Multibody System Dynamics (2023) 59:337–362
https://doi.org/10.1007/s11044-023-09915-7

High-order inverse dynamics of serial robots based on projective geometric algebra

Guangzhen Sun[1] · **Ye Ding**[1]

Received: 29 June 2022 / Accepted: 27 April 2023 / Published online: 18 May 2023

Abstract
The efficient algorithm for high-order inverse dynamics of serial robots is an essential need in design and model-based control of robots equipped with serial elastic joints. Although several efficient algorithms have been proposed, the introduction of new frameworks can lead to new understanding and further improvements. Based on the projective geometric algebra (PGA) for Euclidean geometry, this paper provides a recursive algorithm that is computationally efficient, intuitive, uniform, and coordinate invariant. In the PGA-based method, calculations of the exponential map and Lie brackets are simplified and rigid body motions are represented as vectors instead of matrices. All geometric elements used to model robots are represented as vectors uniformly, and all operations are modeled as algebraic operations with explicit geometric meaning. The validation of the algorithm is presented for the second-order inverse dynamics of the Franka Emika Panda using the algorithm based on PGA and the algorithm based on product of exponentials (POE) respectively. The proposed algorithm is 15% faster than the POE-based algorithm with correct results. It turns out that the kinematics part of the algorithm saves 69.82% multiplications and 73.58% additions than that in the POE-based algorithm. The relation between PGA and other popular concepts in robotics, such as dual quaternions, is also discussed.

Keywords Projective geometric algebra · Serial robots · High-order inverse dynamics

1 Introduction

Various applications of robots necessitate to compute the first-order and the second-order derivatives of the joint torques or forces. Typical examples include the motion planning with constrains on the motion of the end-effector and the motion control of robots equipped with serial elastic actuators and variable stiffness actuators [1]. Taking the elastic actuator as an example, it is usually modeled as a rotor connecting a spring. The equations of motion of the robot are usually formulated as differential equations about the rotation variables of motors rather than those of joints. The variables of joints in the equations are canceled by substituting the equations of motion of rotors, and the derivatives of the joint torques are

✉ Y. Ding
y.ding@sjtu.edu.cn

[1] State Key Laboratory of Mechanical System and Vibration, School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, 200240, China

included [2]. With the equations considering the model of elastic actuators, energy-saving and efficient control strategies can be designed [3–5]. An efficient but simple and intuitive algorithm is thus desired to implement the high-order inverse dynamics of serial robots.

Different methods have been developed to model serial robots and to construct the algorithms for high-order inverse dynamics. The Denavit–Hartenberg method (D-H method) [6] provides a standard procedure to model serial robots, and it has been a classic method. However, the special procedure to attach references to links can be cumbersome. A computationally recursive algorithm for the inverse dynamics was then constructed based on the Newton–Euler formulation and the D-H method. Translations and rotations are treated separately in this method, and it makes the high-order inverse dynamics hard to be formulated. Based on Ball's screw theory, the product of exponentials (POE) is proposed [7] and gains more and more robotics researchers' interest [8]. The POE method models serial robots with lower pair joints directly in terms of geometric data, and it simplifies the modeling procedure significantly. In the perspective of the POE, translations and rotations are both part of screw motions. The dynamic equations of a single rigid body are formulated in a compact form, which simplifies the recursive algorithm for the inverse dynamics. The exponential map in the POE method maps an element in the Lie algebra of the special Euclidean group $SE(3)$, denoted as $se(3)$, to an element in $SE(3)$. It is actually a map from a Lie algebra to a Lie group. The theory of the Lie group and the Lie algebra is then introduced to the POE method [9]. It turns out that the derivatives of inertia items in the dynamics equation and the geometric Jacobian can both be implemented by Lie brackets. It significantly simplifies the calculations of derivatives. Based on this property, Müller proposed an efficient recursive algorithm for 2-order inverse dynamics of serial robots [2, 10] and explored the applications of this property in other problems [11]. Singh et al. [12] proposed an efficient analytical algorithm for this problem using spatial vector algebra. Cibicik and Egeland [13] analyzed kinematics and dynamics of robots using dual screws. Silva et al. [14] provided a comprehensive introduction to the dynamics of robots using dual quaternion algebra. These theories provide diverse views to understand the geometry of serial robots, and efficient algorithms are constructed. However, space for improvement remains. In the POE theory, repetitive elements in the adjoint matrix of twists and homogeneous matrices should be reduced without the loss of physical or geometric meaning. The exponential map with complex matrix calculations should also be simplified. A more uniform and intuitive theory with the ability to construct efficient algorithms still needs to be explored.

One common base of methods mentioned above is the system of Gibbs' vectors [15]. However, a more general system of vectors exists, named geometric algebra (GA), also known as Clifford algebra. GA constructs a space with not only "inner product" but also "outer product". It generalizes the representations of different geometric elements and models geometric operations in algebraic operations. The great power of GA has stood out in the field of theoretical physics [16] [17], signal processing [18], and computer science. Researchers also begin to introduce it in the field of robotics. However, related research is scattered in the literature, and advanced study is still desired. Among the related research, Bayro-Corrochano has discussed GA's applications in robotics and has solved many problems such as the kinematics [19], machine vision [20], and motion control [21]. Selig [22] formulated rigid body dynamics using GA and pointed out that the dynamics of serial robots can be cast into a GA form. Hestenes [23] discussed the application of GA in rigid body mechanics with elastic coupling and proposed the simplification brought by GA.

Gunn [24–26] proposed a method applying projective geometric algebra (PGA) for Euclidean geometry and found it suitable to solve problems in not only Euclidean geometry, but also elliptic and hyperbolic geometry. He also specified the dual projectivized Clifford

algebra and established that it is the smallest structure-preserving GA for Euclidean geometry. Hadfield [27] then implemented constrained dynamics with both the PGA proposed by Gunn and conformal geometric algebra (CGA). PGA is also applied to classical mechanics, and the dynamic models of both mass particles and rigid bodies can be constructed. However, research that applies PGA to serial robots and constructs both the kinematics and the dynamics model is still not reported in the literature.

Based on the research of Gunn [25], this paper proposes a method for modeling serial robots based on the PGA for Euclidean geometry, including the geometric model, the kinematics model, and the dynamics model. A recursive algorithm to implement the high-order inverse dynamics of serial robots is then constructed. It turns out that the algorithm is computationally efficient, intuitive, and friendly to beginners, and it is coordinate invariant, the same as the POE method. In this method, calculations of the exponential map and Lie brackets are simplified and rigid body motions are represented as vectors instead of matrices, which improves the efficiency and saves memories. All operations are modeled as algebraic operations with explicit geometric meaning, and this characteristic makes the method intuitive.

The paper is organized as follows. Section 2 introduces basic concepts of the PGA used to model serial robots. Section 3 presents the geometry model, the kinematics model, and the dynamics model of a serial robot successively and constructs the algorithm for high-order inverse dynamics. In Sect. 4, the computational performance of Lie brackets, the exponential map, and the algorithm is analyzed and mics of the Franka Emika Pand is used to verify the algorithm based on the PGA. Then the algorithm is compared with the algorithm proposed by Müller in [2] for the efficiency. The paper closes with conclusion about the advantages of PGA-based method and discussion about the relation between PGA and other concepts used in robotics, including manifolds, Lie group, Lie algebra, and dual quaternions.

2 Theory about projective geometric algebra

2.1 The projective space $\mathbb{R}P^3$

A projective space is obtained from a vector space V by introducing an equivalence relation: $\forall \boldsymbol{x}, \boldsymbol{y} \in V \setminus \{\boldsymbol{0}\}, \boldsymbol{x} \sim \boldsymbol{y} \iff \exists \lambda \neq 0$, s.t. $\boldsymbol{x} = \lambda \boldsymbol{y}$. If $V = \mathbb{R}^{n+1}$, then the projective space is called a real projective n-space and is denoted as $\mathbb{R}P^n$.

Classical mechanics is constructed in a 3-dimensional Euclidean space $\mathbb{E}^3$, which is an affine space $\mathbb{A}^3$ with Euclidean metric [28]. In the space, a plane is represented as a linear equation: $ax + by + cz + d = 0$. A one-to-one map is then established between the coefficients of the equation and a vector in the vector space $\mathbb{R}^4$: $\boldsymbol{p} = a\boldsymbol{e}_1 + b\boldsymbol{e}_2 + c\boldsymbol{e}_3 + d\boldsymbol{e}_0$, where $\{\boldsymbol{e}_i\}$ is a set of basis vectors. However, $\lambda \boldsymbol{p}$ $(\lambda \in \mathbb{R}, \lambda \neq 0)$ represents the same plane as p. Thus, a plane is treated as an element in the projective space $\mathbb{R}P^3$.

In projective geometry, a projective space is called a dual projective space when its elements represent hyperplane. It is denoted as $(\mathbb{R}P^n)^*$. In this article, the denotation $\mathbb{R}P^n$ is used instead. Projective geometric algebra is then constructed on the basis of $\mathbb{R}P^3$.

2.2 The projectivized exterior algebra $\mathrm{P}\left(\bigwedge \mathbb{R}^4\right)$

Define an anti-symmetric bilinear associative operator in $\mathbb{R}^4$, named outer product and denoted as "$\wedge$". According to the property of anti-symmetric, the outer products of basis vec-

tors are as follows:

$$\boldsymbol{e}_i \wedge \boldsymbol{e}_i = -\boldsymbol{e}_i \wedge \boldsymbol{e}_i = 0, \tag{1a}$$

$$\boldsymbol{e}_i \wedge \boldsymbol{e}_j = -\boldsymbol{e}_j \wedge \boldsymbol{e}_i = \boldsymbol{e}_{ij}\ (i \neq j). \tag{1b}$$

As shown above, new basis vectors emerge: $\{\boldsymbol{e}_{23}, \boldsymbol{e}_{31}, \boldsymbol{e}_{12}, \boldsymbol{e}_{01}, \boldsymbol{e}_{02}, \boldsymbol{e}_{03}\}$, and a new 6-dimensional vector space is generated. We denote the 6-dimensional vector space as $\bigwedge^2 \left(\mathbb{R}^4\right)$. Vectors in this space are called 2-vectors, and the space is with grade 2. Likewise, other two vector spaces of higher grade are generated by the outer product of basis vectors: a 4-dimensional vector space with basis $\{\boldsymbol{e}_{032}, \boldsymbol{e}_{021}, \boldsymbol{e}_{013}, \boldsymbol{e}_{123}\}$ and a 1-dimensional vector space with basis $\{\boldsymbol{I} = \boldsymbol{e}_{0123}\}$. More specifically, $\boldsymbol{e}_{ijk} = \boldsymbol{e}_i \wedge \boldsymbol{e}_j \wedge \boldsymbol{e}_k$ and $\boldsymbol{e}_{0123} = \boldsymbol{e}_0 \wedge \boldsymbol{e}_1 \wedge \boldsymbol{e}_2 \wedge \boldsymbol{e}_3$, where $i, j, k = 0, 1, 2, 3$. The two higher grade spaces are denoted as $\bigwedge^3 \left(\mathbb{R}^4\right)$ and $\bigwedge^4 \left(\mathbb{R}^4\right)$, and vectors in them are called 3-vectors and pseudo-scalars, respectively. The real number is with grade 0 as defined, $\bigwedge^0 \left(\mathbb{R}^n\right) := \mathbb{R}$. The direct sum of them is defined as the exterior algebra $\bigwedge \mathbb{R}^4$:

$$\bigwedge \mathbb{R}^4 := \mathbb{R} \oplus \mathbb{R}^4 \oplus \overset{2}{\bigwedge}\left(\mathbb{R}^4\right) \oplus \overset{3}{\bigwedge}\left(\mathbb{R}^4\right) \oplus \overset{4}{\bigwedge}\left(\mathbb{R}^4\right). \tag{2}$$

The exterior algebra can also be projectivized by applying the equivalence relation to the vector space $\bigwedge^k \left(\mathbb{R}^4\right)$. It yields the projectivized exterior algebra

$$\mathbf{P}\left(\bigwedge \mathbb{R}^4\right) := \mathbf{P}\left(\mathbb{R}\right) \oplus \mathbf{P}\left(\mathbb{R}^4\right) \oplus \cdots \oplus \mathbf{P}\left(\overset{4}{\bigwedge}\left(\mathbb{R}^4\right)\right). \tag{3}$$

Operations between geometric elements can be implemented by outer product. Without loss of generality, two vectors in $\mathbb{R}^4$ are given, representing two planes $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ as follows:

$$\boldsymbol{p}_1 = a_1\boldsymbol{e}_1 + b_1\boldsymbol{e}_2 + c_1\boldsymbol{e}_3 + d_1\boldsymbol{e}_0, \tag{4a}$$

$$\boldsymbol{p}_2 = a_2\boldsymbol{e}_1 + b_2\boldsymbol{e}_2 + c_2\boldsymbol{e}_3 + d_2\boldsymbol{e}_0. \tag{4b}$$

Suppose that they are not parallel to each other, and the outer product is

$$\begin{aligned} \boldsymbol{p}_1 \wedge \boldsymbol{p}_2 =& (b_1c_2 - b_2c_1)\boldsymbol{e}_{23} + (a_2c_1 - a_1c_2)\boldsymbol{e}_{31} \\ &+ (a_1b_2 - a_2b_1)\boldsymbol{e}_{12} + (a_2d_1 - a_1d_2)\boldsymbol{e}_{01} \\ &+ (b_2d_1 - b_1d_2)\boldsymbol{e}_{02} + (c_2d_1 - c_1d_2)\boldsymbol{e}_{03} \\ =& [\boldsymbol{e}_{23} \quad \boldsymbol{e}_{31} \quad \boldsymbol{e}_{12} \quad \boldsymbol{e}_{01} \quad \boldsymbol{e}_{02} \quad \boldsymbol{e}_{03}]\, \underline{\mathbf{L}}. \end{aligned} \tag{5}$$

$\underline{\mathbf{L}} \in \mathbb{R}^6$ is the coordinate with respect to the basis. It turns out that $\underline{\mathbf{L}}$ is the Plücker coordinate of the intersecting line of two planes, up to a scalar factor [29]. In the projective space $\mathbf{P}\left(\bigwedge^2 \left(\mathbb{R}^4\right)\right)$, $\underline{\mathbf{L}}$ and $\lambda\underline{\mathbf{L}}(\lambda \neq 0)$ represent the same line.

