*Robotica* (2024), **42**, pp. 2403–2429
doi:10.1017/S0263574724000808

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

# A machine learning-based approach for automatic motion/constraint and mobility analysis of parallel robots

Xinming Huo, Zehong Song and Tao Sun

Key Laboratory of Mechanism Theory and Equipment Design of Ministry of Education, Tianjin University, Tianjin, RP China
**Corresponding author:** Tao Sun; Email: stao@tju.edu.cn

**Received:** 14 December 2023; **Revised:** 6 April 2024; **Accepted:** 17 April 2024; **First published online:** 17 May 2024

**Keywords:** automatic mobility analysis; neural network; parallel mechanism; reciprocal screw system

## Abstract

Motion and constraint identification are the fundamental issue throughout the development of parallel mechanisms. Aiming at meaningful result with heuristic and visualizable process, this paper proposes a machine learning-based method for motions and constraints modeling and further develops the automatic software for mobility analysis. As a preliminary, topology of parallel mechanism is characterized by recognizable symbols and mapped to the motion of component limb through programming algorithm. A predictive model for motion and constraint with their nature meanings is constructed based on neural network. An increase in accuracy is obtained by the novel loss function, which combines the errors of network and physical equation. Based on the predictive model, an automatic framework for mobility analysis of parallel mechanisms is constructed. A software is developed with WebGL interface, providing the result of mobility analysis as well as the visualizing process particularly. Finally, five typical parallel mechanisms are taken as examples to verify the approach and its software. The method facilitates to attain motion/constraint and mobility of parallel mechanisms with both numerical and geometric features.

## 1. Introduction

Parallel robot consists of a fixed platform and a moving platform that are connected by several limbs. This leads to a compact structure and high potential stiffness; however, it also results in a complex relationship between motions and constraints. In fact, motions and constraints are crucial factors in kinematic and stiffness modeling. Moreover, the number and properties of mobility can be determined by considering motion and constraints. Hence, analyzing motion and constraints automatically by programming is a fundamental task in computer-assisted mobility analysis and performance modeling.

In screw theory [1–2], if a wrench acts on a rigid body in such a way that it produces no work on its infinitesimal twist, the two screws are referred to as reciprocal. The reciprocal product is an algebraic operation defined as a scalar product based on the virtual power of a wrench on a twist. This product is zero between motion and its constraints [1]. So far, solving for motion and constraint has been a null space problem that has been well studied in mathematics. The null space can be found using various numerical methods, such as Gauss–Seidel elimination [3], Gram–Schmidt orthogonalization [4], and an affine augmentation method [5]. However, it is noted that there are multiple solutions when the screw system is lower than five-dimensional, making it difficult to select solutions with explicit physical meanings. Although any selected solution would result in the correct number of degrees of freedom (DoF), it may lead to incorrect properties and confused results in performance modeling. Nevertheless, in automatically solving motion and constraints [6], it is necessary but difficult to generate initial motion axis screws in algebraic format relative to a same coordinate system.

The observation method is an alternative approach that investigates the kinematic features of some typical joints and branches to determine the constraints. Hunt discussed the relationship between three-dimensional geometry and spatial mechanisms in the context of a simplified screw system [2]. An

© The Author(s), 2024. Published by Cambridge University Press.

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

analytical approach to determine the bases of three unknown twist and wrench subspaces of lower mobility serial kinematic chains was presented, stemming from the reciprocal product of screw theory [7]. Thanks to progress made in recent years, the typical geometric conditions for motion and constraints have been determined. These conditions are as follows: the translational motion axis should be perpendicular to the constraint force, and the rotational motion axis should intersect with the force constraint and be perpendicular to the couple constraint. Building on these ideas, Yu proposed a graphical approach that visualizes motion and constraints [8]. Additionally, a geometric algebra approach was presented to determine the motion, constraint, mobility, and singularity of parallel mechanisms [9]. The observation method involves several rules that can be realized through reasoning, such as the "IF. . .THEN. . ." type. However, this automatic manner can only be applied to simple cases where only one rule is involved. Alternatively, Liu subsequently deposits the frequently used results of serial kinematic chains from observation and proposed an automatic approach for identifying the natural reciprocal screw systems based on the invariance properties of topology [10]. But for the cases out of these templates, it is hard to achieve the solution.

When carrying out automatic constraint and motion analysis, we always translate the known formulas, rules, or cases into computer language. However, this "translating manner" is limited to mechanisms within the known ones. In fact, constraint and motion analysis involves learning and reasoning, which are not independent of the experience and intelligence of people and cannot be realized by programming alone. Machine learning, on the other hand, is known for its reasoning abilities and can perform these thinking processes, making it well suited to automatic motion and constraint analysis. If the explicit meanings of the reciprocity between twist and wrench are learned by machine learning, constraint problems can be quickly and visually solved. Currently, machine learning is closely related to robotics, but it is mostly used for trajectory planning, cognition, control algorithms, and some fields of structural optimization [11–13]. In this case, a dataset indicating explicit meanings of the reciprocity will be collected and used to train a prediction model for motion and constraint. However, due to the limited dataset, how to improve the accuracy of the predictive model remains a major issue.

This paper intends to propose a machine learning-based method that can rapidly and accurately identify motion and constraints, resulting in the precise constraint of limbs and motion of moving platform with explicit physical meanings. The proposed method can be applied to solve the mobility of parallel mechanisms automatically. In contrast to existing methods of automatic mobility analysis [14–21], which require complex algorithms such as motion intersection operation, the proposed method is more programming-friendly and does not require such consideration. The primary contributions of this paper are as follows: (1) A highly accurate predictive model for reciprocal screws is constructed using limited database, resulting in precise constraints and motions with both natural meanings and numerical properties. (2) An automatic methodology for motion/constraint and mobility analysis is addressed, and web-based visualization software is developed.

The paper is organized as follows: Section 2 defines the topology and motion matrix of component limbs as preliminary. Section 3 constructs a predictive model for constraints and motion based on machine learning. Section 4 addresses the automatic mobility algorithm, and Section 5 develops software equipped with an interface. Section 6 provides five typical examples to verify the method and software. Conclusions are drawn in Section 7.

## 2. Preliminary

An automatic mobility analysis process begins with topology sketching. Based upon the authors' and other scholars' previous work [9, 22], geometric conditions of one-DoF joint axis play important roles in the automatic recognition of parallel robotic topology. Then the twist of component limb is generated according to the topology characters by algorithm, which is the foundation for constraint and motion solving of moving platform. (To enable constraint and motion solving of a moving platform, an algorithm generates twist for each limb component based on its topology. This is achieved by using a motion screw, which serves as the foundation for the platform's movement.)

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

***Table I.*** *Topology character of jth kinematic joint.*

| Symbol | | Description |
|---|---|---|
| | $Z_j$ | Axis of the $j$th joint is perpendicular to the fixed platform |
| Axis direction | $x_j$ | Axis of the $j$th joint is parallel with the fixed platform |
| | $u_j$ | Axis of the $j$th joint is obliquely intersecting to the fixed platform |
| Axis position | $N_k$ | The $k$th marked point |
| Joint type | R/P/H | Type of the $k$th joint (R, P, and H are rotational joint, prismatic joint, and helical joint, respectively) |
| Connection | \| | Separator between two joints |
| Examples | $x_jN_k\mathrm{R}N_k$ | Rotational joint parallel to the fixed platform passing through point $N_k$ |
| | $x_jN_k\mathrm{R}N_{k+1}$ | Rotational joint parallel to the fixed platform passing through two marked points $N_k$ and $N_{k+1}$ |
| | $x_jx_j\mathrm{P}$ | Prismatic joint parallel to the fixed platform |
| | $N_kN_{k+1}\mathrm{P}$ | Prismatic joint with axis direction as $N_kN_{k+1}$ |
| | $x_jN_k\mathrm{H}N_k$ | Helical joint parallel to the fixed platform with one marked point $N_k$ |

### *2.1. Recognizable representation of parallel mechanism*

Parallel robotic mechanism consists of fixed platform, moving platform, and several component limbs. The topology of a parallel mechanism involves its number and kinematic types of limbs, as well as the adjacency and incidence of joints [23], which are considered as motion generators. To specify all the information of topology, rotational joint and helical joint are defined in the format as "axis direction-axis position 1-joint type-axis position 2," where the axis position is denoted by the points marked on the axis. With free position, prismatic joint is described as "axis direction (double same elements)-joint type," where sometimes the axis direction is determined by two marked points. The definition of topology character is given in Table I.

In this manner, component limb could be obtained by depicting the joints sequentially, beginning from the one connecting with fixed platform. Two adjacent joints are separated by "|". All of the characters of limbs consist of topology features of parallel mechanism. For example, 3-CRR parallel mechanism recorded in ref. [24] has three identical CRR limbs. Herein, R is the rotational joint and C is the cylindrical joint, which could be considered as the combination of a rotational joint and prismatic joint with the same motion axis. The three limbs are symmetrical assembled, and also the first joint of each limb is intersecting at one point and perpendicular to each other. Referring to Table I, 3-CRR could be characterized as

$$\left\{\begin{array}{c} x_1x_1\mathrm{P} \mid x_1N_1\mathrm{R}N_1 \mid x_1N_2\mathrm{R}N_2 \mid x_1N_3\mathrm{R}N_3 \\ x_2x_2\mathrm{P} \mid x_2N_1\mathrm{R}N_1 \mid x_2N_4\mathrm{R}N_4 \mid x_2N_5\mathrm{R}N_5 \\ z_1z_1\mathrm{P} \mid z_1N_1\mathrm{R}N_1 \mid z_1N_6\mathrm{R}N_6 \mid z_1N_7\mathrm{R}N_7 \end{array}\right\} \tag{2a}$$

It is noted that in some cases, the relationship between joint motion axes would be changed resulting from moving. For example, UPU limb is composed of two universal joints (two rotational joints with intersecting axes), connected by a prismatic joint. At the initial pose, assuming that motion axes of the first and the fifth rotational joints are parallel, but once the second or the fourth joints rotated, their parallel relationship would disappear. The motion and constraints are also changed. In this paper, the last rotational joint in UPU limb is defined as a "distal" joint, which should be in discussed in Section 4.

### *2.2. Motion modeling of component limb*

For the sake of visualization, all the modeling would be carried out in framework O-xyz located at the center O of fixed platform, with x-axis along with the first joint of the first limb and z-axis perpendicular with the fixed platform.

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

***Table II.*** *Motion of the kth joint under different geometric conditions with the jth joint.*

| **Geometric relationships** | $\boldsymbol{s_j}$ | $\boldsymbol{r_j}$ |
|---|---|---|
| $x_jN_j\mathrm{R}N_j \mid x_jN_k\mathrm{R}N_k$ (Parallel) | $(a_j, b_j, c_j)$ | $(l_k, m_k, n_k)$ |
| $x_jN_j\mathrm{R}N_j \mid x_jN_j\mathrm{R}N_j$ (Collinear) | $(a_j, b_j, c_j)$ | $(l_j, m_j, n_j)$ |
| $x_jN_j\mathrm{R}N_j \mid x_kN_k\mathrm{R}N_k$ (Perpendicular) | $(a_k, b_k, -(a_ja_k + b_jb_k)/c_j)$ | $(l_k, m_k, n_k)$ |
| $u_jN_j\mathrm{R}N_j \mid u_kN_j\mathrm{R}N_j$ (Intersecting) | $(a_k, b_k, c_k)$ | $(l_j, m_j, n_j)$ |
| $x_jN_j\mathrm{R}N_j \mid x_kN_k\mathrm{R}N_k$ (Any) | $(a_k, b_k, c_k)$ | $(l_k, m_k, n_k)$ |
| $x_jN_j\mathrm{R}N_j \mid u_kN_j\mathrm{R}N_j$ (Orthogonal) | $(a_k, b_k, -(a_ja_k + b_jb_k)/c_j)$ | $(l_j, m_j, n_j)$ |

Different from the well-known twist screw system, the motion of limb is defined in the format as

$$\mathbf{m}_i = \begin{bmatrix} \boldsymbol{s}_{i,1} & \boldsymbol{r}_{i,1} & h_{i,1} \\ \vdots & \vdots & \vdots \\ \mathbf{0} & \boldsymbol{s}_{i,m_i} & h_{i,m_i} \end{bmatrix}, 2 \le i \le n \tag{2b}$$

where $n$ is the number of limbs. $\boldsymbol{s}_{i,j} = [a_{i,j}, b_{i,j}, c_{i,j}]\,(1 \le j \le N_i)$ and $\boldsymbol{r}_{i,j} = [l_{i,j}, m_{i,j}, n_{i,j}]$ are the direction and position vectors of $j$th motion axis in $i$th limb, respectively, which could be formulated subjecting to topology characters. $a_{i,j}, b_{i,j}, c_{i,j}$ and $l_{i,j}, m_{i,j}, n_{i,j}$ are the elements of vectors. $h_{i,j} \in \mathbb{N}$ is an element to identify the type of joint, and when $h_{i,j} \neq 0$, it indicates the helical joint. Motion matrix could be considered as a source of screw format. The motion screw of the $j$th joint could be described as

$$\$_{\mathrm{t}} = \left(\boldsymbol{s}_{i,j}\ \boldsymbol{r}_{i,j} \times \boldsymbol{s}_{i,j} + h_{i,j}\boldsymbol{s}_{i,j}\right) \tag{2c}$$

Compared with screw format, motion matrix in Eq. (2b) provides complete information such as the marked point, which are important geometric items for constraint solving.

