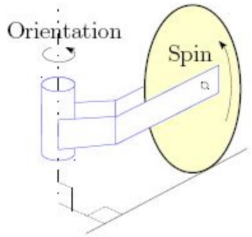
MOBILE ROBOTS

VOCABULARY

- SPEED: time derivative of a scalar position variable
- VELOCITY: a vector, time derivative of the position of a point
- TWIST: instantaneous motion of a rigid body. It can be angular speed around an axis and speed along this axis. $[\dot{x}, \dot{y}, \dot{\theta}]$

NOTE: Speed is a Scalar, Velocity is a Vector

- WHEELED MOBILE ROBOT: a robot capable of locomotion on a surface, through the actuation of wheel assemblies
- WHEEL ASSEMBLY: mechanical device which allows the wheel to rotate around its spin axis. In addition, the device may also allow the wheel to rotate around one orientation axis perpendicular to the spin axis and to the ground



WORK ASSUMPTIONS

- The robot moves on flat horizontal plane (the fact that is horizontal is useful only for dynamics)
- The plane is a rigid, non deformable body
- The robot is composed of a single rigid platform (chassis), to which the wheel assemblies are attached
- The robot has at least three wheels. For stability, the three contact points should never be aligned

ASSUMPTIONS ABOUT WHEELS AND WHEEL-GROUND CONTACT

- The wheels are rigid disks with an equal radius r
- The wheels are perpendicular to the rolling plane
  - Wheel-ground contact is reduced to a single point
- The wheels are always in contact with the ground
- The motion of the wheels is a PURE ROLLING MOTION

There is no SLIPPING: tangential speed is zero
There is no SKIDDING: normal speed is zero
ROTATIONAL SLIPPING is allowed, meaning that ω_z can be non zero (this allows the wheel to turn)

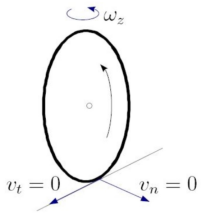
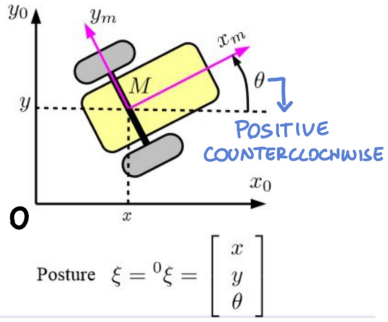


Figure 1.2: Ideal wheel-ground contact.

FRAMES AND ROBOT POSTURE

The frame where the robot moves is characterized by the fixed frame $R_0(O, x_0, y_0, z_0)$, with z_0 being perpendicular to the plane and upward
The frame $R_m(M, x_m, y_m, z_m)$ is attached to the platform, with z_m parallel to z_0, and the x-y plane passing through the center points of the wheel

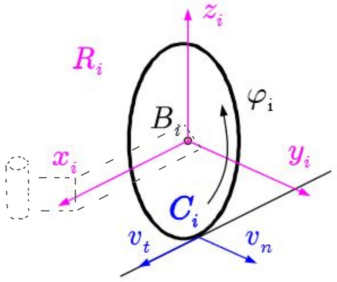


The coordinates (x,y) of M and the orientation θ of the robot with respect to R_0 (angle between x_m and x_0) define the POSTURE of the robot in the fixed frame

$$\xi = {}^0\xi = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

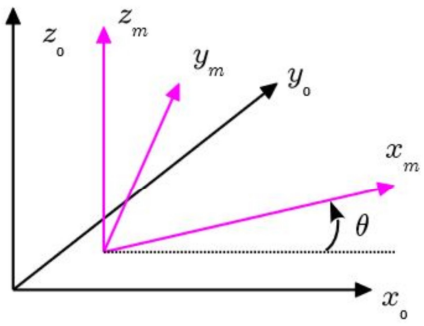
NOTE: The orientation θ of the robot is often called HEADING

FRAME ATTACHED TO WHEEL ASSEMBLY



There is a frame attached to each wheel.
Heach frame has an index, starting from 1
The frames are attached to the supporting body, so axis z_i is always vertical and directed upward.
Axis y_i is perpendicular to the plane of the wheel and hence the wheel spim f_i happen around y_i

OSS! $z_0 = z_1 = \dots = z_m$



Since the rotation θ is always about the z axis, then the rotation matrix between the absolute reference frame and the robot frame will be:

$${}^0R_m = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad {}^0R_m = {}^mR_0^{-1} = {}^mR_0^T$$

TWIST OF THE ROBOT

The twist of the platform with respect to frame R_0, is the time derivative of the posture ${}^0\dot{\xi} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}$

Will be useful to express the twist of the robot either in R_0 or R_m

$${}^0\Omega_m = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad {}^0\Omega_m = {}^m\Omega_0^{-1} = {}^m\Omega_0^T \qquad {}^0\dot{\xi} = {}^0\Omega_m \cdot {}^m\dot{\xi}$$

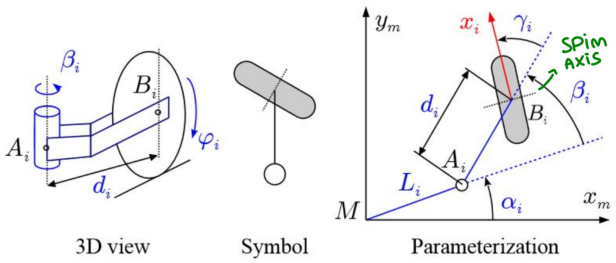
→ It's the same rotation matrix used for posture, this rotation is for twist

NOTE: Ω is used only for twist. It is only a notational convenience, since the real equations should be:

$$\begin{bmatrix} {}^0\dot{x} \\ {}^0\dot{y} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} {}^m\dot{x} \\ {}^m\dot{y} \end{bmatrix} \quad \text{and} \quad {}^m\dot{\theta} = {}^0\dot{\theta}$$

THE DIFFERENT TYPES OF WHEELS

GENERAL WHEEL



3D view Symbol Parameterization

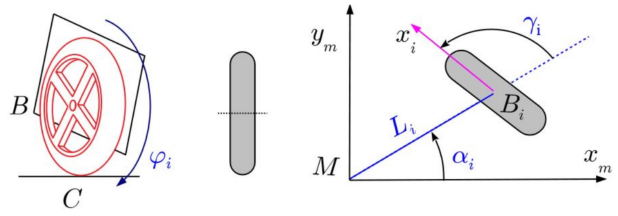
Constant parameters: L, d, α, γ
Variable parameters: β, φ

Parameters L and α locate the orientation axis of the wheel assembly with respect to the platform frame in polar coordinates
Parameters d, β and γ in turn are necessary to locate the wheel frame R_i with respect to the platform
The only variable parameters are β and φ
Finally, one important parameter is:
$\psi = \alpha + \beta + \gamma$ (Angle between x_m and x_i)

OSS: α, β, γ are rotations around z (positive direction is x to y)
φ is a rotation around y (positive direction is z to x)

The general wheel is only a concept. It does not exist (except $\gamma = \pm\frac{\pi}{2}$ ← Perpendicular wheel)
The mobile robots use CONVENTIONAL WHEEL, derived by the general one by setting parameters to 0

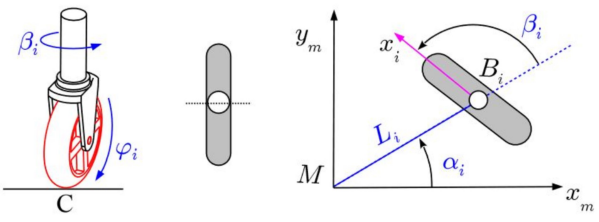
FIXED WHEEL



A fixed wheel is such that its spin axis y_i is fixed with respect to the mobile robot frame R_m
It is completely defined by the position of its center B in R_m and the fixed orientation of x_i

The fixed wheel is a general wheel for which: $A_i = B_i$, $d_i = 0$ and $\beta_i = 0$

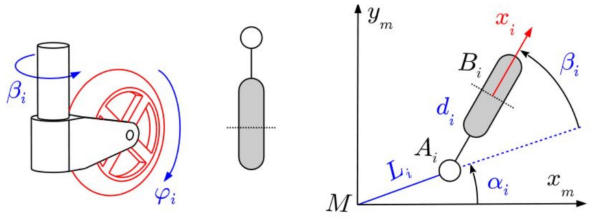
STEERABLE WHEEL (CENTERED ORIENTABLE WHEEL)



The orientation axis goes through the point of contact of the wheel with the ground
NOTE! pay attention to the difference in parameterization: instead of constant γ_i we have a variable angle β

The steerable wheel is a fixed wheel with: $A_i = B_i$, $d_i = 0$ and $\gamma_i = 0$

CASTOR WHEEL (OFF-CENTERED ORIENTABLE WHEEL)



Its orientation axis is in the plane of the wheel
The castor wheel is a general wheel for which: $\gamma = 0$ (x points away from A)
By definition, the parameter d is not zero

NOTE WELL: for any wheel L_i, α_i, d_i, γ_i are constant

PARAMETRIZATION OF A ROBOT

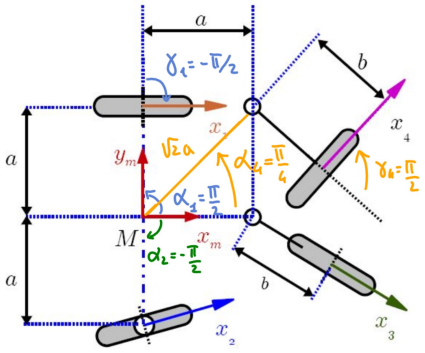
1. Draw wheel symbols according to robot geometry and indicate the characteristic dimensions.
2. Set the robot frame, as much as possible in a way that simplifies the parameters and/or favors symmetry.
3. Number the wheels in the conventional order: fixed wheels (suffix f: $1f$, $2f$...), steerable wheels (suffix s), castor wheels (suffix c), general wheel (suffix g).
4. Draw axis X of each wheel.
5. Fill the parameter table as below: one line per wheel.

Wheel	L	α	d	β	γ	φ	$\psi=\alpha+\beta+\gamma$
$1f$	...						

PAY ATTENTION TO THIS

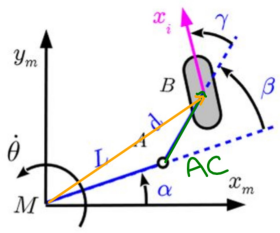
1. **Angles are positive counter-clockwise**. It is the most common source of errors.
2. Look at the proper figure in the memento (the suffix after the wheel number is here to remind you the type of wheel).
3. Angle β is either the value 0 or the name of the angle variable (say β_{3s}).
4. Replace L by the proper value a, b, c…
5. Column φ is just a variable name, say φ_{3s}.
6. Column ψ is redundant. It's useful to avoid re-evaluating this angle each time it appears in an equation.

