

Interval analysis, optimisation and robotics

J-P. Merlet

COPRIN Project

INRIA Sophia-Antipolis



1 The COPRIN project

- INRIA: National Research Institute in Computer Science and Control Theory
- COPRIN: a team of 6 researchers + 7 PhD/Postdoc/Engineers, created in 2002

Purpose: solving problems with **interval analysis**, **constraints programming**, **numerical analysis** and **symbolic computation** for application domains such as:

- mechanism theory
- molecular chemistry
- network analysis
- computer vision
- quantum mechanics
- control theory
- ...

using our libraries **Alia**s, **IcosAlia**s



2 Mathematical viewpoint on robotics problems

Most robotics problems can be basically reduced to:

- **systems solving** problems (involving non-linear equations or inequalities, differential or integral equations)

- **constrained optimization** problems (minimum, maximum, minmax)

- **linear algebra** problems

BUT

- most of them have specificities that must be addressed for efficiency and completeness
- functions involved in these problems are complex \Rightarrow **symbolic computation**

- results must be **certified** (e.g. for safety reasons in medical robotics)

- very large problems \Rightarrow **distributed implementation**

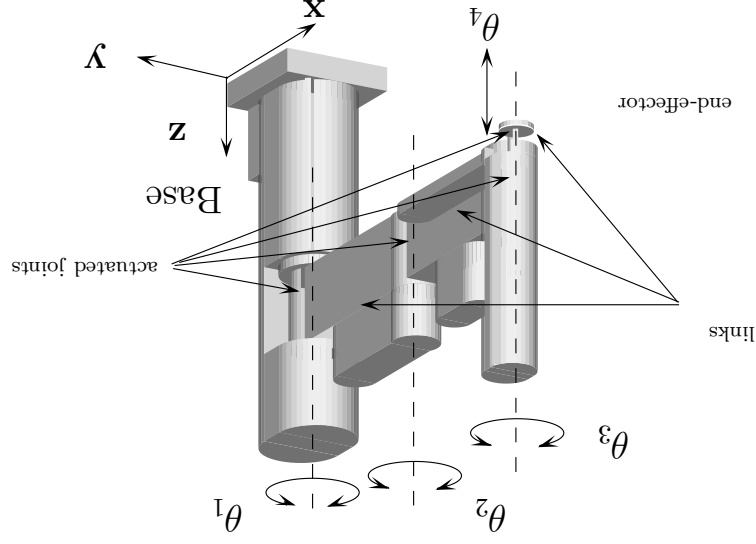
3 What is a robot manipulator?

Manipulator: a mechanical system whose purpose is to move a particular body (the "hand") to a specific pose by controlling the relative motion of internal bodies through "actuators"

- **X:** parameters describing the pose of the hand (**generalized coordinates**)
- **Θ :** parameters describing the motion of the actuators (**joint variables**)

$$\mathbf{X} = \{x, y, z, \theta_z\}$$

$$\Theta = \{\theta_1, \theta_2, \theta_3, \theta_4\}$$



4 degrees-of-freedom SCARA robot

4 Kinematics

Needed for controlling the robot

- the robot sensors give **joint variable measurements** Θ^m
- for control we must have

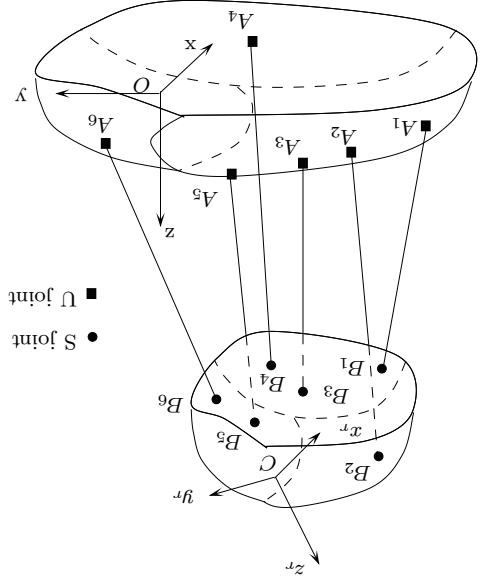
- **inverse kinematics:** $\mathbf{X} \mapsto \Theta = G(\mathbf{X})$
- **direct kinematics:** $\Theta^m \mapsto \mathbf{X} = F(\Theta^m)$

