上海科技大学
ShanghaiTech University

CS289: **Mobile Manipulation Fall 2023**

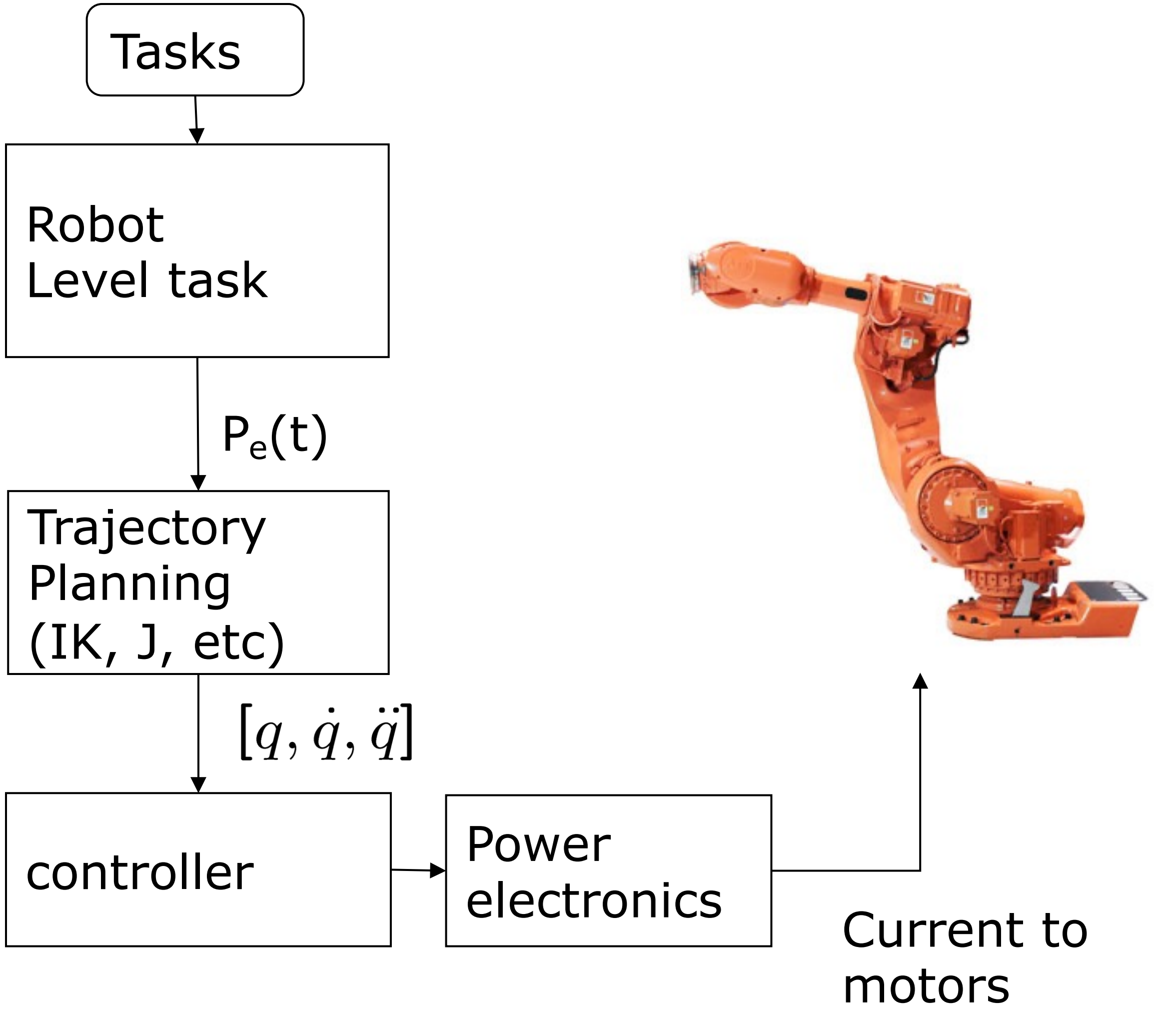
Sören Schwertfeger

ShanghaiTech University

STAR CENTER

Motivation & Overview

- We covered Kinematics, Planning, Perception, etc.
- How to make the robot actually move?
- **Control** the robot motion
  - Dynamics (forces, mass, inertia etc.) =>
  - Kinematics of speeds: Jacobian
  - Control Introduction
  - PID
- Hardware
  - PWM
  - Motor Drivers
  - Motor
  - Gears



What are kinematics?

- Describes the motion of points, bodies (objects), and systems of objects
  - Does not consider the forces that cause them (that would be kinetics)
  - Also known as “the geometry of motion”

- For manipulators
  - Describes the motion of the arm
  - Puts position/ angle and their rate of change (speed) of joints in relation with 3D pose of points on the arm, especially tool center point (tcp, end effector)

Kinematics

Forward Kinematics (angles to pose) (it is straight-forward -> easy)

What you are given: The constant arm parameters (e.g. DH parameters)
The angle of each joint

What you can find: The pose of any point (i.e. it’s (x, y, z) coordinates & 3D orientation)

Inverse Kinematics (pose to angles) (more difficult)

What you are given: The constant arm parameters (e.g. DH parameters)
The pose of some point on the robot

What you can find: The angles of each joint needed to obtain that pose

What are dynamics?

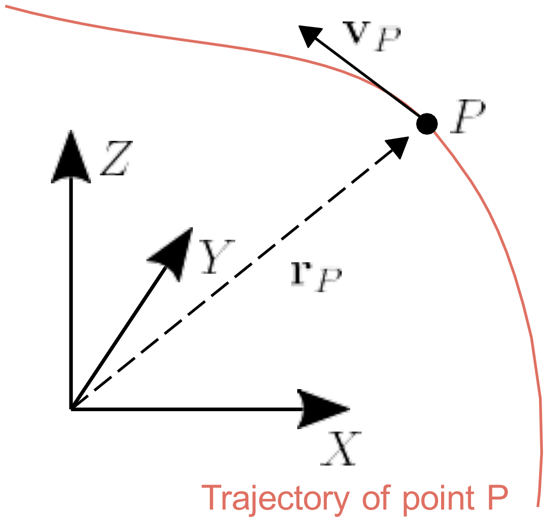
- Kinematics:
  Describes the motion of points, bodies (objects), and systems of objects
  - Including position, speed, acceleration, etc.

- Dynamics:
  Kinematics + physics: forces, mass, inertia, moments – linear and angular

- For manipulators
  - Describes the motion of the arm
  - Puts position/ angle and their rate of change (speed) of joints in relation with 3D pose of points on the arm, especially tool center point (tcp, end effector)
  - Predict motion more accurately
  - Better control because we can predict the force (== motor power) needed

Dynamics of a Rigid Body

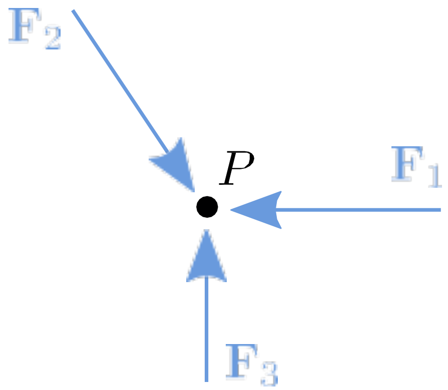
Translatory Motion of a Point:



- Consider **point** P with mass m in $\mathbb{R}^3$
- Let $\boldsymbol{r}_p(t) \in \mathbb{R}^3$ be its **position** in an inertial reference frame
- Let $\boldsymbol{v}_p(t)$ denote its **velocity** and $\mathbf{a}_p(t)$ its **acceleration**
- The **linear momentum** of P is defined as $\mathbf{p}_p(t) = m\mathbf{v}_P(t)$
- By **Newton's second law** we have

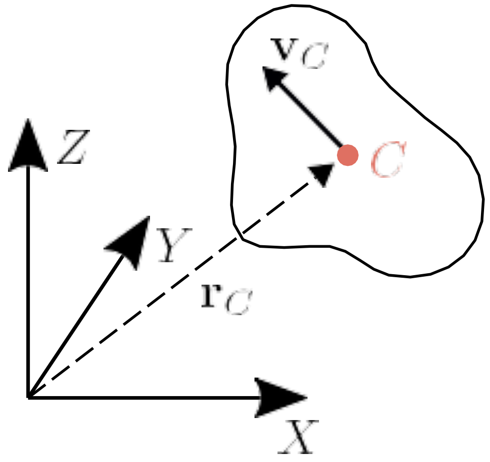
$$\frac{d}{dt}\mathbf{p}_p(t) = m\mathbf{a}_p(t) = F_{net}(t) = \sum_i \mathbf{F}_i(t)$$

where $\mathbf{F}_i(t)$ represent all forces acting on the point mass P



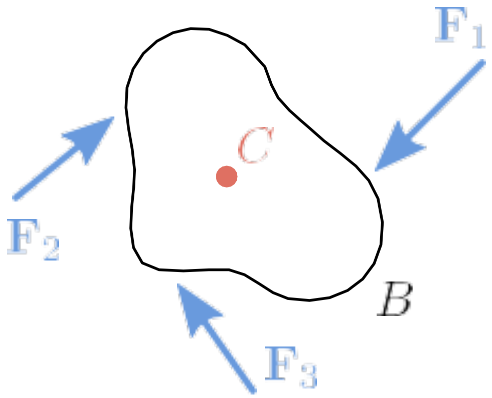
Dynamics of a Rigid Body

Translatory Motion of a Rigid Body:



- Consider a **rigid body** B with mass m in $\mathbb{R}^3$
- Let $\mathbf{r}_C(t) \in \mathbb{R}^3$ be the **position** of its center of gravity C
- Let $\mathbf{v}_C(t)$ denote its **velocity** and $\mathbf{a}_C(t)$ its **acceleration**
- The **linear momentum** of B is defined as $\mathbf{p}_B(t) = m\mathbf{v}_C(t)$
- The center of gravity of a rigid body **behaves like a point mass** with mass m and as if all forces act on that point

$$\frac{d}{dt}\mathbf{p}_B(t) = m\mathbf{a}_C(t) = F_{net}(t) = \sum_i \mathbf{F}_i(t)$$

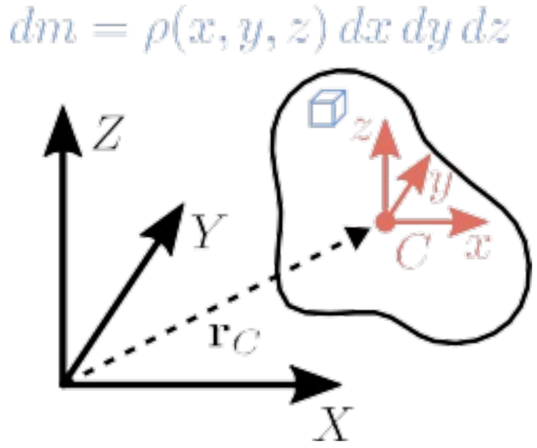


where $\mathbf{F}_i(t)$ represent all forces acting on the rigid body B

Dynamics of a Rigid Body

Rotatory Motion of a Rigid Body:

- For the **rotatory motion,** also the geometric shape of B and the spatial distribution of its mass is important
- Let $\rho(x, y, z)$ be the **body's density function:**



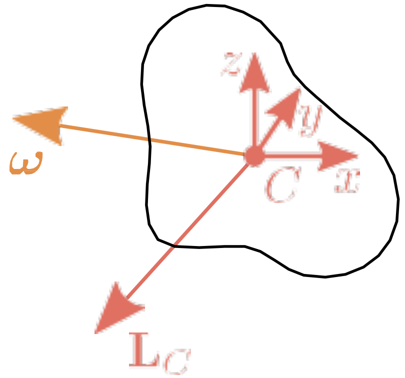
$$m = \int_B \rho(x, y, z)\, dx\, dy\, dz = \int_B dm$$

- The **inertia tensor** of B is defined as

$$\boldsymbol{\Theta} = \begin{bmatrix} I_x & I_{xy} & I_{xz} \\ I_{yx} & I_y & I_{yz} \\ I_{zx} & I_{zy} & I_z \end{bmatrix} \qquad \underbrace{\begin{aligned} I_x &= \textstyle\int_B (y^2 + z^2)\, dm \\ I_y &= \textstyle\int_B (x^2 + z^2)\, dm \\ I_z &= \textstyle\int_B (x^2 + y^2)\, dm \end{aligned}}_{\text{moments of inertia}} \qquad \underbrace{\begin{aligned} I_{xy} &= I_{yx} = -\textstyle\int_B xy\, dm \\ I_{xz} &= I_{zx} = -\textstyle\int_B xz\, dm \\ I_{yz} &= I_{zy} = -\textstyle\int_B yz\, dm \end{aligned}}_{\text{moments of deviation}}$$

Dynamics of a Rigid Body

Rotatory Motion of a Rigid Body:



- Let $\boldsymbol{\omega}$ be the vector of **angular velocities:**

$$\boldsymbol{\omega} = (\omega_x \ \omega_y \ \omega_z)^T$$

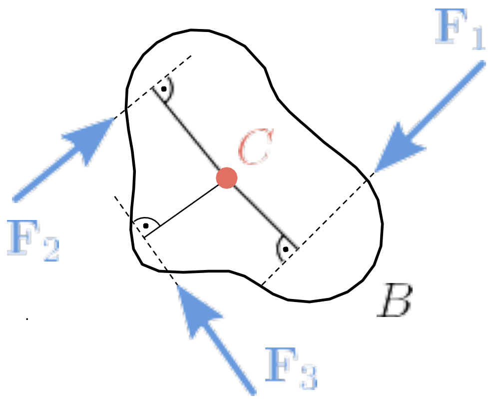
- The **angular momentum** $\mathbf{L}_C$ of the rigid body B is given by

$$\mathbf{L}_C = \boldsymbol{\Theta} \ \boldsymbol{\omega}$$

- By the **angular momentum principle**

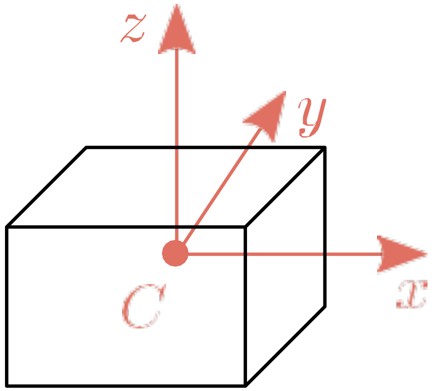
$$\frac{d}{dt}\mathbf{L}_C(t) = \boldsymbol{\Theta}\,\dot{\boldsymbol{\omega}} = \mathbf{M}_{net}(t) = \sum_i \mathbf{M}_i(t)$$

where $\mathbf{M}_i(t)$ are the moments of all forces acting on B with respect to the center of gravity C.



