Robotics

Erwin M. Bakker| LIACS Media Lab

18-2 2025

Universiteit Leiden

Bij ons leer je de wereld kennen

Organization and Overview

Lecturer:
Dr Erwin M. Bakker (erwin@liacs.nl)
Room LIACS Media Lab (LML)
Please email for a meeting.

Teaching assistants:
TBA

Period: February 11th - May 13th 2025
Time: Tuesday 11.15 - 13.00
Place (Rooms): Van Steenis F1.04
Exceptions:
Gorlaeus Building BM.1.33 on April 1st
Gorlaeus Building BM.1.23 on May 20th

Schedule (tentative, visit regularly):

Date	Subject
11-2	Introduction and Overview
18-2	Locomotion and Inverse Kinematics
25-2	Robotics Sensors and Image Processing
4-3	SLAM + Workshop@Home
11-3	Robotics Vision + Introduction Mobile Robot Challenge
18-3	Project Proposals I (by students)
25-3	Project Proposals II (by students)
1-4	Robotics Reinforcement Learning + RL Workshop@Home
8-4	Project Progress Reports I
15-4	Project Progress Reports II
22-4	Mobile Robot Challenge I
29-4	Mobile Robot Challenge II
6-5	TBA
13-5	Project Demos I
20-5	Project Demos II
27-5	Project Deliverables

Website: http://liacs.leidenuniv.nl/~bakkerem2/robotics/

Grading (6 ECTS):

- Presentations and Robotics Project (60% of grade).
- Class discussions, attendance, 2 assignments (pass/no pass)
- 2 Workshops (0-10) (20% of the grade).
- Mobile Robot Challenge (0-10) (20% of the grade)
- ***It is necessary to be at every class and to complete every workshop and assignment.***

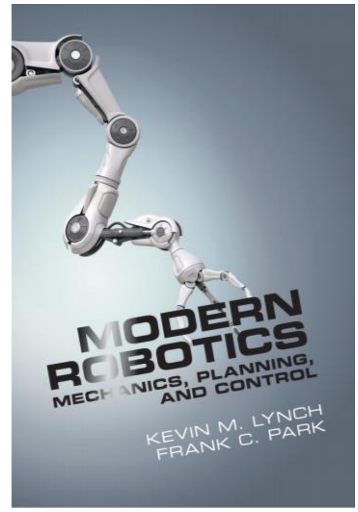
Universiteit Leiden. Bij ons leer je de wereld kennen

Overview

- Robotic Actuators

- Degree of Freedom of Robots
- Introduction to:
  - Configuration Space
  - Rigid Body Motion
  - Forward Kinematics
  - Inverse Kinematics

- Link: http://modernrobotics.org

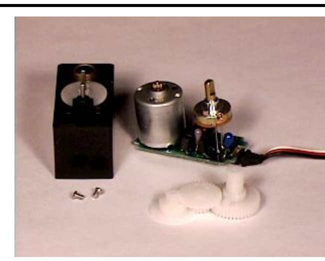


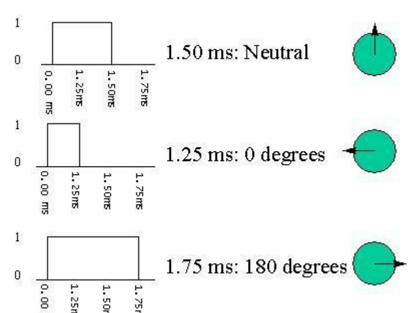
K.M. Lynch, F.C. Park, Modern Robotics: Mechanics, Planning and Control, Cambridge University Press, 2017

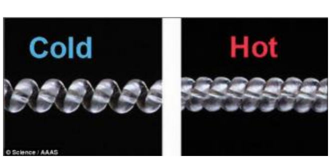
Universiteit Leiden. Bij ons leer je de wereld kennen

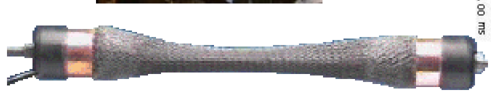
Robotics Actuators

- **Electro motors**
- **Servo's**
- **Stepper Motors**
- **Brushless motors**
- **Solenoids**
- Hydraulic, **pneumatic actuator's**
- Magnetic actuators
- Artificial Muscles
- Etc.



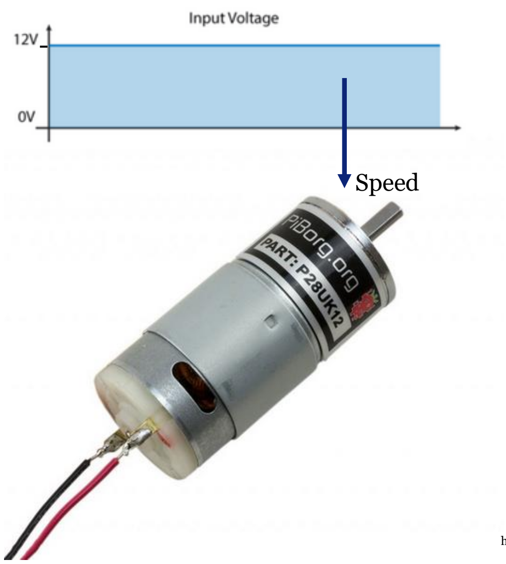


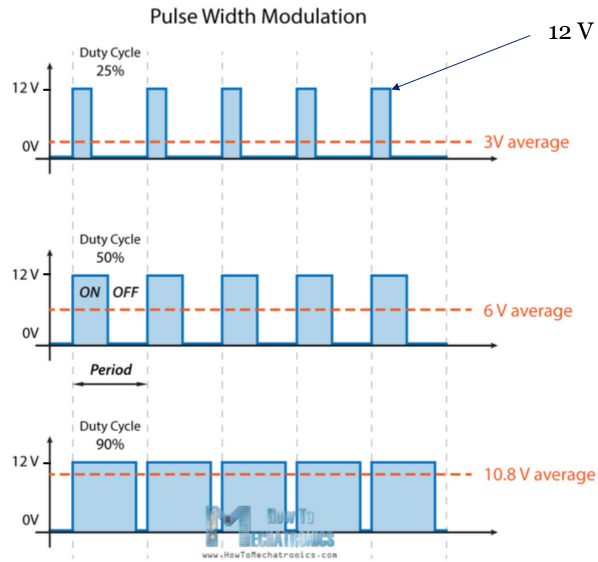




Universiteit Leiden. Bij ons leer je de wereld kennen

DC Motors



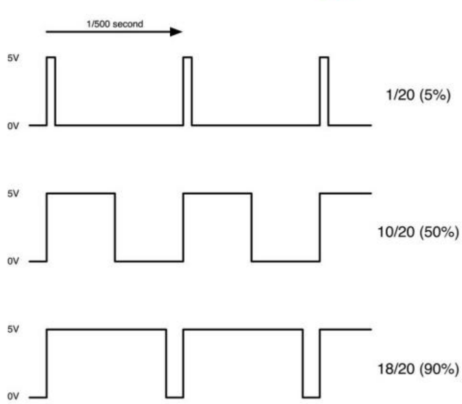


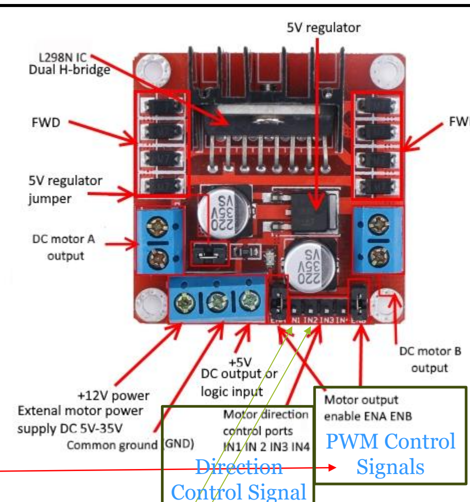
https://howtomechatronics.com/how-it-works/electronics/how-to-make-pwm-dc-motor-speed-controller-using-555-timer-ic/

Direct Current (DC) Electro Motors:

- Duty Cycle

Brushed DC (BDC) motor controller Using L298N





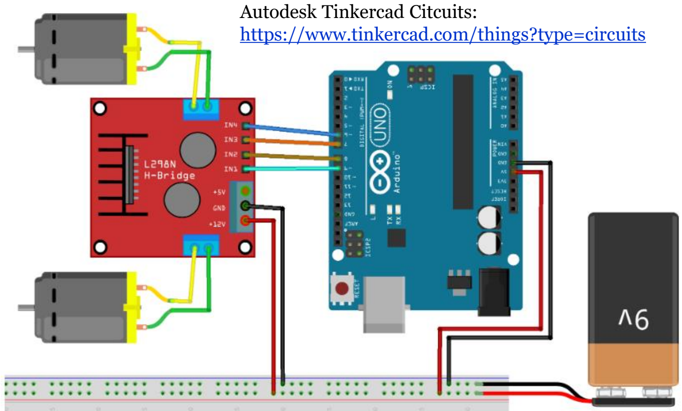
Loop:
PWM(EN**A**,128);
DigitalWrite(IN**1**, HIGH);
DigitalWrite(IN**2**,LOW);
PWM(EN**B**,64);
DigitalWrite(IN**3**, HIGH);
DigitalWrite(IN**4**,LOW);

Direct Current (DC) Electro Motors:

Brushed DC (BDC) motor controller Using L298N

MotoDriver 2	Arduino
Input 1	9
Input 2	8
Input 3	7
Input 4	6

Autodesk Tinkercad Citcuits:
https://www.tinkercad.com/things?type=circuits




```
// connect motor controller pins
   to Arduino digital pins
// motor one
int enA = 10;
int in1 = 9;
int in2 = 8;
// motor two
int enB = 5;
int in3 = 7;
int in4 = 6;

void setup()
{ // set all the motor control pins to
  // outputs
pinMode(enA, OUTPUT);
pinMode(enB, OUTPUT);
pinMode(in1, OUTPUT);
pinMode(in2, OUTPUT);
pinMode(in3, OUTPUT);
pinMode(in4, OUTPUT);
}

void demoOne()
{
// run the motors in one direction at a fixed speed
// Turn on and set forward direction of  motor A
digitalWrite(in1, HIGH);
digitalWrite(in2, LOW);
// Set speed to PWM Value 200  (range 0~255)
analogWrite(enA, 200);
// Turn on and set backward direction of motor B
digitalWrite(in3, LOW);
digitalWrite(in4, HIGH);
// Set speed to PWM Value 100 (range 0~255)
analogWrite(enB, 200);
delay(2000);
}
```

https://cdn.bodanius.com/media/1/d6d1595_img.pdf

DC Motor Controllers

Pololu Simple Motor Controllers

- USB, TTL Serial, Analog, RC Control, I2C

	Original versions, not recommended for new designs (included for comparison purposes)					G2 versions, released November 2018			
	 SMC 18v7	 SMC 18v15	 SMC 24v12	 SMC 18v25	 SMC 24v23	 SMC G2 18v15	 SMC G2 24v12	 SMC G2 18v25	 SMC G2 24v19
Minimum operating voltage:	5.5 V	5.5 V	5.5 V	5.5 V	5.5 V	6.5 V	6.5 V	6.5 V	6.5 V
Recommended max operating voltage:	24 V(1)	24 V(1)	34 V(2)	24 V(1)	34 V(2)	24 V(1)	34 V(2)	24 V(1)	34 V(2)
Max nominal battery voltage:	18 V	18 V	28 V	18 V	28 V	18 V	28 V	18 V	28 V
Max continuous current (no additional cooling):	7 A	15 A	12 A	25 A	23 A	15 A	12 A	25 A	19 A
USB, TTL serial, Analog, RC control:	✓	✓	✓	✓	✓	✓	✓	✓	✓
I²C control:						✓	✓	✓	✓
Hardware current limiting:						✓	✓	✓	✓
Reverse voltage protection:						✓	✓	✓	✓

https://www.pololu.com/category/94/pololu-simple-motor-controllers

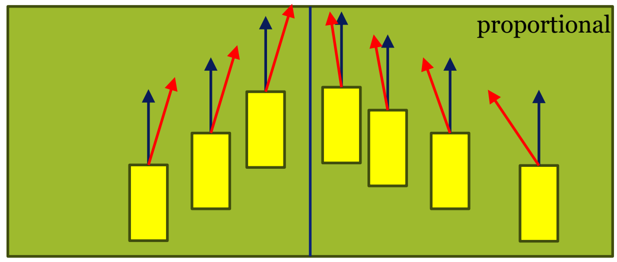
Proportional-Integral-Derivative (PID) Controller

Process Variable is a system parameter that needs to be controlled, e.g., temperature, speed, gripper location, **trajectory**, etc.