Example: parallel robot

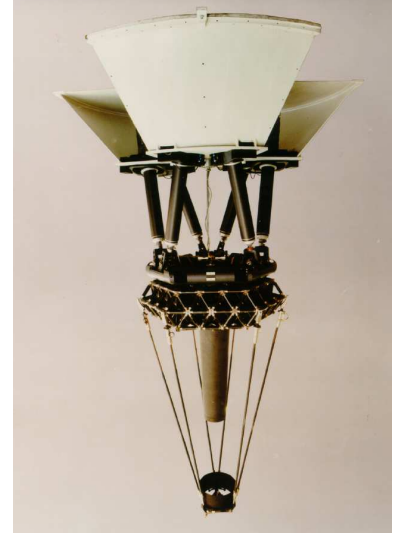
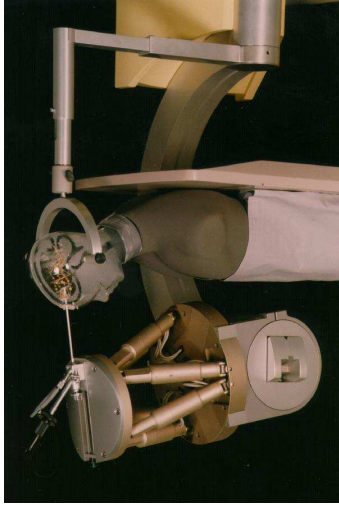
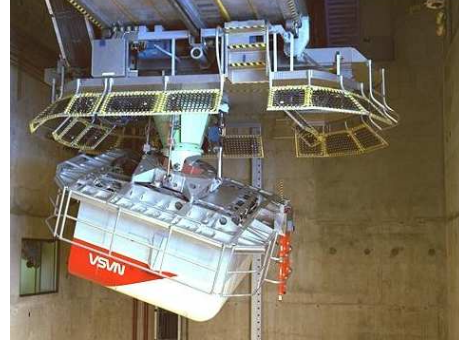
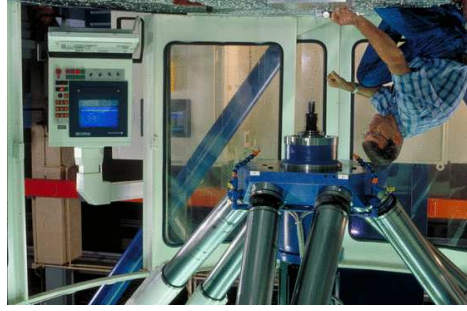
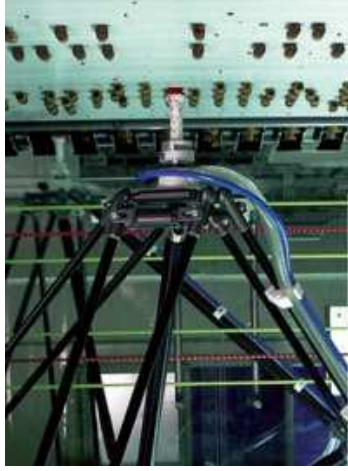
$$\mathbf{X} = \text{OC}, \mathbf{R}(x_r, y_r, z_r \mapsto x, y, z)$$

$$\Theta = \text{leg lengths}$$

We have an explicit inverse kinematics $\Theta = G(\mathbf{X})$



• S joint
 ■ U joint



Direct kinematics

Problem: being given Θ^m find the *actual pose* of the platform

actual pose: pose of the platform when the measurement was performed

Method: solve the inverse kinematic equations $G(\mathbf{X}) = \Theta^m$

Background:

- this system has **multiple real solutions** (here up to 40)
- inverse kinematics leads to a system of 6 to 12 equations, according to the choice of the parameters that are used to define the pose of the platform
- **unknowns are bounded**: $\|\mathbf{OC}\| \leq \|\mathbf{OA}\| + \Theta + \|\mathbf{CB}\|$
- **additional constraints** may have to be considered (limited motion of the passive joints)

There are two different types of direct kinematics problem:

1. **off-line**: find the current pose without any a-priori information on it
Computation time is not critical, within limits

2. **real-time**: determine as quickly as possible the actual pose that should not be "too far"
from a previously computed pose.