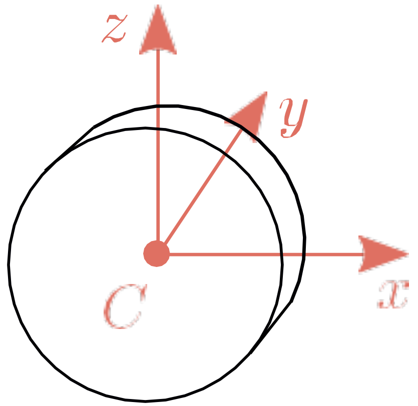
Dynamics of a Rigid Body

Rotatory Motion of a Rigid Body with Canonical Coordinates:



- If the body frame is chosen as a principal axis system for the rigid body (symmetry axes), the inertia tensor is diagonal:

$$\boldsymbol{\Theta} = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}$$



- For the planar motion of a rigid body in the x/y-plane:

$$\omega_x = \omega_y = 0 \quad \text{and} \quad M_x = M_y = 0$$

- Hence the angular momentum becomes $L_z = I_z\ \omega_z(t)$ and the angular momentum principle yields $I_z\ \dot{\omega}_z = \sum_i M_i$

Kinematics with speeds

- We need linear velocities and accelerations:
  $\boldsymbol{v}_p\,(t)$ velocity and $\mathbf{a}_p\,(t)$ its acceleration
- We need angular velocities $\boldsymbol{\omega}$ and accelerations $\dot{\boldsymbol{\omega}}$

- => use Kinematics with speeds => use Jacobians

Jacobian Matrix

- We need to know and to represent the relationship between the rates of change of the individual joint values:

$$\dot{q} = (\dot{q}_1, \dot{q}_2, \ldots, \dot{q}_N)$$

- and the rate of change of ***pose*** == angular $\boldsymbol{\omega}$ and linear velocity $\boldsymbol{v}$

$$\dot{X} = (\dot{\phi}, \dot{\psi}, \dot{\theta}, \dot{x}, \dot{y}, \dot{z})$$

- the matrix which represents this relationship the is called the Jacobian Matrix, J

$$\dot{X} = J\dot{q}$$

Material: University Crne Gore
https://ucg.ac.me/

Kinematics: Velocities

Cartesian Space

Tool Frame (E)

(aka End-Effector)

Base Frame (B)

$$ {}^{B}_{E}V = \begin{Bmatrix} {}^{B}_{E}v \\ {}^{B}_{E}w \end{Bmatrix} $$

v: linear velocity

w: angular velocity

Jacobian

$$ {}^{B}_{E}V = J(q)\dot{q} $$

$$ \dot{q} = J^{-1}(q)\ {}^{B}_{E}V $$

Inverse Jacobian

Joint Space

Joint 1 = $\dot{q}_1$

Joint 2 = $\dot{q}_2$

Joint 3 = $\dot{q}_3$

...

Joint n = $\dot{q}_n$

Rigid body transformation
Between coordinate frames

Linear algebra

Jacobian Calculation

- We can obtain J by differentiating the forward kinematic relationships

$$\boxed{\begin{aligned} x_e &= f_1(q_1, q_2, \ldots, q_N) \\ y_e &= f_2(q_1, q_2, \ldots, q_N) \\ z_e &= f_3(q_1, q_2, \ldots, q_N) \\ \phi_e &= f_4(q_1, q_2, \ldots, q_N) \\ \psi_e &= f_5(q_1, q_2, \ldots, q_N) \\ \theta_e &= f_6(q_1, q_2, \ldots, q_N) \end{aligned}} \Rightarrow \begin{aligned} \dot{x}_e &= \sum_{i=1}^{N} \frac{\partial f_1}{\partial q_i} \dot{q}_i \\ \dot{y}_e &= \sum_{i=1}^{N} \frac{\partial f_2}{\partial q_i} \dot{q}_i \\ &\vdots \\ \dot{\theta}_e &= \sum_{i=1}^{N} \frac{\partial f_6}{\partial q_i} \dot{q}_i \end{aligned}$$

Written in Matrix Form

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\phi} \\ \dot{\psi} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial q_1} & \cdots & \frac{\partial f_1}{\partial q_6} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_6}{\partial q_1} & \cdots & \frac{\partial f_6}{\partial q_6} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \\ \dot{q}_5 \\ \dot{q}_6 \end{bmatrix}$$

$$\dot{X} = J \, \dot{q}$$

Rate of change of joint position

Rate of change of Cartesian pose

Properties of J

- J is not necessarily a square matrix
  - number of joints
  - degrees of freedom of task

$$\dot{X} = J\dot{q}$$

- J depends upon the instantaneous values of q_i, i = 1,…, N so J will be *different* for each different set of joint values (q_1, q_2, …, q_N), i.e. for each different robot arm configuration.

$$\dot{X} = J(q_1, q_2, \ldots, q_N)\dot{q}$$

Robot Jacobian

- To obtain the inverse Jacobian relation we need to invert **J**, which is, in general, hard. Three methods:
  1. Invert **J** symbolically, which is only really practical for very simple robot geometries.
  2. Numerically invert **J** for each configuration of the robot. This is computationally expensive, not always possible (e.g. when det(**J**)=0) and difficult if $n \neq 6$
     - Use pseudo-inverse $(\mathbf{J}¢\, \mathbf{J}^{T})^{-1}\, \mathbf{J}^{T}$
  3. Derive $\mathbf{J}^{-1}$ directly from the Inverse Kinematics equations, much as we uses the Forward Kinematics equations to obtain **J** above.

Singularities

- Robot is in a *singular configuration* when det(J)=0 $$\dot{X} = J\dot{q}$$
  - i.e. when the relationship can't be inverted
- Singular configuration occur when two or more joint axes become aligned in space.
  - When this happens the robot geometry effectively loses one (or more) independent degrees of freedom: two more more of the degrees of freedom become mutually dependent.

Singularities

- The loss of one or more effective degrees of freedom thus occurs not just at a singular configuration, but also in a region (a volume in joint space) around it.
  - Not just when det(J) = 0 but nearby (J is ill-conditioned).
  - Condition number is a useful (scale-independent) measure for matrix condition

Types of Singularities

- *Workspace Boundary Singularities:* the robot manipulator is fully extended or folded onto itself so that P_e is at or near the boundary of the robot workspace.
  - In such configurations, one or more joints will be at their limits of range of movement, so that they will not be able to maintain any movement at some particular speed. This effectively makes **J** a singular matrix.

- *Workspace Interior Singularities:* occur inside the robot workspace away from any of the boundaries, and they are generally caused by two or more joint axes becoming aligned.

Avoid Singularities

- To avoid singular configurations there are three different possibilities:

1. Increase the number if degrees of freedom of the robot manipulator, perhaps by attaching to *Pe* a tool or gripper which has one or two degrees of freedom itself.

2. Restrict the movements that the robot can be programmed to make so as to avoid getting to or near to any singular configurations.

3. Dynamically modify **J** to remove the offending terms, and thus return $det(\mathbf{J}) \neq 0$. This means identifying the column and row of **J** that need to be removed.

CONTROL

Control

Tasks

Robot Level task

$P_e(t)$

Trajectory Planning (IK, J, etc)

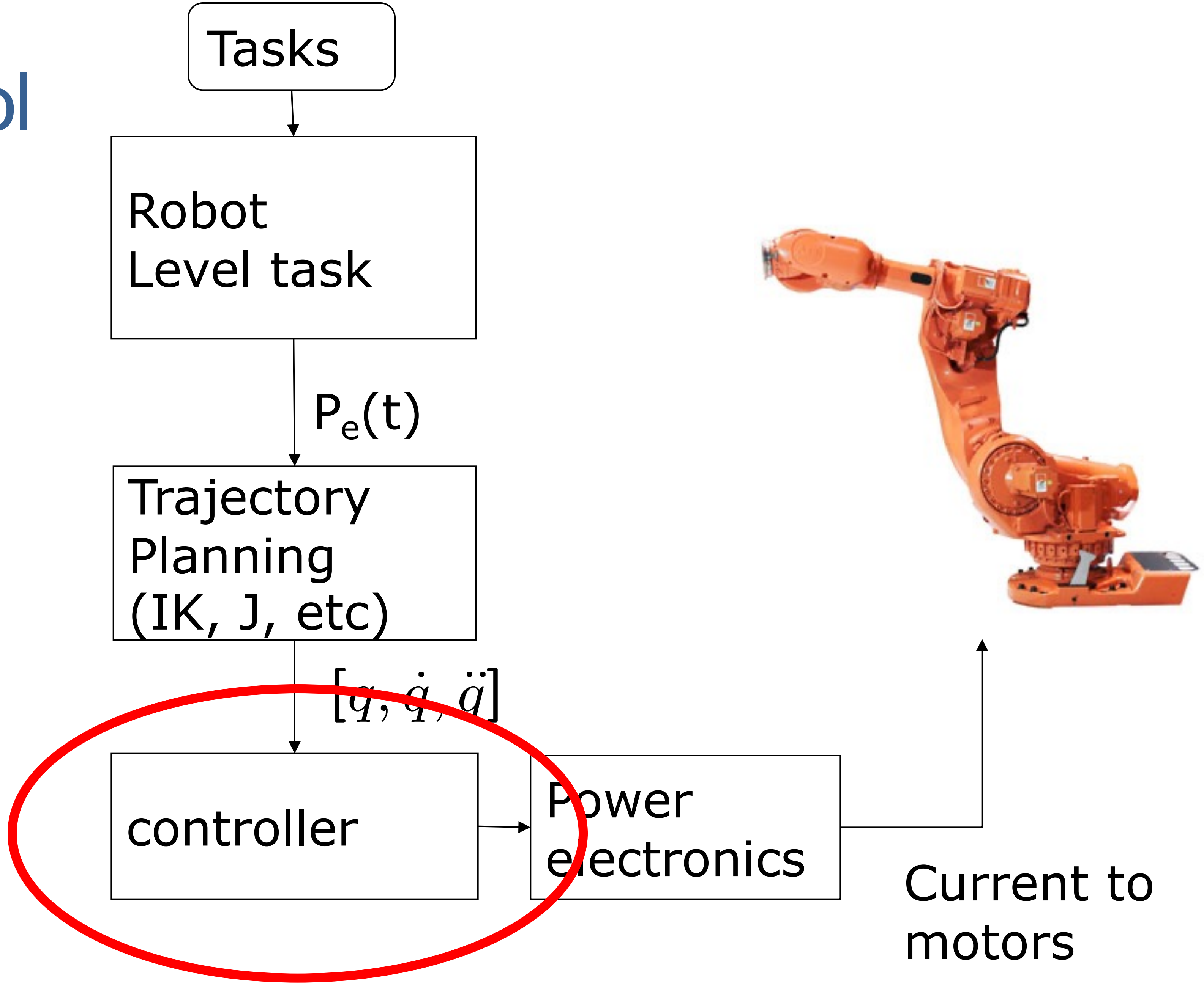
$[q, \dot{q}, \ddot{q}]$

controller

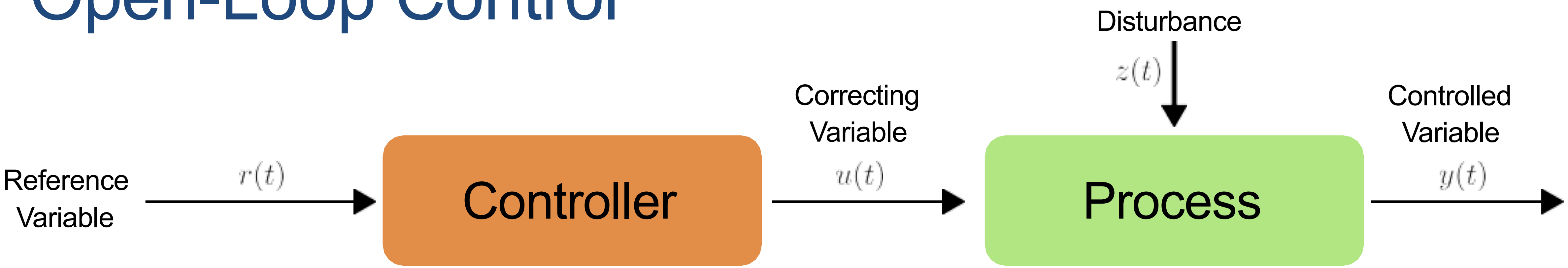
Power electronics

Current to motors

Control

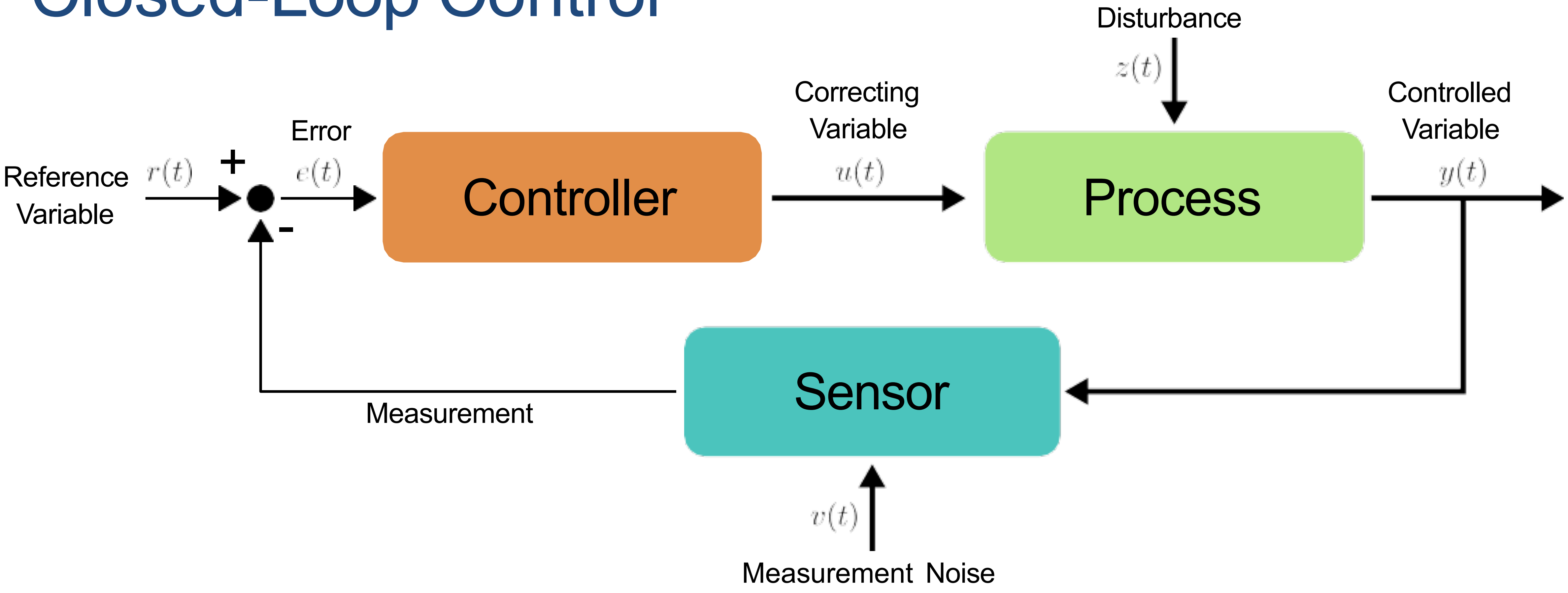

Tasks
Robot Level task
$P_e(t)$
Trajectory Planning (IK, J, etc)
$[q, \dot{q}, \ddot{q}]$
controller
Power electronics
Current to motors

Open-Loop Control



- Requires **precise knowledge** of the plant and the influence factors
- **No feedback** about the controlled variable
- Cannot handle unknown disturbances, resulting in **drift**

