overview

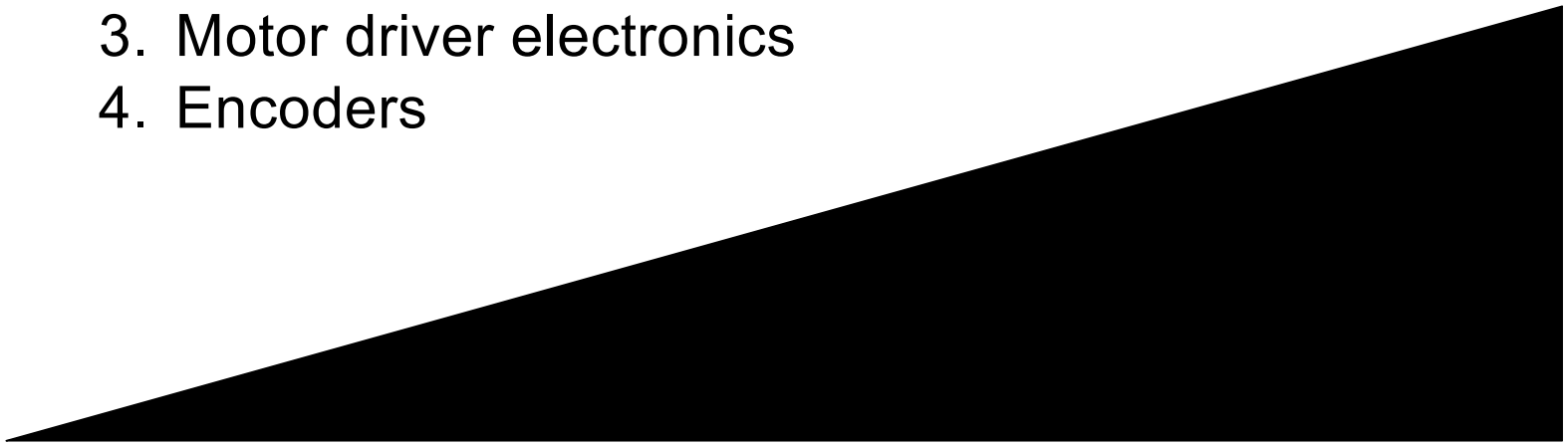
1. Lecture (~40 minutes)
2. Subsystem check-in (~5 minutes only)
3. FabLab training (~4:15pm)

Action: Lecture 1 of 2

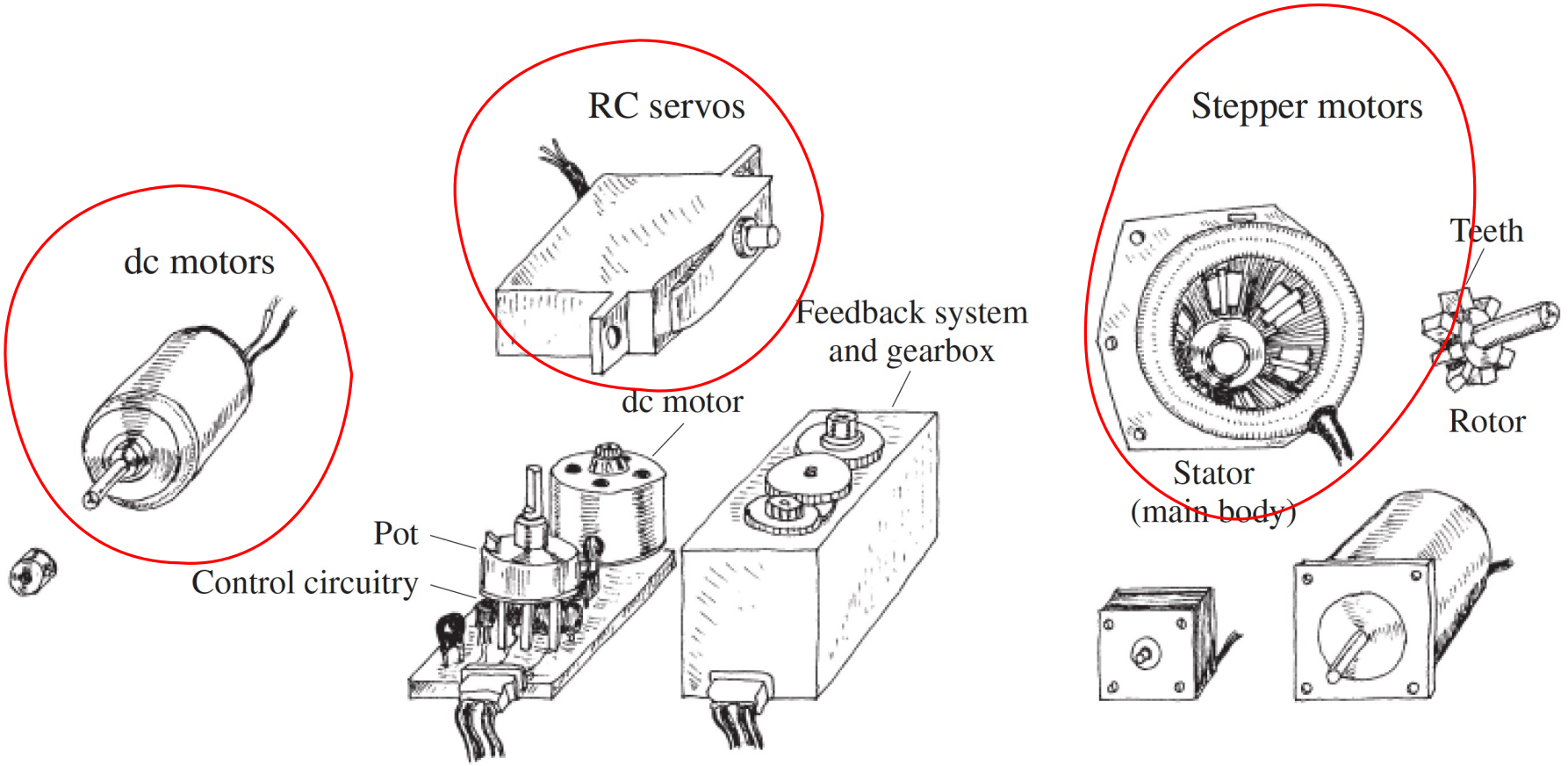
Lecture 1: motors → encoders
Lecture 2: motor control → motion planning

Lecture Overview: Ascending the Stack

1. Electromagnetic motors
2. Motor gearing and mechanics
3. Motor driver electronics
4. Encoders



Electromagnetic Motors: Types and Examples



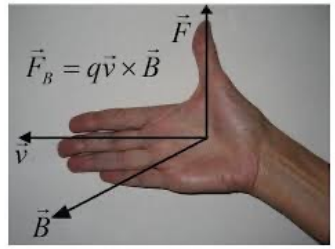
Electromagnetic Motors: Basic Physics

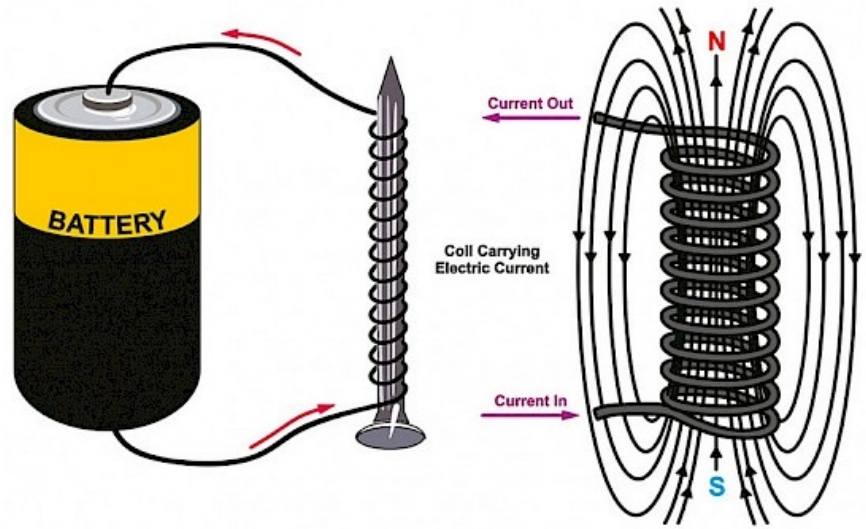
Ampere-Maxwell Law $\oint B\, dl = \mu_0 I + \mu_0 \epsilon_0 \int \frac{\partial E}{\partial t} dA$