- Computation time should be less than the robot controller sampling time (2 to 5 ms)
- Solutions must be certified and singular or multiple solutions must be detected

4.1 Off-line direct kinematics

- No a-priori information on the actual pose of the platform
- Calculation of **all** solutions

Possible methods

- | | | |
|---|---|------------------------------------|
| <ul style="list-style-type: none"> • elimination : numerically unsafe • homotopy : (≥ 30 minutes) • Groebner basis : certified,
<i>fastest method : Fg_B, between 1s and 30s</i> | } | Algebraic
formulation |
| <ul style="list-style-type: none"> • interval analysis | } | formulation
independent |

Algorithm

interval analysis second fastest method: based on **distance equations**

$$(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 = d_{ij}^2 \rightarrow \text{known distance}$$

- **2B and 3B** constraint propagation

- **specific Neumaier and Kantorovich exclusion test**

◦ determine a ball with a unique solution, with a method to compute this solution
 ◦ detect multiple solutions

- **computation time**: 10-60s

but interval analysis is the only method whose computation time may be reduced if

- additional constraints are taken into account

- there are constraints on the search space

But we have still to find an algorithm that will determine the actual pose

4.2 "Real time" direct kinematics

Usual approach: Newton method

BUT Newton is not safe and may leads to the wrong solution with a **critical control effect**

Safer approach: Newton + interval analysis

\mathbf{X}_l : last known position + known maximal velocity of the robot

$$\mathbf{X} \in \mathcal{B} = [\mathbf{X}_l - \epsilon, \mathbf{X}_l + \epsilon]$$

↑

1. run **Newton** with the center of the current box as guess

2. if **Newton** does not converge or converge outside \mathcal{B} , bissect and reiterate

3. if **Newton** converge toward $\mathbf{X}_n \in \mathcal{B}$ use **Kantorovitch, Neumaier** to

• **certify** the solution \mathbf{X}_n and find \mathcal{B}_k that includes only \mathbf{X}_n

4. if $\mathcal{B} \subset \mathcal{B}_k \rightarrow$ **certified solution** otherwise bissect and reiterate

5. if more than one solution or singular jacobian \rightarrow **STOP!**

Properties:

- certified Newton can deal with additional constraints (limits on joint motion)
- computation time Newton: from 1.8 to 5 μs **when convergent**
- computation time interval scheme: 300 to 400 μs , well within controller sampling time
- **distributed calculation**

⇒ **Certified Newton scheme**

5 Inverse kinematic jacobian matrix

$$\Theta = G(\mathbf{X})$$

↑↑

- velocity relation: $\dot{\Theta} = J^{-1}(\mathbf{X})\dot{\mathbf{X}}$

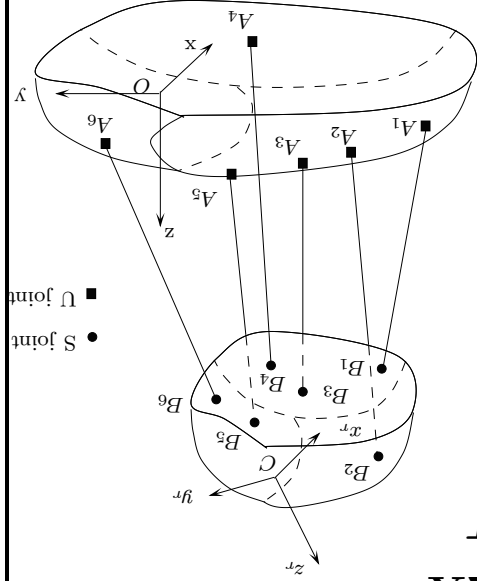
- accuracy relation: $\Delta\Theta = J^{-1}(\mathbf{X})\Delta\mathbf{X}$

