ASME Journal of Mechanisms and Robotics
Online journal at:
https://asmedigitalcollection.asme.org/mechanismsrobotics

Point-Based Models: A Unified Approach for Geometric Constraint Analysis of Planar *n*-Bar Mechanisms With Rotary, Sliding, and Rolling Joints

Zhijie Lyu
Computer-Aided Design and Innovation Lab,
Department of Mechanical Engineering,
Stony Brook University,
Stony Brook, NY 11794-2300
e-mail: zhijie.lyu@stonybrook.edu

Anurag Purwar[1]
Computer-Aided Design and Innovation Lab,
Department of Mechanical Engineering,
Stony Brook University,
Stony Brook, NY 11794-2300
e-mail: anurag.purwar@stonybrook.edu

Kinematic simulation of planar n-bar mechanisms has been an intense topic of study for several decades now. However, a large majority of efforts have focused on position analysis of such mechanisms with limited links and joint types. This article presents a novel, unified approach to the analysis of geometric constraints of planar n-bar mechanisms with revolute joint (R-joint), prismatic joint (P-joint), and rolling joint. This work is motivated by a need to create and program a system of constraint equations that deal with different types of joints in a unified way. A key feature of this work is that the rolling joint constraints are represented by four-point models, which enables us to use the well-established undirected graph rigidity analysis algorithms. As a result, mechanisms with an arbitrary combination of revolute-, prismatic joints, and wheel/gear/wheel-belt chains without any limitations on their actuation scheme can be analyzed and simulated efficiently for potential implementation in interactive computer software and large-scale data generation. [DOI: 10.1115/1.4066963]

Keywords: computational geometry, kinematic simulation, mobility analysis, planar mechanisms, mechanism synthesis and analysis, theoretical and computational kinematics

1 Introduction

Recently, the evolution of design methodologies, particularly in the realm of kinematic systems, has been significantly influenced by the advent and advancement of data-driven approaches [1–11]. These approaches, characterized by their reliance on extensive datasets, are poised to revolutionize the way we approach design and synthesis problems, offering new possibilities and efficiency, hitherto not possible. A pivotal aspect of these data-driven methods is the reliance on efficient simulation techniques. Simulations directly provide data samples, allowing for exploring and analyzing countless design variations without the constraints of physical prototyping [12]. Although there are several educational and commercially available computer-aided design (CAD) systems which have multibody kinematic and/or dynamic simulation capabilities, such as SAM [13], Mekin2D [14], MechGen [15], Linkages [16], Ch Mechanism Toolkit [17], MechDesigner [18], Linkage Mechanism Simulator [19], PMKS+ [20,21], GIM [22], Simionescu [23], Schmidt and Lax [24], and Universal Mechanism [25], it is difficult to mass-produce simulated data samples from these applications efficiently. Furthermore, most of these applications have a limited combination of kinematic elements, joints, and actuation schemes. For example, Mekin2D uses the Assur Group libraries [26,27], which restrict users from combining common Assur Groups to form kinematic structures. However, the number of Assur Groups is infinite, increasing almost exponentially concerning joint numbers [28,29]. Instead of relying on Assur Groups, researchers have developed algorithms based on the *geometric constraint graphs* [30–38]. These algorithms, rooted in graph theory, analyze the structure of the constraint graph to compute point positions efficiently. This article builds upon our previous research [39] on simulating planar multibar mechanisms with rotational and sliding joints by incorporating rolling components such as friction cylinders, wheels, gears, and wheel belts. It introduces a four-point model for rolling joint constraints, facilitating the use of established graph rigidity analysis algorithms. This enhancement enables the efficient simulation of mechanisms featuring any combination of rotational, sliding, and rolling elements, including wheel/gear/wheel-belt systems, without restrictions on their activation methods. While it is widely acknowledged that a wheel[2] rolling on a wheel without slipping can be modeled as angle–angle constraints, we demonstrate that this constraint can be represented using four geometric constraints on four points of every two cylinder pairs. Alongside distance constraints imposed on the rest of the kinematic elements of a linkage mechanism with wheels, we can analyze arbitrarily complex linkage mechanisms with wheels.

Efficiency in simulation entails addressing three main issues. First, it involves assessing the capabilities of the simulation device. For example, Nobari et al. [4] developed a dyadic decomposable planar linkage simulator programmed for systems with one or multiple graphics processing unit (GPUs), which claimed to be "800 times faster than the simple simulation algorithm." While this is advantageous for generating data samples for machine learning, not all devices possess GPUs, nor do all linkages have only simple loops that can be handled by the dyadic decomposition method. Second, efficiency relies on the computation method. According to Ait-Aoudia and Foufou [32], solving for the positions of n planar points simultaneously k times in a particular geometric constraint system typically incurs a time complexity of $O(kn^3)$ using various numerical approaches, such as homotopy [40], Newton's iteration [41], and geometric iterative [42]. The time cost becomes noticeable when k is a large value under the context of mass producing kinematically simulated samples. Other approaches like symbolic computation are typically exponential in time and space [32]. Finally, efficiency involves decomposing the geometric

[1]Corresponding author.

Contributed by the Mechanisms and Robotics Committee of ASME for publication in the Journal of Mechanisms and Robotics. Manuscript received September 7, 2024; final manuscript received October 9, 2024; published online January 20, 2025. Assoc. Editor: Yu She.

[2]In this article, we use the words cylinders, wheels, and gears interchangeably.

constraint system into a sequence of smaller subsystems. As a result, the decomposition lowers the value of the largest number of variables to solve and reduces the overall computation time. Some applications utilize the Assur Group library for decomposing the kinematic structure, while others employ graph-theory-based path-finding algorithms [4,30–34,36,38,39] to varying extents.

This article's primary contribution is the construction method of constraint graphs involving a combination of revolute, prismatic, and rolling joints. Specifically, wheels are treated as linkages in the graph, simplifying the mobility analysis procedure and extending the analysis algorithm for revolute-only joints to an arbitrary combination of the three types of joints and links. The remainder of this article is organized as follows: Sec. 2 reviews our existing simulation algorithm with only revolute and prismatic joints. Section 3 introduces wheels to the linkage system and constraint graph, accompanied by two examples. Section 5 outlines the data-saving structure, and Sec. 6 concludes the article.

2 Conversion and Constraint Graph

We will first review the linkage elements and how they are converted into a point-based kinematic structure using the reinterpretation trick presented in Ref. [39]. Second, we discuss the graph conversion and the *conceptual joints*. Finally, we examine the resultant structure's corresponding constraint graph, which we use to analyze the structure's mobility.