The motion matrix of limb could be derived from its topology character since the joints are serially connected. For the first joint, $\boldsymbol{s}_{i,1}$ and $\boldsymbol{r}_{i,j}$ are formulated in the general format $\boldsymbol{s}_{i,1} = [a_{i,1}, b_{i,1}, c_{i,1}]$ and $\boldsymbol{r}_{i,1} = [l_{i,1}, m_{i,1}, n_{i,1}]$. In some special cases, they would be simplified for programming. For example, when the topology of the first joint is "$x_1x_1P$", $\boldsymbol{s}_{i,1}$ turns to $\boldsymbol{s}_{i,1} = [a_{i,1}, b_{i,1}, 0]$. For two adjacent joints $j$ and $k$, the motion axis of one joint would be deducted according to that of another one. Assuming the direction and position vectors of $j$th motion axis are assigned as $\boldsymbol{s}_{i,j} = [a_{i,j}, b_{i,j}, c_{i,j}]$ and $\boldsymbol{r}_{i,j} = [l_{i,j}, m_{i,j}, n_{i,j}]$, the motion of the $k$th joint could be described as Table II.

Having the topology characters of mechanism at hand, algorithm 1 is presented to construct the motion matrix of limb. Three dictionaries are defined as *x_index*, *n_index* and *h_index* to store direction vectors, position vectors, and pitch, respectively. The motion matrix would be filled as Table II.

For example, submitting the topology character of Eq. (2a) into algorithm 1 as the input, the position and direction vectors of joints in each CRR limb could be inquired in dictionary. The motion matrix of 3-CRR mechanism could be obtained as

$$\mathbf{m} = \begin{bmatrix} \mathbf{m}_1 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & l_{12} & m_{12} & n_{12} & 0 \\ 1 & 0 & 0 & l_{13} & m_{13} & n_{13} & 0 \\ 1 & 0 & 0 & l_{14} & m_{14} & n_{14} & 0 \end{bmatrix} \\ \mathbf{m}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & l_{22} & m_{22} & n_{22} & 0 \\ 0 & 1 & 0 & l_{23} & m_{23} & n_{23} & 0 \\ 0 & 1 & 0 & l_{24} & m_{24} & n_{24} & 0 \end{bmatrix} \\ \mathbf{m}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & l_{32} & m_{32} & n_{32} & 0 \\ 0 & 0 & 1 & l_{33} & m_{33} & n_{33} & 0 \\ 0 & 0 & 1 & l_{34} & m_{34} & n_{34} & 0 \end{bmatrix} \end{bmatrix} \tag{2d}$$

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

**Algorithm 1: Motion modeling of component limbs**

Input: Characters of topology
Output: Motion matrix $\mathbf{m}_i$

1. Initialization:
Create direction vector dictionary: *x_index*, position vector dictionary: *n_index*, pitch dictionary: *h_index*; Create final output storage motion matrix: ***motion_matrix***, motion matrix with transformed "distal" joint geometry: ***undetermined_matrix***.
2 For analysis of each joint of the limb:
3 If Type of joint == R-joint or H-joint:
3 If Dictionary *x_index* does not contain direction $x_i$, $u_i$, $Z_i$:
4 *x_index* adds a new direction $x_i$, $u_i$ or $Z_i$;
5 If Dictionary *n_index* does not contain marked point $N_j$:
6 *n_index* adds a new marked point $N_j$;
7 If Dictionary *h_index* does not contain marked point $h_j$:
6 *h_index* adds a new marked point $h_j$;
7 If type of kinematic joint == P-joint:
8 If Dictionary *x_index* does not contain direction $x_i$, $u_i$, $Z_i$:
9 *x_index* adds a new direction $x_i$, $u_i$ or $Z_i$;
10 If P-joint between two rotational joints:
11 *x_index* adds a new location $N_{j+1}$;
12 If Dictionary *h_index* does not contain marked point $h_j$:
13 *h_index* adds a new marked point $h_j$;
12 End For
13 Fill ***motion_matrix*** according to Table II subjecting to the direction and position in the dictionary: *x_index*, *n_index* and *h_index*;
14 If specific geometric relationships exist for "distal" joints:
15 Fill ***undetermined_matrix*** according to the new geometric relationship;
16. Output the final matrix ***motion_matrix*** and ***undetermined_matrix***, i.e. the requested $\mathbf{m}_i$;

## 3. Predictive model of constraint/motion based on machine learning

In this section, a predictive model of constraints (motions) subjecting to the given motions (constraints) is constructed by machine learning. Not only numerical properties but also geometric conditions between motion and constraints are considered.

Constraints generated by component limbs would be obtained according to their motions:

$$\Gamma\left(\mathbf{m}_i\right) = \mathbf{f}_i \tag{3a}$$

where $\Gamma$ is the predictive model. $\mathbf{m}_i$ and $\mathbf{f}_i$ are the motion matrix of $i$th limb and its corresponding constraint matrix. It is noted that the constraints (motion) not only need to satisfy the reciprocity product with motion (constraint) but also expected to have the explicit meanings. To this end, machine learning is introduced to learn and reason the relationship between motion and constraints. Because motion and

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

constraint have the dual solving process, the predictive model in Eq. (3a) could be also applied for motion solving when the constraints are known.

$$\Gamma\left(\mathbf{f}_i\right) = \mathbf{m}_i \tag{3b}$$

### 3.1. Equivalent motion transformation

For some component limbs which contain rotational joints with parallel axes, the motion could be equivalently transformed firstly. For example, the motion matrix of the $i$th CRR limb could be described as Eq. (2d). It shows that the latter three rows are linear dependent, which could be equivalently transformed as

$$\mathbf{m}_i = \begin{bmatrix} a_{i1} & b_{i1} & 0 & l_{i1} & m_{i1} & n_{i1} & 0 \\ 0 & 0 & 0 & a_{i2} & b_{i2} & c_{i2} & 0 \\ 0 & 0 & 0 & a_{i3} & b_{i3} & c_{i3} & 0 \\ 0 & 0 & 0 & a_{i4} & b_{i4} & c_{i4} & 0 \end{bmatrix} \tag{3c}$$

Equation (3c) indicates that CRR limb has three translational motions and one rotational motion.

To simplify the inputs of machine learning network, the rotational motions with parallel axes would be transformed as translational motions preliminarily. It would not change the resultant constraints. Taking limbs with four-dimensional screws as example, the equivalent motion matrix is listed as Table III.

### 3.2. Optimal datasets with classification

To obtain the accurate constraints with explicit meanings, the geometric conditions between screw and its null space should be satisfied. Therefore, relative position and orientation characters are derived from motion matrix $\mathbf{m}_i$ and taken as the input of machine learning network

$$\mathbf{input} = \begin{bmatrix} a_1 & b_1 & c_1 & p_1 & q_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_m & b_m & c_m & p_m & q_m \end{bmatrix} \tag{3d}$$

where $m$ is the number of joints in the limb. Details of other variables are shown in Table IV.$\boldsymbol{s}_i = (a_i, b_i, c_i)$ and $p_i$ denote the orientation and position characters, respectively, which are classified by 14 and 7 types. It is defined that $\boldsymbol{s}_i = (1\ 0\ 0)$ and $\boldsymbol{s}_i = (0\ 1\ 0)$ indicate the parallel relationship between axis and basic platform and $\boldsymbol{s}_i = (0\ 0\ 1)$ denotes the perpendicular relationship. When the vector is parallel with XY plane, two categories are defined as ±XY and +X-Y. Similarly, other kinds are shown in Table V. It is noted that when the direction is random, $\boldsymbol{s}_i = (2\ 3\ 4)$ is used. When $p_i = 0$, it denotes the arbitrary position relationship. When $p_i = \{1, \cdots, 6\}$, it denotes the order number of key points which can indicate the position. $q_i$ is a sign to show the type of input, which is rotational motion (constraint force), translational motion (constraint couple), and screw motion (constraint screw).

As shown in Table VI, the direction of reciprocal screw could be concluded as five groups: perpendicular with one screw axis, perpendicular with a plan spanning by two screw axes, parallel with one axis, arbitrary, and unknown. Herein, "arbitrary" denotes the force or couple constraint with alternative geometric conditions, and "unknown" indicates the case exists the motion or constraint screws, in which only numerical condition should be satisfied. The position of reciprocal screw is described by the intersection point with the given screws. In this way, the format of output is defined as

$$\mathbf{output} = \begin{bmatrix} s_1 & p_{11} & p_{21} & q_1 \\ \vdots & \vdots & \vdots & \vdots \\ s_n & p_{1n} & p_{2n} & q_n \end{bmatrix} \tag{3e}$$

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

***Table III.*** *Equivalent motions of four-dimensional screws.*

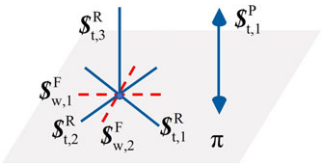
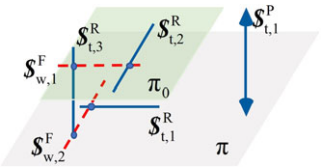
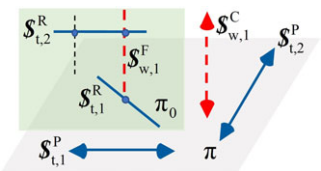
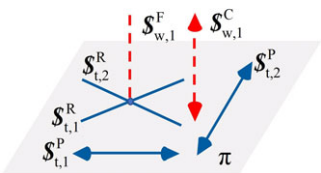
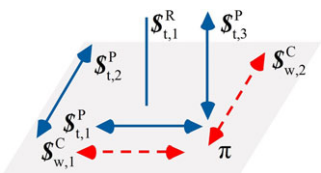
| Motion | Diagrams | Equivalent motion matrix |
|---|---|---|
| Three rotations and one translation (rotational axes are intersecting) |  | $\begin{bmatrix} a_{i1} & b_{i1} & 0 & l_{i1} & m_{i1} & n_{i1} & 0 \\ a_{i2} & b_{i2} & c_{i2} & l_{i1} & m_{i1} & n_{i1} & 0 \\ a_{i3} & b_{i3} & c_{i3} & l_{i1} & m_{i1} & n_{i1} & 0 \\ 0 & 0 & 0 & a_{i4} & b_{i4} & c_{i4} & 0 \end{bmatrix}$ |
| Three rotations and one translation (rotational axes are heterogeneous) |  | $\begin{bmatrix} a_{i1} & b_{i1} & 0 & l_{i1} & m_{i1} & n_{i1} & 0 \\ a_{i2} & b_{i2} & c_{i2} & l_{i2} & m_{i2} & n_{i2} & 0 \\ a_{i3} & b_{i3} & c_{i3} & l_{i3} & m_{i3} & n_{i3} & 0 \\ 0 & 0 & 0 & a_{i4} & b_{i4} & c_{i4} & 0 \end{bmatrix}$ |
| Two rotations and two translations (rotational joints are intersecting) |  | $\begin{bmatrix} a_{i1} & b_{i1} & 0 & l_{i1} & m_{i1} & n_{i1} & 0 \\ a_{i2} & b_{i2} & c_{i2} & l_{i1} & m_{i1} & n_{i1} & 0 \\ 0 & 0 & 0 & a_{i3} & b_{i3} & c_{i3} & 0 \\ 0 & 0 & 0 & a_{i4} & b_{i4} & c_{i4} & 0 \end{bmatrix}$ |
| Two rotations and two translations (rotational joints are heterogeneous) |  | $\begin{bmatrix} a_{i1} & b_{i1} & 0 & l_{i1} & m_{i1} & n_{i1} & 0 \\ a_{i2} & b_{i2} & c_{i2} & l_{i2} & m_{i2} & n_{i2} & 0 \\ 0 & 0 & 0 & a_{i3} & b_{i3} & c_{i3} & 0 \\ 0 & 0 & 0 & a_{i4} & b_{i4} & c_{i4} & 0 \end{bmatrix}$ |
| One rotation and three translations |  | $\begin{bmatrix} a_{i1} & b_{i1} & 0 & l_{i1} & m_{i1} & n_{i1} & 0 \\ 0 & 0 & 0 & a_{i2} & b_{i2} & c_{i2} & 0 \\ 0 & 0 & 0 & a_{i3} & b_{i3} & c_{i3} & 0 \\ 0 & 0 & 0 & a_{i4} & b_{i4} & c_{i4} & 0 \end{bmatrix}$ |

