

Introduction to Cognitive Robotics

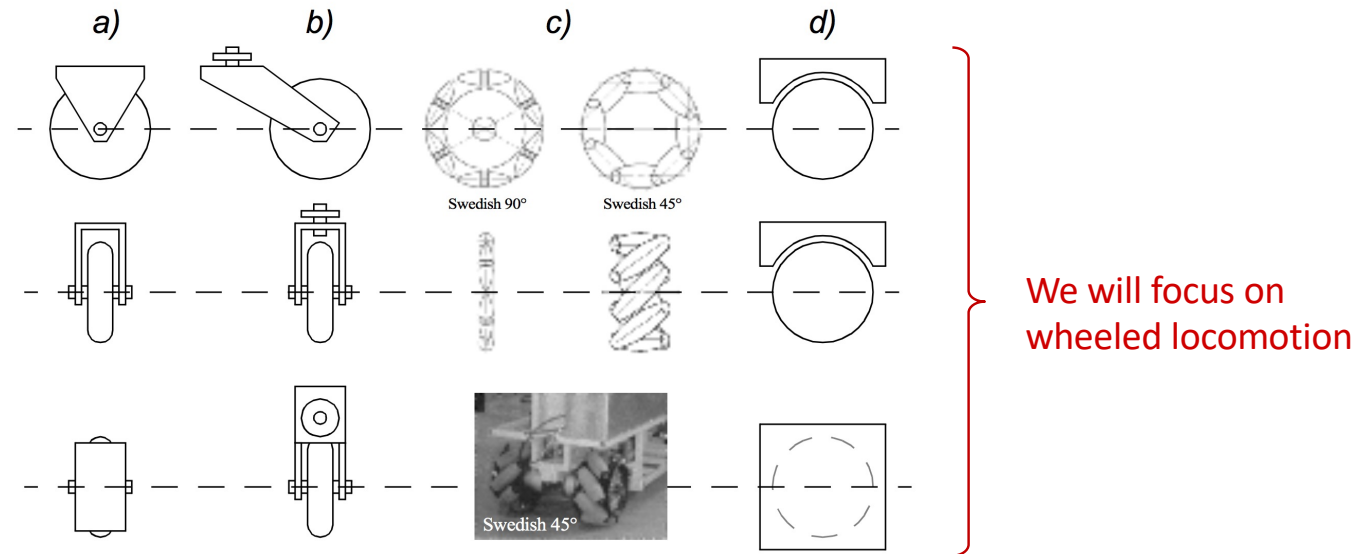
Module 3: Mobile Robots

Lecture 4: Relative position estimation using odometry

David Vernon
Carnegie Mellon University Africa

www.vernon.eu

Wheeled Locomotion



Source: R. Siegwart and I. R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*, MIT Press, 2004

- (a) Standard wheel → Rotation about axle for movement and about contact point for steering
- (b) Castor wheel → Rotation about axle for movement and about vertical axis for steering; imparting a force on the robot body when steering
- (c) Swedish wheel → Rotation about axle for movement but also about rollers allowing movement in any direction
- (d) Ball or spherical wheel → Omnidirectional wheel: can spin in any direction

Wheeled Locomotion

# of wheels	Arrangement	Description	Typical examples
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
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
		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

# of wheels	Arrangement	Description	Typical examples
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)
		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
		Four motorized and steered castor wheels	Nomad XR4000

# of wheels	Arrangement	Description	Typical examples
6		Two motorized and steered wheels aligned in center, 1 omnidirectional wheel at each corner	First
		Two traction wheels (differential) in center, 1 omnidirectional wheel at each corner	Terregator (Carnegie Mellon University)
Icons for the each wheel type are as follows:			
	unpowered omnidirectional wheel (spherical, castor, Swedish);		
	motorized Swedish wheel (Stanford wheel);		
	unpowered standard wheel;		
	motorized standard wheel;		
	motorized and steered castor wheel;		
	steered standard wheel;		
	connected wheels.		