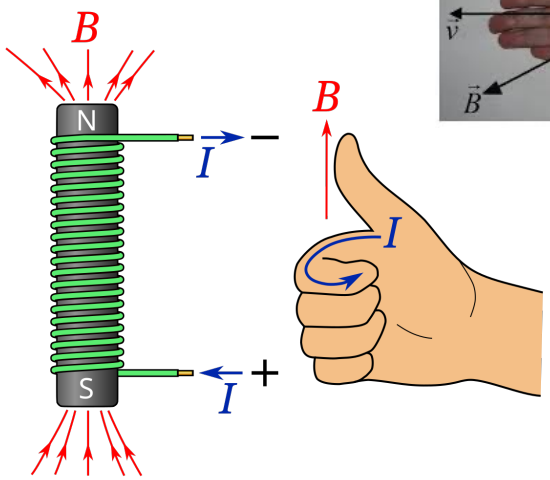
Lorentz force

microscopic $F = q(E + v \times B)$

macroscopic, net force on current-carrying conductor $F = I\, L\, B \qquad I = nqvA$







(curl right hand rule for electromagnets)

Electromagnetic Motors: Basic Physics

"back emf" constant *motor torque constant*

$$I_a = \frac{V - k_e\omega}{R_a} \qquad T = k_t I_a$$

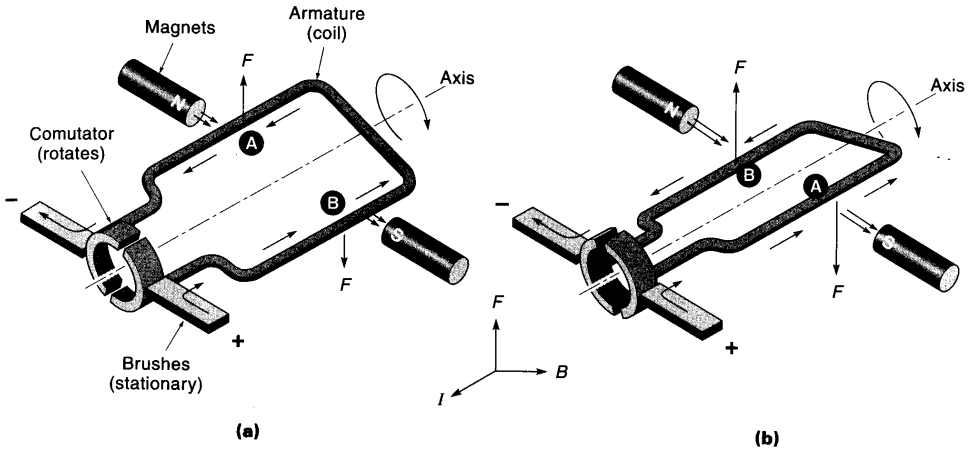
armature current **armature torque**

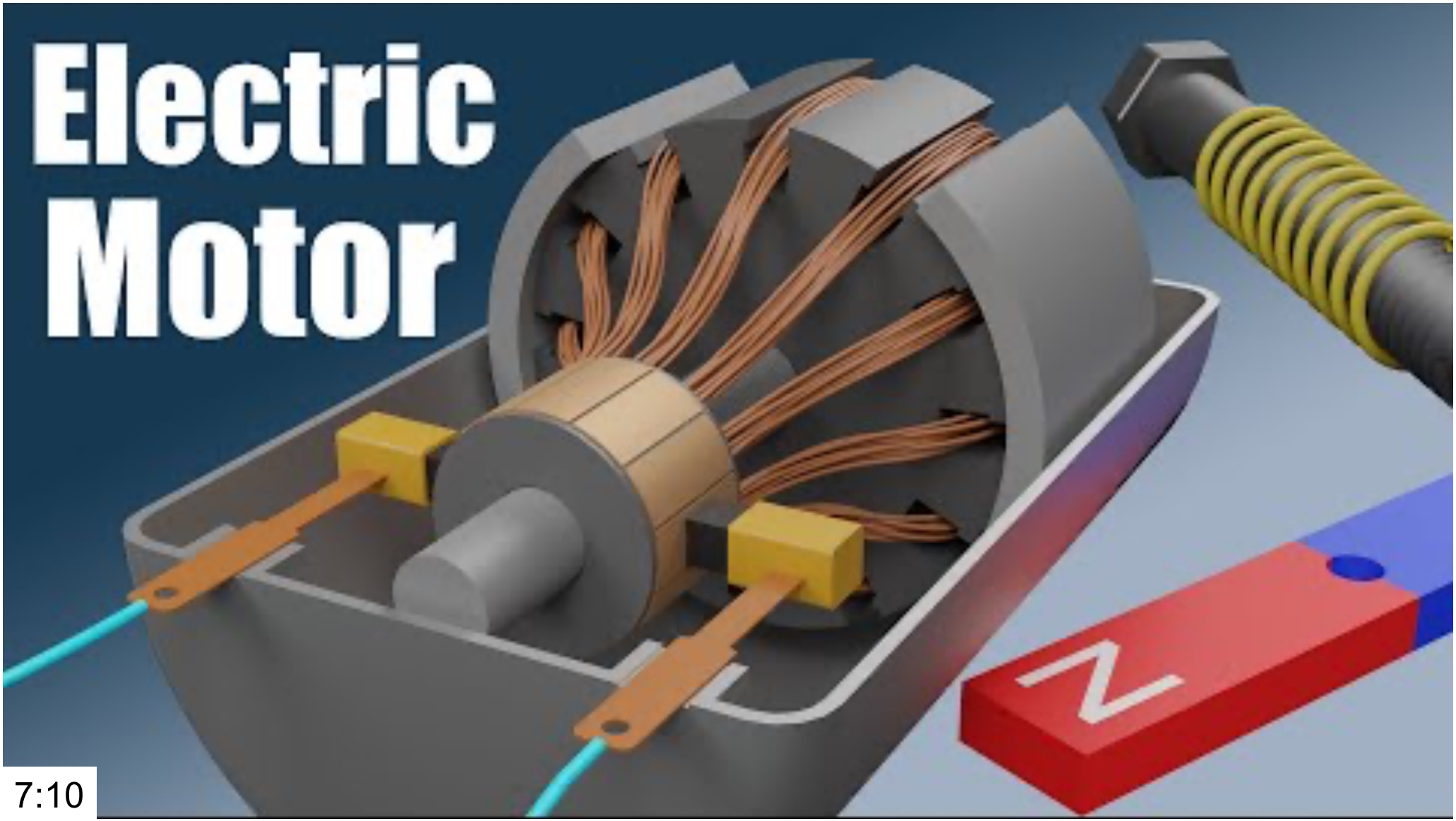
$$J\frac{d\omega}{dt} + (b + \frac{k_t k_e}{R_a})\omega = \frac{k_t}{R_a}V - T_{load}$$