Closed-Loop Control



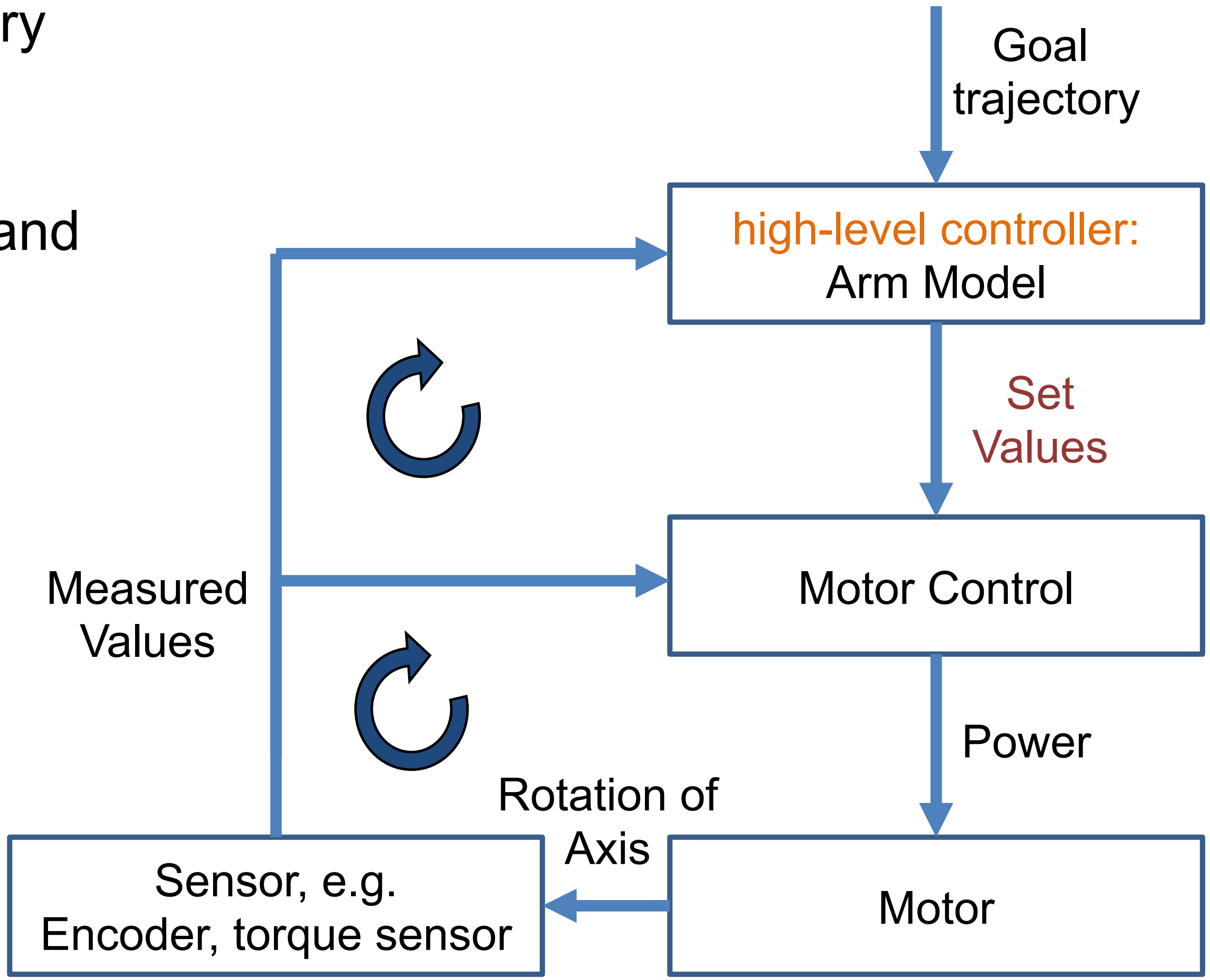
► Exploit feedback loop to minimize error between reference and measurement

Centrifugal Governor

https://www.youtube.com/watch?v=B01LgS8S5C8

Control Hierarchy

- Assume we have a goal trajectory
- What values does the high-level controller set, using arm model and goal trajectory?
  - Position
  - Speed, Acceleration
  - Torque
  - Force
- Different control loops with different speeds
  - e.g. high-level controller 50Hz;
  - e.g. Motor controller 1000Hz;



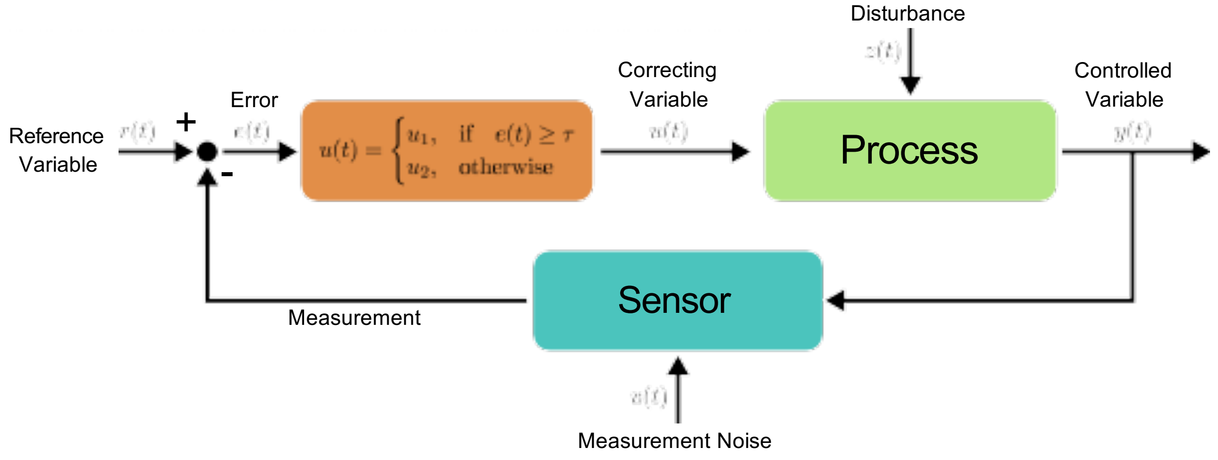
BLACK BOX CONTROL

Bang-Bang Control

Bang-Bang Control

- Also called: hysteresis controller
- Often applied, e.g. in household thermostats
- Switches abruptly between two states
- Mathematical formulation:

$$u(t) = \begin{cases} u_1, & \text{if} \quad e(t) \geq \tau \\ u_2, & \text{otherwise} \end{cases}$$



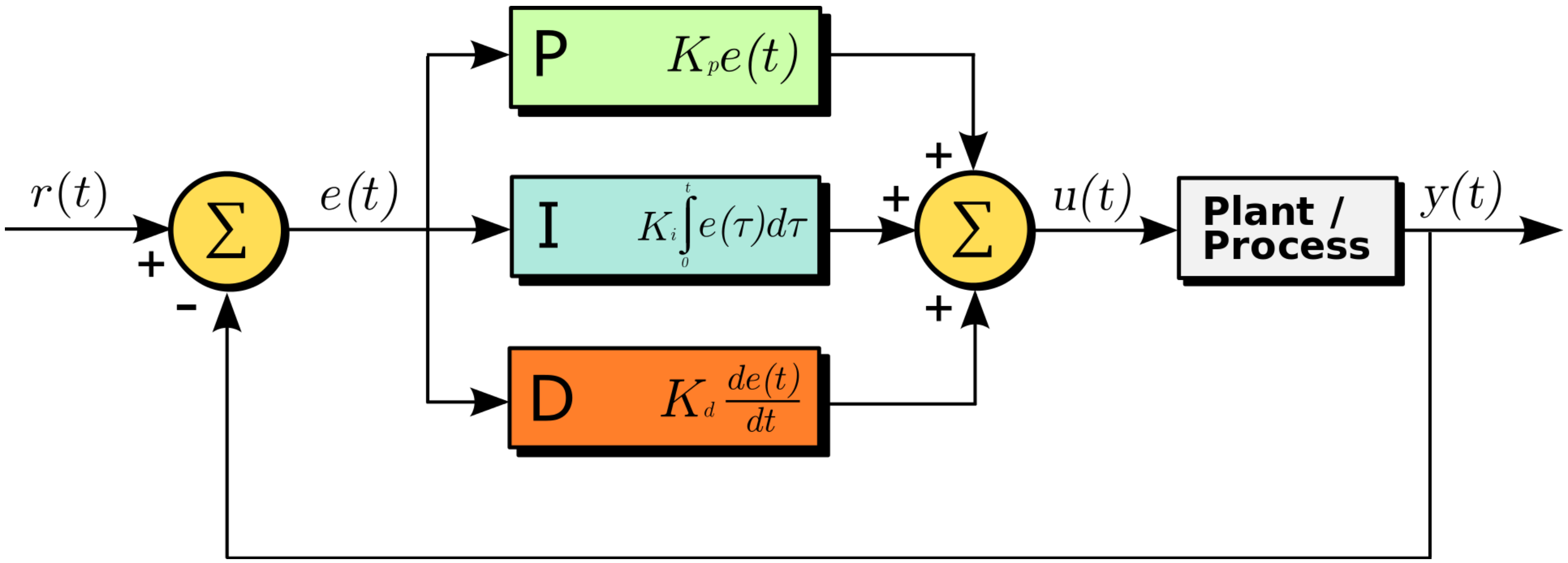
PID Control

PID: Proportional-Integral-Derivative Controller

- Input: Desired Speed (of wheel/ motor)
  - Actually: Error of the current speed (process variable) to the desired speed (setpoint)
- Output: Amount of power to the motor
- Not needed: Model of the plant process (e.g. motor, robot & terrain parameters)
- Parameters:
  - K_p proportional gain constant
  - K_i integral gain
  - K_d derivative gain
- Discrete Version:

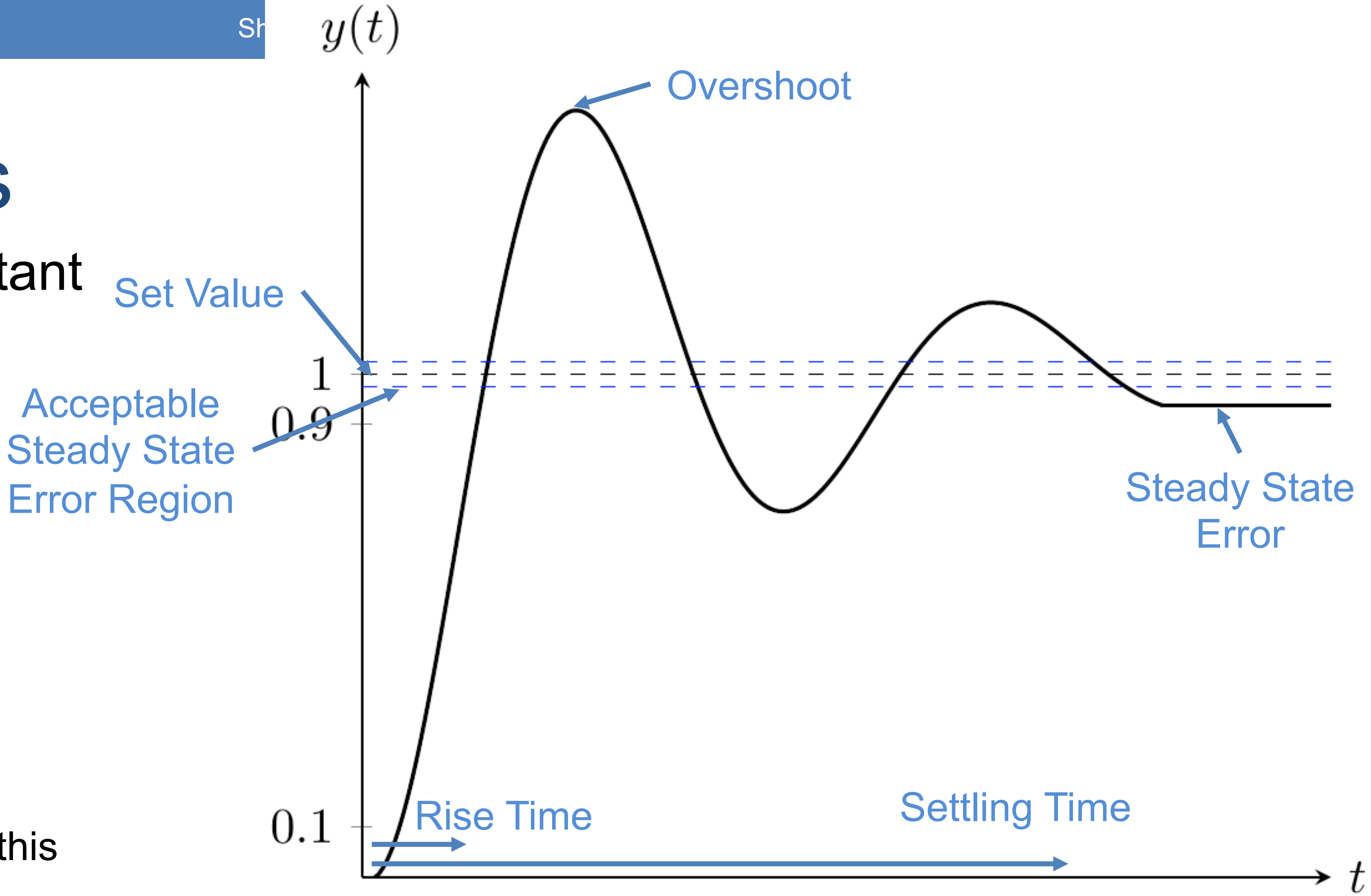
$$\int_0^{t_k} e(\tau)\,d\tau = \sum_{i=1}^{k} e(t_i)\,\Delta t$$

$$\frac{de(t_k)}{dt} = \frac{e(t_k) - e(t_{k-1})}{\Delta t}$$



Tune Parameters

- K_p proportional gain constant
  - Too small: long rise time
  - Too big: big overshoot or even unstable control
  - Should contribute most of the output change
- K_i integral gain
  - Reduces steady state error
  - May cause overshoot
    - Leaky integration may solve this
- K_d derivative gain
  - Predicts error by taking slope into account
  - May reduce settling time and overshoot



Parameter Increase	Rise time	Overshoot	Settling Time	Steady-state error
Kp	↓	↑	Small Change	↓
Ki	↓	↑	↑	Great reduce
Kd	Small Change	↓	↓	Small Change

Table (2) PID controller parameter characteristics on a fan's response

https://people.ece.cornell.edu/land/courses/ece4760/FinalProjects/s2012/fas57_nyp7/Site/pidcontroller.html

Control Theory

- Other controllers used
  - P Controller
  - PD Controller
  - PI Controller

```
previous_error := 0
integral := 0

loop:
    error := setpoint − measured_value
    integral := integral + error × dt
    derivative := (error − previous_error) / dt
    output := Kp × error + Ki × integral + Kd × derivative
    previous_error := error
    wait(dt)
    goto loop
```

Pseudo Code PID Controller

- PID sufficient for most control problems
- PID works well if:
  - Dynamics of system is small
  - System is linear (or close to)
- Lots of Control Theory courses at ShanghaiTech University…

- Popular alternative: Model Predictive Control (MPC)
  - Optimal Control Technique: satisfy a set of constraints
  - Finite time horizon to look into the future (“plan”)
  - Used when PID is not sufficient; e.g.:
    - Very dynamic system
    - Second order system (oscillating system)
    - Multi-variable control
  - Use Cases: Chemical plants; planes; robot arms; legged robots; …