***Table IV.*** *Definition of input and output of neural network.*

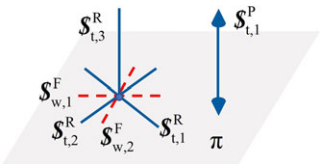
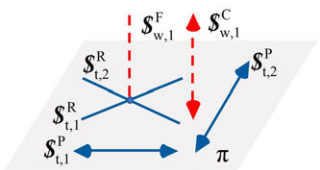
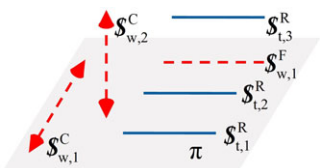
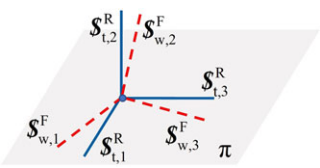
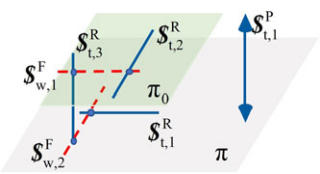
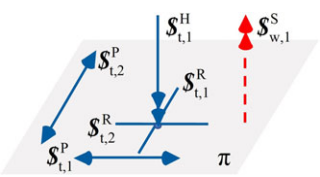
| Symbols | Meanings |
|---|---|
| $(a_i, b_i, c_i)$ | Direction vector in input $(\boldsymbol{s}_i = (a_i \quad b_i \quad c_i))$ |
| $s_i$ | Direction item in output $s_i = 0, 1, \cdots, 4$ |
| $p_i$ | Order number of position character $(p_i = \{0, \cdots, 6\})$ |
| $q_i$ | Type of wrench and twist $(q_i = \{-1, -2, -3\}(-1 \rightarrow \mathrm{P(F_c)}, -2 \rightarrow \mathrm{R(F_f)}, -3 \rightarrow \mathrm{H(F_s)}))$ |

***Table V.*** *Direction definition.*

| Direction | ±X | ±Y | ±Z | ±XY | +X-Y -X+Y | ±YZ | +Y-Z -Y+Z | ±XZ | +X-Z -X+Z |
|---|---|---|---|---|---|---|---|---|---|
| Vector | (1, 0, 0) | (0, 1, 0) | (0, 0, 1) | (1, 1, 0) | (1, −1, 0) | (0, 1, 1) | (0, 1, −1) | (1, 0, 1) | (1, 0, −1) |
| Direction | ±XYZ | -X+Y+Z +X-Y-Z | -X-Y+Z +X+Y-Z | +X-Y+Z -X+Y-Z | Random | – | – | – | – |
| Number | (1, 1, 1) | (1, –1, –1) | (1, 1, –1) | (1, –1, 1) | (2, 3, 4) | – | – | – | – |

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

***Table VI.*** *Relationship between the direction vector and the axis of the joint.*

| Diagrams | Location | Direction |
|---|---|---|
|  | $\$^{R}_{t,1}\$^{R}_{t,2}\$^{R}_{t,3} \rightarrow p_i = 1$<br>$\$^{F}_{w,1}\$^{F}_{w,2} \rightarrow p_{1i} = p_{2i} = 1$<br>(Intersected at a point) | $\$^{F}_{w,1} \perp \$^{P}_{t,1}$ and<br>$\$^{F}_{w,2} \perp \$^{P}_{t,1} (s_i = 1)$ |
|  | $\$^{R}_{t,1}\$^{R}_{t,2} \rightarrow p_i = 1$<br>$\$^{F}_{w,1} \rightarrow p_{1i} = p_{2i} = 1$<br>(Intersected at a point) | $\$^{F}_{w,1} \perp \$^{P}_{t,1}$ and<br>$\$^{F}_{w,1} \perp \$^{P}_{t,2} (s_i = 2)$ |
|  | $\$^{F}_{w,1} \rightarrow p_{1i} = p_{2i} = 0$<br>(Any location) | $\$^{F}_{w,1} / \$^{R}_{t,i} (s_i = 3)$ |
|  | $\$^{R}_{t,1}\$^{R}_{t,2}\$^{R}_{t,3} \rightarrow p_i = 1$<br>$\$^{F}_{w,1}\$^{F}_{w,2}\$^{F}_{w,3} \rightarrow p_{1i} = p_{2i} = 1$<br>(Intersected at a point) | $\$^{F}_{w,i}$ any direction<br>$(s_i = 0)$ |
|  | $\$^{R}_{t,1}\$^{R}_{t,2}\$^{R}_{t,3} \rightarrow p_i = 1, 2, 3$<br>$\$^{F}_{w,1} \rightarrow p_{11} = 1, p_{21} = 3$<br>$\$^{F}_{w,2} \rightarrow p_{12} = 2, p_{22} = 3$<br>(Intersected at two points) | $\$^{F}_{w,1} \perp \$^{P}_{t,1}$ and<br>$\$^{F}_{w,2} \perp \$^{P}_{t,1} (s_i = 1)$ |
|  | $\$^{R}_{t,1}\$^{R}_{t,2} \rightarrow p_i = 1$<br>$\$^{S}_{w,1} \rightarrow p_{11} = 0, p_{21} = 0$<br>(Including an H joint) | Direction unknown<br>$(s_i = 4)$ |

where $s_i (s_i = 0, 1, \cdots, 4)$ is defined as direction item as in Table V. $p_{1i}$ and $p_{2i}$ denote the intersection points of rotational joints. $q_i$ has the same meaning as the input in Table IV.

Taking CRR limb as an example, the input matrix could be described as

$$\textbf{input} = \begin{bmatrix} 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 1 & -2 \\ 0 & 0 & 1 & 2 & -2 \\ 0 & 0 & 1 & 3 & -2 \end{bmatrix} \tag{3f}$$

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

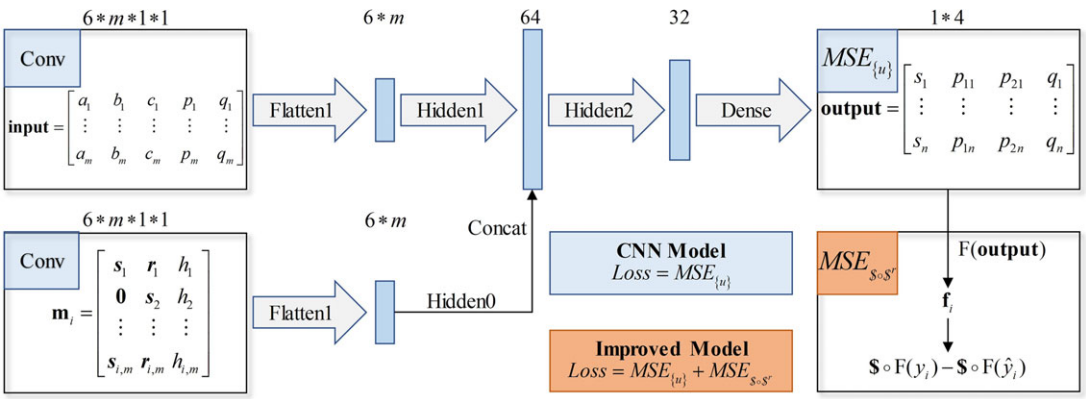



***Figure 1.*** *Architecture of CNN and improved model.*

The output is given as

$$\mathbf{output} = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 1 & 0 & 0 & -1 \end{bmatrix} \tag{3g}$$

Another example is given as R ˆ R ˆ R-P limb; the input matrix could be described as

$$\mathbf{input} = \begin{bmatrix} 2 & 3 & 4 & 1 & -2 \\ 2 & 3 & 4 & 1 & -2 \\ 2 & 3 & 4 & 1 & -2 \\ 2 & 3 & 4 & 0 & -1 \end{bmatrix} \tag{3h}$$

where $d_i$ is random placeholder to expand the dataset and increase the generalization of the model. The output matrix is given as

$$\mathbf{output} = \begin{bmatrix} 1 & 1 & 1 & -2 \\ 1 & 1 & 1 & -2 \end{bmatrix} \tag{3i}$$

### *3.3. Neural network modeling*

In this section, 5000 samples of motion and constraint matrix data are prepared and optimally classified as input and output, respectively. 80% of them are applied for training, and others are used in the test dataset. To achieve the mapping between motions and constraints, neural network (NN) for predictive model is formulated.

As shown in Fig. 1 (a), convolutional NN (CNN) is formulated, which consists of convolutional (Conv), flatten, hidden, and output layers. Conv layer performs element-by-element multiplication between input matrix and kernel of different weights. The weights are randomly generated at the beginning of the training. Following up flatten layer, data in vector form are transferred to the hidden layers, which contains two dense layers. Dense layer is responsible in encoding the features from previous layers in order to come up with relevant class and hence perform classification. In the output layer, constraint matrix would be returned to the model to compute the loss. Layer structure and parameters used in the considered CNN are listed in Table VII.

The loss function of the CNN is defined as

$$MSE_{\{u\}} = \frac{SSE}{n_s} = \frac{1}{n_s}\sum_{i=1}^{n_s} w_{mse1}\left(y_i - \hat{y}_i\right)^2 \tag{3j}$$

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

***Table VII.*** *Parameters of neural network.*

| Layer | Type | Input size | Kernel size | Activation | Dropout | Output size |
|---|---|---|---|---|---|---|
| 1 | Conv | [5000,$m$,6] | [$m$,6] | ReLU | No | [5000,6$m$,1,1] |
| 2 | Flatten | [5000,6$m$,1,1] | – | – | No | [5000,6$m$] |
| 3 | Dense1 | [5000,6$m$] | – | ReLU | No | [5000,32] |
| 4 | Concat | 2*[5000,32] | – | – | No | [5000,64] |
| 5 | Dense2 | [5000,64] | – | ReLU | No | [5000,32] |
| 6 | Dense3 | [5000,32] | – | – | Yes | [5000,4] |

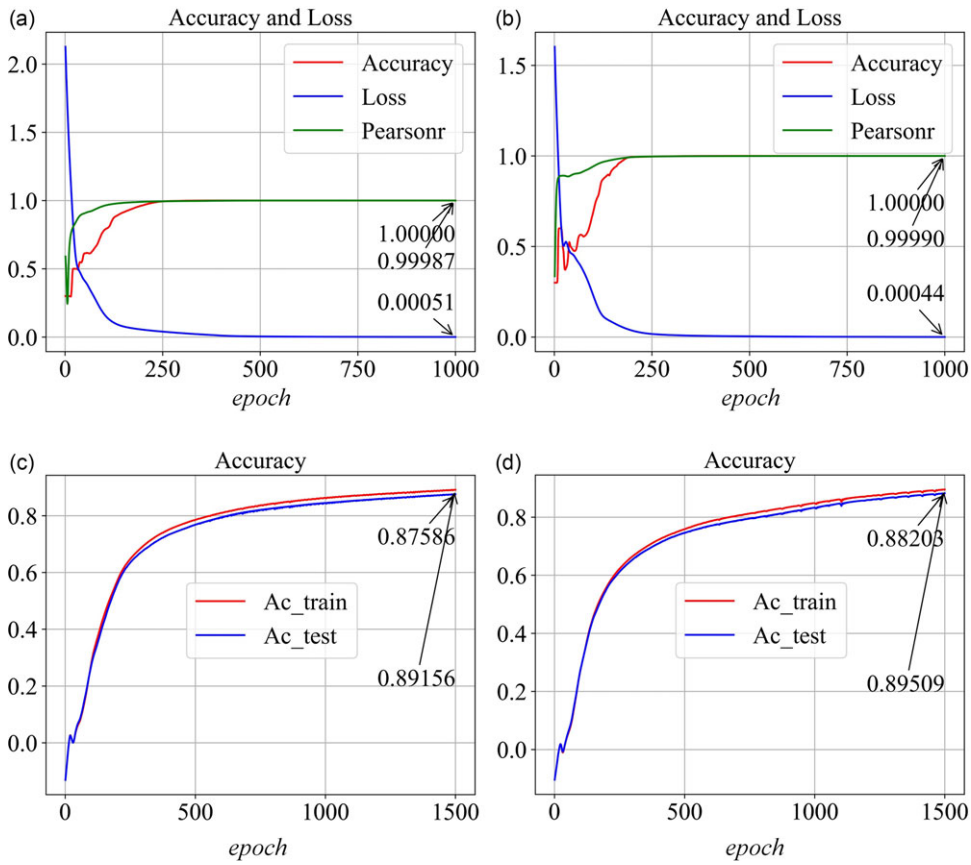



***Figure 2.*** *Performance comparison of CNN and improved model. (a) Accuracy and loss of output matrix of CNN model. (b) Accuracy and loss of output matrix of improved model. (c) Accuracy of motion/constraint prediction of CNN model. (d) Accuracy of motion/constraint prediction of CNN model.*

where $n_s$ is the scale of the specimen. $y_i$ and $\hat{y}_i$ are real value and prediction value of the model, respectively. $MSE_{\{u\}}$ is applied as mean-square error (MSE) of network. $SSE$ denotes the sum of squares due to error. $w_{mse1}$ is the weight of $MSE_{\{u\}}$ in loss function, where $w_{mse1} = 1$.