Set Point is the desired value (or command value), i.e., temperature = 100 C, speed = 10m/s, arm angle = 27.5 deg., gripper location = (10.2 cm, 3 cm, 12 cm), **desired trajectory**

Error = Set Point - Process Variable

The Control System Algorithm (Compensator) uses the **Error** to determine the **Actuator Output (**e.g. **…, steering angle)** to drive the system: plant, robot, robot arm, **self-driving car**



Idea:

Set the Setpoint

Repeat

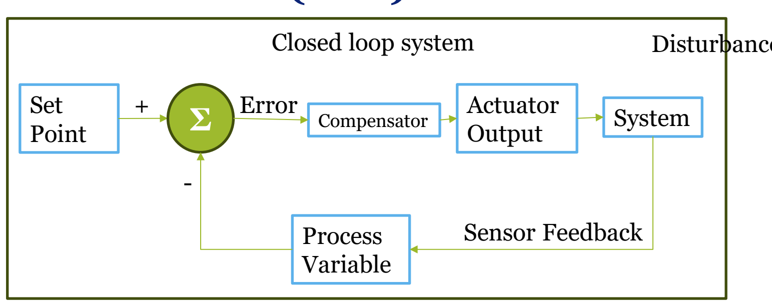
- Process Variable = Read Sensor
- Error = Set Point – Process Variable
  - Calculate a Compensating Response
  - ⇒ actuator output
  - ⇒ System change

Proportional-Integral-Derivative (PID) Controller

Idea:

- Read a sensor
- Compute the desired actuator output
  - Calculate proportional response
  - Calculate integral response
  - Calculate derivative response
  - Sum these to compute the desired actuator output

The Error is the difference between the **Process Variable** and the **Set Point**. This is used as by the Control System Algorithm (Compensator) to determine a desired **Actuator Output** to drive the system (plant, robot, car, arm etc.).

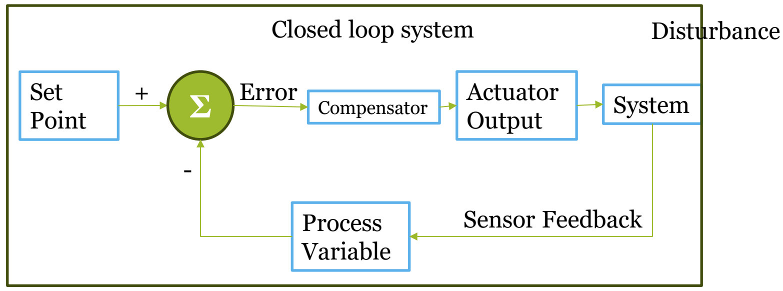


Process Variable is a system parameter that needs to be controlled, e.g., temperature, speed, arm angle, gripper location, trajectory

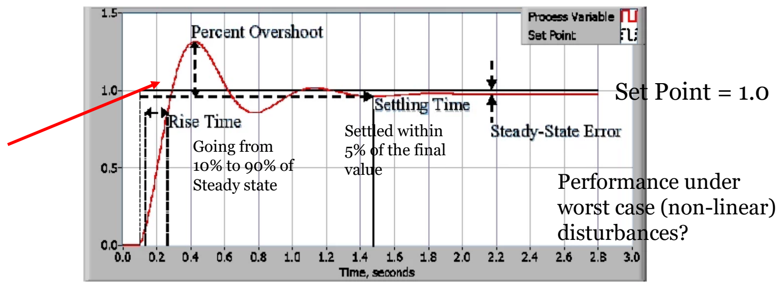
Set Point is the desired value (or command value) for the Process Variable, i.e., temperature = 100 C, speed = 10m/s, arm angle = 27.5 deg., gripper location = (10.2cm, 3 cm, 12 cm), desired trajectory

Proportional-Integral-Derivative (PID) Controller

- **Process Variable** is a system parameter that needs to be controlled, e.g., temperature, speed, …, trajectory
- **Set Point** is the desired value (or command value) for the Process Variable, i.e., temperature = 100 C, speed = 10m/s, desired trajectory
- **The Error** is the difference between the **Process Variable** and the **Set Point**. This is used as by the Control System Algorithm (Compensator) to determine a desired **Actuator Output** to drive the system (cart, robot arm, car etc.), e.g., … steering angle



Example of a response curve of the **Process Variable** of a PID closed loop system:

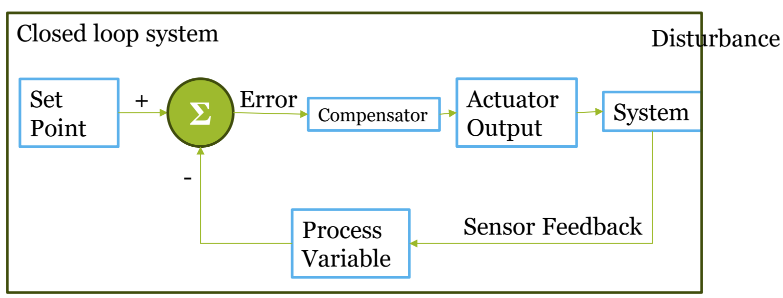


Loop cycle time: interval of time between calls to the control algorithm.
Control scheme depends on the performance requirements.

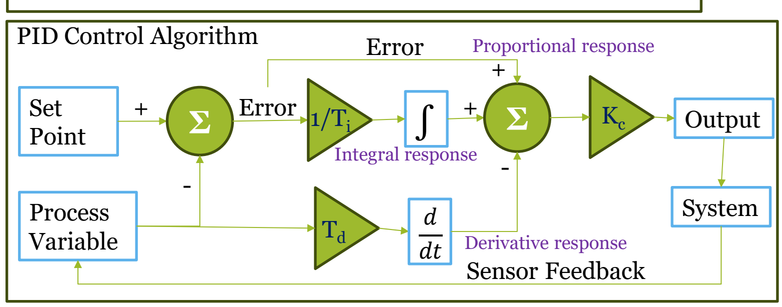
Proportional-Integral-Derivative (PID) Controller

$$u(t) = K_c \cdot \left(e(t) + \frac{\int e(t)\,dt}{T_i} + T_d \cdot \frac{d\,e(t)}{dt} \right)$$

- **Error = Set Point - Process Variable**
- **PID** Control System Algorithm uses the erro to determine a desired **Output** to drive the system (plant, car, robot, etc.).



- **Proportional Response** depends on the error term. The ratio of the output response to the error signal is determined by K_C.
- **Derivative Response** is a response on changes in the Process Variable values. Increasing the derivative time T_d will give stronger reactions, therefor often a small T_d.
- **Integral Response** is the integration of the error term over time, i.e., a small positive error will increase the integral component slowly => continuous increase unless error = 0.



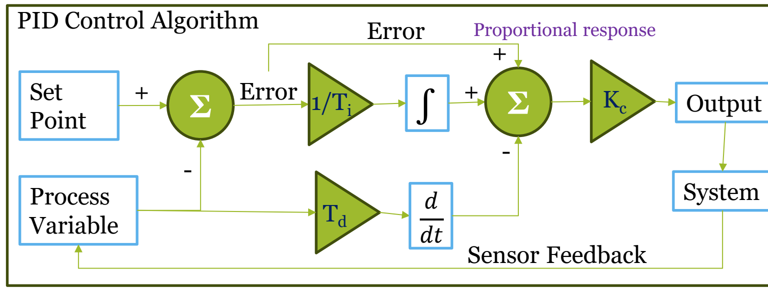
Universiteit Leiden. Bij ons leer je de wereld kennen

Proportional-Integral-Derivative (PID) Controller

$$u(t) = K_c \cdot \left(e(t) + \frac{\int e(t)\,dt}{T_i} + T_d \cdot \frac{d\,e(t)}{dt} \right)$$

- **Error = Set Point - Process Variable**
- **PID** Control System Algorithm uses the error to determine a desired **Output** to drive the system (plant, car, robot, etc.).

- **Proportional Response** depends on the error term. The ratio of the output response to the error signal is determined by K_C.
- **Integral Response** is the integration of the error term over time, i.e., a small positive error will increase the integral component slowly => continuous increase unless error = 0.
- **Derivative Response** is a response on changes in the Process Variable values. Increasing the derivative time T_d will give stronger reactions, therefor often a small T_d.



Tuning: Ziegler-Nichols guess and check for optimal gains for P, I and D.