EXERCISE



wheel	L	α	d	β	γ	φ	ψ
$1f$	a	$\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$	φ_{1f}	0
$2s$	a	$-\frac{\pi}{2}$	0	β_{2s}	0	φ_{2s}	$\beta_{2s}-\frac{\pi}{2}$
$3c$	a	0	b	β_{3c}	0	φ_{3c}	β_{3c}
$4g$	$\sqrt{2}a$	$\frac{\pi}{4}$	b	β_{4g}	$\frac{\pi}{2}$	φ_{4g}	$\beta_{4g}+\frac{3}{4}\pi$

VELOCITY OF THE CONTACT POINT



The velocity of the contact point C is due to: the twist of the robot platform, the orientation speed of the wheel $\dot{\beta}$ and the spinning wheel $\dot{\gamma}$

$$\underline{V_C} = \underline{V_M} + \dot{\theta}\,\underline{z_m} \wedge \underline{MC} + \dot{\beta}\,\underline{z_m} \times \underline{AC} + \dot{\gamma}\underline{y_i} \times \underline{BC}$$

OSS: each term of the sum is an horizontal vector, so $\underline{V_C}$ is horizontal

It's possible to express terms in R_m or in R_i, depending on the easier form that they will have

$$ {}^{i}V_C = {}^{i}R_m \left({}^{m}R_0 \, {}^{0}V_M + \dot{\theta} z_m \times {}^{m}MC + \dot{\beta} z_m \times {}^{m}AC \right) + \dot{\varphi} y_i \times {}^{i}BC $$

NOTE: dropped vector notation for semplicity

with: $ {}^{0}V_M = \begin{bmatrix} \dot{x} \\ \dot{y} \\ 0 \end{bmatrix} \quad {}^{m}AC = \begin{bmatrix} d\cos(\alpha+\beta) \\ d \cdot \sin(\alpha+\beta) \\ -r \end{bmatrix} \quad {}^{i}BC = \begin{bmatrix} 0 \\ 0 \\ -r \end{bmatrix} $

$$ {}^{m}MC = {}^{m}MA + {}^{m}AC = \begin{bmatrix} L\cdot\cos(\alpha) \\ L\cdot\sin(\alpha) \\ 0 \end{bmatrix} + \begin{bmatrix} d\cos(\alpha+\beta) \\ d\cdot\sin(\alpha+\beta) \\ -r \end{bmatrix} \qquad {}^{i}R_m = {}^{m}R_i^{-1} = rot(z_m, -\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

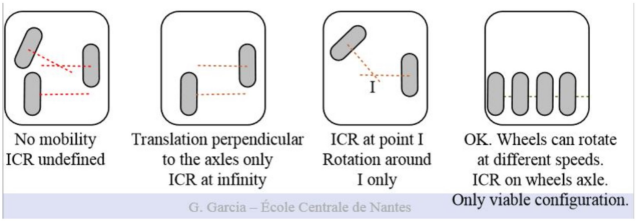
By considering only components on the plane and factorizing time derivatives

$$ \begin{bmatrix} 0 \\ v_t \\ v_m \\ 0 \end{bmatrix} = \begin{bmatrix} \cos\psi & \sin\psi & d\sin\psi + L\sin(\beta+\gamma) \\ -\sin\psi & \cos\psi & d\cos\psi + L\cos(\beta+\gamma) \end{bmatrix} {}^{m}\Omega_0 \, \dot{\xi} + \begin{bmatrix} d\cdot\sin\gamma \\ d\cdot\cos\gamma \end{bmatrix} \dot{\beta} + \begin{bmatrix} -r \\ 0 \end{bmatrix} \dot{\varphi} $$

$$ {}^{m}\Omega_0 = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

NOTE! INSTANTANEOUS CENTER OF ROTATION

For any rigid body moving in a plane, there exists a unique point with a zero velocity → ICR. At any time istant, the istantaneous motion of the solid is a rotation around that point (for translation, ICR is infinity). The ICR is the intersection of the normal vectors to the velocities of all points of the solid



CONSTRAINTS FOR FIXED WHEELS

with $d=0$ and $\beta=0$

$$ \begin{bmatrix} v_t \\ v_m \end{bmatrix} = \begin{bmatrix} \cos\psi & \sin\psi & L\cdot\sin\gamma \\ -\sin\psi & \cos\psi & L\cdot\cos\gamma \end{bmatrix} {}^{m}\Omega_0 \, \dot{\xi} + \begin{bmatrix} -r \\ 0 \end{bmatrix} \dot{\varphi} $$

The first constraint ($v_t = 0$) is always satisfiable, whatever is the twist
The second equation is important for the study of the mobility of the platform

CONSTRAINTS FOR STEERABLE WHEEL

with $d=0$ and $\gamma=0$

$$ \begin{bmatrix} v_t \\ v_m \end{bmatrix} = \begin{bmatrix} \cos\psi & \sin\psi & L\cdot\sin\beta \\ -\sin\psi & \cos\psi & L\cdot\cos\beta \end{bmatrix} {}^{m}\Omega_0 \, \dot{\xi} + \begin{bmatrix} -r \\ 0 \end{bmatrix} \dot{\varphi} $$

For any value of β, $v_t = 0$ can be satisfied by adapting $\dot{\varphi}$
Instead, $v_m = 0$ will be satisfied if β and $\dot{\xi}$ are compatible. The orientation axis (rotation β) must be motorized

NOTE! Two steerable wheels locate the ICR at the intersection of their y axes. Hence, no more than two steerable wheels can be independent. Moreover, a steerable wheel cannot be placed on the axle of fixed wheels, since it would locate the ICR at its point B. (orientation of steerable wheel would bear no interest)

CONSTRAINTS FOR CASTOR WHEELS

With $\gamma = 0$

$$\begin{bmatrix} v_t \\ v_m \end{bmatrix} = \begin{bmatrix} \cos\psi & \sin\psi & L\cdot\sin\beta \\ -\sin\psi & \cos\psi & L\cdot\cos\beta \end{bmatrix} {}^{m}R_0 \dot{\xi} + \begin{bmatrix} 0 \\ d \end{bmatrix} \dot{\beta} + \begin{bmatrix} -r \\ 0 \end{bmatrix} \dot{\varphi}$$

The constraint $v_t = 0$ can always be satisfied by adapting $\dot{\varphi}$
The second constraint $v_m = 0$ can be achieved by adapting $\dot{\beta}$, so whatever is the current orientation, the castor wheel can adapt $\dot{\varphi}$ and $\dot{\beta}$ to any platform motion, without violating pure rolling constraint

NOTE: if neither φ nor β is motorized, the wheel can be called IDLER. In fact $\dot{\varphi}$ and $\dot{\beta}$ will naturally adapt to satisfy the constraints

CONSTRAINTS FOR OTHER WHEEL ASSEMBLIES

$$\begin{bmatrix} v_t \\ v_m \end{bmatrix} = \begin{bmatrix} \cos\psi & \sin\psi & d\sin\psi + L\sin(\beta+\gamma) \\ -\sin\psi & \cos\psi & d\cos\psi + L\cos(\beta+\gamma) \end{bmatrix} {}^{m}R_0 \dot{\xi} + \begin{bmatrix} d\cdot\sin\gamma \\ d\cdot\cos\gamma \end{bmatrix} \dot{\beta} + \begin{bmatrix} -r \\ 0 \end{bmatrix} \dot{\varphi}$$

Assuming $d\cdot\cos(\gamma)$ is not zero, then the constraint $v_m = 0$ determines $\dot{\beta}$ and given that, $\dot{\varphi}$ is determined by solving $v_t = 0$

Tips and tricks - take note

Everything here below is under pure rolling assumption.
When asked a question about the location of the ICR, consider there are no motors, unless instructed otherwise. Then:

- Fixed and steerable wheels constrain robot motion.
- Fixed and steerable wheels constrain the position of the ICR.
- Castor wheels do not constrain robot motion, nor do they constrain the position of the ICR.
- Castor wheels can be ignored in the determination of the ICR.
- **Beware**: If at some point I assume that the orientation of a castor wheel is blocked in a certain position, it becomes a *de facto* fixed wheel, and hence constrains motion.
- With certain wheel combinations of steerable wheel orientations, the ICR may not exist (no motion is possible without violating non slipping constraints).

DEGREE OF MOBILITY

If the wheel configuration allows generating twist components v_x, v_y and ω INDEPENDENTLY, while maintaining pure rolling, then the degree of mobility δ_m of the robot is 3, the maximum for planar motion. Such a robot is called OMNIDIRECTIONAL. $\delta_m = 0$ bears no interest

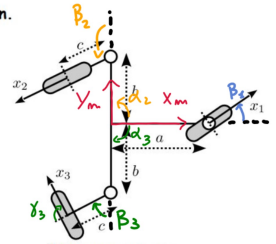
NOTE: if moving in a certain direction and/or with a certain rotation speed requires prior orientation of one or more wheels, the robot is not omnidirectional

NOTATIONS: Total number of wheels: $N_t = N_f + N_s + N_c$
Orientation variables of wheels: $\beta(t) = [\beta_s(t)^T \ \beta_c(t)^T]^T$
Rotation variables of the wheels: $\varphi(t) = [\varphi_f(t)^T \ \varphi_s(t)^T \ \varphi_c(t)^T]^T$
Posture coordinates: $\xi = [x, y, \theta]$

Configuration vector: $q = \begin{bmatrix} \xi \\ \beta_s \\ \beta_c \\ \varphi \end{bmatrix}$, defines the position and orientations of all rigid bodies of the robot

EXERCISE (1ST HOMEWORK)

Exercise 1 Parametrization.



Robot to parametrize

Give the parameter table and the configuration vector.

wheel	L	α	d	β	γ	φ	ψ
1s	a	0	0	β_{1s}	0	φ_{1s}	β_{1s}
2c	b	$\frac{\pi}{2}$	c	β_{2c}	0	φ_{2c}	$\beta_{2s}+\frac{\pi}{2}$
3g	b	$-\frac{\pi}{2}$	c	β_{3f}	$-\frac{\pi}{2}$	φ_{3f}	$\beta_{3f}-\pi$

$$q=\left[x, y, \theta, \beta_1, \beta_2, \beta_3, \varphi_1, \varphi_2, \varphi_3\right]^T$$

Exercise 2. Parametrization

A robot is given by the following table of parameters.

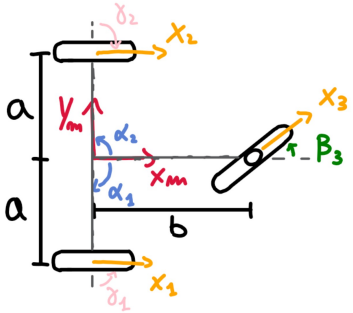
W	L	α	d	β	γ	φ	ψ
1	a	$-\pi/2$	0	0	$\pi/2$	φ_1	0
2	a	$\pi/2$	0	0	$-\pi/2$	φ_2	0
3	b	0	0	β_3	0	φ_3	β_3

Give the configuration vector.

Give the schematic representation of the robot. The figure should show vectors x_1, x_2 and x_3, the dimensions a and b, and the angle β_3. Make sure β_3 is indicated by a circular arrow with the origin at the zero position of the angle. Clearly indicate frame R_m on the figure. The type of wheel has been voluntarily omitted in the table: you can infer it from the table information.

$$q=\left[x, y, \theta, \beta_3, \varphi_1, \varphi_2, \varphi_3\right]^T$$

wheel 1-2: fixed wheel 3: steerable



CONSTRAINTS FOR ALL WHEELS IN MATRIX FORM

The set of N_T non slipping constraint ($v_t=0$) can be put in matrix form

$$J_1(\beta_s, \beta_c)\cdot {}^m\mathcal{R}_0(\theta)\cdot \dot{\xi} + J_2\cdot \dot{\varphi} = 0$$

This equation takes into account that there is no term in $\dot{\beta}$ and no term β for fixed wheels

Instead for the lateral skidding constraints ($v_n=0$) the form is:

$$C_1(\beta_s, \beta_c)\cdot {}^m\mathcal{R}_0(\theta)\,\dot{\xi} + C_2\cdot \dot{\beta}_c = 0$$

There is no term in $\dot{\varphi}$ in this equation

The composition of the matrix for the three kinds of wheels is:

$$J_1(\beta_s, \beta_c)=\underbrace{\begin{bmatrix} J_{1f} \\ J_{1s}(\beta_s) \\ J_{1c}(\beta_c)\end{bmatrix}}_{3} \qquad J_2 = -r\, I_{N_t \times N_t}$$

$$C_1(\beta_s, \beta_c)=\begin{bmatrix} C_{1f} \\ C_{1s}(\beta_s) \\ C_{1c}(\beta_c)\end{bmatrix} \qquad \underbrace{C_2=\begin{bmatrix} 0 \\ 0 \\ C_{2c}\end{bmatrix}}_{N} \qquad \text{where } C_{2c}=\text{diag}(d_1, \dots, d_{N_c})$$

To satisfy the NON slipping constraints: $\dot{\varphi} = \frac{1}{r}\left[J_1(\beta_s, \beta_d) \cdot {}^m\Omega_o(\theta) \cdot \dot{\xi}\right]$

This means that Non slipping will never cause any loss of mobility given that for any $\dot{\xi}$ it's possible to determine $\dot{\varphi}$

To satisfy the NON SNidding for CASTOR WHEELS it's possible to write:

$$C_{1c}(\beta_c) \cdot {}^m\Omega_o(\theta) \cdot \dot{\xi} + C_{2c} \cdot \dot{\beta}_c = 0 \rightarrow \dot{\beta}_c = -C_{2c}^{-1} \cdot C_{1c}(\beta_c) \cdot {}^m\Omega_o(\theta)\,\dot{\xi}$$

with $C_{2c}^{-1} = \text{diag}\left(\frac{1}{d_1}, \dots, \frac{1}{d_{Nc}}\right)$ The NOM SKidding for castor wheels will not cause any loss of mobility

For other wheels: $\begin{cases} C_{1f} \cdot {}^m\Omega_o(\theta) \cdot \dot{\xi} = 0 \\ C_{1s}(\beta_s) \cdot {}^m\Omega_o(\theta) \cdot \dot{\xi} = 0 \end{cases}$ They meant that ${}^m\Omega_o(\theta)\dot{\xi}$ must belong to the kernel of C_1^*