full equation of motion

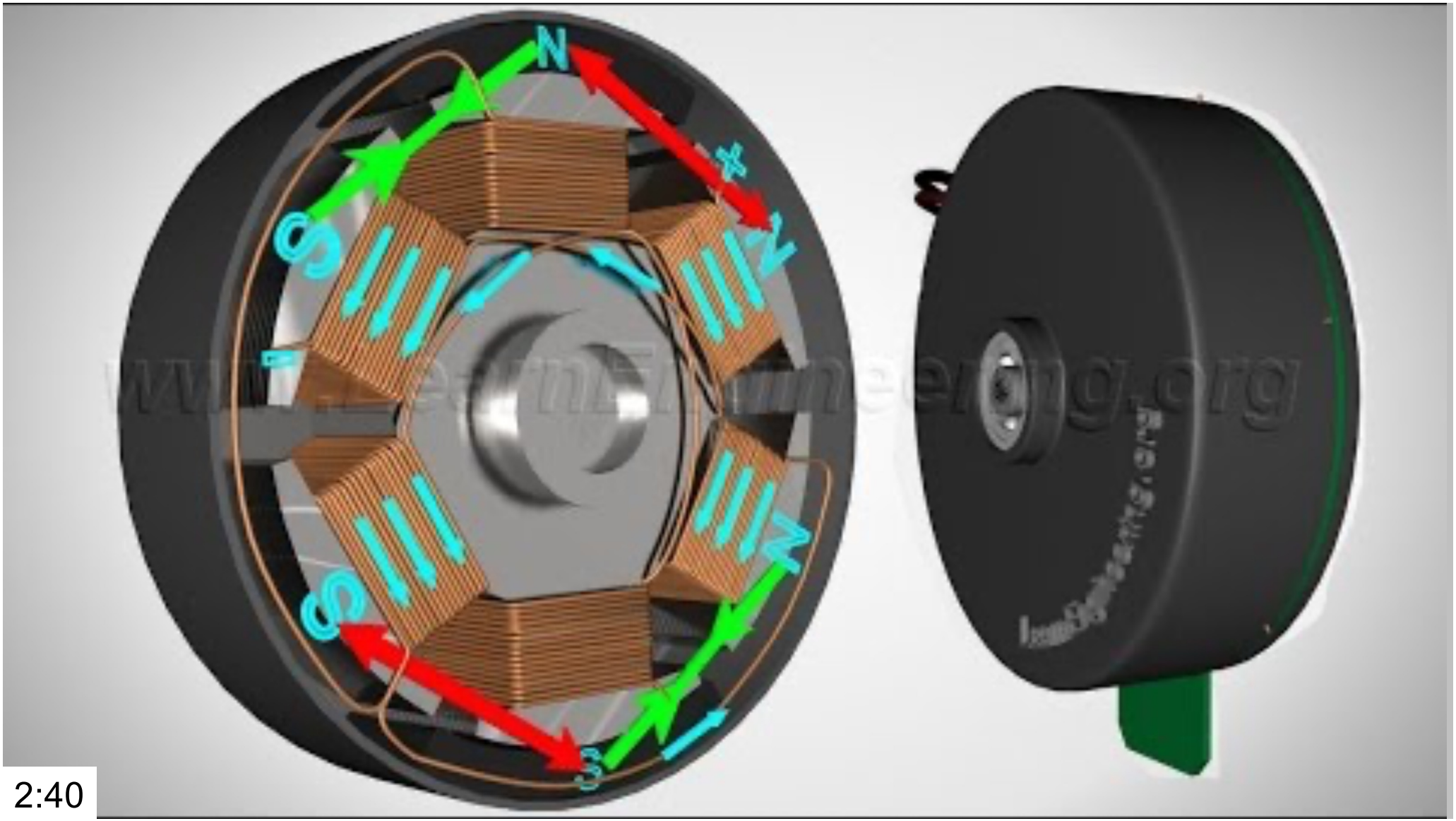
$$\omega = \frac{\frac{k_t}{R_a}V - T_{load}}{\frac{k_t k_e}{R_a}}$$

steady state velocity




Electric
Motor
N
7:10

Electromagnetic Motors: Brushed vs Brushless



2:40

Electromagnetic Motors: Brushed vs Brushless

Rotor vs stator: In brushed motors, the rotor has the electromagnet and the stator has permanent magnets. Brushless are the opposite.

Physical contact vs control complexity: Brushed motors use physical contact for commutation. Brushless motors require electronic controller to time the switching.

High performance vs cost and integration simplicity: Brushless motors are better by every metric, but are more expensive and more complex to integrate.

Brushed

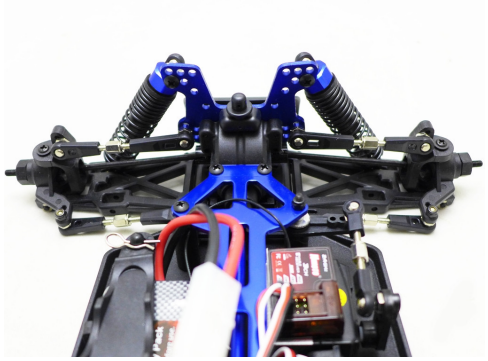
Brushless

Electromagnetic Motors: Servos

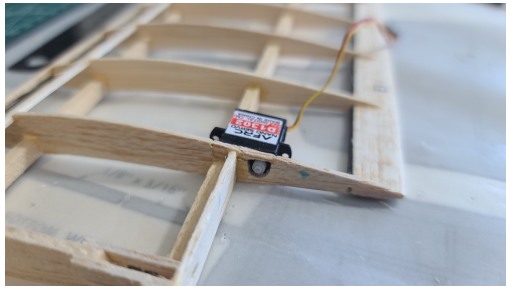
DC motor

RC servo

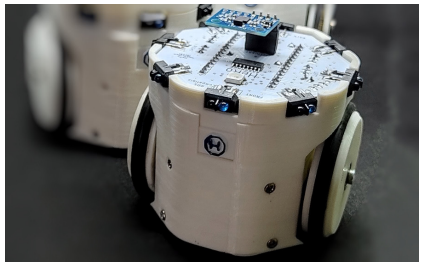
Electromagnetic Motors: Servos



steering



flap control



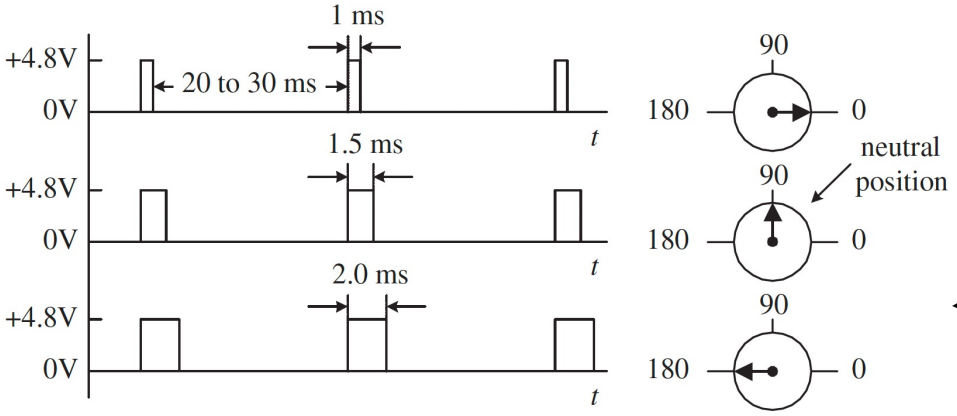
main drive
(continuous servo)

RC servo

Electromagnetic Motors: Servos

```arduino
#include <Servo.h>

Servo myservo;  // create servo object to control a servo
// twelve servo objects can be created on most boards

int pos = 0;    // variable to store the servo position

void setup() {
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservo.write(pos);              // tell servo to go to position in variable 'pos
    delay(15);                       // waits 15ms for the servo to reach the positio
  }
  for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);              // tell servo to go to position in variable 'pos
    delay(15);                       // waits 15ms for the servo to reach the positio
  }
}
```

typical use



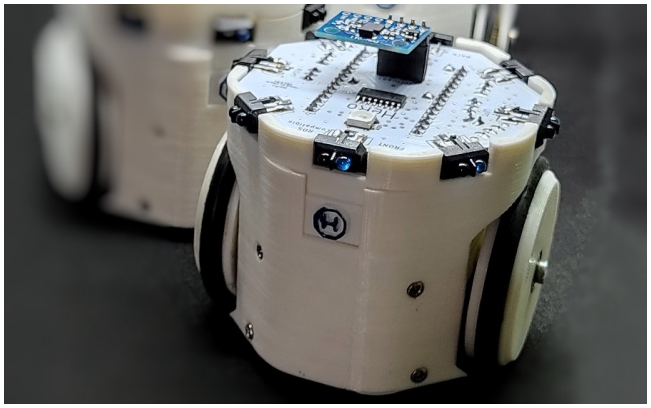
what's actually happening?