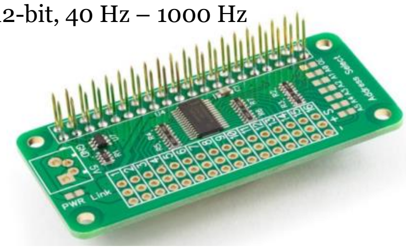
Control	P	T_i	T_d
P	$0.5K_c$	0	0
PI	$0.46K_c$	$P_c/1.2$	0
PID	$0.6K_c$	$0.5P_c$	$P_c/8$

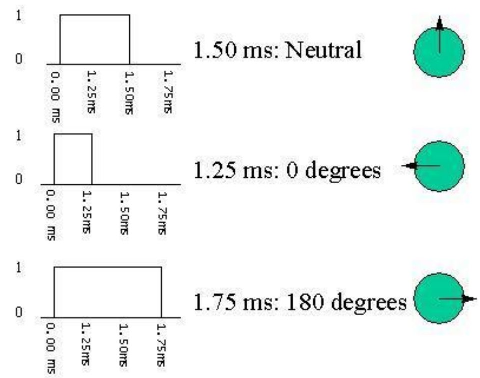
First K_c determined by increasing to point where oscillations with period P_c started.

https://www.youtube.com/watch?v=4Y7zG48uHRo

Servo’s

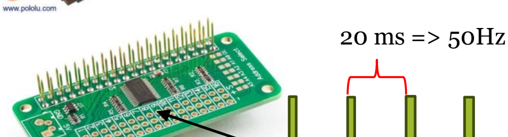
16-ch, 12-bit, 40 Hz – 1000 Hz



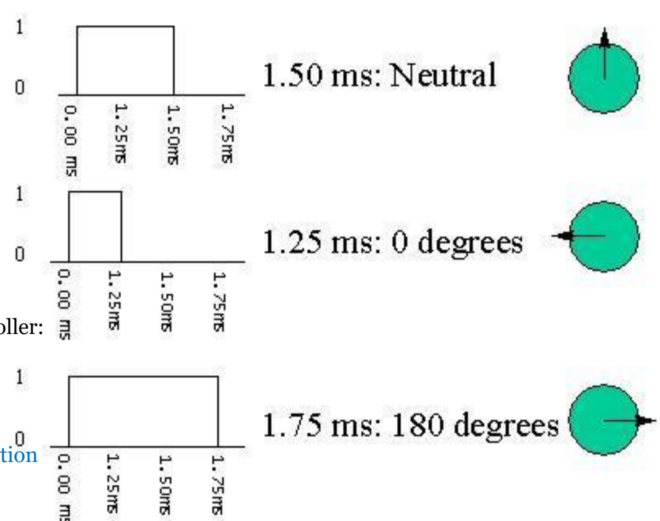


Servo’s

Pulse Width Modulated (PWM) Signal: 50 Hz, i.e. 20ms periods



16-ch, 12-bit, 40 Hz – 1000 Hz



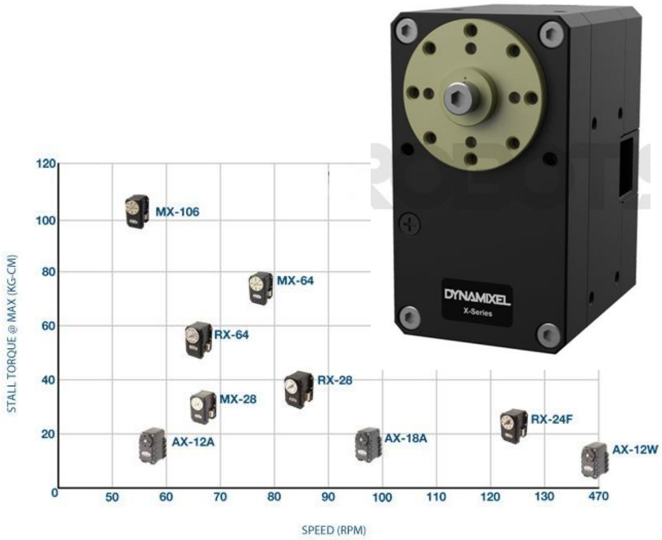
Example usage of maestro.py using Pololu USB Servo Controller:

```
import maestro
servo = maestro.Controller()
servo.setAccel(0,4)      #set servo 0 acceleration to 4
servo.setTarget(0,6000)  #set servo 0 to move to center position
servo.setSpeed(1,10)     #set speed of servo 1
x = servo.getPosition(1) #get the current position of servo 1
servo.close()
```

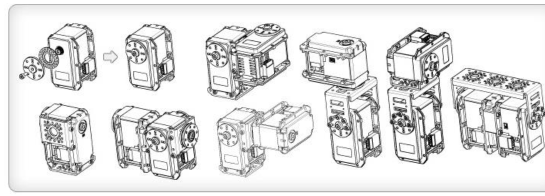
From: https://github.com/FRC4564/Maestro

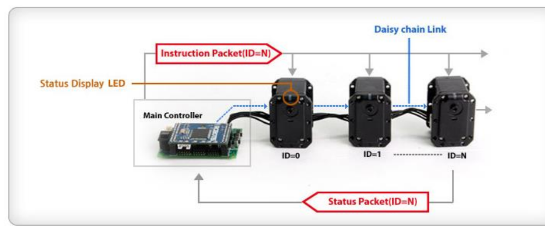
Note: often trimmed Pulse Width

Dynamixel Servo's



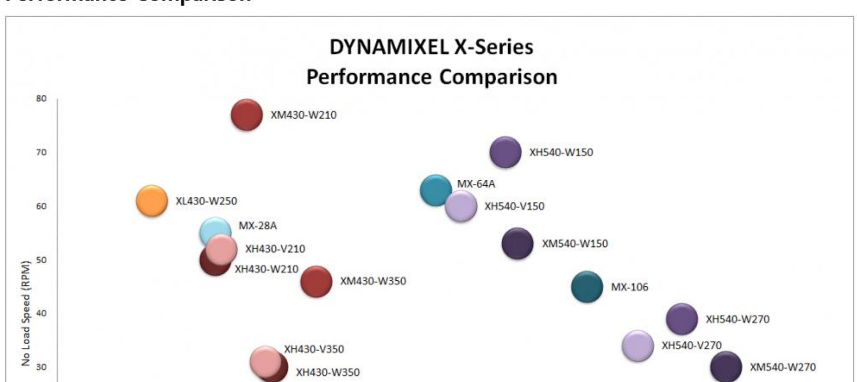
Flexible Construction and Modular Structures





Servo's

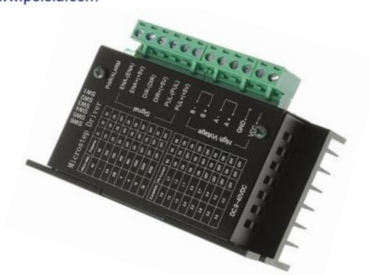
Performance Comparison



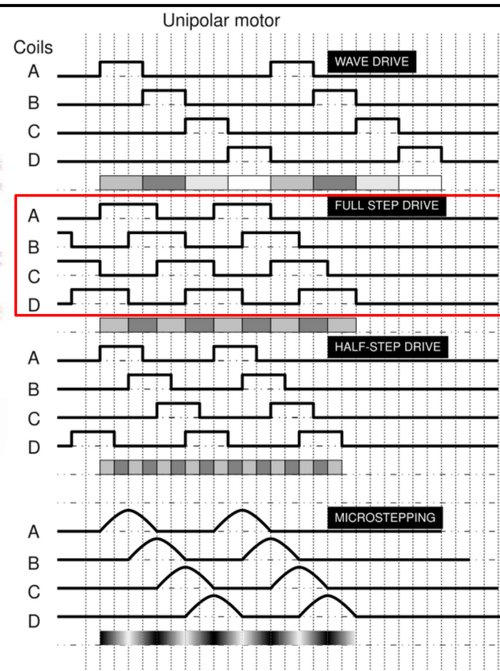
10.6 N.m
30.0 rpm 12.0 V
33.5x58.5x44 (mm³)
165.0 g

9.8N.m ~ 1kgf.m

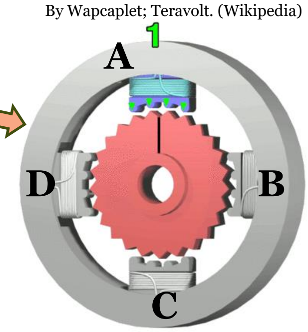
Stepper Motors

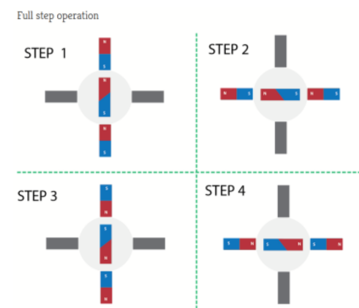


Drivers: low-level, high level

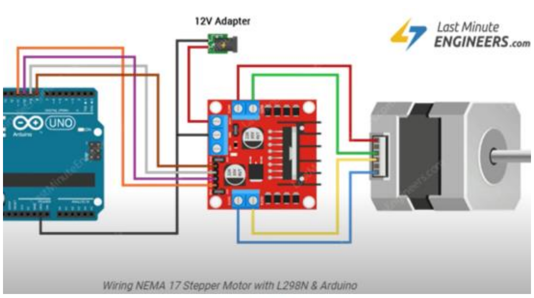


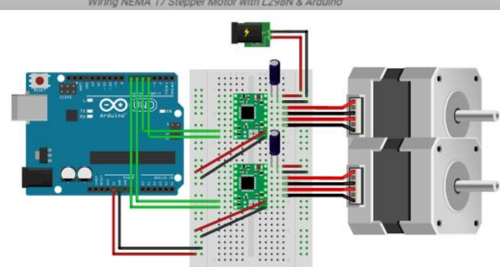
By Wapcaplet; Teravolt. (Wikipedia)



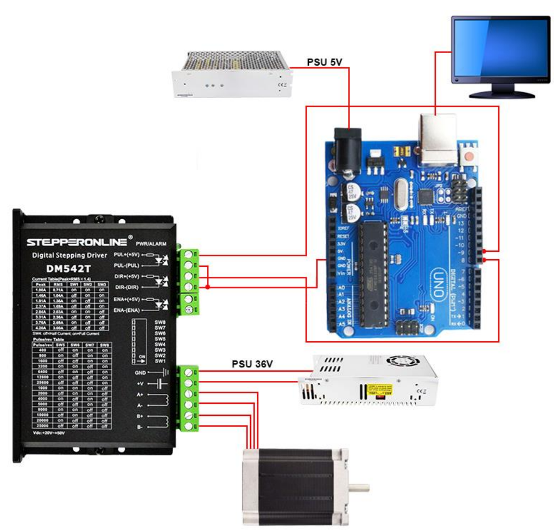


Stepper Motors



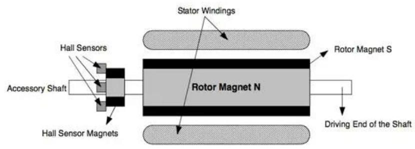


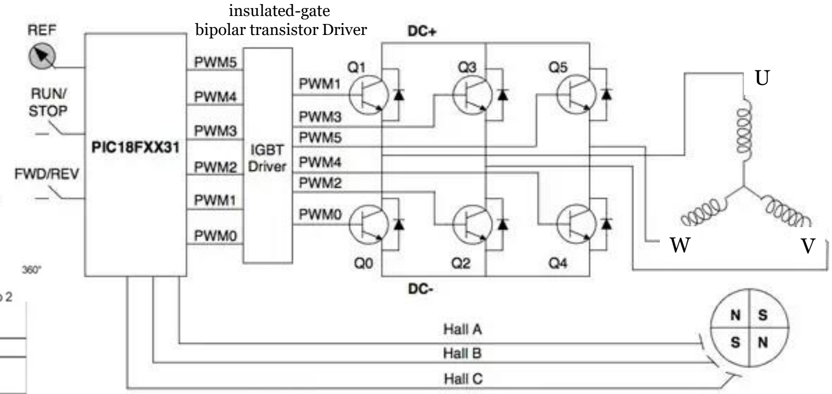
https://forum.arduino.cc

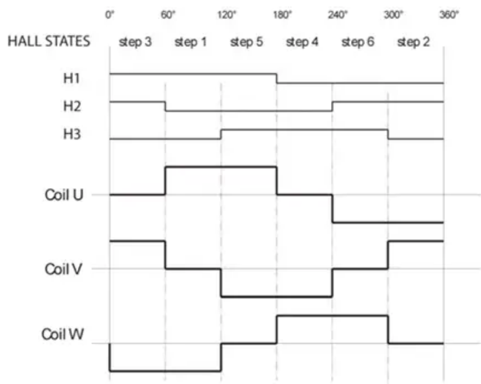


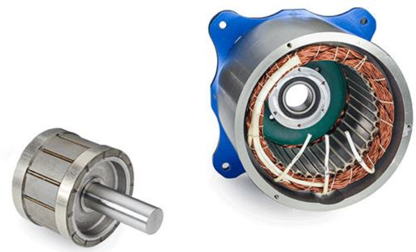
https://www.omc-stepperonline.com/support/can-you-send-me-a-schematic-that-how-to-wire-the-stepper-driver-to-an-arduino