To be more specific, the plane $x - 1 = 0$ and the plane $y = 0$ are represented as $\boldsymbol{e}_1 - \boldsymbol{e}_0$ and $\boldsymbol{e}_2$. The outer product is $\boldsymbol{e}_{12} - \boldsymbol{e}_{02}$, and $\underline{\mathbf{L}} = (0, 0, 1, 0, -1, 0)^T$. It represents the line parallel to z axis and crossing the point $(1, 0, 0)$. Its Plücker coordinate is $(0, 0, 1, 0, -1, 0)^T$.

If two planes are parallel, then $(a_1, b_1, c_1) = \lambda(a_2, b_2, c_2)\ (\lambda \neq 0)$. The outer product is

$$\begin{aligned} \boldsymbol{p}_1 \wedge \boldsymbol{p}_2 =& (d_1 - \lambda d_2)(a_2\boldsymbol{e}_{01} + b_2\boldsymbol{e}_{02} + c_2\boldsymbol{e}_{03}) \\ =& [\boldsymbol{e}_{23} \quad \boldsymbol{e}_{31} \quad \boldsymbol{e}_{12} \quad \boldsymbol{e}_{01} \quad \boldsymbol{e}_{02} \quad \boldsymbol{e}_{03}]\, \underline{\mathbf{L}}. \end{aligned} \tag{6}$$

Table 1 The representation of geometry elements in $\mathbf{P}(\bigwedge \mathbb{R}^4)$

Name	Representation in $\mathbf{P}(\bigwedge \mathbb{R}^4)$
Planes	$\boldsymbol{p} = a\boldsymbol{e}_1 + b\boldsymbol{e}_2 + c\boldsymbol{e}_3 + d\boldsymbol{e}_0$
Lines	$\boldsymbol{l} = d_x\boldsymbol{e}_{23} + d_y\boldsymbol{e}_{31} + d_z\boldsymbol{e}_{12} + p_x\boldsymbol{e}_{01} + p_y\boldsymbol{e}_{01} + p_z\boldsymbol{e}_{03}$
Points	$\boldsymbol{P} = x\boldsymbol{e}_{032} + y\boldsymbol{e}_{013} + z\boldsymbol{e}_{021} + \boldsymbol{e}_{123}$

In projective geometry, the intersection of two parallel planes is an infinite line [30]. The coordinate of the outer product $\underline{\mathbf{L}}$ is again the Plücker coordinate of the intersecting infinite line. In summary, the outer product of any two planes implements the operation "meet" and produces a 2-vector in $\mathbf{P}\left(\bigwedge^2(\mathbb{R}^4)\right)$, representing the intersecting line.

Furthermore, suppose another plane $\boldsymbol{p}_3$ is given, and any two of the three planes are not parallel.

$$\boldsymbol{p}_3 = a_3\boldsymbol{e}_1 + b_3\boldsymbol{e}_2 + c_3\boldsymbol{e}_3 + d_3\boldsymbol{e}_0. \tag{7}$$

The outer product of them is

$$\begin{aligned}\boldsymbol{p}_1 \wedge \boldsymbol{p}_2 \wedge \boldsymbol{p}_3 = &-(b_1c_2d_3 - b_1c_3d_2 + b_2c_3d_1\\ &- b_2c_1d_3 + b_3c_1d_2 - b_3c_2d_1)\boldsymbol{e}_{032}\\ &+ (a_1c_2d_3 - a_1c_3d_2 + a_2c_3d_1\\ &- a_2c_1d_3 + a_3c_1d_2 - a_3c_2d_1)\boldsymbol{e}_{013}\\ &- (a_1b_2d_3 - a_1b_3d_2 + a_2b_3d_1\\ &- a_2b_1d_3 + a_3b_1d_2 - a_3b_2d_1)\boldsymbol{e}_{021}\\ &+ (a_1b_2c_3 - a_1b_3c_2 + a_2b_3c_1\\ &- a_2b_1c_3 + a_3b_1c_2 - a_3b_2c_1)\boldsymbol{e}_{123}\\ =&[\boldsymbol{e}_{032}\quad \boldsymbol{e}_{013}\quad \boldsymbol{e}_{021}\quad \boldsymbol{e}_{123}]\underline{\mathbf{P}}.\end{aligned} \tag{8}$$

$\underline{\mathbf{P}} \in \mathbb{R}^4$ is the coordinate with respect to the basis. With Cramer's rule, it turns out that $\underline{\mathbf{P}}$ is the homogeneous coordinate of the intersecting point, up to a scalar factor. In the projective space $\mathbf{P}\left(\bigwedge^3(\mathbb{R}^4)\right)$, $\boldsymbol{P}$ and $\lambda\boldsymbol{P}$ $(\lambda \neq 0)$ represent the same point.

If any two of the planes are parallel, the coefficient of $\boldsymbol{e_{123}}$ is zero and it produces an infinite point in the projective space $\mathbf{P}\left(\bigwedge^3(\mathbb{R}^4)\right)$. If all the three planes are parallel, the outer product is 0. It means that the intersection of the three planes is not a point.

In summary, planes, lines, and points can be represented by vectors, 2-vectors, and 3-vectors in the corresponding projective space, respectively. It is summarized in Table 1. Meet of geometric elements can be implemented by the outer product. More details about the projectivized exterior algebra can be found in Chap. 2 of [26].

2.3 The metric in $\mathbb{R}P^3$

With a symmetric bilinear form defined in $\mathbb{R}^4$, the element in $\mathbb{R}P^3$ can be represented by one specific vector in $\mathbb{R}^4$. The symmetric bilinear form is called the inner product. According to Sylvester signature theorem, a symmetric bilinear form of dimension n is completely characterized by three integers (p, m, z) satisfying $p + m + z = n$ [30]. The integers (p, m, z)

are the signature of the bilinear form, implying p bases' self inner product is 1, m bases' is -1, and z bases' is 0, in n orthogonal bases. The signature $(3, 0, 1)$ is chosen for doing Euclidean geometry. For a basis $\{\boldsymbol{e}_i\}$ of $\mathbb{R}^4$, the signature means

$$\boldsymbol{e}_i \cdot \boldsymbol{e}_j = 0 (i \neq j), \tag{9a}$$

$$\boldsymbol{e}_i \cdot \boldsymbol{e}_i = 1 (i = 1, 2, 3), \boldsymbol{e}_0 \cdot \boldsymbol{e}_0 = 0. \tag{9b}$$

The inner product with signature $(3, 0, 1)$ naturally induces a norm in $\mathbb{R}^4$. For a vector $\boldsymbol{p} = a\boldsymbol{e}_1 + b\boldsymbol{e}_2 + c\boldsymbol{e}_3 + d\boldsymbol{e}_0$, its norm is defined as

$$\|\boldsymbol{p}\| = \sqrt{\boldsymbol{p} \cdot \boldsymbol{p}} = \sqrt{a^2 + b^2 + c^2}. \tag{10}$$

Applying the norm to $\mathbb{R}P^3$, it measures the length of the plane's normal vector, which is defined as the weight of $\boldsymbol{p}$ in $\mathbb{R}P^3$. The weight of a vector in the projective space is used to model the mass of a point in the dynamics of a rigid body.

Vectors with nonzero norm in $\mathbb{R}^4$ can be normalized as $\hat{\boldsymbol{p}} = \boldsymbol{p}/\|\boldsymbol{p}\|$. Then a normalized vector corresponds to an oriented plane in the Euclidean space with a unit normal vector, and an oriented plane corresponds to a normalized vector. With normalization, a plane in $\mathbb{R}P^3$ can be represented by at most two vectors that differ in a coefficient -1.

The metric in $\mathbb{R}P^3$ is also induced from the inner product. It measures the "distance" between two planes. Based on the theory of Cayley–Klein metric, the metric between two planes $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ in $\mathbb{R}P^3$ is defined as

$$d(\boldsymbol{p}_1, \boldsymbol{p}_2) = \cos^{-1}\left(\hat{\boldsymbol{p}}_1 \cdot \hat{\boldsymbol{p}}_2\right). \tag{11}$$

It turns out that the "distance" between two planes is actually the angle between their normal vectors.

Remark If the signature $(1, 0, 3)$ is chosen and points in an Euclidean space are modeled as elements in $\mathbb{R}P^3$, the Euclidean metric can be induced via limiting process [26]. A normalized vector is in the form $P = x\boldsymbol{e}_1 + y\boldsymbol{e}_2 + z\boldsymbol{e}_3 + \boldsymbol{e}_0$, which is the same as the representation of points in projective geometry.

In summary, the inner product defined in $\mathbb{R}^4$ naturally induces the norm and the metric in $\mathbb{R}P^3$. The "distance" between two geometric elements is then well defined. For more details about metric in projective geometry, readers are referred to Chap. 3 of [26].

2.4 The geometric product and $\mathbf{P}(\mathbb{R}_{(3,0,1)})$

Clifford summarized the property of the outer product and the inner product and introduced a new bilinear associative operator, the geometric product [26]. It is defined as

$$\boldsymbol{ab} = \boldsymbol{a} \cdot \boldsymbol{b} + \boldsymbol{a} \wedge \boldsymbol{b}, \ \forall \boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n. \tag{12}$$

The geometric product is a more general operator. It contains a property of both the inner product and the outer product, like a coin contains two sides. Both operators can be induced from the geometric product.

$$\boldsymbol{a} \cdot \boldsymbol{b} = \frac{\boldsymbol{ab} + \boldsymbol{ba}}{2}, \tag{13}$$

$$\boldsymbol{a} \wedge \boldsymbol{b} = \frac{\boldsymbol{ab} - \boldsymbol{ba}}{2}. \tag{14}$$

The geometric product maps two vectors from $\mathbb{R}^n$ to $\mathbb{R} \oplus \bigwedge^2 (\mathbb{R}^n)$. Successively, it extends a vector space $\mathbb{R}^n$ to the direct sum of vector spaces with grade from 0 to n. The resultant vector space is denoted as $\mathbb{R}_{(p,m,z)}$, where (p, m, z) is the signature of the inner product.

$$\mathbb{R}_{(p,m,z)} := \oplus_{i=0}^{n} \bigwedge^{i} (\mathbb{R}^n). \tag{15}$$

Vectors in this space are called multivectors. The projectivized space based on $\mathbb{R}_{(p,m,z)}$ is denoted as $\mathrm{P}\left(\mathbb{R}_{(p,m,z)}\right)$. The space $\mathrm{P}\left(\mathbb{R}_{(p,m,z)}\right)$ contains the same elements as $\mathrm{P}\left(\bigwedge \mathbb{R}^n\right)$ but with the geometric product. The resultant algebra is called projective geometric algebra (PGA).

For the PGA $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$, vectors representing geometric elements introduced in Sect. 2.2 are all contained. A k-vector $\boldsymbol{v}$ represents the same geometric element as $\lambda \boldsymbol{v} (\lambda \neq 0)$. Nevertheless, because of the inner product, $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$ possesses the metric in $\mathbb{R}P^3$ introduced in Sect. 2.3. According to the property of the geometric product, the metric in $\mathrm{P}(\bigwedge^2 \mathbb{R}^4)$ and $\mathrm{P}(\bigwedge^3 \mathbb{R}^4)$ is then induced. It turns out that the metric in $\mathrm{P}(\bigwedge^3 \mathbb{R}^4)$, the subspace composed of points, is the Euclidean metric [26]. Therefore, the PGA $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$ is suitable to model the 3-dimensional Euclidean space E^3.

The PGA can solve geometry problems in a more compact and concise way. In rigid body dynamics, the most important transform is the screw motion of a rigid body's geometric object. In the PGA, the screw motion is composed of two reflections, which is implemented by the operation called "sandwich".

