Inverse Kinematics and Path Planning of Manipulator Using Real Quantifier Elimination Based on Comprehensive Gröbner Systems

Mizuki Yoshizawa, Akira Terui, and Masahiko Mikawa
University of Tsukuba, Tsukuba, Japan

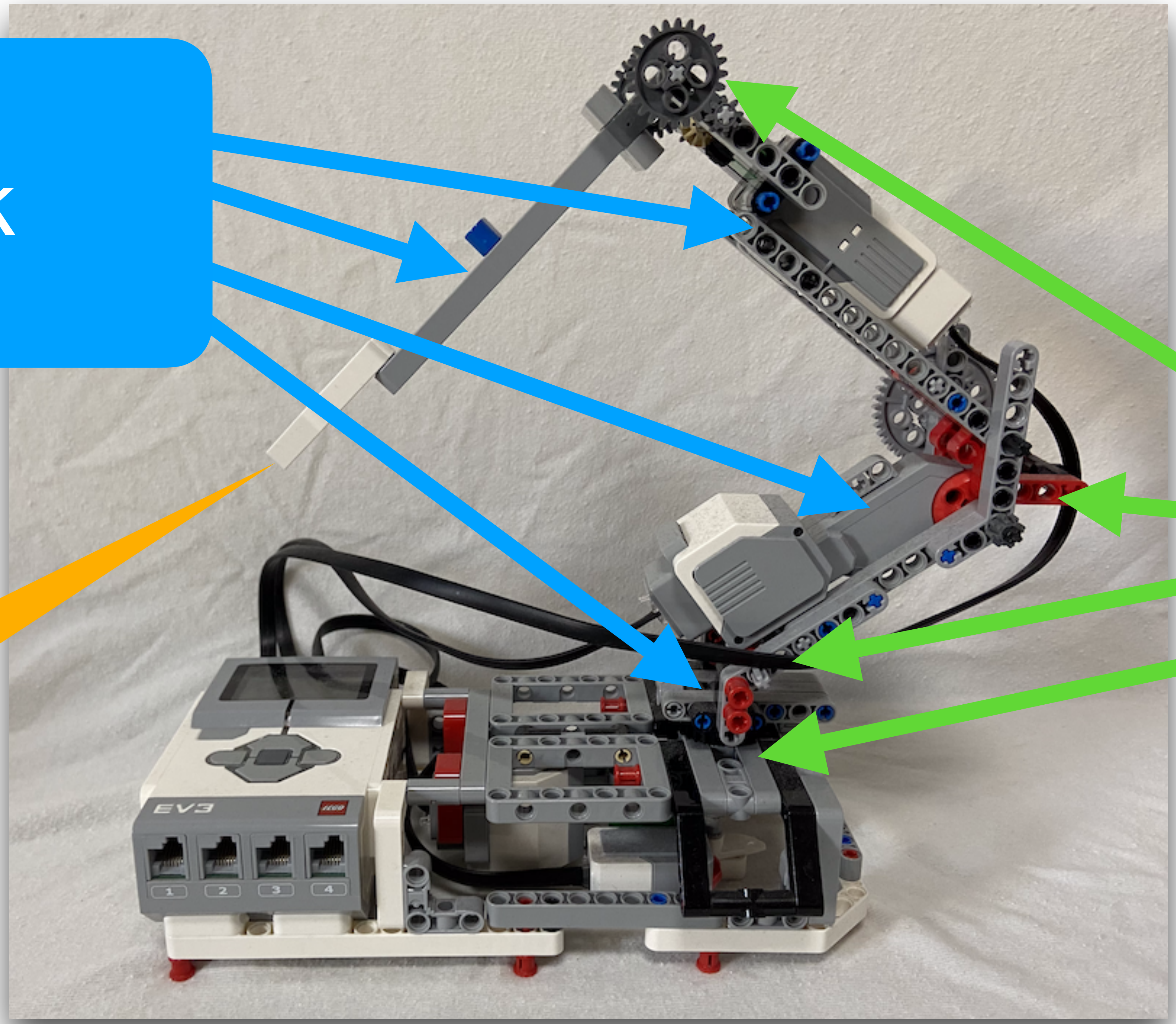
Computer Algebra in Scientific Computing: CASC 2023
August 31, 2023

An example of robot manipulator of 3 DOF

(LEGO MINDSTORMS EV3)

