

Outline

- Waitlist - Bad News - Good News
- Handout of robots? - Monday - Wednesday
- Homework 1 - assignment on Canvas incl link to code
- Mobile Platforms
- AOB

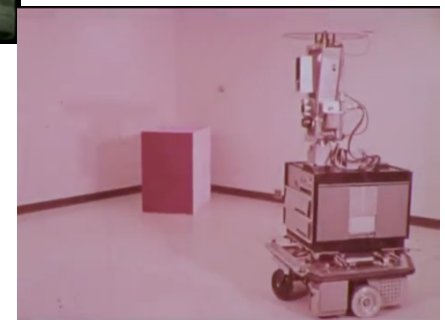
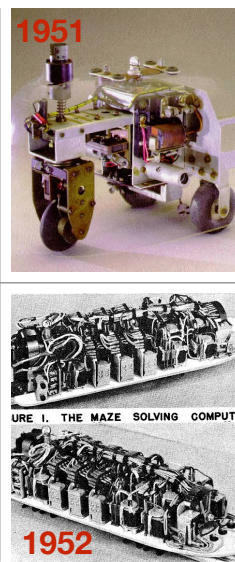
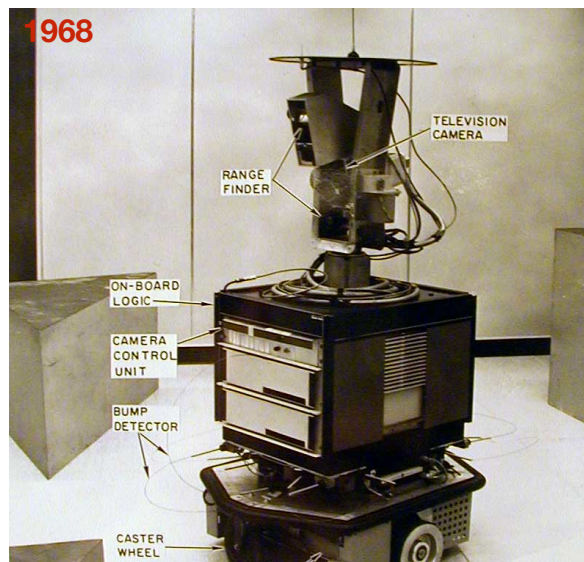
Industrial Use Case



Mobile Systems

Henrik I Christensen

(c) Henrik I Christensen



More Modern AGVs



A widely advertised example - KIVA Systems (now Amazon)

(c) Henrik I Christensen

KIVA example

Other Modalities



Mobility: Train

- Configuration: 1D
- Task Space: R
- $C = T$



Mobility: Hovercraft

- Configuration: (x, y, θ)
- Actuators: 2DOF
- Task Space: SE(2)
- $C = T$



Mobility: Helicopter

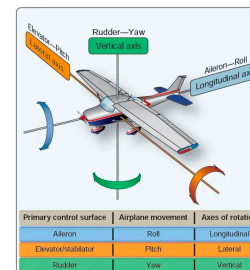
- Configuration: $(x, y, z, \theta_r, \theta_p, \theta_y)$
- Actuators: 4DOF (thrust, pitch, roll, tail)
- Task Space: SE(3)
- $C = T$



Boeing A160 Hummingbird

Mobility: Fixed WING

- Configuration: $(x, y, z, \theta_r, \theta_p, \theta_y)$
- Actuators: 4DOF (thrust, ail, elev, rud)
- Task Space: SE(3)
- $C = T$



QF-16

Mobility: Submersible

- Configuration: 6D
- Fully Actuated
- Task Space: SE(3)
- $C = T$



Wheels

- Regular wheel
 - Non-holonomic constraint
 - $\dot{x} = v$, $\dot{y} = 0$
- Omnidirectional wheel
 - No such constraint



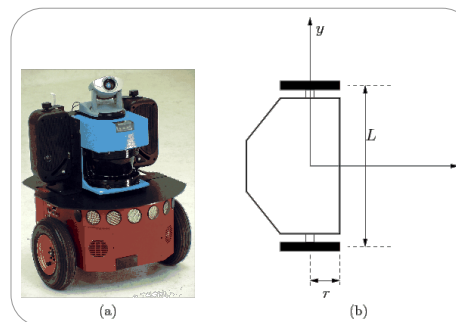
Mobility Recap

Table 4.1.
Summary of parameters for three different types of vehicle. The +g notation indicates that the gravity field can be considered as an extra actuator