From the perspective of algebra, reflection is such a transform that the effect of double transforms is the same as an identity transform. In projective geometry, a reflection is realized by a harmonic homology [26]. A plane b reflecting about another plane a is implemented by the following equation:

$$R(\boldsymbol{b}) = - < \boldsymbol{a}^{\perp}, \boldsymbol{a} > \boldsymbol{b} + 2 < \boldsymbol{a}^{\perp}, \boldsymbol{b} > \boldsymbol{a}. \tag{16}$$

In the equation, both $\boldsymbol{a}$ and $\boldsymbol{b}$ are normalized. $\boldsymbol{a}^{\perp}$ is the polar of $\boldsymbol{a}$ relative to the quadratic form induced from the inner product, and it is a dual vector. The operator $<, >$ is the bilinear function that defines the duality: $<, >: V^* \times V \to \mathbb{R}$, where V^* is the dual linear space of the linear space V.

According to the property of the outer product, the k-vector space is dual to the (n-k)-vector space. We first define an operator as

$$\Delta : \bigwedge^{n} (\mathbb{R}^n) \to \mathbb{R}, \text{ such that } \Delta(\lambda I) = \lambda. \tag{17}$$

$\boldsymbol{I}$ is the basis $\boldsymbol{e}_{012\ldots n}$. Then, suppose that $\boldsymbol{x} \in \bigwedge^k (\mathbb{R}^n)$ and $\boldsymbol{u} \in \bigwedge^{n-k} (\mathbb{R}^n)$. A nondegenerate bilinear function can be defined as

$$< u, x >= \Delta(u \wedge x). \tag{18}$$

The duality is then generated according to $<,>: V^* \times V \to \mathbb{R}$. It is called the Poincaré duality [26]. An explicit form of the Poincaré duality is defined as a linear operator:

$$\begin{gathered} J : \bigwedge^{k}(\mathbb{R}^n) \longrightarrow \bigwedge^{n-k}(\mathbb{R}^n) \\ \text{such that } J(\boldsymbol{e}_i^k) = \boldsymbol{e}_i^{n-k} \iff \boldsymbol{e}_i^k \wedge \boldsymbol{e}_i^{n-k} = \pm \boldsymbol{I}. \end{gathered} \tag{19}$$

In the equation, $\boldsymbol{e}_i^k$ is the ith basis k-vector. In $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$, the duality implies that a plane is dual to a point and a line is dual to another line.

Furthermore, denote the bilinear function defining the inner product as $B(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$. A linear transform is then naturally induced as $L_B : \mathbb{R}^n \to \mathbb{R}^{n*}$, $L_B(\boldsymbol{a}) := B(\boldsymbol{a}, \cdot)$, where $\boldsymbol{a} \in \mathbb{R}^n$. The inner product can also be implemented in another way:

$$\boldsymbol{a} \cdot \boldsymbol{b} =< L_B(\boldsymbol{a}), \boldsymbol{b} >, \ \forall \boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n. \tag{20}$$

According to the property of the geometric product and the duality defined in equation (18), it turns out that in $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$,

$$\boldsymbol{a} \cdot \boldsymbol{b} = \Delta[(-\boldsymbol{a}\boldsymbol{I}) \wedge \boldsymbol{b}], \ \forall \boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^4, \tag{21}$$

$$L_B(\boldsymbol{a}) = -\boldsymbol{a}\boldsymbol{I}, \ \forall \boldsymbol{a} \in \mathbb{R}^4. \tag{22}$$

In projective geometry, the polar of $\boldsymbol{a} \in \mathbb{R}^n$ is defined as a set:

$$\boldsymbol{a}^{\perp} = \{\boldsymbol{x} \in \mathbb{R}^n \mid B(\boldsymbol{a}, \boldsymbol{x}) = 0\}. \tag{23}$$

In $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$, it is represented as

$$\boldsymbol{a}^{\perp} = \{\boldsymbol{x} \in \mathbb{R}^4 \mid \Delta[(-\boldsymbol{a}\boldsymbol{I}) \wedge \boldsymbol{x}] = 0\}. \tag{24}$$

It implies that all vectors in $\boldsymbol{a}^{\perp}$ are contained in a linear subspace defined by $-\boldsymbol{a}\boldsymbol{I}$. Therefore, the polar of $\boldsymbol{a}$ is defined as $\boldsymbol{a}^{\perp} = -\boldsymbol{a}\boldsymbol{I}$ in $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$.

Now review the harmonic homology defining the reflection in PGA. It obtains a much more concise form and is indeed a reflection.

$$\begin{aligned} R_{\boldsymbol{a}}(\boldsymbol{b}) &= -(\boldsymbol{a} \cdot \boldsymbol{a})\boldsymbol{b} + 2(\boldsymbol{a} \cdot \boldsymbol{b})\boldsymbol{a} \\ &= -\boldsymbol{a}^2\boldsymbol{b} + (\boldsymbol{a}\boldsymbol{b} + \boldsymbol{b}\boldsymbol{a})\boldsymbol{a} \\ &= -\boldsymbol{b} + \boldsymbol{a}\boldsymbol{b}\boldsymbol{a} + \boldsymbol{b} \\ &= \boldsymbol{a}\boldsymbol{b}\boldsymbol{a}, \end{aligned} \tag{25}$$

$$\begin{aligned} R_{\boldsymbol{a}}(R_{\boldsymbol{a}}(\boldsymbol{b})) &= \boldsymbol{a}(\boldsymbol{a}\boldsymbol{b}\boldsymbol{a})\boldsymbol{a} \\ &= \boldsymbol{a}^2\boldsymbol{b}\boldsymbol{a}^2 \\ &= \boldsymbol{b}. \end{aligned} \tag{26}$$

From equation (25) and equation (26), it turns out that the result of a reflection is nothing to do with the form of $\boldsymbol{b}$. Even if $\boldsymbol{b}$ is a general multivector, the equation $R_{\boldsymbol{a}}(R_{\boldsymbol{a}}(\boldsymbol{b})) = \boldsymbol{b}$ remains. It means that a reflector $R_{\boldsymbol{a}}$ only depends on a normalized vector $\boldsymbol{a}$. Besides, $R_{-\boldsymbol{a}} = R_{\boldsymbol{a}}$, which implies that a reflection is irrelevant to $\boldsymbol{a}$'s sign. In $\mathbb{R}P^3$, a reflector only depends on a plane. The plane is called the axis of reflection. Readers are referred to Chap. 6 and Chap. 7 of [26] for more details about this section.

2.5 The rigid body motion modeled in $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$

Based on the reflection defined in (25), the rigid body motion can be defined as twice reflections. Suppose two different planes $\boldsymbol{a}$ and $\boldsymbol{b}$ are given, and define the rigid body motion of a multivector $\boldsymbol{X}$ in $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$ as

$$\boldsymbol{M}_{\boldsymbol{ab}}(\boldsymbol{X}) = \boldsymbol{baXab}. \tag{27}$$

Define a motor as $\boldsymbol{M} = \boldsymbol{ab}$, and its reverse $\tilde{\boldsymbol{M}} = \boldsymbol{ba}$. It turns out that

$$\tilde{\boldsymbol{M}}\boldsymbol{M} = \boldsymbol{M}\tilde{\boldsymbol{M}} = 1. \tag{28}$$

The rigid body motion of a multivector $\boldsymbol{x}$ in $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$ is then represented as

$$\boldsymbol{M}_{\boldsymbol{ab}}(\boldsymbol{X}) = \tilde{\boldsymbol{M}}\boldsymbol{X}\boldsymbol{M}. \tag{29}$$

We first suppose that the plane $\boldsymbol{a}$ is not parallel to $\boldsymbol{b}$. According to equation (5) and equations (9a), (9b), the motor $\boldsymbol{M}$ is

$$\boldsymbol{M} = \boldsymbol{ab} = \boldsymbol{a}\cdot\boldsymbol{b} + \boldsymbol{a}\wedge\boldsymbol{b} = \cos\theta + \sin\theta\boldsymbol{L}. \tag{30}$$

θ is the angle between two planes. $\boldsymbol{L}$ is a normalized 2-vector, normalized by $\boldsymbol{L}^2 = -1$. A motor in this form acts as a rotor that rotates an object along the line represented as $\boldsymbol{L}$ by angle 2θ.

Remark For θ_1 and θ_2 such that $\boldsymbol{M}_{\theta_1} = -\boldsymbol{M}_{\theta_2}$, there is $\tilde{\boldsymbol{M}}_{\theta_1}\boldsymbol{X}\boldsymbol{M}_{\theta_1} = \tilde{\boldsymbol{M}}_{\theta_2}\boldsymbol{X}\boldsymbol{M}_{\theta_2}$, $\forall \boldsymbol{X} \in \mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$. The sign of the motor $\boldsymbol{M}$ has no impact on the result of the rigid body motion.

If $\boldsymbol{a}$ is parallel to $\boldsymbol{b}$, then the motor $\boldsymbol{M}$ is formulated according to equation (6):

$$\boldsymbol{M} = \boldsymbol{ab} = 1 + d\boldsymbol{L}_\infty. \tag{31}$$

d is the distance between two planes. $\boldsymbol{L}_\infty$ is an infinite line defined in equation (6). It is normalized by $J(\boldsymbol{L}_\infty)^2 = -1$, where $J(\cdot)$ is the Poincaré duality defined in equation (19). The motor of this kind acts as a translator moving the object in the direction from $\boldsymbol{a}$ to $\boldsymbol{b}$ by the distance $2d$.

Define the exponential mapping by the multinomial series:

$$e^{\boldsymbol{L}} = \sum_{k=0}^{\infty} \frac{\boldsymbol{L}^k}{k!}. \tag{32}$$

Multiplications in the equation are implemented by the geometric product.

Both the rotor and the translator mentioned above can be calculated by the exponential mapping in a uniform way:

$$\boldsymbol{M} = e^{\frac{w}{2}\boldsymbol{L}}, \tag{33}$$

$$\tilde{\boldsymbol{M}} = e^{-\frac{w}{2}\boldsymbol{L}}. \tag{34}$$

w is the intensity of motion, i.e., the angle of the rotation and the distance of the translation. $\boldsymbol{L}$ is a normalized 2-vector that represents a line or an infinite line and implies the axis of the

rotation or the direction of the translation. It is summarized that a rotation or a translation can be uniquely identified by a 2-vector that represents a weighted line.

To be more general, the screw motion should be implemented. A screw motion is such a motion that rotates about an axis and translates along the same axis simultaneously with a pitch. Suppose that an object rotates about a line $\boldsymbol{L}$ by the angle θ and translates along the direction $\boldsymbol{L}_\infty$ by the distance d. It is proved that $\boldsymbol{L}_\infty = \tilde{\boldsymbol{L}}\boldsymbol{I} = \boldsymbol{I}\tilde{\boldsymbol{L}}$ if $\boldsymbol{L}_\infty$'s direction is the same as $\boldsymbol{L}$'s, i.e., the resulting motion is a screw motion. Besides, the order of the rotation and the translation has no impact on the effect of the motion. Then the general screw motion of a rigid body is modeled as follows:

$$\begin{aligned}\boldsymbol{M}(\boldsymbol{X}) &= e^{-\frac{\theta}{2}\boldsymbol{L}}\left(e^{-\frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}}\boldsymbol{X}e^{\frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}}\right)e^{\frac{\theta}{2}\boldsymbol{L}} \\ &= e^{-\frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}}\left(e^{-\frac{\theta}{2}\boldsymbol{L}}\boldsymbol{X}e^{\frac{\theta}{2}\boldsymbol{L}}\right)e^{\frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}} \\ &= \tilde{\boldsymbol{M}}\boldsymbol{X}\boldsymbol{M}.\end{aligned} \tag{35}$$

Because $(\tilde{\boldsymbol{L}}\boldsymbol{I})^k = 0, \forall k \geq 2$, the nth power of $\alpha\boldsymbol{L} + \beta\tilde{\boldsymbol{L}}\boldsymbol{I} (\forall \alpha, \beta \in \mathbb{R})$ satisfies the following equation when $n > 0$:

$$\begin{aligned}\frac{(\alpha\boldsymbol{L} + \beta\tilde{\boldsymbol{L}}\boldsymbol{I})^n}{n!} &= \frac{\alpha^n\boldsymbol{L}^n}{n!} + \frac{\alpha^{n-1}\boldsymbol{L}^{n-1}}{(n-1)!}(\beta\tilde{\boldsymbol{L}}\boldsymbol{I}) \\ &= \frac{\alpha^n\boldsymbol{L}^n}{n!} + (\beta\tilde{\boldsymbol{L}}\boldsymbol{I})\frac{\alpha^{n-1}\boldsymbol{L}^{n-1}}{(n-1)!}.\end{aligned} \tag{36}$$

Then the exponential of $\alpha\boldsymbol{L} + \beta\tilde{\boldsymbol{L}}\boldsymbol{I}$ is

$$\begin{aligned}e^{\alpha\boldsymbol{L}+\beta\tilde{\boldsymbol{L}}\boldsymbol{I}} &= 1 + \sum_{n=1}^{\infty}\left[\frac{\alpha^n\boldsymbol{L}^n}{n!} + \frac{\alpha^{n-1}\boldsymbol{L}^{n-1}}{(n-1)!}(\beta\tilde{\boldsymbol{L}}\boldsymbol{I})\right] \\ &= (1 + \beta\tilde{\boldsymbol{L}}\boldsymbol{I}) + \left(\sum_{n=1}^{\infty}\frac{\alpha^n\boldsymbol{L}^n}{n!}\right)(1 + \beta\tilde{\boldsymbol{L}}\boldsymbol{I}) \\ &= \left(\sum_{n=0}^{\infty}\frac{\alpha^n\boldsymbol{L}^n}{n!}\right)(1 + \beta\tilde{\boldsymbol{L}}\boldsymbol{I}) \\ &= e^{\alpha\boldsymbol{L}}e^{\beta\tilde{\boldsymbol{L}}\boldsymbol{I}} = e^{\beta\tilde{\boldsymbol{L}}\boldsymbol{I}}e^{\alpha\boldsymbol{L}}.\end{aligned} \tag{37}$$

For a motor representing a screw motion, it is

$$\boldsymbol{M} = e^{\frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}}e^{\frac{\theta}{2}\boldsymbol{L}} = e^{\frac{\theta}{2}\boldsymbol{L}}e^{\frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}} = e^{\frac{\theta}{2}\boldsymbol{L}+\frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}} = e^{\frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}+\frac{\theta}{2}\boldsymbol{L}}. \tag{38}$$

Therefore, if a screw motion is along the axis $\boldsymbol{L}$, and it rotates by θ and translates by d, then it is uniquely identified by a 2-vector in the form $\frac{\theta}{2}\boldsymbol{L} + \frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}$. The motor is derived by the exponential map $\boldsymbol{M} = \exp(\frac{\theta}{2}\boldsymbol{L} + \frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I})$.

