Introduction to Robotics

J-P. Merlet

INRIA, COPRIN project team

COPRIN project team

COPRIN project team

A team involved in the development of

- analysis and modeling of robots
- management of uncertainties in robotics
- design methodology for mechanisms

COPRIN project team

A team involved in the development of

- parallel robot
- assistance robotics

Robots

Robots

What is a robot ?

Robots

What is a robot ?

- No clear definition

Robots

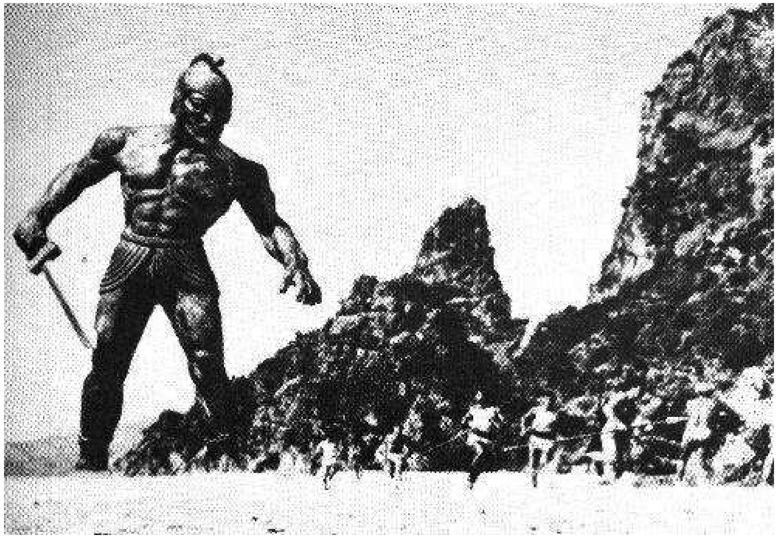
What is a robot ?

- No clear definition
- etymology: *robota* in Czech $\longrightarrow$ labor slave

Robots

What is a robot ?

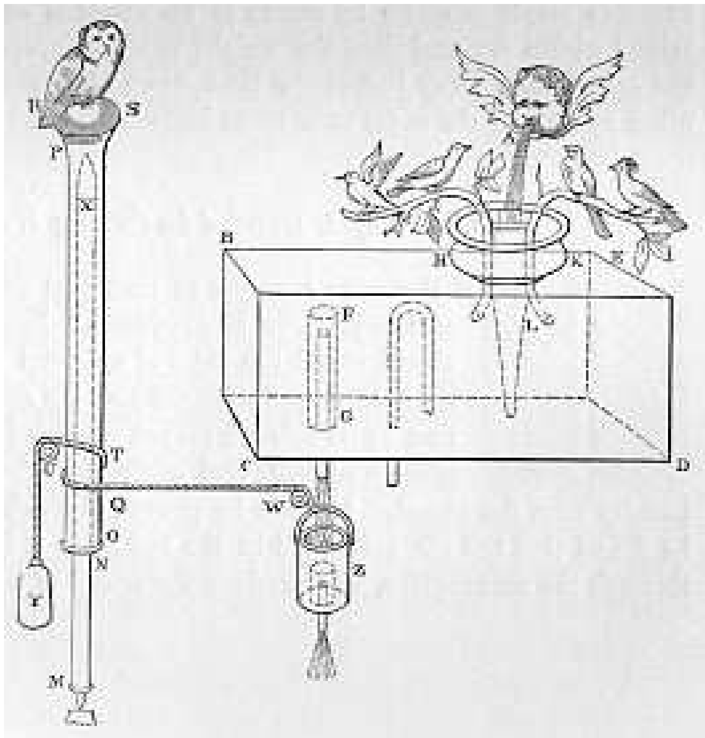
Historically an instrument of the gods and kings



Robots

What is a robot ?

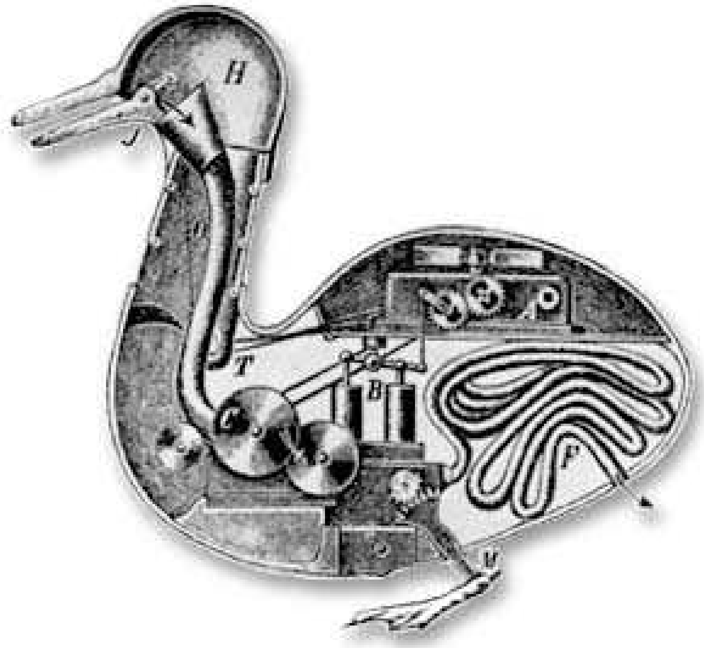
For the Greek and Roman entertainment machinery



Robots

What is a robot ?

During the French Revolution: *automata*



Robots

What is a robot ?

In the 60's a science-fiction concept

Robots

What is a robot ?

In the late 60's industry is looking for machines that are able to produce controlled repetitive motion 24/24

Robots

What is a robot ?

In the late 60's industry is looking for machines that are able to produce controlled repetitive motion 24/24

- to pick and place objects
- to perform assembly tasks
- for grinding and deburing operations
- . . .

Robots

What is a robot ?

In the late 60's industry is looking for machines that are able to produce controlled repetitive motion 24/24

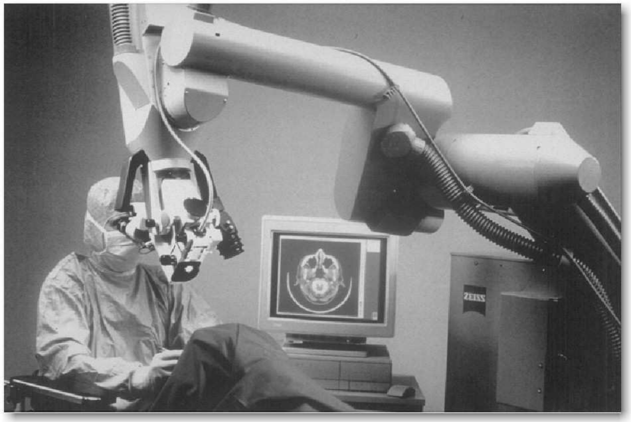
they design the first robot manipulator

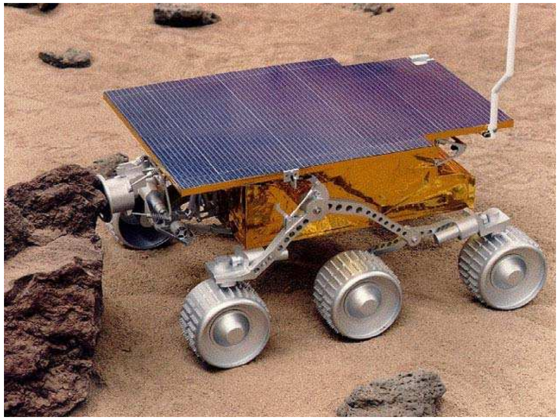
Robots

Nowadays robots are able to perform other tasks than *industrial manipulation*

- navigation
- medical: surgery, assistance
- spatial exploration
- simulator
- domestic tasks
- . . .

