Locomotion of a Mobile Robot in a 2D Space
Differential Drive

Corrado Santoro

ARSLAB - Autonomous and Robotic Systems Laboratory
Dipartimento di Matematica e Informatica - Università di Catania, Italy
santoro@dmi.unict.it

Robotic Systems

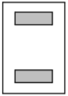
Locomotion of a Robot in a 2-Dimensional Space

Model

- According to the desired freedom degrees, a wide range of **locomotion model exists**
- Each model consider a certain number of wheels (2, 3, 4, etc.)
- and a certain kind of wheels (traction, free, steering, castor, omniball, omnidirectional, etc.)

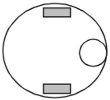
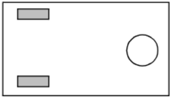
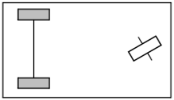
Locomotion of a Robot in a 2-Dimensional Space

Two Wheels

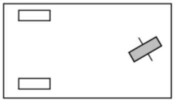
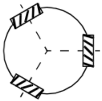
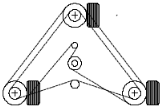
2		One steering wheel in the front, one traction wheel in the rear	Bicycle, motorcycle
		Two-wheel differential drive with the center of mass (COM) below the axle	Cye personal robot

Locomotion of a Robot in a 2-Dimensional Space

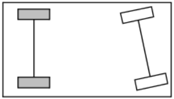
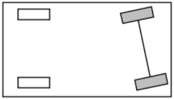
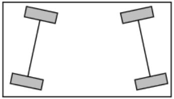
Three Wheels

3		Two-wheel centered differential drive with a third point of contact	Nomad Scout, smartRob EPFL
		Two independently driven wheels in the rear/front, 1 unpowered omnidirectional wheel in the front/rear	Many indoor robots, including the EPFL robots Pygmalion and Alice
		Two connected traction wheels (differential) in rear, 1 steered free wheel in front	Piaggio minitrucks

Three Wheels

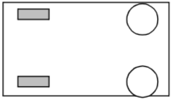
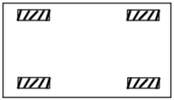
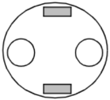
	Two free wheels in rear, 1 steered traction wheel in front	Neptune (Carnegie Mellon University), Hero-1
	Three motorized Swedish or spherical wheels arranged in a triangle; omnidirectional movement is possible	Stanford wheel Tribolo EPFL, Palm Pilot Robot Kit (CMU)
	Three synchronously motorized and steered wheels; the orientation is not controllable	"Synchro drive" Denning MRV-2, Georgia Institute of Technology, I-Robot B24, Nomad 200

Four Wheels

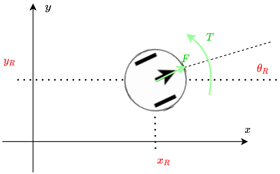
4		Two motorized wheels in the rear, 2 steered wheels in the front; steering has to be different for the 2 wheels to avoid slipping/skidding.	Car with rear-wheel drive
		Two motorized and steered wheels in the front, 2 free wheels in the rear; steering has to be different for the 2 wheels to avoid slipping/skidding.	Car with front-wheel drive
		Four steered and motorized wheels	Four-wheel drive, four-wheel steering Hyperion (CMU)

Locomotion of a Robot in a 2-Dimensional Space

Four Wheels

	Two traction wheels (differential) in rear/front, 2 omnidirectional wheels in the front/rear	Charlie (DMT-EPFL)
	Four omnidirectional wheels	Carnegie Mellon Uranus
	Two-wheel differential drive with 2 additional points of contact	EPFL Khepera, Hyperbot Chip

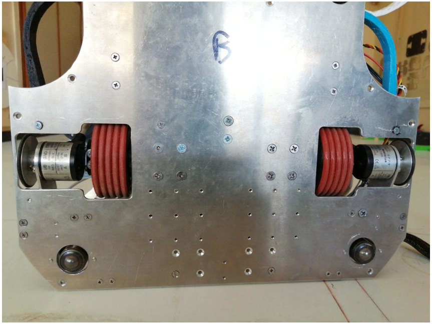
Locomotion of a Robot in a 2-Dimensional Space