Remark The 2-vector $\frac{\theta}{2}\boldsymbol{L} + \frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}$ is not the outer product of any two vectors and does not correspond to a line. The 2-vectors with this property are called nonsimple 2-vectors. The ones which are the outer product of two vectors are called simple 2-vectors. Without a proof,

Table 2 The rigid motion modeled in $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$

Motion type	2-vector $\boldsymbol{l}$	Motor $\boldsymbol{M} = e^{\boldsymbol{l}}$
Rotation	$\frac{\theta}{2}\boldsymbol{L}$	$\cos\frac{\theta}{2} + \sin\frac{\theta}{2}\boldsymbol{L}$
Translation	$\frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}$	$1 + \frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}$
Screw motion	$\frac{\theta}{2}\boldsymbol{L} + \frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}$	$(\cos\frac{\theta}{2} + \sin\frac{\theta}{2}\boldsymbol{L})(1 + \frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I})$

it is pointed out that any nonsimple 2-vectors $\boldsymbol{l}$ can be decomposed as the sum of two simple 2-vectors in the form $\frac{\theta}{2}\boldsymbol{L} + \frac{d}{2}\tilde{\boldsymbol{L}}\boldsymbol{I}$.

In summary, any rigid body motions can be identified uniquely by a 2-vector $\boldsymbol{l}$. Specifically, the pure rotation and the pure translation are identified by a simple 2-vector, while the general screw motion with both the rotation and the translation is identified by a nonsimple 2-vector. The motor $\boldsymbol{M}$ is the exponential of such a 2-vector $\boldsymbol{l}$, i.e., $\boldsymbol{M} = e^{\boldsymbol{l}}$, where the exponential map is defined by the multinomial series. The results are summarized in Table 2.

3 Model of serial robots

3.1 Geometry model of serial robots based on the PGA

As shown above, the rigid body motion in the Euclidean space $\mathbb{E}^3$ can be implemented by the operator M in $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$, $\boldsymbol{M} = e^{\boldsymbol{l}}$, and $\boldsymbol{M}$ is the exponential of a 2-vector. The results are then applied in the model of serial robots.

Generally, joints of serial robots are low-pairs, i.e., either prismatic joints or revolute joints. Relative to the link $i-1$, the motion of the link i is the rotation about or the translation along the axis of the joint connecting both links. Denote the joint as the joint i. From the perspective of geometry, the axis of a joint is a line. In $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$, the line can be represented by a 2-vector $\boldsymbol{L}_i$ and normalized as $\boldsymbol{L}_i^2 = -1$. Then the motion of any geometric objects attached to the link i, modeled as a multivector $\boldsymbol{X}$ in $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$, can be formulated as follows:

$$\boldsymbol{X}(t) = \tilde{\boldsymbol{M}}_i^J(t)\boldsymbol{X}^0\boldsymbol{M}_i^J(t), \tag{39}$$

$$\boldsymbol{M}_i^J(t) = \begin{cases} e^{\frac{\theta(t)}{2}\boldsymbol{L}_i}, & \text{the revolute joint,} \\ e^{\frac{d(t)}{2}\tilde{\boldsymbol{L}}_i\boldsymbol{I}}, & \text{the prismatic joint.} \end{cases} \tag{40}$$

The superscript J and the subscript i together indicate that the motor is generated by the joint i. $\boldsymbol{X}^0$ is the initial configuration of the geometric object $\boldsymbol{X}$ at $t = 0$. $\theta(t)$ and $d(t)$ are the angle and the distance of the motion, respectively. Minor values mean moving in the inverse direction of $\boldsymbol{L}_i$. For simplicity, we denote $\tilde{\boldsymbol{L}}_i\boldsymbol{I}$ also as $\boldsymbol{L}_i$ and denote d also as θ in the following part. Besides, all the expression "(t)" is omitted.

For a serial robot, suppose that the initial position of all joint axis is known. The line representing the axis of the joint i is denoted as $\boldsymbol{L}_i^0$. When the serial robot moves to another configuration, any geometry objects $\boldsymbol{X}$ attached to the link i first move about $\boldsymbol{L}_i^0$, then about $\boldsymbol{L}_{i-1}^0$, and so on. The process is formulated as follows:

$$\boldsymbol{X} = \tilde{\boldsymbol{M}}_i\boldsymbol{X}^0\boldsymbol{M}_i \tag{41}$$

$$\boldsymbol{M}_i = \boldsymbol{M}_i^J\boldsymbol{M}_{i-1}^J\ldots\boldsymbol{M}_1^J = \prod_{k=i}^{1} e^{\frac{\theta_k}{2}\boldsymbol{L}_k^0}. \tag{42}$$

An important fact is that the axis of the joint i is a line attached to the link $i-1$. It demonstrates tremendous power in simplifying the analysis of the differential kinematics of the serial robot, especially the calculation of higher-order Jacobian. We first formulate the motion of the axis i here,

$$\boldsymbol{L}_i = \tilde{\boldsymbol{M}}_{i-1} \boldsymbol{L}_i^0 \boldsymbol{M}_{i-1}. \tag{43}$$

3.2 Differential kinematics

In the PGA, the derivative of a multivector is implemented by taking the derivative of every basis vector's coefficient function. Distribution of geometric product over addition is established. Therefore, the product rule for derivatives is also established in the PGA. Take the derivative on both sides of equation (41), where $\boldsymbol{X}^0$ is a constant. The "velocity" of $\boldsymbol{X}$ is

$$\dot{\boldsymbol{X}} = \dot{\tilde{\boldsymbol{M}}} \boldsymbol{X}^0 \boldsymbol{M} + \tilde{\boldsymbol{M}} \boldsymbol{X}^0 \dot{\boldsymbol{M}}. \tag{44}$$

Here and in what follows, $\boldsymbol{M}$ stands for $\boldsymbol{M}_i$.

According to the property of $\boldsymbol{M}$ shown in equation (28), the derivative of $\boldsymbol{M}$ satisfies

$$\dot{\tilde{\boldsymbol{M}}} \boldsymbol{M} + \tilde{\boldsymbol{M}} \dot{\boldsymbol{M}} = 0 \Rightarrow \dot{\tilde{\boldsymbol{M}}} \boldsymbol{M} = -\tilde{\boldsymbol{M}} \dot{\boldsymbol{M}}. \tag{45}$$

Considering equation (41), equation (28), and equation (45), equation (44) is simplified:

$$\begin{aligned} \dot{\boldsymbol{X}} &= \dot{\tilde{\boldsymbol{M}}} \boldsymbol{M} \boldsymbol{X} + \boldsymbol{X} \tilde{\boldsymbol{M}} \dot{\boldsymbol{M}} \\ &= \boldsymbol{X} \tilde{\boldsymbol{M}} \dot{\boldsymbol{M}} - \tilde{\boldsymbol{M}} \dot{\boldsymbol{M}} \boldsymbol{X} \\ &= \frac{1}{2} (\boldsymbol{X} \boldsymbol{V} - \boldsymbol{V} \boldsymbol{X}) := [\boldsymbol{X}, \boldsymbol{V}]. \end{aligned} \tag{46}$$

Here, define the velocity of the rigid body as

$$\boldsymbol{V} := 2 \tilde{\boldsymbol{M}} \dot{\boldsymbol{M}}. \tag{47}$$

The velocity is independent of the geometric object $\boldsymbol{X}$. It is only identified by the motor $\boldsymbol{M}$ describing the rigid body motion and its derivative $\dot{\boldsymbol{M}}$.

Besides, a commutator $[\cdot, \cdot]$ induced from the geometric product is introduced.

$$[\boldsymbol{X}, \boldsymbol{V}] := \frac{1}{2} (\boldsymbol{X} \boldsymbol{V} - \boldsymbol{V} \boldsymbol{X}). \tag{48}$$

Remark It can be proved that when $\boldsymbol{X}$ is a 2-vector, the commutator is a Lie bracket. It implies that $\bigwedge^2(\mathbb{R}^4)$ induces a Lie algebra, which is closely related to the rigid body motion.

With the commutator, the velocity of the geometric object attached to the rigid body is expressed just like the velocity of a point on a rotating top ($\boldsymbol{v} = \boldsymbol{\omega} \times \boldsymbol{r}$). $\boldsymbol{X}$ serves as the position $\boldsymbol{r}$ and $\boldsymbol{V}$ serves as the angular velocity $\boldsymbol{\omega}$. The commutator serves as the cross product. Actually, the cross product is a special case of the commutator defined above when the geometric algebra is $\mathbb{R}_{(3,0,0)}$.

If $\boldsymbol{X}^0$ in equation (41) is a function of time t, equation (46) is generalized as follows. The general form of $\boldsymbol{X}$'s derivative is applied in dynamics widely.

$$\dot{\boldsymbol{X}} = \tilde{\boldsymbol{M}}_i \dot{\boldsymbol{X}}^0 \boldsymbol{M}_i + [\boldsymbol{X}, \boldsymbol{V}], \tag{49}$$

$$\dot{\boldsymbol{X}}^0 = \boldsymbol{M}_i \dot{\boldsymbol{X}} \tilde{\boldsymbol{M}}_i - [\boldsymbol{X}^0, \boldsymbol{M}_i \boldsymbol{V} \tilde{\boldsymbol{M}}_i]. \tag{50}$$

For a serial robot, the rigid body motion of the link i is identified by $\boldsymbol{M}_i$, so the velocity of link i is formulated as follows. We first take the derivative of $\boldsymbol{M}_i^J$.

$$\begin{aligned}
\dot{\boldsymbol{M}}_i^J &= \frac{\mathrm{d} e^{\frac{\theta(t)}{2}\boldsymbol{L}_i^0}}{\mathrm{d}t} \\
&= \frac{\mathrm{d}}{\mathrm{d}t} \sum_{n=0}^{\infty} \frac{\left(\frac{\theta(t)}{2}\boldsymbol{L}_i^0\right)^n}{n!} \\
&= \left(\sum_{n=0}^{\infty} \frac{\left(\frac{\theta(t)}{2}\boldsymbol{L}_i^0\right)^n}{n!} \right) \left(\frac{\dot{\theta}(t)}{2} \boldsymbol{L}_i^0 \right) \\
&= \boldsymbol{M}_i^J \left(\frac{\dot{\theta}(t)}{2} \boldsymbol{L}_i^0 \right).
\end{aligned} \tag{51}$$

According to equation (47), the velocity caused by the motion of joint i is then

$$\boldsymbol{V}_i^J = 2\tilde{\boldsymbol{M}}_i^J \dot{\boldsymbol{M}}_i^J = \dot{\theta}_i \boldsymbol{L}_i^0. \tag{52}$$

Then the velocity of link i is derived by

$$\begin{aligned}
\dot{\boldsymbol{M}}_i &= \sum_{k=1}^{i} \boldsymbol{M}_i^J \boldsymbol{M}_{i-1}^J \ldots \dot{\boldsymbol{M}}_k^J \ldots \boldsymbol{M}_1^J \\
&= \sum_{k=1}^{i} \boldsymbol{M}_i^J \boldsymbol{M}_{i-1}^J \ldots \dot{\boldsymbol{M}}_k^J \boldsymbol{M}_{k-1}
\end{aligned} \tag{53}$$

$$\tilde{\boldsymbol{M}}_i = \tilde{\boldsymbol{M}}_{k-1} \tilde{\boldsymbol{M}}_k^J \ldots \tilde{\boldsymbol{M}}_{i-1}^J \tilde{\boldsymbol{M}}_i^J \tag{54}$$

$$\begin{aligned}
\boldsymbol{V}_i &= 2\tilde{\boldsymbol{M}}_i \dot{\boldsymbol{M}}_i \\
&= 2 \sum_{k=1}^{i} \left(\tilde{\boldsymbol{M}}_{k-1} \tilde{\boldsymbol{M}}_k^J \ldots \tilde{\boldsymbol{M}}_{i-1}^J \tilde{\boldsymbol{M}}_i^J \right) \left(\boldsymbol{M}_i^J \boldsymbol{M}_{i-1}^J \ldots \dot{\boldsymbol{M}}_k^J \boldsymbol{M}_{k-1} \right) \\
&= \sum_{k=1}^{i} \dot{\theta}_k \tilde{\boldsymbol{M}}_{k-1} \boldsymbol{L}_k^0 \boldsymbol{M}_{k-1}.
\end{aligned} \tag{55}$$

According to equation (43), equation (55) is simplified to

$$\boldsymbol{V}_i = \sum_{k=1}^{i} \dot{\theta}_k \boldsymbol{L}_k. \tag{56}$$

$\boldsymbol{L}_k$ is the position of the line representing the axis of the joint k at time t. $\dot{\theta}_k$ is the rotation velocity for a revolute joint, or the translation velocity for a prismatic joint. This equation shows that the velocity of the link i only relates to the joints between itself and the base. It is a linear combination of the i lines in the sense of the PGA. The interpretation in the PGA is intuitive and much easier to understand than that with the screw theory or the POE method.