- static relation: $\mathcal{F} = J^{-T}(\mathbf{X})\tau$

J^{-1} : inverse kinematic jacobian

$$\frac{A_i B_i}{CB_i \times A_i B_i} \left(\frac{\|A_i B_i\|}{\|A_i B_i\|} \right)$$

→ Plücker vector of the lines associated to the legs



■ U joint
● S joint

Example of problems: **accuracy analysis**

$$\Delta_{\Theta^m} = J^{-1}(\mathbf{X})\Delta\mathbf{X}$$

for bounded Δ_{Θ^m} ($|\Delta_{\Theta^m}| \leq 1$) shows that $\Delta X_i \leq \epsilon_i$ for any \mathbf{X} in a given workspace

classical problem of computing an outer approximation of the solutions of a **linear interval system** $\mathbf{A}\mathbf{y} = \mathbf{b}$ with $\mathbf{A} = J^{-1}(\mathbf{X})$, $\mathbf{y} = \Delta\mathbf{X}$, $\mathbf{b} = \Delta_{\Theta}$

BUT

typical of an application problem: $A_{ij} = f_{ij}(\mathbf{X})$ with \mathbf{X} an interval vector

classical interval methods do not take into account the dependency between the f_{ij}

↑

large overestimation of the solution domains

⇒ taking into account the derivatives of the f_{ij} with respect to \mathbf{X} in the Gaussian elimination scheme leads to an algorithm which is 10 to 100 faster than the classical methods

Problems:

shows that $|j_{-1}(\mathbf{X})| \neq 0$ for any \mathbf{X} in a given workspace (no **singularity**)

$$j^{-1} = \left(\frac{A_i B_i}{CB_i \times A_i B_i} \right) \left(\frac{\|A_i B_i\|}{\|A_i B_i\|} \right)$$

\Rightarrow singularity depends also on the location of \mathbf{A} , \mathbf{B} which are uncertain

Example of problems: **Singularity detection**

$$\mathcal{F} = j_{-T}(\mathbf{X})^T \uparrow \begin{matrix} | \\ \dots \\ | \end{matrix} = \frac{T_i}{|j_{-T}|}$$

\Rightarrow if j_{-T} singular, then $T_i \rightarrow \infty \Rightarrow$ **robot breakdown**

Algorithm

- find a pose in the workspace and compute numerically the sign of $|J^{-1}|$ (say > 0)
- bisection algorithm to determine if a pose with $|J^{-1}| > 0$ exists in the workspace
 - if yes singularity exists: **SINGULARITY, EXIT**
 - otherwise no singularity in the workspace

Checking the regularity of an interval matrix

$$\mathbf{A} = \begin{pmatrix} x & y \\ x & 2y \end{pmatrix}$$

with x, y in $[1, 2]$. Clearly \mathbf{A} does not include singular matrix as $|\mathbf{A}| = xy$

we can compute an interval evaluation of the determinant (direct expansion by row or column, Gaussian elimination)

$$\mathbf{A}_I = \begin{pmatrix} [1, 2] & [1, 2] \\ [1, 2] & [2, 4] \end{pmatrix} \Rightarrow |\mathbf{A}_I| \in [-2, 7] \Rightarrow \text{no direct conclusion}$$

Improvement 1: extremal matrices

Extremal matrices: all scalar matrices with as elements either the lower or upper bound of the corresponding elements in interval matrix

$$\mathbf{A}_I = \begin{pmatrix} [1, 2] & [1, 2] \\ [1, 2] & [2, 4] \end{pmatrix} \Leftrightarrow \mathbf{A}_1^e = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}, \mathbf{A}_2^e = \begin{pmatrix} 1 & 2 \\ 1 & 2 \end{pmatrix}, \mathbf{A}_3^e = \begin{pmatrix} 1 & 2 \\ 2 & 2 \end{pmatrix}, \dots$$