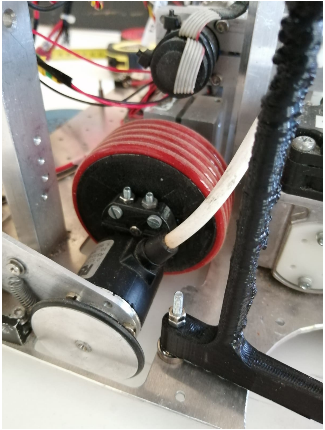


Wheels and Dynamic Model

- For each type of locomotion system, the kinematic and dynamic model must consider the **forces** generated by the **traction wheels**
- Such forces must then be transformated into F and T according to the "generic" 2D robot model
- In a similar way, according to the model, **position sensors** are tied to the wheels
- so they do not generate directly $\{x_R, y_R, \theta_R\}$ and a proper transformation is needed also in this case

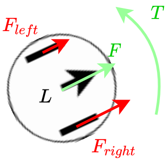
Two Indipendent Wheels
Differential Drive





- Two indipendent **traction** wheels (in red)
- Two indipendent **encoders** (with free wheels, in black) to track the position

Differential Drive3

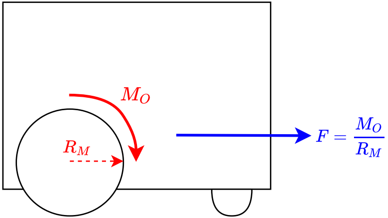


Dynamic Model

- Since we drive the traction wheels, we need a transformation from (F_{left}, F_{right}) to (F, T)
- We have:

$$F = F_{left} + F_{right}$$
$$T = L\,(F_{right} - F_{left})$$

where L is the (estimated) distance between the two traction wheels



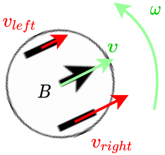
Dynamic Model

- Indeed, wheel motors generate **torques** that then are transformed in forces on the basis of the radius the traction wheels:

$$F_{left} = \frac{M_{O_{left}}}{R_{M_{left}}}$$
$$F_{right} = \frac{M_{O_{right}}}{R_{M_{right}}}$$

where $R_{M_{xxxx}}$ is the radius of the wheel and $M_{O_{xxxx}}$ is the torque generated by the motor

Differential Drive



Kinematic Model

- Since we measure the two wheels, we need a transformation from (v_{left}, v_{right}) to (v, ω)
- We have:

$$v = \frac{v_{left} + v_{right}}{2} \qquad \omega = \frac{v_{right} - v_{left}}{B}$$

$$v_{left} = v - \frac{\omega B}{2} \qquad v_{right} = v + \frac{\omega B}{2}$$

where B (wheelbase) is the (estimated) distance between the **two measurement wheels**

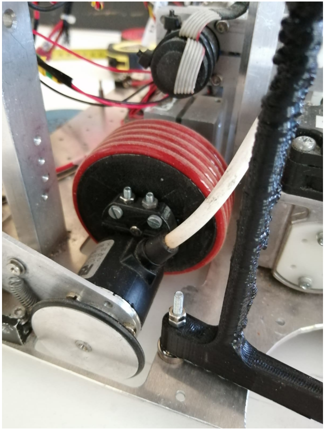
Measurement Wheels

- **Encoders** are digital sensors able to measure **rotation angles** (rather than speed)
- For each ΔT each encoder can provide the (relative) rotation angle of the wheel, i.e. $\Delta\theta_{left}$, $\Delta\theta_{right}$
- Given r_{left}, r_{right} the **radius** of each wheel we can compute the **distance** travelled by each wheel:

$$\Delta p_{left} = \Delta\theta_{left}\, r_{left} \qquad \Delta p_{right} = \Delta\theta_{right}\, r_{right}$$

- Speed can be thus computed as:

$$v_{left} = \frac{\Delta p_{left}}{\Delta T} \qquad v_{right} = \frac{\Delta p_{right}}{\Delta T}$$



Kinematic Model (Odometry or Dead-reckoning)

The final kinematics is given by:

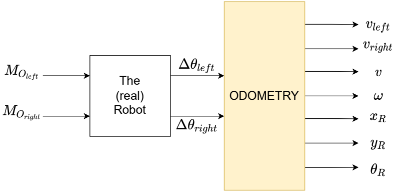
$$\Delta p_{left} = \Delta\theta_{left}\, r_{left} \qquad \Delta p_{right} = \Delta\theta_{right}\, r_{right}$$

$$v_{left} = \frac{\Delta p_{left}}{\Delta T} \qquad v_{right} = \frac{\Delta p_{right}}{\Delta T}$$

$$v = \frac{v_{left} + v_{right}}{2} \qquad \omega = \frac{v_{right} - v_{left}}{B}$$

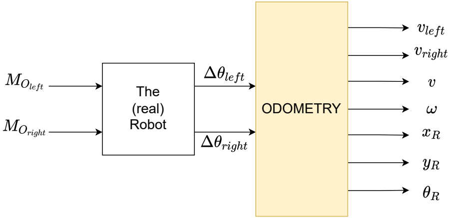
$$\Delta p = \frac{\Delta p_{left} + \Delta p_{right}}{2} \qquad \Delta\theta = \frac{\Delta p_{right} - \Delta p_{left}}{B}$$

$$\begin{aligned} x_R &= x_R + \Delta p \cos(\theta_R + \frac{\Delta\theta}{2}) \\ y_R &= y_R + \Delta p \sin(\theta_R + \frac{\Delta\theta}{2}) \\ \theta_R &= \theta_R + \Delta\theta \end{aligned}$$



The Robot and the Sensors

- We can intervene on the robot by modulating the **torques** generated by left and right motors
- We can sense the robot by gathering data sampled by **encoders** in terms of angle variation of left and right measurement wheels
- The other kinematic paramaters are then computed by means of the **odometry** algorithm

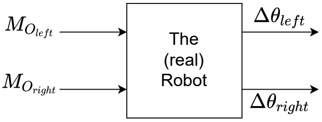


Approximation introducted by the Odometry Algorithm

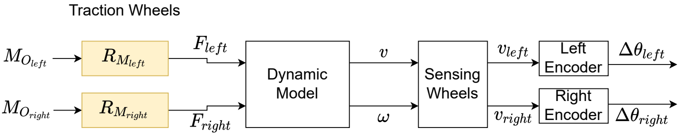
- Encoders are digital sensors so $\Delta\theta_{left}$ and $\Delta\theta_{right}$ are approximated according to a certain resolution
- The algorithm assumes that r_{left} and r_{right} are **known**, but the **real radius** depends on the load on each sensing wheel
- The algorithm assumes that v_{left} and v_{right} are **constant** during ΔT, and this may not be the case
- The algorithm assumes that the mass' center of the robot is placed in the **middle point of the traction axis**, but if the weight is not uniformly distributed the reality is quite different

Model and Simulation

Differential Drive



- According to the study made so far we can model our robot with two independent wheels
- We consider the robot as a system with two inputs, the **torques** generated by the motors ...
- ... and two outputs, the **angles** traveled by left and right sensing wheels



$$F_{left} = \frac{M_{O_{left}}}{R_{M_{left}}} \qquad F_{right} = \frac{M_{O_{right}}}{R_{M_{right}}}$$

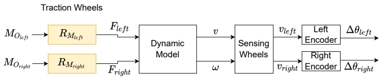
$$F = F_{left} + F_{right} \qquad T = L\left(F_{right} - F_{left}\right)$$

$$\dot{v} = -\frac{b}{M}v + \frac{1}{M}F$$

$$\dot{\omega} = -\frac{2\beta}{Mr^2}\omega + \frac{2}{Mr^2}T$$

$$v_{left} = v - \frac{\omega B}{2} \qquad v_{right} = v + \frac{\omega B}{2}$$

$$\dot{\theta}_{left} = \frac{v_{left}}{r_{left}} \qquad \dot{\theta}_{right} = \frac{v_{right}}{r_{right}}$$