2.1 R-Joints. The R-joint permits relative rotation between two or more links and is represented as a planar point, which we denote using lower-case symbols. A binary link is defined to have two R-joints, which imposes a distance constraint. For example, $l_{i,j}$ denotes the binary link between joints i and j. A link with n number of R-joints might include several pairwise distance constraints. For instance, a ternary link can be considered to have three binary links, while in general, an n-ary link (a link with n joints) has ${}^nC_2$ binary links. These distance constraints do not change throughout the simulation process due to the assumption of rigidity.

2.2 Conversion of Rotary Actuator. A rotary actuator can only be placed at one joint that two binary rigid links share. It defines the relative angle between the two links. This side-angle-side configuration uniquely defines one triangle, where the third side's length can be computed when the angle is given. For example, Fig. 1(*a*) shows the two binary links, $l_{j,k}$ and $l_{j,i}$, and the rotary actuator $\alpha_{i,j,k}$ drawn as concentric rings. The square symbol attached to the rings indicates the actuator reference link, while the circular dot indicates the link that is being actuated. Thus, in this figure, the actuator is fixed to link $l_{j,i}$ while it is rotating the link $l_{j,k}$. We can compute the distance between joints i and k by

$$l_{i,k} = \sqrt{l_{j,k}^2 + l_{j,i}^2 - 2l_{j,k} \cdot l_{j,i} \cos(\alpha_{i,j,k})} \quad (1)$$

Clearly, $l_{i,k}$ varies as the link $l_{j,k}$ rotates and is uniquely determined when $\alpha_{i,j,k}$ is given. However, in a single state, we can treat this as a binary link of constant length. To differentiate this from the rigid links, also called *static* links in this article, we call this one a *dynamic* link. This link can be useful in finding the substructure's rigidity in the decomposition phase.

We note that adding this dynamic link results in the exact degree-of-freedom change that an actuator provides, it is not equivalent to the rotary actuator. For example, say we know the distances between the three joints through Eq. (1), and we know the locations of joints j and i. Then, the location of the joint k can be computed using arc intersection, where the first circle has j as its center and $l_{j,k}$ as the radius, and the second circle has i as its center and the dynamic link $l_{i,k}$ as its radius. This constraint relationship gives two possible solutions for the position of k. To resolve this, we will need to add an angle constraint. This can be resolved by using the cross-product equation, shown in Eq. (2). Only one of the two intersection solutions satisfies the following equation:

$$(x_i - x_j)(y_k - y_j) - (x_k - x_i)(y_i - y_j) - l_{i,j} \cdot l_{j,k} \sin(\alpha_{i,j,k}) = 0 \quad (2)$$

2.3 Conversion of Linear Actuators. A linear actuator $l_{p,q}$ shown in Fig. 1(*b*) appears in between a *slot joint* q and one of the end joints p. It provides the distance value between the two joints as an input and also serves as a dynamic link.

2.4 Conversion of Slots Using Approximation. The conversion of a slot uses the *far joint*, which is on the perpendicular bisector of the segment defined by joint p and r. As shown in Fig. 2, the far joint f connects with two end joints p and r, and the slot joint q through three different binary links with their length computed according to Ref. [39]. When we use the approximation trick, and the far joint is sufficiently far away from the center of the workspace, the curve traversed by joint q becomes flat, approximating the straight segment between the two end joints. Additionally, the numerical precision of the approximation result can be controlled by determining the distance of the far joint from the segment defined by joints p and r [39].

2.5 Representing Angle Constraint With Links. For the precise computation, the slot constraint can be represented by an angle constraint where the angle between vector $\mathbf{v}_{\mathbf{p,r}}$ and $\mathbf{v}_{\mathbf{p,q}}$ is always zero because the two vectors are parallel. Furthermore, while the approximation trick is unnecessary for the precise computation method, we can still use the three binary links to represent the slot constraint in the graph. This means that $l_{p,f}$, $l_{q,f}$, and $l_{r,f}$ lose their length property, and they become *conceptual links* in the graph. However, their combination corresponds to the angle constraint equation. Furthermore, we can consider the far joint to be the *conceptual prismatic joint* because no other joints better fit the role.

Notably, these conceptual objects do not necessarily hold common properties like those of links and joints. A conceptual binary link does not have the length, and a conceptual joint does

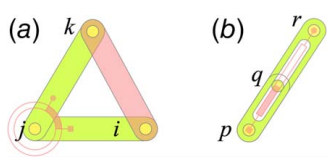

Fig. 1 The letters denote the joints. For the rotary actuator (*a*), $l_{i,k}$ is the corresponding *dynamic link*. For the linear actuator (*b*), $l_{p,q}$ is the corresponding dynamic link.

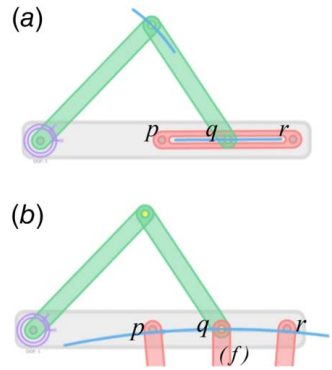

Fig. 2 The reinterpreted RRRP mechanism in (*a*) is shown in (*b*). The far joint *f* is in the far bottom.

not have the location. However, they do not change the degree of freedom in the graph, which is crucial for the mobility analysis.

2.6 Constraint Equations. In total, the types of equations in the system are reduced to two: the distance constraint defined by two points and the cross-product equation defined by three points. For a common binary link $l_{p,q}$, the distance constraint is written as shown in Eq. (3). The i in the equation means it is the ith distance constraint. When this constraint is satisfied, $f_i = 0$.

$$f_i = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2} - l_{p,q} \tag{3}$$

Next, the jth angle constraint function between vector $\mathbf{v_{q,p}}$ and $\mathbf{v_{q,r}}$ can be written as shown in Eq. (4). Here, $\alpha_{p,q,r}$ defines the angle that comes from either a slot or the rotary actuator. Similarly, when this angle constraint is satisfied, $g_j = 0$.

$$g_j = (x_p - x_q)(y_r - y_q) - (x_r - x_q)(y_p - y_q) - l_{p,q} \cdot l_{q,r} \cdot \sin(\alpha_{p,q,r}) \tag{4}$$

Solving the system of distance and angle constraint equations is the core of the kinematic simulation of a linkage mechanism with revolute and prismatic joints. Furthermore, we can concisely write the two types of equations as

$$\begin{cases} f = f(\mathbf{p}_i, \mathbf{p}_j) \\ g = g(\mathbf{p}_p, \mathbf{p}_q, \mathbf{p}_r) \end{cases} \tag{5}$$

These two types of equations take in two points and three points respectively, hence we can call them the *two-point model* and the *three-point model*.