Controlling Self-Driving Cars

Aerospace Controls Laboratory

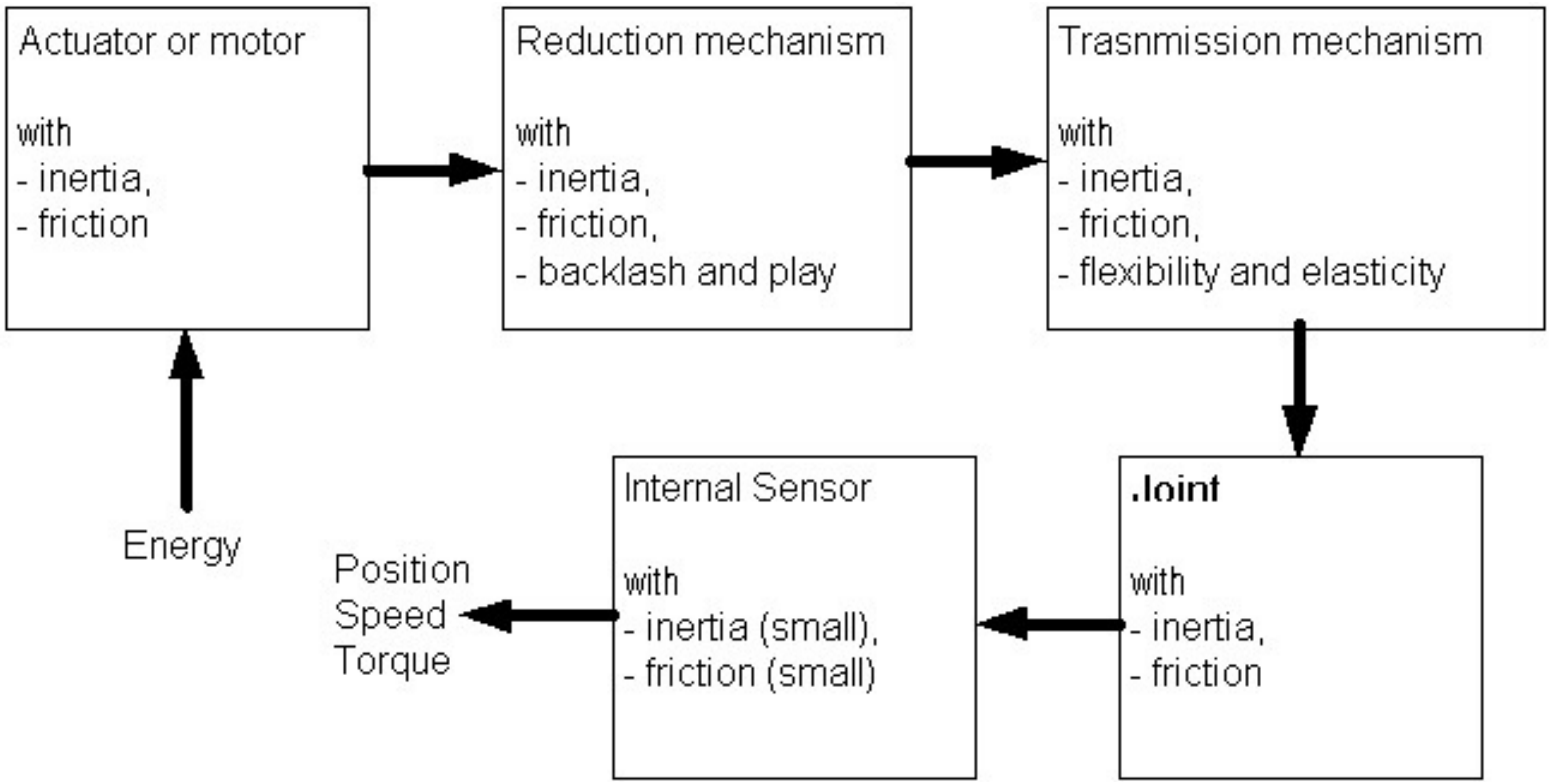
Massachusetts Institute of Technology

Closed-Loop Arm Control

- Independent Joint Control
  - Use computed reference points (setpoints) for each joint
  - Control each joint “independently”
    - Ignore dynamic effects
    - Treat each joint as a stand alone “motor”
- Dynamics Based control
  - Use dynamics model to facilitate control
    - Compute torque feedforward
    - Inverse Dynamic Control
    - Operation Space control
    - and Compliance, Impedance, Force….

Control:
Position
Speed, Acceleration
Torque
Force

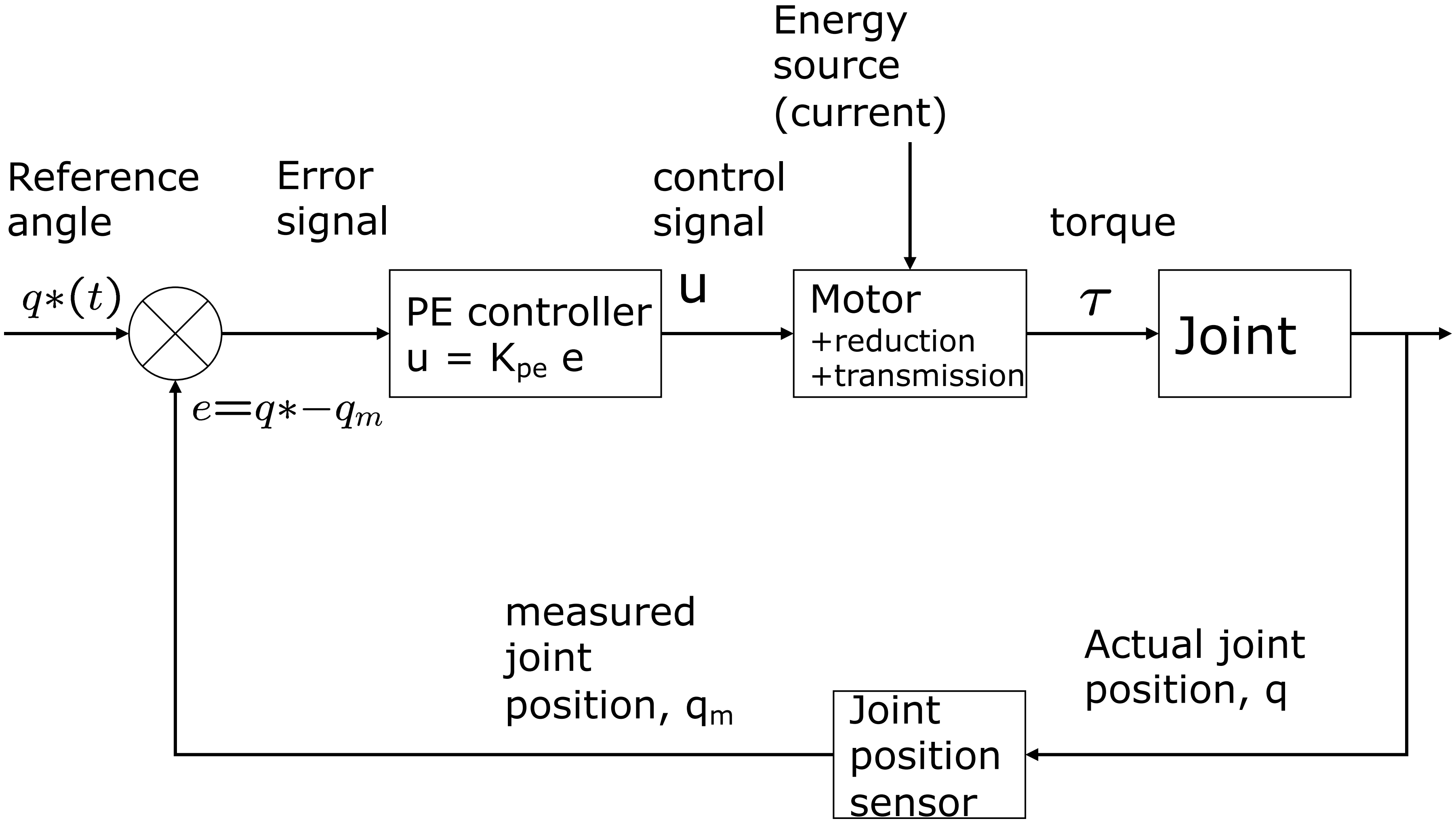
Jointed system components



Independent Joint Control

- Use computed reference points (setpoints) for each joint
- Control each joint “independently”
  - Ignore dynamic effects
  - Treat each joint as a stand alone “motor”
- Simplifies control
- Block Diagram (next slide)

Block Diagram of PE controller for a single joint



Independent Joint Control

- Control each joint independently without "communication" between actuators
- Basic Steps:
  - Model actuator
  - Use kinematics to obtain set-points for each joint
  - Develop a controller for each joint
  - Error for joint i:

$$e_i = (q_i^* - q_m)$$

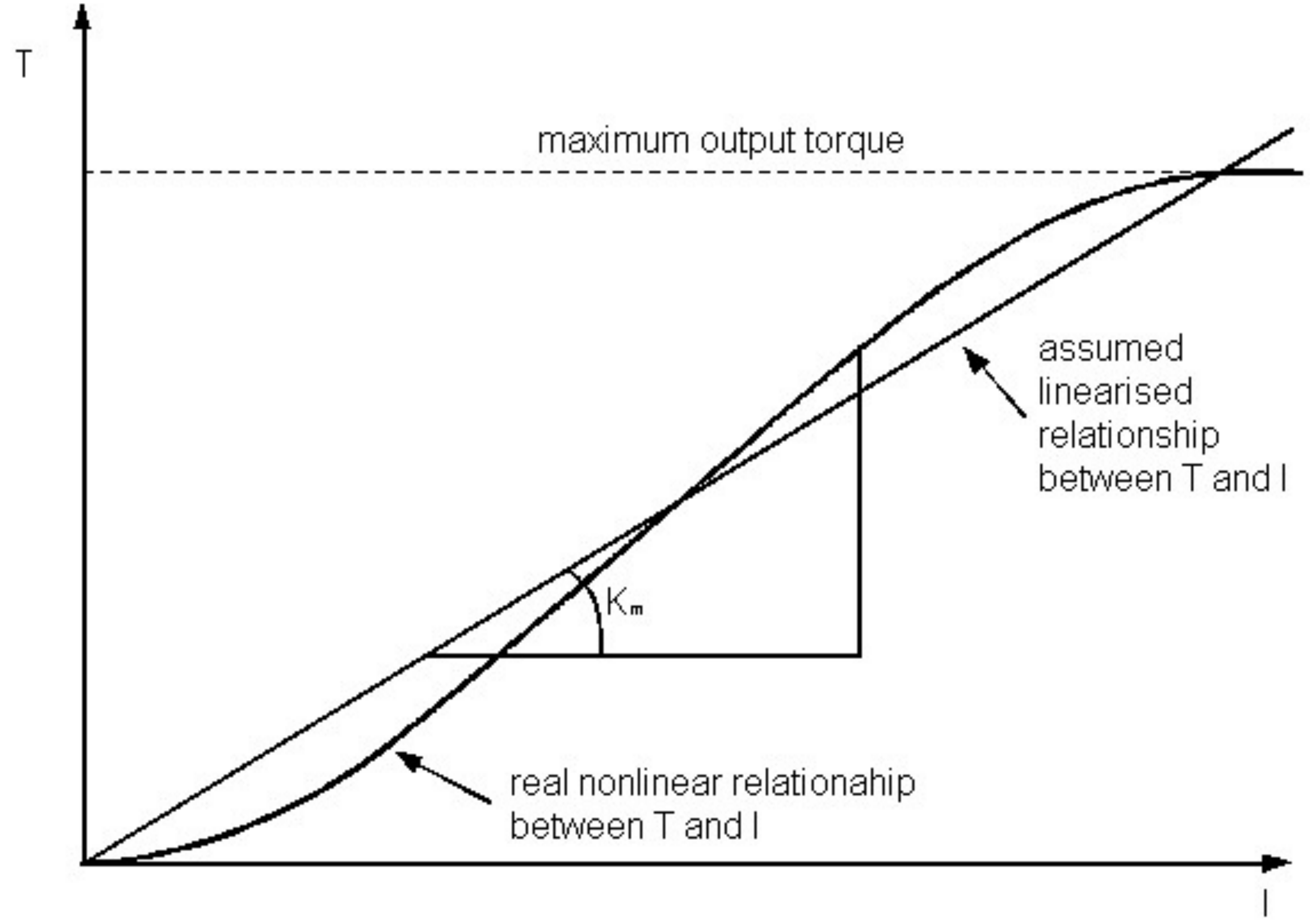
$$\begin{aligned} q_i^* &= \text{desired joint position} \\ q_m &= \text{measured joint position} \end{aligned}$$

Actuator Model

- Need to model relationships:
  - between actuator input (current) and output (torque)
    - Torque is approximately linear with applied current

$$\underbrace{\tau_m}_{\text{Motor torque, Nm}} = \underbrace{K_m}_{\text{Motor constant, Nm/amp}} \underbrace{i_a}_{\text{Applied current, amp}}$$

actuator current vs torque



Actuator Model

- Need to model relationships:
  - between actuator torque and motor angle (q)
    - Second order ode

disturbance

$$\underbrace{J}_{\text{Rotational inertia of joint, kg m^2}}\ddot{q}(t) + \underbrace{B}_{\text{Effective damping (friction, back emf), Nm/amp}}\dot{q}(t) = \underbrace{u(t)}_{\text{control input}} - \overbrace{d}^{\text{disturbance}}$$

Rotational inertia of joint, kg m^2

control input

Effective damping (friction, back emf), Nm/amp

Independent Joint Control

- Control each joint independently without “communication” between actuators
- Basic Steps:
  - ✓Model actuator
  - ✓Use kinematics to obtain setpoints for each joint (IK)
  - Develop a controller for each joint
    - Error for joint i:

$$e_i = (q_i^* - q_m)$$

$$\begin{aligned} q_i^* &= \text{desired joint position} \\ q_m &= \text{measured joint position} \end{aligned}$$

Proportional control for each joint

- Input proportional to position error:

$$u(t) = K_{PE} e_i(t) = K_{PE}(q_i^*(t) - q_m(t))$$

- Neglect disturbance, set reference position to zero

$$u(t) = K_{PE}(0 - q_m(t))$$

$$J\ddot{q}(t) + B\dot{q}(t) = -K_{PE} q(t)$$

- or

$$J\ddot{q}(t) + B\dot{q}(t) + K_{PE} q(t) = 0$$

Proportional control for each joint

- Second order linear differential equation:

$$J\ddot{q}(t) + B\dot{q}(t) + K_{PE}q(t) = 0$$

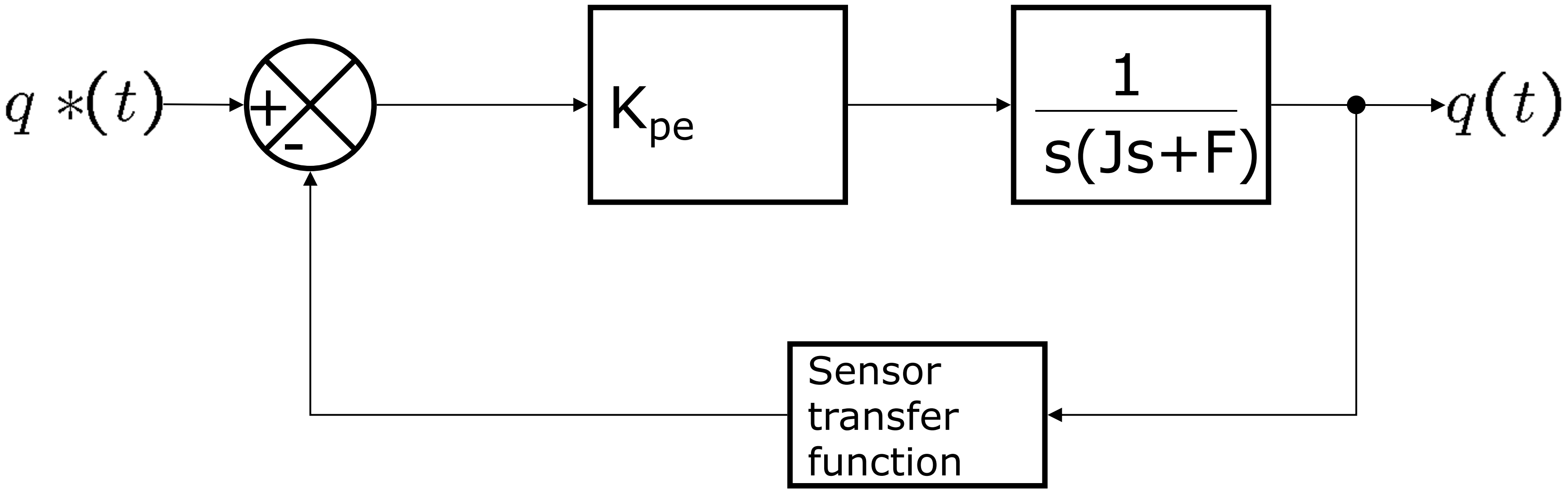
- has general form solution:

$$q(t) = \exp\left(\frac{-Bt}{2J}\right)\left[C_1 \exp\left(\frac{\omega t}{2}\right) + C_2 \exp\left(\frac{-\omega t}{2}\right)\right]$$

  - where

$$\omega = \sqrt{\left(\frac{B^2}{J^2}\right) - \left(\frac{4K_{PE}}{J}\right)}$$

Block Diagram of PE controller



Three solutions

$$q(t) = \exp\left(\frac{-Bt}{2J}\right)\left[C_1 \exp(\frac{\omega t}{2}) + C_2 \exp(\frac{-\omega t}{2})\right]$$

$$\omega = \sqrt{\left(\frac{B^2}{J^2}\right) - \left(\frac{4K_{PE}}{J}\right)}$$

What does this term do?