Besides, a recursion form to calculate the velocity is naturally induced.

$$\boldsymbol{V}_i = \boldsymbol{V}_{i-1} + \dot{\theta}_i \boldsymbol{L}_i. \tag{57}$$

Furthermore, continue taking the derivative on both sides of equation (57), the acceleration of the link i is also calculated in a recursion form:

$$\dot{\boldsymbol{V}}_i = \dot{\boldsymbol{V}}_{i-1} + \ddot{\theta}_i \boldsymbol{L}_i + \dot{\theta}_i \dot{\boldsymbol{L}}_i. \tag{58}$$

According to equation (43) and equation (46), the velocity of $\boldsymbol{L}_i$ is

$$\dot{\boldsymbol{L}}_i = [\boldsymbol{L}_i, \boldsymbol{V}_{i-1}]. \tag{59}$$

Then continue taking the derivative on both sides of equation (58), the high-order derivative of the link i's motion is formulated as follows:

$$\ddot{\boldsymbol{L}}_i = [\dot{\boldsymbol{L}}_i, \boldsymbol{V}_{i-1}] + [\boldsymbol{L}_i, \dot{\boldsymbol{V}}_{i-1}], \tag{60}$$

$$\ddot{\boldsymbol{V}}_i = \ddot{\boldsymbol{V}}_{i-1} + \dddot{\theta}_i \boldsymbol{L}_i + 2\ddot{\theta}_i \dot{\boldsymbol{L}}_i + \dot{\theta}_i \ddot{\boldsymbol{L}}_i, \tag{61}$$

$$\dddot{\boldsymbol{L}}_i = [\ddot{\boldsymbol{L}}_i, \boldsymbol{V}_{i-1}] + 2[\dot{\boldsymbol{L}}_i, \dot{\boldsymbol{V}}_{i-1}] + [\boldsymbol{L}_i, \ddot{\boldsymbol{V}}_{i-1}], \tag{62}$$

$$\dddot{\boldsymbol{V}}_i = \dddot{\boldsymbol{V}}_{i-1} + \ddddot{\theta}_i \boldsymbol{L}_i + 3\dddot{\theta}_i \dot{\boldsymbol{L}}_i + 3\ddot{\theta}_i \ddot{\boldsymbol{L}}_i + \dot{\theta}_i \dddot{\boldsymbol{L}}_i. \tag{63}$$

In the equations, $\ddot{\square}_i$, $\dddot{\square}_i$, and $\ddddot{\square}_i$ are the 2nd, 3rd, and 4th derivative of $\square_i$, respectively. $\square$ can be $\boldsymbol{L}$, $\boldsymbol{V}$, or θ.

The high-order recursive forward kinematics of serial robots algorithm is then summarized in Algorithm 1. The motor $\boldsymbol{M}_i^N$ indicates the motion from the reference frame to the body-fixed frame where the inertia of the rigid body is measured. The motor is used in the dynamics algorithm.

Algorithm 1 The high-order recursive forward kinematics

Require: $q_i,\ \dot{q}_i,\ \ddot{q}_i,\ \dddot{q}_i,\ \ddddot{q}_i,\ \boldsymbol{L}_i^0,\ \boldsymbol{M}_i^0,\ i = 1, 2, \ldots, n$

Forward propagation:
for $i = 1, \ldots n$ **do**
  $\boldsymbol{M}_i = e^{\frac{q_i}{2}\boldsymbol{L}_i^0}\boldsymbol{M}_{i-1}$
  $\boldsymbol{M}_i^N = \boldsymbol{M}_i^0 \boldsymbol{M}_i$
  $\boldsymbol{L}_i = \tilde{\boldsymbol{M}}_{i-1}\boldsymbol{L}_i^0\boldsymbol{M}_{i-1}$
  $\boldsymbol{V}_i = \boldsymbol{V}_{i-1} + \dot{q}_i \boldsymbol{L}_i$
  $\dot{\boldsymbol{L}}_i = [\boldsymbol{L}_i, \boldsymbol{V}_{i-1}]$
  $\dot{\boldsymbol{V}}_i = \dot{\boldsymbol{V}}_{i-1} + \ddot{q}_i \boldsymbol{L}_i + \dot{q}_i \dot{\boldsymbol{L}}_i$
  $\ddot{\boldsymbol{L}}_i = [\dot{\boldsymbol{L}}_i, \boldsymbol{V}_{i-1}] + [\boldsymbol{L}_i, \dot{\boldsymbol{V}}_{i-1}]$
  $\ddot{\boldsymbol{V}}_i = \ddot{\boldsymbol{V}}_{i-1} + \dddot{q}_i \boldsymbol{L}_i + 2\ddot{q}_i \dot{\boldsymbol{L}}_i + \dot{q}_i \ddot{\boldsymbol{L}}_i$
  $\dddot{\boldsymbol{L}}_i = [\ddot{\boldsymbol{L}}_i, \boldsymbol{V}_{i-1}] + 2[\dot{\boldsymbol{L}}_i, \dot{\boldsymbol{V}}_{i-1}] + [\boldsymbol{L}_i, \ddot{\boldsymbol{V}}_{i-1}]$
  $\dddot{\boldsymbol{V}}_i = \dddot{\boldsymbol{V}}_{i-1} + \ddddot{q}_i \boldsymbol{L}_i + 3\dddot{q}_i \dot{\boldsymbol{L}}_i + 3\ddot{q}_i \ddot{\boldsymbol{L}}_i + \dot{q}_i \dddot{\boldsymbol{L}}_i$
end for

Ensure: $\boldsymbol{M}_i^N,\ \boldsymbol{V}_i,\ \dot{\boldsymbol{V}}_i,\ \ddot{\boldsymbol{V}}_i,\ \dddot{\boldsymbol{V}}_i,\ i = 1, 2, \ldots, n$

3.3 Robot dynamics model

According to Newton's second law, force equals the first-order derivative of momentum. The momentum of a rigid body describes the intensity of motion considering its inertia. It includes the linear momentum and the angular momentum in classical mechanics. Correspondingly, "force" includes the force and the moment. In the screw theory, a screw describing both the force and the moment acting on a rigid body is called a wrench. This concept is applied in this article for the same purpose.

In the PGA, a mass particle is modeled as a Euclidean point $\boldsymbol{P}$ with a weight m, i.e., $\boldsymbol{X} = m\boldsymbol{P}$. $\boldsymbol{P}$ is represented as shown in Table 1. When the particle is attached to a rigid body, the velocity of $\boldsymbol{X}$ is $\dot{\boldsymbol{P}}$, which equals $[\boldsymbol{P}, \boldsymbol{V}]$. In view of projective geometry, the velocity is an infinite point weighted by the 2-norm of the velocity vector. The momentum of the mass particle is then modeled as the joining line of $\boldsymbol{X}$ and $\dot{\boldsymbol{P}}$. It is modeled in the PGA as

$$\begin{aligned} \boldsymbol{\mathcal{P}} &:= J[J(m\boldsymbol{P}) \wedge J(\dot{\boldsymbol{P}})] \\ &:= m\boldsymbol{P} \vee \dot{\boldsymbol{P}}. \end{aligned} \tag{64}$$

The operator J is defined in equation (19), and $\vee$ is the operator defined as $\boldsymbol{a} \vee \boldsymbol{b} = J(J(\boldsymbol{a}) \wedge J(\boldsymbol{b}))$, $\forall \boldsymbol{a}, \boldsymbol{b} \in \mathrm{P}(\mathbb{R}_{(3,0,1)})$. The operator implements "join" in contrast to "$\wedge$" implementing "meet".

For a rigid body, the momentum of the rigid body is defined as the integral of all mass particles contained in it:

$$\boldsymbol{\mathcal{P}} = \int_{\Omega} (\rho \boldsymbol{P} \vee [\boldsymbol{P}, \boldsymbol{V}])\, \mathrm{d}\boldsymbol{P}. \tag{65}$$

Ω is the set containing all mass particles in the rigid body and ρ is mass density, a function about $\boldsymbol{P}$.

It is obvious that $\boldsymbol{\mathcal{P}}$ is linear about $\boldsymbol{V}$. Therefore, a linear function $N[\cdot]$ can be defined as

$$N[\cdot] := \int_{\Omega} (\rho \boldsymbol{P} \vee [\boldsymbol{P}, \cdot])\, \mathrm{d}\boldsymbol{P}. \tag{66}$$

This linear function is named the general inertia because it serves the same as the 6×6 general inertia matrix in the POE. Actually, represent $\boldsymbol{V}$ and $\boldsymbol{\mathcal{P}}$ with their coordinate vectors $\underline{\mathbf{V}}$, $\underline{\boldsymbol{\mathcal{P}}}$:

$$\begin{aligned} \boldsymbol{V} &= [\boldsymbol{e}_{23} \quad \boldsymbol{e}_{31} \quad \boldsymbol{e}_{12} \quad \boldsymbol{e}_{01} \quad \boldsymbol{e}_{02} \quad \boldsymbol{e}_{03}]\, \underline{\mathbf{V}}, \\ \boldsymbol{\mathcal{P}} &= [\boldsymbol{e}_{23} \quad \boldsymbol{e}_{31} \quad \boldsymbol{e}_{12} \quad \boldsymbol{e}_{01} \quad \boldsymbol{e}_{02} \quad \boldsymbol{e}_{03}]\, \underline{\boldsymbol{\mathcal{P}}}. \end{aligned} \tag{67}$$

The general inertia is then represented as a 6×6 matrix

$$\begin{aligned} \underline{\mathbf{N}} &= \int_{\Omega} \begin{bmatrix} 0 & z & -y & 1 & 0 & 0 \\ -z & 0 & x & 0 & 1 & 0 \\ y & -x & 0 & 0 & 0 & 1 \\ y^2+z^2 & -xy & -xz & 0 & -z & y \\ -xy & x^2+z^2 & -yz & z & 0 & -x \\ -xz & -yz & x^2+y^2 & -y & x & 0 \end{bmatrix} \rho\, \mathrm{d}\boldsymbol{P} \\ &:= \begin{bmatrix} m[\boldsymbol{c}]^T & m\boldsymbol{I}_3 \\ \boldsymbol{\mathcal{J}} & m[\boldsymbol{c}] \end{bmatrix}. \end{aligned} \tag{68}$$

m is the mass of the rigid body and $\boldsymbol{I}_3$ is a 3×3 identity matrix. $\boldsymbol{c}$ is the position of the center of mass with respect to the reference coordinate system, and $[\boldsymbol{c}]$ is the screw symmetric matrix generated from it. $\mathcal{J}$ is the inertia matrix with respect to the reference coordinates system. These concepts are consistent with those in classical mechanics.

Remark If the coordinates of $J(\mathcal{P})$ instead of $\mathcal{P}$'s are calculated, the matrix corresponding to $J(\boldsymbol{N}[\cdot])$ is the same 6×6 inertia matrix as that in the POE.

$$J(\underline{\mathbf{N}}) := \begin{bmatrix} \mathcal{J} & m[\boldsymbol{c}] \\ m[\boldsymbol{c}]^T & m\boldsymbol{I}_3 \end{bmatrix}. \tag{69}$$

It implies the relation between the POE-based method and the PGA-based method.