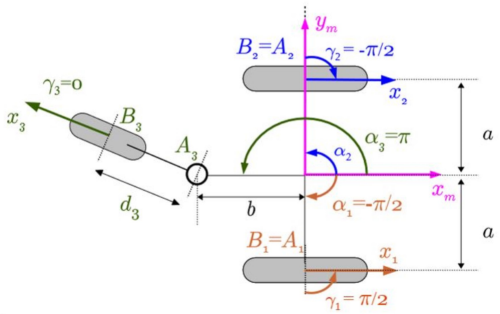
${}^m\Omega_o(\theta)\,\dot{\xi} \in \text{Ker}\left(C_1^*(\beta_s)\right)$ with: $C_1^*(\beta_s) = \begin{bmatrix} C_{1f} \\ C_{1s}(\beta_s) \end{bmatrix}$

Hence, the degree of mobility of the robot is:

$$\delta_m = \dim\left[\text{Ker}\left(C_1^*(\beta_s)\right)\right] = 3 - \text{rank}\left(C_1^*(\beta_s)\right) \leq 3$$

NOTE: if $\delta_m = 0$ then the robot is useless

EXAMPLE:



wheel	L	α	d	β	γ	φ	ψ
1f	a	$-\frac{\pi}{2}$	0	0	$\frac{\pi}{2}$	φ_{1f}	0
2f	a	$\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$	φ_{2f}	0
3c	b	π	d_3	β_{3c}	0	φ_{3c}	$\beta_3+\pi$

$$q = \left[x, y, \theta, \beta_{3c}, \varphi_{1f}, \varphi_{2f}, \varphi_{3f}\right]^T$$

From memento:

$$J_1 = \begin{bmatrix} 1 & 0 & a \\ 1 & 0 & a \\ -\cos\beta_{3c} & -\sin\beta_{3c} & b\sin\beta_{3c} \end{bmatrix} \qquad C_1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ \sin\beta_{3c} & -\cos\beta_{3c} & d+\cos\beta_{3c} \end{bmatrix}$$

$$J_2 = \begin{bmatrix} -r & 0 & 0 \\ 0 & -r & 0 \\ 0 & 0 & -r \end{bmatrix} \qquad C_2 = \begin{bmatrix} 0 \\ 0 \\ d \end{bmatrix} \qquad C_1^*(\beta_s) = \begin{bmatrix} C_{1f} \\ C_{1s}(\beta_s) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\delta_m = 3 - \text{rank}\left(C_1^*(\beta_s)\right) = 2$$

NOTE: $C_1^*\,{}^m\Omega_o\,\dot{\xi} = C_1^*\,{}^m\dot{\xi} = 0$, this means ${}^m\dot{y} = 0$. So, the velocity vector of any point of the axle is perpendicular to the axle

DEGREE OF STEERABILITY

It is denoted as δ_s and defined as $\delta_s = \text{rank}(C_{1s})$. It represents the number of steerable wheels which can be oriented independently in such a way that the ICR of the robot exists

By considering the ICR we know that: $0 \le \delta_s \le 2$
If $\delta_s = 2$ then there are no fixed wheels, while $\delta_s = 0$ implies $N_s = 0$

In a similar way, defining $\delta_f = \text{rank}(C_{1f})$. $\delta_f = 0$ iff $N_f = 0$.
$\delta_f = 2$ reduces possible motions to a rotation around the fixed ICR

Hence: $0 \le \delta_f \le 1$. We have to satisfy $0 \le \delta_s + \delta_f \le 2$, $2 \le \delta_m + \delta_s \le 3$

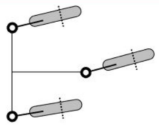
ROBOT TYPES

Robots are classified according to their degree of mobility and their degree of steerability. The possible cases are:

δ_m	3	2	2	1	1
δ_s	0	0	1	2	1

Type (3,0) robots

$\delta_m=3$, $\delta_s=0$. $N_f=N_s=0$, $N_c=3$.

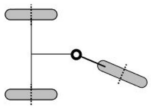


It is called **omnidirectional** robot. It can move in any direction of the plane while simultaneously rotating, starting with any configuration of the wheels.

Type (2,0) robots

$\delta_m=2$, $\delta_s=0$. $N_f=2$, $N_s=0$, $N_c=1$.

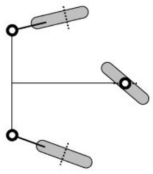
Called **unicycle** or **differential** robot. These terms will be justified later.



Type (2,1) robots

$\delta_m=2$, $\delta_s=1$. $N_f=0$, $N_s=1$, $N_c=2$.

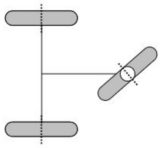
If $N_s>1$, the steerable wheels will be controlled in such a way that $rank(C_{1s})=1$.



Type (1,1) robots

$\delta_m=1$, $\delta_s=1$. $N_f=2$, $N_s=1$, $N_c=0$.

The derivative of the posture belongs to a one-dimensional space parameterized by the angle of the steering wheel.



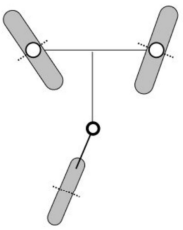
It is the classical model used to represent a car, the two front steering wheels being considered as a single central steering wheel.

Type (1,2) robots

$\delta_m=1$, $\delta_s=2$. $N_f=0$, $N_s=2$, $N_c=1$.

The twist of the platform belongs to a one-dimensional space parameterized by the orientation angles β of the steerable wheels.

Reason: instantaneously, the motion of the robot platform is constrained to be a rotation around the ICR, the location of which is defined by the steerable wheels orientation angles β_1 and β_2.



DEGREE OF MANEUVRABILITY

It is defined as $\delta_M = \delta_m + \delta_s$. Its value is 2 or 3 and corresponds to the number of degrees of freedom which can be manipulated by the control

POSTURE KINEMATIC MODEL

A standard equation for a non linear system is of the form: $\dot{z} = f(z, u)$
In such equation, u is called the input and z the state.
We would like u to contain the speed of the motors (or some combination) while for the state z we need the posture (x, y, θ) of the robot

In the posture kinematic model, we are interested in expressing the twist of the platform in R_0, as a function of the speed (and orientation) of wheels

Recalling that only the non sliding constraints of fixed and steerable wheels can result in limiting the mobility of the platform, using the equation we can say: ${}^{m}\Omega_0(\theta)\,\dot{\xi} \in \ker\left(C_1^*(\beta_s)\right)$

${}^{m}\Omega_0(\theta)\,\dot{\xi}$ can be decomposed on a base of $\ker\left(C_1^*\right) \rightarrow {}^{m}\Omega_0(\theta)\,\dot{\xi} = \Sigma(\beta_s)\,u_m$

where the columns of Σ form a base of the kernel of C_1^*, of dimension δ_m, and u_m has δ_m components. Multiplying left by ${}^{0}\Omega_m$ we obtain:

$${}^{0}\dot{\xi} = {}^{0}\Omega_m(\theta)\cdot\Sigma(\beta_s)\cdot u_m$$

If the robot has no steering wheel: ${}^{0}\dot{\xi} = {}^{0}\Omega_m(\theta)\cdot\Sigma\cdot u_m$
Otherwise, if the robot has steerable wheels, the posture kinematic model is:

$$\begin{cases} {}^{0}\dot{\xi} = {}^{0}\Omega_m(\theta)\,\Sigma(\beta_s)\,u_m \\ \dot{\beta}_s = u_s \end{cases}$$

This equation corresponds to the form $\dot{z} = B(z)\cdot u$ with:

$$B(z) = \begin{bmatrix} {}^{0}\Omega_m(\theta)\cdot\Sigma(\beta_s) & 0 \\ 0 & I \end{bmatrix} \quad u = \begin{bmatrix} u_m \\ u_s \end{bmatrix} \quad z = \begin{bmatrix} \xi \\ \beta_s \end{bmatrix}$$

This relation is the space model of the system. ξ and β_s are the state variables, u_s and u_m the control inputs

CONFIGURATION KINEMATIC MODEL

It is the expression of $\dot{q}$ as a function of the inputs u_s and u_m
We need to establish the expression of $\dot{\beta}_c$ for castor wheels and of $\dot{\varphi}$

They can be derived by:

$$J_1(\beta_s,\beta_c)\cdot{}^{m}\Omega_0(\theta)\cdot\dot{\xi} + J_2\cdot\dot{\varphi} = 0$$
$$C_{1c}(\beta_c)\cdot{}^{m}\Omega_0(\theta)\cdot\dot{\xi} + C_{2c}\cdot\dot{\beta}_c = 0$$

So,

$$\dot{\beta}_c = -C_{2c}^{-1}\cdot C_{1c}(\beta_c)\cdot{}^{m}\Omega_0(\theta)\cdot\dot{\xi}$$
$$\dot{\varphi} = \frac{1}{r}\left[J_1(\beta_s,\beta_c)\cdot{}^{m}\Omega_0(\theta)\cdot\dot{\xi}\right]$$

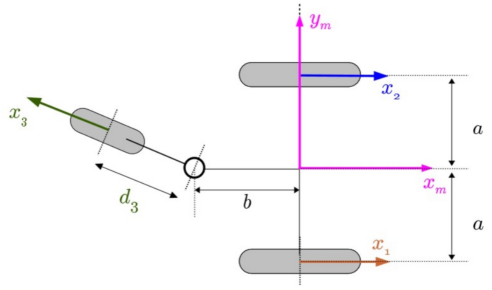
By putting together in a block matrix the expressions of $\dot{\xi}, \dot{\beta}_s, \dot{\beta}_c$ and $\dot{\varphi}$

$$\dot{q} = S(q)\cdot u \quad \text{with,} \quad S(q) = \begin{bmatrix} {}^{0}\Omega_m(\theta)\cdot\Sigma(\beta_s) & 0 \\ 0 & I \\ D(\beta_c)\cdot\Sigma(\beta_s) & 0 \\ E(\beta_s,\beta_c)\cdot\Sigma(\beta_s) & 0 \end{bmatrix} \quad \text{and} \quad u = \begin{bmatrix} u_m \\ u_s \end{bmatrix}$$

(3×3, $3\times\delta_m$, $N_c\times3$, $N_b\times3$)

Having defined $\begin{cases} D(\beta_c) = -C_{2c}^{-1}\cdot C_{1c}(\beta_c) \\ E(\beta_s,\beta_c) = -J_2^{-1}\cdot J_1(\beta_s,\beta_c) \end{cases}$

EXAMPLE: POSTURE KINEMATIC MODEL AND CONFIGURATION KINEMATIC MODEL



wheel	L	α	d	β	γ	φ	ψ
1f	a	$-\pi/2$	0	0	$\pi/2$	φ_{1f}	0
2f	a	$\pi/2$	0	0	$-\pi/2$	φ_{2f}	0
3c	b	π	d_{3c}	β_c	0	φ_{3c}	$\beta_{3c}+\pi$

$$q=\begin{bmatrix} x, & y, & \theta, & \beta_{3c}, & \varphi_{1f}, & \varphi_{2f}, & \varphi_{3c} \end{bmatrix}^T$$

Matrices for constraints:

$$J_1=\begin{bmatrix} 1 & 0 & a \\ 1 & 0 & -a \\ -\cos(\beta_{3c}) & -\sin(\beta_{3c}) & b\sin(\beta_{3c}) \end{bmatrix} \qquad J_2=\begin{bmatrix} -r & 0 & 0 \\ 0 & -r & 0 \\ 0 & 0 & -r \end{bmatrix}$$

$$C_1=\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ \sin(\beta_{3c}) & -\cos(\beta_{3c}) & d_3+L\cos(\beta_{3c}) \end{bmatrix} \qquad C_2=\begin{bmatrix} 0 \\ 0 \\ d_{3c} \end{bmatrix}$$

Taking the sub-matrix C_1^*:

$$C_1^*(\beta_s)=\begin{bmatrix} C_{1f} \\ C_{1s}(\beta_s) \end{bmatrix}=\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

The degree of mobility δ_m is:

$$\delta_m=3-\text{rank}\left(C_1^*(\beta_s)\right)=2$$

The posture kinematic model is given by:

$$C_{1f}\,{}^m\Omega_0\,\dot{\xi}=C_{1f}\,{}^m\dot{\xi}=0 \longrightarrow {}^m\dot{y}=0$$

From this is possible to find a base: $\Sigma=\begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$

$${}^m\Omega_0\cdot\dot{\xi}=\Sigma(\beta_s)\,u_m \longrightarrow u_m=\begin{bmatrix} {}^m\dot{x} \\ \dot{\theta} \end{bmatrix}=\begin{bmatrix} v \\ w \end{bmatrix}$$