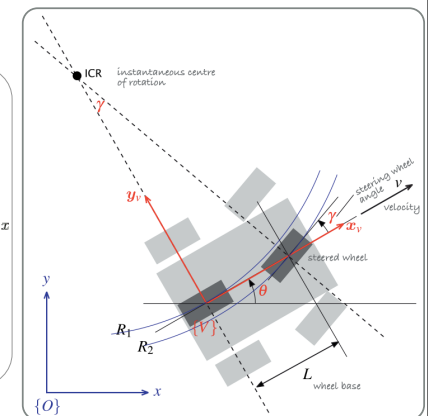
Vehicle	Degrees of freedom	Number of actuators	Fully actuated?
Train	1	1	✓
Hovercraft	3	2	×
Helicopter	6	4+g	×
Fixed wing aircraft	6	4+g	×
6-thruster AUV	6	6	✓
Car	3	2	×

2 Kinematic Models

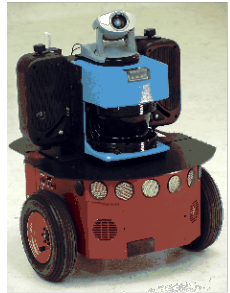
Differential Drive



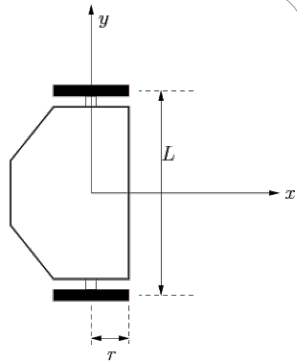
Bicycle Model



Differential drive



(a)



(b)

$$\dot{\xi}_R = \begin{bmatrix} \frac{r}{2} (\dot{\phi}_R + \dot{\phi}_L) \\ 0 \\ \frac{r}{L} (\dot{\phi}_R - \dot{\phi}_L) \end{bmatrix}$$

(this is in the body frame)

Forward only:

$$\dot{\phi}_R = \frac{v_x}{r}$$

$$\dot{\phi}_L = \frac{v_x}{r}$$

Rotation only:

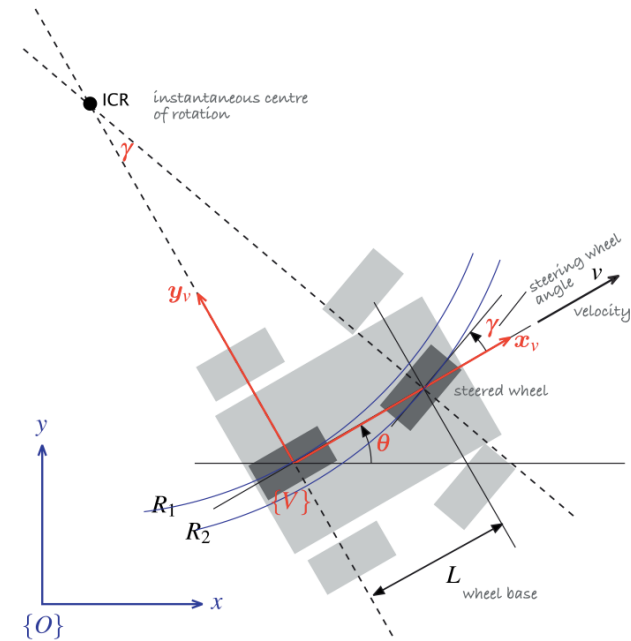
$$\dot{\phi}_R = \frac{\omega L}{2r}$$

$$\dot{\phi}_L = -\frac{\omega L}{2r}$$

Both:

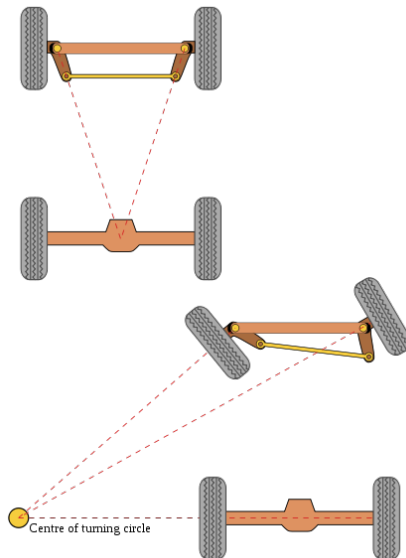
$$\dot{\phi}_R = \frac{\omega L}{2r} + \frac{v_x}{r}$$

$$\dot{\phi}_L = -\frac{\omega L}{2r} + \frac{v_x}{r}$$



Ackerman Steering

- Four-wheeled vehicle
- L and R move on circular paths of different radius
- 1812 patent