When taking partial derivative of network weight $w_0$ of on both sides of $MSE_{\{u\}}$, the descending gradient goes toward the fewer error of network. The loss and Pearson correlation coefficient between the real and prediction outputs are illustrated in Fig. 2 (a), resulting in the accuracy of prediction model.

It is noted that reciprocal screws should satisfy the numerical relationship as accurate as possible. Therefore, a physic constraint is introduced as

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

**Algorithm 2: Neural network-based constraint solving model**

Input: Training samples $\mathrm{P} = [\mathbf{m}_i;\mathbf{input}_i] \rightarrow [\mathbf{output}_i]$

Output: Constraint solving model $\Gamma_1(\mathbf{m}_i) = \mathbf{f}_i$

1. Initialization:

Assign an initial random weight to the neural network model $\omega_0$, learning rate $\alpha_c = 0.001$, $Max_enpoch = 1000$, $enpoch = 0$; Define the sample as $\mathrm{P} = [\mathbf{m}_i;\mathbf{input}_i] \rightarrow [\mathbf{output}_i]$; The inputs to the NN model are $\mathbf{m}_i$ and $\mathbf{input}_i$; The output is $\mathbf{output}_i$; Constraint solution model defines the input as the limb motion matrix $\mathbf{m}_i$ and the output force matrix is defined as $\mathbf{f}_i$.

2 Initialize random weights $\omega_0$;

3 Defining the network net();

4 For *enpoch* in range(1, *Max_enpoch*):

5 Input samples P;

6 Gradient zeroing;

8 $\mathbf{output} = \mathrm{net}(\mathbf{intput})$;

9 $\mathbf{f}_i = \mathrm{F}(\mathbf{output}, \mathbf{m}_i)$;

7 Calculate the gradient loss of the *enpoch*th training by Eq. ;

8 Update network parameters: $\omega_0 = \omega_0 - \alpha_c \cdot \frac{\delta Loss}{\delta \omega_0}$;

9 Calculate Accuracy and save (the accuracy defined in this case is the Pearson correlation coefficient between the matrices);

10 Gradient propagation forward into the model;

11 End For

12. Output Model $\Gamma_1(\mathbf{m}_i) = \mathbf{f}_i$;

$$MSE_{\$\circ\$^r} = \frac{1}{n}\sum_{i=1}^{n_S} w_{mse2}\left(\$_{t,i} \circ \mathrm{F}\left(y_i\right) - \$_{t,i} \circ \mathrm{F}\left(\hat{y}_i\right)\right)^2 \tag{3k}$$

where $MSE_{\$\circ\$^r}$ indicates the mean-square error of reciprocal product. $w_{mse2}$ is the weight where $w_{mse2} = 0.001 * epoch$, and *epoch* is the iteration. "$\circ$" denotes the reciprocal product between the twist $\$_{t,i}$ of the $i$th motion derived from motion matrix and the real constraint wrench $\mathrm{F}(y_i)$ (the predictive constraint wrench $\mathrm{F}(\hat{y}_i)$). Therefore, the compressive and real-time loss function is proposed as

$$Loss = MSE_{\{u\}} + MSE_{\$\circ\$^r} \tag{3l}$$

To this end, a novel network is formulated as Improved Neural Network by Algorithm 2. The architecture of the network is the same with CNN in terms of the convolutional layers used. Besides the loss of model considered in CNN, the novel network also learns the physic principle as reciprocity between motion and constraint, which can improve the generalization of the predictive model. The performance of new model is depicted in Fig. 2 (b), which decreases the loss as small as 0.044%. When applying the prediction model on training and test sets, the probability to get accurate motion/constraint is presented in Fig. 2 (c) and (d), respectively. It shows that the improved model has a better accuracy as high as 89.5% for a new issue.

It shows that improved model could reach higher accuracy. To find out the reason, $MSE_{\$\circ\$^r}$ computed from both CNN and improved model are also compared in Fig. 3. It is recorded that improved model has

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

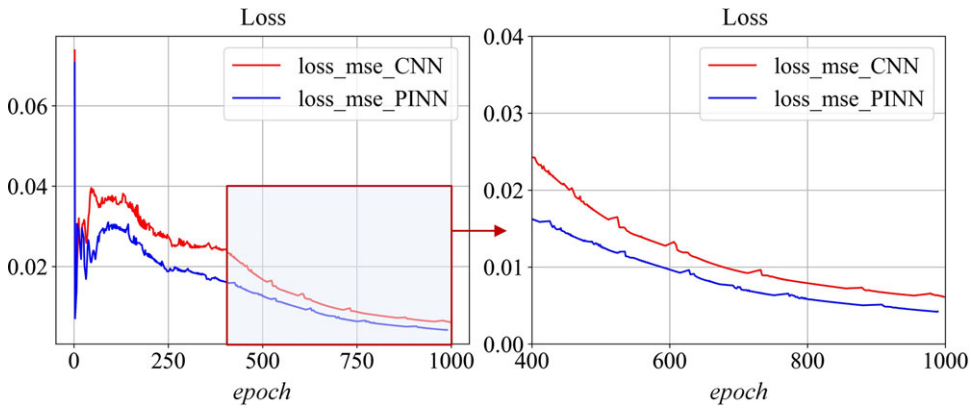



***Figure 3.*** *$MSE_{\$\circ\$^r}$ comparison of improved model and CNN.*

the better performance in the loss of reciprocal product, resulting from the loss function with physical equation. Although the difference would be decreased following with training generations, the former model is prior than the latter one as 30% after iterative 1000.

Therefore, improved model is applied for predictive modeling. In fact, from the view of meaningful motion and constraint, both the errors of numerical and geometric features should be considered; that is why improved model has better performance in this case.

## 4. Application in automatic mobility analysis

Applying the reciprocal screw solving method, the procedure to identify the mobility of parallel mechanisms could be proposed by four steps as shown Fig. 4:

**Step 1:** Having topology character at hand, motion matrix would be obtained automatically by Algorithm 1. It would be identified that whether there exist distal joints. If so, the relationship after changing would also be constructed.

**Step 2:** The motion matrix from step 1 is transformed as boolean matrix, which is the input of improved predictive model. The constraints of all limbs are obtained in the format of boolean matrix as output.

**Step 3:** The independent constraints in boolean matrix format are given to the improved predictive model. The motion of moving platform is derived and transformed as DoF matrix.

$$\mathbf{DoF} = \begin{bmatrix} \boldsymbol{s}_1 & \boldsymbol{r}_1 & h_1 \\ \vdots & \vdots & \vdots \\ \boldsymbol{s}_n & \boldsymbol{r}_n & h_n \end{bmatrix}_{n*7} \tag{3m}$$

Taking 3-CRR mechanism as an example, the DoF matrix would be obtained as

$$\mathbf{DoF} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{3n}$$

**Step 4:** For the mechanism exists distal joints, the full-cycle motion is distinguished. For example, the first and last rotational joints of UPU limb are considered as distal joints. When the second and the fourth joints have parallel motion axes and complementary motion, the geometry condition of distal joints would not be changed in the movement. But in the other case, the motion axes of distal joints would change to be intersecting or in different flats. At this moment, the motion and constraints would be changed. The mobility analysis should be considered separately. The situations for distal joints without changing mobility and constraints are concluded in Table VIII.

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

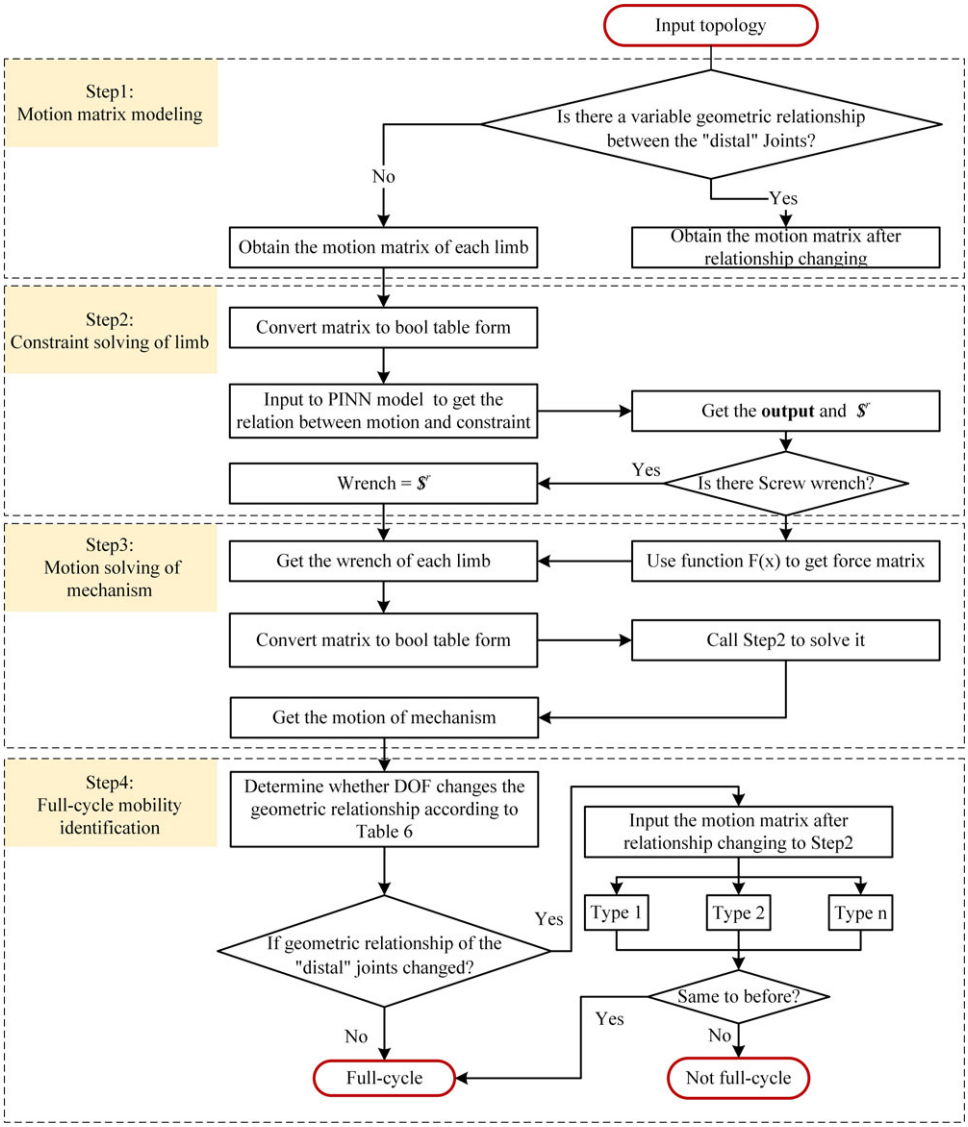



***Figure 4.*** *Flowchart of mobility analysis of parallel mechanisms.*

## 5. Software implementation

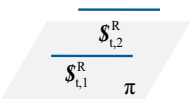
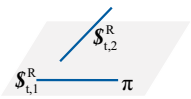
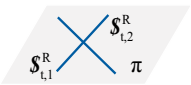
By integrating the algorithms and models with *Python* 3.7 programming language, a software is developed under B/S framework supporting automatic and visualized procedure for motion and constraint analysis of parallel mechanisms. Three interactive windows are designed as input of topology, output of mobility analysis, and visualization of constraints as shown in Fig. 5.

1. Input of topology

This window allows users to input the required topology characters, which is located on the right of the interactive interface. Users would give the number of limbs firstly followed with the topology characters of each limb. If someone needs help, he can find out the rules in tips "?". Thanks to Algorithm 1, topology characters would be identified by the software, which would be transformed as motion matrix to the next step.