Three solutions

$$q(t) = \exp\left(\frac{-Bt}{2J}\right)\left[C_1 \exp(\frac{\omega t}{2}) + C_2 \exp(\frac{-\omega t}{2})\right]$$

- Over-damped ($\omega^2 > 0$) $$\frac{B^2}{4K_{PE}} > J$$
- Critically damped ($\omega^2 = 0$) $$\exp(\frac{\omega t}{2}) = \exp(\frac{-\omega t}{2}) = 1$$

$$q(t) = C_{12} \exp\left(\frac{-Bt}{2J}\right)$$

$$\omega^2 = \left(\frac{B^2}{J^2}\right) - \left(\frac{4K_{PE}}{J}\right)$$

Three solutions

$$q(t) = \exp\left(\frac{-Bt}{2J}\right)\left[C_1 \exp(\frac{\omega t}{2}) + C_2 \exp(\frac{-\omega t}{2})\right]$$

$$\frac{B^2}{4K_{PE}} < J$$

- Under-damped ($\omega^2 < 0$)
  - ω has complex roots

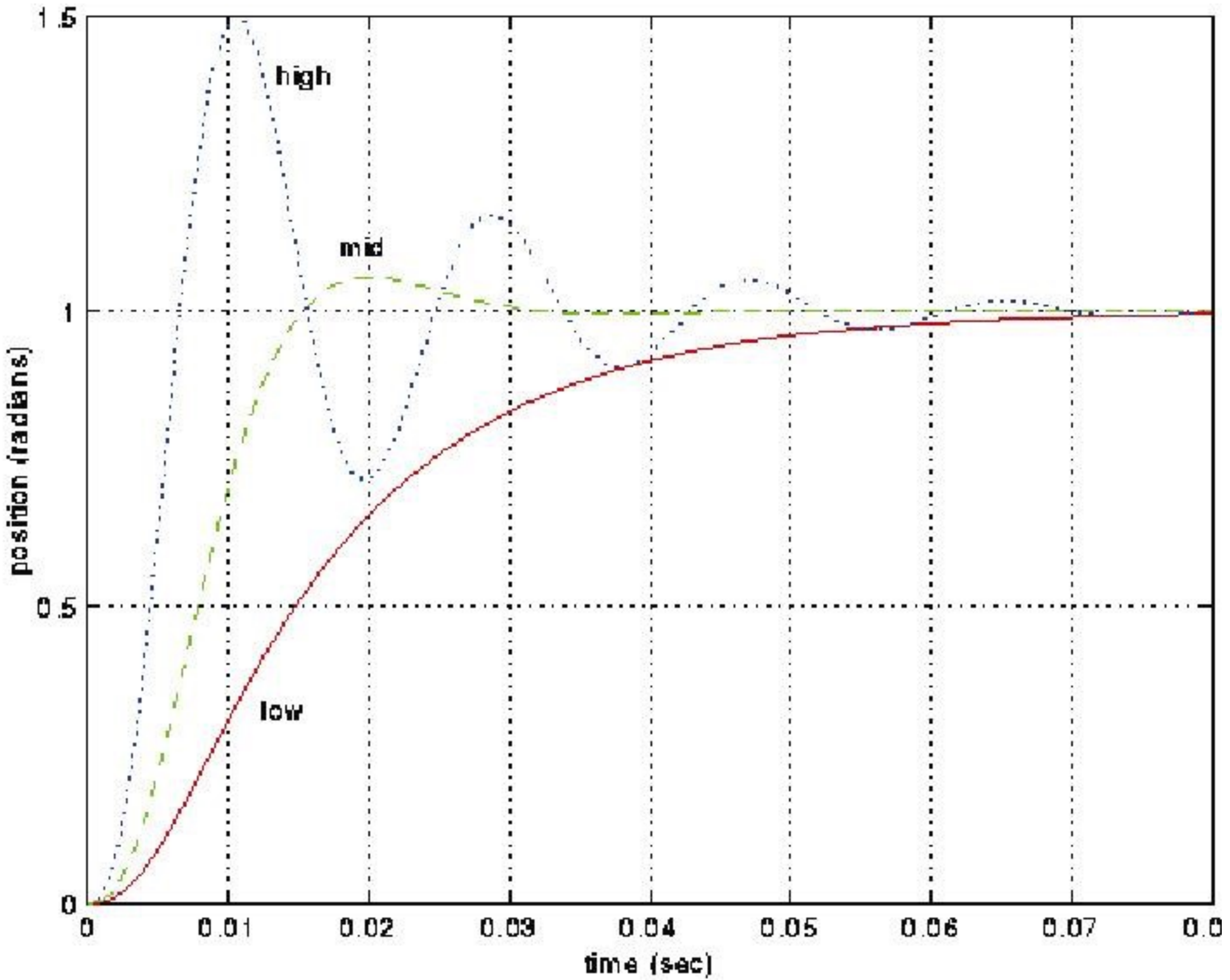
$$q(t) = e^{\frac{-Bt}{2J}}\left[(C_1+C_2)\cos(\frac{\omega t}{2}) + j(C_1-C_2)\sin(\frac{\omega t}{2})\right]$$

  - Oscillates with frequency

$$f = \frac{2\pi}{\omega}\ Hz$$

If B is small and K_{PE} is large: unstable!

Example Step Responses (goal: 1 radian)



PI, PID controllers

- PE controllers can lead to
  - Steady state error
  - Unstable behavior
- Add Integral Term:

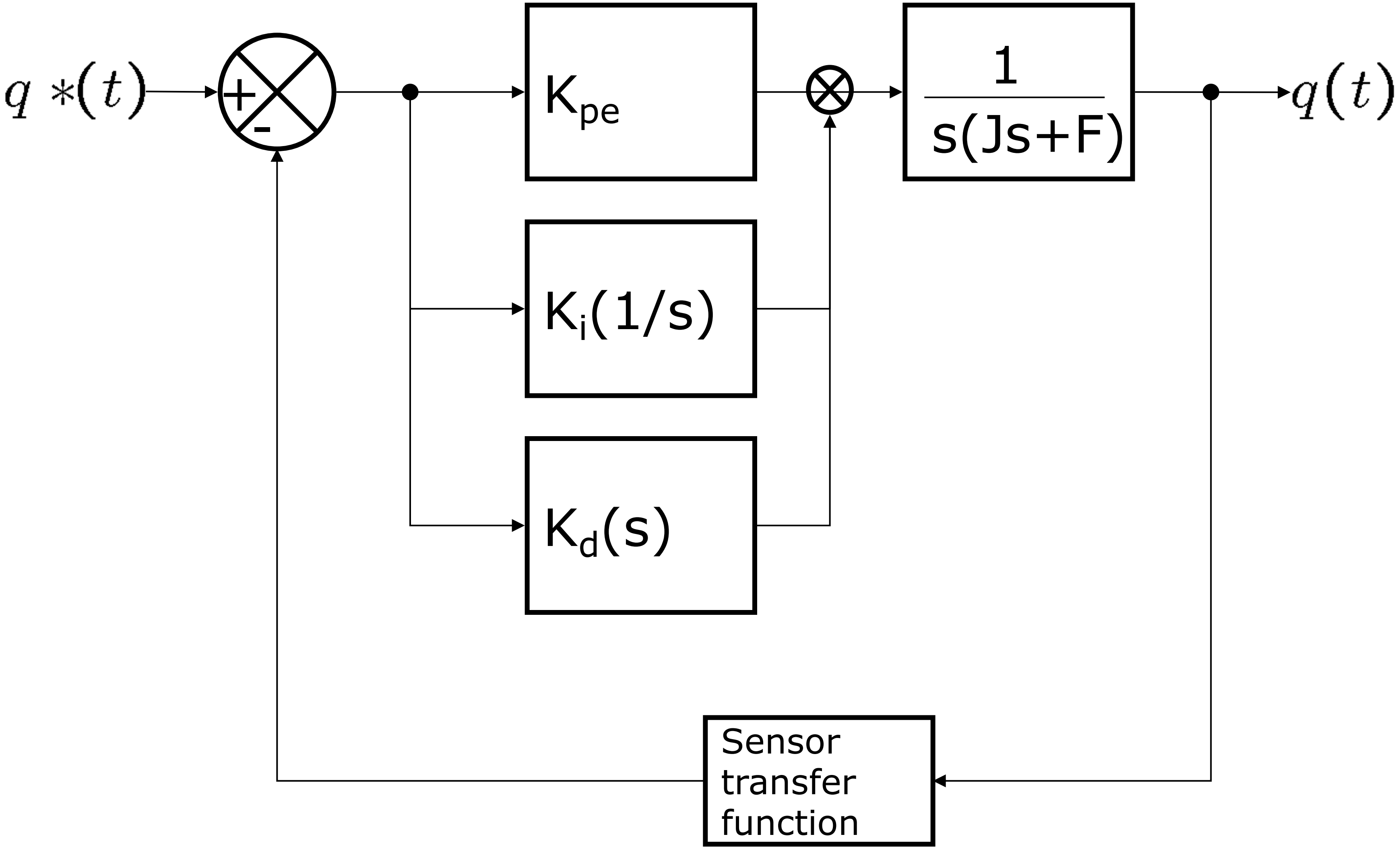
$$\tau_c = K_{pe}(q_r - q_m) + K_I \int_0^t q_r - q_m(u)du$$

….but now we can have overshoot

- Add derivative term (PID Controller)

$$\tau_c = K_{pe}(q_r - q_m) + K_I \int_0^t q_r - q_m(u)du - K_d\dot{q}(t)$$

Block Diagram of PE controller



Set Gains for PID Controller

- wlog set $q^* = 0$ (we al ready have $\dot{q}^* = 0$)

$$J\ddot{q}(t) + B\dot{q}(t) = -K_{pe}q(t) - K_I \int_0^t q(u)du - K_d\dot{q}(t)$$

- Convert to third order equation

$$J\dddot{q} + (B + K_d)\ddot{q} + K_{pe}\dot{q} + K_I q = 0$$

- Solution will be of the form

$$q(t) = f(J, B, K_{pe}, K_I, K_D, \omega, t)$$

  - where

$$\omega = \sqrt{g(J, B, K_{pe}, K_I, K_D)}$$

Set Gains for PID Controller

- Critically damped when ω = 0 or

$$g(J, B, K_{pe}, K_I, K_D) = 0$$

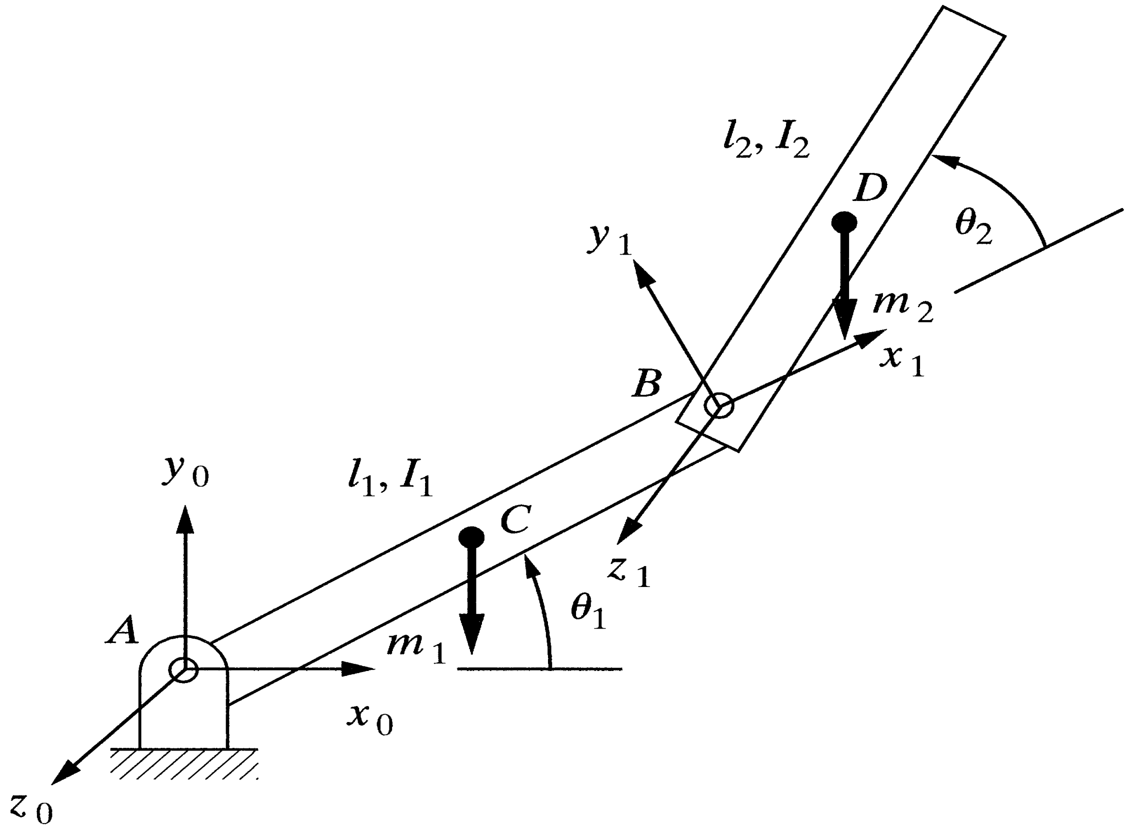
  - An equation in 3 unknowns
  - Need two more constraints:
    - Minimum energy
    - Minimum error
    - Minimum jerk
  - And we need the solution to double minimization
    - Beyond the scope of this class – topic of optimal control class

Problems with Independent Joint Control

- Synchronization ?
  - If one joint does not follow the trajectory, where is the end-effector???
- Ignores dynamic effects
  - Links are connected
  - Motion of links affects other links
  - Could be in-efficient use of energy