Equations of Motion (See book)

- Angular velocity
~ steering angle γ
- Translation velocity
~ velocity v
- Non-holonomic constraint
- Undefined for 90° angle

$$\dot{x} = v \cos \theta$$

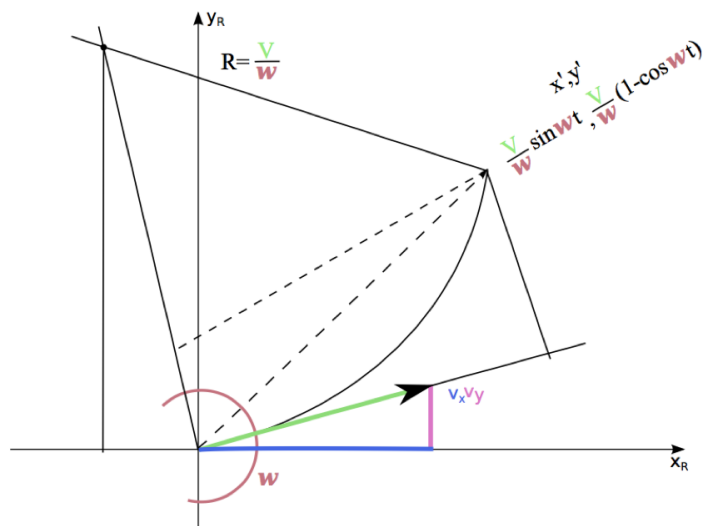
$$\dot{y} = v \sin \theta$$

$$\dot{\theta} = \frac{v}{L} \tan \gamma$$

$$\dot{y} \cos \theta - \dot{x} \sin \theta \equiv 0$$

(this is in the world frame)

2D Twist



2D Twist

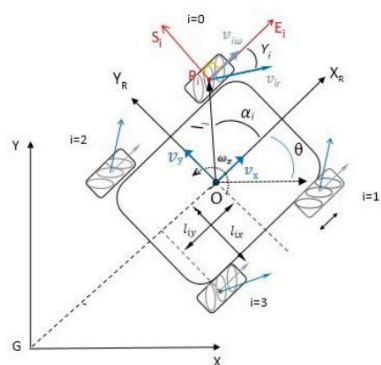
A 2D twist is simply the derivative of a 2D rigid transformation

$$\xi_R^W = \begin{bmatrix} \dot{t}_x \\ \dot{t}_y \\ \dot{\theta} \end{bmatrix}$$

A robot undergoing a constant twist, expressed in the robot frame, traces out a circular trajectory with radius $R = v/\omega$. Starting from the origin, after some time T we obtain

$$\xi(T) = \left(\begin{bmatrix} \cos \omega T & -\sin \omega T \\ \sin \omega T & \cos \omega T \end{bmatrix}, \frac{1}{\omega} \begin{bmatrix} 1 - \cos \omega T & \sin \omega T \\ -\sin \omega T & 1 - \cos \omega T \end{bmatrix} \begin{bmatrix} -v_y \\ v_x \end{bmatrix} \right)$$

Mecanum wheels



Kinematic Model of a Four Mecanum Wheeled Mobile Robot

Hamid Taheri
College of Electronic and

Bing Qiao
College of Astronautics

Nurallah Ghaeminezhad
College of Automation

The general kinematic model

$$\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} = \frac{-1}{r} \begin{bmatrix} \frac{\cos(\beta_1 - \gamma_1)}{\sin \gamma_1} & \frac{\sin(\beta_1 - \gamma_1)}{\sin \gamma_1} & \frac{l_1 \sin(\beta_1 - \gamma_1 - \alpha_1)}{\sin \gamma_1} \\ \frac{\cos(\beta_2 - \gamma_2)}{\sin \gamma_2} & \frac{\sin(\beta_2 - \gamma_2)}{\sin \gamma_2} & \frac{l_2 \sin(\beta_2 - \gamma_2 - \alpha_2)}{\sin \gamma_2} \\ \frac{\cos(\beta_3 - \gamma_3)}{\sin \gamma_3} & \frac{\sin(\beta_3 - \gamma_3)}{\sin \gamma_3} & \frac{l_3 \sin(\beta_3 - \gamma_3 - \alpha_3)}{\sin \gamma_3} \\ \frac{\cos(\beta_4 - \gamma_4)}{\sin \gamma_4} & \frac{\sin(\beta_4 - \gamma_4)}{\sin \gamma_4} & \frac{l_4 \sin(\beta_4 - \gamma_4 - \alpha_4)}{\sin \gamma_4} \end{bmatrix} \begin{bmatrix} v_X \\ v_Y \\ \omega_Z \end{bmatrix}$$