To compute the configuration kinematic model it is necessary to find the matrix $S(q)$

$$D(\beta_c)=-C_{2c}^{-1}\cdot C_{1c}(\beta_c)=-\frac{1}{d_{3c}}\begin{bmatrix} -\sin(\beta_{3c}) & \cos(\beta_{3c}) & d+\cos(\beta_{3c}) \end{bmatrix}$$

$$E(\beta_s,\beta_c)=-J_2^{-1}\cdot J_1(\beta_s,\beta_c)=\frac{1}{r^3}\begin{bmatrix} r^2 & 0 & 0 \\ 0 & r^2 & 0 \\ 0 & 0 & r^2 \end{bmatrix}\begin{bmatrix} 1 & 0 & a \\ 1 & 0 & -a \\ -\cos(\beta_{3c}) & -\sin(\beta_{3c}) & b\sin(\beta_{3c}) \end{bmatrix}$$

$$=\begin{bmatrix} 1/r & 0 & a/r \\ 1/r & 0 & -a/r \\ -\frac{1}{r}\cos(\beta_{3c}) & -\frac{1}{r}\sin(\beta_{3c}) & \frac{b}{r}\sin(\beta_{3c}) \end{bmatrix}$$

$$\dot{q} = \mathbf{S}(q) \cdot u$$

$$\mathbf{S}(q) = \begin{bmatrix} {}^0\boldsymbol{\Omega}_m(\theta) \cdot \Sigma(\beta_s) & 0 \\ 0 & \mathbf{I} \\ \mathbf{D}(\beta_c) \cdot \Sigma(\beta_s) & 0 \\ \mathbf{E}(\beta_s, \beta_c) \cdot \Sigma(\beta_s) & 0 \end{bmatrix}$$

$$\begin{bmatrix} \dot{x}_0 \\ \dot{y}_0 \\ \dot{\theta} \\ \dot{\beta}_{3c} \\ \dot{\varphi}_{1f} \\ \dot{\varphi}_{2f} \\ \dot{\varphi}_{3f} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 1 & 0 \\ -\frac{\sin(\beta_{3c})}{d} & -\frac{d + b\cos(\beta_{3c})}{d} \\ 1/r & a/r \\ 1/r & -a/r \\ -\frac{\cos(\beta_{3c})}{r} & \frac{b\sin(\beta_{3c})}{r} \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix}$$

MOTORIZATION OF THE WHEELS

We want to find the relation between u and the motors' parameters
Recall that all orientation joints of steerable wheels must be motorized
Moreover, it is necessary to motorize N_m other joints, with $N_m \geq \delta_m$, to generate u_m. Finally, the aim is to find a relation between the motor speeds and u_m, finding the right joints to motorize.
From the configuration model:

$$\begin{bmatrix} \dot{\xi} \\ \dot{\beta}_s \\ \dot{\beta}_c \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} {}^0\Omega_m(\theta) \cdot \Sigma(\beta_s) & 0 \\ 0 & I \\ D(\beta_c) \cdot \Sigma(\beta_s) & 0 \\ E(\beta_s, \beta_c) \cdot \Sigma(\beta_s) & 0 \end{bmatrix} \cdot \begin{bmatrix} u_m \\ u_s \end{bmatrix}$$

From these equations, we extract the lines corresponding to joints which can potentially be equipped with a motor to produce u_m.

$$\begin{bmatrix} \dot{\beta}_c \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} D(\beta_c) \cdot \Sigma(\beta_s) \\ E(\beta_s, \beta_c) \cdot \Sigma(\beta_s) \end{bmatrix} \cdot u_m = \underbrace{F(\beta_s, \beta_c)}_{\text{MOTORIZATION MATRIX}} \cdot u_m$$

OSS! The orientation of steerable wheels it's discarded, because are used for u_s

The system is over-determined and so we want to extract δ_m independent lines of $F(\beta_s, \beta_c)$ so that:

1) The corresponding sub-system can be inverted
2) The inverse sub-system determines u_m as a function of the joint speeds corresponding to the selected lines
3) Equipping those joints with motors solves the motorization of the robot

EXERCISE: motorization of robot type (2, 0) previous scheme

From the configuration model is possible to extract $F(\beta_s, \beta_c)$

$$F(\beta_s, \beta_c) = \begin{bmatrix} D(\beta_c) \cdot \Sigma(\beta_s) \\ E(\beta_s, \beta_c) \cdot \Sigma(\beta_s) \end{bmatrix}$$

$$D(\beta_c) \cdot \Sigma(\beta_s) = -\frac{1}{d} \begin{bmatrix} \sin(\beta_{3c}) & d + b\cos(\beta_{3c}) \end{bmatrix}$$

$$E(\beta_s, \beta_c) \cdot \Sigma(\beta_s) = \frac{1}{r} \begin{bmatrix} 1 & a \\ 1 & -a \\ -\cos(\beta_{3c}) & b\sin(\beta_{3c}) \end{bmatrix}$$

$$\rightarrow F(\beta_s, \beta_c) = \begin{bmatrix} -\frac{\sin(\beta_{3c})}{d} & -\frac{d + b\cos(\beta_{3c})}{d} \\ 1/r & a/r \\ 1/r & -a/r \\ -\frac{\cos(\beta_{3c})}{r} & \frac{b\sin(\beta_{3c})}{r} \end{bmatrix}$$

We have to select two independent lines of $F(\beta_s, \beta_c)$, the corresponding joints will be motorised. The simplest solution is to select lines 2 and 3 (both fixed wheels are motorised). So:

$$u_m = \begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} r/2 & r/2 \\ r/2a & -r/2a \end{bmatrix} \begin{bmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \end{bmatrix}$$

Remarks

- Now we have the relation between u_m and the speed of the motors. This is when the work is complete.
- When the motorization involves orientable wheels (see later), the relation will involve also the position of orientation motors, not just their speed.
- Practical point: if a wheel orientation is involved, you will need to equip the corresponding joint with an **absolute** encoder and identify the encoder offset.
- Preferably use an **absolute** encoder, as opposed to an incremental encoder, to avoid having to calibrate the joint at startup (move the joint until the encoder gets to its zero position).

Another solution can be done by fully motorizing the castor wheel. In that case:

$$\det(F_{4u}) = -\frac{1}{dr}(b + d\cos\beta_{3c})$$

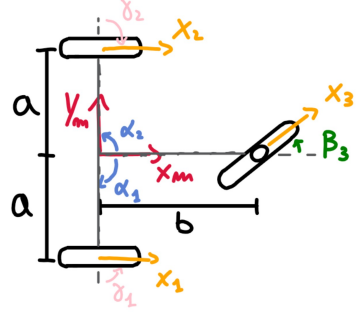
but when the determinant will be zero, we will lose one degree of mobility -> singularity

Similar thing will happen selecting one fixed wheel and the castor orientation ($\dot{\beta}_{3c}$) or castor drive ($\dot{\rho}_{3c}$)

Exercise (2nd homework)

W	L	α	d	β	γ	φ	ψ
$1f$	a	$-\pi/2$	0	0	$\pi/2$	φ_{1f}	0
$2f$	a	$\pi/2$	0	0	$-\pi/2$	φ_{2f}	0
$3s$	b	0	0	β_{3s}	0	φ_{3s}	β_{3s}

1) Schematic of the robot:



2) The configuration vector is:

$$q = \begin{bmatrix} x & y & \theta & \beta_{3s} & \varphi_{1f} & \varphi_{2f} & \varphi_{3s} \end{bmatrix}^T$$

3) The matrices are:

$$J_1 = \begin{bmatrix} 1 & 0 & a \\ 1 & 0 & -a \\ \cos\beta_{3s} & \sin\beta_{3s} & b\sin\beta_{3s} \end{bmatrix} \qquad J_2 = \begin{bmatrix} -r & 0 & 0 \\ 0 & -r & 0 \\ 0 & 0 & -r \end{bmatrix}$$

$$C_1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ -\sin\beta_{3s} & \cos\beta_{3s} & b\cos\beta_{3s} \end{bmatrix} \qquad C_2 = \begin{bmatrix} \\ \end{bmatrix}$$

C_2 empty

4) The matrix $C_1^*(\beta_s)$ is:

$$C_1^*(\beta_s) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ -\sin\beta_{3s} & \cos\beta_{3s} & b\cos\beta_{3s} \end{bmatrix}$$

The determinant of this matrix is: $\det(C_1^*) = 0$. Moreover, the two sub-matrices made using the last two rows cannot have both determinant equals to zero. So, the rank is 2.
In conclusion, the degree of mobility is $\delta_m = 3 - \text{rank}\left(C_1^*(\beta_s)\right) = 1$

5) The kernel of the matrix C_1^* is given by: $C_1^* \dot{\xi} = 0$
The linear system is:

$$\begin{cases} \dot{y} = 0 \\ -\sin\beta_{3s}\,\dot{x} + b\cos\beta_{3s}\,\dot{\theta} = 0 \end{cases} \qquad \dot{\theta} = \frac{1}{b}\tan\beta_{3s}\,\dot{x}$$

A basis of the kernel is given by: $\sum = \begin{bmatrix} 1 \\ 0 \\ \frac{1}{b}\tan(\beta_{3s}) \end{bmatrix}$ and as expected the dimension is 1 ($\delta_m = 1$)

6) The configuration kinematic model is found as: $\dot{q} = S(q)\,u$

$$S(q) = \begin{bmatrix} {}^0\Omega_m(\theta) \cdot \sum(\beta_s) & 0 \\ 0 & I \\ D(\beta_c) \cdot \sum(\beta_s) & 0 \\ E(\beta_s, \beta_c) \cdot \sum(\beta_s) & 0 \end{bmatrix} \qquad D(\beta_c) = 0$$

$$E(\beta_s,\beta_c)=\frac{1}{r}I\cdot\begin{bmatrix}1 & 0 & a\\ 1 & 0 & -a\\ \cos\beta_{3s} & \sin\beta_{3s} & b\sin\beta_{3s}\end{bmatrix}=\begin{bmatrix}1/r & 0 & a/r\\ 1/r & 0 & -a/r\\ \frac{\cos(\beta_{3s})}{r} & \frac{\sin(\beta_{3s})}{r} & \frac{b\sin(\beta_{3s})}{r}\end{bmatrix}$$

$$\Sigma(\beta_s)=\begin{bmatrix}1\\ 0\\ \frac{\tan(\beta_{3s})}{b}\end{bmatrix}\qquad S(q)=\begin{bmatrix}\cos\theta & 0\\ \sin\theta & 0\\ \tan(\beta_{3s})/b & 0\\ 0 & 1\\ \frac{1}{r}+\frac{a}{rb}\tan(\beta_{3s}) & 0\\ \frac{1}{r}-\frac{a}{rb}\tan(\beta_{3s}) & 0\\ \frac{\cos(\beta_{3s})}{r}+\frac{\sin^2(\beta_{3s})}{r\cos(\beta_{3s})} & 0\end{bmatrix}\qquad u=\begin{bmatrix}\dot{x}\\ \dot{\beta}_{3s}\end{bmatrix}$$

The configuration kinematic model is: $\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\\ \dot{\beta}_{3s}\\ \dot{\varphi}_{1f}\\ \dot{\varphi}_{2f}\\ \dot{\varphi}_{3s}\end{bmatrix}=S(q)\begin{bmatrix}\dot{x}\\ \dot{\beta}_{3s}\end{bmatrix}$

7) Surely, it is necessary to motorize the steerable wheel.
For the others motorization we have to compute $F(\beta_s,\beta_c)$

$$F(\beta_s,\beta_c)=\begin{bmatrix}\frac{1}{r}+\frac{a}{rb}\tan(\beta_{3s})\\ \frac{1}{r}-\frac{a}{rb}\tan(\beta_{3s})\\ \frac{\cos(\beta_{3s})}{r}+\frac{\sin^2(\beta_{3s})}{r\cos(\beta_{3s})}\end{bmatrix}$$

Since $\delta_m=1$ I have to select only 1 line

$$\frac{1}{r}+\frac{a}{rb}\tan(\beta_{3s})=0\ \rightarrow\ \beta_{3s}=\arctan\left(-\frac{b}{a}\right)\qquad \beta_{3s}\neq\pm\frac{\pi}{2}$$

$$\frac{1}{r}-\frac{a}{rb}\tan(\beta_{3s})=0\ \rightarrow\ \beta_{3s}=\arctan\left(\frac{b}{a}\right)\qquad \beta_{3s}\neq\pm\frac{\pi}{2}$$

$$\frac{\cos(\beta_{3s})}{r}+\frac{\sin^2(\beta_{3s})}{r\cos(\beta_{3s})}=0\ \rightarrow\ \frac{1}{\cos(\beta_{3s})}\qquad \beta_{3s}\neq\pm\frac{\pi}{2}$$