Degree of freedom

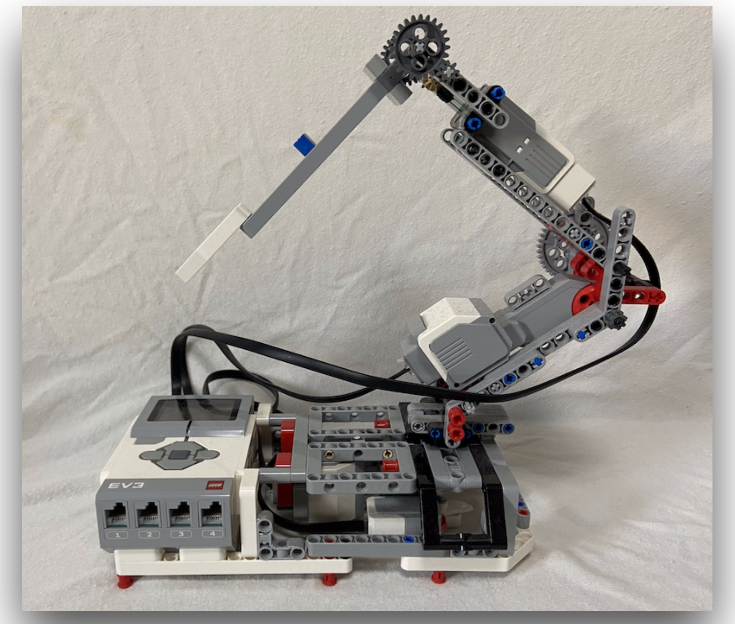
Segment / Link



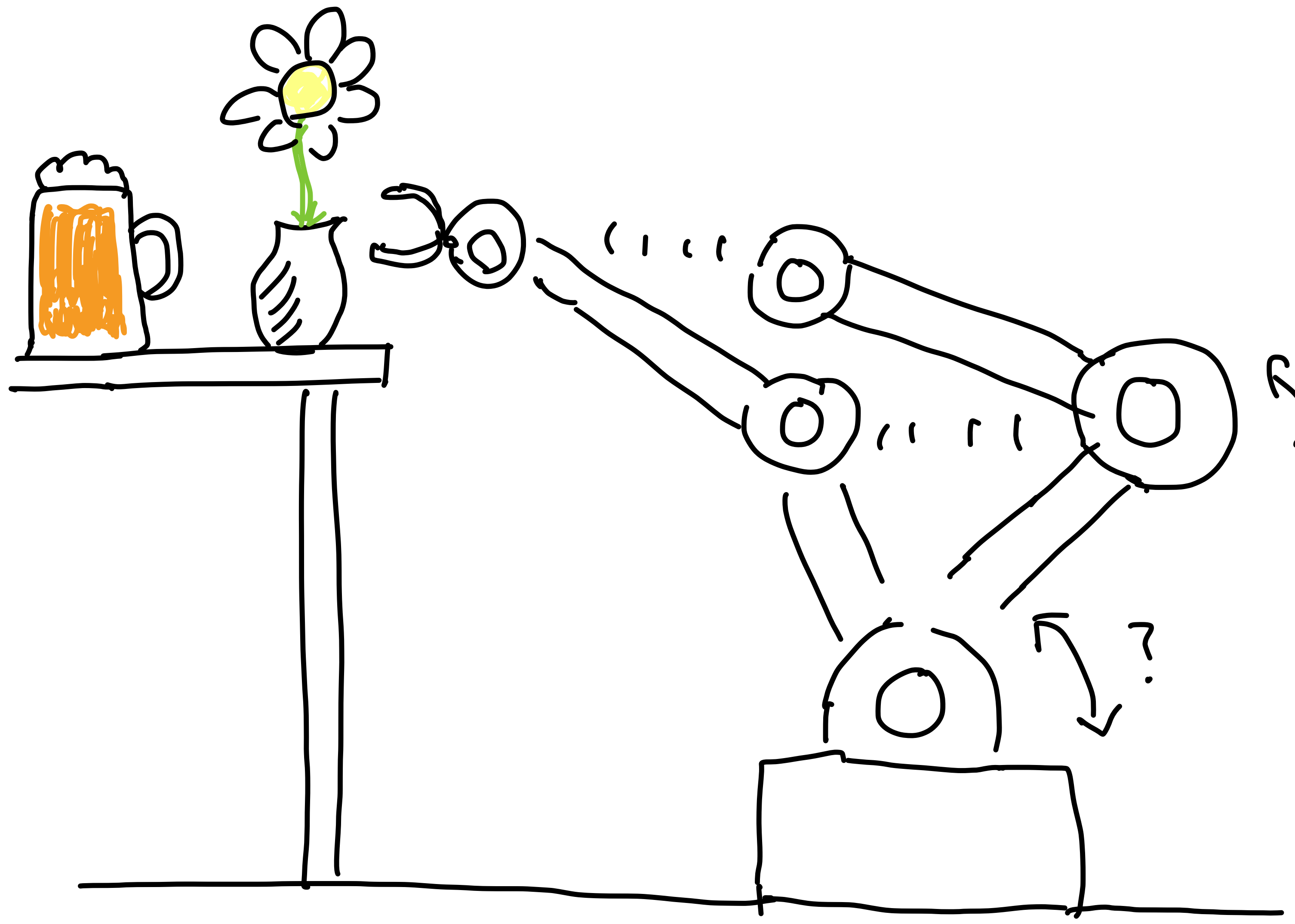
Joint

End-effector

Motion planning of a robot

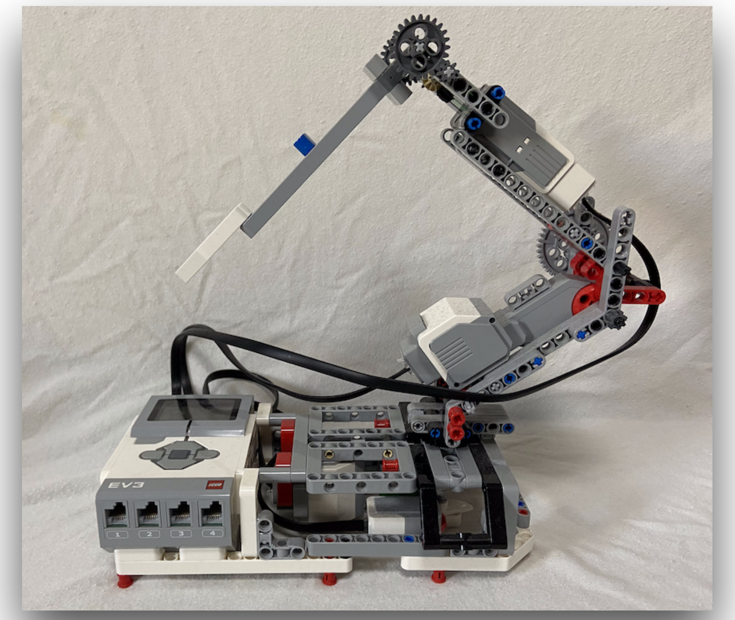


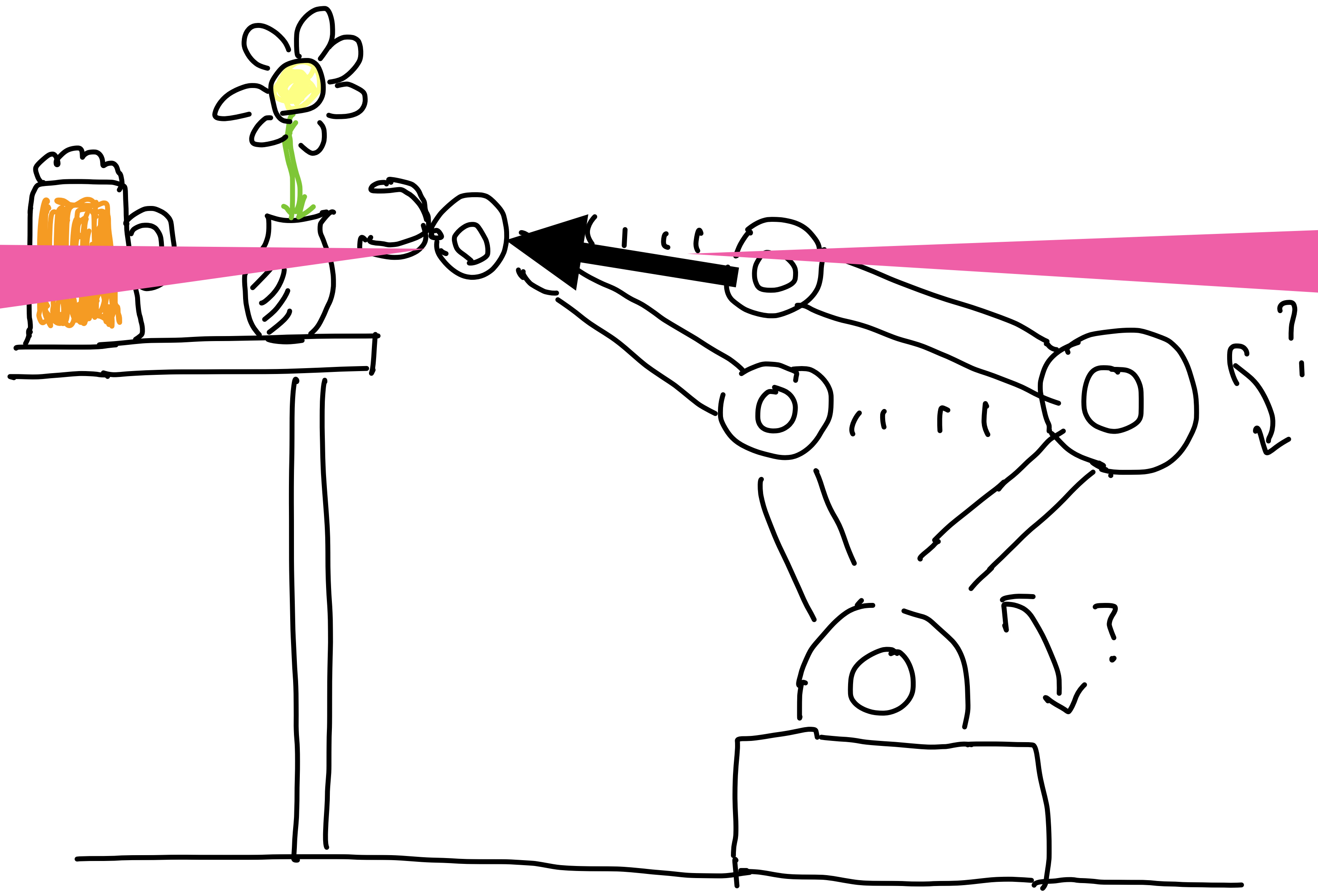
How to move the manipulator (the end-effector) from the initial position to a desired (given) position? (The problem and a calculation)



What we investigate

Inverse kinematic problem and path planning problem

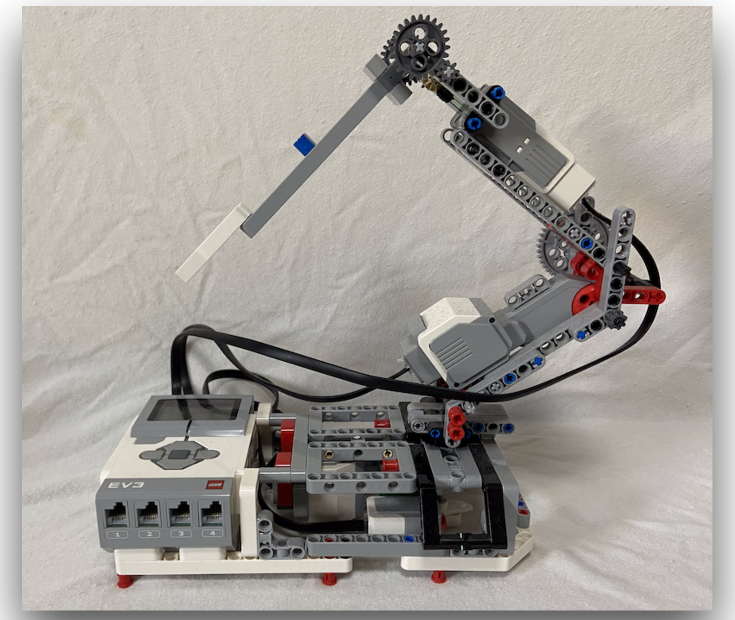




Inverse kinematic problem: verify if the end-effector can be located in the desired position → calculate corresponding algles of the joints

Path planning problem: verify if the end-effector can be moved along the given path → calculate a series of corresponding algles of the joints

You may know…



Inverse kinematic problems have been frequently solved with Gröbner basis computation

A system of polynomial equations; $I = \langle f_1, f_2, f_3, f_4 \rangle$

The Gröbner basis of I w.r.t. a certain monomial ordering

$$\begin{cases} f_1(x,y,z,w) = 0 \\ f_2(x,y,z,w) = 0 \\ f_3(x,y,z,w) = 0 \\ f_4(x,y,z,w) = 0 \end{cases} \Rightarrow \begin{cases} g_1(x) = 0 \\ g_2(x,y) = 0 \\ g_3(x,y,w) = 0 \\ g_4(x,y,z,w) = 0 \end{cases}$$

The system of polynomial equations can be solved by solving

$$g_1(x) = 0 \to g_2(x,y) = 0 \to g_3(x,y,w) = 0 \to g_4(x,y,z,w) = 0$$

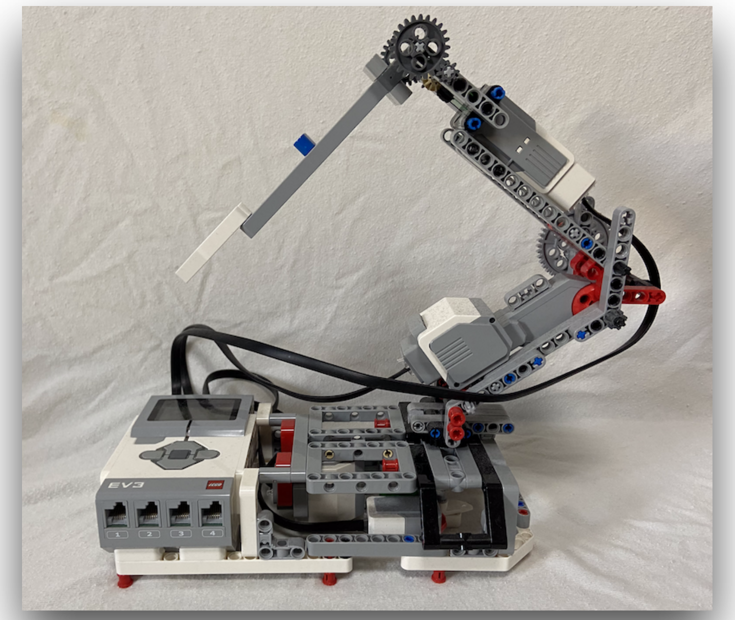
👍

One can know the feasibility of the motion before the actual computation

👎

Tends to be computationally expensive

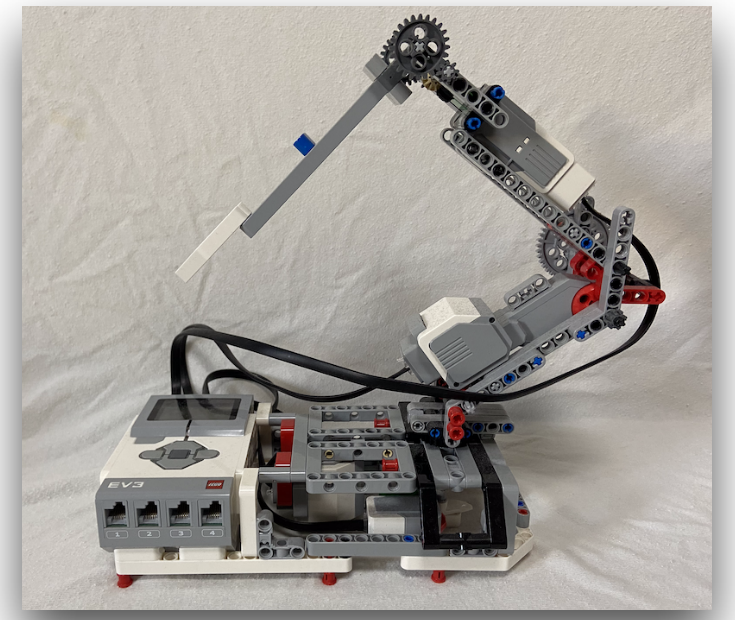
Our previous results



Solving the inverse kinematic problem using Comprehensive Gröbner Systems (CGS) and real quantifier elimination based on the CGS (CGS-QE method) (Otaki et al., CASC 2021)

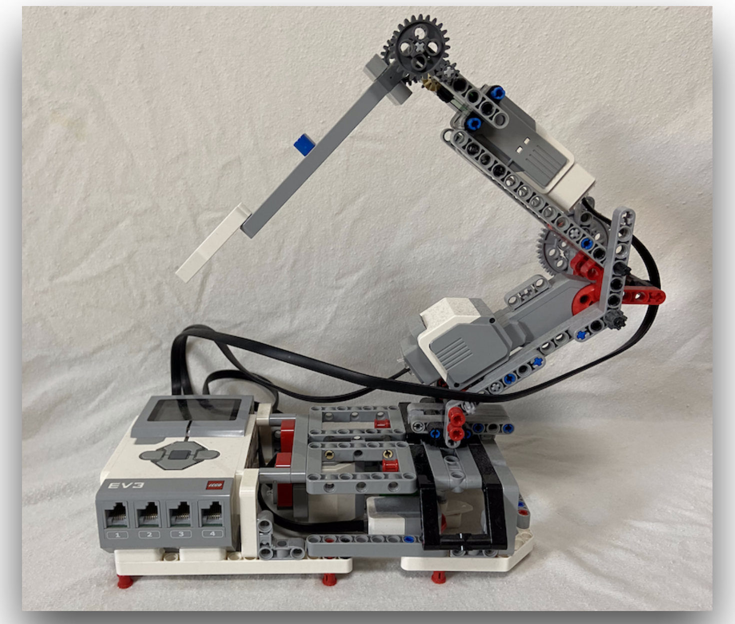
- **Features:**
  - Verification of inverse kinematics solutions with the CGS-QE method
  - Preventing repeated calculation of Gröbner bases by the use of CGS
- **A remaining issue:** “the preparation steps” (preparation of the solver before the actual calculation) were executed by hand

Our new contributions



1. A new and efficient implementation (automating “the preparation steps”)
2. An extension of the inverse kinematics computation to the **trajectory planning**
    1. Repeated calculation of inverse kinematics computation
    2. For a given path expressed with a parameter, certify that the whole motion along the path is feasible by the use the CGS-QE method

Plan of the talk



1. Formulation of the inverse kinematic problem

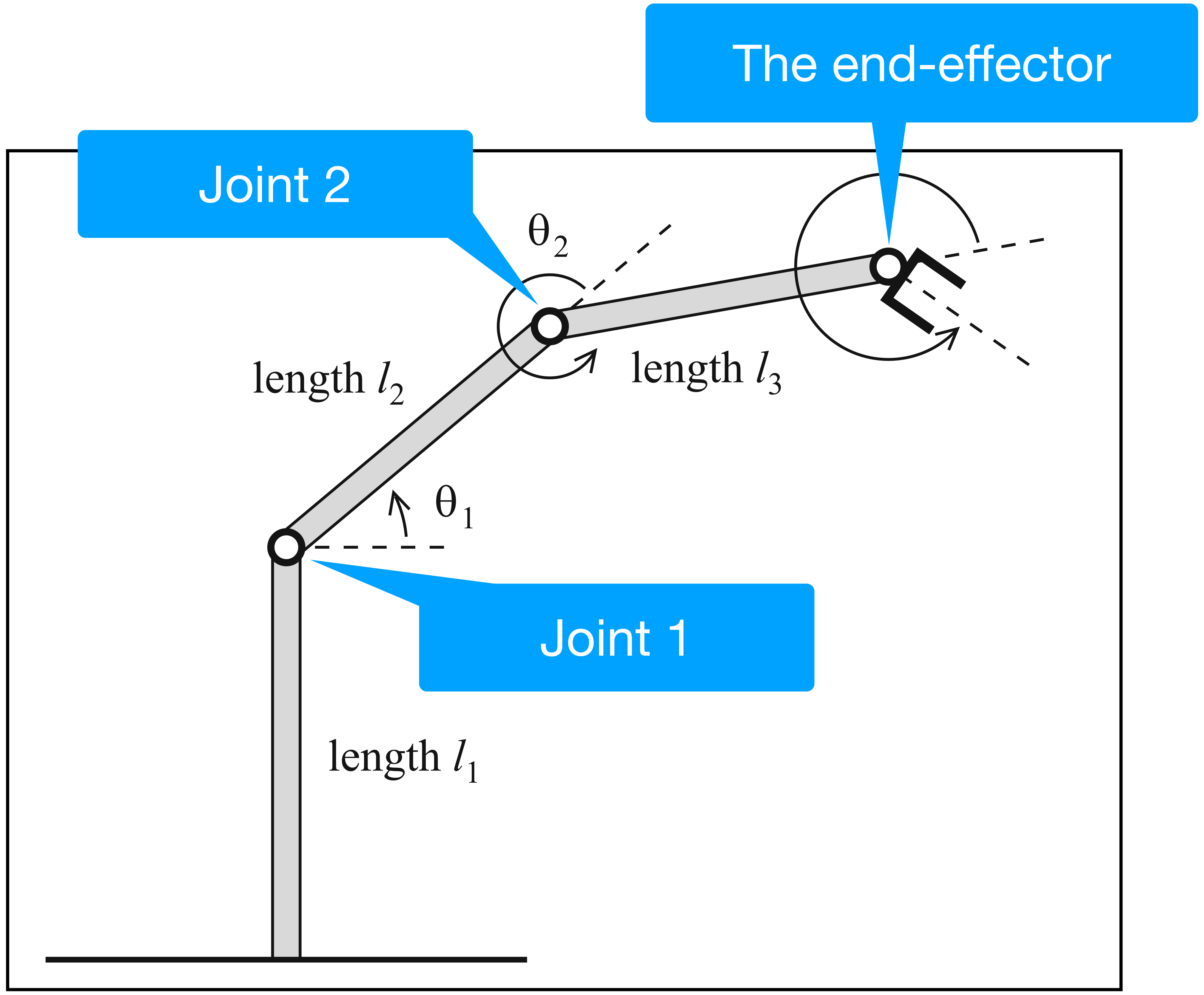
2. Solving the inverse kinematic problem using the CGS-QE method