Robots





The components of a robot

The components of a robot

- a collection of links (rigid bodies)

The components of a robot

- a collection of links (rigid bodies)
  - the end-effector: the link that has to perform the motion
  - the base: the link of the robot that is connected to the ground

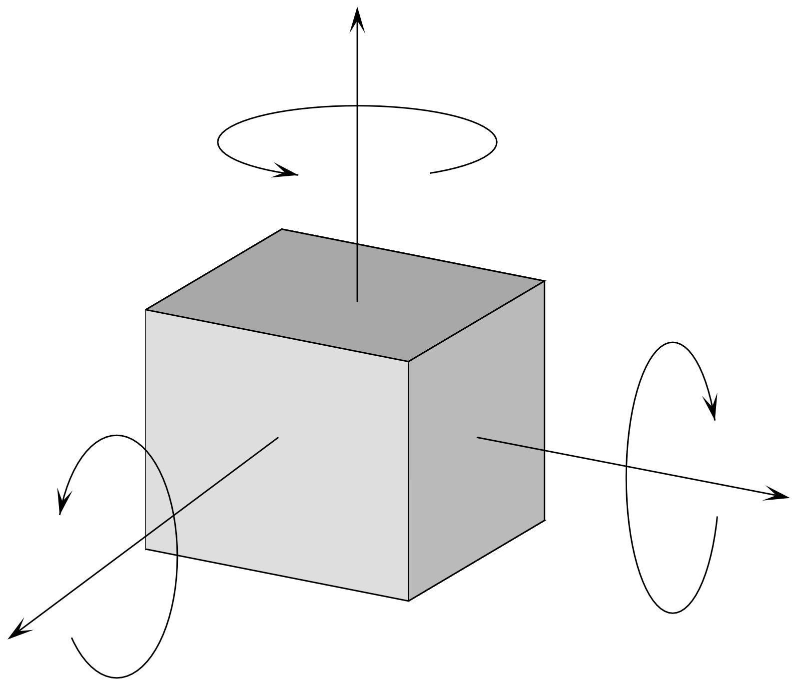
The components of a robot

- a collection of links (rigid bodies)
- that are connected to each other by joints that allow motion(s) between the links

The components of a robot

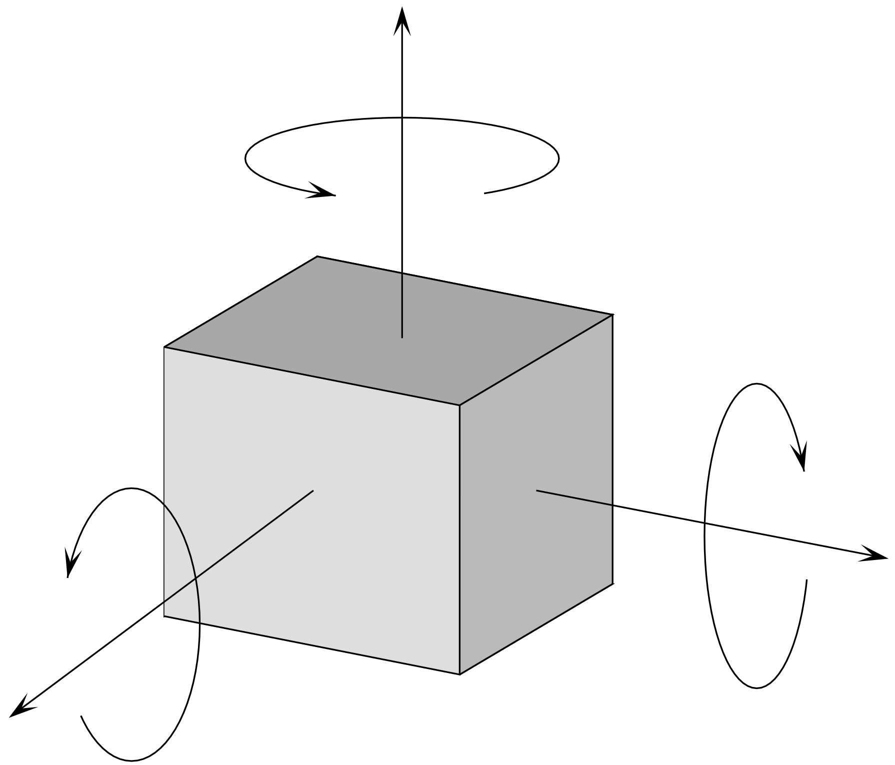
Motion(s) of a rigid body

- a rigid body may translate in 3 directions
- a rigid body may rotate around these 3 directions



The components of a robot

Motion(s) of a rigid body



They are the 6 degrees of freedom (dof) of the body

The components of a robot

A joint will allow some dof between the links it connects

The components of a robot

A joint will allow some dof between the links it connects

Typically

- 1 dof: a rotation around a fixed axis (revolute joint)

The components of a robot

A joint will allow some dof between the links it connects

Typically

- 1 dof: a rotation around a fixed axis (revolute joint)
- 1 dof: a translation along a fixed axis (prismatic joint)

The components of a robot

A joint will allow some dof between the links it connects

But there are joint that allows more dof

- 3 dof: ball and socket, 3 rotations around a fixed point

The components of a robot

A joint may be:

- passive: the link may move freely along the dof of the joint

The components of a robot

A joint may be:

- passive: the link may move freely along the dof of the joint

There are numerous type of passive joints and they are very important for robotics

$\Rightarrow$ Tuesday afternoon, T. Gayral

The components of a robot

A joint may be:

- passive: the link may move freely along the dof of the joint
- actuated: an actuator imposes motion of the dof of the joint

The components of a robot

A joint may be:

- passive: the link may move freely along the dof of the joint
- actuated: an actuator imposes motion of the dof of the joint
  - rotary motor for revolute joints
  - linear motor for prismatic joints

The components of a robot

dof a robot: may be

- the number of dof of the end-effector that can be controlled by the robot
- the number of independent actuators

The components of a robot

- a collection of links (rigid bodies)
- that are connected to each other by joints
- actuators that move the actuated joints

The components of a robot

- a collection of links (rigid bodies)
- that are connected to each other by joints
- actuators that move the actuated joints
- sensors that measure the motion of the actuated joints

Mechanical architecture

Mechanical architecture

The way the links and joints are assembled to produce the motion of the end-effector

Mechanical architecture

The way the links and joints are assembled to produce the motion of the end-effector

For *manipulators*

Mechanical architecture

The way the links and joints are assembled to produce the motion of the end-effector

For *manipulators*

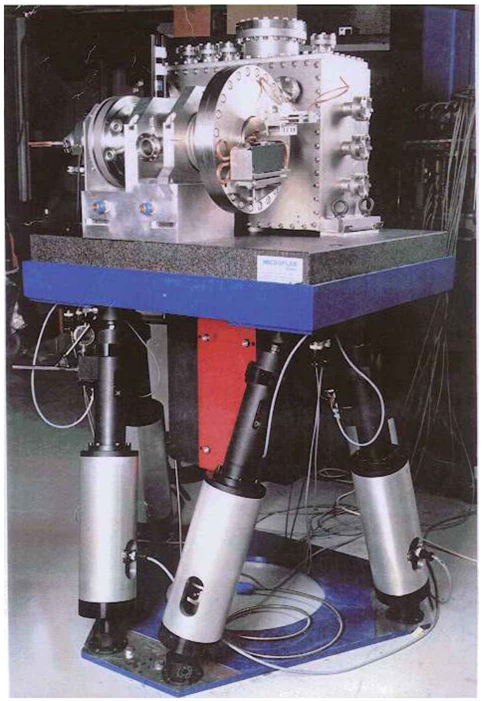
- serial structure: a serie of link-joint/link-joint

Mechanical architecture

The way the links and joints are assembled to produce the motion of the end-effector

For *manipulators*

- serial structure: a serie of link-joint/link-joint
- parallel structure: several independent chains connect the base to the end-effector



Mechanical architecture

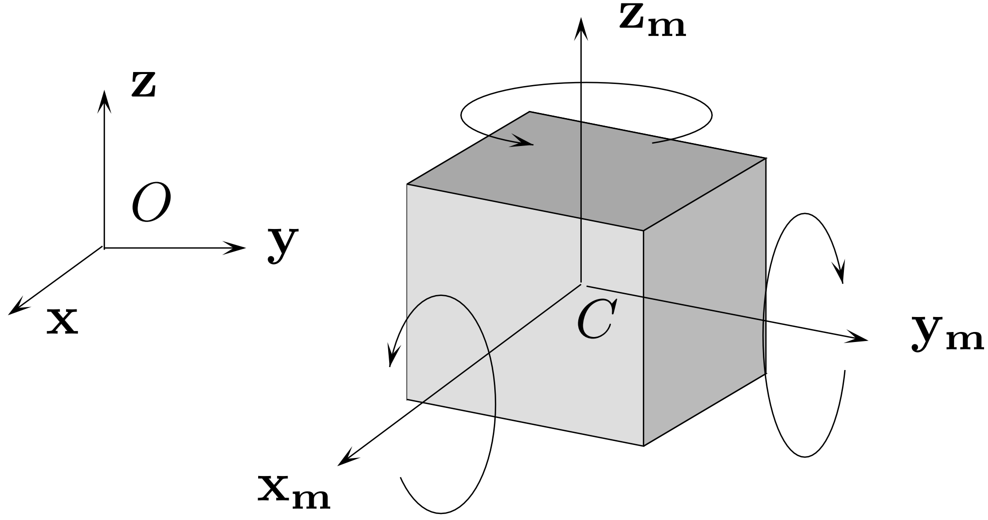
A special case of parallel robot: parallel wire-driven robot: link are extensible wires

The robotics variables

The robotics variables

To define the pose of a rigid body you need:

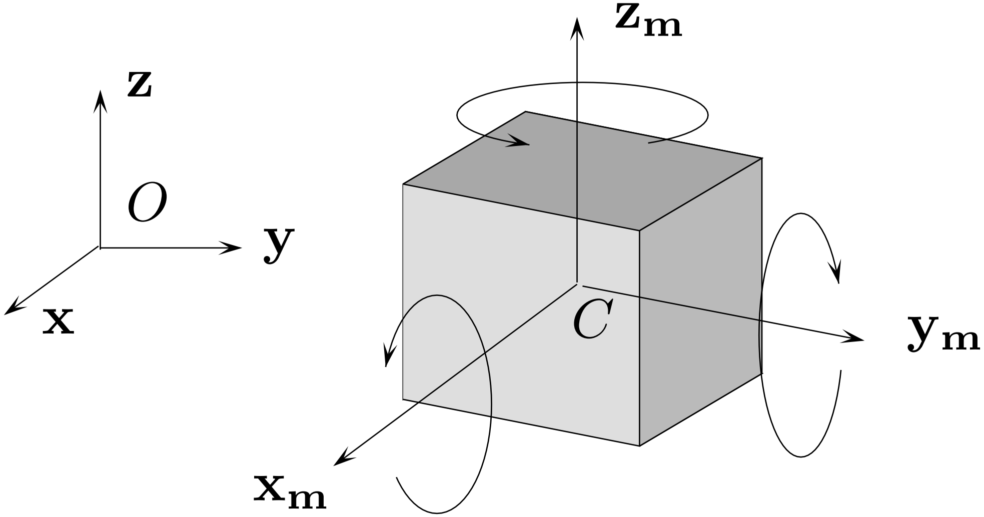
- to define a reference frame (O, **x**, **y**, **z**)



The robotics variables

To define the pose of a rigid body you need:

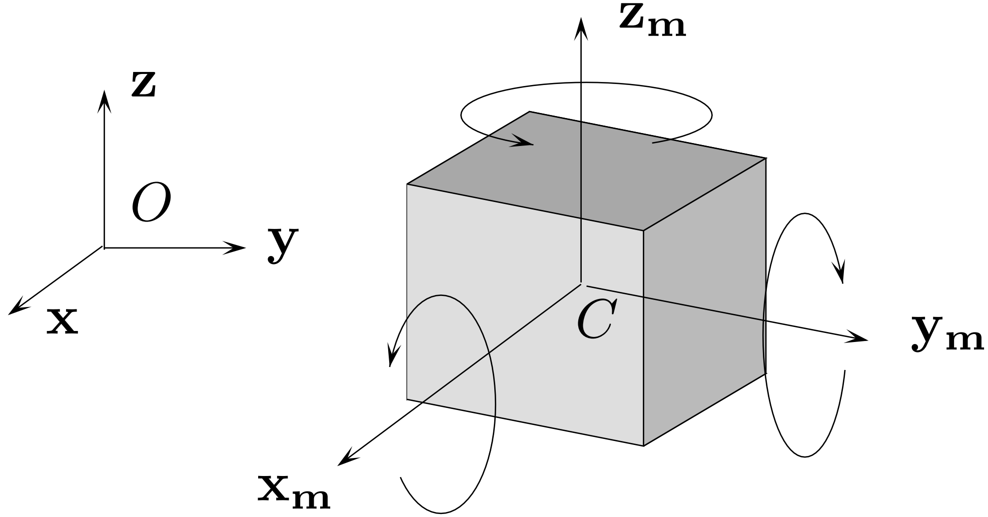
- to define a reference frame (O, **x**, **y**, **z**)
- to choose a point C on the body



The robotics variables

To define the pose of a rigid body you need:

- to define a reference frame (O, **x**, **y**, **z**)
- to choose a point C on the body

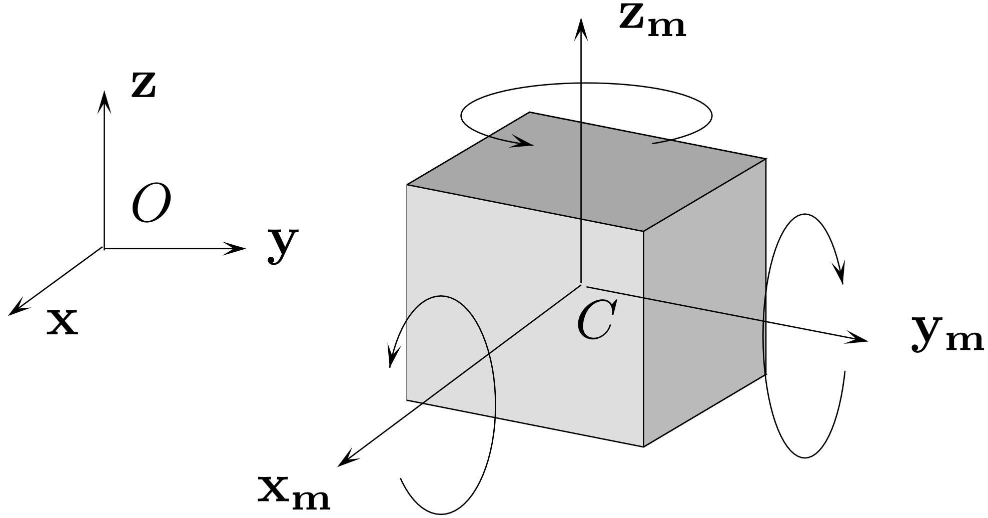


the coordinates of C in the reference frame allows to define the position of the body

The robotics variables

To define the pose of a rigid body you need:

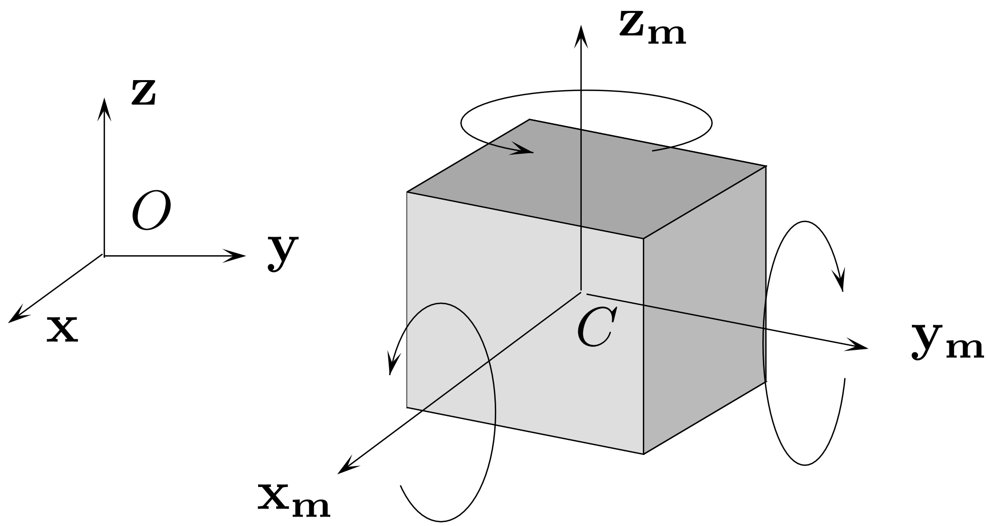
- to define a mobile frame $(C, \mathbf{x_m}, \mathbf{y_m}, \mathbf{z_m})$ that is rigidly attached to the body



The robotics variables

To define the pose of a rigid body you need:

- to define a mobile frame $(C, \mathbf{x_m}, \mathbf{y_m}, \mathbf{z_m})$ that is rigidly attached to the body
- to define a rotation matrix R such that $\mathbf{v} = \mathrm{R}\mathbf{v_m}$



The robotics variables

Rotation matrix:

- a 3×3 orthogonal matrix:

$$\mathbf{R}^T\mathbf{R} = I_3 \quad ||R_i|| = 1 \qquad R_i.R_j = 0$$

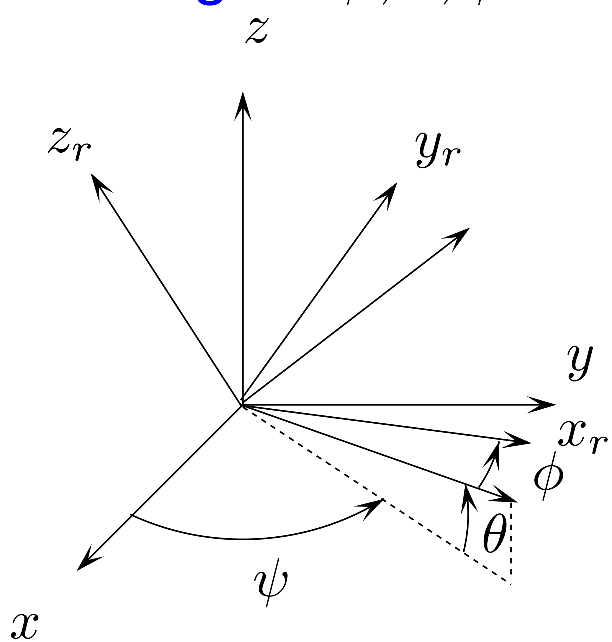
The robotics variables

Rotation matrix:

- a 3×3 orthogonal matrix:
- although a rotation matrix has 9 components its special properties allows to obtain it with a minimum of 3 parameters