Basic kinematic model

$$\begin{bmatrix} v_x \\ v_y \\ \omega_z \end{bmatrix} = \frac{r}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 \\ -\frac{1}{(l_x+l_y)} & \frac{1}{(l_x+l_y)} & -\frac{1}{(l_x+l_y)} & \frac{1}{(l_x+l_y)} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}$$

i	Wheels	α_i	β_i	γ_i	l_i	l_{ix}	l_{iy}
0	1sw	$\pi/4$	$\pi/2$	$-\pi/4$	1	l_x	l_y
1	2sw	$-\pi/4$	$-\pi/2$	$\pi/4$	1	l_x	l_y
2	3sw	$3\pi/4$	$\pi/2$	$\pi/4$	1	l_x	l_y
3	4sw	$-3\pi/4$	$-\pi/2$	$-\pi/4$	1	l_x	l_y

<https://research.ijcaonline.org/volume113/number3/pxc3901586.pdf>

Motion of omnidirectional platform

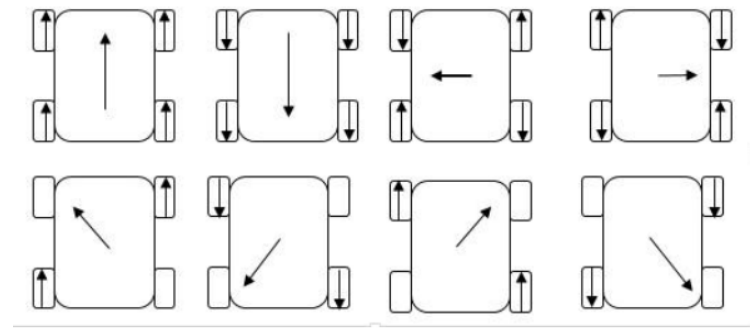
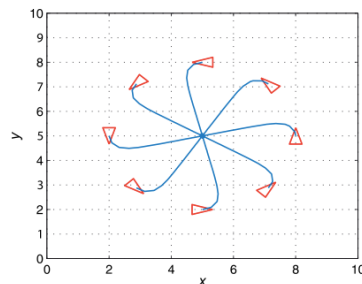


Fig 4: Motions of Omnidirectional platform

Moving to a Point

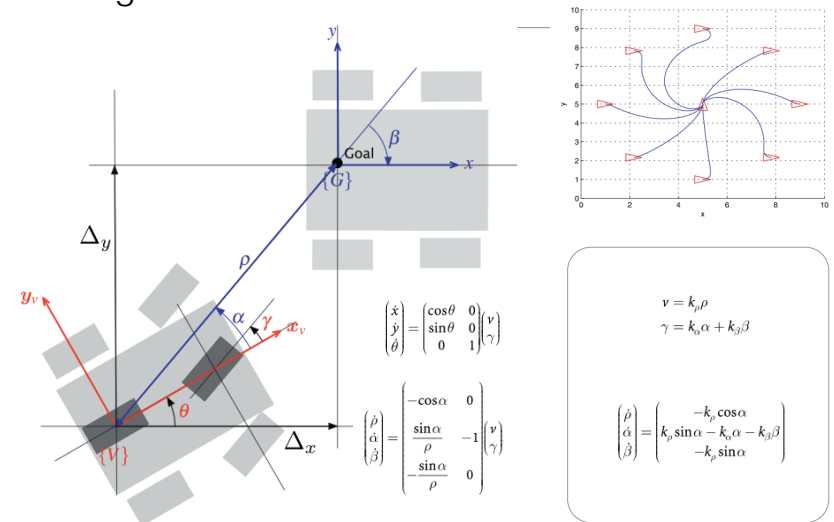


$$v^* = K_v \sqrt{(x^* - x)^2 + (y^* - y)^2}$$

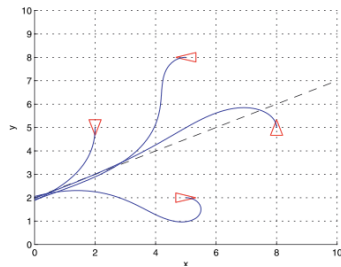
$$\theta^* = \tan^{-1} \frac{y^* - y}{x^* - x}$$