Source: R. Siegwart and I. R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*, MIT Press, 2004

Wheeled Locomotion

# of wheels	Arrangement	Description	Typical examples
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
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
		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

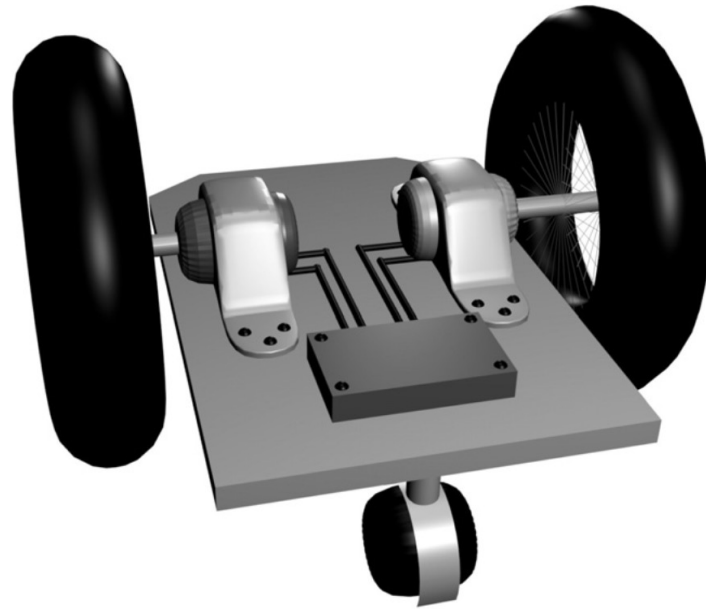
# of wheels	Arrangement	Description	Typical examples
4		Two motorized wheels in the rear, 2 steered wheels in the front; steering has to be different for the 2 wheels to avoid slipping/skidding.	Car with rear-wheel drive
		Two motorized and steered wheels in the front, 2 free wheels in the rear; steering 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)
		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
		Four motorized and steered castor wheels	Nomad XR4000

# of wheels	Arrangement	Description	Typical examples
6		Two motorized and steered wheels aligned in center, 1 omnidirectional wheel at each corner	First
		Two traction wheels (differential) in center, 1 omnidirectional wheel at each corner	Terregator (Carnegie Mellon University)
Icons for the each wheel type are as follows:			
	unpowered omnidirectional wheel (spherical, castor, Swedish);		
	motorized Swedish wheel (Stanford wheel);		
	unpowered standard wheel;		
	motorized standard wheel;		
	motorized and steered castor wheel;		
	steered standard wheel;		
	connected wheels.		

We will study two-wheel differential drive locomotion

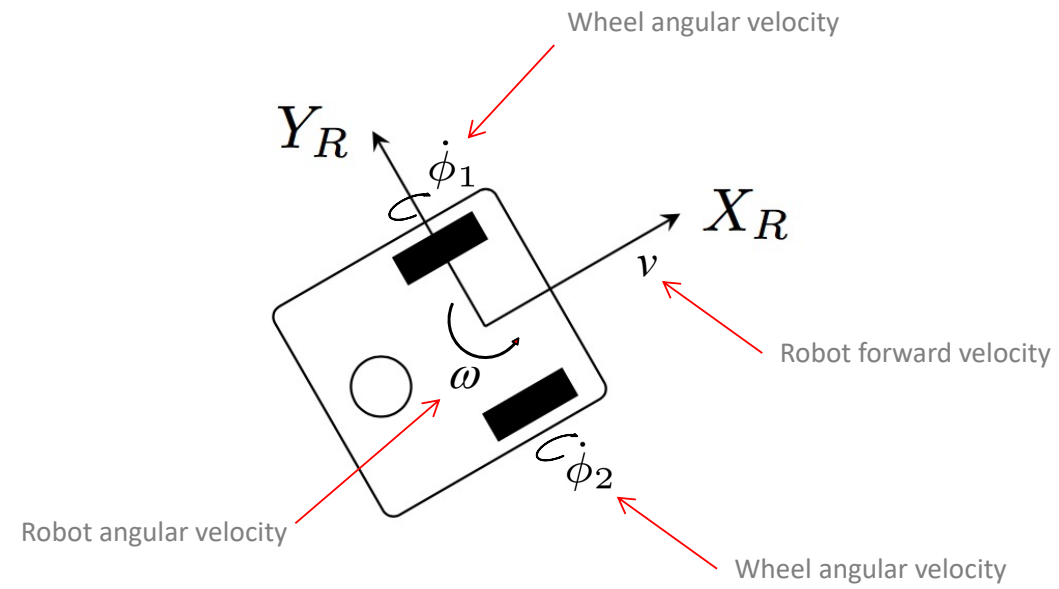
Source: R. Siegwart and I. R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*, MIT Press, 2004