The robotics variables

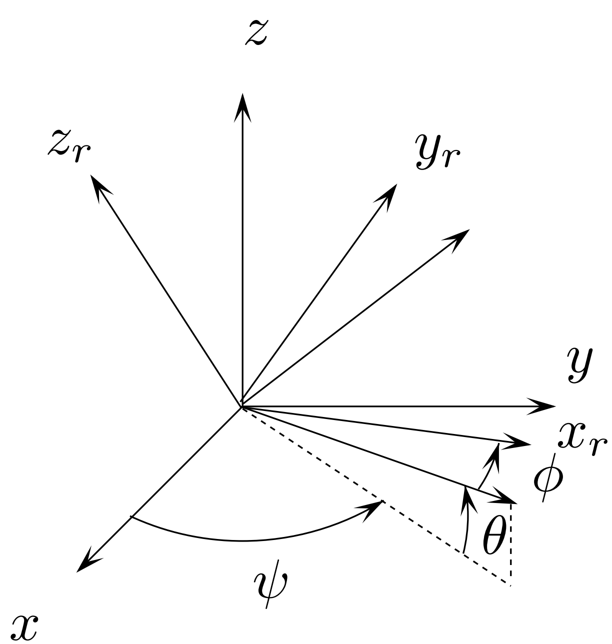
For example the Euler angles ψ, θ, ϕ



The robotics variables

For example the Euler angles ψ, θ, ϕ

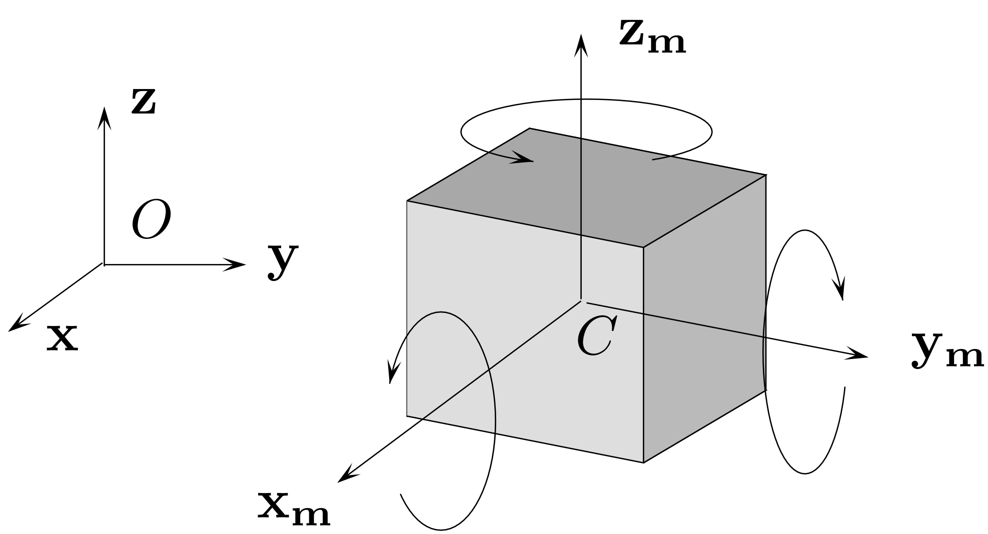
$$\mathbf{R} = \begin{pmatrix} \cos\psi\cos\phi - \sin\psi\cos\theta\sin\phi & -\cos\psi\sin\phi - \sin\psi\cos\theta\cos\phi & \sin\psi\sin\theta \\ \sin\psi\cos\phi + \cos\psi\cos\theta\sin\phi & -\sin\psi\sin\phi + \cos\psi\cos\theta\cos\phi & -\cos\psi\sin\theta \\ \sin\theta\sin\phi & \sin\theta\cos\phi & \cos\theta \end{pmatrix}$$



The robotics variables

To define the pose of a rigid body you need:

- to define a mobile frame $(C, \mathbf{x_m}, \mathbf{y_m}, \mathbf{z_m})$ that is rigidly attached to the body
- to define a rotation matrix $\mathcal{R}$ such that $\mathbf{v} = \mathbf{R}\mathbf{v_m}$



the parameters of the rotation matrix allows to define the orientation of the rigid body

The robotics variables

To define the pose of a rigid body you need:

- 3 parameters for the translation
- at least 3 parameters for the orientation

The robotics variables

Variables for a robot:

The robotics variables

Variables for a robot:

- **x**: the parameters that define the pose of the end-effector, generalized cartesian coordinates

The robotics variables

Variables for a robot:

- **x**: the parameters that define the pose of the end-effector, generalized cartesian coordinates
  - they are what you want to control

The robotics variables

Variables for a robot:

- **x**: the parameters that define the pose of the end-effector, generalized cartesian coordinates
  - they are what you want to control
  - but what you are able to effectively control is the actuators

The robotics variables

Variables for a robot:

- $\mathbf{X}$: the parameters that define the pose of the end-effector, generalized cartesian coordinates
- $\boldsymbol{\Theta}$: the parameters of the actuator, the joint variables

The robotics variables

Variables for a robot:

- $\mathbf{x}$: the parameters that define the pose of the end-effector, generalized cartesian coordinates
- Θ: the parameters of the actuator, the joint variables
  - to reach a given value for a joint variable you may use open loop control: very approximate control

The robotics variables

Variables for a robot:

- $\mathbf{x}$: the parameters that define the pose of the end-effector, generalized cartesian coordinates
- $\mathbf{\Theta}$: the parameters of the actuator, the joint variables
  - to reach a given value for a joint variable you may use open loop control: very approximate control
  - closed-loop control: you use the sensors to measure the value of the joint parameters and you have a control scheme that allows to reach the desired value: accurate control but not perfect

The robotics variables

Variables for a robot:

- $\mathbf{x}$: the parameters that define the pose of the end-effector, generalized cartesian coordinates
- $\mathbf{\Theta}$: the parameters of the actuator, the joint variables
- $\mathbf{\Theta_m}$: the measured joint variables

The robotics variables

Variables for a robot: you may also be in interested in the velocity

- $\dot{\mathbf{X}}$: the translation and angular velocity of the end-effector, generalized velocities
- $\dot{\mathbf{\Theta}}$: the joint velocities
- $\dot{\mathbf{\Theta}}_{\mathbf{m}}$: the measured joint velocities

The robotics variables

Variables for a robot: you may also be in interested in the force/torques

- $\mathcal{F}$: the forces/torques exerted by the end-effector
- τ: the joint forces/torques

The robotics variables

Generally speaking you are interested in

- controlling the parameters in the end-effector space

The robotics variables

Generally speaking you are interested in

- controlling the parameters in the end-effector space
- but by applying a control in the joint space

The robotics variables

Generally speaking you are interested in

- controlling the parameters in the end-effector space
- but by applying a control in the joint space

Hence you have to establish the relations between these 2 spaces

The robotics variables

Generally speaking you are interested in

- controlling the parameters in the end-effector space
- but by applying a control in the joint space

Hence you have to establish the relations between these 2 spaces

This is the purpose of Modeling:

Monday 2-5 pm, Y. Papegay

Modeling example: Kinematics

Modeling example: Kinematics

Kinematics: the relations between $\mathbf{X}$ and Θ

Modeling example: Kinematics

Kinematics: the relations between $\mathbf{X}$ and Θ

- inverse kinematics: $\mathbf{X} \longrightarrow \boldsymbol{\Theta}$

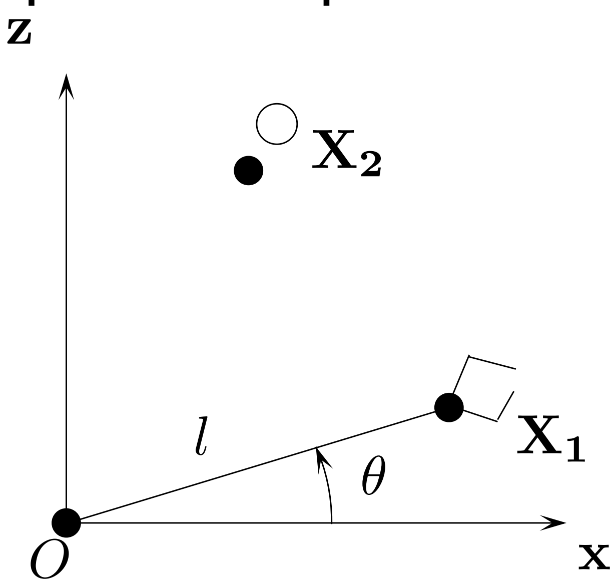
Modeling example: Kinematics

Kinematics: the relations between $\mathbf{X}$ and Θ

- inverse kinematics: $\mathbf{X} \rightarrow \mathbf{Theta}$
- direct kinematics: $\mathbf{\Theta_m} \rightarrow \mathbf{X}$

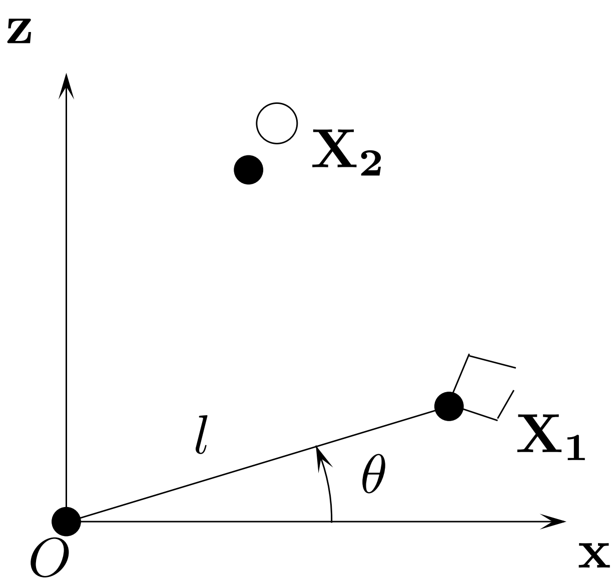
Modeling example: Kinematics

A very simple example: 1 dof planar arm



Objective: starting from the current pose of the robot $\mathbf{X_1}$ grasp the object located at $\mathbf{X_2}$