The momentum of the rigid body is then formulated as

$$\mathcal{P} = N[\boldsymbol{V}]. \tag{70}$$

Suppose that the general inertia of the rigid body at the initial configuration is $N_b[\cdot]$ and its corresponding matrix $\underline{\mathbf{N}}_b$ is known. Then the momentum of the rigid body after a rigid motion $\boldsymbol{M}$ with the velocity $\boldsymbol{V}$ is

$$\begin{aligned} \mathcal{P} &= \int_\Omega \left[\rho(\tilde{\boldsymbol{M}}\boldsymbol{P}^0\boldsymbol{M}) \vee [(\tilde{\boldsymbol{M}}\boldsymbol{P}^0\boldsymbol{M}), \boldsymbol{V}]\right] \mathrm{d}(\tilde{\boldsymbol{M}}\boldsymbol{P}^0\boldsymbol{M}) \\ &= \tilde{\boldsymbol{M}}\left[\int_\Omega \left(\rho\boldsymbol{P}^0 \vee [\boldsymbol{P}^0, \boldsymbol{M}\boldsymbol{V}\tilde{\boldsymbol{M}}]\right) \mathrm{d}\boldsymbol{P}^0\right]\boldsymbol{M} \\ &= \tilde{\boldsymbol{M}} N_b[\boldsymbol{M}\boldsymbol{V}\tilde{\boldsymbol{M}}]\boldsymbol{M}. \end{aligned} \tag{71}$$

$\boldsymbol{P}^0$ is the initial position of $\boldsymbol{P}$. Denote $\boldsymbol{M}\boldsymbol{V}\tilde{\boldsymbol{M}}$ as $\boldsymbol{V}^b$ and $N_b[\boldsymbol{V}^b]$ as $\mathcal{P}_b$, i.e., the rigid body velocity and the momentum expressed in the body-fixed coordinate frame, respectively. Then equation (71) is simplified as

$$\mathcal{P} = \tilde{\boldsymbol{M}}\mathcal{P}^b\boldsymbol{M}, \tag{72}$$

$$\mathcal{P}^b = N_b[\boldsymbol{V}^b], \tag{73}$$

$$\boldsymbol{V}^b = \boldsymbol{M}\boldsymbol{V}\tilde{\boldsymbol{M}}. \tag{74}$$

With these results, the derivative of the momentum is formulated. First, take the derivative on both sides of equation (74) according to equation (50), and the following equations are derived:

$$\dot{\boldsymbol{V}}^b = \boldsymbol{M}\dot{\boldsymbol{V}}\tilde{\boldsymbol{M}} - \left[\boldsymbol{V}^b, \boldsymbol{V}^b\right] = \boldsymbol{M}\dot{\boldsymbol{V}}\tilde{\boldsymbol{M}}, \tag{75}$$

$$\ddot{\boldsymbol{V}}^b = \boldsymbol{M}\ddot{\boldsymbol{V}}\tilde{\boldsymbol{M}} - \left[\dot{\boldsymbol{V}}^b, \boldsymbol{V}^b\right], \tag{76}$$

$$\dddot{\boldsymbol{V}}^b = \boldsymbol{M}\dddot{\boldsymbol{V}}\tilde{\boldsymbol{M}} - \left[\boldsymbol{M}\ddot{\boldsymbol{V}}\tilde{\boldsymbol{M}} + \ddot{\boldsymbol{V}}^b, \boldsymbol{V}^b\right]. \tag{77}$$

Then, take the derivative on both sides of equation (73):

$$\dot{\mathcal{P}}^b = N_b[\dot{\boldsymbol{V}}^b], \tag{78}$$

$$\ddot{\boldsymbol{\mathcal{P}}}^b = N_b[\ddot{\boldsymbol{V}}^b], \tag{79}$$

$$\dddot{\boldsymbol{\mathcal{P}}}^b = N_b[\dddot{\boldsymbol{V}}^b]. \tag{80}$$

Because $N_b[\cdot]$ is the inertia at the initial configuration, it remains constant in the calculation.

Finally, take the derivative on both sides of equation (72):

$$\dot{\boldsymbol{\mathcal{P}}} = \tilde{\boldsymbol{M}}\dot{\boldsymbol{\mathcal{P}}}^b\boldsymbol{M} + [\boldsymbol{\mathcal{P}}, \boldsymbol{V}], \tag{81}$$

$$\ddot{\boldsymbol{\mathcal{P}}} = \tilde{\boldsymbol{M}}\ddot{\boldsymbol{\mathcal{P}}}^b\boldsymbol{M} + \left[\tilde{\boldsymbol{M}}\dot{\boldsymbol{\mathcal{P}}}^b\boldsymbol{M} + \dot{\boldsymbol{\mathcal{P}}}, \boldsymbol{V}\right] + \left[\boldsymbol{\mathcal{P}}, \dot{\boldsymbol{V}}\right], \tag{82}$$

$$\begin{aligned}\dddot{\boldsymbol{\mathcal{P}}} =& \tilde{\boldsymbol{M}}\dddot{\boldsymbol{\mathcal{P}}}^b\boldsymbol{M} + \left[2\tilde{\boldsymbol{M}}\ddot{\boldsymbol{\mathcal{P}}}^b\boldsymbol{M} + \ddot{\boldsymbol{\mathcal{P}}} + \left[\tilde{\boldsymbol{M}}\dot{\boldsymbol{\mathcal{P}}}^b\boldsymbol{M}, \boldsymbol{V}\right], \boldsymbol{V}\right] \\ &+ \left[\tilde{\boldsymbol{M}}\dot{\boldsymbol{\mathcal{P}}}^b\boldsymbol{M} + 2\dot{\boldsymbol{\mathcal{P}}}, \dot{\boldsymbol{V}}\right] + \left[\boldsymbol{\mathcal{P}}, \ddot{\boldsymbol{V}}\right].\end{aligned} \tag{83}$$

For a force $\boldsymbol{f}$ acting on a mass particle $m\boldsymbol{P}$, the position it acts on, the direction, and the intensity all matter. In classical mechanics, $\boldsymbol{f}$ is modeled as a free vector, while in the PGA, the force's direction $\boldsymbol{f}$ is modeled as a 3-vector, like the velocity of a mass particle, i.e., a weighted infinite point. The force is then modeled as $\boldsymbol{P} \vee \boldsymbol{f}$, a 2-vector representing a weighted line. From the perspective of force's geometry, translating the force along the line determined by its direction and its acting point has no effect on its action, so it is rational to model it in this way.

As for a rigid body, the integral of all forces acting on it is calculated, and a 2-vector that may be nonsimple is derived. Define the 2-vector as a wrench acting on the rigid body.

$$\boldsymbol{w} = \int_\Omega (\boldsymbol{P} \vee \boldsymbol{f})\,\mathrm{d}\boldsymbol{P}. \tag{84}$$

Because any nonsimple 2-vector can be decomposed as the sum of two simple vectors, the wrench is decomposed as

$$\boldsymbol{w} = \alpha\boldsymbol{L} + \beta\tilde{\boldsymbol{L}}\boldsymbol{I}\ (\alpha, \beta \in \mathbb{R}). \tag{85}$$

$\boldsymbol{L}$ is a normalized line and $\tilde{\boldsymbol{L}}\boldsymbol{I}$ is an infinite line pointing in the same direction as $\boldsymbol{L}$. Define the weighted line $\alpha\boldsymbol{L}$ as the force acting on the rigid body and the weighted infinite line $\beta\tilde{\boldsymbol{L}}\boldsymbol{I}$ as the moment acting on it,

$$\boldsymbol{w} = \boldsymbol{\mathcal{F}} + \boldsymbol{M}. \tag{86}$$

It implies that forces and moments acting on a rigid body are simplified to a total force and a total moment that point to the same direction. The result is consistent with that in classical mechanics.

According to the Newton's second law, the force acting on a particle equals the first-order derivative of the particle's momentum. In the PGA, the law is formulated as

$$\sum_k \boldsymbol{w}_k = \dot{\boldsymbol{\mathcal{P}}}. \tag{87}$$

w_k is a wrench acting on the rigid body, and it has the form

$$\boldsymbol{w}_k = \begin{cases} \alpha\boldsymbol{L}, & \text{a pure force,} \\ \beta\tilde{\boldsymbol{L}}\boldsymbol{I}, & \text{a pure moment,} \\ \alpha\boldsymbol{L} + \beta\tilde{\boldsymbol{L}}\boldsymbol{I} & \text{an ordinary wrench.} \end{cases} \tag{88}$$

For a serial robot, wrenches act on a link i is composed of the wrench from the joint i, the wrench from the joint $i+1$ (0 if the joint $i+1$ does not exist), and other wrenches from the environment. Denote the wrench from the joint i acting on the link i as $\boldsymbol{w}_i$ and the wrench from the environment acting on the link i as $\boldsymbol{w}_i^a$. Then the dynamic equation of the link i is

$$\boldsymbol{w}_i - \boldsymbol{w}_{i+1} + \boldsymbol{w}_i^a = \dot{\boldsymbol{\mathcal{P}}}_i. \tag{89}$$

In nonconstrained dynamics problem, $\boldsymbol{w}_i^a = 0$. Then the wrench $\boldsymbol{w}_i$ is calculated recursively:

$$\boldsymbol{w}_i = \dot{\boldsymbol{\mathcal{P}}}_i + \boldsymbol{w}_{i+1}. \tag{90}$$

Take the derivative on both sides of equation (90), and the derivative of the wrench $\boldsymbol{w}_i$ is derived:

$$\dot{\boldsymbol{w}}_i = \ddot{\boldsymbol{\mathcal{P}}}_i + \dot{\boldsymbol{w}}_{i+1}, \tag{91}$$

$$\ddot{\boldsymbol{w}}_i = \dddot{\boldsymbol{\mathcal{P}}}_i + \ddot{\boldsymbol{w}}_{i+1}. \tag{92}$$

In mechanics, power is defined by a bilinear function which maps force and velocity into a real number: $p :=< f, v >$. In the PGA, force is modeled as a 2-vector named wrench, and the velocity of a rigid body is also modeled as a 2-vector but named twist. The bilinear function defined in equation (18) is used to define the power caused by the wrench,

$$p :=< \boldsymbol{w}, \boldsymbol{V} >= \Delta(\boldsymbol{w} \wedge \boldsymbol{V}). \tag{93}$$

Denote the torque driving a revolute joint i as τ_i and the force driving a prismatic joint i as f_i. Then the power generated in the joint i can be formulated as follows:

$$p_i = \tau_i \dot{q}_i = \Delta(\boldsymbol{w}_i \wedge \dot{q}_i \boldsymbol{L}_i), \tag{94a}$$

$$p_i = f_i \dot{q}_i = \Delta(\boldsymbol{w}_i \wedge \dot{q}_i \tilde{\boldsymbol{L}}_i \boldsymbol{I}). \tag{94b}$$

Thus the actuation in joint i is formulated as

$$\tau_i = \Delta(\boldsymbol{w}_i \wedge \boldsymbol{L}_i), \tag{95a}$$

$$f_i = \Delta(\boldsymbol{w}_i \wedge \tilde{\boldsymbol{L}}_i \boldsymbol{I}). \tag{95b}$$

Summarize both equations as

$$\tau_i = \boldsymbol{L}_i * w_i. \tag{96}$$

Then, take the derivative on both sides of equation (96)

$$\dot{\tau}_i = \dot{\boldsymbol{L}}_i * w_i + \boldsymbol{L}_i * \dot{w}_i, \tag{97}$$

$$\ddot{\tau}_i = \ddot{\boldsymbol{L}}_i * w_i + 2\dot{\boldsymbol{L}}_i * \dot{w}_i + \boldsymbol{L}_i * \ddot{w}_i. \tag{98}$$

The first derivatives of the joint torques or forces are calculated. The algorithm is then summarized in Algorithm 2.

Algorithm 2 The second-order recursive inverse dynamics

Require: $\boldsymbol{M}_i^N,\ \boldsymbol{V}_i,\ \dot{\boldsymbol{V}}_i,\ \ddot{\boldsymbol{V}}_i,\ \dddot{\boldsymbol{V}}_i,\ \underline{\mathrm{N}}_i^b,\ i = 1, 2, \ldots, n$

Backward propagation
for $i = n, \ldots, 1$ **do**
$\quad \boldsymbol{V}_i^b = \boldsymbol{M}_i^N \boldsymbol{V}_i \tilde{\boldsymbol{M}}_i^N$
$\quad \dot{\boldsymbol{V}}_i^b = \boldsymbol{M}_i^N \dot{\boldsymbol{V}}_i \tilde{\boldsymbol{M}}_i^N$
$\quad \ddot{\boldsymbol{V}}_i^b = \boldsymbol{M}_i^N \ddot{\boldsymbol{V}}_i \tilde{\boldsymbol{M}}_i^N - \left[\dot{\boldsymbol{V}}_i^b, \boldsymbol{V}_i^b\right]$
$\quad \dddot{\boldsymbol{V}}_i^b = \boldsymbol{M}_i^N \dddot{\boldsymbol{V}}_i \tilde{\boldsymbol{M}}_i^N - \left[\boldsymbol{M}_i^N \ddot{\boldsymbol{V}}_i \tilde{\boldsymbol{M}}_i^N + \ddot{\boldsymbol{V}}_i^b, \boldsymbol{V}_i^b\right]$

$\quad \boldsymbol{\mathcal{P}}_i^b = N_{bi}[\boldsymbol{V}^b] = \underline{\mathrm{N}}_i^b \underline{\boldsymbol{V}}_i^b$
$\quad \dot{\boldsymbol{\mathcal{P}}}_i^b = N_{bi}[\dot{\boldsymbol{V}}^b] = \underline{\mathrm{N}}_i^b \underline{\dot{\boldsymbol{V}}}_i^b$
$\quad \ddot{\boldsymbol{\mathcal{P}}}_i^b = N_{bi}[\ddot{\boldsymbol{V}}^b] = \underline{\mathrm{N}}_i^b \underline{\ddot{\boldsymbol{V}}}_i^b$
$\quad \dddot{\boldsymbol{\mathcal{P}}}_i^b = N_{bi}[\dddot{\boldsymbol{V}}^b] = \underline{\mathrm{N}}_i^b \underline{\dddot{\boldsymbol{V}}}_i^b$