2.7 Geometric Constraint Graph. A geometric constraint graph is used to determine the structural rigidity of its corresponding kinematic structure and substructure. For the constraint graph, the fundamental geometric constraint is the Euclidean distance constraint, while the angle constraint for the rotary actuator and the slot can be considered two special rules. In Fig. 1(*a*), the angle constraint does not change the mobility if we already have a corresponding distance constraint, i.e., the dynamic link, in the system. Similarly, the three-link representation of the slot constraint shown in Fig. 2(*b*) does not change the mobility because joints p, r, and f form a rigid triangle, and the only movable joint q is "rotating" with respect to this triangle, which is only one degree of freedom.

If we extract the unweighted graph, i.e., we ignore the actual distance values from the weighted constraint graph we constructed, the mobility of this graph is not changed in the *generic* sense [28,30–32]. Luckily, this kind of undirected graph's structural rigidity is well-studied, and many researchers have proposed different mobility analysis algorithms [30–38]. If we introduce the angle–angle constraint relationship found in wheels rolling on wheels with links as the interpretation, just like what we did for the conversion of slots, then we can use these well-established algorithms to subdivide the kinematic structure. This reduces the number of variables to solve simultaneously as much as possible. As a result, a solving path can be found, which is generally faster than solving unknowns all at once.

3 Representation of Angle–Angle Constraint

It is worth noting that modern CAD software handles geometric constraint systems with the following types of constraints: distance, angle, incidence, tangency, parallelism, and perpendicular between geometric elements [32]. However, the *angle–angle* constraint has largely remained undiscussed, or more precisely, not merged into graph-theory-based geometric constraint solvers.

We may see the angle constraint relationship in a certain kinematic system with a gear train. For example, Eq. (6) is the kinematic constraint relationship we want to introduce:

$$\int_0^{\theta_i} w_i(\theta_i)\, \mathrm{d}\theta_i \pm \int_0^{\theta_j} w_j(\theta_j)\, \mathrm{d}\theta_j = 0, \tag{6}$$

where θ_i and θ_j correspond to two rigid bodies' orientations, respectively, while w_i and w_j are the radius of the curvature of the two contacting objects, the $\pm$ sign is determined by whether the two wheels interact internally or externally. For circular wheels, w_i and w_j are the radii of two rotating rigid bodies. Therefore, the equation can be simplified into the following linear equation:

$$r_i\theta_i \pm r_j\theta_j = 0. \tag{7}$$

While an angle can be represented with three points, the above-mentioned angle–angle constraint can be represented with four points (c_1, c_2, p_2, p_3), as shown in Fig. 3. Angles α_1 and α_2 are the relative angles defined by (p_1, c_1, p_2) and (p_1, c_2, p_3), respectively.

This section provides a system of equations set up from a point-based perspective. We will first go through the *four-point model* equation to describe the linear in-between angle relationship in Sec. 3.1, then we will discuss the constraint geometry so that we can find a solving path in Sec. 3.3. Finally, we provide two examples with gear trains and linkages in Sec. 4.

However, before going any further, the following expressions are useful for two-dimensional data points $\mathbf{p}_1 = (x_1, y_1)$ and $\mathbf{p}_2 = (x_2, y_2)$:

$$\begin{cases} \det(\mathbf{p}_1, \mathbf{p}_2) & = x_1y_2 - x_2y_1 \\ \mathrm{dot}(\mathbf{p}_1, \mathbf{p}_2) & = x_1x_2 + y_1y_2 \\ \mathrm{dist}(\mathbf{p}_1, \mathbf{p}_2) & = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \end{cases} \tag{8}$$

3.1 Four-Point Model Equations. This section presents the equations from the four-point model as shown in Fig. 3. These four points are mathematically represented by $\mathbf{c}_1$, $\mathbf{c}_2$, $\mathbf{p}_2$, and $\mathbf{p}_3$. We use three rigid bodies to represent the wheel–wheel relationship because (1) the two wheels' centers do not change their distances, meaning there is a distance constraint inside this constraint relationship, and (2) the two wheels' rotational behaviors can be measured with respect to the two centers. Particularly, the rotated angle of one wheel with respect to the segment defined by the two centers is proportional to the rotated angle of another wheel with respect to the same segment. This property is crucial when we need to derive a wheel train with multiple moving wheels. Furthermore, we note that all four points are assumed to be freely movable in space, and their mobility depends on the overall kinematic structure.

In Fig. 3, $\mathbf{c}_1$ and $\mathbf{c}_2$ represent the coordinates of the center of the two wheels. Their relative distance will not change throughout their interaction, and we can consider there is one binary link between the two nodes/joints. The point $\mathbf{p}_1$ is where the two wheels contact each other. The point $\mathbf{p}_2$ is the point that previously met $\mathbf{p}_3$ at $\mathbf{p}_1$, whose distance from $\mathbf{c}_1$ is r_1. Similarly, the distance between $\mathbf{p}_3$ and $\mathbf{c}_2$

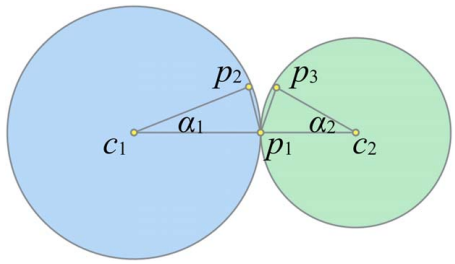

Fig. 3 This wheel pair model has three rigid bodies: the two wheels and a link connecting the two centers

is r_2. If there is no sliding between the two wheels, and if we fix the reference frame to the binary link connecting the centers of the wheels, then the two wheels' rotations have a constraint relationship given by $\alpha_1 \cdot r_1 = \alpha_2 \cdot r_2$, where $\alpha_1 = \angle<\mathbf{p}_1, \mathbf{c}_1, \mathbf{p}_2>$ and $\alpha_2 = \angle<\mathbf{p}_1, \mathbf{c}_2, \mathbf{p}_3>$. We note that the latter two equations are informative here because (1) we will use points to represent angle-related values such as the sine and cosine values at the variable computing phase and (2) the three points are order sensitive, but the α notation omits this property. Furthermore, $\mathbf{p}_1$ is not considered an independent point because it is determined by the positions of $\mathbf{c}_1$ and $\mathbf{c}_2$, and the two wheel's radius. Its position is given by Eq. (9)

$$\mathbf{p}_1 = \frac{r_2}{r_1 + r_2}\mathbf{c}_1 + \frac{r_1}{r_1 + r_2}\mathbf{c}_2 \tag{9}$$

This way, we have four points of interest in three rigid bodies: the binary link and two wheels. To analyze its degrees of freedom, we can first fix the binary link, which means we reduced three degrees of freedom in the system. Additionally, one wheel's rotational behavior will determine the other's rotation behavior. In other words, fixing the binary link leads to one degree of freedom remaining, i.e., the rotational behavior. Together with three degrees of freedom of the binary link, we see that the three-body system has four degrees of freedom. While four points correspond to eight variables, we need exactly four equations to reduce the degree of freedom to four.