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

***Table VIII.*** *Identification of distal joints.*

| Initial assembly conditions | Diagrams | Possible changes | DoF not change relationship |
|---|---|---|---|
| $\overbrace{\text{R.....R}}^{\text{parallel}}$ |  | Heterogeneous or intersect | 1. Any translations in space<br>2. Rotation with parallel axis to these two lines |
| $\overbrace{\text{R....R}}^{\text{vertical}}$ |  | Heterogeneous | 1. Any translations in space<br>2. Rotation with parallel axis to the line |
| $\overbrace{\text{R....R}}^{\text{intersect}}$ |  | Heterogeneous or parallel | 1. Translations in the plane along two lines<br>2. Any rotations with axis through the intersection point |

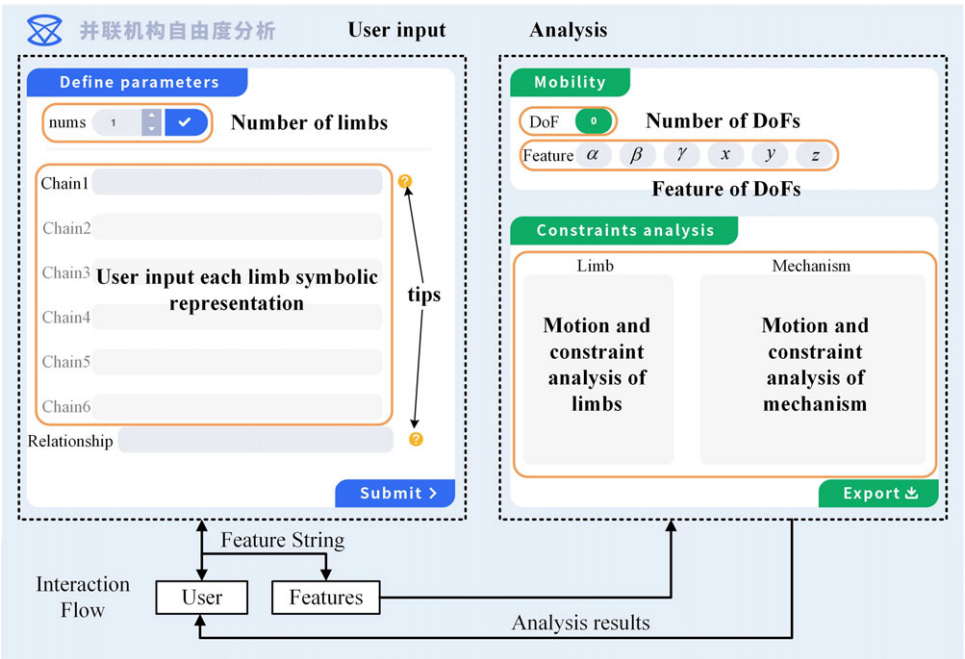



***Figure 5.*** *Module of web platform.*

2. Visualization of constraints

This module illustrates the motion axes and constraints of limbs and moving platform with lines, points, and other geometric elements. The types of geometric elements are defined and associated with different motion and constraints thanks to WebGL interface. A drawer engine is programmed. Firstly, the framework assigned as *O-xyz* is constructed on the canvas. When the topology is given and motion matrix is solved, the first joint of one limb is located at a fixed point and along *x*-axis, which is taken as a reference. Other joints and limbs are illustrated according to the motion matrix with positions determined by the marked points and directions selected by $\boldsymbol{s}_i$. The constraints would be drawn thanks

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

to the geometric conditions involved in input and output matrix. The lines of different meanings are applied by *matplotlib three-dimensional*, which is a graphic library of *Python*.

3. Output of mobility analysis

Once the topology characters are provided, the user accesses the mobility including the number and type of motions. The results are provided to users by concise alternatives on the right side. Here, NN for null space of screw system solving plays key roles for this function.

The software could be implemented by the following steps:

1. Input the topology characters such as the number and types of limbs into the software.
2. Motion matrix of each limb is obtained by Algorithm 1 and equivalent transformed to be $\mathbf{m}_i$.
3. Taking $\mathbf{m}_i$ as the input of model $\Gamma(\mathbf{m}_i) = \mathbf{f}_i$, constraint matrix of each branch would be obtained as.
4. Taking $\mathbf{f} = [\mathbf{f}_1, \cdots, \mathbf{f}_n]$ as the input of model $\Gamma(\mathbf{f}) = \mathbf{m}$, motion matrix of moving platform would be obtained as $\mathbf{m}$.
5. Special geometric relationship as "distal" kinematic joints would be determined based on Table VIII. If so, steps (3) and (4) are repeated.
6. Input $\mathbf{m}_i$, $\mathbf{f}_i$, $\mathbf{m}$, $\mathbf{f}$ to WebGL drawer engine.
7. Output the results of mobility and constraints to users.

## 6. Cases study and discussion

### *6.1. 3-CRR mechanism*

The motion and constraints analysis of 3-CRR has been addressed above, which is not introduced here again. When topology character as in Eq. (2a) is entered the into the software, the result could be obtained as in Fig. 6.

It is concluded that each limb of 3-CRR mechanism has two constraint couples, which are perpendicular with the motion axes of R joint. The moving platform of mechanism undergoes three constraint couples, resulting in three translational motions, which is a full-cycle mobility.

### *6.2. 3-UPU mechanism*

3-UPU is a typical parallel mechanism with different assembly conditions and alternative mobilities. It is assumed that the motion axes of 1st and 5th rotational joints are parallel at initial state, which constitute as "distal" joints. However, not only assembly conditions in the same limb but also among different limbs would affect the motion and constraints of parallel mechanism. Here, two cases are discussed to verify the method:

(a) Three UPU limbs are symmetrically assembled.

In this case, three limbs are symmetrically assembled with non-parallel plane of U joint. The topology characters should be given as

$$\left\{\begin{array}{l} x_1N_1\mathrm{R}N_1 \mid u_1N_1\mathrm{R}N_1 \mid N_1N_2\mathrm{P} \mid u_1N_2\mathrm{R}N_2 \mid x_1N_2\mathrm{R}N_2 \\ x_2N_3\mathrm{R}N_3 \mid u_2N_3\mathrm{R}N_3 \mid N_3N_4\mathrm{P} \mid u_2N_4\mathrm{R}N_4 \mid x_2N_4\mathrm{R}N_4 \\ x_3N_5\mathrm{R}N_5 \mid u_3N_5\mathrm{R}N_5 \mid N_5N_6\mathrm{P} \mid u_3N_6\mathrm{R}N_6 \mid x_3N_6\mathrm{R}N_6 \end{array}\right\} \tag{6a}$$

The motion matrix of limb is solved as

$$\mathbf{m}_i = \begin{bmatrix} a_{i1} & b_{i1} & 0 & l_{i1} & m_{i1} & n_{i1} \\ a_{i2} & b_{i2} & c_{i2} & l_{i1} & m_{i1} & n_{i1} \\ 0 & 0 & 0 & l_{i3} - l_{i1} & m_{i3} - m_{i1} & n_{i3} - n_{i1} \\ a_{i2} & b_{i2} & c_{i2} & l_{i3} & m_{i3} & n_{i3} \\ a_{i1} & b_{i1} & 0 & l_{i3} & m_{i3} & n_{i3} \end{bmatrix} \tag{6b}$$

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

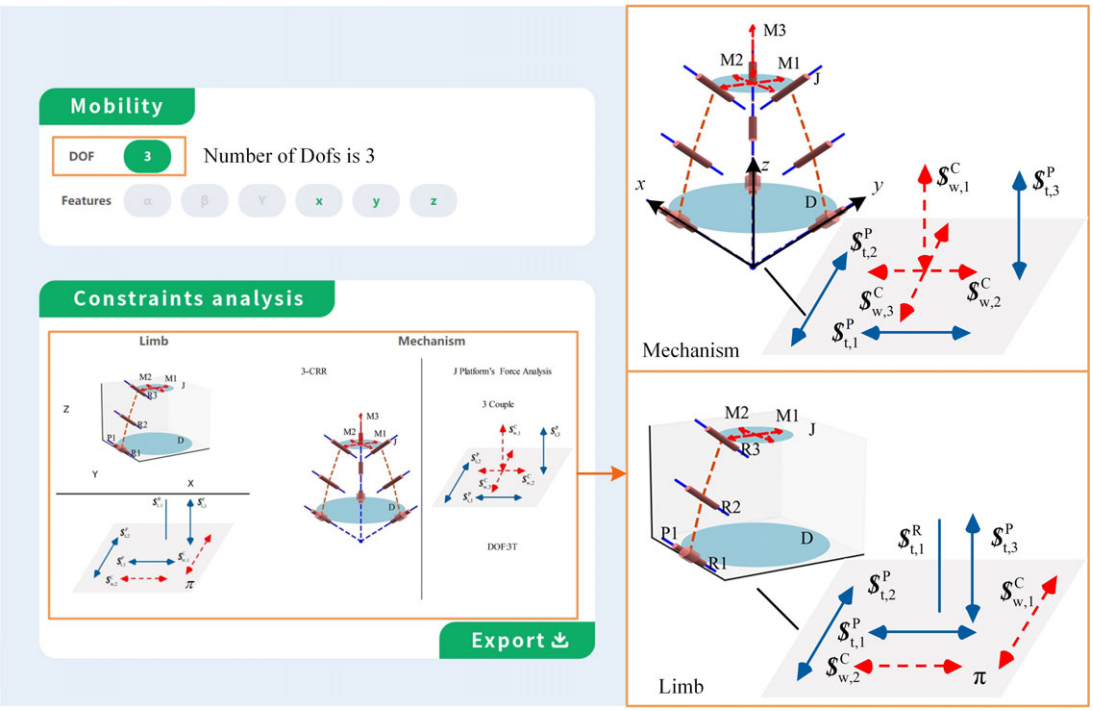



***Figure 6.*** *Motion and constraint analysis result of 3-CRR mechanism.*

The motion matrix of the $i$th limb lies in the format as

$$\mathbf{input}_i = \begin{bmatrix} 2 & 3 & 4 & 1 & -2 \\ 2 & 3 & 4 & 1 & -2 \\ 2 & 3 & 4 & 0 & -1 \\ 2 & 3 & 4 & 0 & -1 \\ 2 & 3 & 4 & 0 & -1 \end{bmatrix} \tag{6c}$$

As illustrated in Fig. 7, each UPU limb provides a couple constraint, which is perpendicular to its plane of U joint. The moving platform has three independent couples, leading to the matrix for DoF

$$\mathbf{DoF} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{6d}$$

It indicates that 3-UPU with fully symmetric topology has three translational motions. The "distal" joints are falling in situation (a) listed in Table VIII, which would not change the constraints and further the mobility. Therefore, 3-UPU has a full-cycle three-dimensional translational motion [25].

(b) Motion axes of first joint of three limbs are parallel.

In this case, the planes of U joint of limbs are parallel with each other. The input would be

$$\left\{ \begin{matrix} x_1N_1\mathrm{R}N_1 \mid x_2N_1\mathrm{R}N_1 \mid N_1N_2P \mid x_2N_2\mathrm{R}N_2 \mid x_1N_2\mathrm{R}N_2 \\ x_1N_3\mathrm{R}N_3 \mid x_2N_3\mathrm{R}N_3 \mid N_1N_2P \mid x_2N_4\mathrm{R}N_4 \mid x_1N_4\mathrm{R}N_4 \\ x_1N_5\mathrm{R}N_5 \mid x_2N_5\mathrm{R}N_5 \mid N_1N_2P \mid x_2N_6\mathrm{R}N_6 \mid x_1N_6\mathrm{R}N_6 \end{matrix} \right\} \tag{6e}$$

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

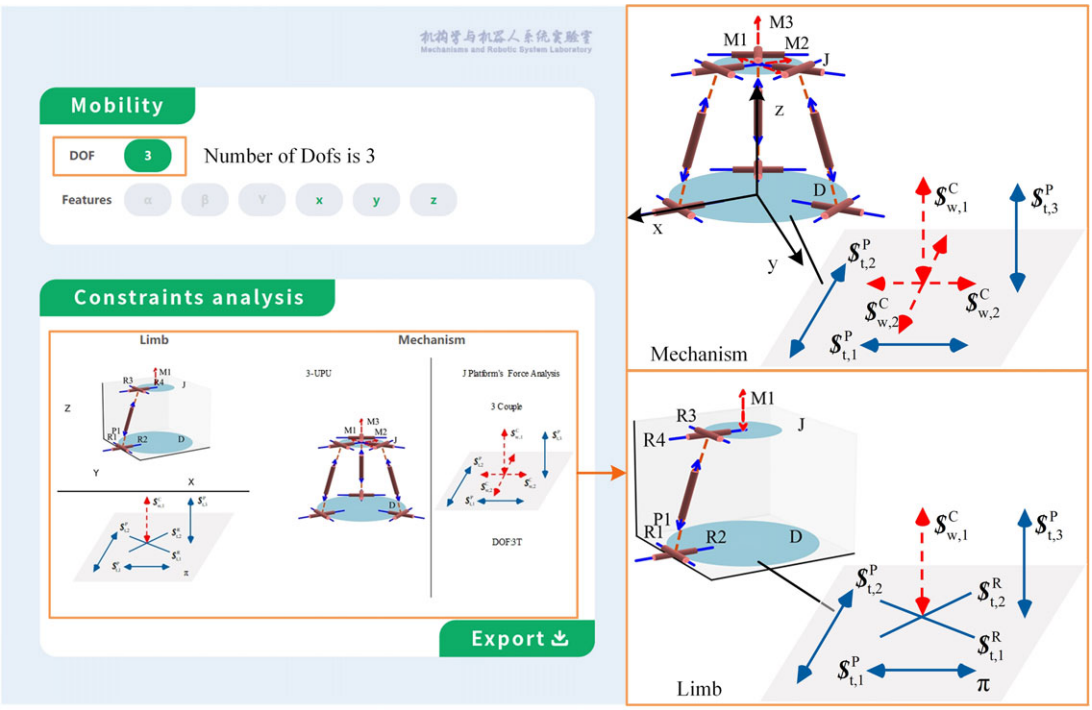



***Figure 7.*** *Motion and constraint analysis result of 3-UPU mechanism with symmetric structure.*

Motion matrix of limb is formulated easily as

$$\mathbf{m}_i = \begin{bmatrix} a_{i1} & b_{i1} & 0 & l_{i1} & m_{i1} & n_{i1} & 0 \\ a_{i2} & b_{i2} & 0 & l_{i1} & m_{i1} & n_{i1} & 0 \\ 0 & 0 & 0 & l_{i3} & m_{i3} & n_{i3} & 0 \\ a_{i2} & b_{i2} & 0 & l_{i4} & m_{i4} & n_{i4} & 0 \\ a_{i1} & b_{i1} & 0 & l_{i4} & m_{i4} & n_{i4} & 0 \end{bmatrix} \tag{6f}$$