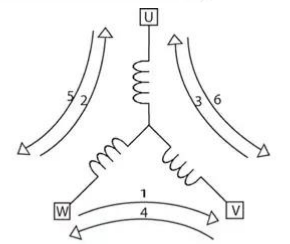
Brushless Motors







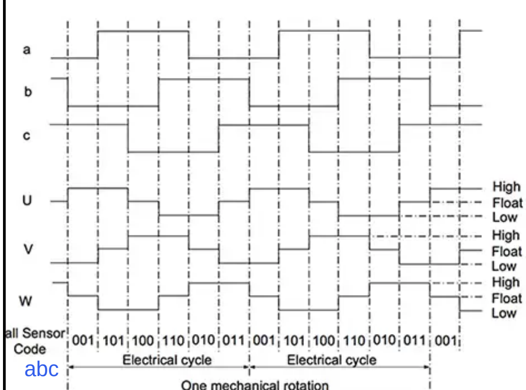


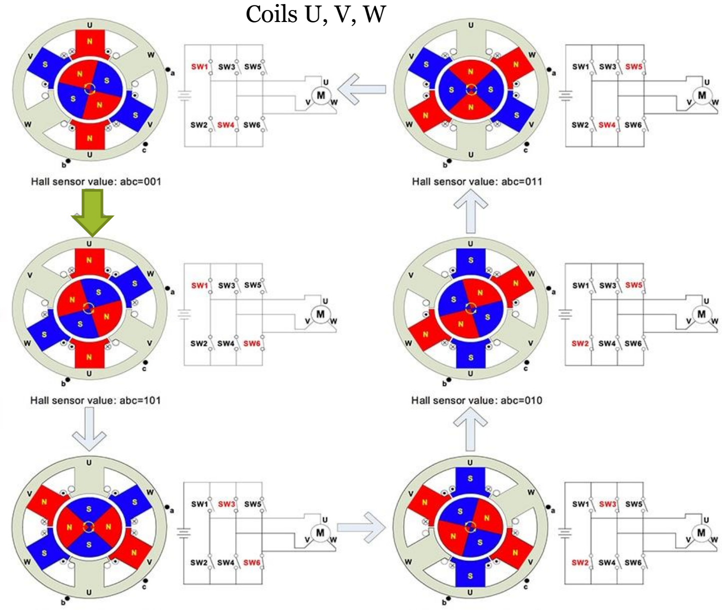


https://www.digikey.com

Brushless Motors

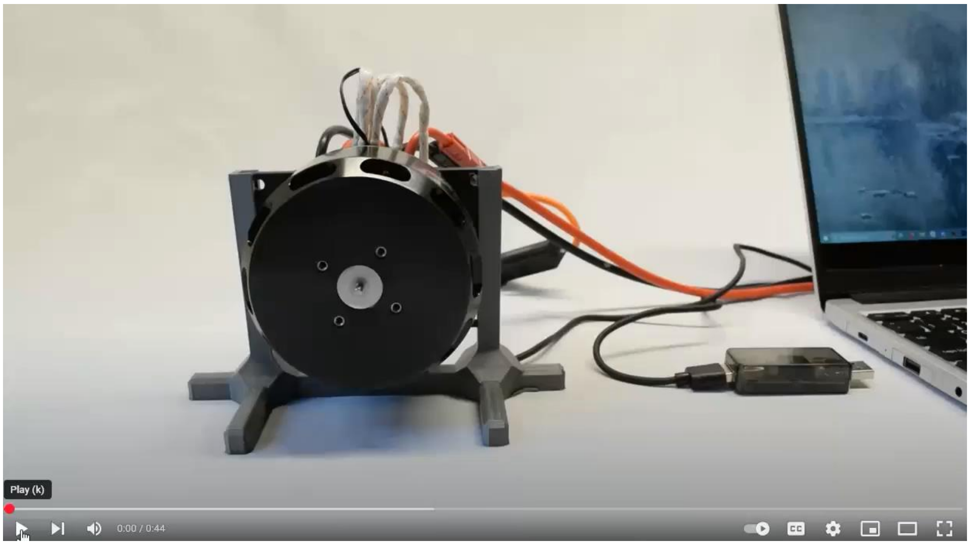
Coils U, V, W





https://www.digikey.com/en/articles/techzone/2016/dec/how-to-power-and-control-brushless-dc-motors

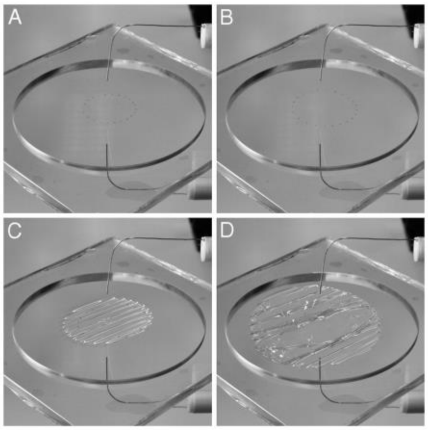
Odrive Brushless Motor Controller



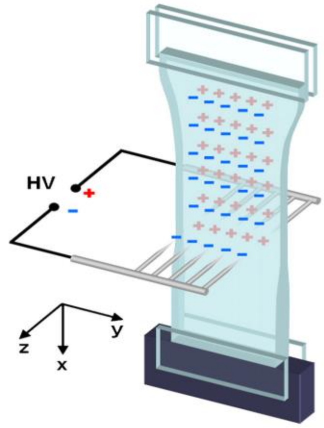
Odrive Installation

- https://github.com/Nate711/StanfordDoggoProject (2019) (EOL)
- https://pupper-v3-documentation.readthedocs.io/en/latest/index.html (Custom motor driver)

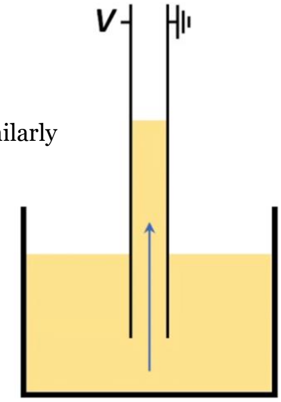
Artificial Muscles



0 – 25 kV



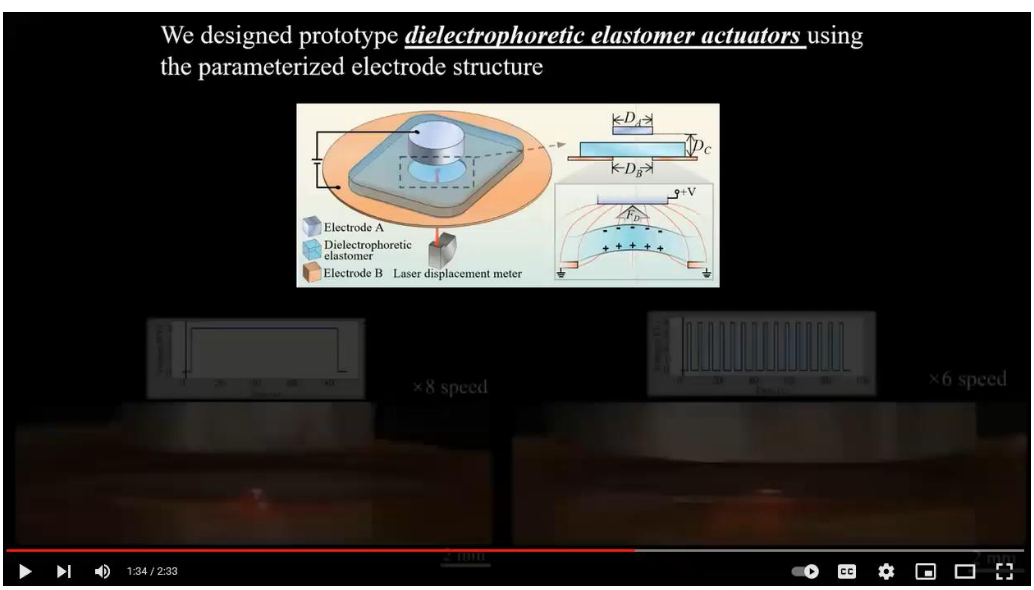
similarly



Maxwell Stress

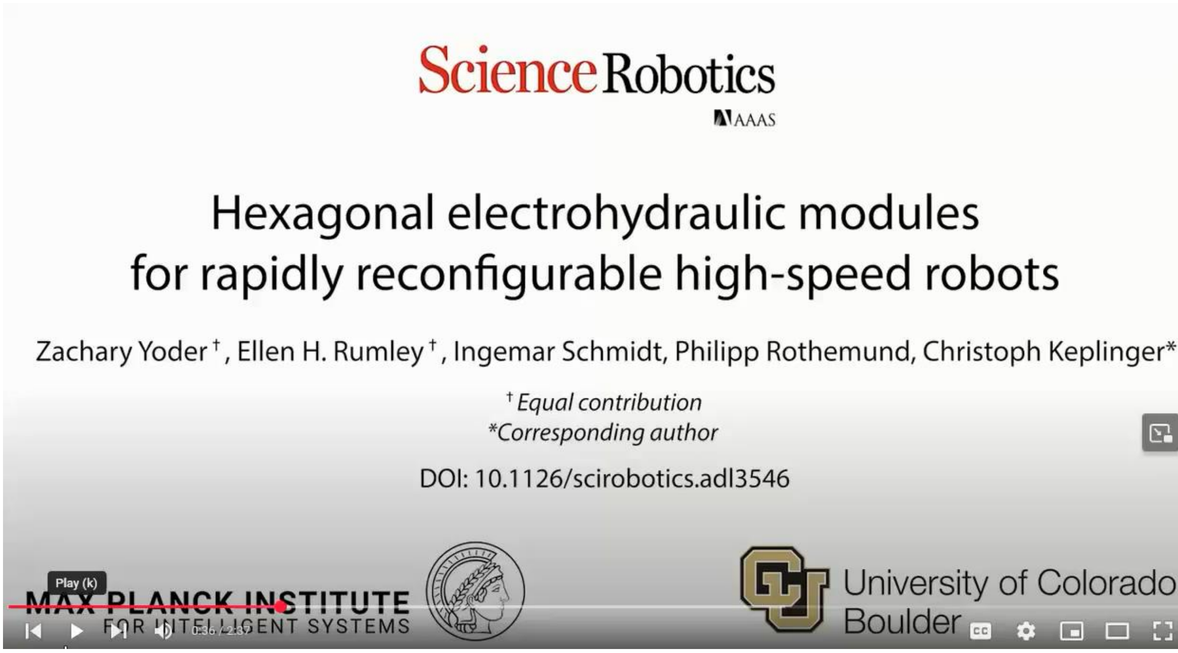
Röntgen's electrode-free elastomer actuators without electromechanical pull-in instability by C. Keplinger, et al. PNAS March 9, 2010 107 (10) 4505-4510; https://doi.org/10.1073/pnas.0913461107

Röntgen WC (1880) Ueber die durch Electricität bewirkten Form—und Volumenänderungen von dielectrischen Körpern. Ann Phys Chem 11:771–786.