3. Solving the trajectory planning problem using the CGS-QE method

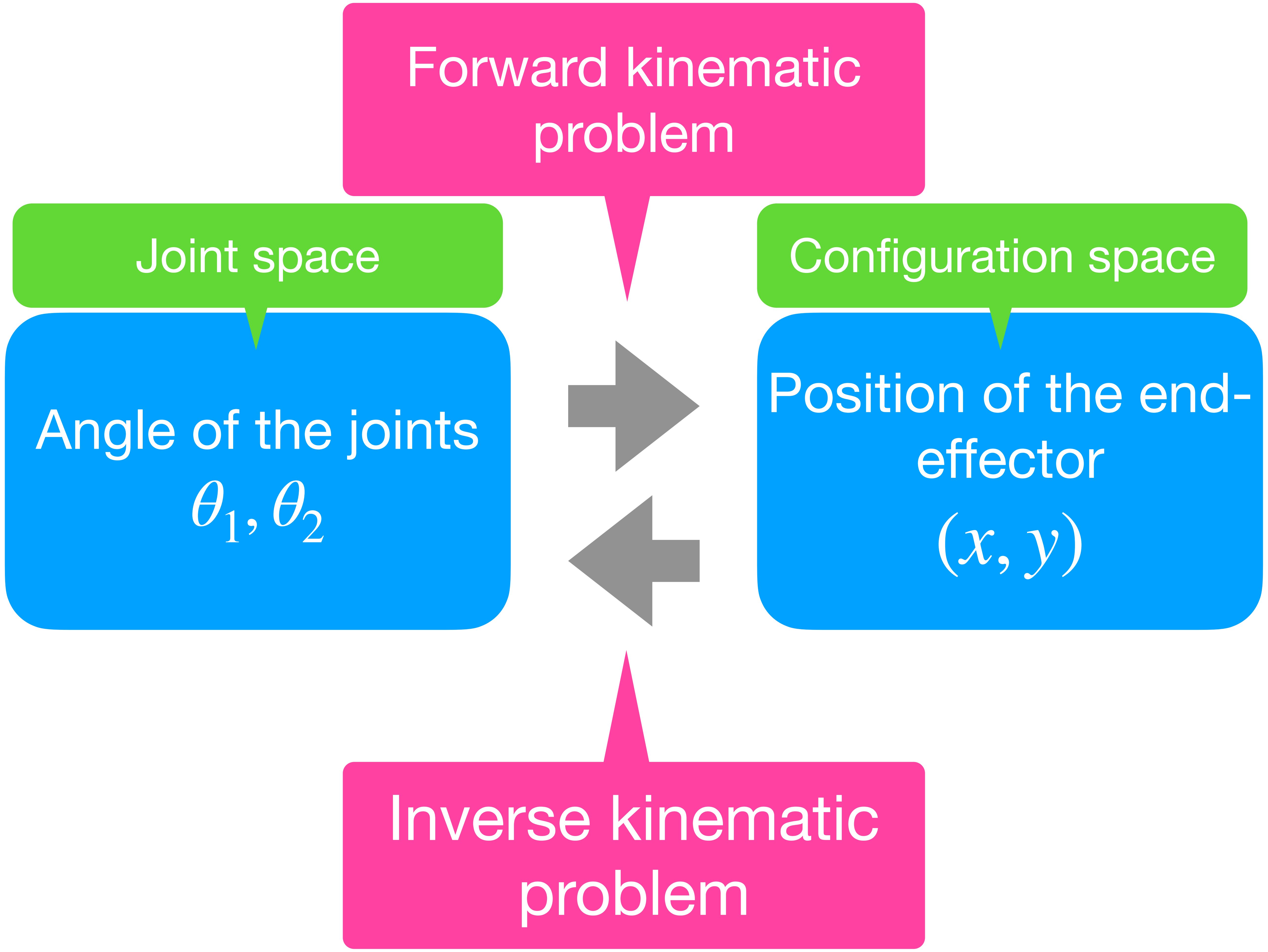
4. Trajectory planning with verification of the feasibility of the whole motion along the path using the CGS-QE method

Formulation of the inverse kinematic problem

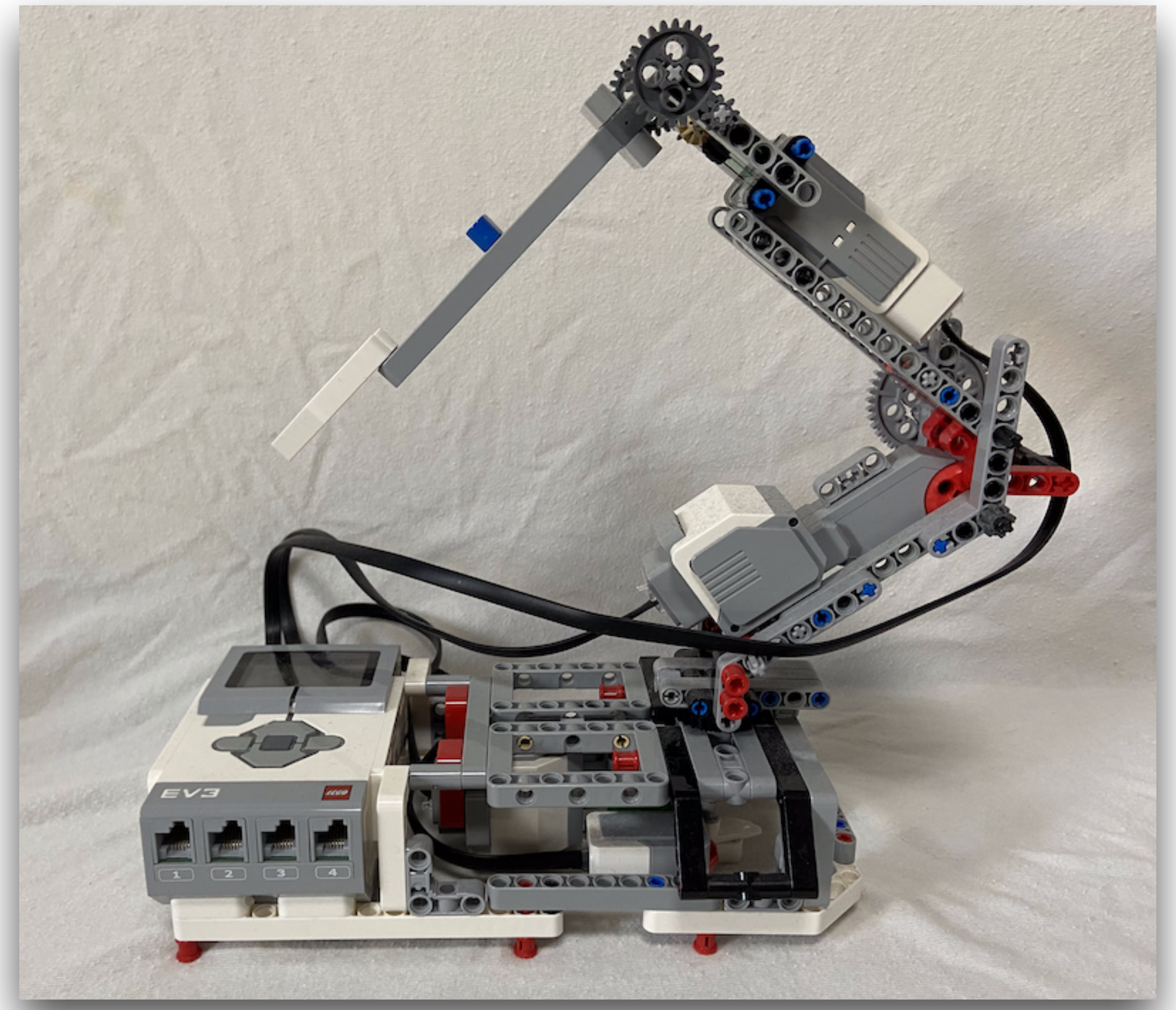
Forward and inverse kinematic problems



Cox et al. (2015)



The joint space and the configuration space



(1) Formulate the forward kinematic problem

Joint space

$S^1 \times S^1 \times S^1$
(Angles of joints 1, 4 and 7)

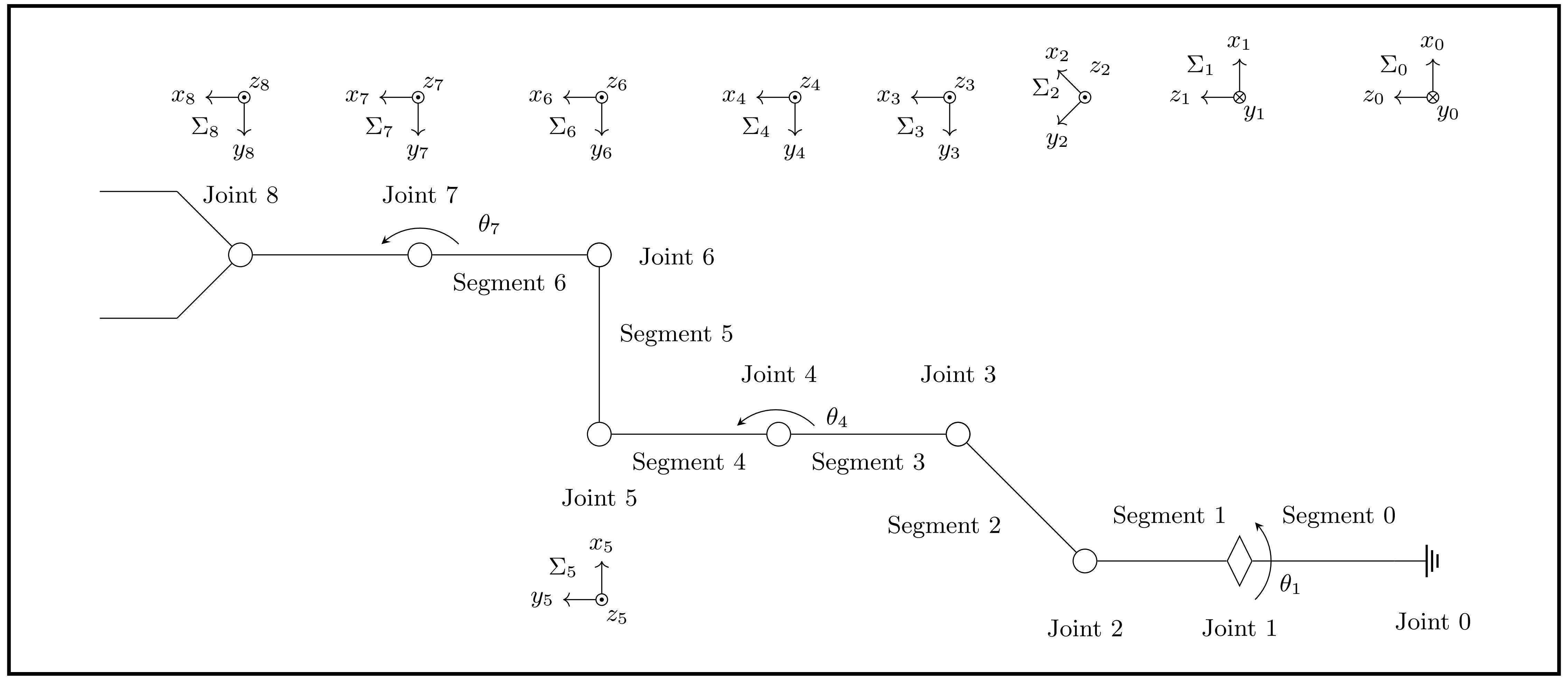
Configuration space

$\mathbb{R}^3$
(Position of the end-effector)