Wheeled Locomotion



Source: M. Mataric, The Robotics Primer, MIT Press, 2007

Wheeled Locomotion



Odometry-based Position Estimation

Overall approach

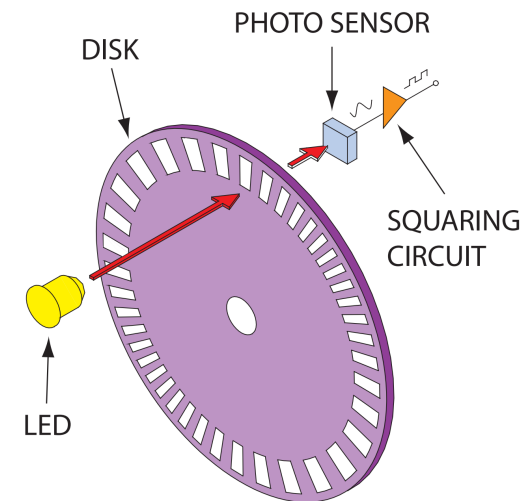
- Measure the **angular velocities** of the left and right wheels
- Compute the instantaneous velocities in the robot frame of reference
- Compute the displacement and change in orientation (in a given time interval) in the robot frame of reference R
- Compute the displacement and change in orientation (in a given time interval) in the global frame of reference (inertial frame of reference I)
- Update the position of the robot with respect to its previous position
- Repeat update (e.g. every 100 ms)

$$\dot{\phi}_1 \quad \dot{\phi}_2$$

Angular velocities of the left and right wheels

Need to use **proprioceptive** sensors

- Typically, **encoders** on the **wheels**
- Go to <http://encoder.com/videos/> to watch a video explaining the operation of encoders

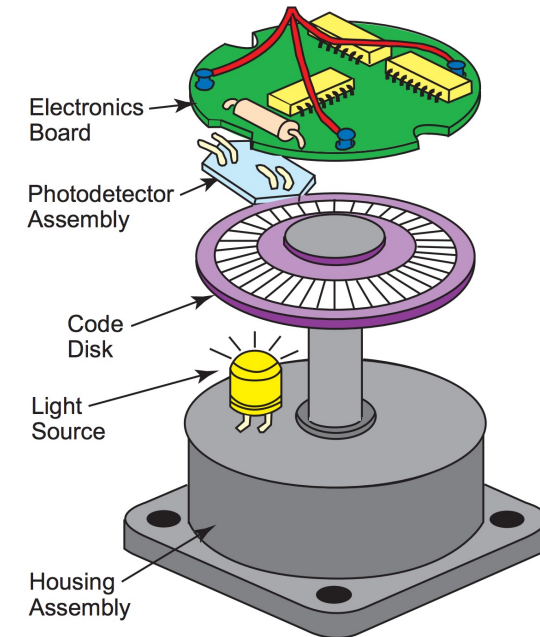


Source: <http://encoder.com/core/files/encoder/uploads/files/WP-2011.pdf>

Angular velocities of the left and right wheels

Need to use **proprioceptive** sensors

- Typically, **encoders** on the **wheels**
- Go to <http://encoder.com/videos/> to watch a video explaining the operation of encoders



Source: <http://encoder.com/core/files/encoder/uploads/files/WP-2011.pdf>

Odometry-based Position Estimation

Overall approach

- Measure the angular velocities of the left and right wheels
- Compute the **instantaneous velocities** in the robot frame of reference
- Compute the displacement and change in orientation (in a given time interval) in the robot frame of reference R
- Compute the displacement and change in orientation (in a given time interval) in the global frame of reference (inertial frame of reference I)
- Update the position of the robot with respect to its previous position
- Repeat update (e.g. every 100 ms)

$$\begin{array}{cc} \dot{\phi}_1 & \dot{\phi}_2 \\ \downarrow & \\ v, \omega & \end{array}$$

Odometry-based Position Estimation

Overall approach

- Measure the angular velocities of the left and right wheels
- Compute the instantaneous velocities in the robot frame of reference
- Compute the **displacement** and **change in orientation** (in a given time interval) in the **robot frame of reference R**
- Compute the displacement and change in orientation (in a given time interval) in the global frame of reference (inertial frame of reference I)
- Update the position of the robot with respect to its previous position
- Repeat update (e.g. every 100 ms)

v, ω



$$\Delta\xi_R = (\Delta x, \Delta y, \Delta\theta)$$

Odometry-based Position Estimation

Overall approach

- Measure the angular velocities of the left and right wheels
- Compute the instantaneous velocities in the robot frame of reference
- Compute the displacement and change in orientation (in a given time interval) in the robot frame of reference R
- Compute the **displacement** and **change in orientation** (in a given time interval) in the **global frame of reference** (inertial frame of reference I)
- Update the position of the robot with respect to its previous position
- Repeat update (e.g. every 100 ms)