The output indicates a couple

$$\mathbf{output}_i = \begin{bmatrix} 2 & 0 & 0 & -1 \end{bmatrix} \tag{6g}$$

It is noted that each limb provides a same couple. In this moment, the DoF matrix would be

$$\mathbf{DoF}_{11} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6h}$$

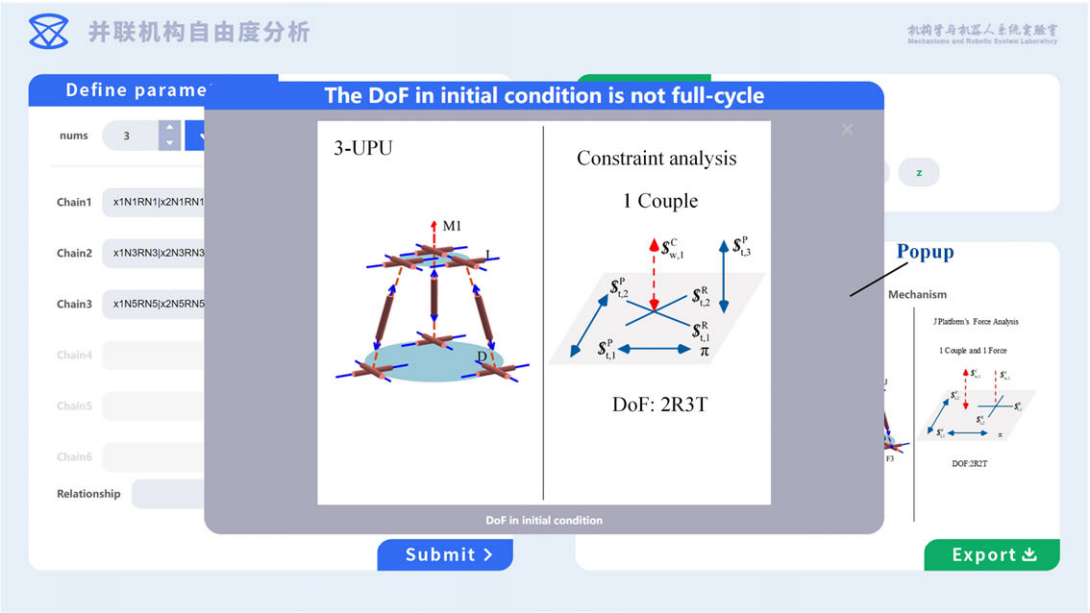
which indicates two rotational and three translational motions. There exist "distal" joints, but out of the scope without changing, it will affect the constraints and mobility. Therefore, the mobility indicated in Eq. (6e) is not full cycle, in which case the software would inform users by a popup as in Fig. 8 (a).

The motion matrix would be changed as

$$\mathbf{m}_i = \begin{bmatrix} a_{i1} & b_{i1} & 0 & l_{i1} & m_{i1} & n_{i1} & 0 \\ a_{i2} & b_{i2} & 0 & l_{i1} & m_{i1} & n_{i1} & 0 \\ 0 & 0 & 0 & l_{i3} & m_{i3} & n_{i3} & 0 \\ a_{i2} & b_{i2} & 0 & l_{i4} & m_{i4} & n_{i4} & 0 \\ a_{i1} & b_{i1} & c_{i5} & l_{i1} & m_{i1} & n_{i1} & 0 \end{bmatrix} \tag{6i}$$

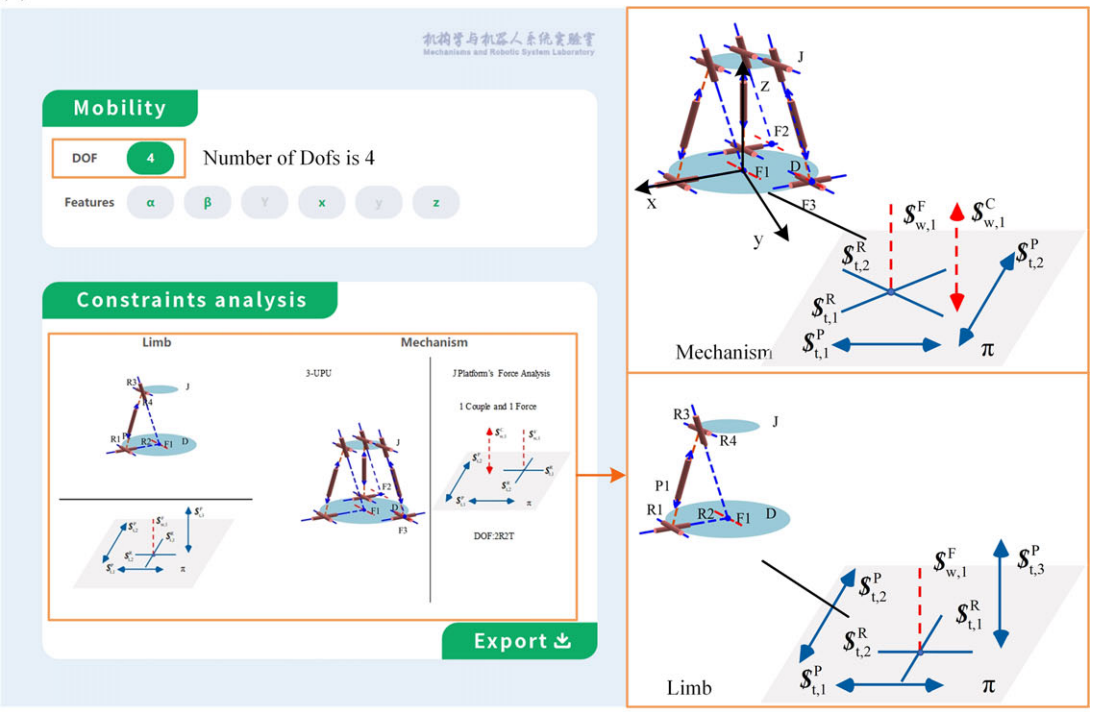
https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

(a)




Non full cycle mobility of 3-UPU with parallel axes

(b)




Full-cycle mobility of 3-UPU with parallel axes

***Figure 8.*** *Motion and constraint analysis result of 3-UPU mechanism with parallel axes.*

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

The output of constraint from the $i$th limb is written as

$$\mathbf{output}_i = \begin{bmatrix} 2 & 1 & 1 & -2 \end{bmatrix} \tag{6j}$$

which indicates a constraint force. In this moment, the DoF matrix would be

$$\mathbf{DoF} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{6k}$$

which indicates two rotational and two translational motions, as illustrated in Fig. 8 (b).

Compared with case (a), it could be seen that the arrangement of limbs would affect the resultant mobility. It is necessary to observe the constraints in the process. In this situation, the mobility at initial pose is considered as non-full cycle DoF, which would be changed once there is a rotational movement between the 1st and the last joints. The software depicts this phenomenon and shows it to users by a window. When the motion axes have been changed, it falls in the four DoF, which is a full-cycle mobility which is illustrated in the main page. From this point, the initial pose could be considered as a singular pose of the mechanism. In this way, the software can provide much information to users rather than the mobility.

### *6.3. Exechon robot*

Exechon robot is famous in machining, which is composed of two UPR limbs and one SPR limb. The topology could be characterized as

$$\left\{ \begin{array}{c} x_1N_1\mathrm{R}N_1 \mid x_2N_1\mathrm{R}N_1 \mid N_1N_2\mathrm{P} \mid x_2N_2\mathrm{R}N_2 \\ x_1N_1\mathrm{R}N_1 \mid x_2N_1\mathrm{R}N_1 \mid N_1N_2\mathrm{P} \mid x_2N_2\mathrm{R}N_2 \\ u_1N_4\mathrm{R}N_4 \mid u_2N_4\mathrm{R}N_4 \mid u_3N_4\mathrm{R}N_4 \mid N_4N_5\mathrm{P} \mid x_3N_5\mathrm{R}N_5 \end{array} \right\} \tag{6l}$$

Based on the algorithms and predictive model, the result of mobility and constraint analysis is generated by the software as shown in Fig. 9, which is consistent with the result in ref. [26].

It could be seen from the visual constraints that UPR limb provides one couple and one force constraints, and SPR limb provides one force constraints. It is a redundant mechanism which has two couples and three forces. Among them, one couple and two forces are independent. Therefore, the mechanism has two rotational motions and one translational motion. The detail process is given in appendix.

### *6.4. Omni III robot*

Omni III robot is composed of four RRRR limbs. The first joints of limbs are perpendicular with each other, and the adjacent two joints are intersected at one point. The input could be given as

$$\left\{ \begin{array}{c} x_1N_1\mathrm{R}N_1 \mid u_1N_1\mathrm{R}N_2 \mid u_2N_2\mathrm{R}N_3 \mid x_1N_3\mathrm{R}N_3 \\ x_2N_1\mathrm{R}N_1 \mid u_3N_1\mathrm{R}N_4 \mid u_4N_4\mathrm{R}N_3 \mid x_2N_3\mathrm{R}N_3 \\ x_2N_1\mathrm{R}N_1 \mid u_5N_1\mathrm{R}N_5 \mid u_6N_5\mathrm{R}N_3 \mid x_2N_3\mathrm{R}N_3 \\ x_1N_1\mathrm{R}N_1 \mid u_7N_1\mathrm{R}N_6 \mid u_8N_6\mathrm{R}N_3 \mid x_1N_3\mathrm{R}N_3 \end{array} \right\} \tag{6m}$$

Based on the algorithms and predictive model, the result of mobility and constraint analysis is generated by the software as shown in Fig. 10, which is consistent with the result in ref. [27].

It could be seen from the visual constraints that each RRRR limb provides two force constraints in the plane $\pi$. It is a redundant mechanism which has eight forces. Among them, three forces and one couple are independent. Therefore, the mechanism has two rotational motions, which can move as a simi-sphere and is famous as simi-sphere mechanism. The detail process is given in appendix.