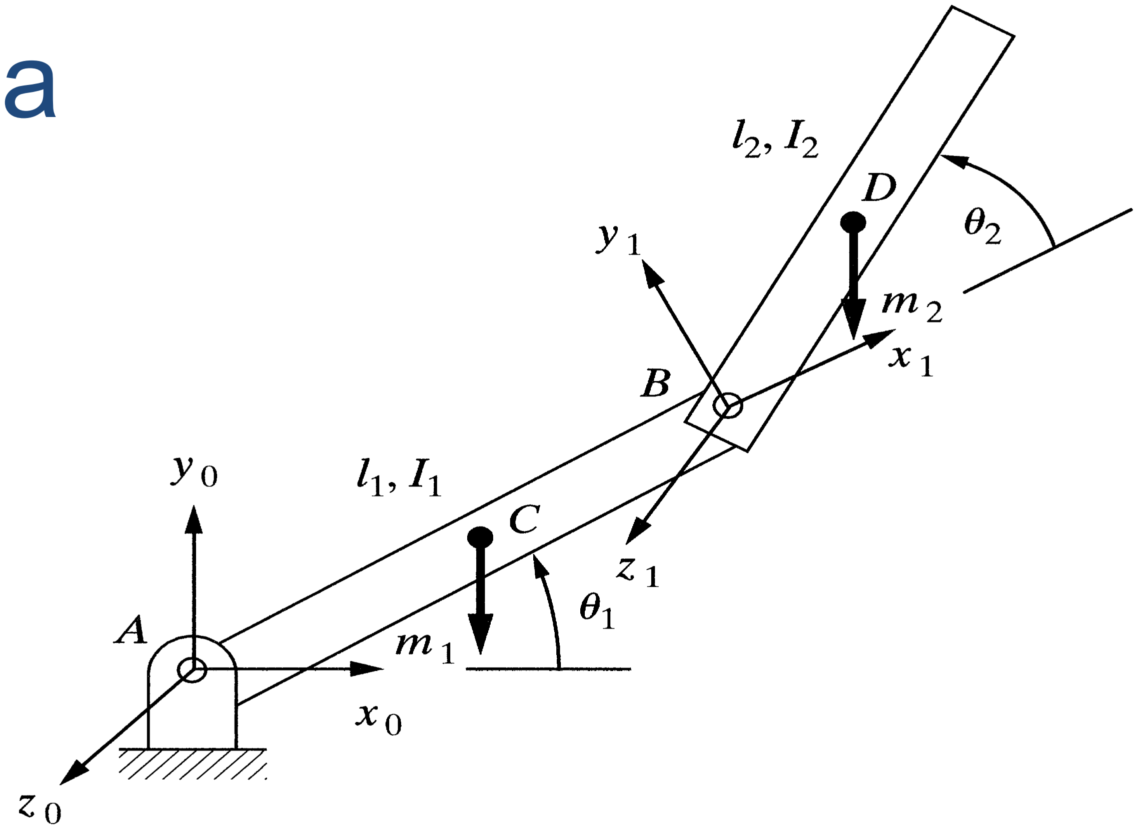
Dynamics

- For all links, consider:
  - Linear Momentum
  - Angular Momentum
  - Gravity Force (potential energies)
  - Force/ Torque of Actuators (in joints)



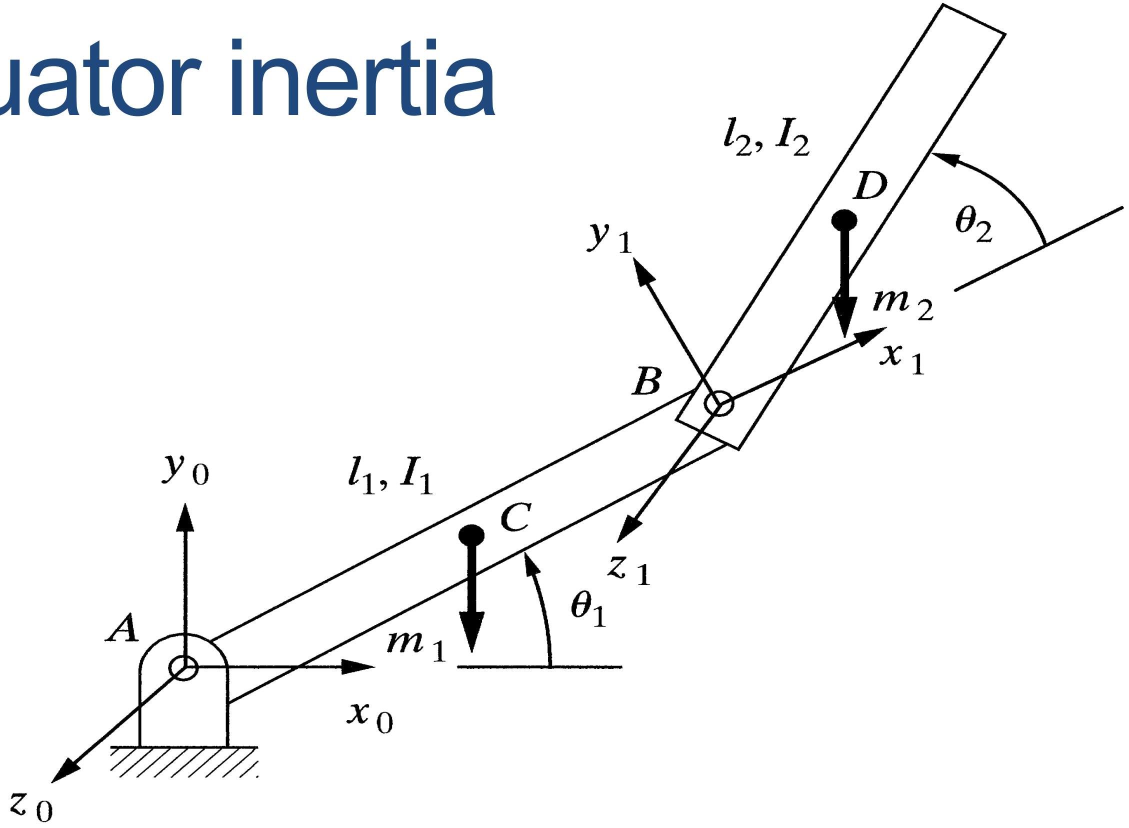
Effective Moments of Intertia

- To Simplify the equation of motion, Equations can be rewritten in symbolic form.



$$\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = \begin{bmatrix} D_{ii} & D_{ij} \\ D_{ji} & D_{jj} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_i \\ \ddot{\theta}_j \end{bmatrix} + \begin{bmatrix} D_{iii} & D_{ijj} \\ D_{jii} & D_{jjj} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_2^2 \end{bmatrix} = \begin{bmatrix} D_{iii} & D_{ijj} \\ D_{jii} & D_{jjj} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \dot{\theta}_2 \\ \dot{\theta}_2 \dot{\theta}_1 \end{bmatrix} + \begin{bmatrix} D_i \\ D_j \end{bmatrix}$$

Final equations without actuator inertia



$$T_1 = \left(\frac{1}{3}m_1 l^2 + \frac{4}{3}m_2 l^2 + m_2 l^2 C_2\right)\ddot{\theta}_1 + \left(\frac{1}{3}m_2 l^2 + \frac{1}{2}m_2 l^2 C_2\right)\ddot{\theta}_2$$

$$+\left(\frac{1}{2}m_2 l^2 S_2\right)\dot{\theta}_2^2 + \left(m_2 l^2 S_2\right)\dot{\theta}_1\dot{\theta}_2 + \frac{1}{2}m_1 g l C_1 + \frac{1}{2}m_2 g l C_{12} + m_2 g l C_1 + I_{1(act)}\ddot{\theta}_1$$

$$T_2 = \left(\frac{1}{3}m_2 l^2 + \frac{1}{2}m_2 l^2 C_2\right)\ddot{\theta}_1 + \left(\frac{1}{3}m_2 l^2\right)\ddot{\theta}_2 + \left(\frac{1}{2}m_2 l^2 S_2\right) + \frac{1}{2}m_2 g l C_{12} + I_{2(act)}\ddot{\theta}_1$$

Force Control

- Force Control:
  - The robot encounters with unknown surfaces and manages to handle the task by adjusting the uniform depth while getting the reactive force.

- Examples:
  - Tapping a Hole - move the joints and rotate them at particular rates to create the desired forces and moments at the hand frame.
  - Peg Insertion – avoid the jamming while guiding the peg into the hole and inserting it to the desired depth.

OPTIMAL CONTROL

Model Predictive Control

Linear Quadratic Regulator (LQR): For the continuous-time linear system

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}\delta \qquad \text{with} \qquad \mathbf{x}(0) = \mathbf{x}_{\text{init}}$$

and quadratic cost functional defined as ($\mathbf{Q}$ is a diagonal weight matrix)

$$J = \frac{1}{2}\int_0^{\infty} \Delta\mathbf{x}^T(t)\mathbf{Q}\Delta\mathbf{x}(t) + q\delta(t)^2\,dt$$

with $\Delta\mathbf{x}(t) = \mathbf{x}(t) - \mathbf{x}_{\text{target}}$. The feedback control $\delta(t)$ that minimizes J is given by

$$\delta(t) = -\mathbf{k}^T(t)\Delta\mathbf{x}(t)$$

with $\mathbf{k}(t) = \frac{1}{q}\mathbf{b}^T\mathbf{P}(t)$ and $\mathbf{P}(t)$ the solution to a Ricatti equation (no details here).

Model Predictive Control

Generalizes LQR to:

- **Non-linear** cost function and dynamics (consider straight road leading into turn)
- **Flexible:** allows for receding window & incorporation of constraints
- **Expensive:** non-linear optimization required at every iteration (for global coordinates)

Formally:

$$\underset{\delta_1,\ldots,\delta_T}{\operatorname{argmin}} \quad \sum_{t=1}^{T} C_t(\mathbf{x}_t, \delta_t) \qquad \text{(sum of costs)}$$

$$s.t. \quad \mathbf{x}_1 = \mathbf{x}_{\text{init}} \qquad \text{(initialization)}$$

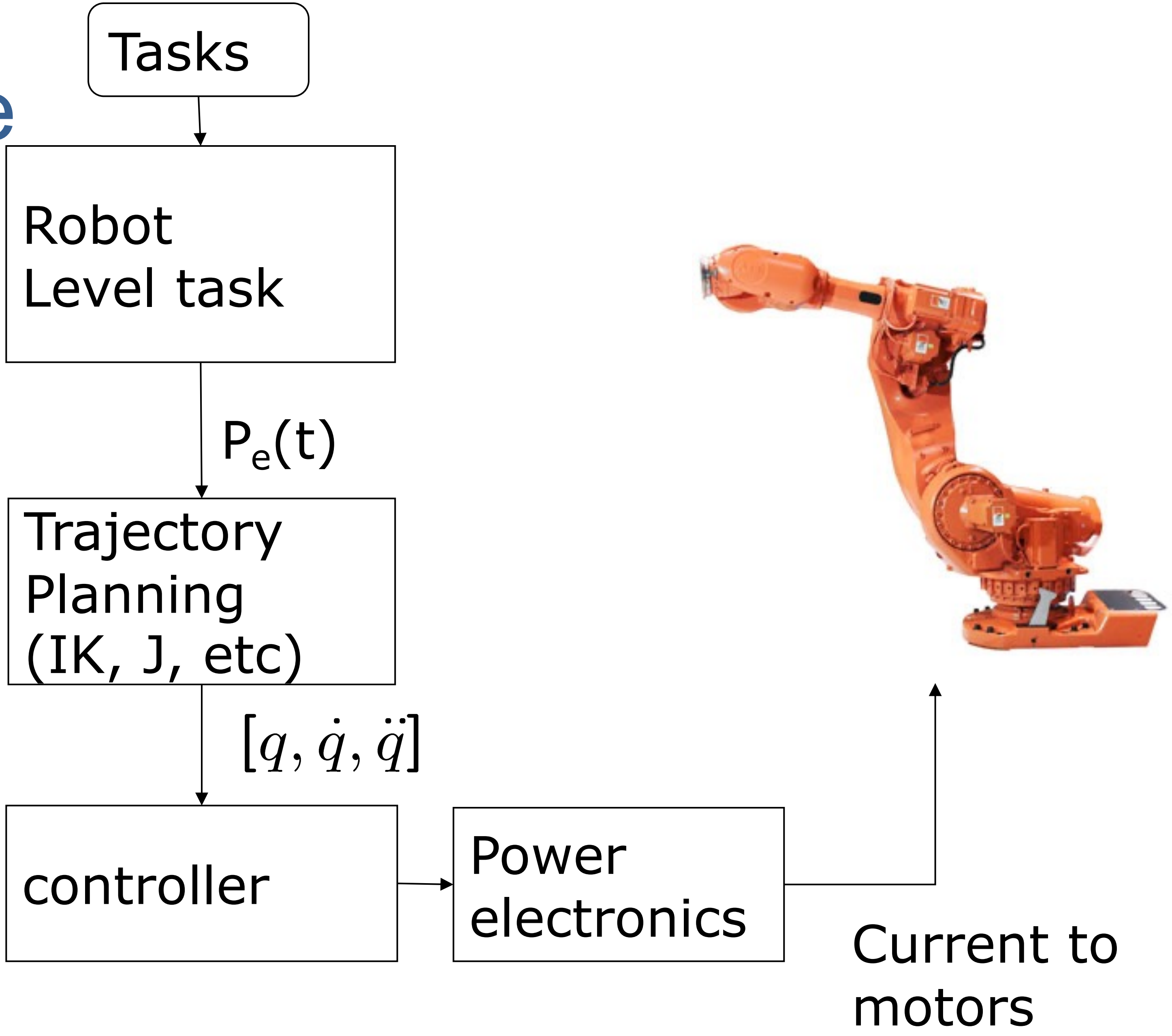
$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \delta_t) \qquad \text{(dynamics model)}$$

$$\underline{\delta} \leq \delta_t \leq \overline{\delta} \qquad \text{(constraints)}$$

- Unroll dynamic model T times ⇒ apply non-linear optimization to find $\delta_1, \ldots \delta_T$

HARDWARE

Hardware


Tasks
Robot Level task
$P_e(t)$
Trajectory Planning (IK, J, etc)
$[q, \dot{q}, \ddot{q}]$
controller
Power electronics
Current to motors

Hardware

Tasks

Robot Level task

$P_e(t)$

Trajectory Planning (IK, J, etc)

$[q, \dot{q}, \ddot{q}]$

controller

Power electronics

Current to motors

Control Hierarchy

- Assume we have a goal trajectory
- Calculate needed joint speeds using Kinematics =>
- Desired joint speeds
  - Typically not just one joint =>
  - Many motor controllers, motors, encoders
- Motor control loop
- Pose control loop