Encoders, the matter of resolution

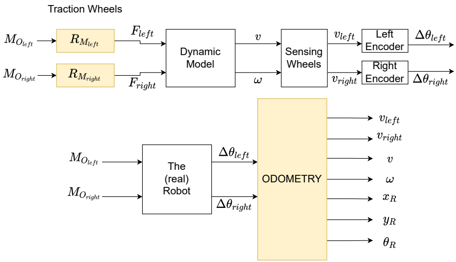
$$\dot{\theta}_{left} = \frac{v_{left}}{r_{left}} \qquad \dot{\theta}_{right} = \frac{v_{right}}{r_{right}}$$

$$\Delta\theta_{left} = \frac{v_{left}}{r_{left}}\Delta T \qquad \Delta\theta_{right} = \frac{v_{right}}{r_{right}}\Delta T$$

- Encoders are **digital sensors** so they feature a certain specific **resolution** which is given as the minimum angle (variation) ϵ they can percieve
- The **real** sensed data is therefore:

$$\Delta\theta_{left} = \left\lfloor \frac{v_{left}}{r_{left}}\Delta T \frac{1}{\epsilon} \right\rfloor \epsilon \qquad \Delta\theta_{right} = \left\lfloor \frac{v_{right}}{r_{right}}\Delta T \frac{1}{\epsilon} \right\rfloor \epsilon$$

Differential Drive

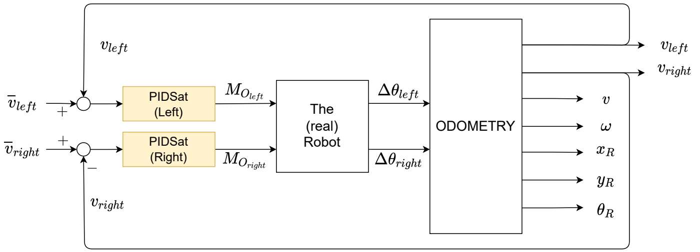


The Code

See:

- `lib/system/cart.py`
- `examples/differential_drive/test_differential_drive.ipynb`

Motion Control

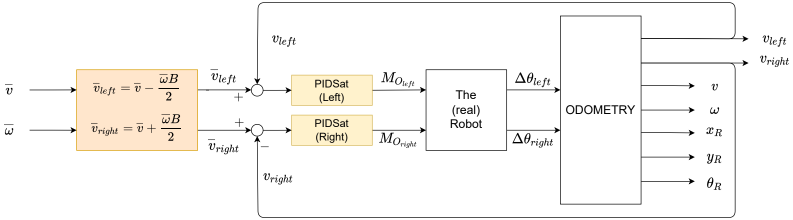


Speed Control

- We can control each wheel in **speed** by comparing the actual speed $v_{left\,/\,right}$ with the relevant reference $\overline{v}_{left\,/\,right}$
- To this aim, we use **two different PID controllers**, one for each wheel
- The output of each controller is the **command for the motors**, i.e. the torque in our case

(see `examples/differential_drive/wheel_speed_control.ipynb`)

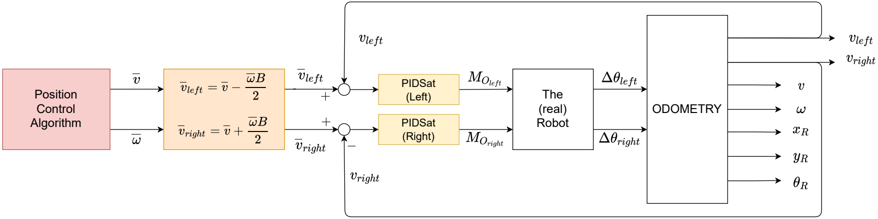
Differential Drive



Speed Control

- If we add a transformation block from (v, ω) to (v_{left}, v_{right}), we can control the robot using the speeds of the rigid body

(see **examples/differential_drive/polar_speed_control.ipynb**)



Position Control

- In turn, we can easily apply all position control algorithms we studied (polar, speed profile, etc.)

(see `examples/differential_drive/trajectory_control.ipynb`)

Locomotion of a Mobile Robot in a 2D Space
Differential Drive

Corrado Santoro

ARSLAB - Autonomous and Robotic Systems Laboratory
Dipartimento di Matematica e Informatica - Università di Catania, Italy
santoro@dmi.unict.it

Robotic Systems