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

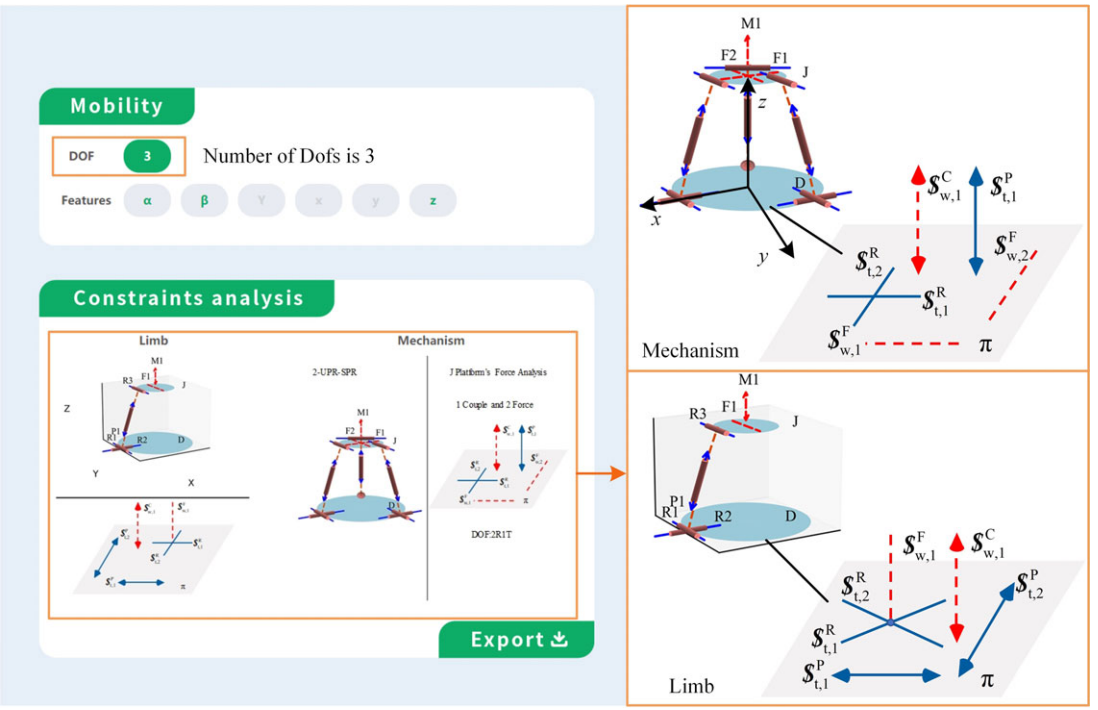



***Figure 9.*** *Motion and constraint analysis result of exechon mechanism.*

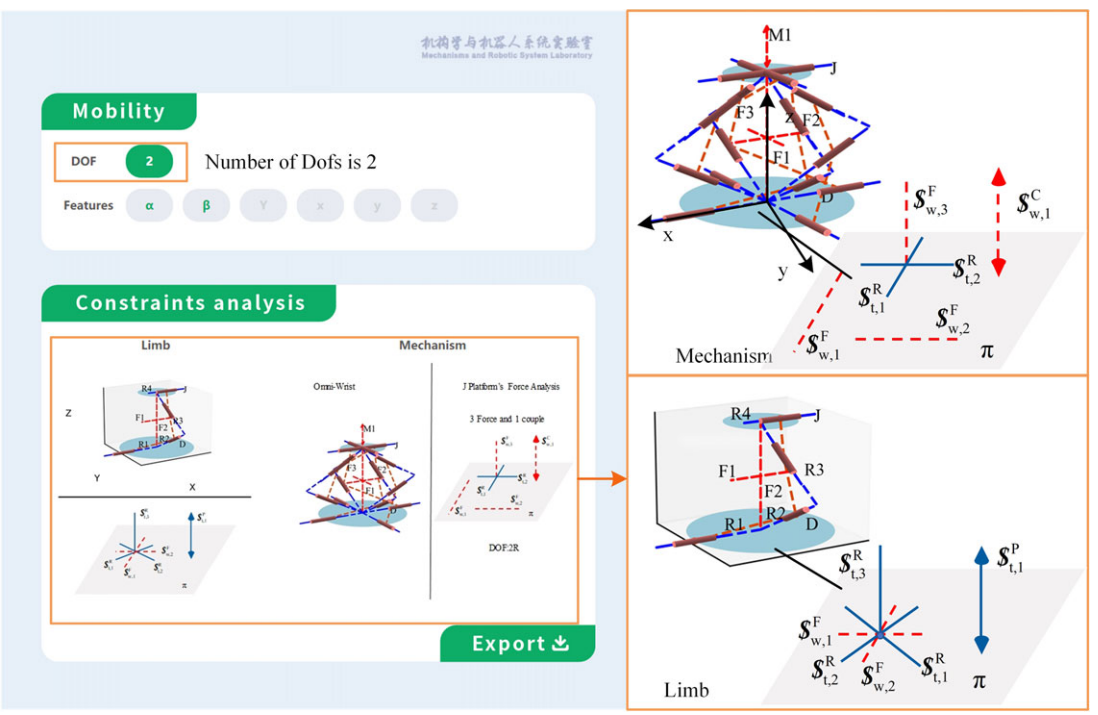



***Figure 10.*** *Motion and constraint analysis result of Omni III robot.*

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

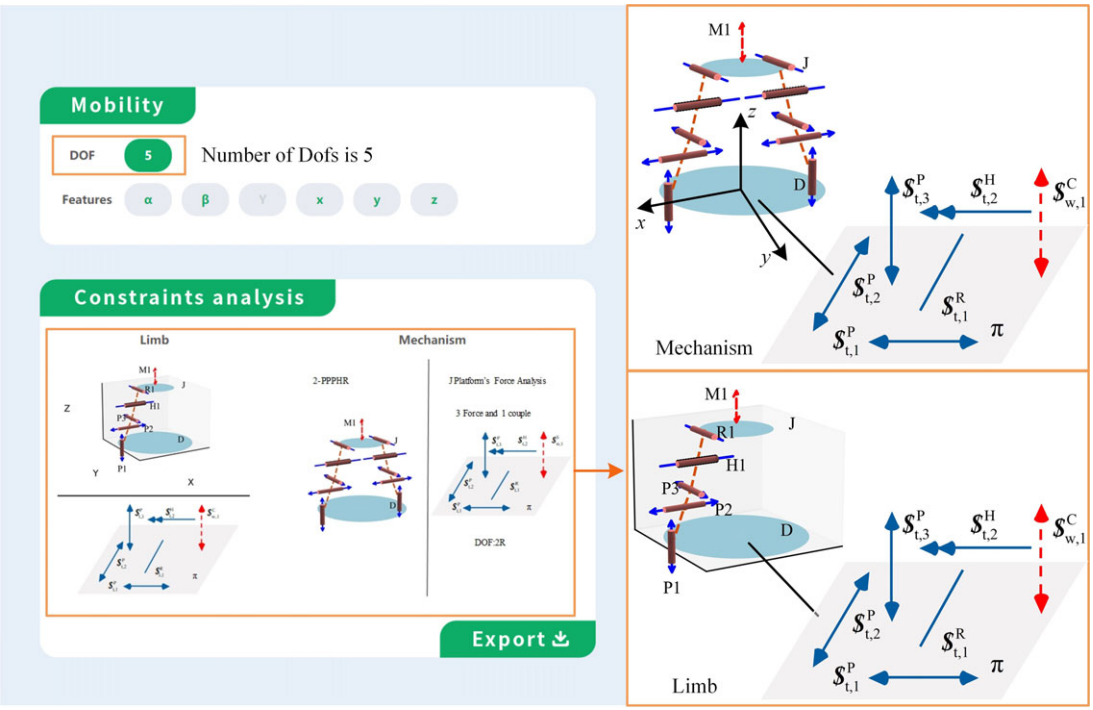



***Figure 11.*** *Motion and constraint analysis result of 2-PPPHR.*

### *6.5. 2-PPPHR mechanism*

2-PPPHR mechanism is composed of two PPPHR limbs. The first joints of limbs are perpendicular with each other, and the adjacent two joints are intersected at one point. The input could be given as

$$\left\{\begin{matrix} z_1 z_1 \mathrm{P} \mid x_1 x_1 \mathrm{P} \mid x_2 x_2 \mathrm{P} \mid x_2 N_1 \mathrm{H} N_1 \mid x_3 N_2 \mathrm{R} N_2 \\ z_1 z_1 \mathrm{P} \mid x_1 x_1 \mathrm{P} \mid x_2 x_2 \mathrm{P} \mid x_2 N_3 \mathrm{H} N_3 \mid x_3 N_4 \mathrm{R} N_4 \end{matrix}\right\} \tag{6n}$$

Based on the algorithms and predictive model, the result of mobility and constraint analysis is generated by the software as shown in Fig. 11, which is consistent with the result in ref. [22].

It could be seen from the visual constraints that the two PPPHR limbs provide a same couple constraint. Therefore, the mechanism has two rotational and three translational motions. The detail process is given in appendix.

### *6.6. Discussion*

As a preliminary step, twist and wrench should be solved in a concise and efficient manner. To achieve this, two primary methods have been developed: numerical computation and observation of null-space.

For the numerical method, alternative solutions could be obtained when the number of equations is less than five. These solutions satisfy the algebraic relationship but are failing with the geometric features. For example, the constraints of limbs in cases 6.1–6.4 would have multi-solutions by solving null-space. In contrast, the predictive approach applies artificial intelligence to learn the geometric features of reciprocal screws, which are granted the real meanings.

For the observation method, it requires extensive human knowledge, which is hard to be programmed and handle with the amount of cases. Furthermore, when there exist helical joints in the screw system, observation method would not be comfortable anymore. Besides geometric features, the predictive

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

***Table IX.*** *Comparison with existing theoretical analysis methods.*

| | **Observation method** | **Numerical method** | **This method** |
|---|---|---|---|
| With screw motion/force | × | √ | √ |
| Programming | × | √ | √ |
| Visualization | √ | × | √ |
| Explicit meanings | √ | × | √ |

approach also takes algebraic properties as loss function, which can improve the accuracy and solve the case with helical joint like case 6.5.

As a universal function fitter, NN has become another way to solve the above problems. A single perceptron can simulate three operations: AND, OR, and NOT. Three perceptrons forming a perceptron network including one hidden layer can simulate all operations. Therefore, NNs can simulate any combination of logical functions in theory. Specifically, this ability is mainly based on the following aspects:

1. Machine learning can identify complex patterns and rules from data, including relationships between inputs and outputs. These patterns and rules can be abstracted into the "IF. . .THEN. . ."-type conditional statements.
2. Machine learning constructs models by learning and training on datasets, rather than relying on manually crafted rules. This allows machine learning to automatically extract features and patterns from data.
3. Once the patterns and rules are learned from data, the machine learning typically generalizes this knowledge to previously unseen data. This means that even when dealing with new situations or data, the model can make accurate predictions or decisions.

Using the powerful fitting capabilities of machine learning, the results obtained from this approach would be consistent with those of manual observation and numerical computation, and the advantages of the proposed automated method are summarized in Table IX.

## 7. Conclusions

This paper investigates the automatic identification method for motion/constraint and mobility of parallel mechanisms based on machine learning. The following conclusions could be drawn:

1. Topology is characterized by symbols, including the type of limbs with the direction and position of motion axis relative to fixed platform. Motion matrix is defined by signing key points. The mapping between topology and motion is formulated by algorithm.
2. Predictive model for reciprocal screw is constructed by NN, which reaches 89.5% accuracy by defining a physical-informed loss function. In this way, motion and constraints are solved with natural meanings and numerical properties.
3. Automatic mobility analysis procedure is proposed by four steps based on the predictive model. Full-cycle mobility is identified considering the "distal" joints.
4. The online software for visualization constraint and mobility analysis is developed, and five typical examples are given.

The automatic approach proposed in this article not only provides the results from mobility analysis but also gives an insight investigation of meaningful motion and constraints. It offers additional benefits by providing clear visual information and aiding in kinematic modeling, including singularity analysis. In our future work, time series data structures for finite motion description would be well investigated.

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

Additionally, we aim to integrate topology synthesis, analysis, and design visually and intelligently, taking into account both the characteristics of finite motion and constraints.

**Author contributions.** Huo and Song constructed the machine learning model. Sun developed the demo and wrote the article.

**Financial support.** This research work was supported by the National Natural Science Foundation of China (NSFC) under Grant No. 52205028, No. 62027812 and No. 52275027, and Tianjin Research Program of Technology under Grant No. 20201193 and No. 22JCZDJC00350.