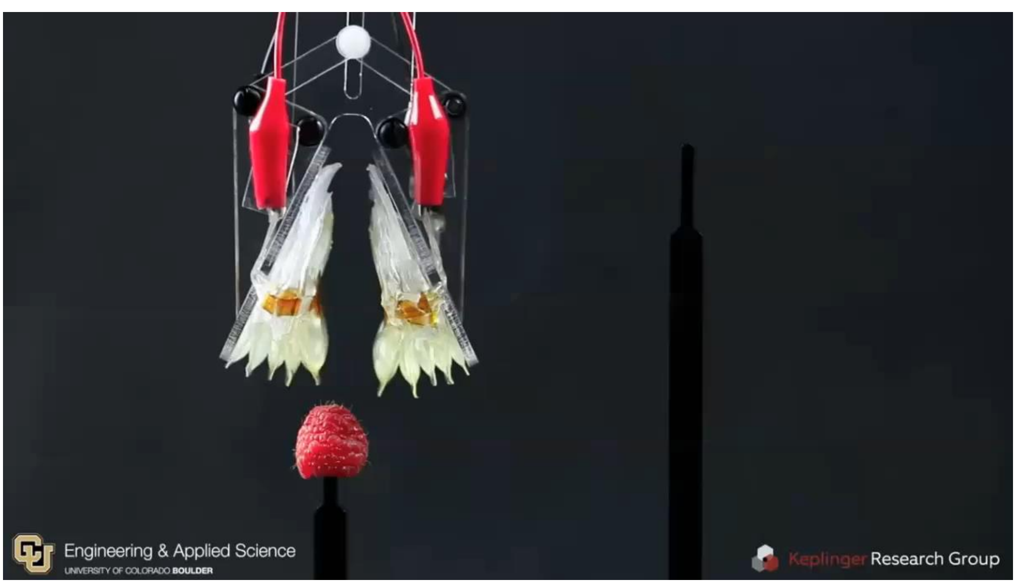


https://www.youtube.com/@softlabbristol4788 (2023)
https://www.youtube.com/watch?v=cTvSycRh-Ik

Universiteit Leiden. Bij ons leer je de wereld kennen

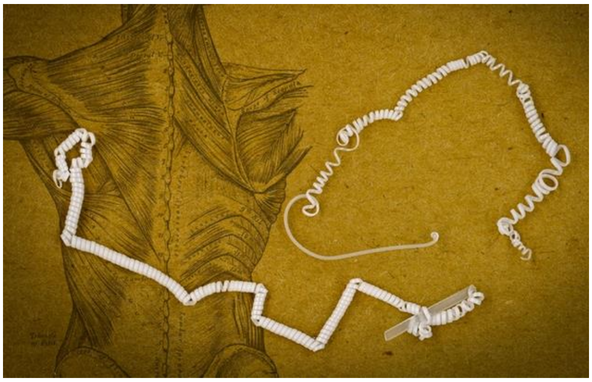


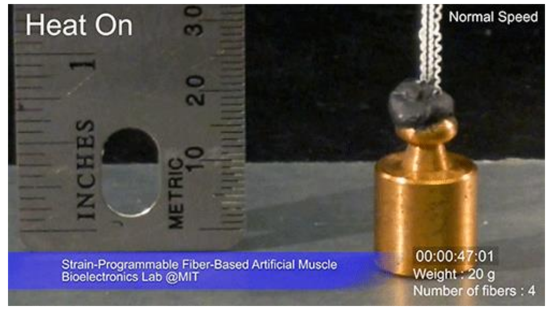
Universiteit Leiden. Bij ons leer je de wereld kennen https://techxplore.com/news/2024-09-combining-soft-artificial-muscles-rigid.html

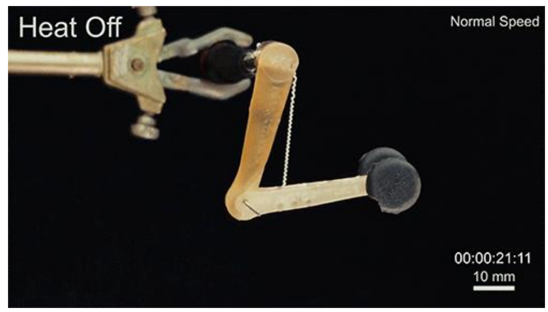


See also TED Talk **The artificial muscles that will power robots of the future by Christoph Keplinger** https://www.youtube.com/watch?v=ER15KmrB8h8

Universiteit Leiden. Bij ons leer je de wereld kennen







MIT Artificial Muscles

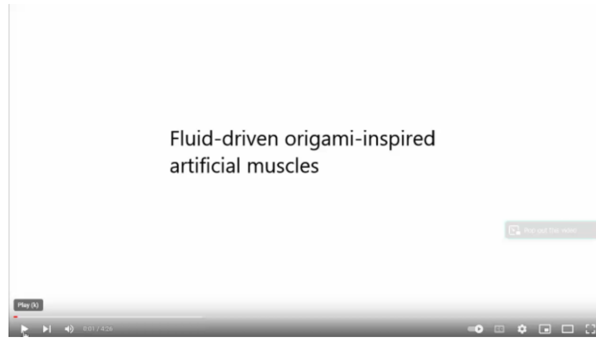
- Combination of two dissimilar polymers into a single fiber
- The polymers have very different thermal expansion coefficients (as in bimetals)
- Developed by Mehmet Kanik, Sirma Örgüç, working with Polina Anikeeva, Yoel Fink, Anantha Chandrakasan, and C. Cem Taşan, and five others

http://news.mit.edu/2019/artificial-fiber-muscles-0711

Spanish Dancer by Micha Heilman and Stella Tsilia

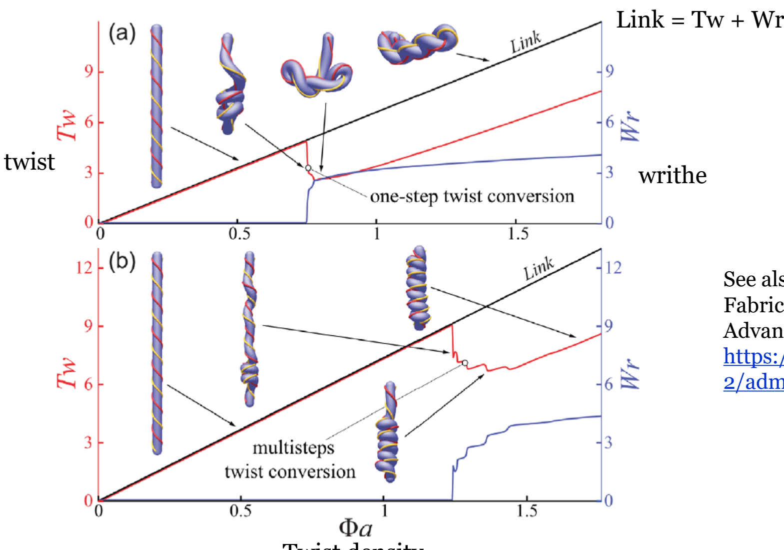
Artificial Muscle
S. Raadscheiders, Marton Menyhert, Yven Lommen, Raffi Mirzoyan

Max. Vacuum:
≤-420 mmHg



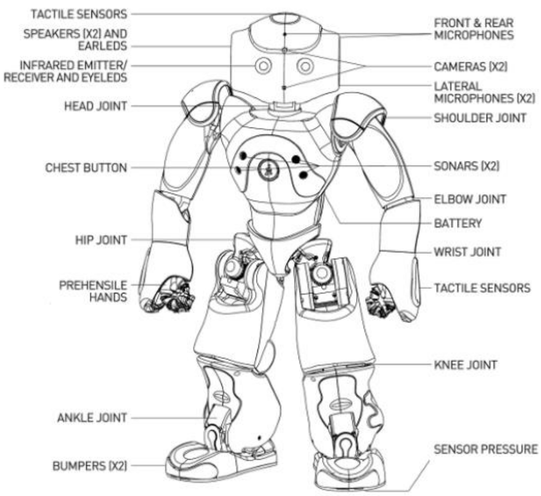
Shuguang Li et al. "Fluid-driven origami-inspired artificial muscles". In: Proceedings of the National Academy of Sciences 114.50 (2017), pp. 13132–13137.
https://www.youtube.com/watch?v=YK6giJglqjE

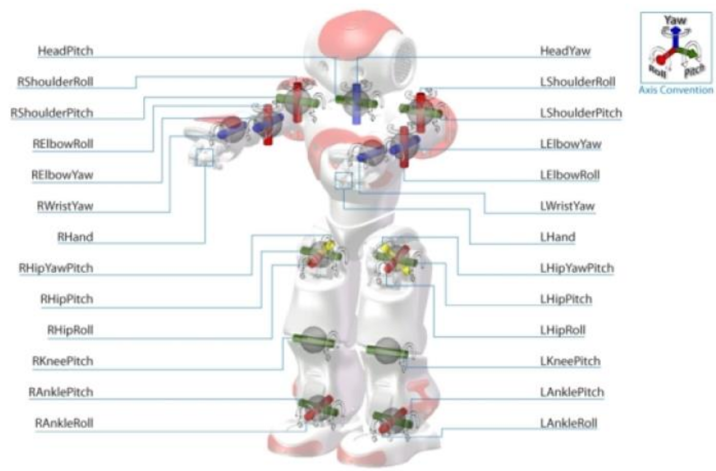
N. Charles, M. Gazzola, and L. Mahadevan, **Topology, Geometry, and Mechanics of Strongly Stretched and Twisted Filaments: Solenoids, Plectonemes, and Artificial Muscle Fibers**
PHYSICAL REVIEW LETTERS 123, 208003 (2019)



See also: J. Xiong et al. Functional Fiber and Fabrics for Soft Robotics, Wearables, and HRI. Advanced Materials, Wiley, May 2021.
https://onlinelibrary.wiley.com/doi/full/10.1002/adma.202002640

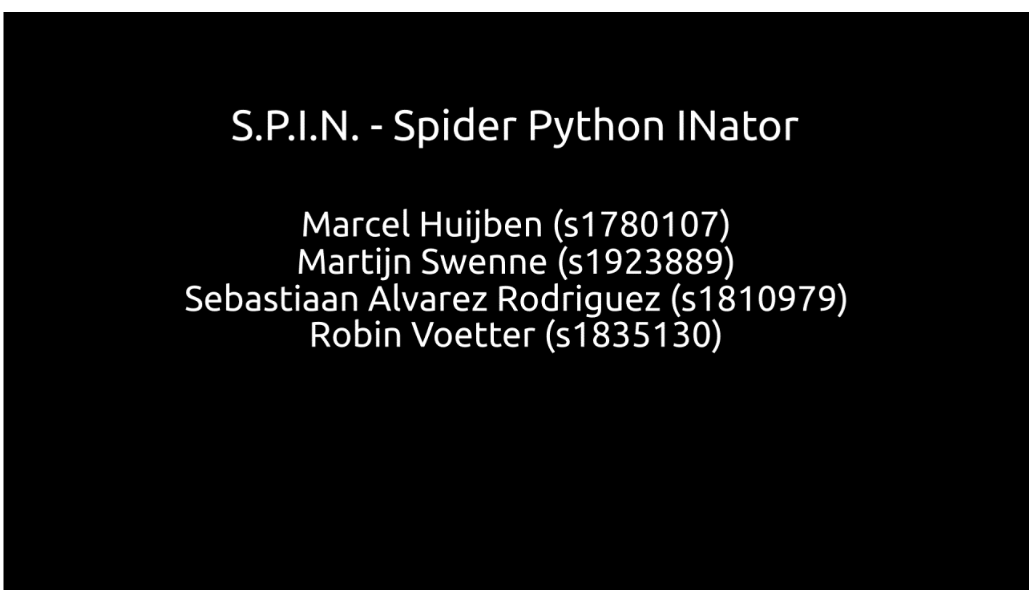
NAO





http://doc.aldebaran.com/2-1/family/nao_dcm/actuator_sensor_names.html

Hexapod: S.P.I.N. by M. Huijben, M. Swenne, R. Voeter, S. Alvarez Rodriguez.