$$\Delta\xi_R = (\Delta x, \Delta y, \Delta\theta)$$



$$\Delta\xi_I = (\Delta x_I, \Delta y_I, \Delta\theta)$$

Odometry-based Position Estimation

Overall approach

- Measure the angular velocities of the left and right wheels
- Compute the instantaneous velocities in the robot frame of reference
- Compute the displacement and change in orientation (in a given time interval) in the robot frame of reference R
- Compute the displacement and change in orientation (in a given time interval) in the global frame of reference (inertial frame of reference I)
- Update the position of the robot with respect to its previous position
- Repeat update (e.g. every 100 ms)

$$x_I(t+1) = x_I(t) + \Delta x_I$$

$$y_I(t+1) = y_I(t) + \Delta y_I$$

$$\theta(t+1) = \theta(t) + \Delta\theta$$

Update the position of the robot with respect to its previous position

Assuming we know the position at time $t = 0$,
we estimate the position at time in the next time period $t = 1$ as follows

$$x_I(1) = x_I(0) + \Delta x_I$$

$$y_I(1) = y_I(0) + \Delta y_I$$

$$\theta(1) = \theta(0) + \Delta\theta$$

Odometry-based Position Estimation

Overall approach

- Measure the angular velocities of the left and right wheels
- Compute the instantaneous velocities in the robot frame of reference
- Compute the displacement and change in orientation (in a given time interval) in the robot frame of reference
- Compute the displacement and change in orientation in the global frame of reference (inertial frame of reference)
- Update the position of the robot with respect to its previous position
- Repeat update (e.g. every 100 ms)

Repeat Update

We update the position at each time period

$$x_I(t + 1) = x_I(t) + \Delta x_I$$

$$y_I(t + 1) = y_I(t) + \Delta y_I$$

$$\theta(t + 1) = \theta(t) + \Delta\theta$$

Odometry-based Position Estimation

Overall approach

- Measure the **angular velocities** of the left and right wheels $\dot{\phi}_1 \quad \dot{\phi}_2$
- Compute the **instantaneous velocities** in the robot frame of reference v, ω
- Compute the **displacement** and **change in orientation** (in a given time interval) in the **robot frame of reference R** $\Delta\xi_R = (\Delta x, \Delta y, \Delta\theta)$
- Compute the **displacement** and **change in orientation** (in a given time interval) in the **global frame of reference** (inertial frame of reference I) $\Delta\xi_I = (\Delta x_I, \Delta y_I, \Delta\theta)$
- Update the position of the robot with respect to its previous position $x_I(t+1) = x_I(t) + \Delta x_I$
- Repeat update (e.g. every 100 ms) $y_I(t+1) = y_I(t) + \Delta y_I$
 $\theta(t+1) = \theta(t) + \Delta\theta$

Odometry-based Position Estimation

Limitations of Odometry – Sources of Error

- Assumptions
 - Assumes control variables (angular velocities) are constant
- Robot's geometry (**systematic errors**)
 - wheel imbalance: one wheel larger than the other
 - axis imbalance: one axis is longer than the other
 - actual wheel distance not the same as nominal one (D)
- Interaction with environment (**non-systematic errors**)
 - slippage of one wheel
 - bumps and gaps in the floor
- The main problem
 - **Odometry error can grow without bounds**

Odometry-based Position Estimation

Limitations – Sources of Error

- Difference in wheel radius E_r
 - the robot makes a **curved** instead of a **straight** path

(because, for identical left and right angular velocities, the associated displacements will be different)
- Error in wheel distance E_l
 - the robot over-estimates or under-estimates the amount of turning

(because the angle is a function of l , the wheel distance)

Summary

- Absolute position estimation (localization)
 - match observed features with prior knowledge (map)
 - requires good perception, good map
- Relative position estimation (position tracking)
 - update previous position by measuring displacement
 - inevitably drifts with time
- General solution: combined
 - absolute localization whenever possible
 - continuous position tracking in between localizations

Odometry-based Position Estimation

Now we need to **know how to compute these three transformations**:

- Measure the angular velocities of the left and right wheels
- Compute the **instantaneous velocities** in the robot frame of reference
- Compute the **displacement** and **change in orientation** (in a given time interval) in the **robot frame of reference R**
- Compute the **displacement** and **change in orientation** (in a given time interval) in the **global frame of reference** (inertial frame of reference I)
- Update the position of the robot with respect to its previous position
- Repeat update (e.g. every 100 ms)