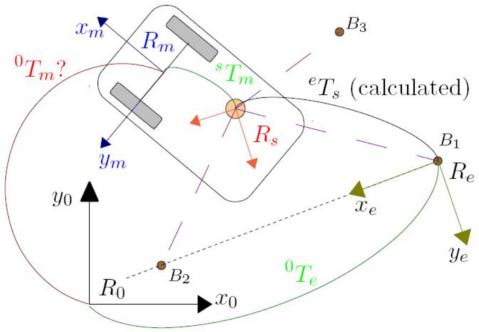
The three choices have the same singularity, so I can choose to motorize the first fixed wheel. In conclusion, the motorization is: $\dot{\varphi}_{1f}$ and β_{3s}

Mobile Robots Localization

Definitions

- Pose: it is the position and orientation in 3D space of a rigid body
- Localization: is the process of determining some or all variables of pose

The classical frames



- R_m: Robot frame, usually attached to the chassis
- R_0: Absolute frame, in which the user wants to know the pose of the vehicle. (${}^0T_m$ is unknown)
- R_e: Environment frame is attached to the known reference points of the environment, which the localization system uses to perform measurements
- R_s: sensor frame, attached to the localization sensor(s)

In general, the sensor determines its own location ${}^eT_s$ with respect to the environment frame R_e. So, ${}^0T_m = {}^0T_e \cdot {}^eT_s \cdot {}^sT_m$

Classes of localization

- Static localization: the vehicle is motionless while sensor data necessary for localization is read and processed (all measurements correspond to the same pose)
- Dynamic localization: the vehicle moves while the sensor data is read and processed
- Quasi-static localization: the vehicle moves while sensor data is read and processed but the motion is negligible with respect to the required precision
- Relative localization: determines the pose of the vehicle with respect to its initial location. In this case are used proprioceptive sensors which don't need any known reference in the environment to perform measurements
- Absolute localization: determines the pose of the vehicle with respect to a frame attached to the environment. Uses of exteroceptive sensors

Relative localization

The characteristics of this algorithms are:

1) It does not require any reference to the environment and hence is always available
2) It is possible to freely choose the sampling frequency (typically high) and the period between calculations can be fixed
3) Processing data time is usually short
4) Precision degrades with time and travelled distance (noise is introduced in integration)

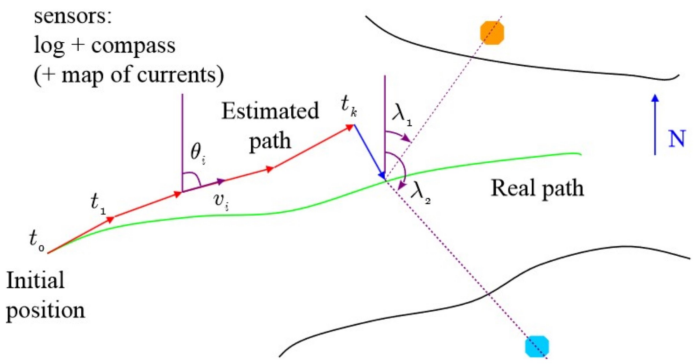
Absolute localization

1) It is performed with respect to a fixed frame of the environment
2) Frequency of sensor readings is often low and data are available asynchronously
3) Processing time can create delays
4) The precision varies with the number of beacons and their position

As we have seen the two algorithms are complementary

OSS: Localization errors increase as a relative localization is used this phenomenon is called drift

EXAMPLE: THE OLD SAILOR METHOD



log: is a sensor used to measure the relative speed of the boat with respect to the water

Compass: measures the heading of the boat with respect to magnetic north

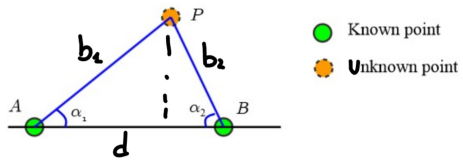
HYBRID LOCALIZATION

```
begin
    Calculate initial location
    loop (fast)
        Read internal sensors (e.g. Encoders)
        Calculate relative localization
        if external sensor data available then
            Update estimated location
        endif
    endloop
end
```

G. Garcia – École Centrale de Nantes

It is used to associate external and internal sensors. In this way is possible to retain advantages of absolute and relative localizations

ABSOLUTE LOCALIZATION



We want to determine the 2D position of a point using two points as reference. α_1 and α_2 are found using the goniometer. Also the baseline is known

From the trigonometry is possible to obtain:

$$b_1 = \frac{d \sin\alpha_2}{\sin(\alpha_1+\alpha_2)} \qquad b_2 = \frac{d \sin\alpha_1}{\sin(\alpha_1+\alpha_2)} \qquad \begin{bmatrix} x \\ y \end{bmatrix} = f(d, \alpha_1, \alpha_2)$$

$$x = \frac{d \cos\alpha_2 \sin\alpha_2}{\sin(\alpha_1+\alpha_2)} \qquad y = \frac{d \sin\alpha_1 \sin\alpha_2}{\sin(\alpha_1+\alpha_2)}$$

Now I want to compute the accuracy of this method. To do so is possible to find the covariance matrix of our input vector

$M = [d, \alpha_1, \alpha_2]$ MEASUREMENT VECTOR

$$C_M = \begin{bmatrix} \sigma_D^2 & 0 & 0 \\ 0 & \sigma_\alpha^2 & 0 \\ 0 & 0 & \sigma_\alpha^2 \end{bmatrix}$$

$\alpha_1 = \alpha_2 = \alpha$ because the instrument is the same

OSS: There is no correlation between d, α_1, α_2

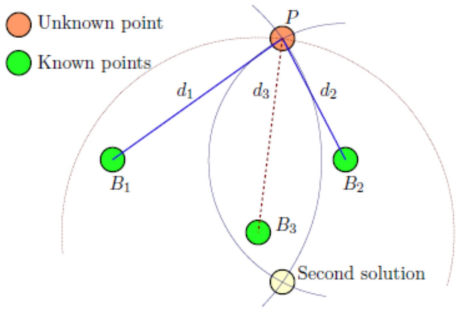
REMARK: Let x and y be two random vectors of dimensions $n \times 1$ and $p \times 1$, for which there is the relation $Y = Kx$. In that case the covariance matrices satisfy: $P_y = K P_x K^T$

So, it is possible to find the covariance matrix of the point using the Jacobian of f with respect to M.

$$P = [x, y]^T \qquad J = \begin{bmatrix} \frac{\partial x}{\partial d} & \frac{\partial x}{\partial \alpha_1} & \frac{\partial x}{\partial \alpha_2} \\ \frac{\partial y}{\partial d} & \frac{\partial y}{\partial \alpha_1} & \frac{\partial y}{\partial \alpha_2} \end{bmatrix} \qquad C_P = J C_M J^T$$

In this way is possible to calculate the covariance matrix of the errors on output variables as a function of the covariance matrix of input variables

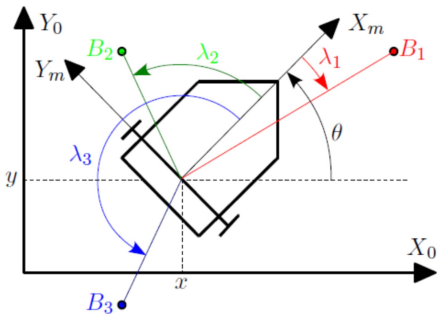
LOCALIZATION USING DISTANCES TO KNOWN POINTS



The circles are centered on A_1, A_2, A_3 With two measurements, two solutions are found. In that case is necessary to to know which solution is correct, by prior knowledge, or to add another measure

$$\begin{cases} (x-x_1)^2 + (y-y_1)^2 = d_1^2 \\ (x-x_2)^2 + (y-y_2)^2 = d_2^2 \end{cases} \qquad (x-x_3)+(y-y_3)^2 = d_3^2$$

STATIC 2D LOCALIZATION USING AZIMUTH ANGLES



The sensors are on point M and it is necessary to compute the posture
Beacon coordinates in robot frame R_m:

$$\begin{cases} {}^m x_i = b_i \cos \lambda_i \\ {}^m y_i = b_i \sin \lambda_i \end{cases}$$

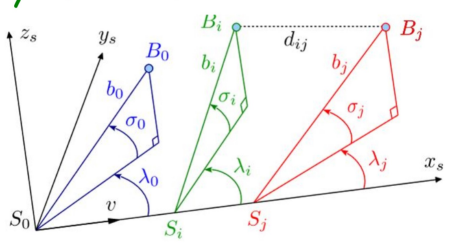
Then, is possible to use the inter-beacon distances that is INVARIANT

$$d_{ij}^2 = b_i^2 + b_j^2 - 2 b_i b_j \cos(\lambda_i - \lambda_j)$$

So, the steps to follow are: find b_1, b_2, b_3. Deduce beacon coordinates in R_m. Calculate ${}^m T_b$ and then ${}^o T_m$

OSS! a similar solution can be found for 3D localization

DYNAMIC LOCALIZATION



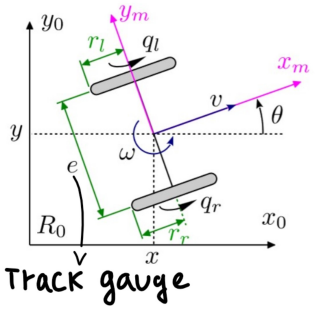
The simplest hypotheses

- Locally:
  - The rolling surface is approximated by a plane.
  - The motion is a straight line at constant speed.
  - The speed is parallel to axis x_m.
- Consequences:
  - Seven unknowns: position + attitude + speed.
  - The hypotheses must be satisfied during the time needed to detect a sufficient number of beacons to determine the seven unknowns.

How to solve

- It is necessary to detect four beacons (one of three beacons seen twice at different locations)
- Similar to previous cases:
  - Write beacon coordinates in sensor frame corresponding to time t_o (first beacon detection).
  - Write d_{ij}^2 (there are more than necessary).
  - Solve in the least squares sense.
  - Determine position and attitude at time t_o plus speed.

Relative localization method: odometry



Track gauge

Under pure rolling assumption (see kinematic modelling course):

$$\begin{cases} v = (r_r \dot{q}_r + r_l \dot{q}_l)/2 \\ \omega = (r_r \dot{q}_r - r_l \dot{q}_l)/e \\ \dot{x} = v\cos\theta \\ \dot{y} = v\sin\theta \\ \dot{\theta} = \omega \end{cases}$$

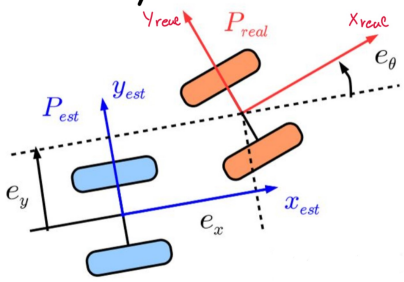
Slight adaptation: left and right wheel radii can be slightly different.

$$\begin{cases} \Delta D_k = (r_r \Delta q_{r,k} + r_l \Delta q_{l,k})/2 \\ \Delta\theta_k = (r_r \Delta q_{r,k} - r_l \Delta q_{l,k})/e \\ \theta_{k+1} = \theta_k + \Delta\theta_k \\ x_{k+1} = x_k + \Delta D_k \cos(\theta_k)\ \dot{x} \rightarrow \frac{\Delta D_k}{x_{k+1} - x_k} \\ y_{k+1} = y_k + \Delta D_k \sin(\theta_k) \end{cases}$$

$$\begin{cases} \Delta D_k = (r_r \Delta q_{r,k} + r_l \Delta q_{l,k})/2 \\ \Delta\theta_k = (r_r \Delta q_{r,k} - r_l \Delta q_{l,k})/e \\ \theta_{k+1} = \theta_k + \Delta\theta_k \\ x_{k+1} = x_k + \Delta D_k \cos(\theta_k + \Delta\theta_k/2) \\ y_{k+1} = y_k + \Delta D_k \sin(\theta_k + \Delta\theta_k/2) \end{cases}$$

→ This is an alternative form, more accurate in perfect conditions. Anyway, it will be more expensive in subsequent computation

When analysing the performance of localization algorithm, it is common to display the error between the ESTIMATED and REAL position



The POSTURE ERROR is formed by:

- e_x: Longitudinal error
- e_y: Lateral error
- e_θ: Heading error

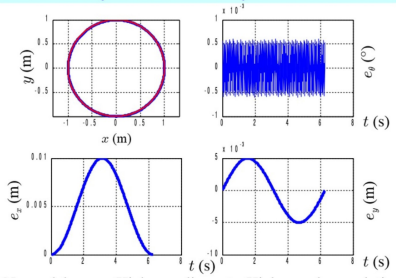
$${}^{est}P_e = \begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix} = {}^{est}\Omega_0(\theta_{est})\left[{}^{0}P_{real} - {}^{0}P_{est}\right]$$

$${}^{est}P_e = \begin{bmatrix} (x_{real} - x_{est})\cos\theta_{est} + (y_{real} - y_{est})\sin\theta_{est} \\ -(x_{real} - x_{est})\sin\theta_{est} + (y_{real} - y_{est})\cos\theta_{est} \\ \theta_{real} - \theta_{est} \end{bmatrix}$$

Analysis of simulation results

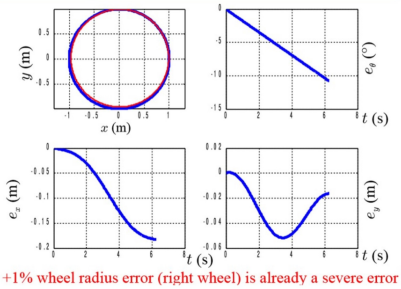
An important note is to plot also the errors and not only the real and estimated path. Otherwise only lateral errors are visible, not longitudinal ones

Odometry in a near-perfect world



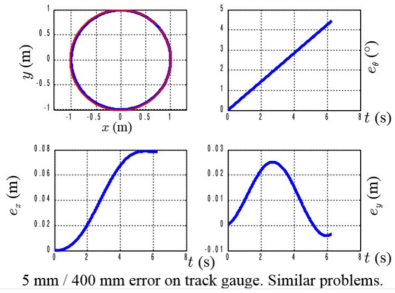
No model errors. High sampling rate. High encoder resolution

Odometry with a wheel radius error



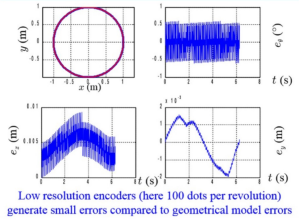
+1% wheel radius error (right wheel) is already a severe error !