First, we can derive the two distance constraints from the two wheels, respectively:

$$\begin{cases} \text{dist}(\mathbf{c}_1, \mathbf{p}_2) - r_1 = 0 \\ \text{dist}(\mathbf{c}_2, \mathbf{p}_3) - r_2 = 0 \end{cases} \tag{10}$$

These two are the distance constraint equations, or as we call them, f-type equations. For the two circle centers, two cases of the f-type equation are given in Eq. (11). The value of $l_{\mathbf{c}_1,\mathbf{c}_2}$ is $r_1 + r_2$ when they are not overlapping. When they overlap, this value is the absolute value of $(r_1 - r_2)$.

$$\text{dist}(\mathbf{c}_1, \mathbf{c}_2) - l_{\mathbf{c}_1,\mathbf{c}_2} = 0 \tag{11}$$

The two relative angle values are intermediate variables and should not appear in the final expressions. Therefore, we can use the cross-product and dot product of positional vectors to represent the constraints as follows:

$$\begin{cases} t_1 &= \det(\mathbf{p}_1 - \mathbf{c}_1, \mathbf{p}_2 - \mathbf{c}_1) = r_1^2 \sin(\alpha_1) \\ t_2 &= \text{dot}(\mathbf{p}_1 - \mathbf{c}_1, \mathbf{p}_2 - \mathbf{c}_1) = r_1^2 \cos(\alpha_1) \\ t_3 &= \det(\mathbf{p}_1 - \mathbf{c}_2, \mathbf{p}_3 - \mathbf{c}_2) = r_2^2 \sin(\alpha_2) \\ t_4 &= \text{dot}(\mathbf{p}_1 - \mathbf{c}_2, \mathbf{p}_3 - \mathbf{c}_2) = r_2^2 \cos(\alpha_2) \end{cases} \tag{12}$$

While the expressions for $t_1 \sim t_4$ are complicated, we successfully avoid computing the exact angle value. Additionally, the constraint between α_1 and α_2 can be expressed as

$$h_1 = \arctan\left(\frac{t_1}{t_2}\right) \cdot r_1 \pm \arctan\left(\frac{t_3}{t_4}\right) \cdot r_2 - l_s = 0 \tag{13}$$

where l_s is a function that accounts for the sliding between wheels. Its value is zero when the wheels roll on each other without slipping, a condition we will assume in this article.

In total, we have derived a total of four geometric constraint equations, i.e., three f-type equations and one h-type equation, for this wheel pair model. The total number of variables is eight from the four points, meaning that the model has four degrees of freedom. Last but not least, the $\pm$ sign in Eq. (13) is determined by how the two wheels interact. If the two wheels rotate in the same direction (overlapping case), then it is a minus sign; otherwise, it is a plus sign.

The four-point system can be used in a wheel-belt model as well. However, $\mathbf{p}_1$ no longer exists since the two wheels do not touch physically, as shown in Fig. 4. Instead, we can introduce two points $\mathbf{p}_4$ and $\mathbf{p}_5$, which share the same intermediate variable

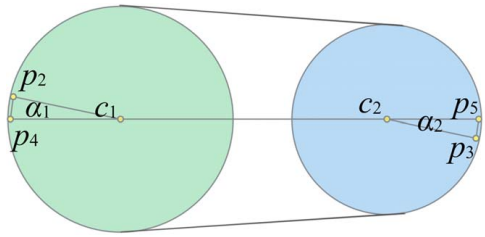

Fig. 4 Wheel-Belt System

property with $\mathbf{p}_1$ that the points lie on the axis defined by $\mathbf{c}_1$ and $\mathbf{c}_2$

$$\begin{cases} \mathbf{p}_4 = \mathbf{c}_2 + (\mathbf{c}_1 - \mathbf{c}_2) + (\mathbf{c}_1 - \mathbf{c}_2) \cdot \dfrac{r_1}{r_1 + r_2} = \mathbf{c}_1 \cdot \dfrac{2r_1 + r_2}{r_1 + r_2} + \mathbf{c}_2 \cdot \dfrac{-r_1}{r_1 + r_2} \\ \mathbf{p}_5 = \mathbf{c}_1 + (\mathbf{c}_2 - \mathbf{c}_1) + (\mathbf{c}_2 - \mathbf{c}_1) \cdot \dfrac{r_2}{r_1 + r_2} = \mathbf{c}_2 \cdot \dfrac{r_1 + 2r_2}{r_1 + r_2} + \mathbf{c}_1 \cdot \dfrac{-r_2}{r_1 + r_2} \end{cases} \tag{14}$$

Specifically, $\mathbf{p}_4$ and $\mathbf{p}_5$ are the points on two wheels as shown in Fig. 4. The t equations will need to be rewritten as

$$\begin{cases} t_1 &= \det(\mathbf{p}_4 - \mathbf{c}_1, \mathbf{p}_2 - \mathbf{c}_1) = r_1^2 \sin\alpha_1 \\ t_2 &= \text{dot}(\mathbf{p}_4 - \mathbf{c}_1, \mathbf{p}_2 - \mathbf{c}_1) = r_1^2 \cos\alpha_1 \\ t_3 &= \det(\mathbf{p}_5 - \mathbf{c}_2, \mathbf{p}_3 - \mathbf{c}_2) = r_2^2 \sin\alpha_2 \\ t_4 &= \text{dot}(\mathbf{p}_5 - \mathbf{c}_2, \mathbf{p}_3 - \mathbf{c}_2) = r_2^2 \cos\alpha_2 \end{cases} \tag{15}$$

In this wheel-belt system, $\alpha_1 = \angle<\mathbf{p}_4, \mathbf{c}_1, \mathbf{p}_2>$ and $\alpha_2 = \angle<\mathbf{p}_5, \mathbf{c}_2, \mathbf{p}_3>$. In addition to the use of $\mathbf{p}_4$ and $\mathbf{p}_5$, the distance constraint between the two wheel centers is more general, which is simply $l_{\mathbf{c}_1,\mathbf{c}_2}$ and it should be predefined before the positional analysis phase.