2/18/2025

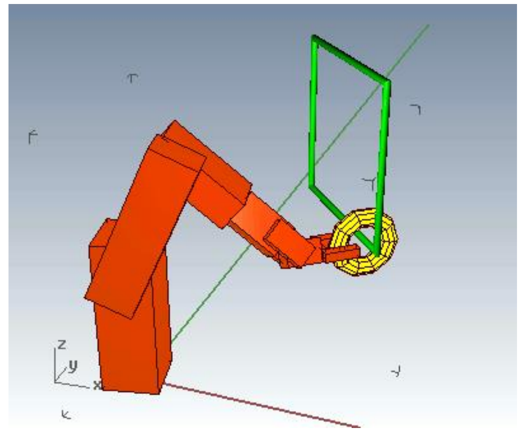
How to move to a goal?

Problem: How to move to a goal?

- Grasp, Walk, Stand, Dance, Follow, etc.

Solution:

1. *Program step by step*
   - *Computer Numerical Control (CNC), Automation.*
2. *Inverse kinematics*
   - take end-points and move them to designated points.
3. *Record and Replay movements*
   - by specialist, human, etc.
4. ***Learn the right movements***
   - **Reinforcement Learning**, give a reward when the movement resembles the designated movement.



https://pybullet.org/wordpress/

Configuration Space

Robot Question: Where am I?

Answer:

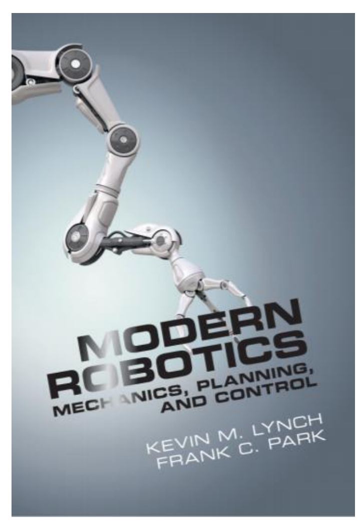
The robot's configuration: a specification of the positions of all points of a robot.

Here we assume:

Robot links and bodies are rigid and of known shape

=>

only a few variables needed to describe it's configuration.

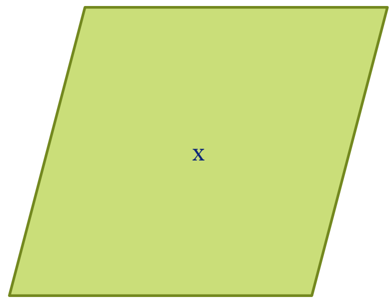


K.M. Lynch, F.C. Park, Modern Robotics: Mechanics, Planning and Control, Cambridge University Press, 2017
http://hades.mech.northwestern.edu/index.php/Modern_Robotics

Universiteit Leiden. Bij ons leer je de wereld kennen

Configuration Space

Door in 3D Space.



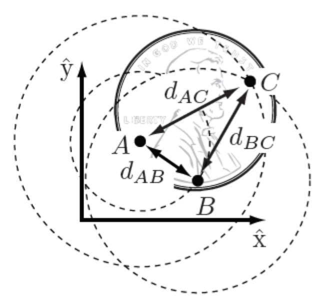
Point in a plane.

Coin on a table.

Degrees of Freedom of a Rigid Body:

the smallest number of real-valued coordinates needed to represent its configuration

Configuration Space



In the plane:
Assume a coin (heads) with 3 points A, B, C on it.

In the plane A,B,C have 6 degrees of freedom: ***(x_A,y_A) , (x_B,y_B) , (x_C,y_C) (6 variables)***
A coin is rigid => 3 extra constraints on distances: ***d_{AB}, d_{AC}, d_{BC}*** ***(3 constraints)***
These are fixed, wherever the location of the coin.

1. The coin and hence **A** can be placed everywhere => (x_A,y_A) free to choose.
2. **B** can only be placed under the constraint that its distance to **A** would be equal to $\mathbf{d_{AB}}$. (1 constraint)
=> freedom to turn the coin around A with angle φ_{AB} => (x_A, y_A, φ_{AB}) are free to choose.
3. **C** should be placed at distance $\mathbf{d_{AC}}$, $\mathbf{d_{BC}}$ from **A** and **B**, respectively (2 constraints)
=> only 1 possibility, hence no degree of freedom added.

Degrees of Freedom (DOF) of a Coin

= sum of freedoms of the points – number of independent constraints
= number of variables – number of independent equations **= 6 – 3 = 3**

Configuration Space

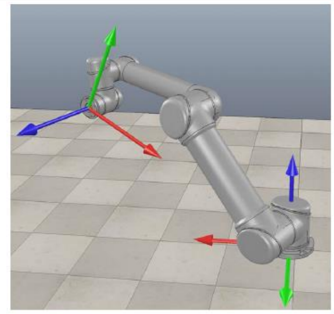
[1] Definition 2.1.

The **configuration** of a robot is a complete specification of the position of every point of the robot.

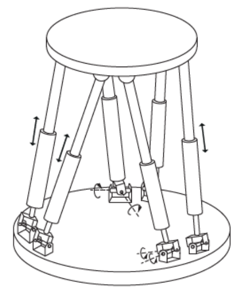
The minimum number ***n*** of real-valued coordinates needed to represent the configuration is the number of **degrees of freedom (dof)** of the robot.

The ***n*****-dimensional** space containing all possible configurations of the robot is called the **Configuration Space (C-space).**

The configuration of a robot is represented by a point in its **C-space**.



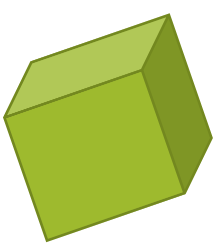
Open-chain robot: Manipulator (in V-REP). [1]



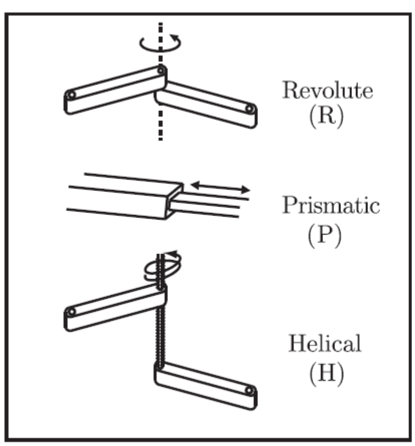
Closed-chain robot: Stewart-Gough platform. [1]

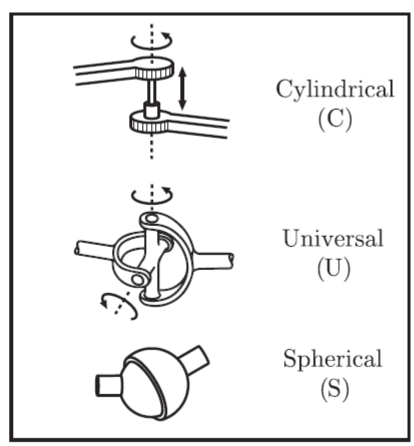
Degrees of Freedom of a Robot

- A rigid body in 3D Space has **6 DOF**



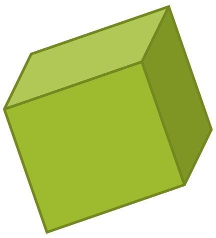
- A joint can be seen to put constraints on the rigid bodies it connects
- It also allows freedom to move relative to the body it is attached to.



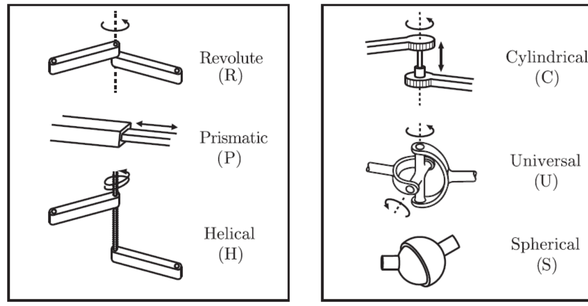


Degrees of Freedom of a Robot

- A **rigid body** in 2D Space has 3 DOF
- A **rigid body** in 3D Space has 6 DOF



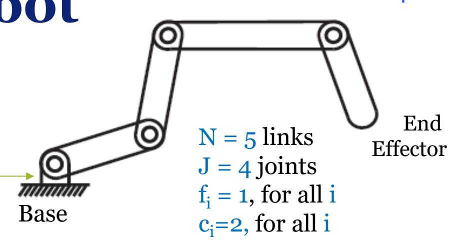
- A **joint** can be seen to put constraints on the rigid bodies it connects
- It also allows freedom to move relative to the body it is attached to.



Joint type	dof f	Constraints c between two planar 2D rigid bodies	Constraints c between two spatial 3D rigid bodies
Revolute (R)	1	2	5
Prismatic (P)	1	2	5
Helical (H)	1	N/A	5
Cylindrical (C)	2	N/A	4
Universal (U)	2	N/A	4
Spherical (S)	3	N/A	3

Degrees of Freedom of a Robot

Planar Mechanism DOF = 4



Proposition (Grübler's formula)

Consider a mechanism consisting of

- N links, where ground (!) is also regarded as a link
- J number of joints
- m number of degrees of freedom of a rigid body
  (m = 3 for planar (2D) mechanisms and m = 6 for spatial (3D) mechanisms)
- f_i the number of freedoms provided by joint i
- c_i the number of constraints provided by joint i, where $f_i + c_i = m$ for all i.

Then *Grübler's formula* for the number of degrees of freedom of the robot is

$$dof = m(N-1) - \sum_{i=1}^{J} c_i = m(N-1-J) + \sum_{i=1}^{J} f_i$$

This formula holds only if all joint constraints are **independent**.

If they are not independent then the formula provides a lower bound on the number of degrees of freedom.