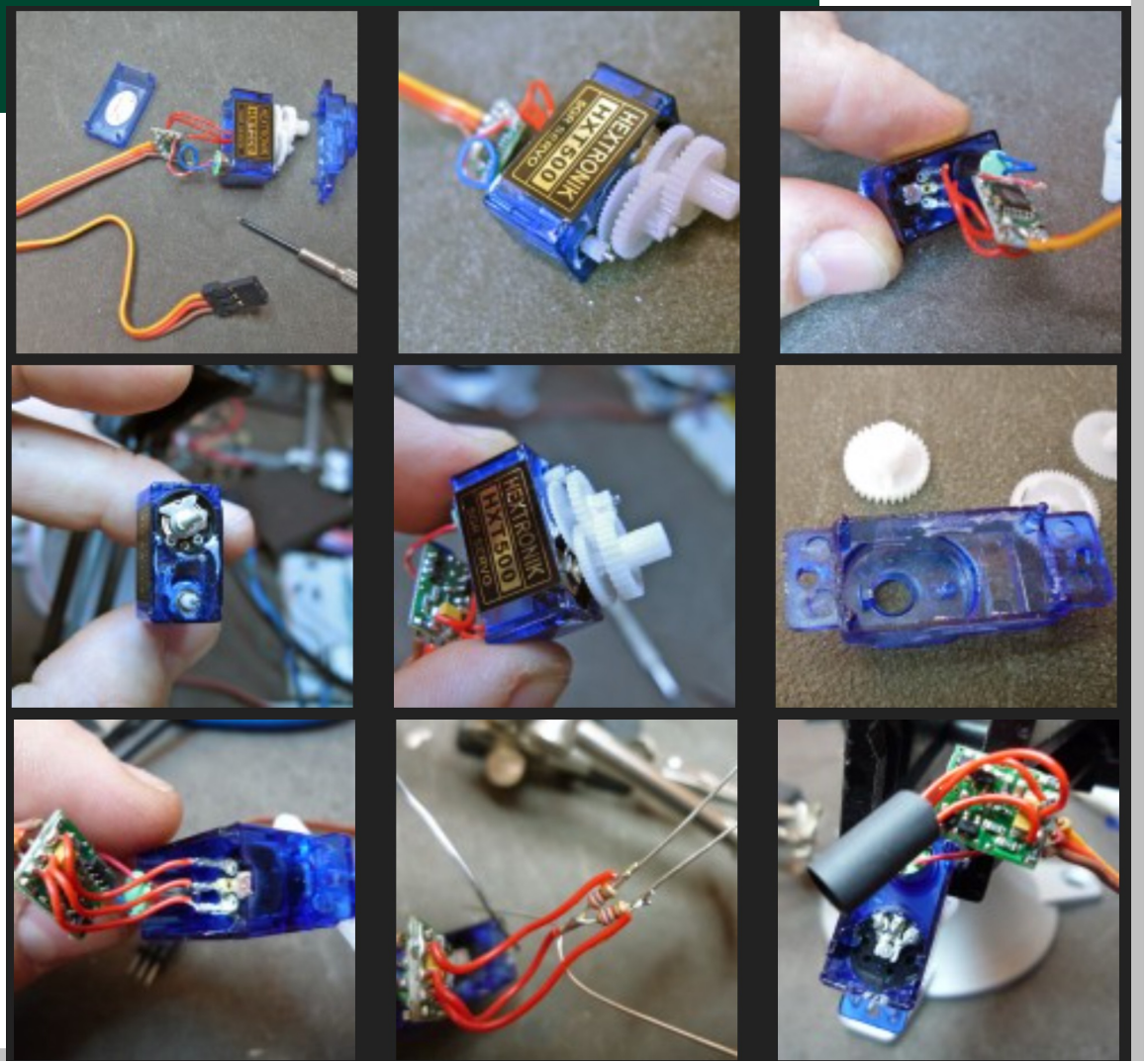
but what's ACTUALLY happening?