As a result, the wheel-belt system also provides four equations to the system: two distance constraints for wheel radius, one distance constraint for the two wheels' centers, and one relative angle constraint shown in Eq. (13). The $\pm$ sign is minus in this case because two wheels rotate in the same direction. Notably, we can simply represent the h-type equation with Eq. (16), because $\mathbf{p}_1$, $\mathbf{p}_4$, and $\mathbf{p}_5$ can be considered an intermediate set of variables. Apparently, the h-type equation is the four-point model.

$$h = h(\mathbf{c}_1, \mathbf{c}_2, \mathbf{p}_2, \mathbf{p}_3) \tag{16}$$

3.2 System of Equations. As a result of introducing the angle–angle relationship, the system of equations adds one more type of equation, the h-type equation. Therefore, the positional analysis problem is formulated as: given a specific state $\Phi(t)$, find the stack of variables $\mathbf{x}$ that satisfies $\mathbf{f} = \mathbf{0}$. All the states are solved one by one. Vectors $\mathbf{x}$ and $\mathbf{f}$ are shown in Eq. (17)

$$\mathbf{x} = \begin{bmatrix} x_p \\ y_p \\ x_q \\ y_q \\ \vdots \\ x_r \\ y_r \end{bmatrix} \quad \mathbf{f}(\mathbf{x}, \Phi(t)) = \begin{bmatrix} f_1 \\ \vdots \\ g_1 \\ \vdots \\ h_1 \\ \vdots \end{bmatrix} = \mathbf{0} \tag{17}$$

3.3 Four-Point Model in the Constraint Graph. To derive the constraint graph for the four-point model, we first review the P-joint conversion shown in Fig. 2. While we have a conceptual prismatic joint in the constraint graph, this joint does not necessarily need the location property if we consider the combination of the three binary links and the far joint to represent an angle constraint.

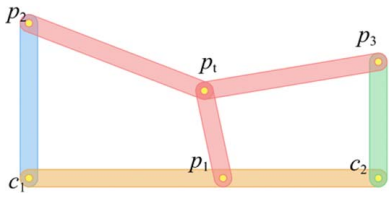

Fig. 5 This is the geometric constraint graph for both Figs. 3 and 4. The three conceptual links ($l_{1,t}$, $l_{2,t}$, and $l_{3,t}$) are the added structure to mimic the rolling behavior, and they directly correspond to the angle–angle constraint equation. For the wheel-belt system, we can consider p_1 to be in some position on the connecting binary link between c_1 and c_2, but it is not used in numerical computation.

Following this idea, we can create the geometric constraint model for the wheels. In Fig. 5, a new point p_t is added and joined with p_2 and p_3 to represent the constraint relationship. Besides that, it does not necessarily correspond to any (x, y) coordinates. The point p_t contributes the following properties to the constraint graph:

(1) There is only one degree of freedom for p_2 and p_3. Specifically, p_2 will change its position if p_3 changes its position, and vice versa. In other words, the mapping between p_2 and p_3 locations is bijective.
(2) A link with length property is a distance constraint. A link without length property combines with two other particular links and a particular joint to form either an angle constraint or an angle–angle constraint.

Similar to the conceptual prismatic joint, we can call p_t the *conceptual rolling joint*, and the combination of the three added links corresponds to one *h*-type equation, just like the three added links for the slot constraint in Fig. 2 correspond to one *g*-type equation. We will show the usage of p_t with examples in the next section.

4 Examples

This section delineates the examination of two distinct systems incorporating wheel components, as illustrated in Fig. 6. An in-depth analysis will demonstrate the decomposition process of these systems. Both systems are characterized by an identical assembly of static kinematic elements, configured similarly and driven by dual actuators, reflective of their 2-degrees-of-freedom nature. The primary distinction between the systems resides in the placement of the left actuator, while the right actuator is placed between a moving wheel and a floating link (w_2 and l_3); in system 1, the left actuator is situated between the ground link and the connecting link (l_1 and l_2), whereas in the other system, the left actuator is positioned between the wheel and the connecting link (w_1 and l_2). Despite similar locations, the actuators introduce angle constraints between different pairs of rigid bodies.

The initial step involves enumerating the geometric constraint equations. Subsequently, a constraint graph is constructed, the analysis of which determines the closed subsystems formed by these constraints. Through solving these subsystems, the overall solution to the original system is derived, as depicted in Fig. 6 through their respective constraint graphs.

The procedure for solving these two systems diverges significantly owing to the disparate actuator placements, as illustrated in Figs. 7 and 8. Rigidity analysis can be conducted via multiple approaches, with the application of Grubler's formula [43,44] or Laman's theorem [32,45] recommended assessing the rigidity at each step of the solution process, as shown in the aforementioned figures.

For the first system, a particular point is solved in one step. Comparatively, the initial step in the second system entails determining

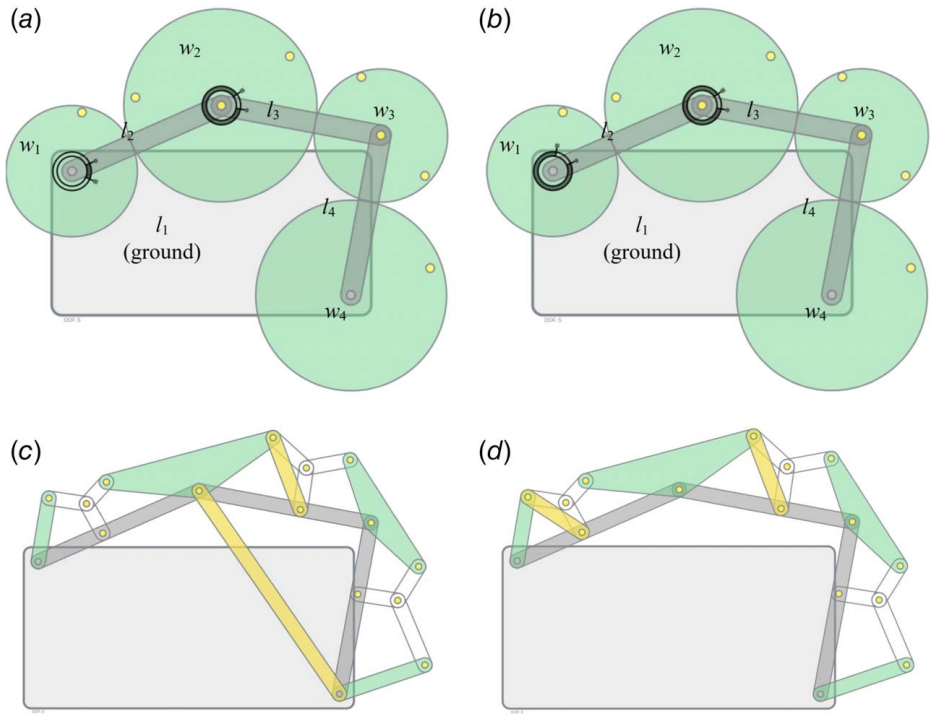

Fig. 6 The two kinematic systems' constraints shown in (*a*) and (*b*) are derived in (*c*) and (*d*), respectively, according to the aforementioned conversions. Particularly, the intersection point of three transparent conceptual links represents the angle–angle constraint between the corresponding wheel pair. The two wheels in the middle have two angle–angle constraints, respectively, so these wheels are represented with triads. The dynamic link that actuators bring to the systems is also placed in the constraint graph during the mobility analysis.