Universiteit Leiden. Bij ons leer je de wereld kennen

Joint reactions in rigid body mechanisms with dependent constraints

Marek Wojtyra *

Elsevier, Mechanism and Machine Theory, Vol. 44, 2009

Warsaw University of Technology, Institute of Aeronautics and Applied Mechanics, ul. Nowowiejska 24, 00-665 Warsaw, Poland

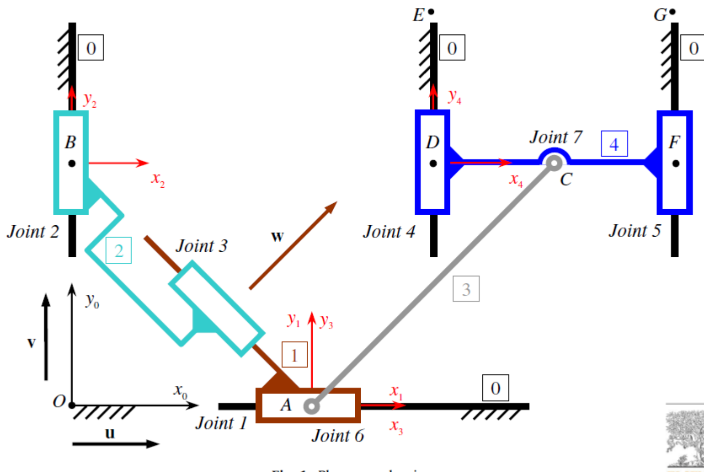


Fig. 1. Planar mechanism.

$$dof = m(N-1) - \sum_{i=1}^{J} c_i = m(N-1-J) + \sum_{i=1}^{J} f_i$$

See also: A. Mueller, Dynamics of parallel manipulators with hybrid complex limbs – Modular modeling and parallel computing, [Mechanism and Machine Theory, Vol. 167, Jan. 2022]().

Links
1
3
3
6
3
1

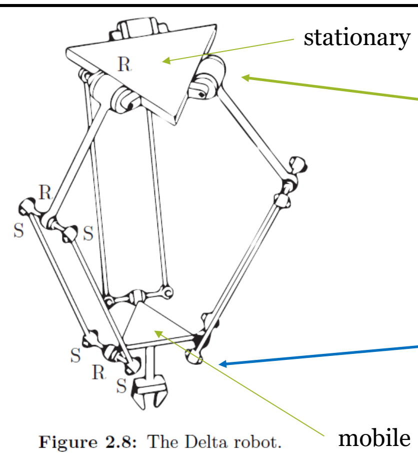
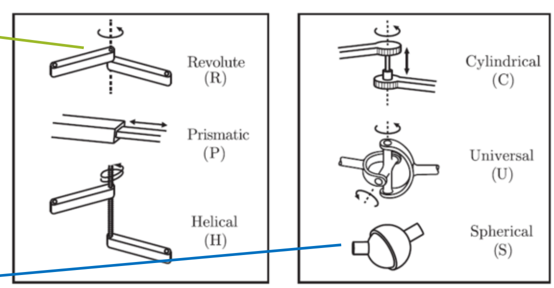


Figure 2.8: The Delta robot.



Example 2.7 (Delta robot). The Delta robot of Figure 2.8 consists of two platforms – the lower one mobile, the upper one stationary – connected by three legs. Each leg contains a parallelogram closed chain and consists of three revolute joints, four spherical joints, and five links. Adding the two platforms, there are $N = 17$ links and $J = 21$ joints (nine revolute and 12 spherical). By Grübler's formula,

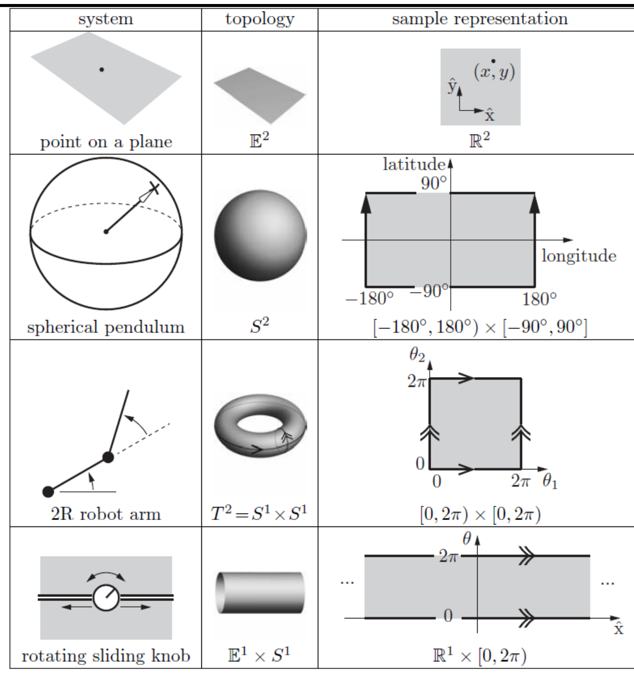
$$\text{dof} = 6(17-1-21) + 9(1) + 12(3) = 15.$$

- Links: 1 + 3 + 3 + 6 + 3 + 1 = 17
- Joints: 21: 9x R(1 dof) and 12 x S(3 dof)
- m= 6

$$dof = m(N-1) - \sum_{i=1}^{J} c_i = m(N-1-J) + \sum_{i=1}^{J} f_i$$

Systems
and their
Topologies

Note: $S^1 \times S^1 = T^2$ (not S^2)



system	topology	sample representation
point on a plane	$\mathbb{E}^2$	$\mathbb{R}^2$
spherical pendulum	S^2	$[-180°, 180°) \times [-90°, 90°]$
2R robot arm	$T^2 = S^1 \times S^1$	$[0, 2\pi) \times [0, 2\pi)$
rotating sliding knob	$\mathbb{E}^1 \times S^1$	$\mathbb{R}^1 \times [0, 2\pi)$

Coordinates can be:

Explicit Coordinates
- Euclidean (x,y)
- Polar (r,φ)
- Combined (x,y) x (r, φ)

Implicit Coordinates
- $\{ (x,y,z) \mid x^2+y^2+z^2=1 \}$

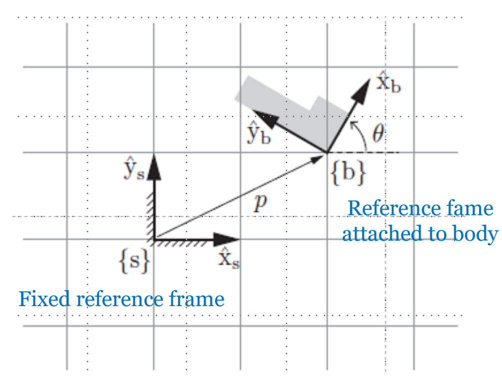
C-Space (Configuration Space)

How to describe a rigid body's position and orientation in C-Space?

Fixed reference frame {s}
Reference fame attached to body {b}

In $\mathbb{R}^3$ described by a 4x4 matrix with 10 constraints
(constraints, e.g.: unit-length, orthogonal)
Note: a point in $\mathbb{R}^3 x S^2 x S^1$

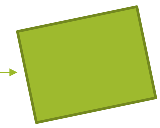
Matrix can be used to:
1. Translate or rotate a vector or a frame
2. Change the representation of a vector or a frame
   - for example from relative to {s} to relative to {b}

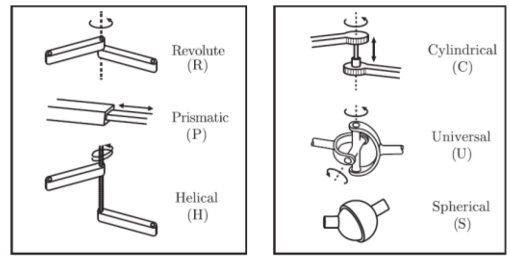


in the plane $\mathbb{R}^2 x S^1$

C-Spaces

C-space of a rigid body in the plane = $\mathbb{R}^2 x S^1$ as configuration can be denoted as (x,y,θ), i.e., location (x,y) in $\mathbb{R}^2$ and angle θ in S^1





C-space of a Prismatic-Revolute (PR) robot arm is equal to $\mathbb{R}^1 x S^1$

C-space of a 2R robot arm is $S^1 x S^1 = T^2$

C-space of a 3R robot arm is $S^1 x S^1 x S^1 = T^3$

C-space of a planar mobile robot with a 2R robot arm is $\mathbb{R}^2 x S^1 x T^2 = \mathbb{R}^2 x T^3$

C-space of a rigid body in space is $\mathbb{R}^3 x S^2 x S^1$

Task Space and Work Space

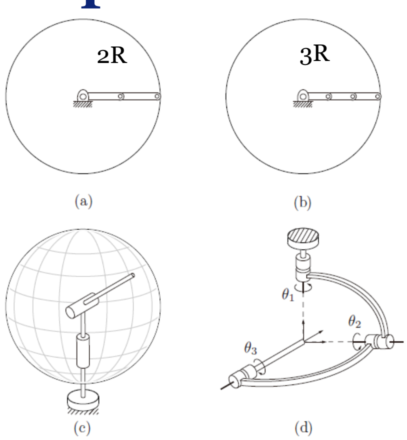
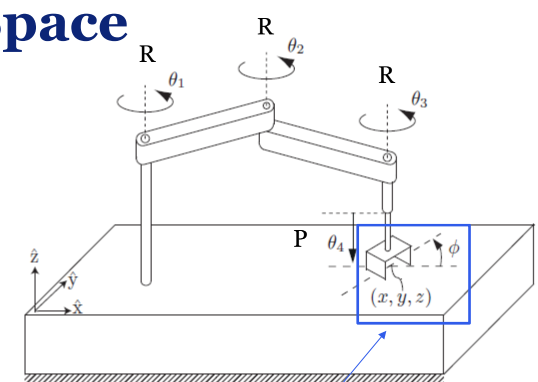


Figure 2.12: Examples of workspaces for various robots: (a) a planar 2R open chain; (b) a planar 3R open chain; (c) a spherical 2R open chain; (d) a 3R orienting mechanism.

The **workspace** is a specification of the configurations that the end-effector of the robot can reach.

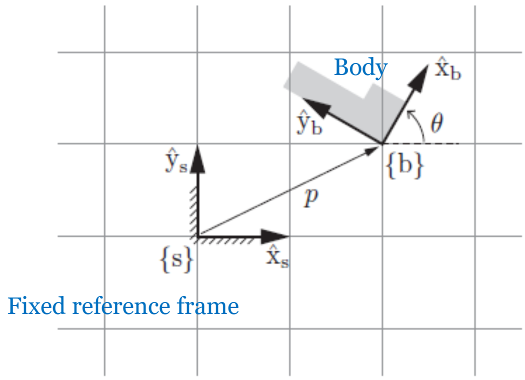


The SCARA robot is an RRRP open chain that is widely used for tabletop pick-and-place tasks. The end-effector configuration is completely described by (x, y, z, φ)

$\Rightarrow$ **task space** $R^3 \, x \, S^1$ and

$\Rightarrow$ **workspace** as the reachable points in (x, y, z), since all orientations φ can be achieved at all reachable points.