How servo
motors
really work
4:10

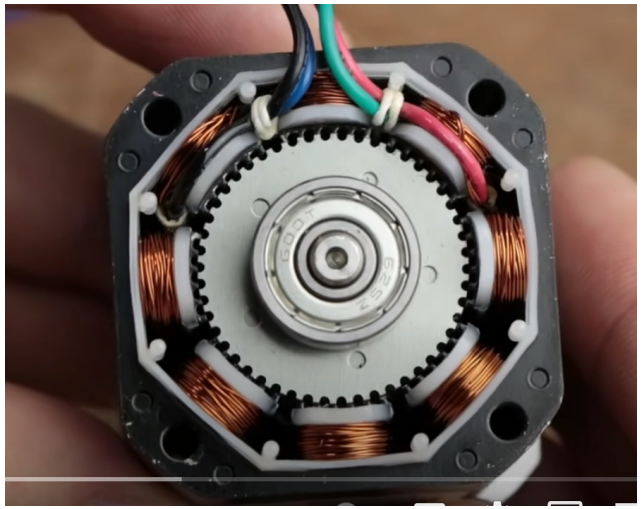
Servo Hacking

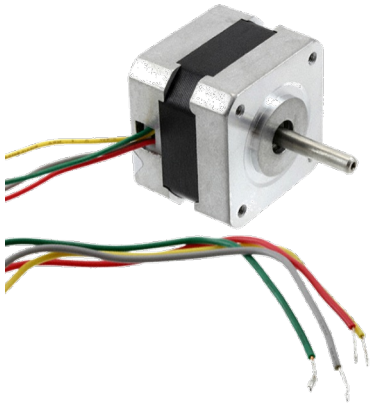
as seen in…

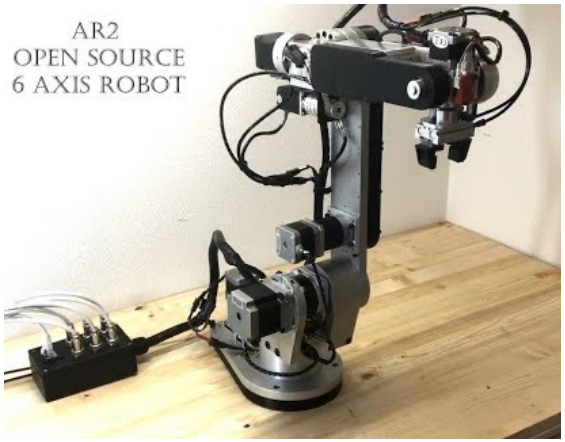


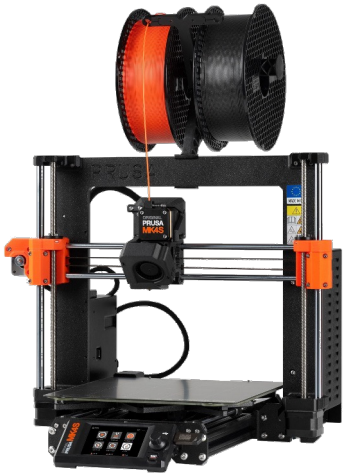


Electromagnetic Motors: Steppers

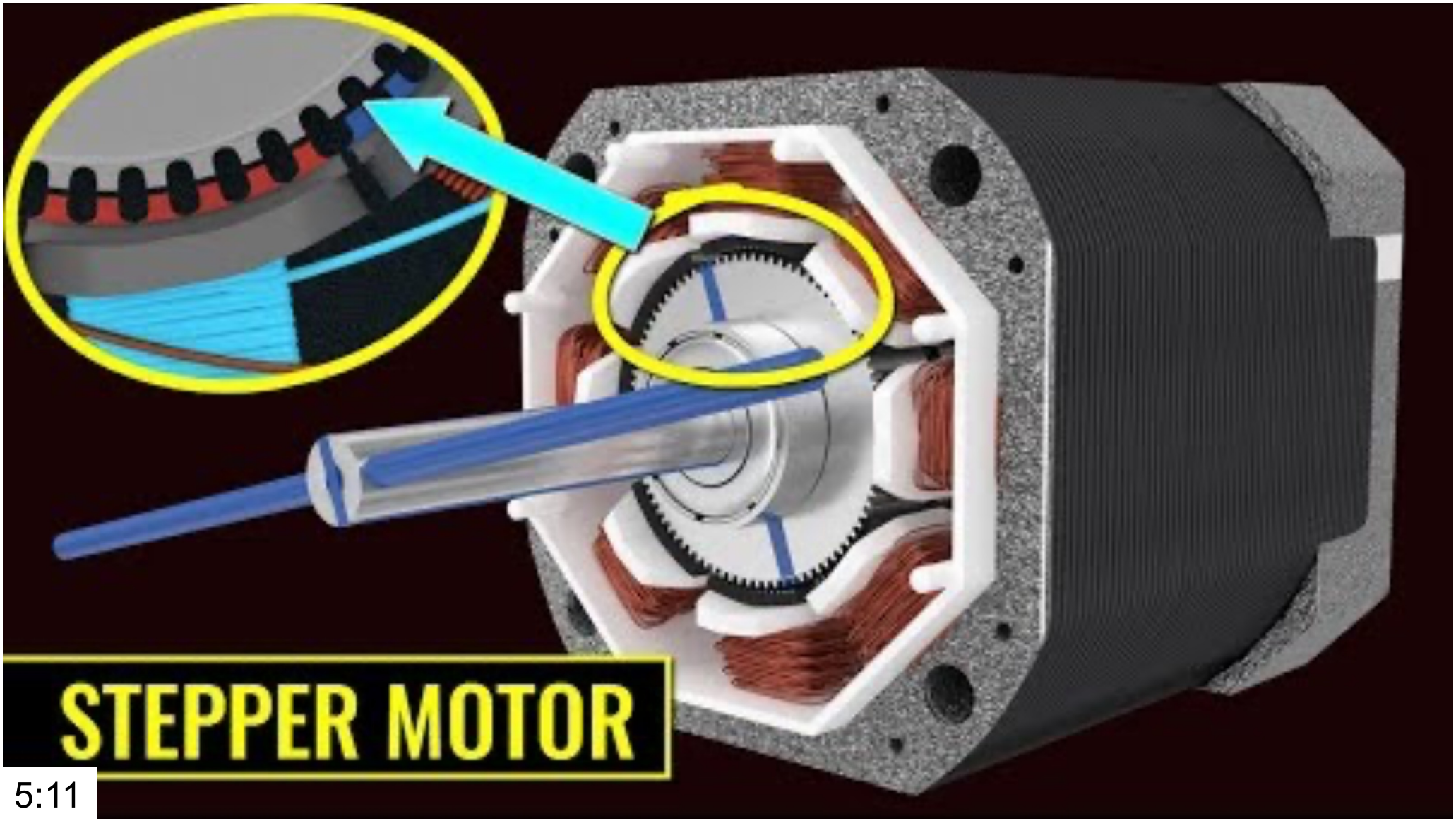






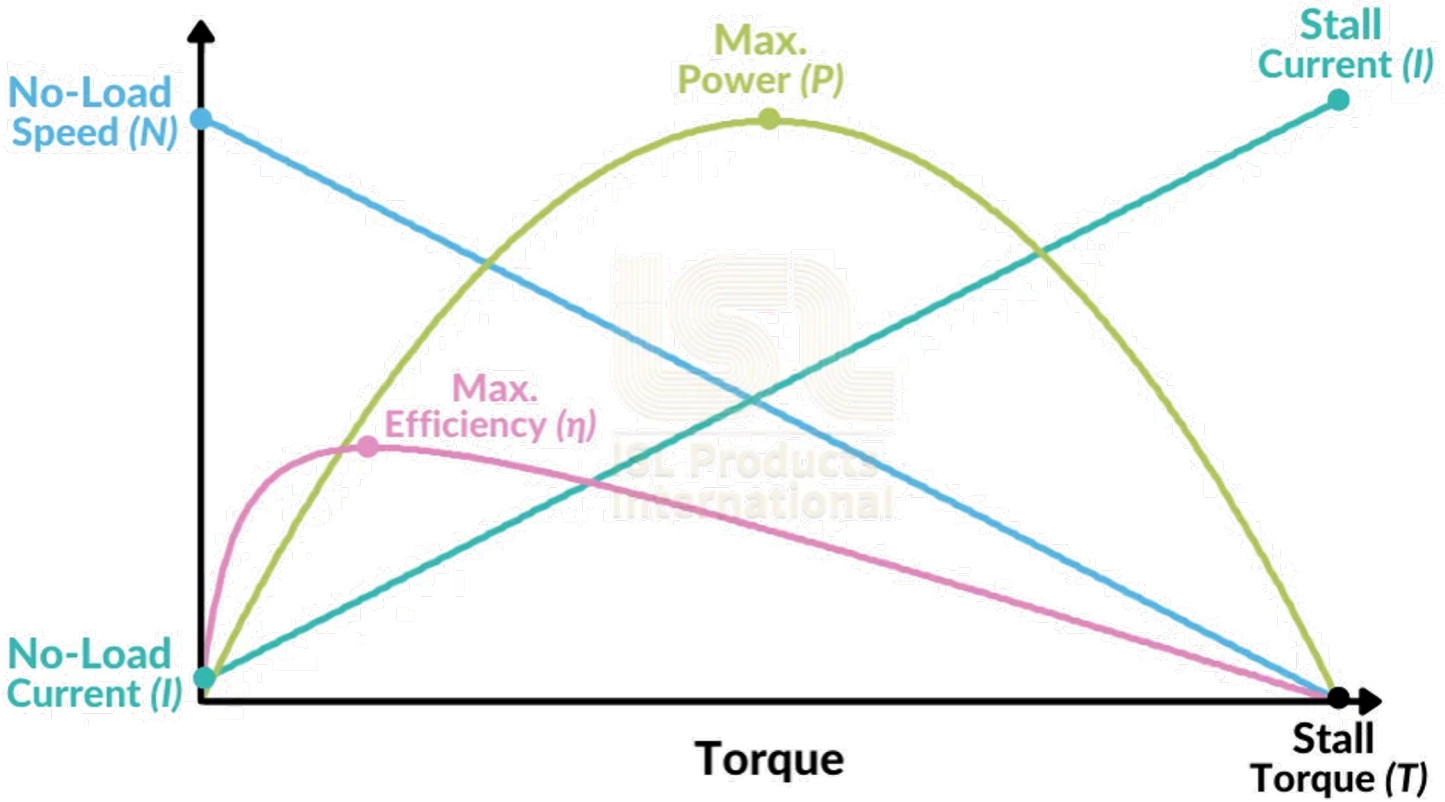


very small step sizes, high torque, “easy” to control precisely, relatively cheap for performance you get



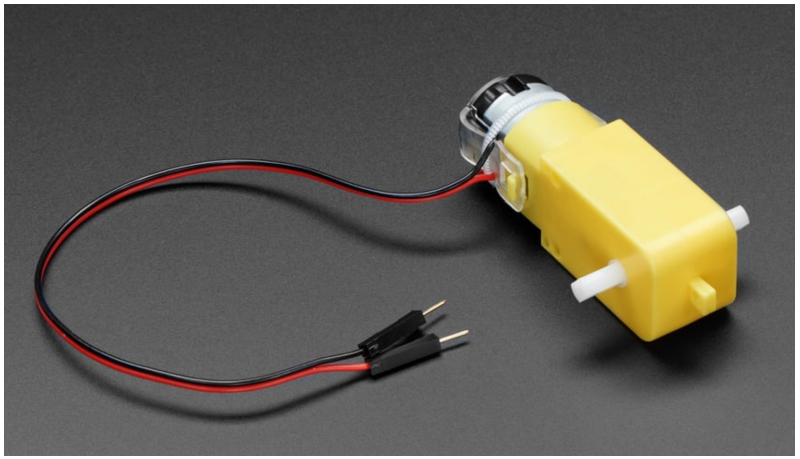
5:11

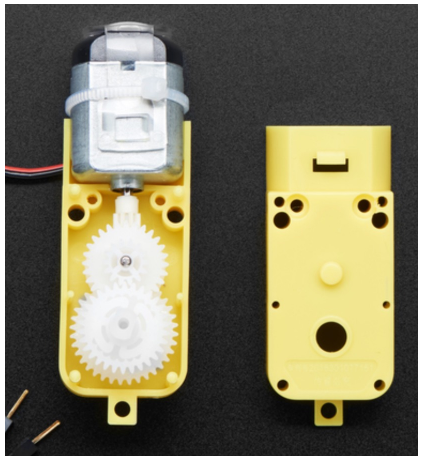
DC Motor Gearing



(recommended to operate dc motor at no more than 25% of stall current)

DC Motor Gearing





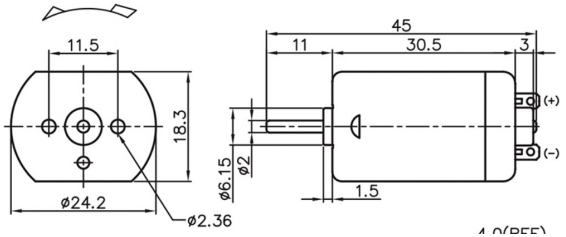
No load speed: 6600rpm
Stall current: 0.8A。

These durable (but affordable!) plastic gearbox motors (also known as 'TT' motors) are an easy, low-cost way to get your projects moving. This is a **TT DC Gearbox Motor** with a gear ratio of **1:48**, and it comes with 2 x 200mm wires with breadboard-friendly 0.1" male connectors. Perfect for plugging into a breadboard or terminal blocks.