(2) Solve the inverse kinematic problem

Formulation of the forward kinematic problem

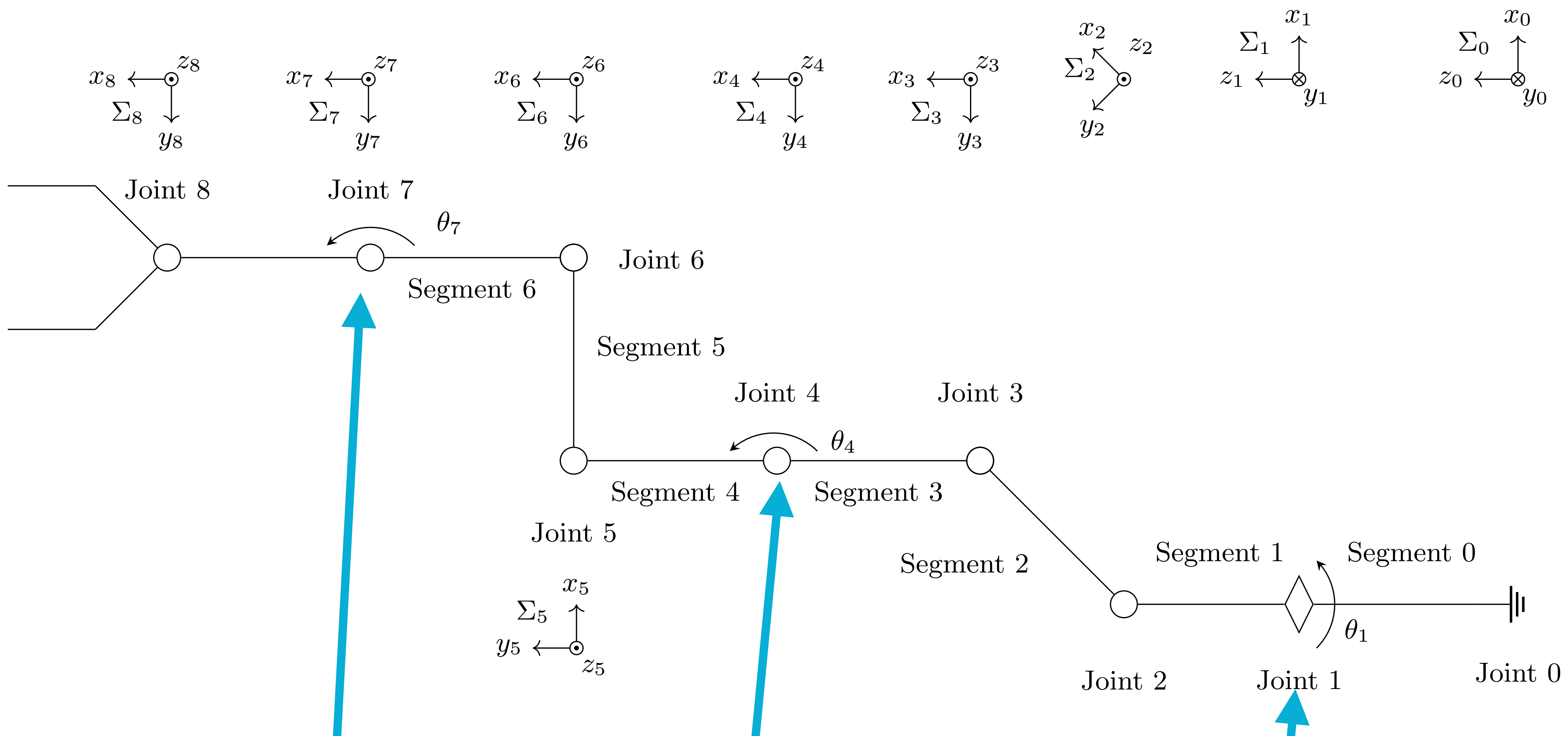
1. Define a coordinate system for each joint
2. Calculate coordinate transformations between adjacent joints



Defining coordinate-system for each joint

A modified Denavit-Hartenberg convention (Siciliano, et al. (2008))

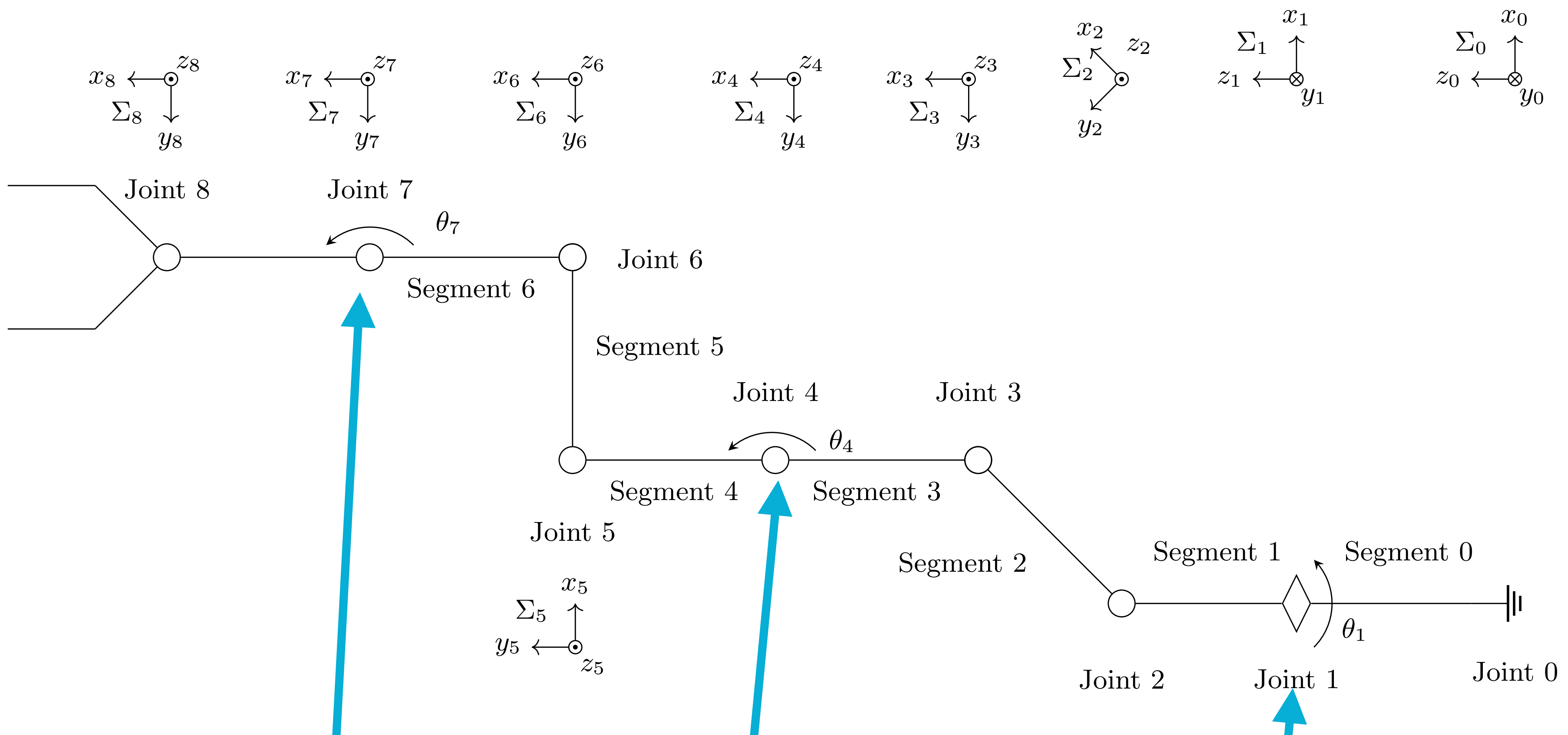
- Σ_i : the coordinate system with the origin at Joint i ($i = 1,\ldots,7$):
  - z_i: along with the axis of Joint i
  - x_i: the common normal to z_{i-1} and z_i (overlapping with the link)
  - y_i: defined so that Σ_i forms the right-handed coordinate system



Joints 1, 4 and 7 are revolutional joints

Parameters that specify the position and the orientation of Σ_{i-1} w.r.t. Σ_i

- a_i: the distance between axes z_{i-1} and z_i
- α_i: the angle between axes z_{i-1} and z_i with respect to x_i axis
- d_i: the distance between axes x_{i-1} and x_i
- θ_i: the angle between axes x_{i-1} and x_i with respect to z_i axis



Joints 1, 4 and 7 are revolutional joints

The transformation matrix ${}^{i-1}T_i$ from Σ_i to Σ_{i-1} w.r.t. Σ_{i-1}

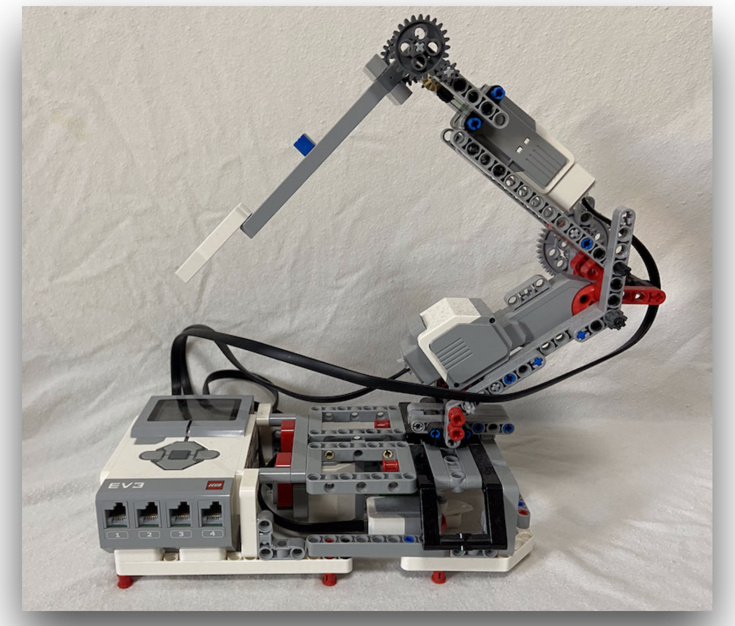
$$
{}^{i-1}T_i = \begin{pmatrix} \boxed{\begin{matrix} \cos\theta_i & -\sin\theta_i & 0 \\ \cos\alpha_i \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \\ \sin\alpha_i \sin\theta_i & \sin\alpha_i \cos\theta_i & \cos\alpha_i \end{matrix}}_{\text{Rotation}} & \boxed{\begin{matrix} a_i \\ -d_i \sin\alpha_i \\ d_i \cos\alpha_i \end{matrix}}_{\text{Translation}} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{pmatrix}
$$

$$T = {}^{0}T_1 \, {}^{1}T_2 \cdots {}^{6}T_7 \, {}^{7}T_8$$

for transformation from Σ_8 to Σ_0

i	a_i (mm)	α_i	d_i (mm)	θ_i
1	0	0	80	θ_1
2	0	$\pi/2$	0	$\pi/4$
3	88	0	0	$\pi/4$
4	24	0	0	θ_4
5	96	0	0	$-\pi/2$
6	16	0	0	$\pi/2$
7	40	0	0	θ_7
8	120	0	0	0

The forward kinematic problem



${}^t(x, y, z)$: position of the end-effector, θ_i: the angle of joint i ($i = 1,\ 4,\ 7$)

Position of the end-effector w.r.t. Σ_0 is expressed as:

$$x = -\,120\cos\theta_1\cos\theta_4\sin\theta_7 + 16\cos\theta_1\cos\theta_4 - 120\cos\theta_1\sin\theta_4\cos\theta_7$$

$$\quad -136\cos\theta_1\sin\theta_4 + 44\sqrt{2}\cos\theta_1$$

$$y = -\,120\sin\theta_1\cos\theta_4\sin\theta_7 + 16\sin\theta_1\cos\theta_4 - 120\sin\theta_1\sin\theta_4\cos\theta_7$$

$$\quad -136\sin\theta_1\sin\theta_4 + 44\sqrt{2}\sin\theta_1$$

$$z = 120\cos\theta_4\cos\theta_7 + 136\cos\theta_4 - 120\sin\theta_4\sin\theta_7 + 16\sin\theta_4 + 104 + 44\sqrt{2}$$