2^{n^2} extremal matrices

Theorem: if determinant of all extremal matrices have same sign \Rightarrow no singular matrices in \mathbf{A}_I

Improvement by Rex, Rohm (1998): not **all** extremal matrices should be considered, only 2^{n-1} **but** the set of matrices to test includes

$$\mathbf{A}_1 = \begin{pmatrix} 1 & 1 \\ 1 & 4 \end{pmatrix} \quad \mathbf{A}_2 = \begin{pmatrix} 1 & 2 \\ 2 & 2 \end{pmatrix} \quad \mathbf{A}_3 = \begin{pmatrix} 1 & 2 \\ 1 & 2 \end{pmatrix}, \dots$$

$\mathbf{A}_1, \mathbf{A}_2$ have a determinant of opposite sign, \mathbf{A}_3 is singular \Rightarrow **no direct conclusion**

Improvement 2: pre-conditioning

Numerical pre-conditioning: $\|\mathbf{AK}\| = \|\mathbf{A}\|\|\mathbf{K}\|$

Use $\mathbf{K} = \mathbf{A}^{-1}(\text{Mid}(\mathbf{X}))$ as pre-conditioning matrix to get \mathbf{AK} "close" to identity

$$\mathbf{K} = \begin{pmatrix} 4/3 & -2/3 & 2/3 \\ -2/3 & 2/3 & 2/3 \\ 0, 2 & [-4/3, 0] & [2/3, 2] \end{pmatrix} \quad \mathbf{AK} = \begin{pmatrix} [0, 2] & [-2/3, 2/3] \\ [-4/3, 0] & [2/3, 2] \end{pmatrix}$$

$$\|\mathbf{K}\| = 4/9$$

$\|\mathbf{AK}\| = [-8/9, 44/9] \Rightarrow$ **improvement** on the evaluation width but **no direct conclusion**

Symbolic/numeric pre-conditioning:

off-line

- keep \mathbf{K} symbolic
- compute \mathbf{AK} symbolically
- re-arrange the elements of \mathbf{AK} to decrease the dependency problem

on-line

- plug the numerical values of the element of \mathbf{K} in the symbolic \mathbf{AK}

$$\begin{aligned}
 \text{(off - line) } \mathbf{AK} &= \begin{pmatrix} xk_{11} + xk_{21} & yk_{11} + 2yk_{21} \\ xk_{12} + xk_{22} & yk_{12} + 2yk_{22} \end{pmatrix} \Rightarrow \mathbf{AK} = \begin{pmatrix} x(k_{11} + k_{21}) & y(k_{11} + 2k_{21}) \\ x(k_{12} + k_{22}) & y(k_{12} + 2k_{22}) \end{pmatrix} \\
 \text{(on - line) } \mathbf{K} &= \begin{pmatrix} 4/3 & -2/3 \\ -2/3 & 2/3 \end{pmatrix} \Rightarrow \mathbf{AK} = \begin{pmatrix} 2/3[1, 2] & 0 \\ 0 & 2/3[1, 2] \end{pmatrix} \\
 |\mathbf{AK}| &= [4/9, 16/9] \Rightarrow \text{no singular matrix}
 \end{aligned}$$

Improvement 3: better extremal matrices (Donelan, Merlet, 2005)

- assume that some elements of some rows are linearly dependent of some subset of parameters $\{x_i, \dots, x_j\}$

- construct all matrices H whose rows are independently obtained by setting x_i, \dots, x_j to their extremal values