You can power these motors with 3VDC up to 6VDC, they'll of course go a little faster at the higher voltages. We grabbed one motor and found these stats when running it from a bench-top supply

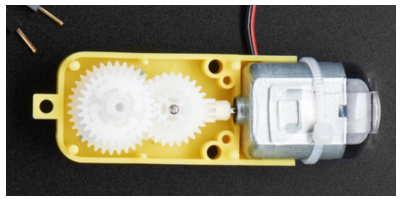
- At **3VDC** we measured 150mA @ 120 RPM no-load, and 1.1 Amps when stalled.
- At **4.5VDC** we measured 155mA @ 185 RPM no-load, and 1.2 Amps when stalled
- At **6VDC** we measured 160mA @ 250 RPM no-load, and 1.5 Amps when stalled.

DC Motor Gearing



Torque requirement for 500g robot, driving 0.5m/s, with 2.5cm radius wheels, is on the order of 13mNm at ~200RPM

No load speed: 6600rpm
Stall current: 0.8A。

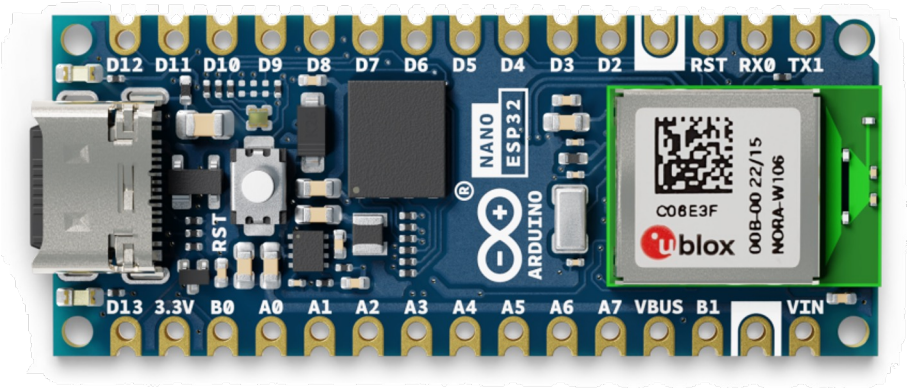


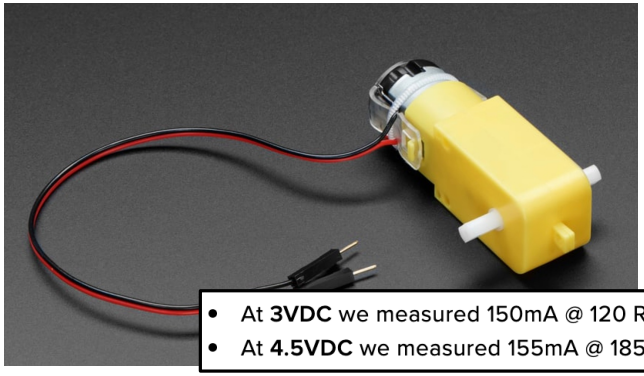
1:48 gear ratio

Stall current = ~10x hobby motor
Max eff. torque / 10 = 0.5mNm
6600 RPM / 48 = ~140 RPM
0.5mNm * 48 = ~24 mNm
25% of 24 = 6mNm
Two motors => 12mNm total torque

Motor Model	NFP-PC-280-6453P
Voltage Range	3V – 4.5V
Nominal Voltage	3.6V DC
No Load Speed	12,000RPM
No Load Current	0.48A
Max Efficiency Speed	10,000RPM
Max Efficiency Current	2.2A
Max Efficiency Torque	52g.cm
Max Efficiency Torque	5.1mN.m
Max Efficiency Output	5.39W
Max Efficiency	70%
Stall Current	11.62A
Stall Torque	330g.cm
Stall Torque	32.35mN.m

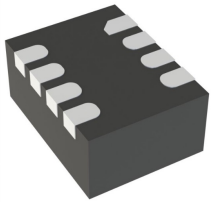
Motor Drivers: H-Bridges – Motivation





- At **3VDC** we measured 150mA @ 120 RPM no-load, and 1.1 Amps when stalled.
- At **4.5VDC** we measured 155mA @ 185 RPM no-load, and 1.2 Amps when stalled

How would we power the motor?



MP2322GQH-Z

DigiKey Part Number	1589-MP2322GQH-ZTR-ND - Tape 1589-MP2322GQH-ZCT-ND - Cut Ta 1589-MP2322GQH-ZDKR-ND - Digi
Manufacturer	Monolithic Power Systems Inc.
Manufacturer Product Number	MP2322GQH-Z
Description	IC REG BUCK ADJ 1A 8QFN
Manufacturer Standard Lead Time	12 Weeks
Current - Output	1A

11.5 Using the 3.3 V Pin

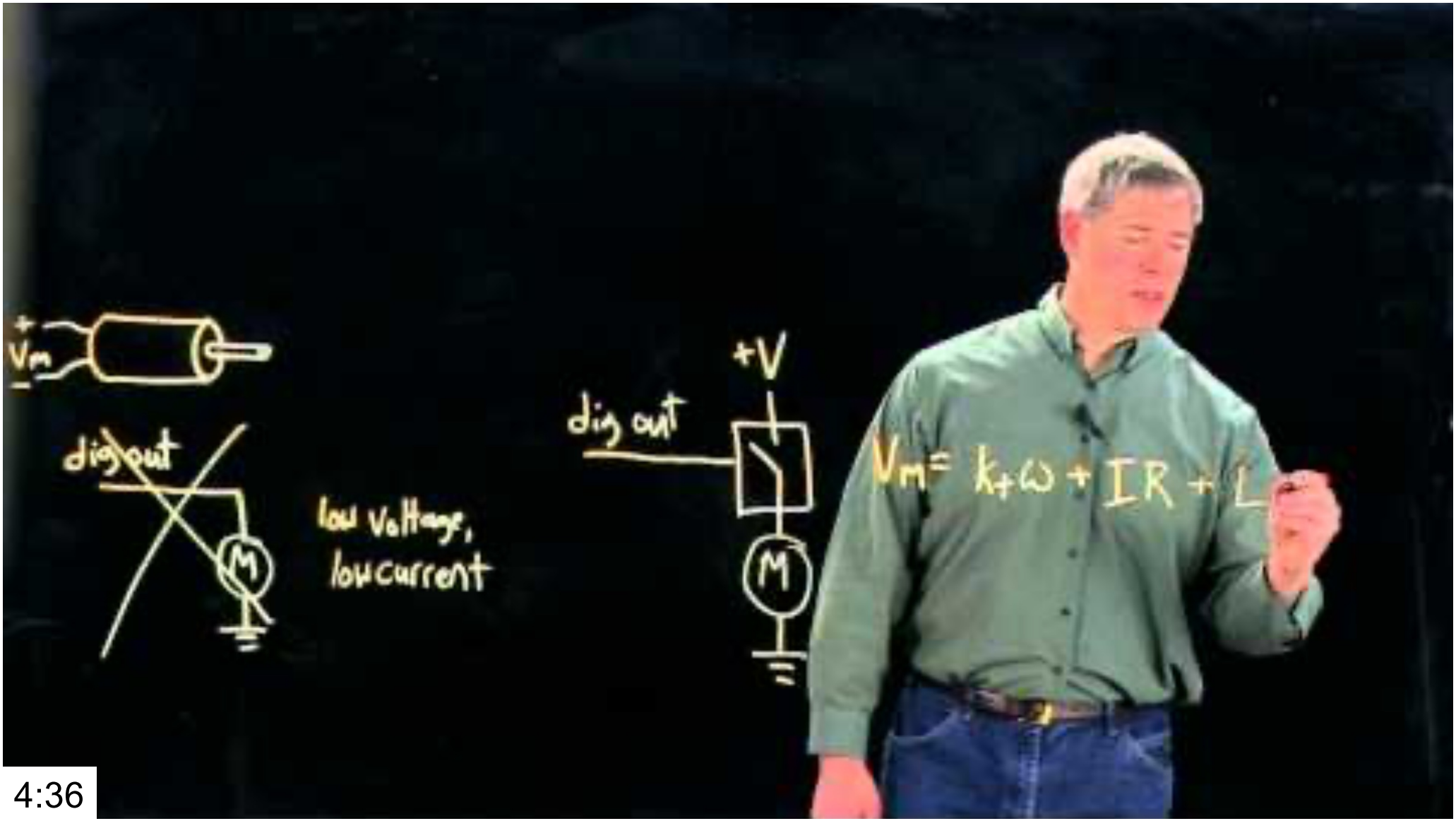
The 3.3 V pin is connected to the 3.3 V rail which is connected to the output of the MP2322GQH step down converter. This pin is primarily used to power external components.