Modeling example: Kinematics

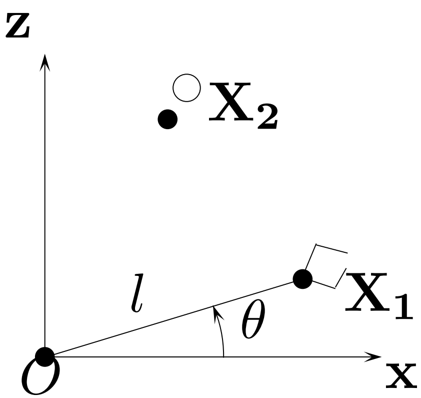


in the reference frame $\mathbf{X}$=(x,y)

direct kinematics

$$x = l \cos \theta \qquad y = \sin \theta$$

Modeling example: Kinematics



in the reference frame **X**=(x,y)

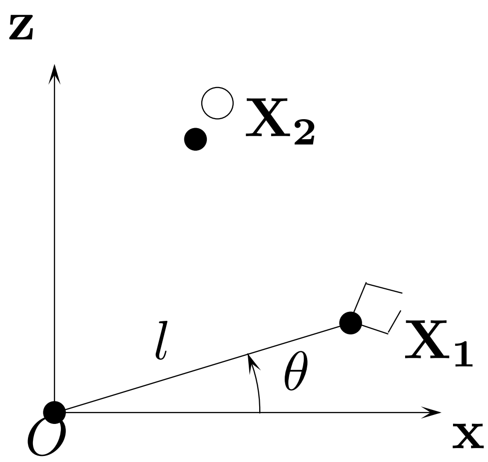
direct kinematics

$$x = l\cos\theta \qquad y = \sin\theta$$

inverse kinematics

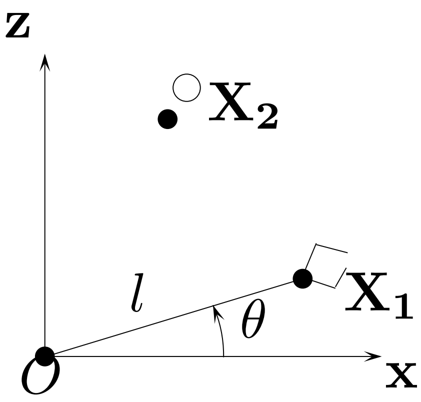
$$\theta = arctan(y/x)$$

Modeling example: Kinematics



- use the sensors to obtain θ_1
- use the inverse kinematics to determine θ_2

Modeling example: Kinematics



- use the sensors to obtain θ_1
- use the inverse kinematics to determine θ_2

Typical control law: **proportional**

Modeling example: Kinematics

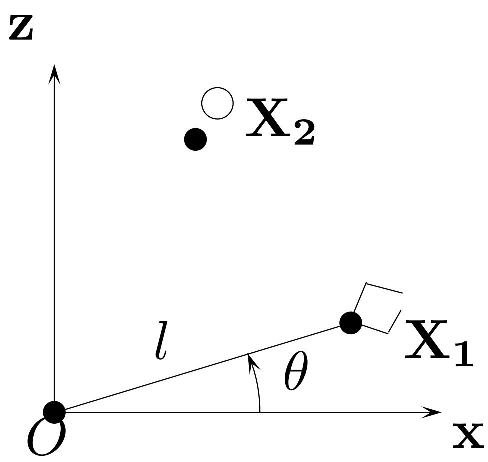
- use the sensors to obtain θ_1
- use the inverse kinematics to determine θ_2

Typical control law: **proportional**, the order θ_c (voltage, current) send to the motor at each sampling time is

$$\theta_c = K_p(\theta_2 - \theta_m)$$

where K_p is a constant gain that has to be determined, not too large, not to low

Modeling example: Kinematics

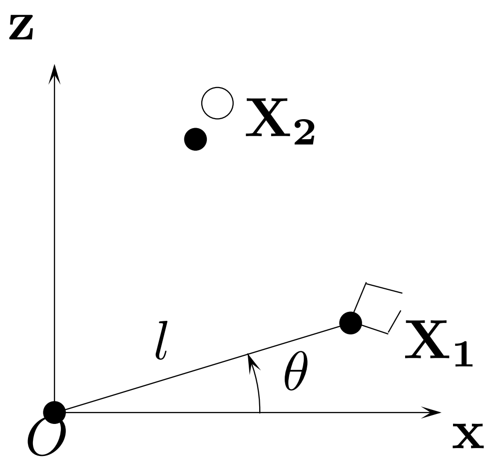


$$\theta_c = K_p(\theta_2 - \theta_m)$$

Drawback:

- when θ_m become close to θ_2, then $\theta_c \approx 0$: static error, the end-effector does not reach $\mathbf{X_2}$

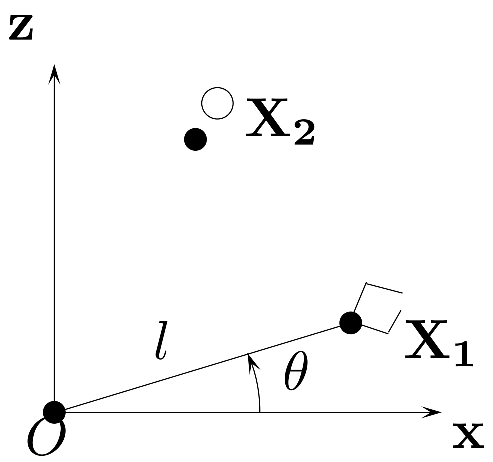
Modeling example: Kinematics



Solution: proportional-integral control

$$\theta_c = K_p(\theta_2 - \theta_m) + K_i \int (\theta_2 - \theta_m)$$

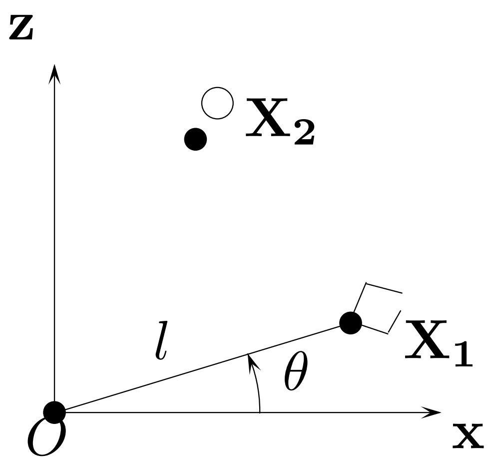
Modeling example: Kinematics



$$\theta_c = K_p(\theta_2 - \theta_m) + K_i \int (\theta_2 - \theta_m)$$

Still a drawback: at the start if θ_1 is far from θ_2 the robot may move very quickly and we may overshoot θ_2

Modeling example: Kinematics



$$\theta_c = K_p(\theta_2 - \theta_m) + K_i \int (\theta_2 - \theta_m)$$

Solution: add a derivative term that limits the initial acceleration of the robot

$$\theta_c = K_p(\theta_2 - \theta_m) + K_i \int (\theta_2 - \theta_m) - K_d\dot{\theta}$$

Modeling example: Kinematics

Note: in this example the inverse and direct kinematics are very simple. This is not always the case

- for serial structure: inverse kinematics is complex, direct kinematics is simple
- for parallel structure: inverse kinematics is simple, direct kinematics is complex
- there may be multiple solutions for both the inverse and direct kinematics

Modeling example: Kinematics

Note: in this example the inverse and direct kinematics are very simple. This is not always the case

- for serial structure: inverse kinematics is complex, direct kinematics is simple
- for parallel structure: inverse kinematics is simple, direct kinematics is complex
- there may multiple solutions for both the inverse and direct kinematics

Kinematics requires sophisticated solving methods for non linear system of equations

Modeling example: Kinematics

Associated problems:

Modeling example: Kinematics

Associated problems:

- control assumes a perfect knowledge of l

Modeling example: Kinematics

Associated problems:

- control assumes a perfect knowledge of l

the initial knowledge of l may be improved by calibration

Modeling example: Kinematics

Associated problems:

- control assumes a perfect knowledge of l

the initial knowledge of l may be improved by calibration

Here for example we may use an external measurement mean that locate the end-effector

Modeling example: Kinematics

Associated problems:

- control assumes a perfect knowledge of l

the initial knowledge of l may be improved by calibration

Here for example we may use an external measurement mean that locate the end-effector

- measure the end-effector location for various θ
- least-square estimation of l

Modeling example: Kinematics

Associated problems:

- control assumes a perfect knowledge of l

calibration: Tuesday 9-12 am, D. Daney

Modeling example: Kinematics

Associated problems:

- control assumes a perfect knowledge of l
- control assumes a perfect knowledge of $\mathbf{X_2}$

Modeling example: Kinematics

Associated problems:

- control assumes a perfect knowledge of l
- control assumes a perfect knowledge of $\mathbf{X_2}$

External mean may measure the real location of the object

Visual servoing: Wednesday 9-12am, R. Ramadour

Uncertainties

Uncertainties

A robot is a mechatronic system for which uncertainties are unavoidable

Uncertainties

A robot is a mechatronic system for which uncertainties are unavoidable

- in the modeling: the value of l in the previous example, in spite of the calibration

Uncertainties

A robot is a mechatronic system for which uncertainties are unavoidable

- in the modeling: the value of l in the previous example, in spite of the calibration
- in the environment: the location of $\mathbf{X_2}$

Uncertainties

A robot is a mechatronic system for which uncertainties are unavoidable

- in the modeling: the value of l in the previous example, in spite of the calibration
- in the environment: the location of $\mathbf{X_2}$
- in the measurement: the value of $\mathbf{\Theta_m}$

Uncertainties

These uncertainties leads to several problems

Uncertainties

These uncertainties leads to several problems

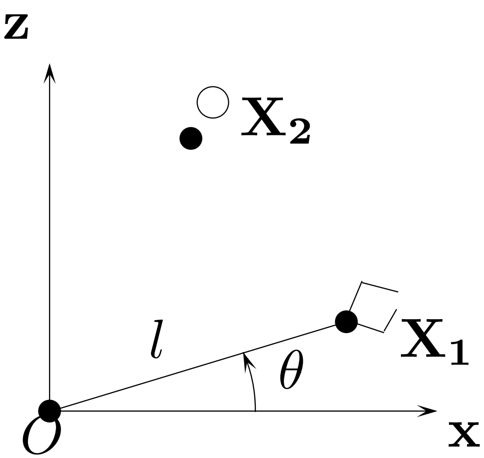
- what are the effects of the modeling and sensing on the performances of the robot: the analysis problem

Uncertainties

These uncertainties leads to several problems

- what are the effects of the modeling and sensing on the performances of the robot: the analysis problem
- what are the values of the modeling parameters that minimize these effects: the synthesis problem

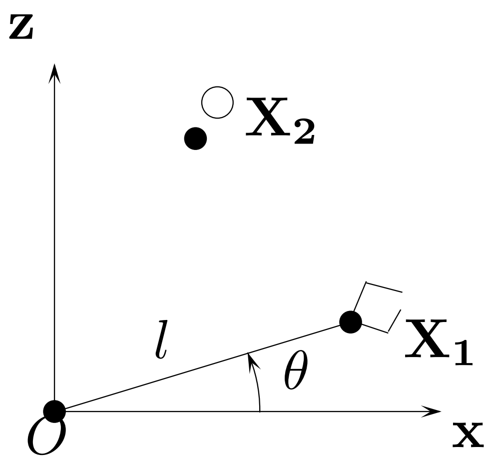
Uncertainties



We have seen that

$$x = l\cos\theta \qquad y = \sin\theta$$

Uncertainties



We have seen that

$$x = l\cos\theta \qquad y = \sin\theta$$

Hence for small errors on θ_m we have

$$\Delta x = -l\sin\theta\Delta\theta$$

$$\Delta y = l\cos\theta\Delta\theta$$

Uncertainties

Hence for small errors on θ_m we have

$$\Delta x = -l \sin\theta \Delta\theta$$

$$\Delta y = l \cos\theta \Delta\theta$$

or in matrix form

$$\Delta \mathbf{X} = \begin{pmatrix} -l\sin\theta \\ l\cos\theta \end{pmatrix} \Delta\theta$$

Uncertainties

In general for a robotics system we have

$$\Delta\mathbf{X} = \mathbf{J}(\mathbf{X}, \boldsymbol{\Theta})\Delta\theta \quad \Delta\theta = \mathbf{J}^{-\mathbf{1}}(\mathbf{X}, \boldsymbol{\Theta})\Delta\mathbf{X}$$

and $\mathbf{J}$ is called the Jacobian matrix of the robot

Uncertainties

In general for a robotics system we have

$$\Delta\mathbf{X} = \mathbf{J}(\mathbf{X}, \boldsymbol{\Theta})\Delta\theta \quad \Delta\theta = \mathbf{J}^{-\mathbf{1}}(\mathbf{X}, \boldsymbol{\Theta})\Delta\mathbf{X}$$

and $\mathbf{J}$ is called the Jacobian matrix of the robot

- as soon as the number of dof become large only one of the matrices $\mathbf{J}, \mathbf{J}^{-\mathbf{1}}$ is known in symbolic form, while the other is not

Uncertainties

In general for a robotics system we have

$$\Delta\mathbf{X} = \mathbf{J}(\mathbf{X}, \boldsymbol{\Theta})\Delta\theta \quad \Delta\theta = \mathbf{J}^{-\mathbf{1}}(\mathbf{X}, \boldsymbol{\Theta})\Delta\mathbf{X}$$

and $\mathbf{J}$ is called the Jacobian matrix of the robot

- as soon as the number of dof become large only one of the matrices $\mathbf{J}, \mathbf{J}^{-\mathbf{1}}$ is known in symbolic form, while the other is not
- these matrices are pose dependent

Uncertainties

In general for a robotics system we have

$$\Delta\mathbf{X} = \mathbf{J}(\mathbf{X}, \mathbf{\Theta})\Delta\theta \quad \Delta\theta = \mathbf{J}^{-\mathbf{1}}(\mathbf{X}, \mathbf{\Theta})\Delta\mathbf{X}$$

- $|\mathbf{J}^{-\mathbf{1}}| = 0$: even if $\Delta\theta = 0$ we have $\mathbf{\Delta X} \neq 0$ (singularity)
- $|\mathbf{J}| = 0$: even if $\Delta\mathbf{X} = 0$ we have $\mathbf{\Delta}\theta \neq 0$ (singularity)

Uncertainties

Managing uncertainties is important (imagine you are at the wrong end of a surgical robot!)

Thursday: 9-12 am, O. Pourtallier

Interval analysis

Interval analysis

There is another source of uncertainty that we have not yet mentioned . . .

Interval analysis

There is another source of uncertainty that we have not yet mentioned . . .

the computer

Interval analysis

A computer knows only a limited set of real numbers

Interval analysis

A computer knows only a limited set of real numbers

For example

- a computer does not know the number 0.1

Interval analysis

A computer knows only a limited set of real numbers

For example

- a computer does not know the number 0.1
- the closest number to 0.1 it knows are
  - 0.099999994039536,
  - 0.100000008940697

Interval analysis

A computer knows only a limited set of real numbers

A consequence is that a computer may calculate wrongly and sometimes by a very large amount: numerical round-off errors

Interval analysis

To manage this uncertainty we may use interval analysis

Interval analysis

To manage this uncertainty we may use interval analysis

Basically instead of computing with **numbers** that are wrong we calculate with **intervals** that are guaranteed to include the **exact** value

Interval analysis

We may perform with intervals the same calculation than with numbers

For example if we have two intervals $X = [a, b], Y = [u, v]$ then

$$Z = X + Y = [a + u, b + v]$$

- interval operators may be implemented so that numerical round-off errors are managed
- hence the interval Z is guaranteed to include the **exact** result of the addition of X, Y

Interval analysis

There is no free lunch!. Hence we may have to pay for this guarantee

Example: let $X = [-1, 2]$ and let us compute $X - X$

- $[-1, 2] - [-1, 2] = [-1, 2] + [-2, 1] = [-3, 3]$

Hence

$$X - X \neq 0$$

Interval analysis

Example: $F = x^2 + \cos(x)$, $x \in [0, 1]$

Problem: find $[A, B]$ such that: $A \leq F(x) \leq B \; \forall \; x \;\; \in [0, 1]$

Interval analysis

$$F = [0,1]^2 + \cos([0,1])$$

Interval analysis

$$F = [0,1]^2 + \cos([0,1])$$

Interval analysis

$$F = [0,1]^2 + \cos([0,1]) = [0,1] + \cos([0,1])$$

Interval analysis

$$F = [0,1]^2 + \cos([0,1]) = [0,1] + \cos([0,1])$$

Interval analysis

$$F = [0,1]^2 + \cos([0,1]) = [0,1] + [0.54, 1]$$

Interval analysis

$$F = [0,1]^2 + \cos([0,1]) = \text{[0,1]+[0.54,1]}$$

Interval analysis

$$F = [0,1]^2 + \cos([0,1]) = \text{[0,1]+[0.54,1]} = [\mathbf{0.54}, \mathbf{2}]$$

Interval analysis

$$F = \ [0,1]^2 \ + \cos([0,1]) \ = \text{[0,1]+[0.54,1]} = [\mathbf{0.54}, \mathbf{2}]$$