$$\gamma = K_\theta (\theta^* \ominus \theta), \quad K_\theta > 0$$

Moving to a Pose



Following a Line



$$d = \frac{(a, b, c) \cdot (x, y, 1)}{\sqrt{a^2 + b^2}}$$

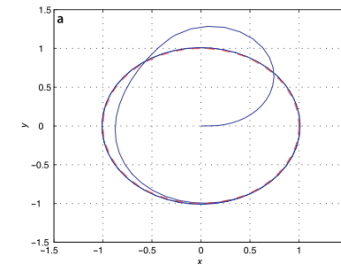
$$\alpha_d = -K_d d, K_d > 0$$

$$\theta^* = \tan^{-1} \frac{-a}{b}$$

$$\alpha_h = K_h(\theta^* \ominus \theta), K_h > 0$$

$$\gamma = -K_d d + K_h(\theta^* \ominus \theta)$$

Following a path



$$e = \sqrt{(x^* - x)^2 + (y^* - y)^2} - d^*$$

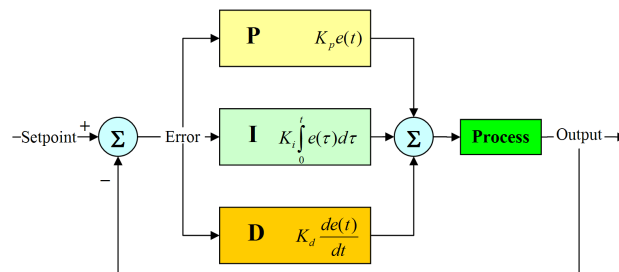
$$\dot{v} = K_v e + K_i \int e dt$$

$$\theta^* = \tan^{-1} \frac{y^* - y}{x^* - x}$$

$$\alpha = K_h(\theta^* \ominus \theta), K_h > 0$$

PID Control

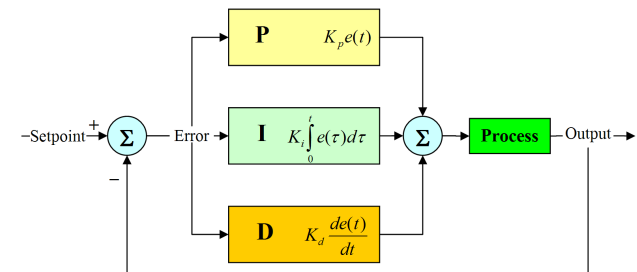
- **P** = Proportional
- **I** = Integral
- **D** = Derivative



- Simple
- Robust, excellent results in many cases
- Controls 95% of all industrial processes

PID Control

- **P** = Proportional: correct
- **I** = Integral: reduce tracking error
- **D** = Derivative: stabilize (anticipate)



Code Example

```
class PID:
    def __init__(self, Kp, Ki, Kd):
        self.previous_error = 0
        self.integral = 0
        self.Kp = Kp
        self.Ki = Ki
        self.Kd = Kd

    def calc(self, dt, setpoint, y):
        error = setpoint - y
        self.integral += error*dt
        derivative = (error - self.previous_error)/dt
        u = self.Kp*error + self.Ki*self.integral + self.Kd*derivative
        self.previous_error = error
        return u
```

Python example due to Andy Henshaw at GTRI

PID Tuning

- Ziegler-Nichols
 - Start with P only
 - Increase Kp until output oscillates
 - Call that “Ultimate gain” K_u , and call the oscillation period T_u

Ziegler-Nichols method ^[1]			
Control Type	K_p	K_i	K_d
P	$0.5K_u$	-	-
PI	$0.45K_u$	$1.2K_p/T_u$	-
PD	$0.8K_u$	-	$K_pT_u/8$
classic PID ^[2]	$0.60K_u$	$2K_p/T_u$	$K_pT_u/8$

Source: Wikipedia

Summary

- Configuration of mobile systems
 - Configuration Space
 - Actuators
 - 2D vs 3D space
 - Is the system fully controlled
- Simple example of kinematic models for mobile systems
 - Bi-cycle, Differential drive, Ackerman,
- Basic control algorithms