11.6 Pin Current

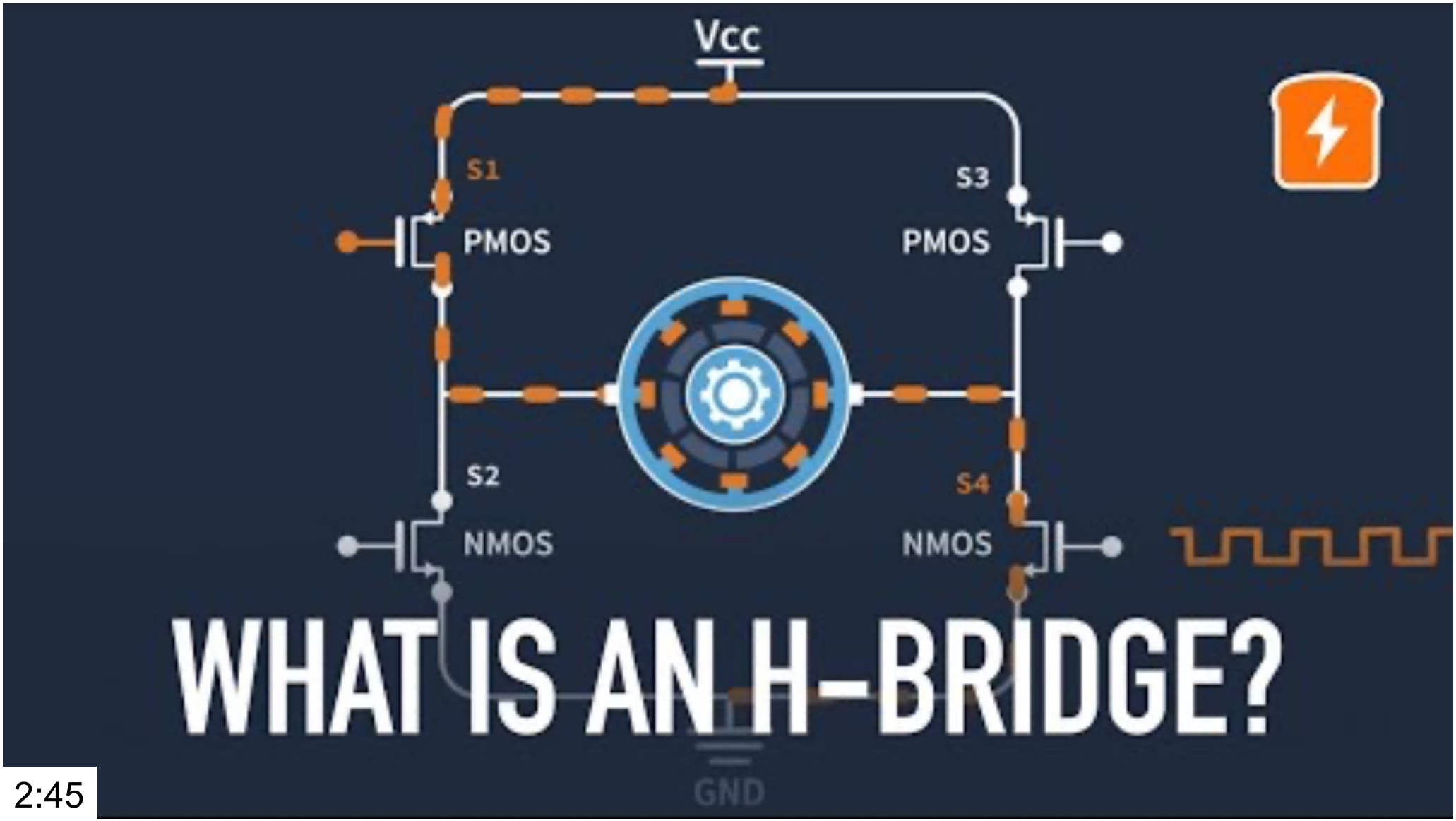
The GPIOs on the Nano ESP32 can handle **source currents** up to **40 mA**, and **sink currents** up to **28 mA**. Never connect devices that draw higher current directly to a GPIO.

900mA

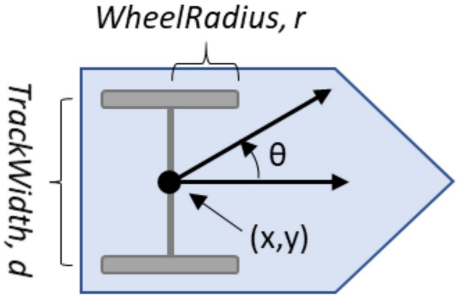
4.5 Watts



4:36

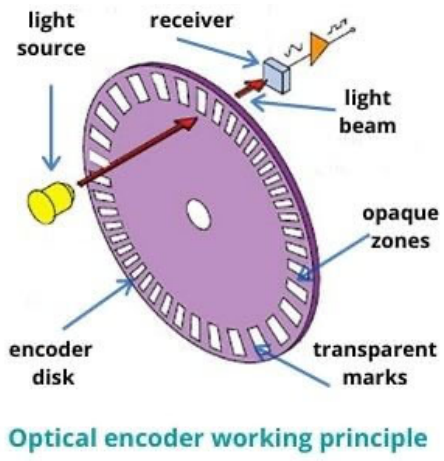

Vcc
S1
PMOS
S3
PMOS
S2
NMOS
S4
NMOS
WHAT IS AN H-BRIDGE?
GND
2:45

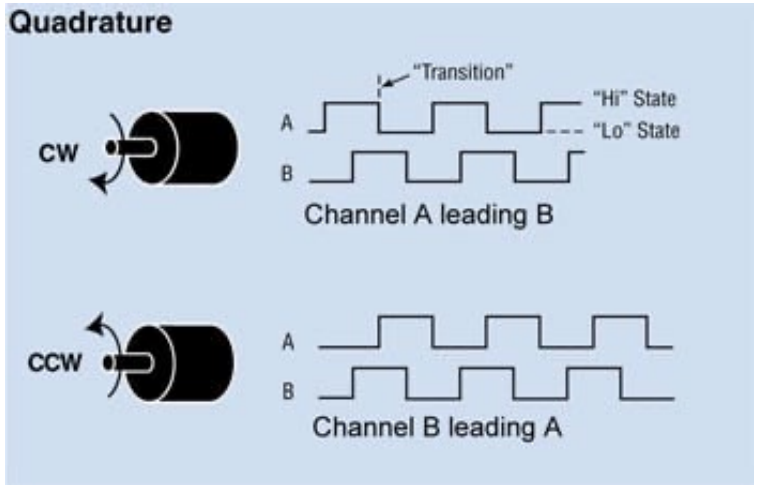
Encoders: Integration and Use



$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r}{2}\cos(\theta) & \frac{r}{2}\cos(\theta) \\ \frac{r}{2}\sin(\theta) & \frac{r}{2}\sin(\theta) \\ -r/d & r/d \end{bmatrix} \begin{bmatrix} \dot{\phi}_L \\ \dot{\phi}_R \end{bmatrix}$$

$[x, y]$ = Vehicle position
$\dot{\phi}$ = Wheel speed
ω = Vehicle heading angular velocity