Theorem: if all the determinants of the matrices in H have same sign \Rightarrow no singular matrices in A

$$\mathbf{A} = \begin{pmatrix} x & x \\ y & 2y \end{pmatrix} \left| \begin{array}{l} \rightarrow \text{linear in } x \\ \rightarrow \text{linear in } y \end{array} \right. \mathbf{A}_1 = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \quad \mathbf{A}_2 = \begin{pmatrix} 1 & 1 \\ 2 & 4 \end{pmatrix} \quad \mathbf{A}_3 = \begin{pmatrix} 2 & 2 \\ 1 & 2 \end{pmatrix} \quad \mathbf{A}_4 = \begin{pmatrix} 2 & 2 \\ 2 & 4 \end{pmatrix}$$

all determinants have the same sign \Rightarrow no singular matrix

Orientation ψ, θ, ϕ	Regular	Regular+Rohn	$\mathbf{J}^{-1}\mathbf{K}$	$\mathbf{J}^{-1}\mathbf{K} + \text{Rohn}$	$\mathbf{K}\mathbf{J}^{-1}$
[-1,1]	106.74	0.37	1.09	0.39	0.01
[-2,2]	338.9	0.4	4.3	0.42	0.01
[-5,5]	9076.2	2.6	34.79	2.8	0.01

No uncertainty on \mathbf{A}, \mathbf{B}

Example: robot with base radius 13, platform radius 8, workspace $x, y \in [-5, 5], z \in [45, 50]$

- check over a workspace for \mathbf{X}
 - allow uncertainties on the coordinates of \mathbf{A}, \mathbf{B}
 - if their range width is small keep them in the matrix element
 - otherwise add these coordinates as new unknowns
- Now we have a large panel of methods to check the regularity of $\mathbf{J}^{-\mathbf{T}}$

Interval analysis is the only known method allowing a certified singularity verification even with uncertainties on the robot modeling

Uncertainty on A,B	$x_c, y_c \in [-5, 5]$ $z_c \in [45, 50]$ $\psi, \theta, \phi \in [-5^\circ, 5^\circ]$	$x_c, y_c \in [-5, 5]$ $z_c \in [45, 50]$ $\psi, \theta, \phi \in [-15^\circ, 15^\circ]$	$x_c, y_c \in [-15, 15]$ $z_c \in [45, 50]$ $\psi, \theta, \phi \in [-15^\circ, 15^\circ]$
(D) $\epsilon = \pm 0.05$	0.01	0.23	5.5 (7.32 with Rohn)
(V) $\epsilon = \pm 0.05$	0.01	0.63	14.07 (4.54 with Rohn)
(D) $\epsilon = \pm 0.1$	0.01	4.47	1540.74 (514.5 with Rohn)
(V) $\epsilon = \pm 0.1$	0.02	2.55	2614.55 (402.2 with Rohn)

- (D): as it in the matrix elements
- (V): added as new variables
- with \mathbf{KJ}^{-1} pre-conditioning

With uncertainties on **A,B**

Examples of problems: Motion verification

Data:

- a trajectory for the robot defined by arbitrary analytical time-function $\mathbf{X}_i = f_i(t)$, $t \in [0, 1]$
- possible bounded errors on \mathbf{A}, \mathbf{B}
- possible bounded control errors $\mathbf{X}_i^r = f_i^r(t) + \epsilon_i$

Objective: check that the real trajectory followed by the robot is **valid**

- the trajectory lie within the workspace of the robot: $\Theta_{min} \leq \Theta(t) \leq \Theta_{max}$ $\forall t \in [0, 1]$
- trajectory is singularity-free
- possibly has to verify other constraints: $\mathbf{G}(\mathbf{X}(t)) \leq 0$

for most usual trajectory the verification is almost real-time

Examples of problems: **Workspace analysis**

Workspace: set of all poses that can be reached by the hand being constraints on the mechanism