Odometry with a track gauge error



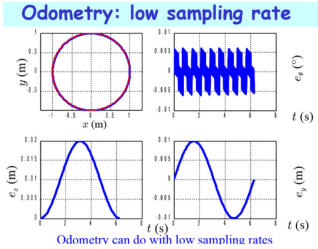
5 mm / 400 mm error on track gauge. Similar problems.

Odometry: low resolution encoders



Low resolution encoders (here 100 dots per revolution) generate small errors compared to geometrical model errors

Odometry: low sampling rate



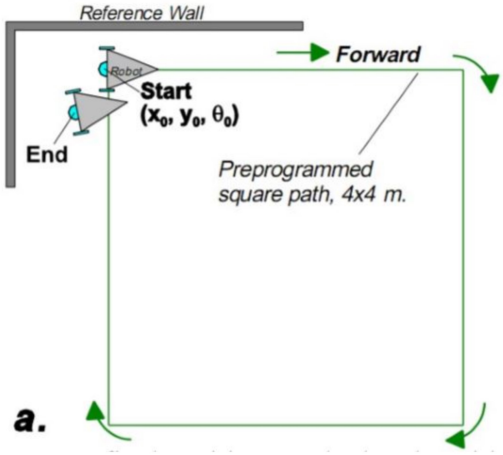
Odometry can do with low sampling rates

CLASSES OF ODOMETRY ERRORS

Errors can be classified into SYSTEMATIC ERRORS and NON-SYSTEMATIC ERRORS The first type includes: wheel radius and track gauge errors, misalignment of the wheels, finite resolution and sampling rate. The second type, instead, includes: uneven floor, wheel slippage. Typically, the problem is mainly given by systematic errors, since their effects accumulate over time

EVALUATION OF ODOMETRY PERFORMANCE

Permormances are usually characterised by measuring the difference between actual and estimated robot positions after a certain travelled distance. The performance is expressed as a percentage of the travelled distance and tests are repeated a certain number of times



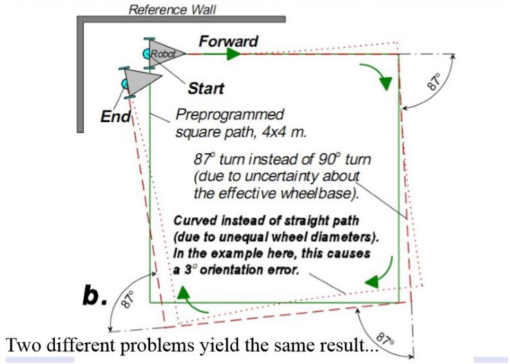
In this case the robot performs a nominally close trajectory, then the actual end position and the estimated one, calculated by odometry, are compared.

NOTE WELL: the end position must be compared with the actual one and not the desired one in order to not include controller errors

OSS: it is simple to compute the position using three robot points and the reference wall

OSS: performing square path is the bare minimum motion a robot should be able to do

Unidirectional square-path test

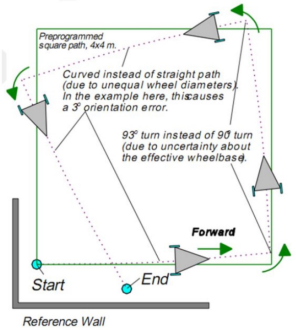


Two different problems yield the same result...

The unidirectional test is not enough due to the fact that two sources of error can yield the same error or two sources of error can cancel out yielding the illusion of a perfect result.

A simple way to avoid this, is to perform the path in two directions, rotating clockwise and counter-clockwise

BIDIRECTIONAL SQUARE-PATH

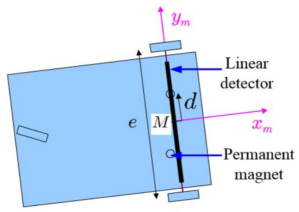


PRATICAL CONSIDERATIONS

- Orientation errors grow faster for vehicles with narrow track gauge
- Castor wheels can induce slippage when reversing direction, especially with an high load
- Odometry wheel should be thin and not compressible → Use auxilary wheels non loaded

EXAMPLE OF SEQUENTIAL USE OF RELATIVE AND ABSOLUTE LOCALIZATION

System description

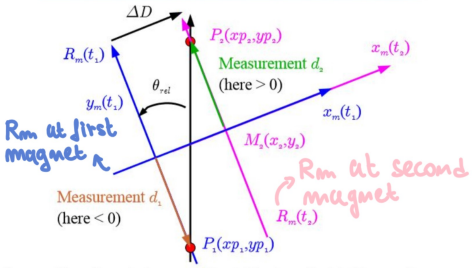


• Permanent magnets are inserted in the ground at known locations.
• The robot is equipped with a linear detector which measures the position of the block wrt the center of the detector, when crossed.

To implement the robot sensors can be used a set of solenoids or a set of reed switches

The principle of operation is simple: between two passages over magnetic stations the robot uses the wheel encoders for relative localization, when it crosses a magnetic station, it calculates its posture

Absolute localization equations



Assumption: the robot crosses the station in a straight line motion

Calling ΔD the travelled distance between t_1 and t_2 is possible to compute the posture at time t_2

$$\sin(\theta_{rel}) = \frac{\Delta D}{dist(P_2, P_1)}$$ and then the absolute distance is given by the relative orientation and the orientation P_1P_2

$$\theta = \arctan(y_{P_2} - y_{P_1};\ x_{P_2} - x_{P_1}) + \theta_{rel} - \frac{\pi}{2}$$

The coordinates of the point P_2 can be found as: $P_2 = (\underbrace{x_2 - d_2 \sin\theta}_{x_{P_2}};\ \underbrace{y_2 + d_2 \cos\theta}_{y_{P_2}})$

Finally, the posture at time t_2 is:

$$\begin{cases} x_2 = x_{P_2} + d_2 \sin\theta \\ y_2 = y_{P_2} - d_2 \cos\theta \\ \theta_2 = \arctan2(y_{P_2} - y_{P_1};\ x_{P_2} - x_{P_1}) + \theta_{rel} - \frac{\pi}{2} \end{cases}$$

HYBRID LOCALIZATION METHOD USING KALMAN FILTER

A non linear discrete system can be written in the state-space form as:

$$\begin{cases} X_{k+1} = f(X_k, U_k) \\ Y_k = g(X_k) \end{cases}$$

The first equation is called EVOLUTION EQUATION, while the second OBSERVATION EQUATION. X is the state vector and f the state function. U is the system input and y the output

A real system is characterised by additive noise, α for evolution and γ for observation. They are characterised by the covariance matrices Q_α and Q_γ
The notation will use: hat ($\hat{}$) for estimated values, pedix $k+1/k$ means at time $k+1$ before taking into account any measurement available at time t_{k+1}. While, $k+1/k+1$ means after taking into account the measurement
The equation of the EXTENDED KALMAN FILTER are:

PREDICTION PHASE:

$$\begin{cases} X_{k+1/k} = f(\hat{X}_{k/k}, U_k) \\ P_{k+1/k} = A_k \cdot P_{k/k} \cdot A_k^T + Q_\alpha \end{cases}$$

↳ estimate covariance

ESTIMATION PHASE:

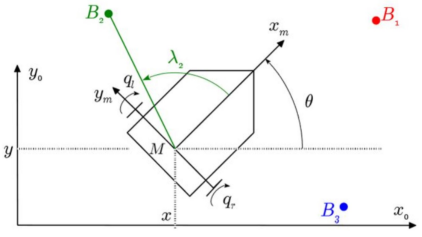
$$\begin{cases} \hat{X}_{k+1/k+1} = \hat{X}_{k+1/k} + K_k \left[Y_k - g(\hat{X}_{k+1/k}) \right] \\ P_{k+1/k+1} = (I - K_k \cdot C_k) P_{k+1/k} \end{cases}$$

with: $A_N = \frac{\partial f}{\partial X}\left(\hat{X}_{K/K};\ U_K\right)$ $\quad \begin{cases} C_K = \frac{\partial g}{\partial X}\left(\hat{X}_{K+1/K}\right) \\ K_K = P_{K+1/K} \cdot C_K^T \left(C_K \cdot P_{K+1/K} \cdot C_K^T + Q_\gamma\right)^{-1} \end{cases}$

The prediction phase predicts the state at time t_{K+1} from the last estimated state $\hat{X}_{K/K}$ and the known input U_K (corresponds to numerical integration)
The second equation takes into account the drift given by repeated prediction
The estimation phase, states that the correction between $\hat{X}_{K+1/K}$ and $\hat{X}_{K+1/K+1}$ is proportional to Y_K and its expected value $\hat{Y} = g\left(\hat{X}_{K+1/K}\right)$
The second system is used to compute the gain as function of the covariance

APPLICATION EXAMPLE

The robot and its sensors



- The sensor is a rotating sensor. It detects only one beacon at a time, and not frequently (it is slow).
- Its motion between two measurements cannot be neglected.

We want to estimate the posture of the robot. Recalling the equations:

$$X_{K+1} = \begin{bmatrix} x_{K+1} \\ y_{K+1} \\ \theta_{K+1} \end{bmatrix} = \begin{bmatrix} x_K + \Delta D_K \cos\theta_K \\ y_K + \Delta D_K \sin\theta_K \\ \theta_K + \Delta\theta_K \end{bmatrix} = f\left(X_K, U_K\right)$$

with: $U_K = \begin{bmatrix} \Delta D_K \\ \Delta\theta_K \end{bmatrix} = \begin{bmatrix} (r_r \Delta q_{r,K} + r_e \Delta q_{e,K})/2 \\ (r_r \Delta q_{r,K} - r_e \Delta q_{e,K})/e \end{bmatrix}$

Under a more synthetic form: $\begin{cases} X_{K+1} = f\left(X_K, U_K\right) + \alpha_K \\ U_K = [\Delta D_K,\ \Delta\theta_K]^T \end{cases}$ → adding noise

$\alpha = X_{K+1} - f(X_K, U_K)$
noise = reality − model

It is possible to express the outputs as a function of the state using the observation equation. There are as many different equations as many beacons. When a sensor detects a beacon we obtain a measurement λ_J, that depends on the actual posture of the vehicle

$$Y_J = \lambda_J = \arctan2\left(y_B - y_J;\ x_B - x_J\right) - \theta_J = g_B(X) \qquad B = [x_B,\ y_B]$$

↳ beacon

Using this formula is possible to compute the estimated measurement $\hat{\lambda}_J$ and correct X accordingly.
The correction gain will include these terms:

- Innovation: $\lambda_J - \hat{\lambda}_J$ (size $m \times 1$)
- The quality of the measurements, characterised by Q_γ (size $m \times m$) ← $Y = g(X) + \gamma$
- The last available estimate $\hat{X}_{J/J-1}$ (size $n \times 1$), via its quality $P_{J/J-1}$
- How a small vehicle displacement influences the measurement $C_J = \frac{\partial g_B}{\partial X}\left(X_{J/J-1}\right)$

The covariance matrix of $\lambda_J - \hat{\lambda}_J$ is $Q_\gamma + C_J \cdot P_{J+1/J} \cdot C_J^T$ and the correction gain should be inversely proportional to this term.