The inverse kinematic problem expressed as a system of polynomial equations

With $c_i = \cos\theta_i$, $s_i = \sin\theta_i \quad (i = 1,4,7)$, we have:

$$f_1 = 120c_1c_4s_7 - 16c_1c_4 + 120c_1s_4c_7 + 136c_1s_4 - 44\sqrt{2}c_1 + x = 0$$

$$f_2 = 120s_1c_4s_7 - 16s_1c_4 + 120s_1s_4c_7 + 136s_1s_4 - 44\sqrt{2}s_1 + y = 0$$

$$f_3 = -120c_4c_7 - 136c_4 + 120s_4s_7 - 16s_4 - 104 - 44\sqrt{2} + z = 0$$

$$f_4 = s_1^2 + c_1^2 - 1 = 0$$

$$f_5 = s_4^2 + c_4^2 - 1 = 0$$

$$f_6 = s_7^2 + c_7^2 - 1 = 0$$

Constraints on the trigonometric functions

Solving the inverse kinematic problem using the CGS-QE method

Quantifier elimination algorithms based on CGS

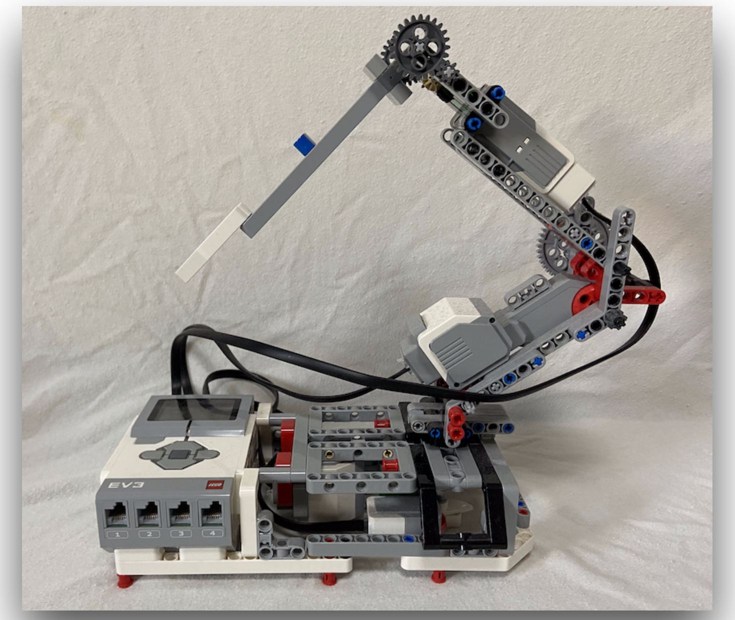
- Weispfenning (1998): using Comprehensive Gröbner Bases and counting the number of real roots of a system of polynomial equations

- Fukasaku et al. (2015): using an improved CGS algorithm by Suzuki and Sato (2006) with further improvements (the CGS-QE algorithm)

- CGS-QE = (CGS calculation) + (the theory of real root counting)

CGS-QE algorithm

(Weispfenning (1998), Fukasaku et al. (2015))



$f_1, \ldots, f_r \in R[\bar{A}, \bar{X}]$

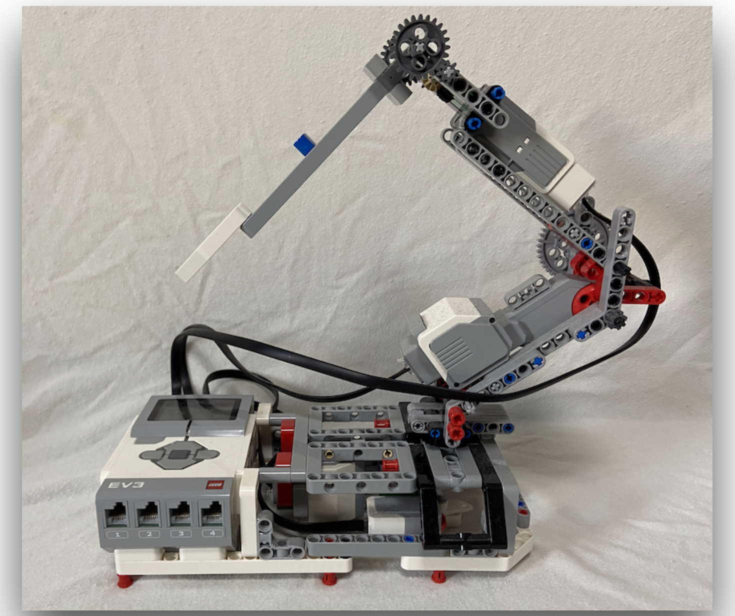
$\bar{X} = X_1, \ldots, X_n$ (variables)

$\bar{A} = A_1, \ldots, A_m$ (parameters)

$\exists \bar{X}\,(f_1(\bar{A}, \bar{X}) = 0 \wedge \cdots \wedge f_r(\bar{A}, \bar{X}) = 0)$ … (*) subject of quantifier elimination

CGS-QE algorithm

(Weispfenning (1998), Fukasaku et al. (2015))



1. Calculate CGS of $\langle f_1, \ldots, f_r \rangle$: let $\mathcal{G} = \{(S_1, G_1), \ldots, (S_t, G_t)\}$ (S_i: a segment: the set of parameters)

2. For each S_i, by using the theory of real root counting (Becker, Wöermann (1994), Pedersen et al. (1993)), derive a condition ψ_i on parameters $\bar{A}$ such that (the system of polynomial equations defined by) G_i has real roots

3. $\displaystyle\bigvee_{i=1}^{t}$ ((the difining formula of S_i) $\wedge\, \psi_i$) is equivalent to the given formula (*), with quantifiers eliminated

Solving the inverse kinematic problem

Variables $c_1, s_1, c_4, s_4, c_7, s_7$, parameters x, y, z, position of the end-effector $(x_0, y_0, z_0) \in \mathbb{R}^3$

$$f_1 = 120c_1c_4s_7 - 16c_1c_4 + 120c_1s_4c_7 + 136c_1s_4 - 44\sqrt{2}c_1 + x = 0$$

$$f_2 = 120s_1c_4s_7 - 16s_1c_4 + 120s_1s_4c_7 + 136s_1s_4 - 44\sqrt{2}s_1 + y = 0$$

$$f_3 = -120c_4c_7 - 136c_4 + 120s_4s_7 - 16s_4 - 104 - 44\sqrt{2} + z = 0$$

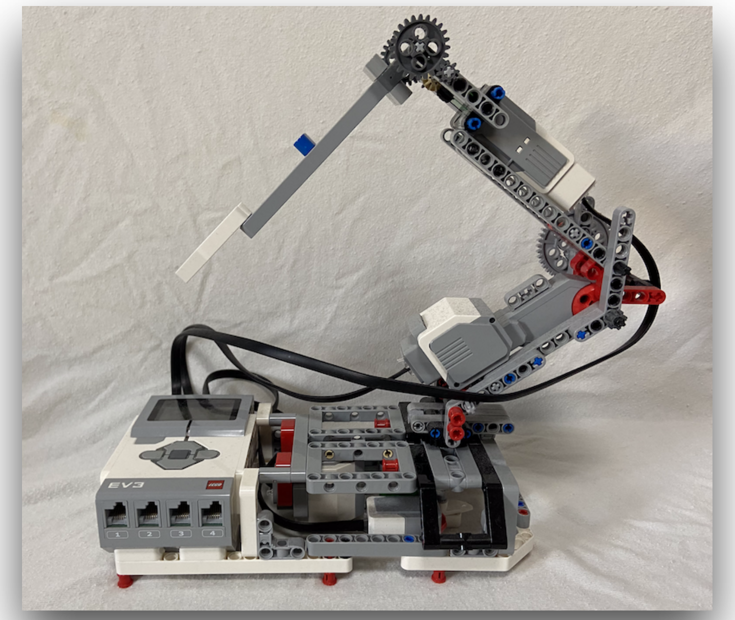
$$f_4 = s_1^2 + c_1^2 - 1 = 0$$

$$f_5 = s_4^2 + c_4^2 - 1 = 0$$

$$f_6 = s_7^2 + c_7^2 - 1 = 0$$

Algorithm 1

Solving the inverse kinematic problem



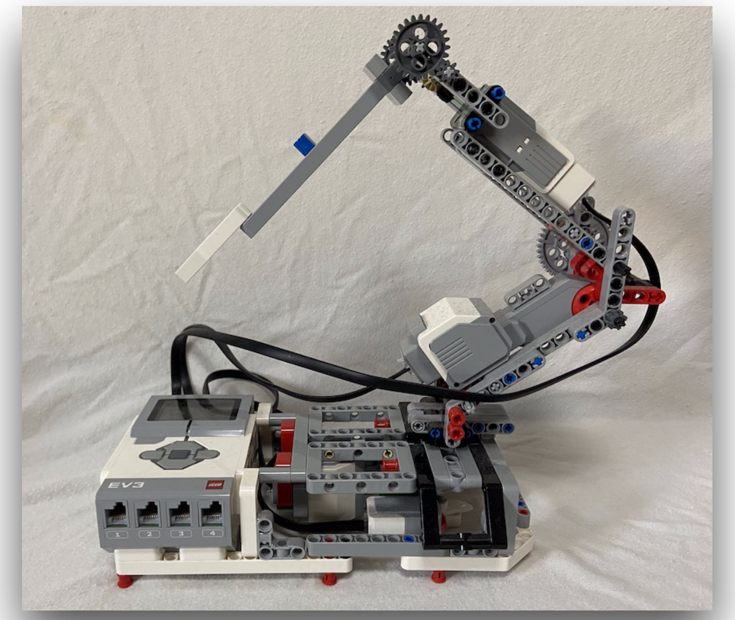
Inputs:

1. $F = \{f_1, \ldots, f_r\}$
2. Variables $\bar{X} = \{c_1, s_1, c_4, s_4, c_7, s_7\}$
3. Parameters $\bar{A} = \{x, y, z\}$
4. Position of the end-effector $a = (x_0, y_0, z_0) \in \mathbb{R}^3$

Output: configuration of the joints $\Theta = \{\theta_1, \theta_4, \theta_7\}$

Algorithm 1

Solving the inverse kinematic problem



1. Calculate CGS of $\langle F \rangle$: $\mathcal{G} = \{(S_1, G_1(\bar{A}, \bar{X})), \ldots, (S_t, G_t(\bar{A}, \bar{X}))\}$ (S_i: a segment) (CGS can be calculated in advance)
2. From $\mathcal{G} = \{(S_1, G_1), \ldots, (S_t, G_t)\}$, eliminate (S_i, G_i) satisfying $S_i \cap \mathbb{R}^3 = \emptyset$, and define $\mathcal{G} = \{(S_1, G_1(\bar{A}, \bar{X})), \ldots, (S_t, G_t(\bar{A}, \bar{X}))\}$ again (the "preprocessing step" in our previous method)
3. For $a = (x_0, y_0, z_0)$, choose (S_i, G_i) from $\mathcal{G}$ satisfying $a \in S_i$
4. Count the number of real roots of $G_i(a, \bar{X})$
    1. If real roots exist, then calculate real roots of $G_i(a, \bar{X})$ and return $\Theta = \{\theta_1, \theta_4, \theta_7\}$
    2. If $\langle G_i(a, \bar{X}) \rangle$ is not zero-dimensional, process it separately

Step 2 (the “preprocessing step” in our previous method)

For $S_k = V_{\mathbb{C}}(I_{k,1}) \backslash V_{\mathbb{C}}(I_{k,2})$, $I_{k,j} = \langle F_{k,j} \rangle$ $(j = 1,2), f \in F_{k,1}$:

- If f is univariate: count the real roots (with the discriminant ($\deg f = 2$), or the Sturm’s method ($\deg f \geq 3$)) → eliminate S_k if real root does not exist
- If f has trivial root(s): substitute the roots for $g \in F_{k,2}$; then if $g = 0$, eliminate S_k

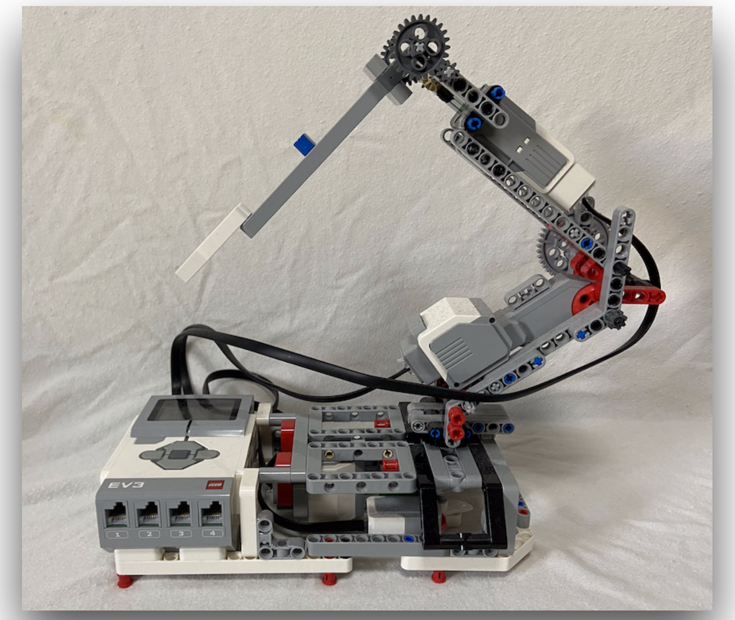
Step 4-2: in the case $\langle G_i(a, \bar{X}) \rangle$ is not zero-dimensional

1. If there exists $g(a, \bar{X}) \in G_i$ which has trivial root(s), then substitute the root(s) with $g(a, \bar{X}) \in G_i(a, \bar{X})$