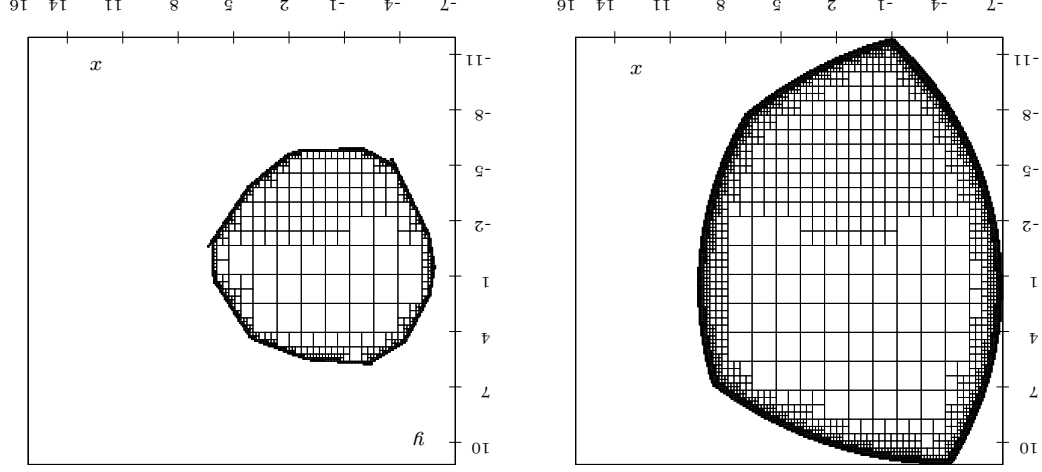
Constraints examples:

$$\rho_{min} \leq G(\mathbf{X}(t)) \leq \rho_{max}$$

$$\frac{AB.n}{AB} \geq \cos \alpha$$

We are dealing with a n dimensional variety and interval analysis is able to provide an

approximation of the variety



Examples:

cross-section in the $x - y$ plane of all poses for which all orientation angles in $[-5^\circ, 5^\circ]$ are allowed

Workspace analysis with additional constraints

Workspace = $\{\mathbf{X}\}$ satisfying kinematic constraints + performance constraints

Examples: if λ_i are the eigenvalues of $J^{-1}J^{-T}$ then

Workspace = $\{\mathbf{X}\}$ satisfying kinematic constraints + $a_i \leq \lambda_i(\mathbf{X}) \leq b_i$

For a box in the workspace:

• $J^{-1}J^{-T}$ is an **interval matrix**

• characteristic polynomial P of $J^{-1}J^{-T}$ has **interval coefficients** $U_i(\mathbf{X})$

Bounds $[\underline{a}, \underline{a}]$ on the eigenvalues of $J^{-1}J^{-T}$:

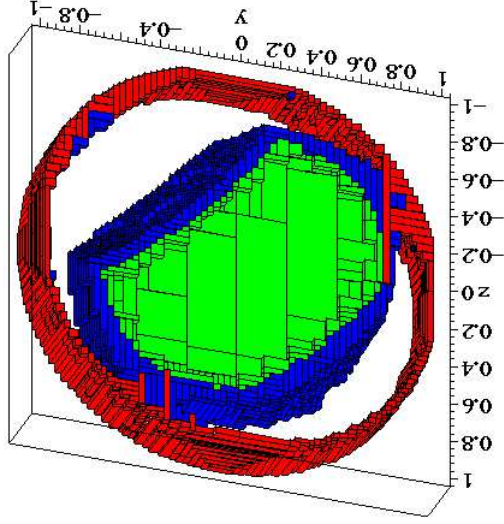
• directly from $J^{-1}J^{-T}$: **Gerschgorin circles, Cassini ovals**

• from the $U_i(\mathbf{X})$: **algebraic geometry theorems**

adapted for
interval coefficients

Note: also useful in control theory

Quality index: $1/4 \leq \text{eigenvalues of } J^{-1}J^{-T} \leq 4$



Example: the Orthoglide robot, 3 dof translation robot



Examples of problems: **Optimal design**

Objectives: find the design parameters of a robot such that it satisfies a list of requirements

Usual approach: **minimization of a weighted cost function:**

$$\sum_i \underbrace{w_i}_{\text{weight}} \underbrace{I_i}_{\text{performance index}} \in [0, 1]$$