Rembember

- The KF is statistically optimal for a linear system, disturbed by white Gaussian noises, independent of each other and independent of the state.
- The EKF does not meet these criteria and is not necessarily statistically optimal.
- But in many situations it is a good, reasonably simple and computationally efficient estimator.

General equations

$$\begin{cases} \widehat{X}_{k+1/k} = f\left(\widehat{X}_{k/k}, U_k\right) \\ Y_k = g(X_k) \end{cases}$$
$$A_k = \frac{\partial f}{\partial X}\left(\widehat{X}_{k/k}, U_k\right)$$
$$P_{k+1/k} = A_k \cdot P_{k/k} \cdot A_k^T + Q_\alpha$$
$$C_k = \frac{\partial g}{\partial X}\left(X_{k+1/k}\right)$$
$$K_k = P_{k+1/k} C_k^T \left(C_k P_{k+1/k} C_k^T + Q_\gamma\right)^{-1}$$
$$\widehat{X}_{k+1/k+1} = \widehat{X}_{k+1/k} + K_k \left[Y_k - g\left(\widehat{X}_{k+1/k}\right)\right]$$
$$P_{k+1/k+1} = (I - K_k \cdot C_k) P_{k+1/k}$$

Fundamental: The output vector Y must be a function of the state vector only. **Any parameter in the expression of g which is not from X must be a known constant**.

Overall algorithm

What happens at time t_{k+1}

// **Prediction phase:**

// Read sensors and calculate input:
$$\mathrm{U} \leftarrow [\Delta D, \Delta\theta]^T$$

// Calculate linear approx. of system:
$$\mathrm{A} \leftarrow \frac{\partial f}{\partial X}(X, U)$$

// Predict state :
$$\mathrm{X} \leftarrow f(X, U)$$

// Propagate error:
$$\mathrm{P} \leftarrow A \cdot P \cdot A^T + Q_\alpha$$

If an external measurement Y has been obtained during last period
then
// Consider Y corresponds to time t_{k+1}. Perform **estimation phase:**

// Calculate gain:
$$C \leftarrow \frac{\partial g_B}{\partial X}(X)$$
$$K \leftarrow PC^T\left(CPC^T + Q_\gamma\right)^{-1}$$

// Update state:
$$X \leftarrow X + K\left[Y - g(X)\right]$$
$$P \leftarrow (I - K \cdot C) P$$
endif

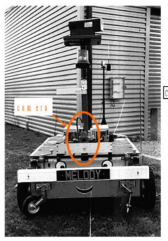
COHERENCE TEST

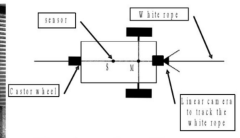
Consider the sensors that detect light of the beacons. It is necessary to take into account the possibility to detect the reflection of the light. It is possible to test the coherence of a measurement by calculating the associated MAHALANOBIS DISTANCE:

$$d^2 = \left(Y_k - \hat{Y}_{k+1/k}\right)^T \left(C_k \cdot P_{k+1/k} \cdot C_k^T + Q_\gamma\right)^{-1} \left(Y_k - \hat{Y}_{k+1/k}\right)$$

If $d <$ threshold, then the measurement is coherent and can be used in the estimation phase. It can happen that two or more distances are lower than d_{thresh} than there is an ambiguity

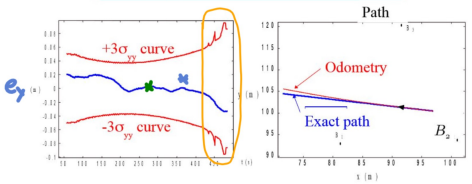
Experimental setup



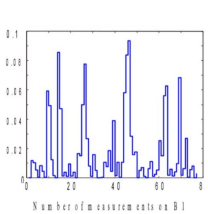


- The robot tracks a white rope tightened between know points.
- The points have been precisely localised by surveyors with sub-centimeter accuracy.

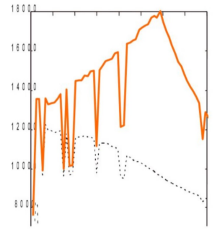
Experimental results



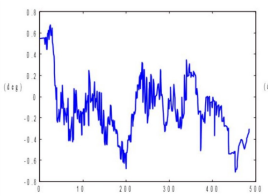
- The (estimated) σ is extracted from the covariance matrix.
- The error signal should always be in the +/- 3σ interval.
- The increase of σ at the end of the path reflects a less favorable beacon-vehicle configuration.
- B_2 is sometimes missed at the end: things get worse...



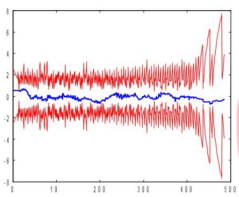
Mahalanobis distance calculated with B_1 when B_1 is actually detected



Mahalanobis distances calculated with B_2 and B_3 when B_1 is actually detected



Heading error



Heading error with corresponding estimated +/- 3σ.

* Lateral error $(\hat{y} - y)$
* There is a mimimum because the beacons allow a better localisation in that position
* exceeding this threshold is extremely unlikely. (For Gaussian at 99,7% we are inside
* at the end only two beacons are detected → poor performance

LOCALISATION AND OBSERVABILITY

Observability of a dynamical system informs on the ability to reconstruct its state, given the outputs and inputs of the system. A continous time system is described by a differential system of the form:

$$\begin{cases} \dot{x} = f(x,u) & x \in \mathbb{R}^n \\ y = g(x) & y \in \mathbb{R}^p \end{cases}$$

The case $n=p$ is special because x can be expressed as function of y inverting g

PROP: A system is observable *iff* it is possible to express the state vector as a function of the input vector, the output vector and their time derivatives

$$x = \varphi\left(y, \dot{y}, \dots, y^{(k)}, u, \dot{u}, \dots, u^{(k_2)}\right) \quad \text{with } k_1 \in \mathbb{N} \text{ and } k_2 \in \mathbb{N}$$

The system is GENERICALLY OBSERVABLE if the set of points for which this condition is not satisfied is a zero-measure set

LIE DERIVATIVE: Let h be a differentiable real-valued function, and f a real vector function, then the Lie derivative of h with respect to g is the vector field:

$$\underbrace{L_f h}_{\text{Scalar}} = \frac{\partial h}{\partial x} f \qquad \text{with } h(x) \in \mathbb{R}^2$$

This is useful, because taking a dynamic system:

$$\begin{cases} \dot{x} = f(x,u) \\ y = g(x) = (g_1(x), \dots, g_p(x))^t, & y \in \mathbb{R}^p \end{cases}$$

The derivatives of the outputs with respect to time yield:

$$\dot{y}_i = \frac{\partial g_i}{\partial t} = \frac{\partial g_i}{\partial x}\dot{x} = \frac{\partial g_i}{\partial x} f = L_f g_i \qquad \ddot{y}_i = \frac{\partial [L_f g_i]}{\partial t} = L_f^2 g_i$$

Observability rank condition: Let $g: \mathbb{R}^m \to \mathbb{R}^p$. Its component (are real valued function) can be written as: $g = [g_1 \dots g_p]^t$

Let the observability matrix O be defined by:

$$O = \left[dg^1 \; dL_f g^1 \dots dL_f^{m-1} g^1 \dots dg^p \dots dL_f^{m-1} g^p \right]$$

If the rank of O is full (equal to n), then the system is observable

In the following examples it will be used a continous state of the system denoted by $X = [x \; y \; \theta]^T$, and m beacons that will gives m measurements. The observation equation corresponding to beacon B_k, which coordinates are (x_{B_k}, y_{B_k}) is written as: $\lambda_k = \arctan2\left(y_{B_k} - y, x_{B_k} - x\right) - \theta = g_k(X)$ with $1 \le k \le m$
These m measurements are grouped in a single vector λ

In addition the input vector of the system is $U=[v\ \omega]^T$ and the evolution model is:

$$\begin{cases}\dot{x}=v\cos\theta\\ \dot{y}=v\sin\theta\\ \dot{\theta}=\omega\end{cases} \qquad \dot{X}=f(X,U)$$

OBSERVABILITY WITH THREE BEACONS

This is the standard situation for localisation problem and we have

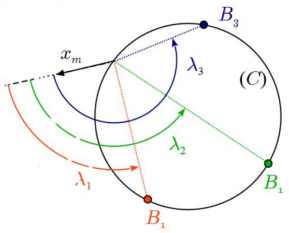
$$\begin{cases}\dot{X}=f(X(t),U(t)) & X\in\mathbb{R}^3\\ \lambda(t)=g(X(t))\end{cases}$$

if g can be inverted, then X can be written as a function of λ

Let us consider the Jacobian matrix $O_1=[dg_1\ dg_2\ dg_3]^T$ and

$$dg_i=\left[\frac{y_{Bi}-y}{D_i}\quad\frac{x_B-x}{D_i}\quad -1\right]^T \text{ with } D_i=(y-y_B)^2+(x-x_B)^2$$

So, the function can be inverted if the determinant of O_1 is not equal to zero in that case the observability is granted. We want to check what happen when the determinant is zero. The solutions of the equation $\det(O_1)=0$ correspond to the unique circle passing through the three non aligned beacons.



The observability matrix is:

$$O=[dg_1\ dL_fg_1\ dL_f^2g_1\ \dots dg_3\dots dL_f^2g_3]\in\mathbb{R}^{3\times g}$$

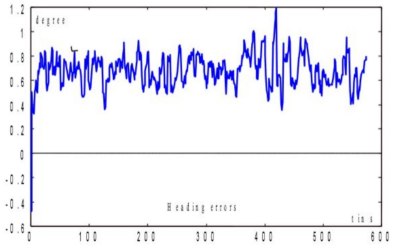
The first order Lie derivatives are the following:

$$L_fg_i(X)=v\left(\frac{y_{Bi}-y}{D_i}\cos\theta+\frac{x-x_{Bi}}{D_i}\sin\theta\right)-\omega$$

If $v=0$ then $L_fg_i(X)=-\omega$ and $dL_fg_i=dL_{fgi}^2=0$ then $O=[dg_1\ dg_2\ dg_3\ 0\dots 0]$

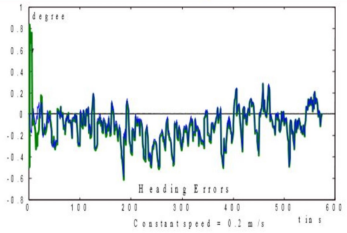
Hence, if the translation speed is zero $\det(O)=\det(O_1)$ and $\text{rank}(O)<3$ Still, this does not prove that the system is not observable because the rank condition is a sufficient condition, but not necessary.

Simulation
motionless robot on the circle



The estimator does not converge: it cannot drive the initial error to zero.

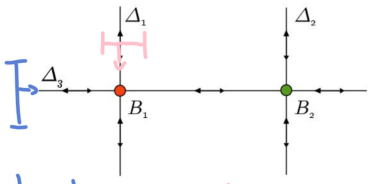
Simulation - moving on the circle



After a transient phase, the error curves with and without initial error are « the same ». Precision is not as good as outside the circle.
This example illustrates the influence of the input on observability, for a non linear system.

Observability with two beacons

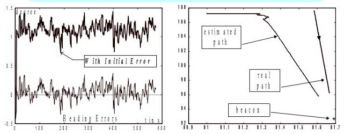
We assume that possibly a beacon is hidden or too far. The analysis of the rank of O becomes complex as higher derivatives need to be used. By studying the determinants of the various square sub-matrices of O with a software, we found three potentially dangerous vehicle motions.



$\lambda_1 = \lambda_2 = 0$ $\quad \lambda_2 = 0$

$\dot{\lambda}_1 = \dot{\lambda}_2 = 0$

Simulations on $\Delta 1$



When the vehicle moves along Δ_1, the observer does not converge.

Remark: our observer converges for other straight line paths through B_1, other than Δ_1. Other researchers tested a least-square observer which did not converge: observability does not guarantee that a particular observer will converge...

Some conclusions

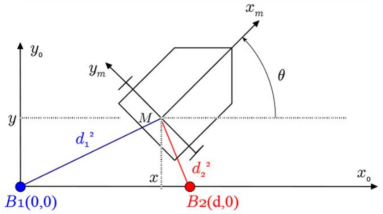
- Observability analysis allows to find dangerous situations which are sometimes not easy to suspect intuitively.
- Observability is not a « binary » phenomenon. When the vehicle comes close to non observable situations, the precision degrades and numerical problems can appear. So much for null sets!