$\quad \boldsymbol{\mathcal{P}}_i = \tilde{\boldsymbol{M}}_i^N \boldsymbol{\mathcal{P}}_i^b \boldsymbol{M}_i^N$
$\quad \dot{\boldsymbol{\mathcal{P}}}_i = \tilde{\boldsymbol{M}}_i^N \dot{\boldsymbol{\mathcal{P}}}_i^b \boldsymbol{M}_i^N + [\boldsymbol{\mathcal{P}}_i, \boldsymbol{V}_i]$
$\quad \ddot{\boldsymbol{\mathcal{P}}}_i = \tilde{\boldsymbol{M}}_i^N \ddot{\boldsymbol{\mathcal{P}}}_i^b \boldsymbol{M}_i^N + \left[\tilde{\boldsymbol{M}}_i^N \dot{\boldsymbol{\mathcal{P}}}_i^b \boldsymbol{M}_i^N + \dot{\boldsymbol{\mathcal{P}}}_i, \boldsymbol{V}_i\right] + \left[\boldsymbol{\mathcal{P}}_i, \dot{\boldsymbol{V}}_i\right]$
$\quad \dddot{\boldsymbol{\mathcal{P}}}_i = \tilde{\boldsymbol{M}}_i^N \dddot{\boldsymbol{\mathcal{P}}}_i^b \boldsymbol{M}_i^N + \left[\tilde{\boldsymbol{M}}_i^N \ddot{\boldsymbol{\mathcal{P}}}_i^b \boldsymbol{M}_i^N + 2\dot{\boldsymbol{\mathcal{P}}}_i, \dot{\boldsymbol{V}}_i\right] + \left[\boldsymbol{\mathcal{P}}_i, \ddot{\boldsymbol{V}}_i\right]$
$\qquad + \left[2\tilde{\boldsymbol{M}}_i^N \ddot{\boldsymbol{\mathcal{P}}}_i^b \boldsymbol{M}_i^N + \ddot{\boldsymbol{\mathcal{P}}}_i + \left[\tilde{\boldsymbol{M}}_i^N \dot{\boldsymbol{\mathcal{P}}}_i^b \boldsymbol{M}_i^N, \boldsymbol{V}_i\right], \boldsymbol{V}_i\right]$

$\quad \boldsymbol{w}_i = \dot{\boldsymbol{\mathcal{P}}}_i + \boldsymbol{w}_{i+1}$
$\quad \dot{\boldsymbol{w}}_i = \ddot{\boldsymbol{\mathcal{P}}}_i + \dot{\boldsymbol{w}}_{i+1}$
$\quad \ddot{\boldsymbol{w}}_i = \dddot{\boldsymbol{\mathcal{P}}}_i + \ddot{\boldsymbol{w}}_{i+1}$

$\quad \tau_i = \boldsymbol{L}_i * \boldsymbol{w}_i$
$\quad \dot{\tau}_i = \dot{\boldsymbol{L}}_i * \boldsymbol{w}_i + \boldsymbol{L}_i * \dot{\boldsymbol{w}}_i$
$\quad \ddot{\tau}_i = \ddot{\boldsymbol{L}}_i * \boldsymbol{w}_i + 2\dot{\boldsymbol{L}}_i * \dot{\boldsymbol{w}}_i + \boldsymbol{L}_i * \ddot{\boldsymbol{w}}_i$
end for

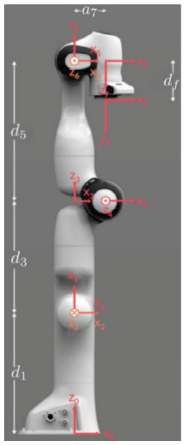
Ensure: $\tau_i,\ \dot{\tau}_i,\ \ddot{\tau}_i,\ i = 1, 2, \ldots, n$

4 Validation and comparison

4.1 Validation: Franka Emika Panda

The Franka Emika Panda, a redundant 7-DOF robotic arm, is used to demonstrate the algorithm proposed in this paper. Its zero reference configuration and the modified Denavit–Hartenberg parameters are shown in Fig. 1(a) and Fig. 1(b), respectively. The axis of a joint, the z axis of the body-fixed frame, is modeled as a line in the PGA. The lines at their initial configuration, i.e., $z_1, z_2, \ldots, z_7$, are shown in Fig. 1(a). The lines at the initial positions are represented as follows:

$$\boldsymbol{L}_1^0 = \boldsymbol{e}_{12},$$
$$\boldsymbol{L}_2^0 = \boldsymbol{e}_{31} - 0.333\ \boldsymbol{e}_{01},$$



(a)

i	a_i	α_i	d_i	θ_i
1	0	0	d_1	q_1
2	0	$-\pi/2$	0	q_2
3	0	$\pi/2$	d_3	q_3
4	a_4	$\pi/2$	0	q_4
5	a_5	$-\pi/2$	d_5	q_5
6	0	$\pi/2$	0	q_6
7	a_7	$\pi/2$	0	q_7
8	0	0	d_f	0

(b)

Fig. 1 (a) The zero reference configuration of the Franka Emika Panda and Denavit–Hartenberg frames of each link [31]. (b). The modified Denavit–Hartenberg parameters [31]. $d_1 = 0.333$ m, $d_3 = 0.316$ m, $d_5 = 0.384$ m, $d_f = 0.107$ m, $a_4 = 0.0825$ m, $a_5 = -0.0825$ m, $a_7 = 0.088$ m

$$
\begin{aligned}
\boldsymbol{L}_3^0 &= \boldsymbol{e}_{12}, \\
\boldsymbol{L}_4^0 &= -\boldsymbol{e}_{31} + 0.649\,\boldsymbol{e}_{01} - 0.0825\,\boldsymbol{e}_{03}, \\
\boldsymbol{L}_5^0 &= \boldsymbol{e}_{12}, \\
\boldsymbol{L}_6^0 &= -\boldsymbol{e}_{31} + 1.033\,\boldsymbol{e}_{01}, \\
\boldsymbol{L}_7^0 &= -\boldsymbol{e}_{12} + 0.088\,\boldsymbol{e}_{02}.
\end{aligned}
$$

The body-fixed reference frames are then defined, as indicated in Fig. 1(a). The dynamic parameters are measured according to these frames. The motors representing the motion from the spatial reference frame to the initial body-fixed reference frame are then calculated according to the D-H parameters $\{\alpha, a, \theta, d\}$. In the initial configuration, $\theta_i = 0$. Then the equation to calculate the motor is

$$
\boldsymbol{M}_i^0 = e^{\frac{\alpha_i}{2}\boldsymbol{e}_{23} + \frac{a_i}{2}\boldsymbol{e}_{01}} e^{\frac{d_i}{2}\boldsymbol{e}_{03}}. \tag{99}
$$

The initial motors of the Franka Emika Panda are as follows:

$$
\begin{aligned}
\boldsymbol{M}_1^0 &= 1 + 0.1665\,\boldsymbol{e}_{03}, \\
\boldsymbol{M}_2^0 &= 0.7071 - 0.7071\,\boldsymbol{e}_{23} - 0.1177\,\boldsymbol{e}_{02} + 0.1177\,\boldsymbol{e}_{03}, \\
\boldsymbol{M}_3^0 &= 1 + 0.3245\,\boldsymbol{e}_{03}, \\
\boldsymbol{M}_4^0 &= 0.7071 + 0.7071\,\boldsymbol{e}_{23} + 0.0292\,\boldsymbol{e}_{01} \\
&\quad + 0.2295\,\boldsymbol{e}_{02} + 0.2295\,\boldsymbol{e}_{03} + 0.0292\,\boldsymbol{I}, \\
\boldsymbol{M}_5^0 &= 1 + 0.5165\,\boldsymbol{e}_{03}, \\
\boldsymbol{M}_6^0 &= 0.7071 + 0.7071\,\boldsymbol{e}_{23} + 0.3652\,\boldsymbol{e}_{02} + 0.3652\,\boldsymbol{e}_{03}, \\
\boldsymbol{M}_7^0 &= \boldsymbol{e}_{23} + 0.5165\,\boldsymbol{e}_{02} + 0.044\,\boldsymbol{I}.
\end{aligned}
$$

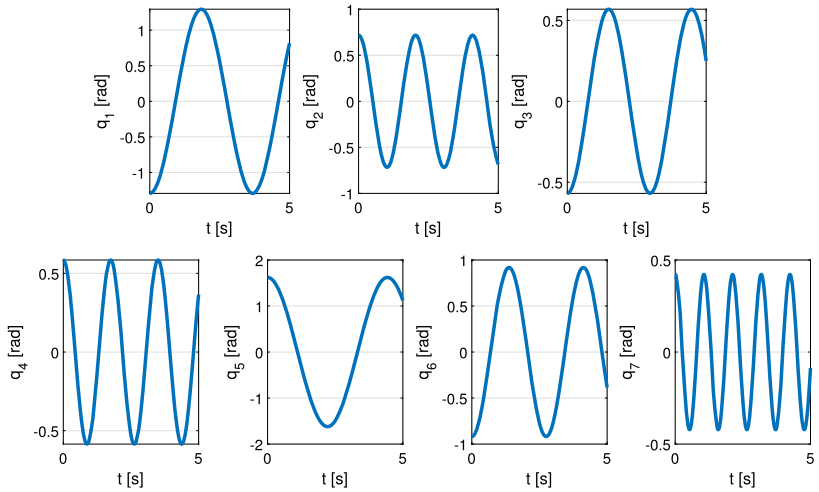


Fig. 2 The trajectory in the joint space used for the validation [2]

The lines and the motors are represented as arrays with six elements and eight elements in Matlab, respectively. For example, $\boldsymbol{L}_1^0 = [1; 0; 0; 0; 0; 0]$ and $\boldsymbol{M}_1^0 = [1; 0; 0; 0; 0; 0; 0.1665; 0]$. The operations involved in the algorithm include the reverse of a motor $\tilde{\boldsymbol{M}}$, the exponential map of a 2-vector $e^{\boldsymbol{L}}$, the Lie bracket of two 2-vectors $[\boldsymbol{X}, V]$, the rigid body motion motor of a 2-vector $\tilde{\boldsymbol{M}}\boldsymbol{X}\boldsymbol{M}$, and the geometric product of two motors $\boldsymbol{M}_1\boldsymbol{M}_2$. All the operations are implemented in Matlab as m-functions and the calculations are simplified by omitting irrelevant and repetitive operations with the help of the Symbolic Math Toolbox.

The motion data used for validation is provided in [2] and can be downloaded online.[1] The robot's trajectory in the joint space is shown in Fig. 2. The POE-based algorithm proposed by Müller in [2] is used to calculate joint torques and the first-order and the second-order derivatives of torques. The results obtained by the PGA-based algorithm are compared with the results obtained by this POE-based algorithm.

The joint torques and their first-order and second-order derivatives obtained by the two algorithms are indicated in Fig. 3, Fig. 4, Fig. 5, respectively. 31 points are calculated by the PGA-based algorithm, and they are all identical to the results calculated by the POE-based method. Only the results of three joints are presented for brief figures. The results of other joints have also been demonstrated.

4.2 Efficiency comparison

For a preliminary comparison of the computational efficiency, the Matlab R2022a code is called for 10 000 evaluation. On a PC (i5-12500H, 2.50 GHz) running Windows 11, the PGA-based algorithm needed 2.415 s and the POE-based algorithm proposed in [2] 2.871 s, i.e., the PGA-based algorithm is 15.8% faster.

Furthermore, the computational time of the exponential map, the Lie brackets, and the rigid body motion operations are measured. Each of the three operations is called for 10 000

[1] The data file can be downloaded from https://ieeexplore.ieee.org/ielx7/7083369/9285111/9290369/lra-3044028-mm.zip?arnumber=9290369.