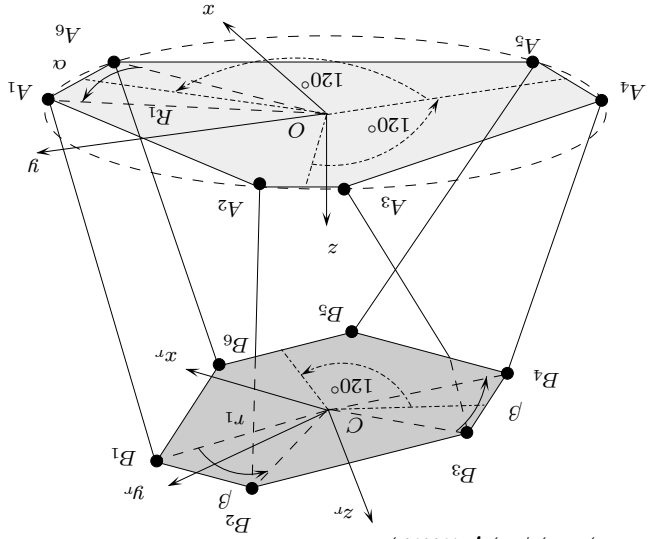
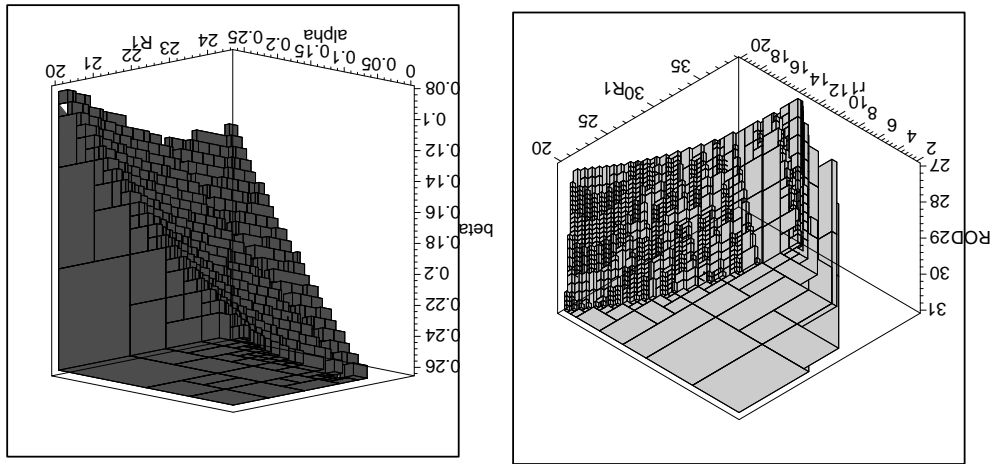
Drawbacks:

- difficulty to figure out the cost function
- result very sensitive to the weights
- provides only one design solution: **but** the designer is usually not aware of all design constraints

- real robot will differ from the theoretical solution (manufacturing tolerances)

Interval analysis

determination of an approximation of **all** parameters values so that the robot satisfies one requirement



Example: Gough platform with design parameters $\mathcal{P} : R_1, r_1, \alpha, \beta, \rho_{min}, S$

Objectives: find the design parameters

such that a set of given poses $\{X_1, \dots, X_n\}$ are

included in the workspace of the robot

$$\rho_{min} \leq G(X_i) \leq \rho_{min} + S \quad \frac{\|AB\|}{AB_n} \geq \cos \alpha$$

deployment movie



wire robot

- only an **approximation** is obtained, box with width lower than twice the manufacturing tolerances are neglected \Rightarrow **robust design solutions**
- computing the intersection of the region obtained for all requirement allows to determine **all design solutions**

6 Conclusion: what I learn

- practical applications specificities \Rightarrow **software flexibility**
- theorems from other fields must be adapted for intervals: linear algebra, geometry
- practical interval analysis applications are embedded in a complex process



few users will accept to learn new languages for using interval analysis
 \Rightarrow better to offer programming interface through **general-purpose scientific languages**
 (Matlab, Scilab, Maple, Mathematica)

- **symbolic computation must** be used to improve the efficiency of the solving
- interval analysis should not be presented as a **black box** because its efficiency is too variable
- **tutorial** for as many problems as possible should be made available, **interval movie**
- **parallel implementation** should be generalized

Try our on-line interval solvers: <http://www.inria.fr/coprin>