Observability with one beacon

In this situation the system is never observable, in simulation the algorithm diverge slower than odometry. (some information better than nothing)

Exercise

Exercise



State : $X=[x,y,\theta]^T$
Study observability. $\quad Y = g(X) = \begin{bmatrix} x^2+y^2 \\ (x-d)^2+y^2 \end{bmatrix}$

$$X = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \quad U = \begin{bmatrix} v \\ w \end{bmatrix} \quad f = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ w \end{bmatrix}$$

$$Y = g(X) = \begin{bmatrix} x^2+y^2 \\ (x-d)^2+y^2 \end{bmatrix} = \begin{bmatrix} g_1(x) \\ g_2(x) \end{bmatrix}$$

Step 1: Calculate the gradients $\quad dg_1 = \begin{bmatrix} 2x \\ 2y \\ 0 \end{bmatrix} \quad dg_2 = \begin{bmatrix} 2(x-d) \\ 2y \\ 0 \end{bmatrix}$

There are not enough columns to find a sub-matrix of O we need a first order Lie derivative

Step 2: Gradients of the first order Lie derivatives: $L_f g_1 = \frac{\partial g_1}{\partial x} f = \begin{bmatrix} 2x & 2y & 0 \end{bmatrix} \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ w \end{bmatrix}$

$$L_f g_1 = 2v\left[x\cos\theta + y\sin\theta\right] \quad dL_f g_1 = \begin{bmatrix} \cos\theta \\ \sin\theta \\ -x\sin\theta + y\cos\theta \end{bmatrix}$$

Step 3: Study the rank of the first sub-matrix of O

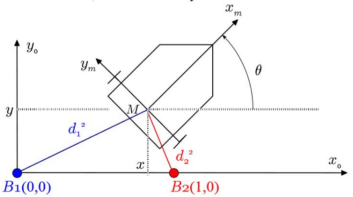
$$O_{red_1} = \begin{bmatrix} 2x & 2(x-1) & 2v\cos\theta \\ 2y & 2y & 2v\sin\theta \\ 0 & 0 & 2v(-x\sin\theta + y\cos\theta) \end{bmatrix}$$

$$\det(O_{red_1}) = 8vy(y\cos\theta - x\sin\theta)$$

Study the cases where determinant is equal to 0

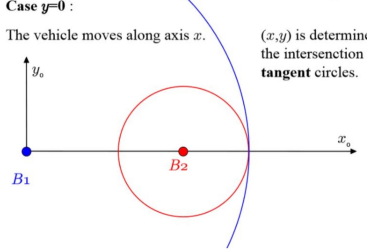
Case v=0 :

The vehicle rotates around point M. There is no way to determine theta, which is clearly not observable.



Case y=0 :

The vehicle moves along axis x.



(x,y) is determined t
the intersection of t
tangent circles.

Case $-x\sin\theta + y\cos\theta = 0$:

We can either try to interpret these situations or look at another sub-matrix, using $dL_f g_2$, in case the smoke clears..

$$dL_f g_2 = 2v \begin{bmatrix} \cos\theta \\ y\cos\theta - \sin\theta\,(x-1) \end{bmatrix}$$

$$O_{red_2} = \begin{bmatrix} 2x & 2(x-1) & 2v\cos\theta \\ 2y & 2y & 2v\sin\theta \\ 0 & 0 & 2vy\cos\theta - v\sin\theta\,(2x-2) \end{bmatrix}$$

Case $-x\sin\theta + y\cos\theta = 0$

$det(O_{red_2}) = 8v\,y[\sin\theta - x\sin\theta + y\cos\theta]$

Does not solve the v=0 problem (expected).
Does not solve the y=0 problem.
Solves the $-x\sin\theta + y\cos\theta = 0$ problem.

The y=0 case (robot moving along the x axis) remains. The gradients of second order and higher Lie derivatives all have w as a factor, and hence cannot guarantee observability in this case, as w will be zero.

MOBILE ROBOT CONTROL

IRREDUCIBILITY: a system is reducible if there exists a change of coordinates such that some of the new coordinates are identically to zero along any motion of the system. Our goal is to reduce the posture kinematic model: $\dot{z} = B(z) \cdot u$. This is related to the involutive closure $\bar{\Delta}$ of the distribution: $\Delta(z) = \text{span}\{\text{col}(B(z))\}$ and the system is reducible if $\dim(\bar{\Delta}) < \dim(z)$
To calculate $\dim(\bar{\Delta})$ it is necessary to take $B = [b_1, b_2 \dots, b_m]$ and all the Lie brackets of vector b_i: $M = [b_1, b_2, \dots, b_m, [b_1, b_2], \dots, [b_1, b_m], \dots [b_i, [b_j, b_j]] \dots]$
If the rank of this matrix is $\dim(z)$ then the system is irreducible

NOTE: Lie bracket: $[b_1, b_2] = \frac{\partial b_1}{\partial z} b_2 - \frac{\partial b_2}{\partial z} b_1$ (result is a vector)

EXAMPLE:

The posture kinematic model of the (2,0) robot is: $\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}$

To check if it is irreducible $b_1 = \begin{bmatrix} c\theta \\ s\theta \\ 0 \end{bmatrix}$ $b_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

$\frac{\partial b_1}{\partial z} = \begin{bmatrix} 0 & 0 & -s\theta \\ 0 & 0 & c\theta \\ 0 & 0 & 0 \end{bmatrix}$ $\frac{\partial b_2}{\partial z} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ $b_3 = [b_1, b_2] = \frac{\partial b_1}{\partial z} b_2 - \frac{\partial b_2}{\partial z} b_1 = \begin{bmatrix} -s\theta \\ c\theta \\ 0 \end{bmatrix}$

$[x, y, \theta]$

$\frac{\partial b_3}{\partial z} = \begin{bmatrix} 0 & 0 & -c\theta \\ 0 & 0 & -s\theta \\ 0 & 0 & 0 \end{bmatrix}$ $b_4 = [b_1, b_3] = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$ $\frac{\partial b_4}{\partial z} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ $b_5 = [b_2, b_3] = b_1$

$\dim(\bar{\Delta}(z)) = \text{rank}(b_1, b_2, b_3) = 3 = \dim(z)$

PROPERTIES:

1) For the posture kinematic model $\dot{z} = B(z) \cdot u$ of a wheeled mobile robot the input matrix $B(z)$ has: $\text{rank}(B(z)) = \delta_m + \delta_s \quad \forall z$
2) The involutive distribution $\bar{\Delta}(z)$ has constant maximal dimension $\dim(\bar{\Delta}(z)) = 3 + \delta_s \quad \forall z$

As a consequence the posture kinematic model of a wheeled robot is IRREDUCIBLE

CONTROLLABILITY

Considering an equilibrium configuration $\bar{z} = (\bar{\xi}^t; \bar{\beta}_s^t)^t$. The robot is motionless, constant posture, constant orientation of the wheels and the input is $\bar{u} = 0$

PROPERTY 3: the controllability rank of the linear approximation of the posture kinematic model $\dot{z} = B(z) \cdot u$ around an equilibrium configuration is $\delta_m + \delta_s = \delta_M$

The consequence is that the linear approximation of the PKM of (3,0) robot is completely controllable. While the others are not controllable However, the PKM of the wheeled mobile robot is controllable, in pratice *

* see next page

FEEDBACK CONTROL

For an omnidirectional robot the following control law can be used:

$u(z) = B^{-1}(z)\, A(z - z^*)$ with A a stable matrix

The closed loop dynamics governed by: $\frac{d}{dt}(z - z^*) = A(z - z^*)$
and so there is an exponential convergence to z^*

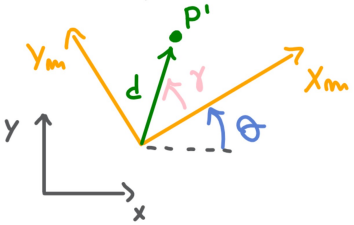
The PNM of restricted mobility robot is not stabilizable by a continous static time-invariant state feedback $u(z)$ but is stabilizable by a continous time-varying static state feedback $u(z,t)$

Our goal is the posture tracking. We want to find a stable state feedback controller to track a reference moving posture $\xi_r(t)$ assumed twice differentiable. To do that we need a control law s.t: ↳ desired trajectory

- $\bar{\xi}(t) = \xi(t) - \xi_r(t)$ and $v(t)$ are bounded at all times
- $\lim_{t \to \infty} \left(\xi(t) - \xi_r(t) \right) = 0$
- If $\xi(0) = \xi_r(0)$ then $\xi(t) = \xi_r(t) \quad \forall t$

POINT TRACKING

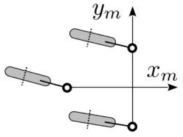
In some cases, it is sufficient to control the position of a given point
It is possible to define it by its polar coordinates in the robot frame (d, γ)



$$^{o}p' = \begin{bmatrix} x + d\cos(\theta + \gamma) \\ y + d\sin(\theta + \gamma) \end{bmatrix}$$

OSS: When using the PNM the control u is homogeneous to a velocity. Hence, it is a speed control. It's possible to use also the torque control (not seen)

EXAMPLE: STATIC FEEDBACK (3,0)



$$z = \xi = (x, y, \theta)^t$$

$$u = u_m = ({}^m\dot{x}, {}^m\dot{y}, \dot{\theta})^t = (v_x, v_y, \omega)^t$$

$$\Sigma(\beta_s) = \Sigma = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \rightarrow \text{NO } \beta_s$$

$$B(z) = {}^0\Omega_m(\theta).\Sigma = {}^0\Omega_m(\theta) \overset{\mathbb{1}}{=} \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

So, we have: $\dot{z} = B(z) \cdot u$

$$\dot{z} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v_x \\ v_y \end{bmatrix}$$

Then $\dot{\xi} = {}^{o}\Omega_m(\theta) \cdot u \rightarrow u = {}^{m}\Omega_o(\theta) \cdot \dot{\xi}$

Let's introduce an auxilary control w : $u = {}^{m}\Omega_o(\theta)\, w$, so $\dot{\xi} = w$
The desired trajectory is defined by: $(\xi^d, \dot{\xi}^d)$ and the control is defined as:
$w = \dot{\xi} + K_p(\xi^d - \xi)$ The tracking will be: $e(t) = e^d - e$ and $\dot{e}(t) + K_p \cdot e(t) = 0 \rightarrow e(t) = e^{-K_p t}$
$\dot{\xi}$ FEED-FORWARD TERM ↳ ERROR

see slides for case (2,0)

PATH PLANNING

The navigation of the robot is defined by the PATH, a geometric curve or a sequence of points and the TRAJECTORY, the law of motion along the path
It is possible to have four different tasks:

- MOTION PLANNING: find a collision free motion between start and goal states
- COVERAGE: pass a tool/sensor over all points of a certain area
- LOCALIZATION: use a map and sensors to determine the configuration of the robot
- MAPPING: explore an unknown environment to build a representation of it

It is important to define: the number of degrees of freedom of the system, the configuration space topology, the kinematic constraints, the model
About the algorithm is important to know if it is optimal and complete, the computational complexity

DEFINITIONS AND NOTATIONS

In Euclidian space:
- Robots move in the workspace W (planar $\to R^2$ or 3D $\to R^3$)
- W may contain workspace obstacles WO_i
- Free workspace: $W_{free} = W - \cup_i WO_i$

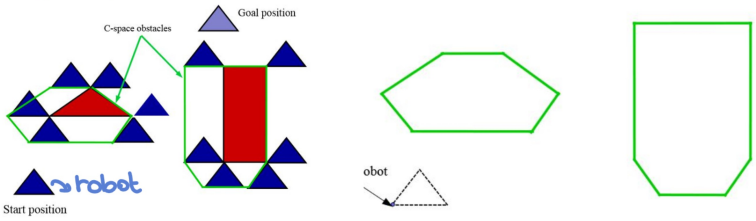
In configuration space Q:
- q is a configuration of Q and R(q) is the space occupied when the configuration is q
- Configuration space obstacle $QO_i = \{q \mid R(q) \cap WO_i \neq \emptyset\}$
- Free configuration space $Q_{free} = Q - \cup_i QO_i$

PATH PLANNING: find a function c of class C^0 such that:
$c: [0,1] \to Q \quad c(0) = q_{start} \quad c_1 = q_{goal} \quad c(s) \in Q_{free}$ for all s in $[0,1]$

When c is parametrized by time t, c(t) is a trajectory. By taking the derivatives is possible to take velocities and acceleration, so c must be of class C^2. Finding such a function is MOTION PLANNING

Finally, about the robot system it is made by one or more solids attached by joints. The configuration is a complete specification of every point of the robot, while the configuration space (C-space) is the space Q of all possible configurations. To conclude, the number of degree of freedom is the dimension of the configuration space and also the minimum number of parameters needed to specify the configuration.

OBSTACLE GROWING



This is the C-space representation and the plan must be found in this space