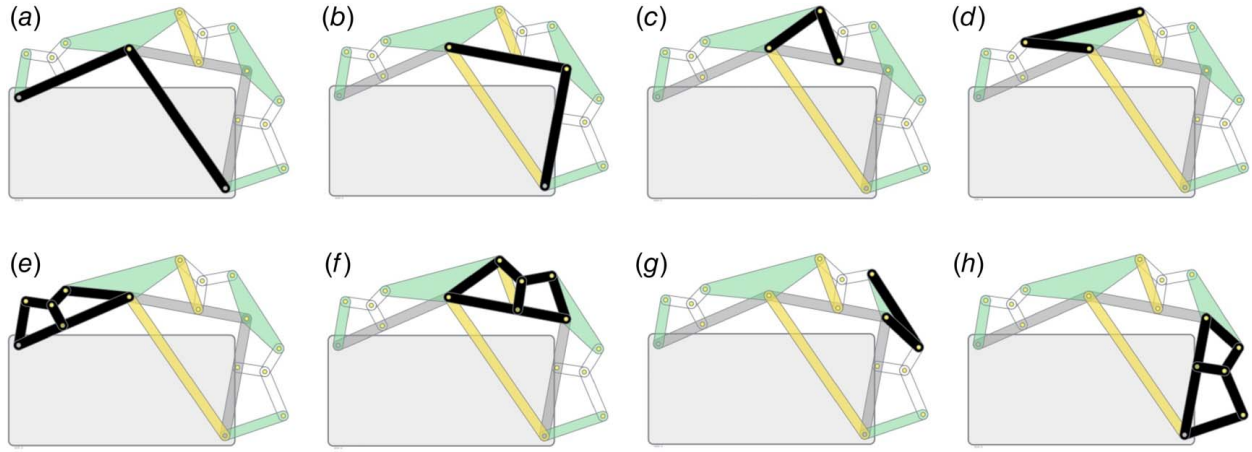

Fig. 7 The solving path for the system shown in Fig. 6(c) is displayed here. This system can be solved with eight steps. The constraint equations in each step correspond to the black-colored opaque links. Furthermore, while each binary link corresponds to one distance constraint, the angle–angle constraint links are represented with transparent links. When these angle–angle constraint links are black colored, the angle–angle constraint equation is inside the system.

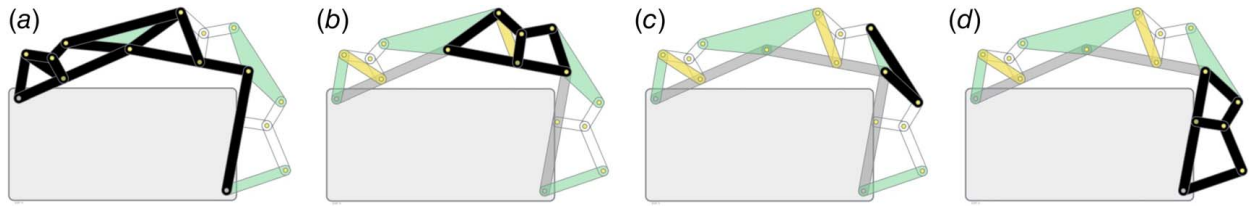

Fig. 8 The solving path for the system shown in Fig. 6(*d*) is displayed here. This system can be solved with four steps. The constraint equations in each step correspond to the black-colored opaque links. Furthermore, unlike the first system, the first subsystem uses more than 10 constraint equations.

the positions of five points, with the subsequent three points resolved individually. Furthermore, we can see that the solving path follows the convention that

(1) Even if the points on the wheel are used in some steps such as Fig. 7(*c*), the angle–angle constraint equation is not used.
(2) The angle–angle constraint equation is in the subsystem exactly when the point without physical property, i.e., p_t, is used in the constraint graph.

We should note that this four-point model only applies when each relative angle change is small. The locations of p_2 and p_3 should be refreshed for each input. At each refresh, the locations of p_2 and p_3 are at exactly p_1 for the wheel–wheel system, and p_2 is at p_4 while p_3 is at p_5 for the wheel-belt system. In summary, the computation process is

(1) Convert all slots and wheels to create the geometric constraint graph.
(2) Find the solving procedure for this graph using the mobility analysis algorithm.
(3) Determine the input value and refresh the location of p_2 and p_3 on each wheel pair.
(4) Refresh all the corresponding distance constraints according to the new location of p_2 and p_3.
(5) Solve the point locations according to the solving procedure.

Remark. This methodology seamlessly incorporates wheels into the linkage system. It permits flexible placement of actuators within the wheel train, unrestricted movement of wheels, and arbitrary positioning of R-joints on the wheels. It also accommodates diverse combinations of R-joint and P-joint linkages alongside wheels/gears within the system design.

The additional cost of using these point models arises from using a pair of coordinates (x, y) instead of an angle for a specific wheel. In other words, we use one more variable to represent the same system. However, these models enable the expansion of the constraint graph method, thereby facilitating the direct application of established properties and theorems pertinent to generic point-based structural rigidity analysis, a field that has been extensively studied [30–38]. Consequently, point-based models offer simplicity, high modularity, and considerable creative flexibility.

5 Data Storage

Data storage is critical to large-scale simulations for machine learning applications. We want the data structure to be easily accessible and extensible, while at the same time, it should not take up a lot of storage space.

We have demonstrated using the *LJ*-matrix (or, the *B* matrix [46]), the Euclidean Distance Matrix [29] (or the *D* matrix), Slot matrix *S*, and the actuator matrix *I* to describe the kinematic structure of a mechanism with P-joints and R-joints in our previous work [39]. While the introduction of wheels results in the inclusion of two types of equations, the wheel–wheel constraint relationship only needs one additional matrix. We can denote this as follows:

$$W = \begin{bmatrix} c_{00} & c_{01} & r_{00} & r_{01} & l_{01} & l_{02} & l_{03} & W_o \\ c_{10} & c_{11} & r_{10} & r_{11} & l_{11} & l_{12} & l_{13} & W_i \\ c_{20} & c_{21} & r_{20} & r_{21} & l_{21} & l_{22} & l_{23} & B \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{wx5} \tag{18}$$

Here, c is the point index for the center point and r is used to denote the radius of the circles. Each wheel pair and its connecting link can be represented with three links, so the three ls for each row correspond to the indices of the three rigid bodies. In the last column, the values specify what type of four-point model to use: the wheel can interact with each other externally (W_o) or internally (W_i), or their rotation is related through a certain belt (B). The radius is saved instead of points due to the fact that we need to reinitiate the

calculation of the circular edge points every time we launch the positional analysis procedure. This is because we do not keep track of α_1 and α_2 in Eq. (12), which are relative angles with respect to the line segment defined by the two wheels' centers.