**Competing interests.** The authors declare no competing interests.

**Ethical approval.** None.

## References

[1] Z. Huang, Q. Li and H. Ding, "Dependency and Reciprocity of Screws," **In:** *Mechanisms and Machine Science* (Springer International Publishing, Switzerland, 2013).
[2] K. H. Hunt, *Kinematic Geometry of Mechanisms* (Oxford University Press, Oxford University, 1978).
[3] C. D. Meyer. *Matrix Analysis and Applied Linear Algebra* (Society for Industrial and Applied Mathematics, 2000).
[4] K. Sugimoto and J. Duffy, "Application of linear algebra to screw systems," *Mech Mach Theory* **17**(1), 73–83 (1982).
[5] J. S. Dai and J. R. Jones, "A linear algebraic procedure in obtaining reciprocal screw systems," *J Robot Syst* **20**(7), 401–412 (2003).
[6] H. Ding, W. Cao, C. Cai and A. Kecskeméthy, "Computer-aided structural synthesis of 5-DOF parallel mechanisms and the establishment of kinematic structure databases," *Mech Mach Theory* **83**, 14–30 (2015).
[7] T. Huang, S. Yang, M. Wang, T. Sun and D. G. Chetwynd, "An approach to determining the unknown twist/wrench subspaces of lower mobility serial kinematic chains," *J Mech Robot* **7**(3), 031003 (2015).
[8] X. Pei and J. Yu, "A visual graphic approach for mobility analysis of parallel mechanisms," *Front Mech Eng* **6**(1), 92–95 (2011).
[9] X. Huo, T. Sun and Y. Song, "A geometric algebra approach to determine motion/constraint, mobility and singularity of parallel mechanism," *Mech Mach Theory* **116**, 273–293 (2017).
[10] H. Liu, A. Kecskeméthy and T. Huang, "An automatic approach for identification of natural reciprocal screw systems of serial kinematic chains based on the invariance properties matrix," *Mech Mach Theory* **107**, 352–368 (2017).
[11] G. Atmeh and K. Subbarao, "A dynamic neural network with feedback for trajectory generation," *IFAC-PapersOnLine* **49**(1), 367–372 (2016).
[12] X. Liu, J. Yao, Q. Li and Y. Zhao, "Coordination dynamics and model-based neural network synchronous controls for redundantly full-actuated parallel manipulator," *Mech Mach Theory* **160**, 104284 (2021).
[13] S. Freitag, S. Peters, P. Edler and G. Meschke, "Reliability-based optimization of structural topologies using artificial neural networks," *Probab Eng Mech* **70**, 103356 (2022).
[14] X. Zhu, H. Shen, C. Wu, D. Chablat and T. Yang, "Computer-aided mobility analysis of parallel mechanisms," *Mech Mach Theory* **148**, 103810 (2020).
[15] X. Meng and F. Gao, "A framework for computer-aided type synthesis of parallel robotic mechanisms, proceedings of the institution of mechanical engineers," *Proceed Inst Mech Eng Part C: J Mech Eng Sci* **228**(18), 3496–3504 (2014).
[16] X. Zhu, X. Yao, H. Shen and T. Yang, "Structural Synthesis Based On POC Set for Lower-Mobility Non-Overconstrained Parallel Mechanisms," **In:** *Mechanisms and Machine Science* (Springer International Publishing, Switzerland, 2016).
[17] X. Zhu, H. Shen, C. Wu and Yang T., "Automatic Mobility Analysis of Parallel Mechanisms Based On Position and Orientation Characteristic Equation. Part II: Mobility Analysis and Examples," **In:** *Mechanisms and Machine Science* (Springer International Publishing, Switzerland, 2019).
[18] Q. Jin and T.-L. Yang, "Theory for topology synthesis of parallel manipulators and its application to three-dimension-translation parallel manipulators," *J Mech Design* **126**(4), 625–639 (2004).
[19] T. Yang, A. Liu, H. Shen, L. Hang, Y. Luo and Q. Jin. *Topology Structure Design of Robot Mechanisms* (Springer Nature Singapore Pte Ltd, Singapore, 2018).
[20] J. Du, "Automatic mobility analysis of parallel mechanisms based on geometric algebra," *Tran Chin Soc Agric Mach* **49**(6), 400–407 (2018).
[21] G. Gogu, "Mobility of mechanisms: A critical review," *Mech Mach Theory* **40**(9), 1068–1097 (2005).
[22] J. Li, G. Wu, H. Shen, A. Muller, A. Liu and T.-L Yang, "Topology of robotic mechanisms: Framework and mathematics methods-in conjunction with a review of four original theories," *Mech Mach Theory* **175**, 104895 (2022).
[23] T. Sun, S. Yang and B. Lian. *Finite and Instantaneous Screw Theory in Robotic Mechanism, Springer Tracts in Mechanical Engineering* (Springer International Publishing, 2020).
[24] X. Kong and C. M. Gosselin, "Kinematics and singularity analysis of a novel type of 3-CRR 3-DOF translational parallel manipulator," *Int J Robot Res* **21**(9), 791–798 (2002).

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

[25] C. Zhao, Z. Chen, Y. Li and Z. Huang, "Motion characteristics analysis of a novel 2R1T 3-UPU parallel mechanism," *J Mech Design* **142**(1), 012302 (2019).
[26] B. Hu, "Kinematically identical manipulators for the exechon parallel manipulator and their comparison study," *Mech Mach Theory* **103**, 117–137 (2016).
[27] E. Chang-Siu, A. Snell, B. W. McInroe, X. Balladarez and R. J. Full, "How to use the omni-wrist III for dexterous motion: An exposition of the forward and inverse kinematic relationships," *Mech Mach Theory* **168**, 104601 (2022).

## Appendix

a. ***Exechon robot***

(1) Step 1: motion matrix could be

$$\mathbf{m}=\begin{bmatrix} \mathbf{m}_1=\begin{bmatrix} 1 & 0 & 0 & l_{11} & m_{11} & n_{11} \\ 0 & 1 & 0 & l_{11} & m_{11} & n_{11} \\ 0 & 0 & 0 & l_{14}-l_{11} & m_{14}-m_{11} & n_{14}-n_{11} \\ 1 & 0 & 0 & l_{14} & m_{14} & n_{14} \end{bmatrix} \\ \mathbf{m}_2=\begin{bmatrix} 1 & 0 & 0 & l_{21} & m_{21} & n_{21} \\ 0 & 1 & 0 & l_{21} & m_{21} & n_{21} \\ 0 & 0 & 0 & l_{24}-l_{21} & m_{24}-m_{21} & n_{24}-n_{21} \\ 1 & 0 & 0 & l_{24} & m_{24} & n_{24} \end{bmatrix} \\ \mathbf{m}_3=\begin{bmatrix} a_{31} & b_{31} & c_{31} & l_{31} & m_{31} & n_{31} \\ a_{32} & b_{32} & c_{32} & l_{31} & m_{31} & n_{31} \\ a_{33} & b_{33} & c_{33} & l_{31} & m_{31} & n_{31} \\ 0 & 0 & 0 & l_{35}-l_{31} & m_{35}-m_{31} & n_{35}-n_{31} \\ 0 & 1 & 0 & l_{35} & m_{35} & n_{35} \end{bmatrix} \end{bmatrix} \tag{A-1}$$

(2) Step 2: the inputs to the classification optimization would be

$$\mathbf{input}=\begin{bmatrix} \mathbf{input}_1=\begin{bmatrix} 1 & 0 & 0 & 1 & -2 \\ 0 & 1 & 0 & 1 & -2 \\ 2 & 3 & 4 & 0 & -1 \\ 2 & 3 & 4 & 0 & -1 \end{bmatrix} \\ \cdots \\ \mathbf{input}_3=\begin{bmatrix} 2 & 3 & 4 & 1 & -2 \\ 2 & 3 & 4 & 1 & -2 \\ 2 & 3 & 4 & 1 & -2 \\ 2 & 3 & 4 & 0 & -1 \\ 0 & 1 & 0 & 2 & -2 \end{bmatrix} \end{bmatrix} \tag{A-2}$$

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

The constraints of each limb would be obtained as

$$\mathbf{output} = \left\{ \begin{array}{l} \mathbf{output}_1 = \begin{bmatrix} 2 & 1 & 1 & -2 \\ 2 & 0 & 0 & -1 \end{bmatrix} \\ \mathbf{output}_2 = \begin{bmatrix} 2 & 1 & 1 & -2 \\ 2 & 0 & 0 & -1 \end{bmatrix} \\ \mathbf{output}_3 = \begin{bmatrix} 2 & 1 & 1 & -2 \end{bmatrix} \end{array} \right\} \tag{A-3}$$

(3) Step 3: DoF analysis

$$\mathbf{DoF} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{A-4}$$

(4) Step 4: full-cycle mobility determination
There is no "distal" joint existing in the limb. The mechanism has a full-cycle mobility.

b. ***Omni III robot***

(1) Step 1: motion matrix could be

$$\mathbf{m} = \begin{bmatrix} \mathbf{m}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ a_{12} & b_{12} & c_{12} & a_{12} & b_{12} & c_{12} \\ a_{13} & b_{13} & c_{13} & 0 & 0 & c_{14} \\ 1 & 0 & 0 & 0 & 0 & c_{14} \end{bmatrix} \\ \mathbf{m}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ a_{22} & b_{22} & c_{22} & a_{22} & b_{22} & c_{22} \\ a_{23} & b_{23} & c_{23} & 0 & 0 & c_{14} \\ 1 & 0 & 0 & 0 & 0 & c_{14} \end{bmatrix} \\ \mathbf{m}_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ a_{32} & b_{32} & c_{32} & a_{32} & b_{32} & c_{32} \\ a_{33} & b_{33} & c_{33} & 0 & 0 & c_{14} \\ 1 & 0 & 0 & 0 & 0 & c_{14} \end{bmatrix} \\ \mathbf{m}_4 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ a_{42} & b_{42} & c_{42} & a_{42} & b_{42} & c_{42} \\ a_{43} & b_{43} & c_{43} & 0 & 0 & c_{14} \\ 1 & 0 & 0 & 0 & 0 & c_{14} \end{bmatrix} \end{bmatrix} \tag{A-5}$$

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

(2) Step 2: the inputs to the classification optimization would be

$$\mathbf{input} = \begin{bmatrix} \mathbf{input}_1 = \begin{bmatrix} 1 & 0 & 0 & 1 & -2 \\ 2 & 3 & 4 & 2 & -2 \\ 2 & 3 & 4 & 3 & -2 \\ 2 & 3 & 4 & 0 & -1 \end{bmatrix}, \mathbf{input}_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & -2 \\ 2 & 3 & 4 & 2 & -2 \\ 2 & 3 & 4 & 3 & -2 \\ 2 & 3 & 4 & 0 & -1 \end{bmatrix} \\ \mathbf{input}_3 = \begin{bmatrix} 0 & 1 & 0 & 1 & -2 \\ 2 & 3 & 4 & 2 & -2 \\ 2 & 3 & 4 & 3 & -2 \\ 2 & 3 & 4 & 0 & -1 \end{bmatrix}, \mathbf{input}_4 = \begin{bmatrix} 1 & 0 & 0 & 1 & -2 \\ 2 & 3 & 4 & 2 & -2 \\ 2 & 3 & 4 & 3 & -2 \\ 2 & 3 & 4 & 0 & -1 \end{bmatrix} \end{bmatrix} \tag{A-6}$$

The outputs would be

$$\mathbf{output} = \left\{ \begin{matrix} \mathbf{output}_1 = \begin{bmatrix} 1 & 1 & 3 & -2 \\ 1 & 2 & 3 & -2 \end{bmatrix}, \mathbf{output}_2 = \begin{bmatrix} 1 & 1 & 3 & -2 \\ 1 & 2 & 3 & -2 \end{bmatrix} \\ \mathbf{output}_3 = \begin{bmatrix} 1 & 1 & 3 & -2 \\ 1 & 2 & 3 & -2 \end{bmatrix}, \mathbf{output}_4 = \begin{bmatrix} 1 & 1 & 3 & -2 \\ 1 & 2 & 3 & -2 \end{bmatrix} \end{matrix} \right\} \tag{A-7}$$

(3) Step 3: DoF analysis

$$\mathbf{DoF} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{A-8}$$

(4) Step 4: full-cycle mobility determination
There is no "distal" joint existing in the limb. The mechanism has a full-cycle mobility.

c. ***2-PPPHR mechanism***

(1) Step 1: motion matrix could be

$$\mathbf{m} = \begin{bmatrix} \mathbf{m}_1 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & l_{14} & m_{14} & n_{14} & h \\ 1 & 0 & 0 & l_{15} & m_{15} & n_{15} & 0 \end{bmatrix} \\ \mathbf{m}_2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & l_{24} & m_{24} & n_{24} & h \\ 1 & 0 & 0 & l_{25} & m_{25} & n_{25} & 0 \end{bmatrix} \end{bmatrix} \tag{A-9}$$

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press

(2) Step 2: the inputs to the classification optimization would be

$$\mathbf{input} = \begin{bmatrix} \mathbf{input}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & -1 \\ 0 & 1 & 0 & 1 & -3 \\ 1 & 0 & 0 & 2 & -2 \end{bmatrix} \\ \mathbf{input}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & -1 \\ 0 & 1 & 0 & 1 & -3 \\ 1 & 0 & 0 & 2 & -2 \end{bmatrix} \end{bmatrix} \tag{A-10}$$

The outputs would be

$$\mathbf{output} = \begin{Bmatrix} \mathbf{output}_1 = \begin{bmatrix} 2 & 0 & 0 & -1 \end{bmatrix} \\ \mathbf{output}_2 = \begin{bmatrix} 2 & 0 & 0 & -1 \end{bmatrix} \end{Bmatrix} \tag{A-11}$$

(3) Step 3: DoF analysis

$$\mathbf{DoF} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{A-12}$$

(4) Step 4: full-cycle mobility determination
There is no "distal" joint existing in the limb. The mechanism has a full-cycle mobility.

**Cite this article:** X. Huo, Z. Song and T. Sun (2024). "A machine learning-based approach for automatic motion/constraint and mobility analysis of parallel robots", Robotica **42**, 2403–2429. https://doi.org/10.1017/S0263574724000808

https://doi.org/10.1017/S0263574724000808 Published online by Cambridge University Press