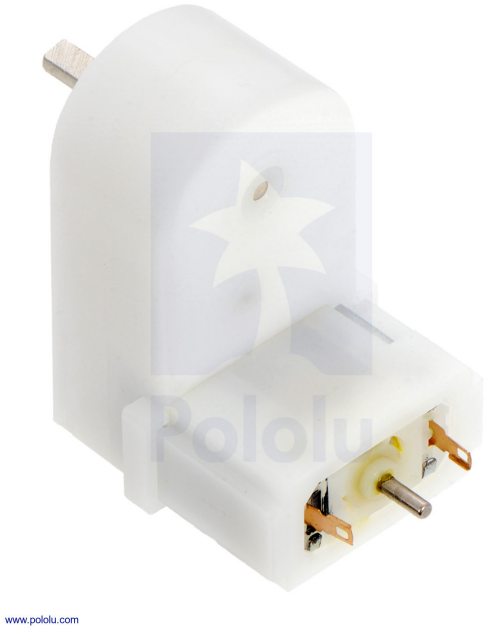




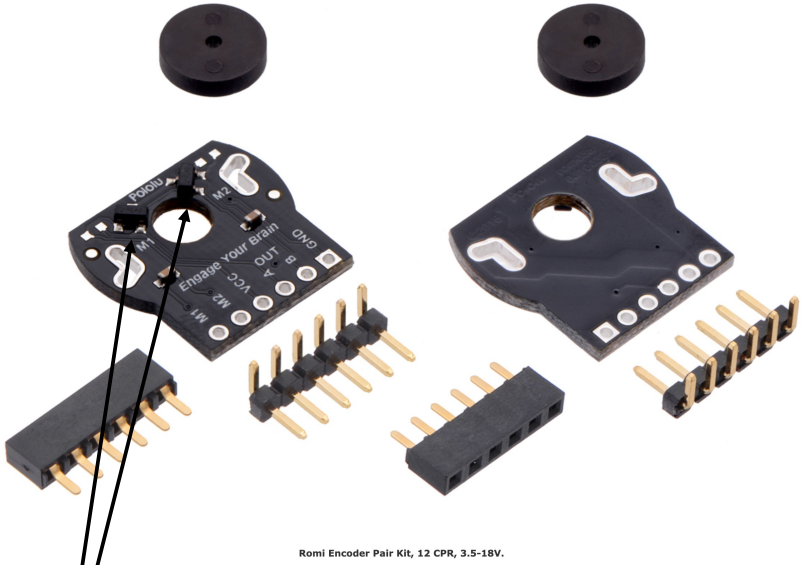
Incremental encoders are typical for mobility

Quadrature encoders give direction

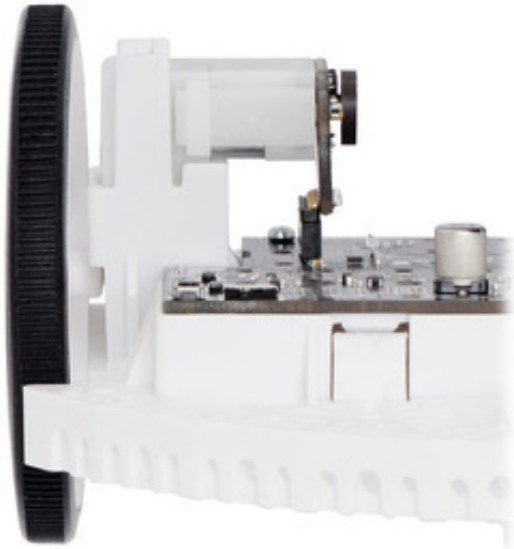
Encoders: Integration and Use



1:120 TT motor

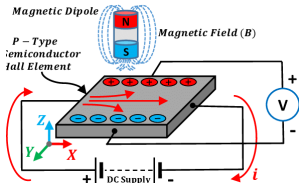


Magnetic quadrature encoder

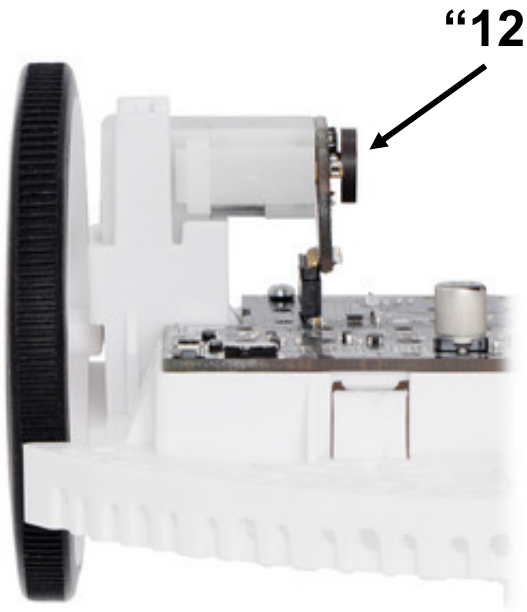


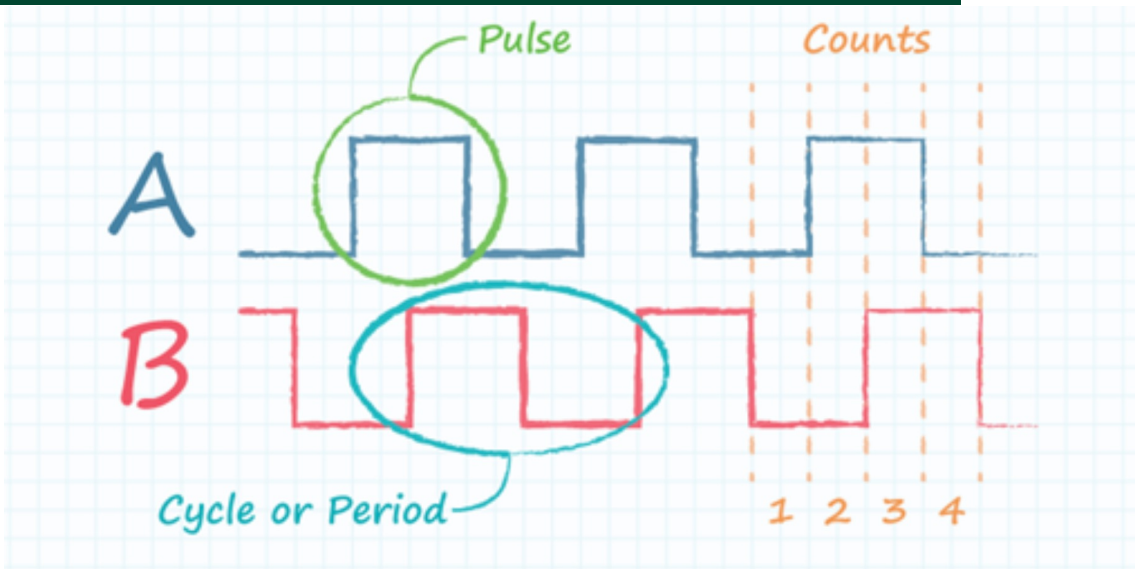
Integrated

Hall effect sensors



Encoders: Integration and Use





Counts per revolution of wheel shaft: CPR * gear ratio
To calculate RPM: counts since last read / delta t
e.g.,

```
counts = encoder.counts – last_counts
countspersec = counts / deltat
rpm = countspersec * (60 / (12*120))
```

1. Lecture (~40 minutes)
2. Subsystem check-in (~5 minutes only)
3. FabLab training (~4:15pm)

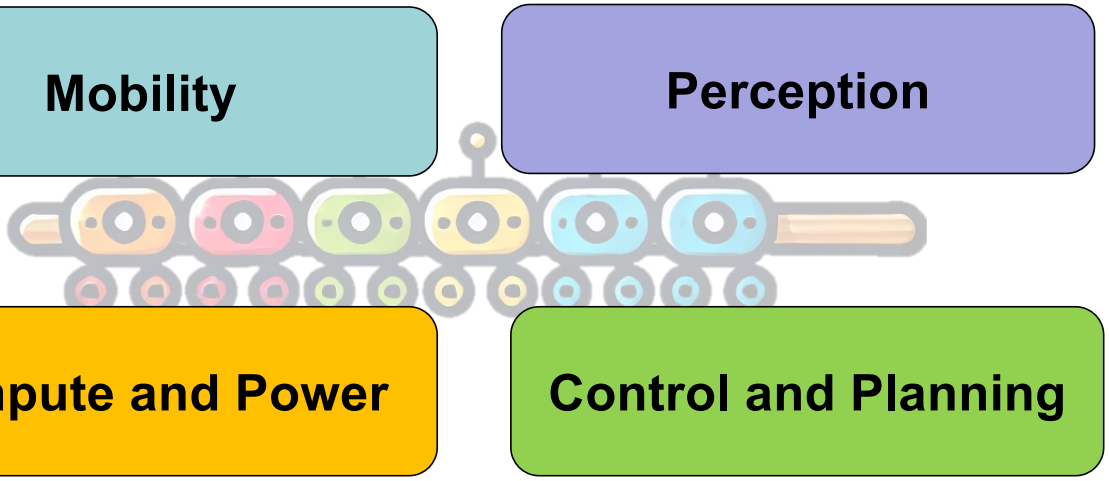
Questions?

Lecture 1: motors → encoders
Lecture 2: motor control → motion planning

693H Robotic Subsystems

Motors, drive electronics, encoders, PID controllers, top-down tracking

Sensor suite (IMU, prox.), calibration code, general IO (buttons and LEDs)



Battery and charging, wireless comms, GPIO and power distribution routing

microROS, ROS2, Gazebo simulation, enclosure design

Mechanics / Prototyping

Coding

PCB Design