Rigid Body Motions in the Plane



Translation

$p = p_x \hat{x}_s + p_y \hat{y}_s.$

Rotation

$$\hat{x}_b = \cos\theta\, \hat{x}_s + \sin\theta\, \hat{y}_s,$$
$$\hat{y}_b = -\sin\theta\, \hat{x}_s + \cos\theta\, \hat{y}_s.$$

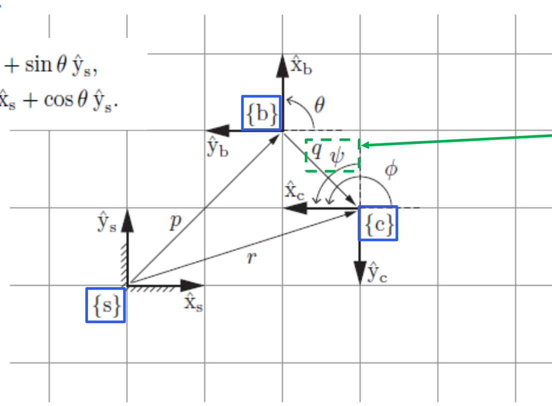
Figure 3.3: The body frame {b} is expressed in the fixed-frame coordinates {s} by the vector p and the directions of the unit axes $\hat{x}_b$ and $\hat{y}_b$. In this example, $p = (2, 1)$ and $\theta = 60°$, so $\hat{x}_b = (\cos\theta, \sin\theta) = (0.5, 1/\sqrt{2})$ and $\hat{y}_b = (-\sin\theta, \cos\theta) = (-1/\sqrt{2}, 0.5)$.

Rigid Body Motions in the Plane

Previously:

$p = p_x \hat{x}_s + p_y \hat{y}_s.$

$$\hat{x}_b = \cos\theta\, \hat{x}_s + \sin\theta\, \hat{y}_s,$$
$$\hat{y}_b = -\sin\theta\, \hat{x}_s + \cos\theta\, \hat{y}_s.$$



{b} relative to {s}

$$p = \begin{bmatrix} p_x \\ p_y \end{bmatrix}$$

$$P = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

{c} relative to {b}

$$q = \begin{bmatrix} q_x \\ q_y \end{bmatrix}, \quad Q = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix}$$

{c} relative to {s}

$$r = \begin{bmatrix} r_x \\ r_y \end{bmatrix}, \quad R = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$$

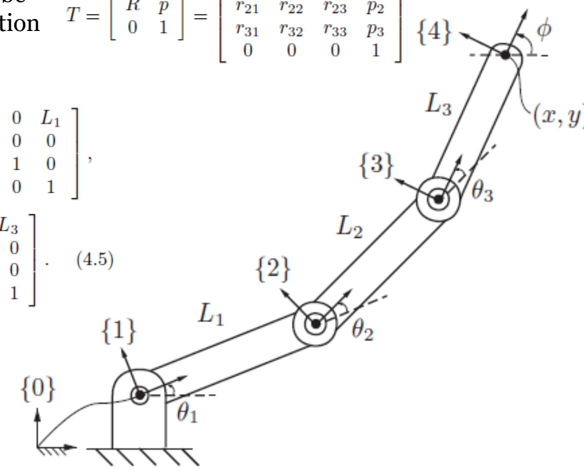
Figure 3.4: The frame {b} in {s} is given by (P, p), and the frame {c} in {b} is given by (Q, q). From these we can derive the frame {c} in {s}, described by (R, r). The numerical values of the vectors p, q, and r and the coordinate-axis directions of the three frames are evident from the grid of unit squares.

Note and verify:
R = PQ, convert Q to {s}-frame
r = Pq+p, convert q to {s}-frame and add p

Forward Kinematics

The forward kinematics of 3R Planar Open Chain can be written as a product of four homogeneous transformation matrices: $T_{04} = T_{01}T_{12}T_{23}T_{34}$, where

$$T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_1 \\ r_{21} & r_{22} & r_{23} & p_2 \\ r_{31} & r_{32} & r_{33} & p_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$T_{01} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad T_{12} = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & L_1 \\ \sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

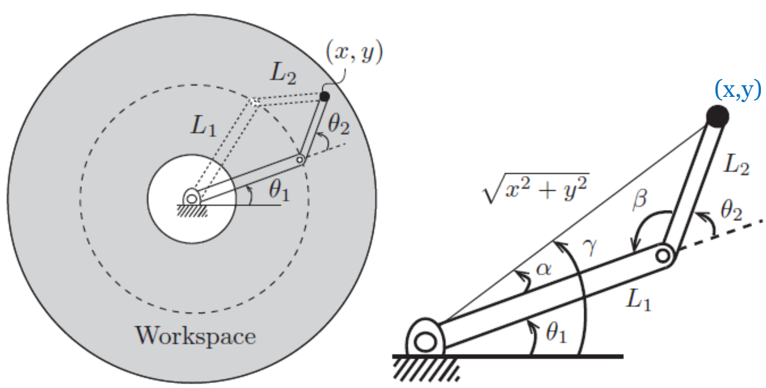
$$T_{23} = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & L_2 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad T_{34} = \begin{bmatrix} 1 & 0 & 0 & L_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (4.5)$$

Home position M:

$$M = \begin{bmatrix} 1 & 0 & 0 & L_1+L_2+L_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Inverse Kinematics

Which angles θ_1, and θ_2 will lead to location (x,y)?



(a) A workspace, and lefty and righty configurations.

(b) Geometric solution.

Figure 6.1: Inverse kinematics of a 2R planar open chain.

Law of cosines gives:

$$L_1^2 + L_2^2 - 2L_1L_2\cos\beta = x^2 + y^2$$

, hence

$$\beta = \cos^{-1}\left(\frac{L_1^2 + L_2^2 - x^2 - y^2}{2L_1L_2}\right)$$

,and similarly

$$\alpha = \cos^{-1}\left(\frac{x^2 + y^2 + L_1^2 - L_2^2}{2L_1\sqrt{x^2+y^2}}\right)$$

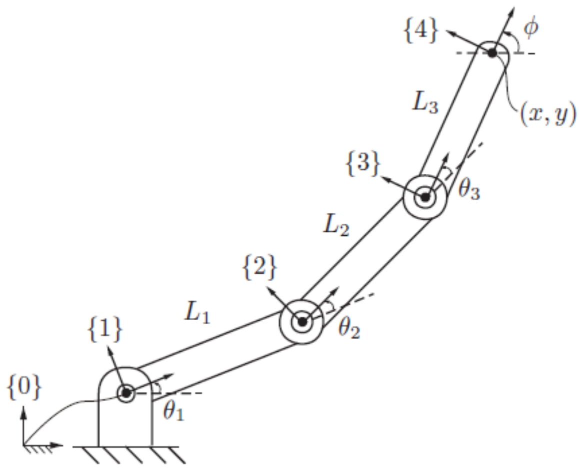
γ= atan2(y,x)

Answer:

$$\theta_1 = \gamma - \alpha, \qquad \theta_2 = \pi - \beta$$

In general: IK-Solvers, Newton-Raphson, etc.

Inverse Kinematics



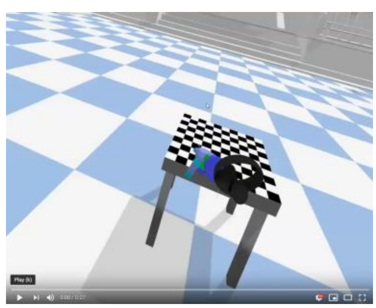
How would you solve this?

Which angles θ_1, θ_2, and θ_3 will lead to location (x,y)?

Real Time Physics Modelling

https://pybullet.org/wordpress/

CoppeliaSim
(SLAM Workshop@Home)
https://www.coppeliarobotics.com/



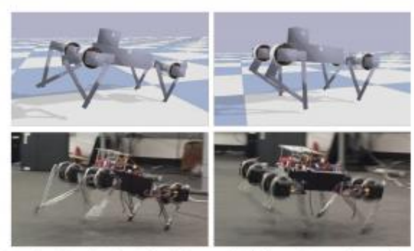
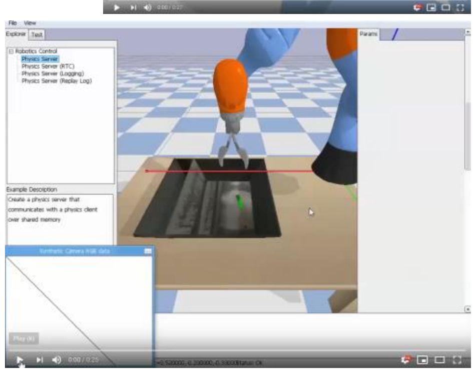
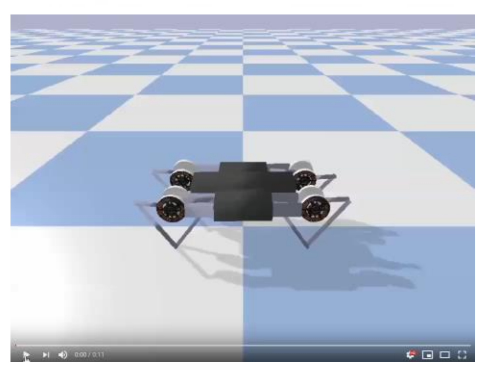


Fig. 1: The simulated and the real Minitaurs learned to gallop using deep reinforcement learning.

pybullet KUKA
grasp training



Using Tensorflow
OpenAI gym
Baselines
DeepQNetworks (DQNs)



Organization and Overview

Lecturer:
Dr Erwin M. Bakker (erwin@liacs.nl)
Room LIACS Media Lab (LML)
Please email for a meeting.

Teaching assistants:
TBA

Period: February 11th - May 13th 2025
Time: Tuesday 11.15 - 13.00
Place (Rooms): Van Steenis F1.04
Exceptions:
Gorlaeus Building BM.1.33 on April 1st
Gorlaeus Building BM.1.23 on May 20th

Schedule (tentative, visit regularly):

Date	Subject
11-2	Introduction and Overview
18-2	Locomotion and Inverse Kinematics
25-2	Robotics Sensors and Image Processing
4-3	SLAM + Workshop@Home
11-3	Robotics Vision + Introduction Mobile Robot Challenge
18-3	Project Proposals I (by students)
25-3	Project Proposals II (by students)
1-4	Robotics Reinforcement Learning + RL Workshop@Home
8-4	Project Progress Reports I
15-4	Project Progress Reports II
22-4	Mobile Robot Challenge I
29-4	Mobile Robot Challenge II
6-5	TBA
13-5	Project Demos I
20-5	Project Demos II
27-5	Project Deliverables

Website: http://liacs.leidenuniv.nl/~bakkerem2/robotics/

Grading (6 ECTS):
- Presentations and Robotics Project (60% of grade).
- Class discussions, attendance, 2 assignments (pass/no pass)
- 2 Workshops (0-10) (20% of the grade).
- Mobile Robot Challenge (0-10) (20% of the grade)
- ***It is necessary to be at every class and to complete every workshop and assignment.***

Universiteit Leiden. Bij ons leer je de wereld kennen

Robotics Homework II

Visit http://modernrobotics.org and obtain the pdf of the book.

Read Chapters 1 and 2.

Note: also the videos are highly recommended!

The exercises of Homework II (due Friday February 28th)
will be available on BrightSpace (Thursday February 20th).

Universiteit Leiden. Bij ons leer je de wereld kennen

References

1. K.M. Lynch, F.C. Park, Modern Robotics: Mechanics, Planning and Control, Cambridge University Press, 2017. (DOI: 10.1017/9781316661239)
2. https://pybullet.org/wordpress/
3. https://www.coppeliarobotics.com/