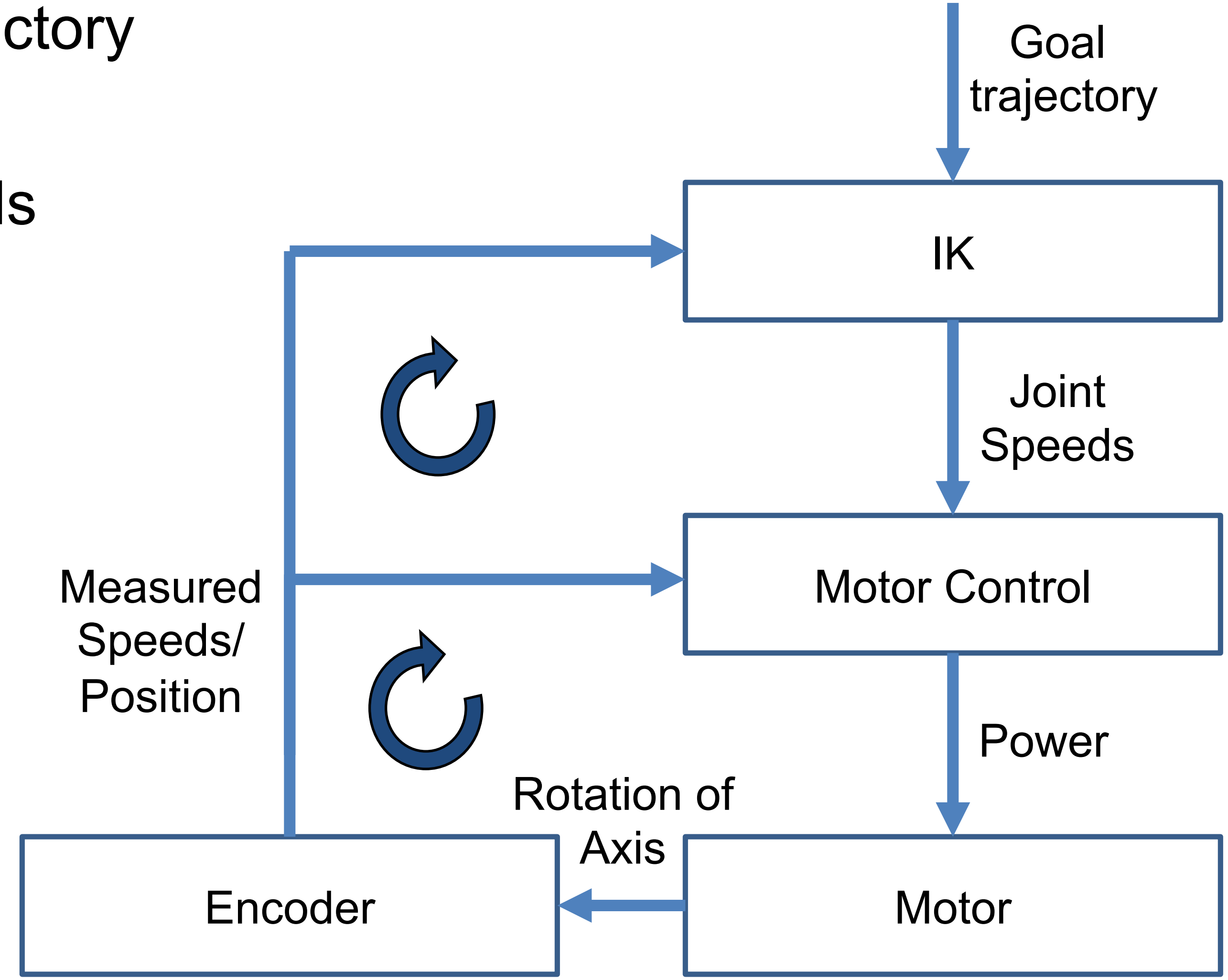
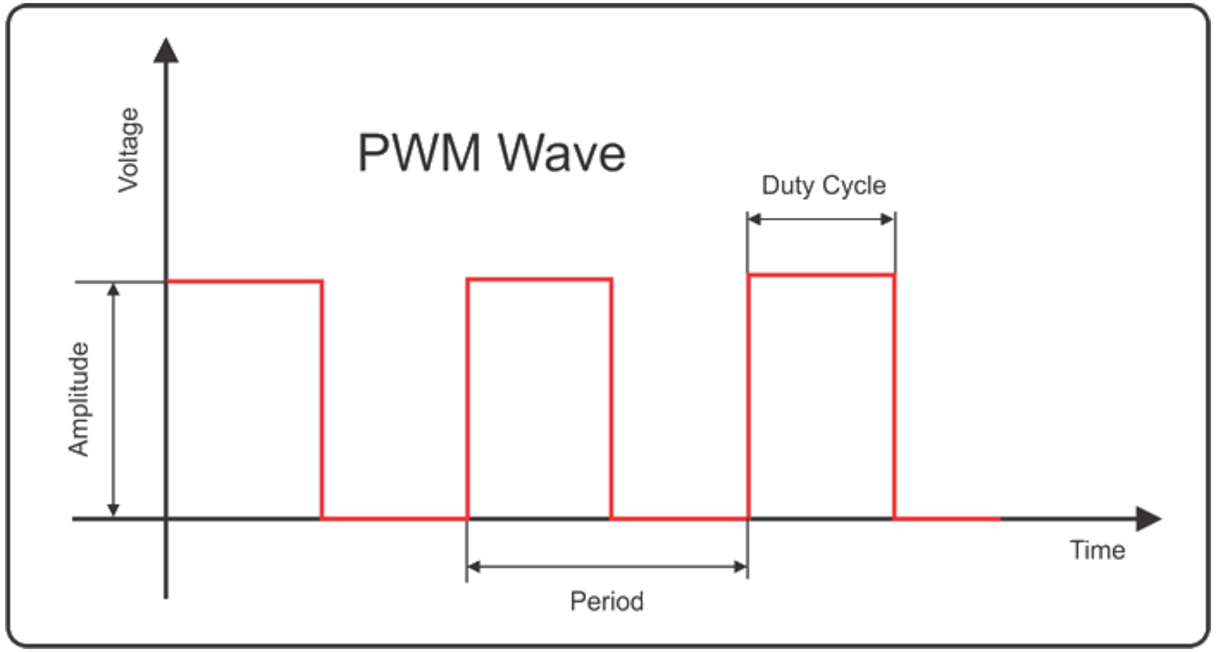
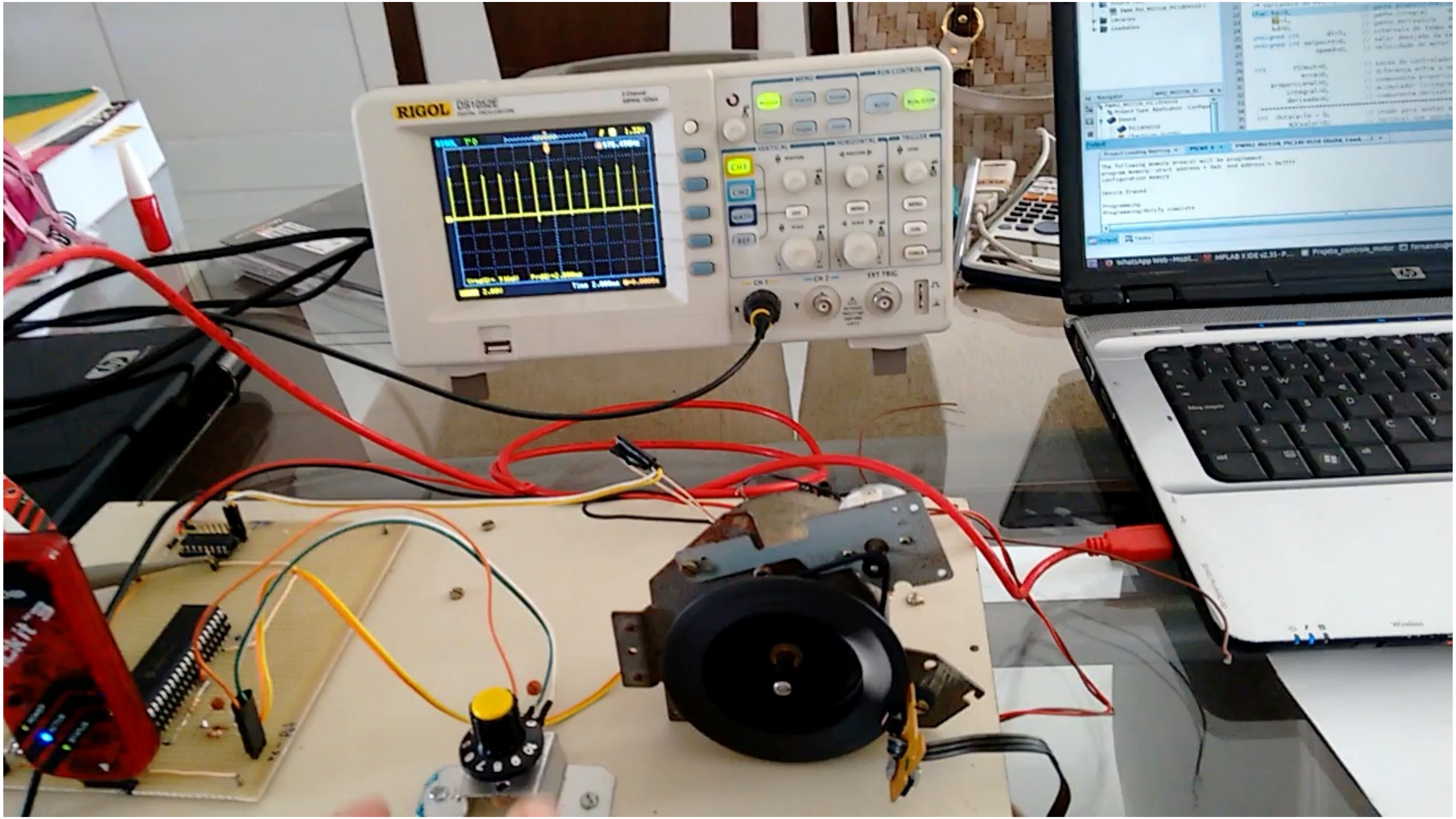


Image: zembedded.com

Pulse Width Modulation



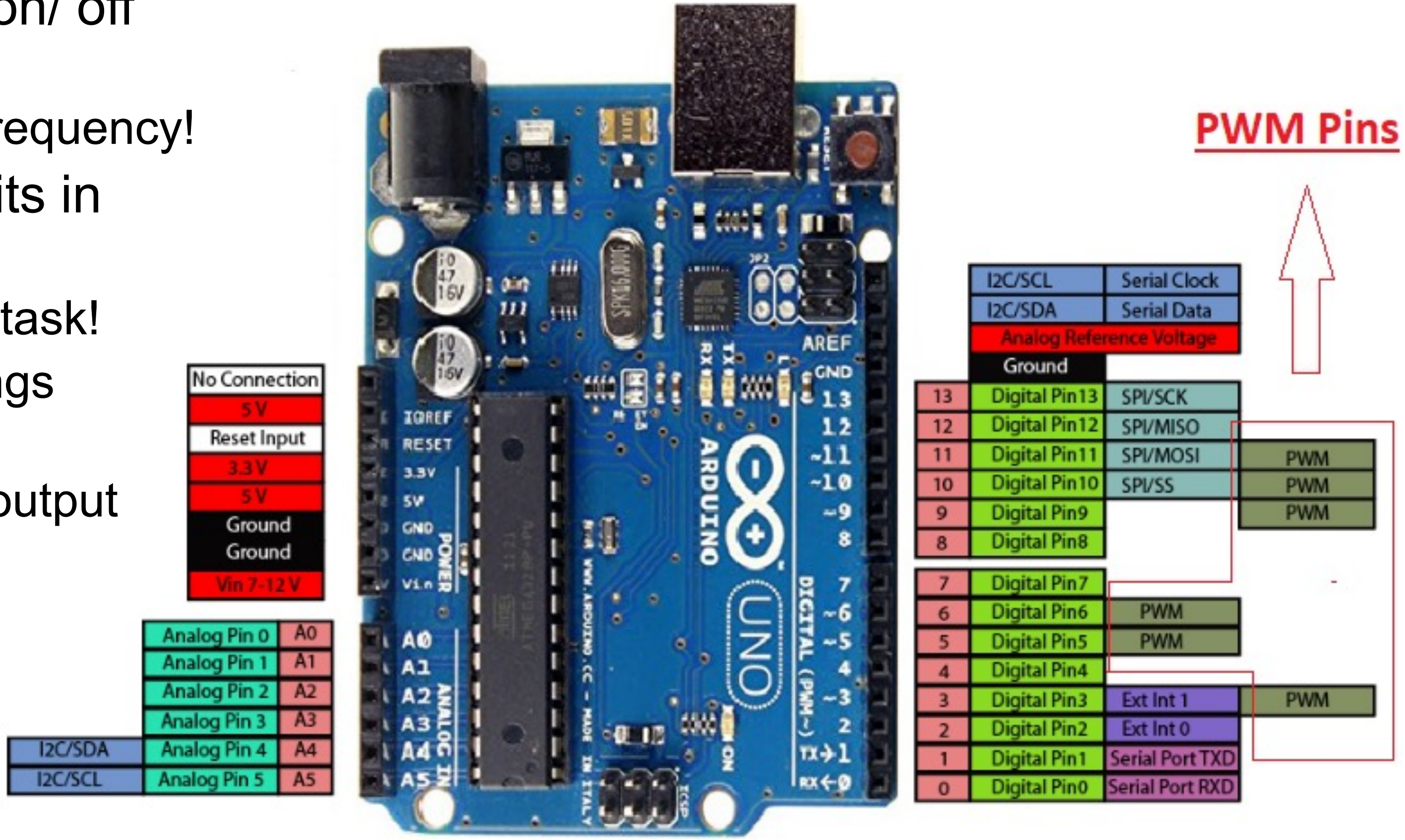
- How can Controller control power?
  - Cannot just tell the motor “use more power”
  - Output of (PID) controller is a signal
  - Typical: Analogue signal
- Pulse Width Modulation (PWM)
  - Signal is either ON or OFF
  - Ratio of time ON vs. time OFF in a given interval: amount of power
  - Frequency in kHz (= period less than 1ms)
  - Very low power loss
- Signal (typica 5V or 3.3V) to Motor Driver
- Used in all kinds of applications:
  - electric stove; audio amplifiers, computer power supply (hundreds of kHz!)



https://www.youtube.com/watch?v=4QzyG5g1blg

PWM Generation

- Motor Control:
  - Frequency in kHz:
  - Smooth motion of motor wanted
  - Use inertia of the motor to smooth the on/ off cycle
    - Still: Sound of motor often from control frequency!
  - High frequency => use dedicated circuits in microcontroller to generate PWM!
    - CPU is not burdened with this mundane task!
    - CPU would suffer from inconsistent timings
      - Interrupts; preemptive computing
    - E.g. Arduino (ATmega48P) has 6 PWM output channels
    - Timer running independently of CPU
    - Comparing to a set register value – if it is up, the output signal is switched

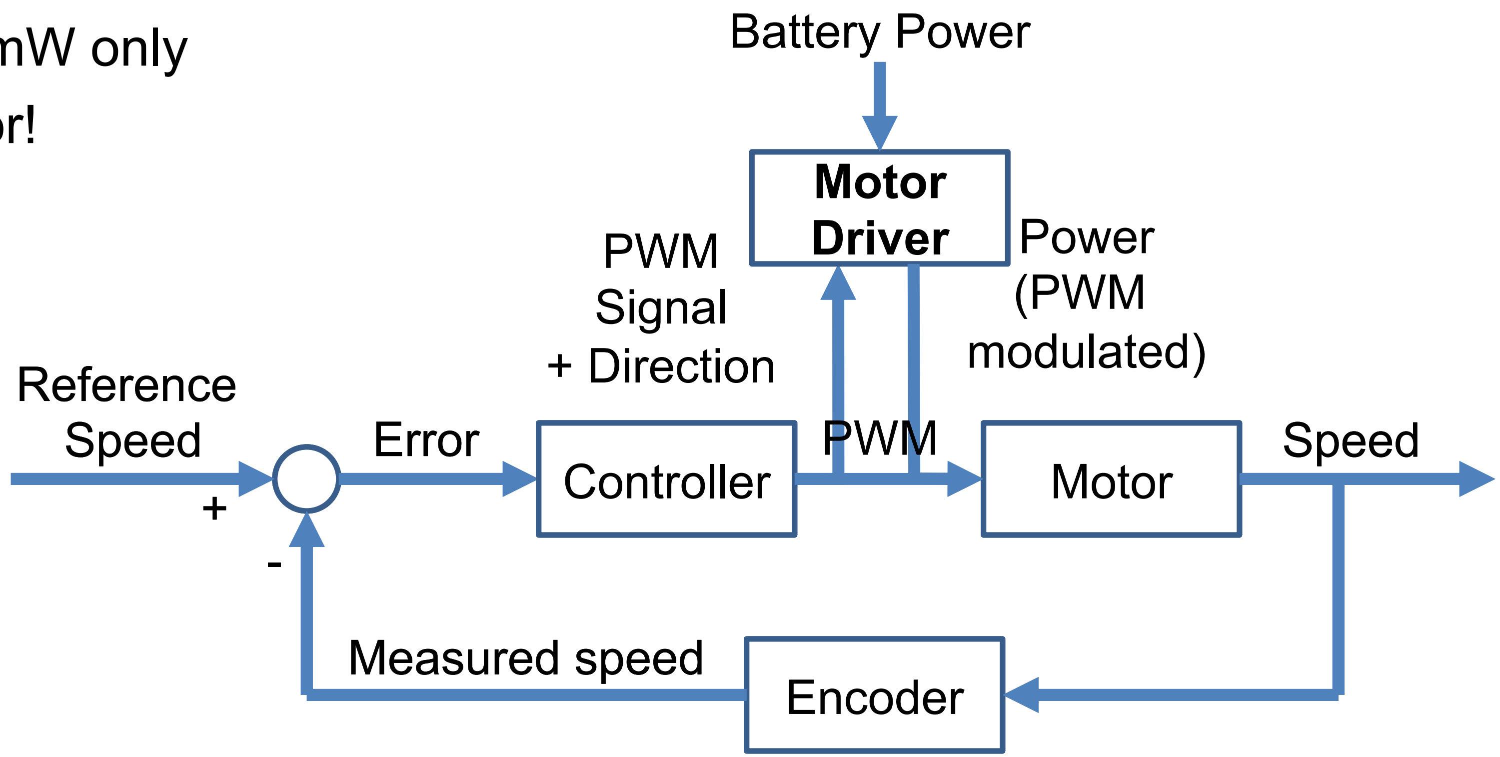


MOTOR DRIVER

Power to the Motor

- Direct Current Motor (DC Motor):
  - Two wires for power input
  - Directly connect DC motor to PWM signal?
  - Limited current!
  - E.g.: Arduino: max 30mA => 150mW only
  - Clearpath Jackal: 250W per motor!