As for the simulation input and output, they follow the same convention as in Ref. [39] that the input is stored in a matrix, and the output is stored in a tensor, as shown in Eqs. (19) and (20), respectively.

$$\Phi = \begin{pmatrix} s_0 & s_1 & \cdots & s_{s-1} \\ \alpha_0(t_0) & \alpha_0(t_1) & \cdots & \alpha_0(t_{s-1}) \\ \alpha_1(t_0) & \alpha_1(t_1) & \cdots & \alpha_1(t_{s-1}) \\ \alpha_2(t_0) & \alpha_2(t_1) & \cdots & \alpha_2(t_{s-1}) \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{a-1}(t_0) & \alpha_{a-1}(t_1) & \cdots & \alpha_{a-1}(t_{s-1}) \end{pmatrix}_{a\times s} \tag{19}$$

$$P = [P_0, P_1, P_2, \ldots, P_{s-1}]_{s\times n\times 2} \tag{20}$$

Here, each slice of P corresponds to one specific input state. For example, for the kth state, the input and the joint positions are saved in Φ_{k-1} and P_{k-1}, respectively, as shown in Eq. (21).

$$\Phi_{k-1} = \begin{pmatrix} s_{k-1} \\ \alpha_0(t_{k-1}) \\ \alpha_1(t_{k-1}) \\ \alpha_2(t_{k-1}) \\ \vdots \\ \alpha_{a-1}(t_{k-1}) \end{pmatrix}_{a\times 1} \quad P_{k-1} = \begin{matrix} \\ p_0 \\ p_1 \\ p_2 \\ \vdots \\ p_{n-1} \end{matrix} \begin{matrix} x & y \end{matrix} \begin{pmatrix} x_0 & y_0 \\ x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \end{pmatrix}_{n\times 2} \tag{21}$$

6 Conclusion

In this article, we reviewed the constraint graph for the linkage system; then, we introduced the rolling joint to the constraint graph, where we added a virtual point to the four-point model. Two examples are provided to show the two system's decomposability. The data storage for the mechanisms is also briefly discussed. As a result, we provide a streamlined constraint graph construction method that captures the necessary information and assists us in finding the rigid subsystems within the kinematic systems that incorporate wheel trains and linkages. Such a system can be analyzed by many well-established path-finding algorithms, and the construction method brings extensive freedom in creating the combination of R-joint, P-joint, and rolling joints.

Acknowledgment

This paper was presented at the 2024 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC-CIE2024).

Conflict of Interest

This publication has research support from a National Science Foundation STTR phase II award (No. 2126882) to Co-PI Purwar, who also holds stocks in Mechanismic Inc. The research findings included in this publication may or may not necessarily relate to the interests of Mechanismic Inc. The terms of this arrangement have been reviewed and approved by Stony Brook University in accordance with its policy on objectivity in research.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