2. Now, is $\langle G_i(a, \bar{X}) \rangle$ zero-dimensional?

- Yes → calculate real roots of $G_i(a, \bar{X})$
- No → cancel the calculation

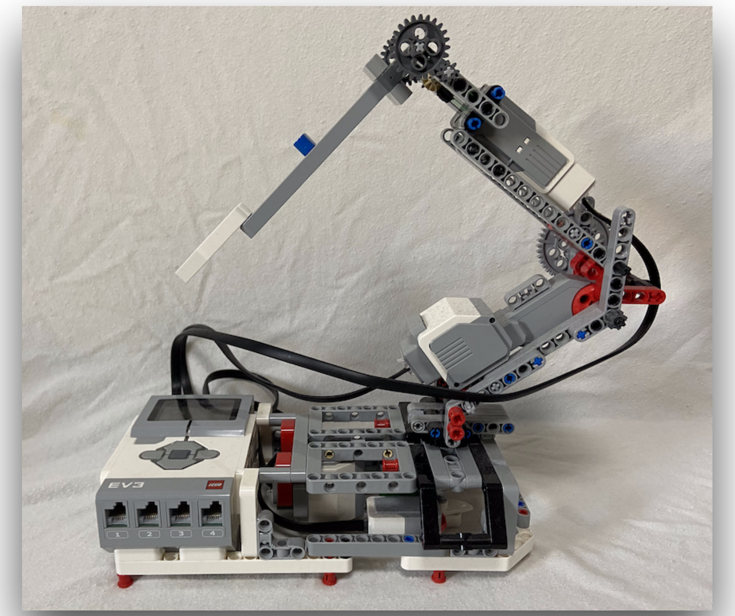
Implementation and experiments



https://github.com/teamsnactsukuba/ev3-cgs-qe-ik-2

- Implemented on Risa/Asir Version 20230315
- With the use of PARI-GP 2.3.11 (called from Asir for numerical solving of equations)
- CGS calculation: with Risa/Asir (implementation by Nabeshima (2018))
- Computing environment:
    - Intel Xeon Silver 4210 3.2 GHz, RAM 256 GB
    - Linux Kernel 5.4.0

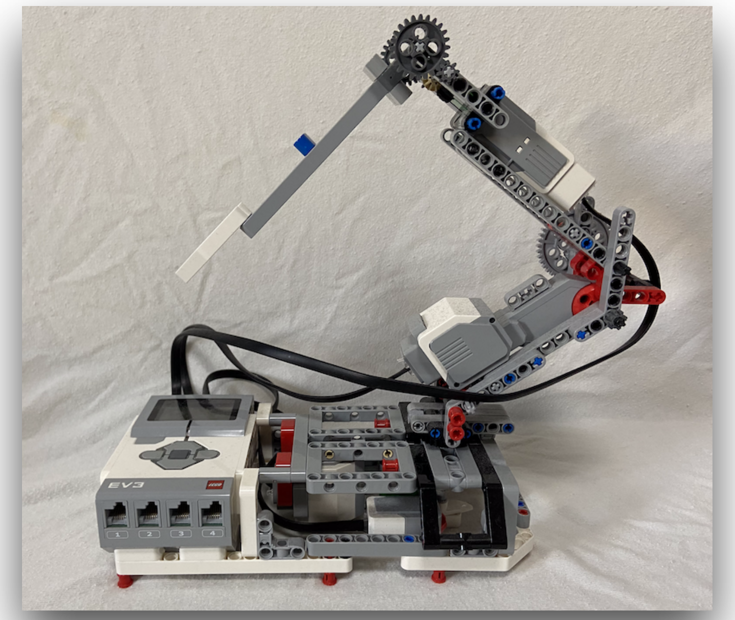
Experiments



1. Choose 1000 positions of the end-effector $\{(x, y, z)_j\}_{j=1}^{1000}$ randomly within the feasible region ($x, y, z \in \mathbb{Q}$, $x = a/b$ with $|b| < 100$; the same condition applies to the denominator of y and z)
2. Solve the inverse kinematic problem at $(x, y, z)_j$ and calculate the configuration of the joint $(\theta'_1, \theta'_2, \theta'_3)_j$
3. Using the forward kinematics with $(\theta'_1, \theta'_2, \theta'_3)_j$, calculate the position of the end-effector $(x', y', z')_j$ and the error (the euclidean distance) from the given position of the end-effector $(x, y, z)_j$ by $\sqrt{(x' - x)^2 + (y' - y)^2 + (z' - z)^2}$

Note: the CGS has been pre-calculated (calculation time: 62.3s)

Experimental results



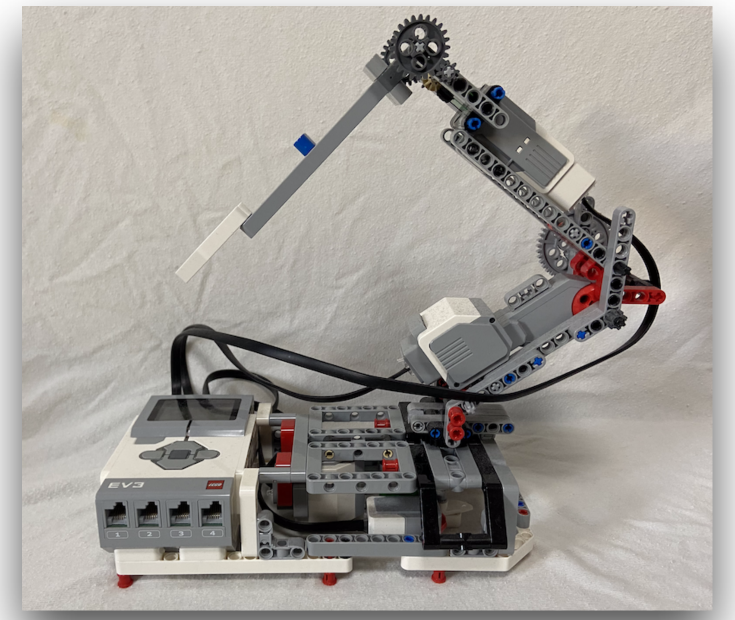
10 tests against 100 points:
Against 1000 points in total

Test	Time (sec.)	Error (mm)
1	0.1386	1.2428×10^{-12}
2	0.1331	2.3786×10^{-12}
3	0.1278	1.0845×10^{-12}
4	0.1214	1.6150×10^{-12}
5	0.1147	1.5721×10^{-12}
6	0.1004	1.6229×10^{-12}
7	0.0873	2.2518×10^{-12}
8	0.0792	1.3923×10^{-12}
9	0.0854	1.2919×10^{-12}
10	0.0797	1.8674×10^{-12}
Average	0.1068	1.6319×10^{-12}

Solving the trajectory planning problem using the CGS-QE method

A path of the end-effector

As a line segment



- $\boldsymbol{p}_0 = {}^t(x_0, y_0, z_0)$: the initial position
- $\boldsymbol{p}_d = {}^t(x, y, z)$: present position
- $\boldsymbol{p}_f = {}^t(x_f, y_f, z_f)$: the final position

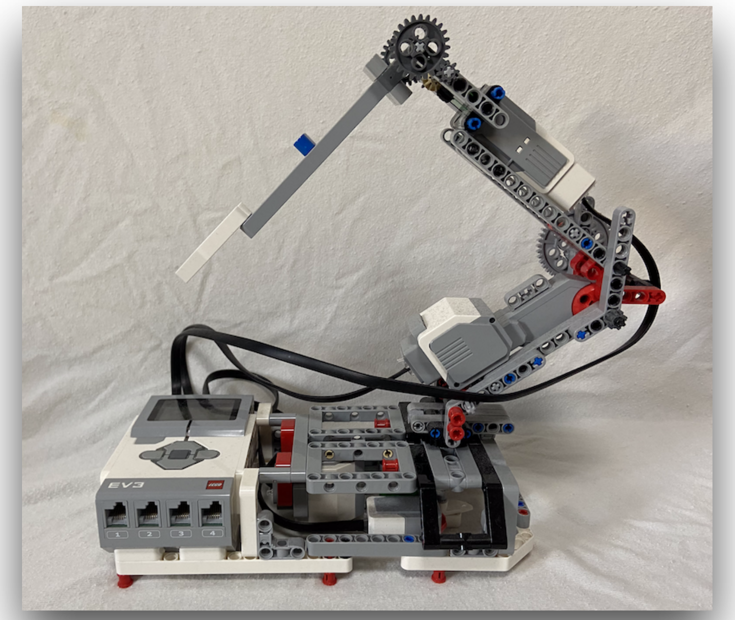
$(\boldsymbol{p}_d, \boldsymbol{p}_0, \boldsymbol{p}_f \in \mathbb{R}^3, \boldsymbol{p}_d \neq \boldsymbol{p}_f)$

The path: $\boldsymbol{p}_d = \boldsymbol{p}_0(1 - s) + \boldsymbol{p}_f s,\ s \in [0,1]$

Changing the position s as a function of t

- A trajectory by expressing $s \in [0,1]$ as $s = s(t)$, a function of $t \in [0..T]$ (T: positive integer)
- Let $\dot{s} = s'(t)$ (velocity) and $\ddot{s} = s''(t)$ (acceleration)
- Express $s(t)$ as a polynomial
- If we restrict $s'(0) = s''(0) = 0$ and $s'(T) = s''(T) = 0$, then $s(t)$ becomes a polynomial of degree 5 in t (Lynch and Park (2017))

Derivation of $s(t), \dot{s}(t), \ddot{s}(t)$



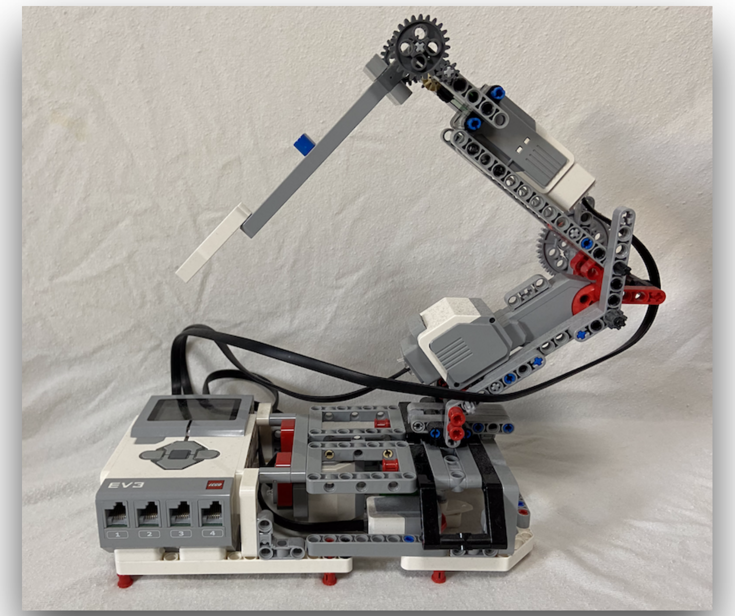
For $s(t) = \frac{a_4 T}{5}\left(\frac{t}{T}\right)^5 + \frac{a_3 T}{4}\left(\frac{t}{T}\right)^4 + \frac{a_2 T}{3}\left(\frac{t}{T}\right)^3 + \frac{a_1 T}{2}\left(\frac{t}{T}\right)^2 + a_0 t,$

$\dot{s}(t)$ and $\ddot{s}(t)$ are expressed as

$$\dot{s}(t) = a_4\left(\frac{t}{T}\right)^4 + a_3\left(\frac{t}{T}\right)^3 + a_2\left(\frac{t}{T}\right)^2 + a_1\left(\frac{t}{T}\right) + a_0,$$

$$\ddot{s}(t) = \frac{4a_4}{T}\left(\frac{t}{T}\right)^3 + \frac{3a_3}{T}\left(\frac{t}{T}\right)^2 + \frac{2a_2}{T}\left(\frac{t}{T}\right) + \frac{a_1}{T}$$

Derivation of $s(t), \dot{s}(t), \ddot{s}(t)$



With $s(0) = \dot{s}(0) = \ddot{s}(0) = 0, s(T) = 1, \dot{s}(T) = \ddot{s}(T) = 0$, we have a system of linear equations as:

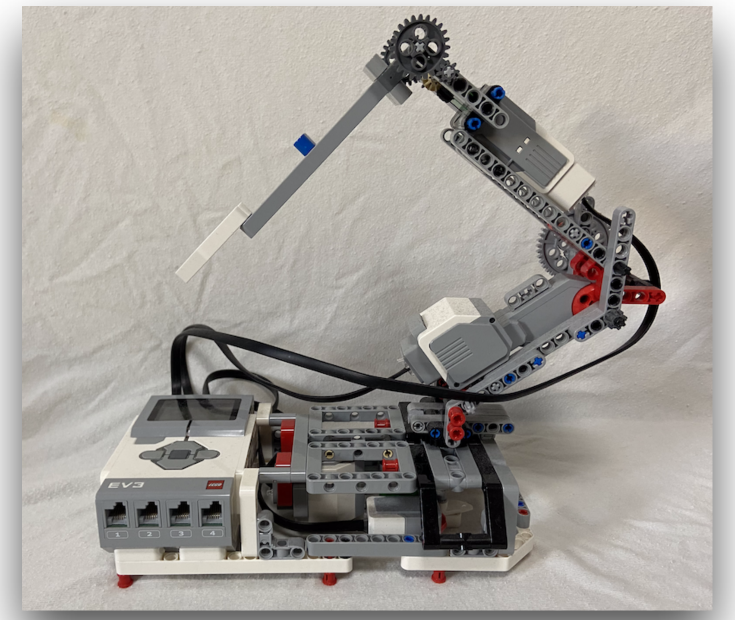
$$20a_2 + 15a_3 + 12a_4 - \frac{60}{T} = 0, \quad a_2 + a_3 + a_4 = 0, \quad 2a_2 + 3a_3 + 4a_4 = 0$$