- 0 not included in [0.54,2] $\Rightarrow \ F \neq 0 \ \forall \ x \in [0,1]$

Interval analysis

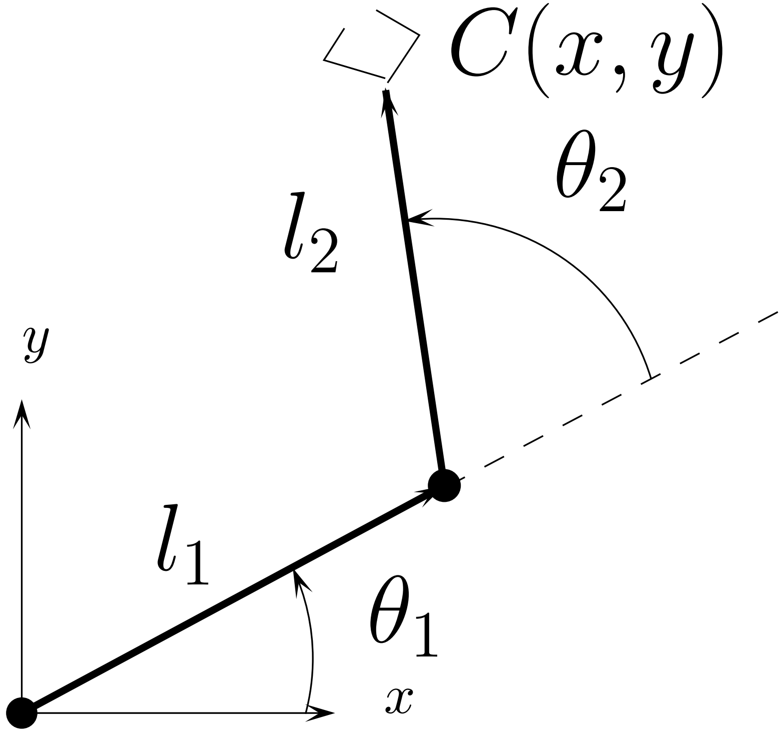
$$F = \ [0,1]^2 \ + \ \cos([0,1]) \ \ = \text{[0,1]+[0.54,1]} = \mathbf{[0.54, 2]}$$

- 0 not included in [0.54,2] $\Rightarrow \ F \neq 0 \; \forall \; x \in [0,1]$
- $F > 0 \ \ \forall \; x \in [0,1]$
- $\forall \; x \in [0,1]$ we have $0.54 \leq F \leq 2$ (global optimization)

Another kinematic example

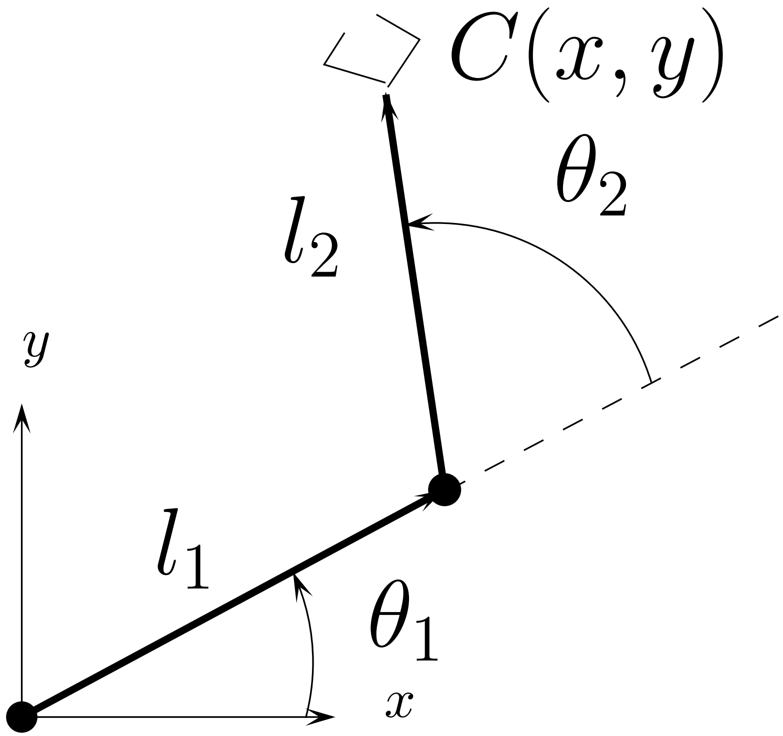
Another kinematic example

2 degrees of freedom planar robot:



- end-effector position defined by x, y
- joint variables: θ_1, θ_2

Another kinematic example



$$x = l_1 \cos \theta 1 + l_2 \cos(\theta_2 - \theta_1)$$

$$y = l_1 \sin \theta 1 + l_2 \sin(\theta_2 - \theta_1)$$

Another kinematic example

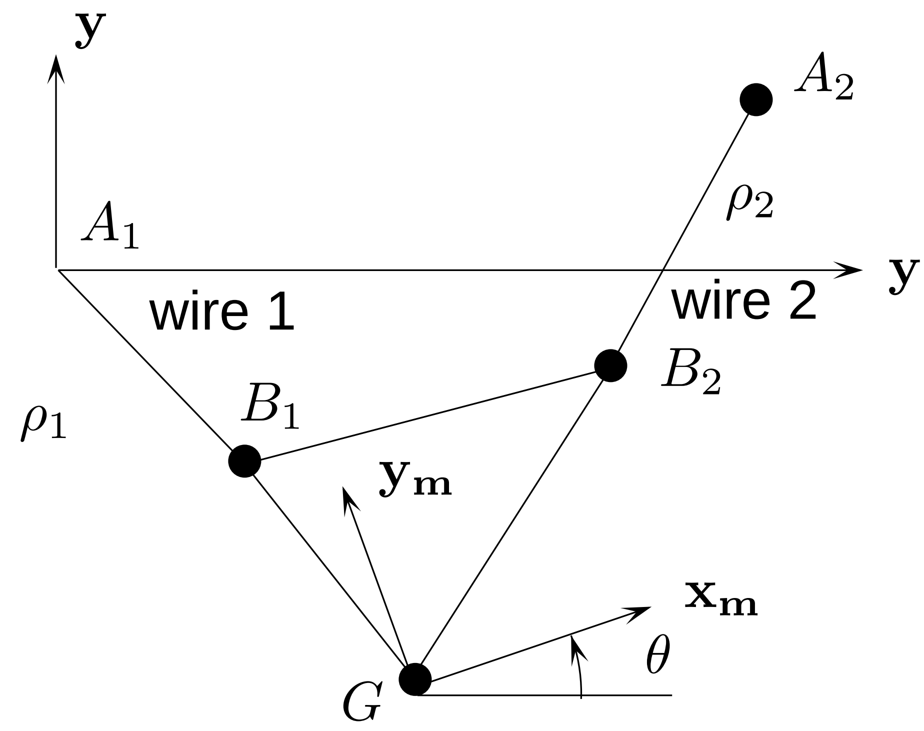
Hence

$$\begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} l_1 \sin\theta_1 - lsin(\theta_1 - \theta_2) & l_2 \sin(\theta_1 - \theta_2) \\ l_1 \cos\theta_1 - l_2 \cos(\theta_1 - \theta_2) & l_2 \cos(\theta_1 - \theta_2) \end{pmatrix} \begin{pmatrix} \Delta\theta_1 \\ \Delta\theta_2 \end{pmatrix}$$