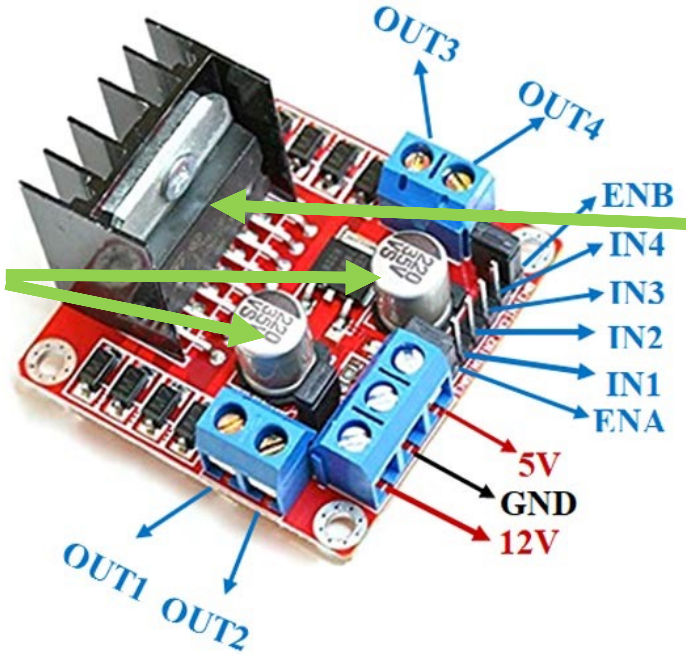
- Need a device to power the motor
- Mobile robots: battery power!

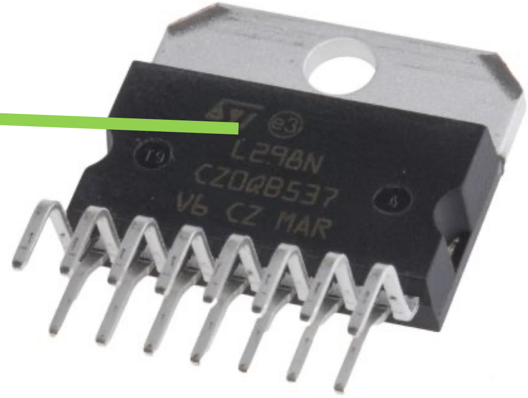


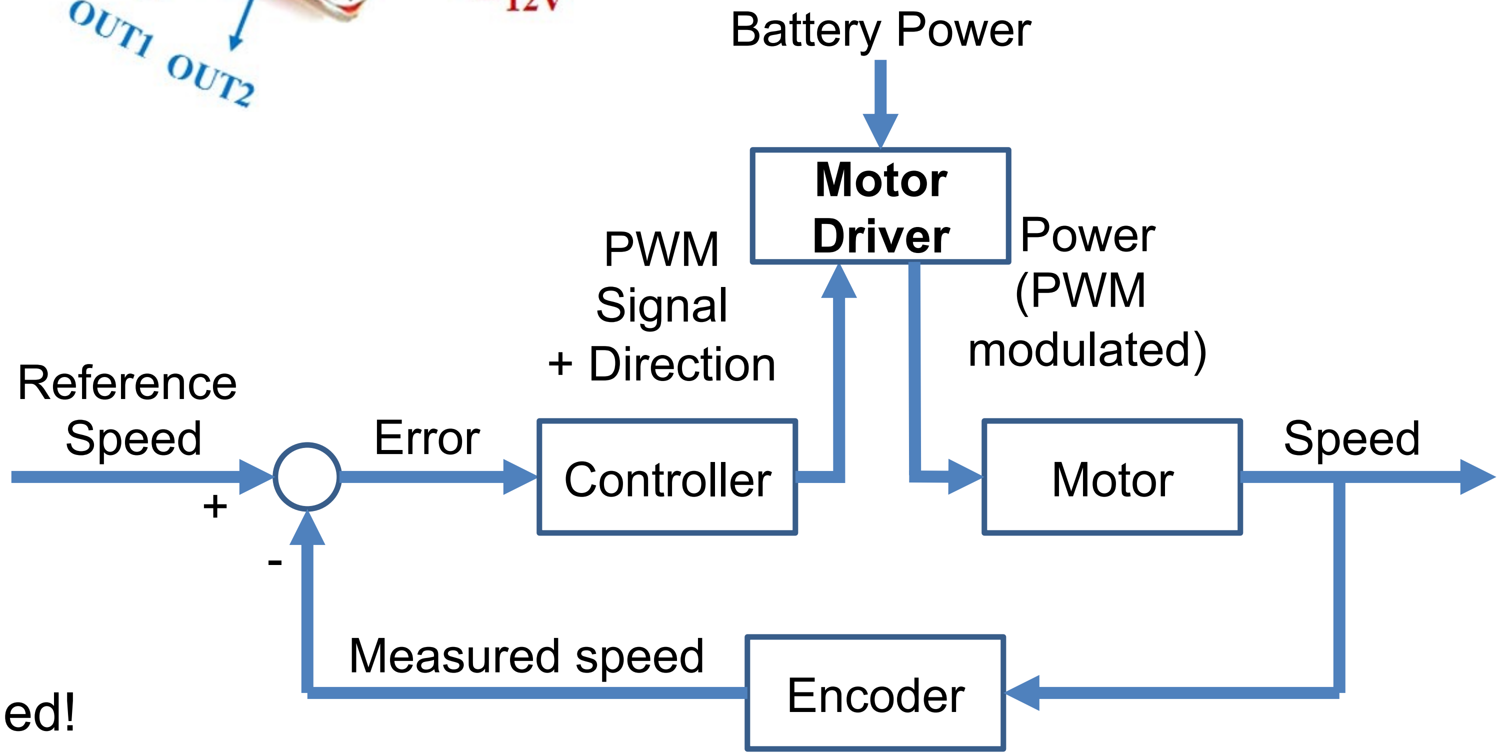
Motor Driver

- Motor Driver
  - Input:
    - PWM signal
    - Direction of rotation
    - Battery + & -
    - Optional: Enable =>
      - Emergency Stop
  - Output:
    - Two lines to the DC motor
  - Popular: L298N dual motor driver
    - Up to 48V & 4A
  - High Efficiency (maybe 95%) – but still get's hot – cooling needed!

Capacitors to smooth output power



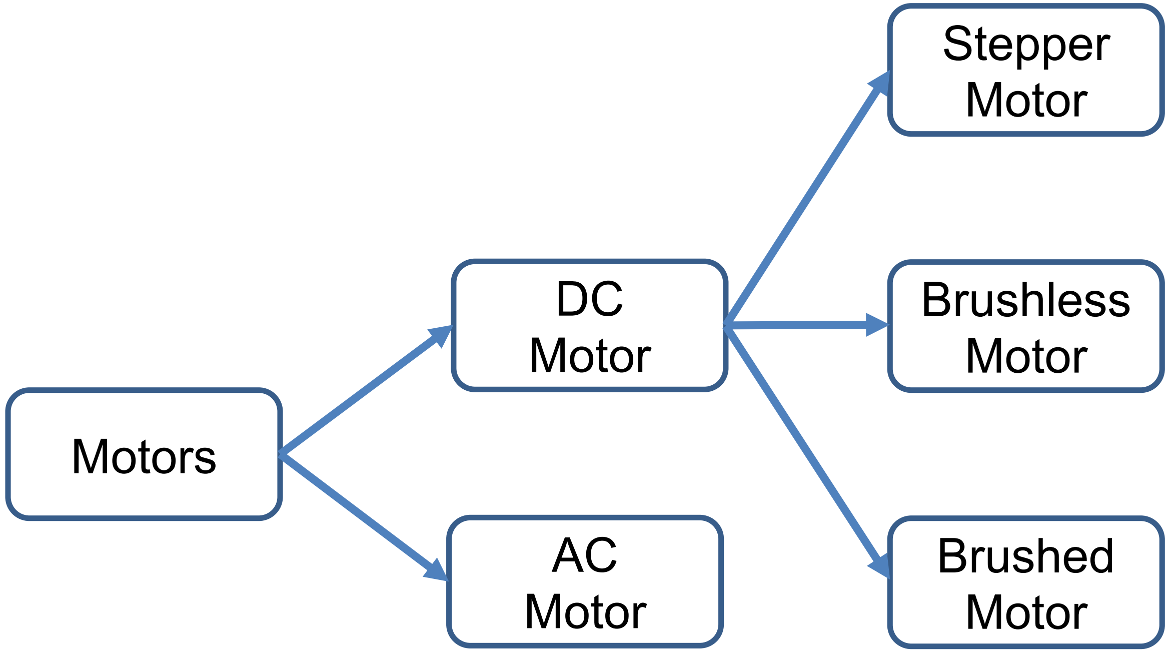




MOTORS

Electrical Motor Types

- DC Motor: Direct Current Motor
- AC Motor: Alternating Current Motor
- Stepper motor:
  - Switching power steps one tooth/ coils forward
  - Open loop control: no encoder needed
  - Low resolution; open loop; torque must be well known
- Brushed motor:
  - Use brushes to power rotating coils => low efficiency and high wear
- Brushless (BL) motor:
  - Electronically control which coil to power => high efficiency low wear
  - Need dedicated controller



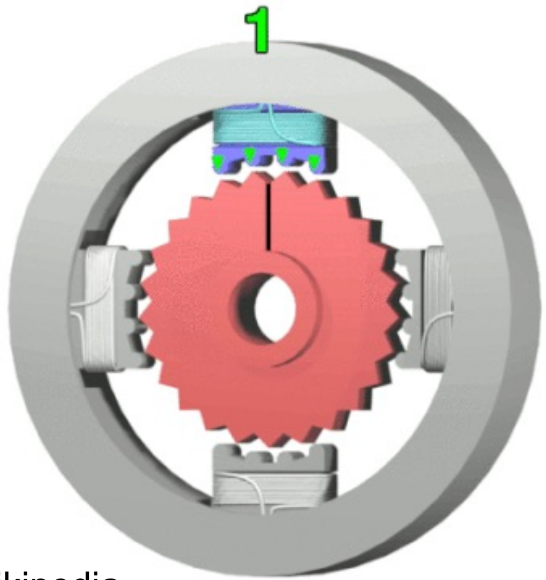
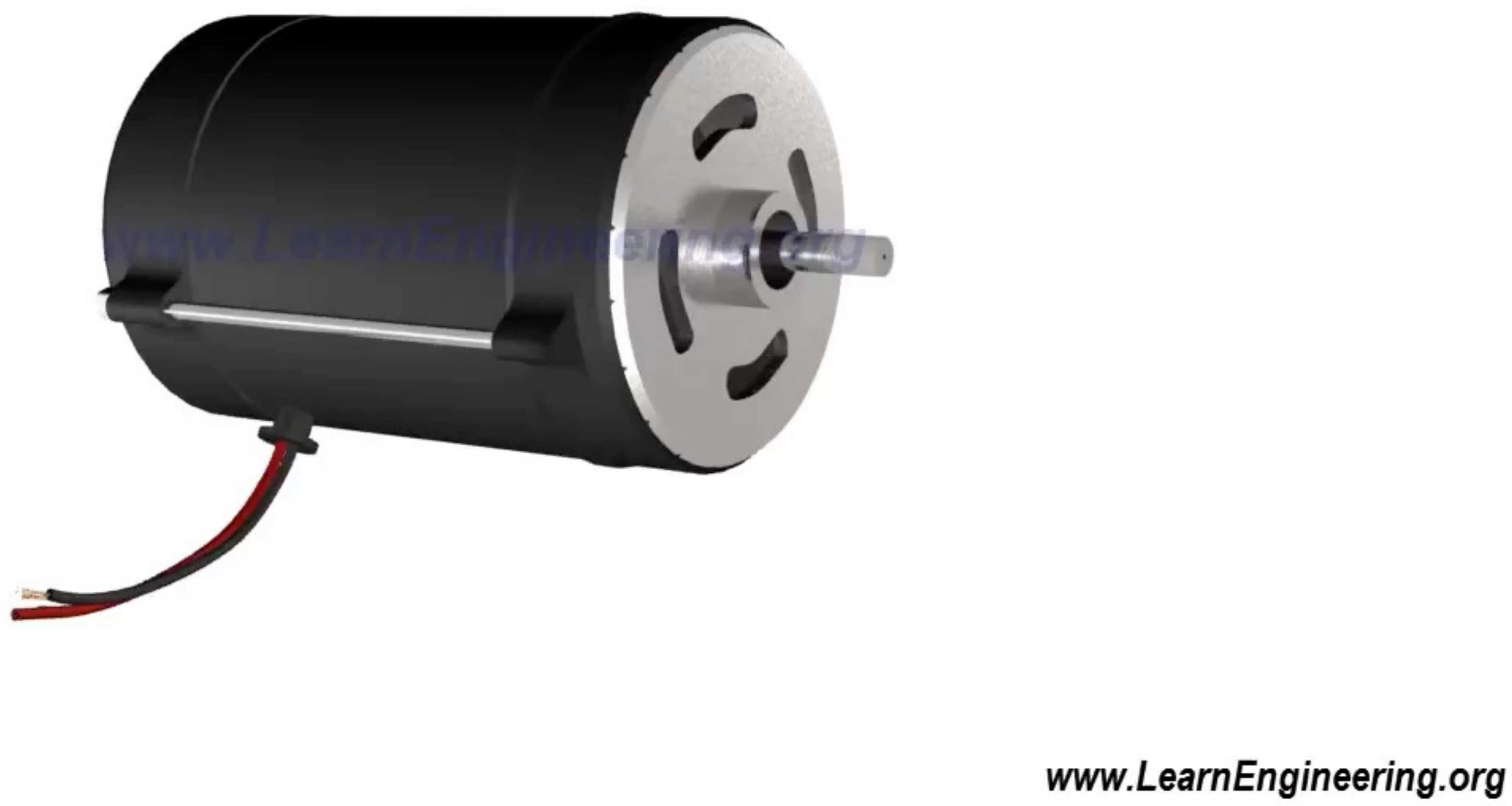


Image: Wikipedia



https://www.youtube.com/watch?v=CWulQ1ZSE3c