By solving the equations, we have $a_2 = \frac{30}{T}, a_3 = -\frac{60}{T}, a_4 = \frac{30}{T}$, thus

$$s(t) = \frac{6}{T^5}t^5 - \frac{15}{T^4}t^4 + \frac{10}{T^3}t^3, \quad \dot{s}(t) = \frac{30}{T^5}t^4 - \frac{60}{T^4}t^3 + \frac{30}{T^3}t^2, \quad \ddot{s}(t) = \frac{120}{T^5}t^3 - \frac{180}{T^4}t^2 + \frac{60}{T^3}t$$

Algorithm 2

Solving trajectory planning problem



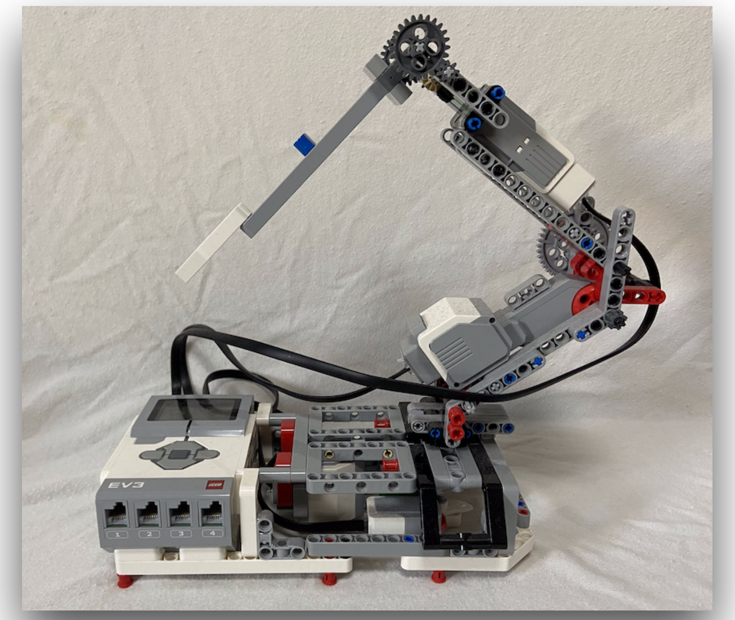
Inputs:

1. $F = \{f_1, \ldots, f_r\}$
2. Variables $\bar{X} = \{c_1, s_1, c_4, s_4, c_7, s_7\}$
3. Parameters $\bar{A} = \{x, y, z\}$
4. The initial position of the path $\boldsymbol{p}_0 = (x_0, y_0, z_0) \in \mathbb{R}^3$
5. The final position of the path $\boldsymbol{p}_f = (x_0, y_0, z_0) \in \mathbb{R}^3$
6. The length of time series $T \in \mathbb{N}$

Output: a series of the configuration of the joints $L = \{\Theta_t = (\theta_{1,t}, \theta_{4,t}, \theta_{7,t}) \mid t = 1, \ldots, T\}$

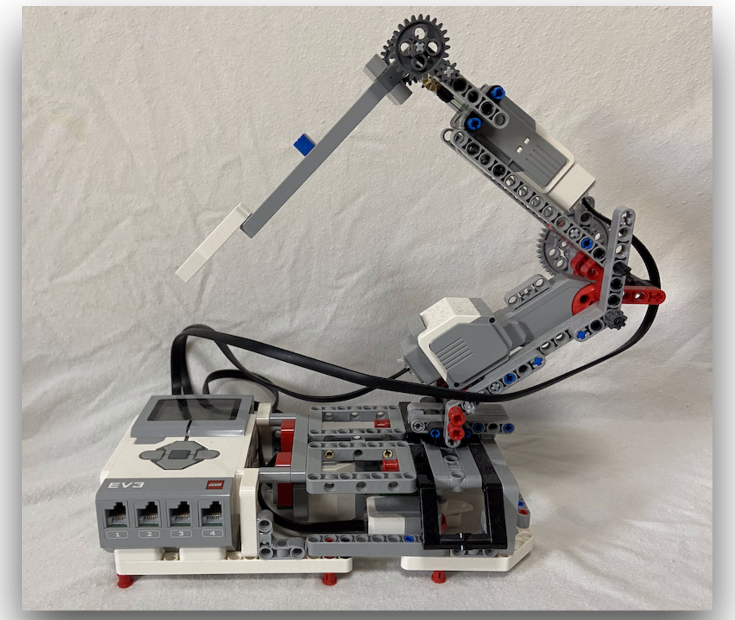
Algorithm 2

Solving trajectory planning problem



1. Calculate CGS of $\langle F \rangle$: $\mathscr{G} = \{(S_1, G_1(\bar{A}, \bar{X})), \ldots, (S_t, G_t(\bar{A}, \bar{X}))\}$ (S_i: a segment)
   ((S_i, G_i) satisfying $S_i \cap \mathbb{R}^3 = \emptyset$ can be eliminated)
2. $L \leftarrow \emptyset$
3. For $t = 1, \ldots, T$, repeat the following:
   1. Calculate the position of the end-effector as $s \leftarrow \frac{6}{T^5}t^5 - \frac{15}{T^4}t^4 + \frac{10}{T^3}t^3$; $\boldsymbol{p}_d \leftarrow \boldsymbol{p}_0(1-s) + \boldsymbol{p}_f$;
   2. Calculate the solution Θ of the inverse kinematic problem by Algorithm 1 with the inputs $(F, \bar{X}, \bar{A}, \boldsymbol{p}_d)$
   3. If the solution Θ is calculated, then $L \leftarrow L \cup \{\Theta\}$
4. Return L

Algorithm 2: an example

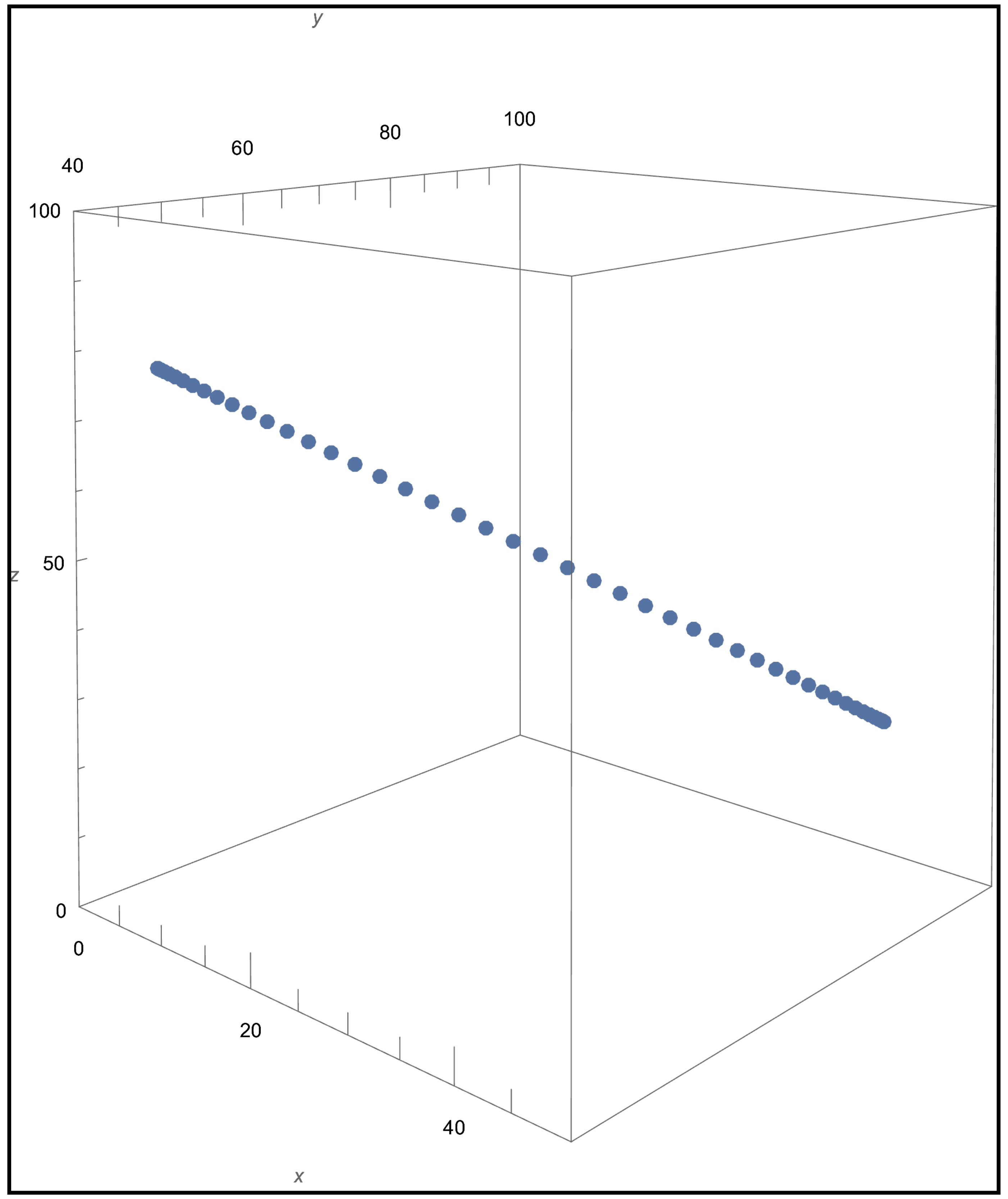


$$\boldsymbol{p}_0 = {}^t(x_0, y_0, z_0) = {}^t(10,40,80),$$

$$\boldsymbol{p}_f = {}^t(x_f, y_f, z_f) = {}^t(40,100,20),\ T = 50$$

Total computing time of inverse kinematic computations for 51 points: 3.377s

By eliminating (S_i, G_i) satisfying $S_i \cap \mathbb{R}^3 = \emptyset$ before the inverse kinematic computation: 2.246s



Trajectory planning with verification of the feasibility of the inverse kinematic solution

Substitute the path of the end-effector into the equations

Substitute $\boldsymbol{p}_d = \boldsymbol{p}_0(1-s) + \boldsymbol{p}_f s = {}^t(x, y, z) = {}^t(x_0(1-s) + x_f s, y_0(1-s) + y_f s, z_0(1-s) + z_f s)$

into the system of polynomial equations

$$
\begin{aligned}
f_1 &= 120c_1c_4s_7 - 16c_1c_4 + 120c_1s_4c_7 + 136c_1s_4 - 44\sqrt{2}c_1 + \boxed{x} = 0\\
f_2 &= 120s_1c_4s_7 - 16s_1c_4 + 120s_1s_4c_7 + 136s_1s_4 - 44\sqrt{2}s_1 + \boxed{y} = 0\\
f_3 &= -120c_4c_7 - 136c_4 + 120s_4s_7 - 16s_4 - 104 - 44\sqrt{2} + \boxed{z} = 0\\
f_4 &= s_1^2 + c_1^2 - 1 = 0\\
f_5 &= s_4^2 + c_4^2 - 1 = 0\\
f_6 &= s_7^2 + c_7^2 - 1 = 0
\end{aligned}
$$

x, y, z: parameters

Verification of the existence of a real root

For $s \in [0,1]$, verify that the system of equations has a real root with the CGS-QE method

$$
\begin{aligned}
f_1 &= 120c_1c_4s_7 - 16c_1c_4 + 120c_1s_4c_7 + 136c_1s_4 - 44\sqrt{2}c_1 + x_0(1-s) + x_f s = 0,\\
f_2 &= 120s_1c_4s_7 - 16s_1c_4 + 120s_1s_4c_7 + 136s_1s_4 - 44\sqrt{2}s_1 + y_0(1-s) + y_f s = 0,\\
f_3 &= -120c_4s_7 - 136c_4 + 120s_4s_7 - 16s_4 - 104 - 44\sqrt{2} + z_0(1-s) + z_f s = 0,\\
f_4 &= s_1^2 + c_1^2 - 1 = 0\\
f_5 &= s_4^2 + c_4^2 - 1 = 0\\
f_6 &= s_7^2 + c_7^2 - 1 = 0
\end{aligned}
$$

s: a parameter

Algorithm 3

Verification of the feasibility of the inverse kinematic solution

Inputs:

1. $F = \{f_1, \ldots, f_r\}$
2. Variables $\bar{X} = \{c_1, s_1, c_4, s_4, c_7, s_7\}$
3. Parameters $\bar{A} = \{x, y, z\}$
4. The initial position of the path $\boldsymbol{p}_0 = (x_0, y_0, z_0) \in \mathbb{R}^3$
5. The end (target) position of the path $\boldsymbol{p}_f = (x_0, y_0, z_0) \in \mathbb{R}^3$
6. The length of time series $T \in \mathbb{N}$