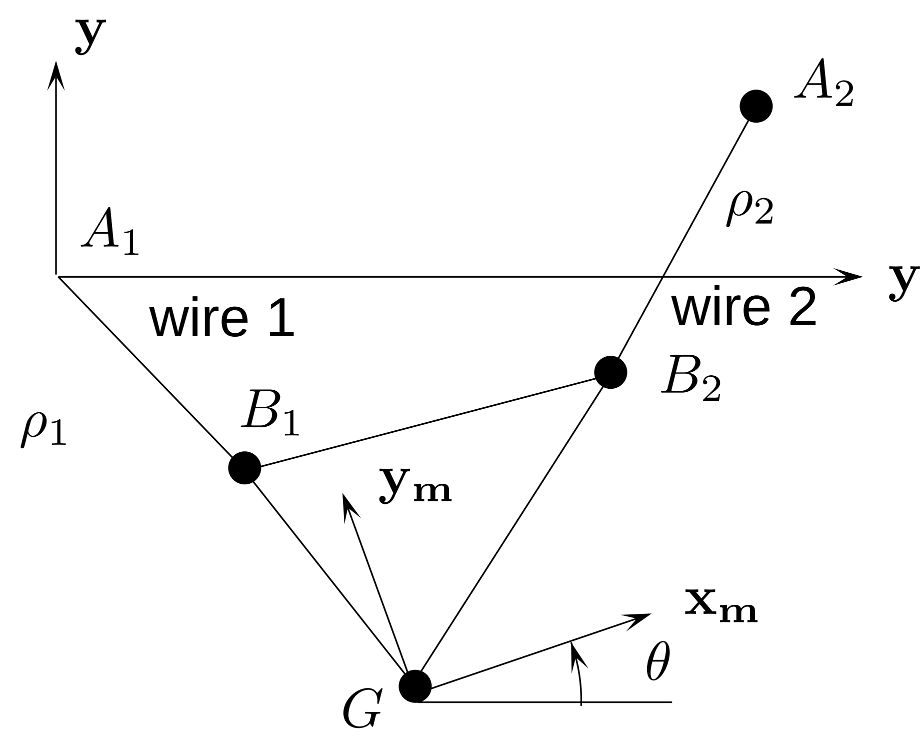
A more complex kinematic example

A more complex kinematic example

A planar wire-driven parallel robot with 2 wires

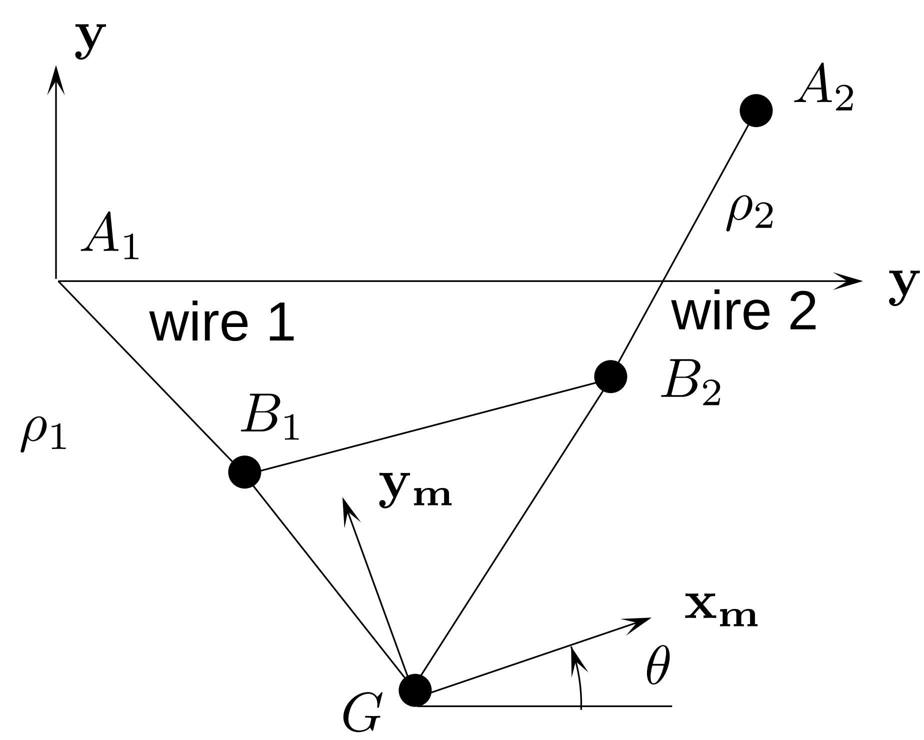


A more complex kinematic example



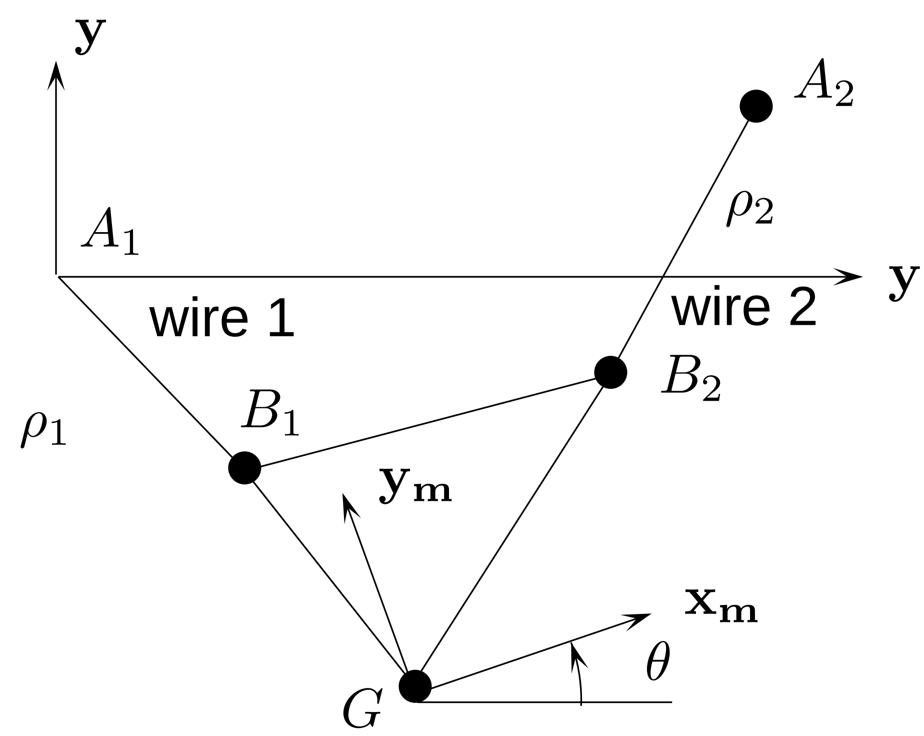
- the platform has 3 dof: x_G, y_G, θ
- we control only two joint variables ρ_1, ρ_2

A more complex kinematic example



inverse kinematics: if we give x_G, y_G, θ, then the wire lengths are easy to calculate

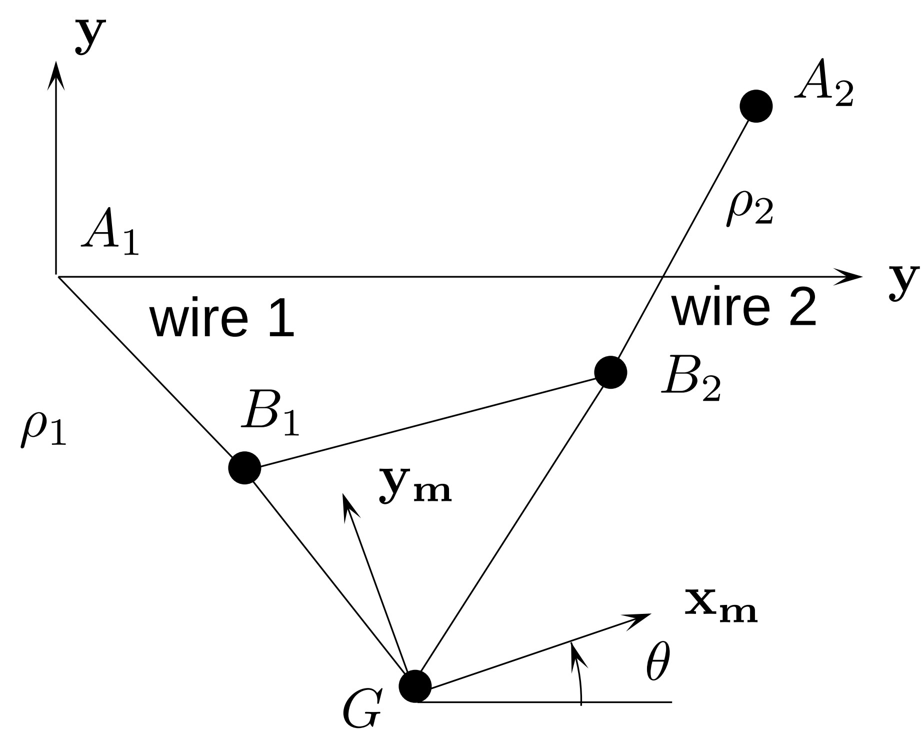
A more complex kinematic example



inverse kinematics: if we give x_G, y_G, θ, then the wire lengths are easy to calculate

But will the robot moves to the desired position ?

A more complex kinematic example



direct kinematics:

- we know ρ_1, ρ_2
- determine x_G, y_G, θ

A more complex kinematic example

direct kinematics:

- we know ρ_1, ρ_2
- determine x_G, y_G, θ

Equations

- $||A_iB_i|| = \rho_i$: 2 equations

2 equations, 3 unknowns, something is missing . . .

A more complex kinematic example

direct kinematics:

- we know ρ_1, ρ_2
- determine x_G, y_G, θ

Equations

- $||A_iB_i|| = \rho_i$: 2 equations

mechanical equilibrium

$$\mathcal{F} = \mathbf{J^{-T}}\tau$$

3 equations, 2 more unknowns

A more complex kinematic example

direct kinematics:

- 5 unknowns: $x_G, y_G, \theta, \tau_1, \tau_2$
- 5 equations: $||A_iB_i|| = \rho_i, \mathcal{F} = \mathbf{J}^{-\mathbf{T}}\tau$

A more complex kinematic example

direct kinematics:

- 5 unknowns: $x_G, y_G, \theta, \tau_1, \tau_2$
- 5 equations: $||A_iB_i|| = \rho_i, \mathcal{F} = \mathbf{J}^{-\mathbf{T}}\tau$

Result

- there cannot be more than 24 solutions
- these solutions may be obtained by solving two 12-th order univariate polynomial
- up to now only examples with 8 solutions have been found

Conclusions

Conclusions

Robotics is very multidisciplinary field that involves numerous other scientifi domains:

- mechanism science
- sensors and actuators
- electronic
- computer science
- mathematics: system solving, geometry
- control theory

We hope you will enjoy this module!