References

[1] Deshpande, S., and Purwar, A., 2019, "A Machine Learning Approach to Kinematic Synthesis of Defect-Free Planar Four-Bar Linkages," ASME J. Comput. Inf. Sci. Eng., **19**(2), p. 021004.
[2] Deshpande, S., and Purwar, A., 2019, "Computational Creativity via Assisted Variational Synthesis of Mechanisms Using Deep Generative Models," ASME J. Mech. Des., **141**(12), p. 121402.
[3] Deshpande, S., and Purwar, A., 2020, "An Image-Based Approach to Variational Path Synthesis of Linkages," ASME J. Comput. Inf. Sci. Eng., **21**(2), p. 021005.
[4] Nobari, A. H., Srivastava, A., Gutfreund, D., and Ahmed, F., 2022, "LINKS: A Dataset of a Hundred Million Planar Linkage Mechanisms for Data-Driven Kinematic Design," International Design Engineering Technical Conference: Volume 3A: 48th Design Automation Conference (DAC), St. Louis, MO, Aug. 14–17, American Society of Mechanical Engineers.
[5] Regenwetter, L., Nobari, A. H., and Ahmed, F., 2022, "Deep Generative Models in Engineering Design: A Review," J. Mech. Des, **144**(7), p. 071704.
[6] Yu, S.-C., Chang, Y., and Lee, J.-J., 2022, "A Generative Model for Path Synthesis of Four-Bar Linkages via Uniform Sampling Dataset," Proc. Inst. Mech. Eng., Part C: J. Mech. Eng. Sci., **237**(4), pp. 811–829.
[7] Vermeer, K., Kuppens, R., and Herder, J., 2018, "Kinematic Synthesis Using Reinforcement Learning," International Design Engineering Technical Conference: Volume 2A: 44th Design Automation Conference, Quebec City, Quebec, Canada, Aug. 26–29, American Society of Mechanical Engineers.
[8] Sharma, S., and Purwar, A., 2022, "A Machine Learning Approach to Solve the Alt-Burmester Problem for Synthesis of Defect-Free Spatial Mechanisms," J. Comput. Inf. Sci. Eng., **22**(2), p. 021003.
[9] Khan, N., Ullah, I., and Al-Grafi, M., 2015, "Dimensional Synthesis of Mechanical Linkages Using Artificial Neural Networks and Fourier Descriptors," Mech. Sci., **6**(1), pp. 29–34.
[10] Nobari, A. H., Srivastava, A., Gutfreund, D., Xu, K., and Ahmed, F., 2024, "Link: Learning Joint Representations of Design and Performance Spaces Through Contrastive Learning for Mechanism Synthesis." https://arxiv.org/abs/2405.20592.
[11] Lyu, Z., and Purwar, A., 2024, "Deep Learning Conceptual Design of Sit-to-Stand Parallel Motion Six-Bar Mechanisms," J. Mech. Des., **147**(1), p. 013306.
[12] Picard, C., Schiffmann, J., and Ahmed, F., 2023, "Dated: Guidelines for Creating Synthetic Datasets for Engineering Design Applications," International Design Engineering Technical Conference; Volume 3A: 49th Design Automation Conference (DAC), Boston, MA, Aug. 20–23.
[13] Artas Engineering, "SAM (Synthesis and Analysis of Mechanisms)," http://www.artas.nl/en, Accessed October 25, 2024.
[14] Simionescu, P. A., 2016, "MeKin2D: Suite for Planar Mechanism Kinematics," ASEE Virtual Annual Conference: IDETC-CIE, Volume 5B: 40th Mechanisms and Robotics Conference, Charlotte, NC, August, p. V05BT07A083.
[15] "MechGen," http://mechanicaldesign101.com/mechanism-generator-2-0/#MechGen3, Accessed October 25, 2024.
[16] Norton Associates Engineering, "Linkages," http://www.designofmachinery.com/Linkage/index.html, Accessed October 25, 2024.
[17] SoftIntegration, "Ch Mechanism Toolkit," http://www.softintegration.com/products/toolkit/mechanism/, Accessed October 25, 2024.
[18] PS Motion, "MechDesigner," http://www.psmotion.com/, Accessed October 25, 2024.
[19] Rector, D., "Linkage," http://blog.rectorsquid.com/linkage-mechanism-designer-and-simulator/, Accessed October 25, 2024.
[20] Campbell, M., "Planar Mechanism Kinematic Simulator," http://design.engr.oregonstate.edu/pmksintro.html, Accessed October 25, 2024.
[21] Dowd, T. D., Zhang, H., and Dutile, R. A., "PMKS+," https://pmksplus.mech.website/. Accessed October 25, 2024.
[22] Petuya, V., Macho, E., Altuzarra, O., and Pinto, C., 2011, "Educational Software Tools for the Kinematic Analysis of Mechanisms," Comp. Appl. Eng. Edu., **6**(4), pp. 261–266.
[23] Simionescu, P., 2004, "Enhancing Programming Skills to Engineering Students Using MeKin2D Modular Kinematics Subroutines." ASEE Virtual Annual Conference.
[24] Schmidt, P., and Lax, P., 2018, "Use of Computer Coding to Teach Design in a Mechanics Course, Resulting in an Implementation of a Kinematic Mechanism Design Tool Using PYTHON," ASEE Annual Conference & Exposition.
[25] Laboratory of Computational Mechanics, Bryansk State Technical University, "Universal Mechanism."
[26] Galletti, C., and Giannotti, E., 2009, "Assur's-Groups-Based Simulation for Teaching Kinematics of Planar Linkages," Ancona, Italy, https://api.semanticscholar.org/CorpusID:195258905.
[27] Galletti, C. U., 1979, "On the Position Analysis of Assur's Groups of High Class," Meccanica, **14**(1), pp. 6–10.
[28] Glue, C. M., 2022, "Generating All Rigidity Circuits on At Most 10 Vertices and All Assur Graphs on At Most 11 Vertices," J. Integer Sequ., **25**(2), p. 3.
[29] Rojas, N., and Thomas, F., 2012, "Distance-Based Formulations for the Position Analysis of Kinematic Chains," Ph.D. thesis, Universidad Politécnica de Cataluña, Barcelona, Spain.
[30] Fudos, I., and Hoffmann, C., 2004, "A Graph-Constructive Approach to Solving Systems of Geometric Constraints," ACM Trans. Graph., **16**(2), pp. 179–216.
[31] Bouma, W., Fudos, I., Hoffmann, C., Cai, J., and Paige, R., 1995, "Geometric Constraint Solver," Comput.-Aid. Des., **27**(6), pp. 487–501.
[32] Ait-Aoudia, S., and Foufou, S., 2010, "A 2D Geometric Constraint Solver Using a Graph Reduction Method," Adv. Eng. Softw., **41**(10–11), pp. 1187–1194.
[33] Jacobs, D. J., and Thorpe, M. F., 1995, "Generic Rigidity Percolation: The Pebble Game," Phys. Rev. Lett., **75**(22), pp. 4051–4054.

[34] Sljoka, A., Shai, O., and Whiteley, W., 2011, "Checking Mobility and Decomposition of Linkages Via Pebble Game Algorithm," International Design Engineering Technical Conference: Volume 6: 35th Mechanisms and Robotics Conference, Parts A and B Washington, DC, Aug. 28–31.
[35] Nordstrom, J., 2013, "Pebble Games, Proof Complexity, and Time-Space Trade-Offs," Logical Methods Comput. Sci., **9**(3).
[36] Shai, O., and Müller, A., 2013, "A Novel Combinatorial Algorithm for Determining the Generic/topological Mobility of Planar and Spherical Mechanisms," International Design Engineering Technical Conference: Volume 6B: 37th Mechanisms and Robotics Conference, Portland, OR, Aug. 4–7.
[37] Sidman, J., John, A. S., and Braun, B., 2017, "The Rigidity of Frameworks: Theory and Applications," https://dx.doi.org/10.1090/noti1575.
[38] Müller, A., and Shai, O., 2017, "Constraint Graphs for Combinatorial Mobility Determination," Mech. Mach. Theory., **108**, pp. 260–275.
[39] Lyu, Z., Purwar, A., and Liao, W., 2023, "A Unified Real-Time Motion Generation Algorithm for Approximate Position Analysis of Planar N-Bar Mechanisms," ASME J. Mech. Des., **146**(6), p. 063302.
[40] Bates, D. J., Hauenstein, J. D., Sommese, A. J., and Wampler, C. W., "Bertini: Software For Numerical Algebraic Geometry," bertini.nd.edu with permanent dx.doi.org/10.7274/R0H41PB5.
[41] Moré, J. J., 1978, "The Levenberg-Marquardt Algorithm: Implementation and Theory," *Numerical Analysis*, G. A. Watson, ed., Springer, Berlin/Heidelberg, pp. 105–116.
[42] Hernández, A., and Petuya, V., 2004, "Position Analysis of Planar Mechanisms With R-Pairs Using a Geometrical-Iterative Method," Mech. Mach. Theory., **39**(2), pp. 133–152.
[43] McCarthy, J. M., and Soh, G. S., 2010, *Geometric Design of Linkages*, Vol. 11, Springer, New York.
[44] Gogu, G., 2005, "Chebychev–Grübler–Kutzbach's Criterion for Mobility Calculation of Multi-Loop Mechanisms Revisited via Theory of Linear Transformations," Eur. J. Mech. A Solids, **24**(3), pp. 427–441.
[45] Laman, G., 1970, "On Graphs and Rigidity of Plane Skeletal Structures," J. Eng. Math., **4**, pp. 331–340.
[46] Tsai, L., 2001, *Mechanism Design: Enumeration of Kinematic Structures According to Function*, CRC Press LLC, Boca Raton, FL.