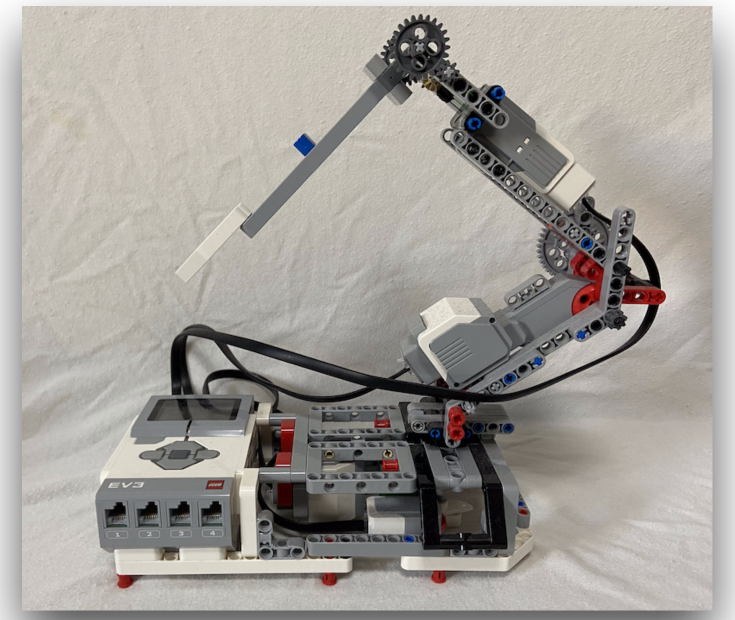
Output: a time series of the configuration of the joints $L = \{\Theta_t = (\theta_{1,t}, \theta_{4,t}, \theta_{7,t}) \mid t = 1, \ldots, T\}$

Algorithm 3

Verification of the feasibility of the inverse kinematic solution

1. $F' \leftarrow$ (Substituting x, y, z in F with the path $\boldsymbol{p}_d \leftarrow \boldsymbol{p}_0(1-s) + \boldsymbol{p}_f$);
2. Calculate CGS of $\langle F' \rangle$ as $\mathcal{G} = \{(S_1, G_1(\bar{A}, \bar{X})), \ldots, (S_t, G_t(\bar{A}, \bar{X}))\}$ (S_i: a segment)
   ((S_i, G_i) satisfying $S_i \cap \mathbb{R}^3 = \emptyset$ can be eliminated)
3. For $\mathcal{G}$, with the CGS-QE algorithm, calculate the range M of s for which F' has a real root
4. If $[0,1] \subset M$, then call Algorithm 2 with $(F, \bar{X}, \bar{A}, \boldsymbol{p}_0, \boldsymbol{p}_f, T)$ to calculate the series of the configuration of the joints $L = \{\Theta_t = (\theta_{1,t}, \theta_{4,t}, \theta_{7,t}) \mid t = 1, \ldots, T\}$

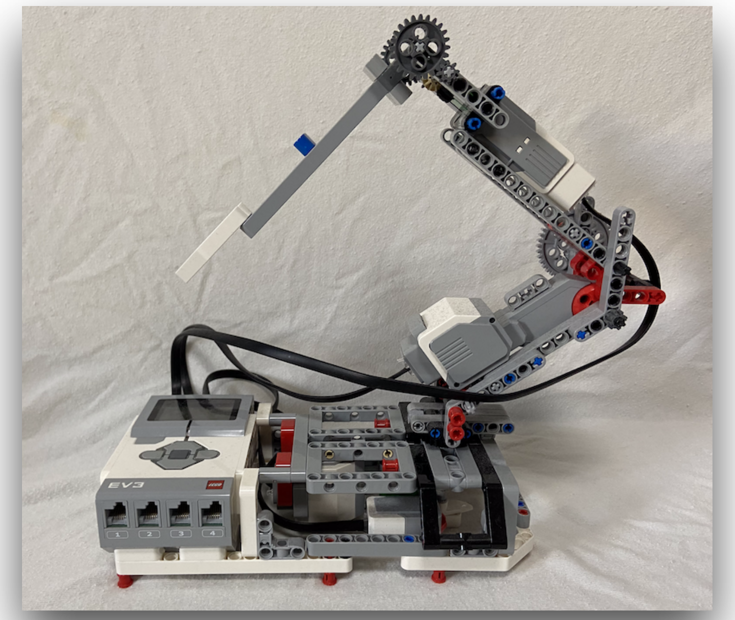
Algorithm 3: an example



- $\boldsymbol{p}_0 = {}^t(x_0, y_0, z_0) = {}^t(10,40,80), \boldsymbol{p}_f = {}^t(x_f, y_f, z_f) = {}^t(40,100,20)$
- CGS with 6 segments have been calculated in 485.8s
- For CGS $\mathscr{G} = \{(S_1, G_1), \ldots, (S_6, G_6)\}$, segments satisfying $S_i \cap \mathbb{R}^3 = \emptyset$ have been eliminated in 0.009s; three segments have been remained
- With the CGS-QE method, it has been verified that the system of equations has a real root for $s \in [0,1]$ in 1.107s
- The trajectory planning has been executed with Algorithm 2

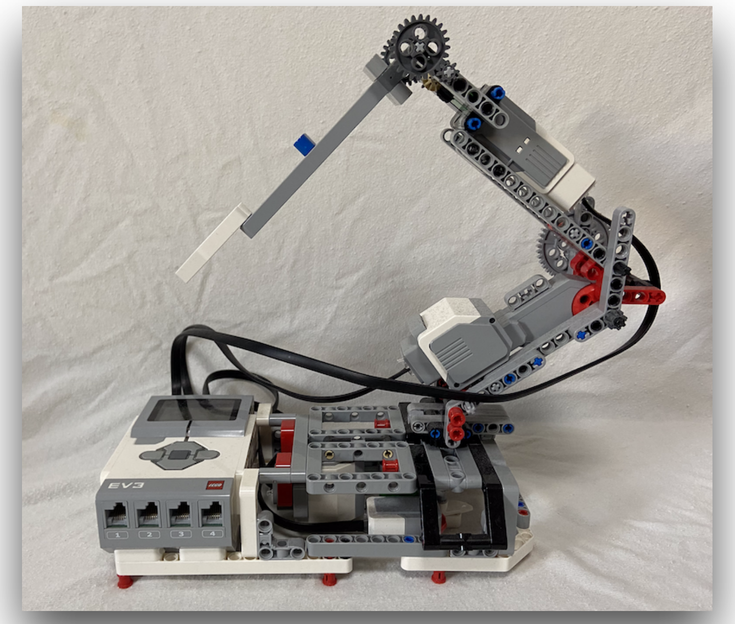
Concluding remarks

Summary

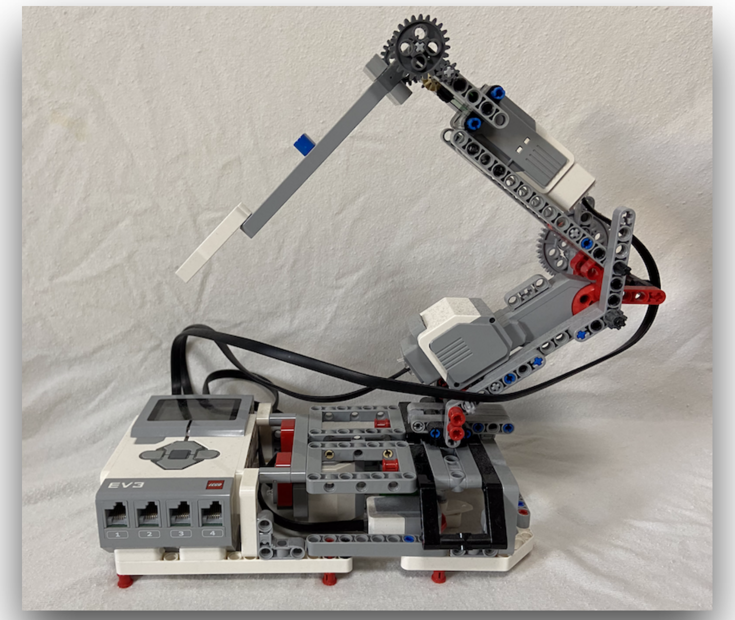


- Proposed methods with the use of the CGS-QE method for solving the inverse kinematic problem and the path planning problem of robot manipulator of 3 DOF:
  - Solving an inverse kinematic problem for *a single point* of the end-effector
  - Solving inverse kinematic problems *repeatedly for a series of points in the path* given as a line segment
  - Verifying feasibility of path planning computation for the path given as a line segment with a parameter

Future work



- Guarantee continuity of inverse kinematic solutions
    - Detect/analyze singular points of polynomial systems with parameters
- Improvement of the efficiency of the solver
- Path planning with more general curves represented by polynomials
- Developing a method for manipulators of higher DOF



Thank you!

References

Becker, E., Wöermann, T.: On the trace formula for quadratic forms. In: Recent advances in real algebraic geometry and quadratic forms. Contemp. Math., vol. 155, pp. 271–291. AMS (1994).

Chen, C., Maza, M.M.: Semi-algebraic description of the equilibria of dynamical systems. In: Gerdt, V.P., Koepf, W., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2011. LNCS, vol. 6885, pp. 101–125. Springer (2011).

Cox, D.A., Little, J., O'Shea, D.: Using Algebraic Geometry, 2nd edn. Springer, Heidelberg (2005).

Cox, D.A., Little, J., O'Shea, D.: Ideals, Varieties and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra (4th Ed.). Springer (2015).

Faugère, J.C., Merlet, J.P., Rouillier, F.: On solving the direct kinematics problem for parallel robots. Research report RR-5923, INRIA (2006).

Fukasaku, R., Iwane, H., Sato, Y.: Real Quantifier Elimination by Computation of Comprehensive Gröbner Systems. In: Proceedings of ISSAC '15 p. 173–180. ACM (2015).

Horigome, N., Terui, A., Mikawa, M.: A design and an implementation of an inverse kinematics computation in robotics using Gröbner bases. In: ICMS 2020. LNCS, vol. 12097, pp. 3–13. Springer (2020).

References (continued)

Kalker-Kalkman, C.M.: An implementation of Buchbergers' algorithm with applications to robotics. Mech. Mach. Theory 28(4), 523–537 (1993).

Lazard, D., Rouillier, F.: Solving parametric polynomial systems. J. Symb. Comput. 42(6), 636–667 (2007).

Lynch, K.M., Park, F.C.: Modern Robotics: Mechanics, Planning, and Control. Cambridge University Press, Cambridge (2017).

Maekawa, M., Noro, M., Ohara, K., Takayama, N., Tamura, K.: The design and implementation of OpenXM-RFC 100 and 101. In: ASCM 2001, pp. 102–111. World Scientific (2001).

Nabeshima, K.: CGS: a program for computing comprehensive Gr¨obner sys- tems in a polynomial ring [computer software] (2018). https://www.rs.tus.ac.jp/~nabeshima/softwares.htm

Noro, M.: A computer algebra system: Risa/Asir. In: Algebra, Geometry and Software Systems, pp. 147–162. Springer (2003).

Otaki, S., Terui, A., Mikawa, M.: A design and an implementation of an inverse kinematics computation in robotics using real quantifier elimination based on comprehensive Gr¨obner systems. Preprint (2021). arXiv:2111.00384

References (continued)

Pedersen, P., Roy, M.F., Szpirglas, A.: Counting real zeros in the multivariate case. In: Computational algebraic geometry, Progr. Math., vol. 109, pp. 203–224. Birkhäuser (1993).

Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G.: Springer. Robotics: Modelling, Planning and Control (2008).

da Silva, S.R.X., Schnitman, L., Cesca Filho, V.: A solution of the inverse kinematics problem for a 7-degrees-of-freedom serial redundant manipulator using Gröbner bases theory. Math. Probl. Eng. 2021, 6680687 (2021).

Terui, A., Yoshizawa, M., Mikawa, M.: ev3-cgs-qe-ik-2: an inverse kinematics solver based on the CGS-QE algorithm for an EV3 manipulator [computer software] (2023). https://github.com/teamsnactsukuba/ev3-cgs-qe-ik-2

The PARI Group, Univ. Bordeaux: PARI/GP version 2.13.1 (2021). https://pari.math.u-bordeaux.fr/

Uchida, T., McPhee, J.: Triangularizing kinematic constraint equations using Gröbner bases for real-time dynamic simulation. Multibody Syst. Dyn. 25, 335–356 (2011).

Uchida, T., McPhee, J.: Using Gröbner bases to generate efficient kinematic solutions for the dynamic simulation of multi-loop mechanisms. Mech. Mach. Theory 52, 144–157 (2012).

References (continued)

Weispfenning, V.: A new approach to quantifier elimination for real algebra. In: Quantifier Elimination and Cylindrical Algebraic Decomposition. pp. 376–392. Springer (1998).

Wolfram Research Inc: Mathematica, Version 13.1 [computer software] (2022).

Yang, L., Hou, X., Xia, B.: A complete algorithm for automated discovering of a class of inequality-type theorems. Sci. China Ser. F Inf. Sci. 44(1), 33–49 (2001).