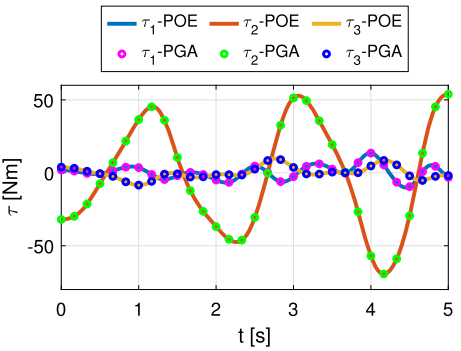


Fig. 3 The joint torques τ in the joint-1, joint-2, and joint-3

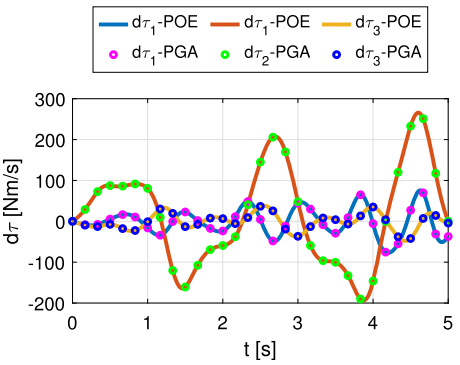


Fig. 4 The first derivatives of the joint torques $\dot{\tau}$ in the joint-1, joint-2, and joint-3

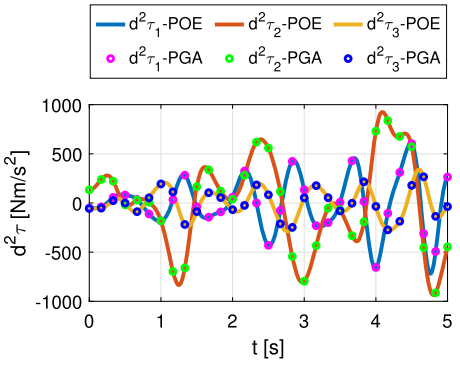


Fig. 5 The second derivatives of the joint torques $\ddot{\tau}$ in the joint-1, joint-2, and joint-3

evaluations using the PGA-based method and the POE-based method. In the POE method, the exponential map of a twist $\boldsymbol{s} = [\boldsymbol{\omega}^T, \boldsymbol{v}^T]^T$ is calculated by

$$\boldsymbol{R} = \boldsymbol{I}_3 + \sin\theta\ \hat{\boldsymbol{\omega}} + (1 - \sin\theta)\hat{\boldsymbol{\omega}}^2, \tag{100a}$$

$$\boldsymbol{p} = (\boldsymbol{I}_3 - \boldsymbol{R})\hat{\boldsymbol{\omega}}\boldsymbol{v} + (\boldsymbol{\omega}^T\boldsymbol{v}\theta)\boldsymbol{\omega}, \tag{100b}$$

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{p} \\ \boldsymbol{0}_{1\times 3} & 1 \end{bmatrix}. \tag{100c}$$

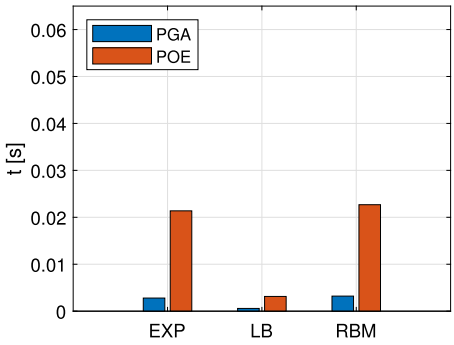


Fig. 6 The computational time of the exponential map (EXP), the Lie brackets (LB), and the rigid body motion operations (RBM)

Then the rigid body motion of the twist is

$$\boldsymbol{s} = \mathrm{Ad}_{\boldsymbol{T}}\boldsymbol{s}^0. \tag{101}$$

The Lie bracket is calculated by

$$[\boldsymbol{s}, \boldsymbol{V}] = \mathrm{ad}_{\boldsymbol{V}}\boldsymbol{s}. \tag{102}$$

The rigid body motion operations are composed of equations (100a)–(100c) and equation (101) in the POE method, and the corresponding operations in the PGA method are used for comparison. Both are simplified by omitting irrelevant and repetitive operations with the help of the Symbolic Math Toolbox. The results are shown in Fig. 6. It turns out that PGA does improve the computational efficiency of the three operations.

Furthermore, as shown in Algorithm 1 and Algorithm 2, the PGA-based algorithm is implemented by n forward iterations and n backward iterations, where n is the DOF of the serial robot. It implies that the time complexity of the proposed algorithm is $O(n)$, the same as the POE-based algorithm proposed in [2]. However, for the two algorithms, the coefficients of n are different, causing the running time different.

For a more detailed comparison, referring to the analysis method proposed on pages 294–306 of [32], the number of additions and multiplications in both algorithms for serial robots with revolute joints are counted. Firstly, key algebraic operations in both algorithms are listed in Table 3 with the number of multiplications and additions counted. Secondly, the number of operations in each algorithm is counted line by line according to the pseudo-code. Finally, the total number of multiplications and additions is calculated and listed in Table 4. It turns out that the exponential map and the Lie bracket in the PGA method saves plenty of multiplications and additions, while the rigid body motion operation costs more. However, the tedious memory operations in the POE method hold the efficiency back, causing more computational time than those in the PGA method. As for the algorithm, it turns out that the PGA method saves 69.82% multiplications and 73.58% additions in each kinematics propagation. However, the multiplications and additions cost in the dynamics propagation are at the same level as that in the POE method. By further analysis, we find that the implicit representations of the general inertia defined in equation (66) is the main obstacle to further computational improvement. The matrix representations of general inertia are used in this paper, and a complicated backward recursive algorithm is thus derived. The simplification of the representations of general inertia will be considered in future work.

Table 3 The computation cost of key operations in the algorithms (M for multiplications, and A for additions). Key operations include the exponential map (EXP), the Lie bracket (LB), the rigid body motion operation (RBM), the geometric product (GP), the 6th order matrix multiplication (6MM), the 4th order matrix multiplication (4MM), the product of a 6 by 6 matrix and a 6-dimensional vector (6MV). The symbol "—" means that the operation is not used in the algorithm

Operation	M (PGA)	M (POE)	A (PGA)	A (POE)
EXP	6	70	0	63
LB	18	36	12	30
RBM	186	106	138	93
GP	48	—	40	—
6MM	—	216	—	180
4MM	—	64	—	48
6MV	36	36	30	30

Table 4 The computation cost of the algorithms for an n-DOF serial robot with revolute joints (M for multiplications and A for additions)

Process	M (PGA)	M (POE)	A (PGA)	A (POE)
Kinematics propagation	459 n	1521 n	368 n	1393 n
Dynamics propagation	2700 n	2706 n	2166 n	2443 n
Total	3159 n	4227 n	2534 n	3836 n

5 Conclusion

In this article, the model of serial robots based on the projective geometric algebra $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$ is proposed and an efficient algorithm for high-order inverse dynamics is constructed. We provide a brief introduction to the basic concepts in PGA. Then we propose a method to construct the geometric model, the kinematics model, and the dynamics model of a serial robot totally based on the PGA. It turns out that the method based on the PGA possesses several advantages. It is computationally efficient, uniform, intuitive, and coordinate invariant. In the PGA, the exponential map and Lie brackets are both implemented without matrices. The homogeneous matrix and its adjoint matrix are replaced by a multivector. As a result, some repetitive operations and zero elements are saved and the efficiency is improved. All vectors and algebraic operations correspond to some geometric elements and geometric operations, which is helpful to understand the operations in an intuitive way. Besides, the velocity of any geometric objects attached to a rigid body is calculated in a uniform form by Lie brackets.

Actually, the PGA proposed in this paper is closely related to other concepts popular in robotics. The projective space $\mathbb{R}P^3$ is actually a Grassmann manifold. The geometric algebra $\mathrm{P}\left(\mathbb{R}_{(3,0,1)}\right)$ induces a Lie algebra where the Lie brackets are defined by the commutator based on the geometric product. The even-grade space is actually a subalgebra, and it is isomorphic to dual quaternions. The multivector $\boldsymbol{M}$ with odd grades and satisfying $\tilde{\boldsymbol{M}}\boldsymbol{M} = 1$ generates a Lie group. The exponential map maps a 2-vector to an element in this Lie group. It is consistent with the results obtained in the screw theory. If all vectors and linear transforms in the PGA-based method are written as coordinates and matrices relative to the basis, the formulations in the POE method are then obtained.

The application of the PGA is limited to serial robots, where the elasticity of links is omitted. The inertia of a rigid body is still in an implicit form in the presented formulation and repetitive elements and operations remain. The power of PGA requires to be explored in the future work.

Funding This work was partially supported by the National Natural Science Foundation of China (Grant Nos. 51935010, 51822506).

Declarations

Competing Interests The authors declare no competing interests.

References

1. Gattringer, H., Springer, K., Müller, A., Jörgl, M.: An efficient method for the dynamical modeling of serial elastic link/joint robots. In: Moreno-Díaz, R., Pichler, F., Quesada-Arencibia, A. (eds.) Computer Aided Systems Theory – EUROCAST 2015. Lecture Notes in Computer Science, pp. 689–697. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-27340-2_85
2. Müller, A.: An o(n)-algorithm for the higher-order kinematics and inverse dynamics of serial manipulators using spatial representation of twists. IEEE Robot. Autom. Lett. **6**(2), 397–404 (2021). https://doi.org/10.1109/LRA.2020.3044028
3. Oaki, J., Adachi, S.: Vibration suppression control of elastic-joint robot arm based on designless nonlinear state observer. IEEJ Trans. Ind. Appl. **135**(5), 571–581 (2015). https://doi.org/10.1541/ieejias.135.571
4. Bruzzone, L., Bozzini, G.: A statically balanced SCARA-like industrial manipulator with high energetic efficiency. Meccanica **46**(4), 771–784 (2011). https://doi.org/10.1007/s11012-010-9336-6
5. Feng, H., Xu, Y., Mao, D.: A novel adaptive balance-drive mechanism for industrial robots using a series elastic actuator. Int. J. Comput. Integr. Manuf. **33**(10–11), 991–1005 (2020). https://doi.org/10.1080/0951192X.2020.1736715
6. Denavit, J., Hartenberg, R.S.: A kinematic notation for lower-pair mechanisms based on matrices. J. Appl. Mech. **22**(2), 215–221 (1955). https://doi.org/10.1115/1.4011045
7. Lynch, K.M., Park, F.C.: Modern Robotics. Cambridge University Press, Cambridge (2017)
8. Chen, G., Kong, L., Li, Q., Wang, H., Lin, Z.: Complete, minimal and continuous error models for the kinematic calibration of parallel manipulators based on POE formula. Mech. Mach. Theory **121**, 844–856 (2018). https://doi.org/10.1016/j.mechmachtheory.2017.11.003
9. Selig, J.M.: Geometrical Foundations of Robotics. Springer, New York (2005). https://doi.org/10.1007/b138859
10. Müller, A., Kumar, S.: Closed-form time derivatives of the equations of motion of rigid body systems. Multibody Syst. Dyn. **53**(3), 257–273 (2021). https://doi.org/10.1007/s11044-021-09796-8
11. Müller, A.: An overview of formulae for the higher-order kinematics of lower-pair chains with applications in robotics and mechanism theory. Mech. Mach. Theory **142**, 103594 (2019). https://doi.org/10.1016/j.mechmachtheory.2019.103594
12. Singh, S., Russell, R.P., Wensing, P.M.: Efficient analytical derivatives of rigid-body dynamics using spatial vector algebra. IEEE Robot. Autom. Lett. **7**(2), 1776–1783 (2022). https://doi.org/10.1109/lra.2022.3141194
13. Cibicik, A., Egeland, O.: Kinematics and dynamics of flexible robotic manipulators using dual screws. IEEE Trans. Robot. **37**(1), 206–224 (2021). https://doi.org/10.1109/tro.2020.3014519
14. Silva, F.F.A., Quiroz-Omaña, J.J., Adorno, B.V.: Dynamics of mobile manipulators using dual quaternion algebra. J. Mech. Robot. **14**(6), 061005 (2022). https://doi.org/10.1115/1.4054320
15. Crowe, M.G., Williamson, R.E.: A History of Vector Analysis, vol. 37, p. 844 (1969). https://doi.org/10.1119/1.1975883
16. Hestenes, D.: Spacetime physics with geometric algebra. Am. J. Phys. **71**(7), 691–714 (2003). https://doi.org/10.1119/1.1571836
17. Doran, C., Gullans, S.R., Lasenby, A., Lasenby, J., Fitzgerald, W.: Geometric Algebra for Physicists. Cambridge University Press, Cambridge (2003). https://doi.org/10.1017/CBO9780511807497
18. Lopes, W.B., Lopes, C.G.: Geometric-algebra adaptive filters. IEEE Trans. Signal Process. **67**(14), 3649–3662 (2019). https://doi.org/10.1109/TSP.2019.2916028

19. Bayro-Corrochano, E., Zamora-Esquivel, J.: Differential and inverse kinematics of robot devices using conformal geometric algebra. Robotica **25**(1), 43–61 (2007). https://doi.org/10.1017/S0263574706002980
20. Bayro-Corrochano, E., Reyes-Lozano, L., Zamora-Esquivel, J.: Conformal geometric algebra for robotic vision. J. Math. Imaging Vis. **24**(1), 55–81 (2006). https://doi.org/10.1007/s10851-005-3615-1
21. Bayro-Corrochano, E.: Robot modeling and control using the motor algebra framework. In: 2019 12th International Workshop on Robot Motion and Control (RoMoCo), pp. 1–8 (2019). https://doi.org/10.1109/RoMoCo.2019.8787386
22. Selig, J.M., Bayro-Corrochano, E.: Rigid body dynamics using Clifford algebra. Adv. Appl. Clifford Algebras **20**(1), 141–154 (2008). https://doi.org/10.1007/s00006-008-0144-1
23. Hestenes, D., Fasse, E.D.: Homogeneous Rigid Body Mechanics with Elastic Coupling. Birkhäuser, Boston (2002). https://doi.org/10.1007/978-1-4612-0089-5_19.
24. Gunn, C.G.: Doing Euclidean plane geometry using projective geometric algebra. Adv. Appl. Clifford Algebras **27**(2), 1203–1232 (2017). https://doi.org/10.1007/s00006-016-0731-5
25. Gunn, C.: Geometric algebras for Euclidean geometry. Adv. Appl. Clifford Algebras **27**(1), 185–208 (2017). https://doi.org/10.1007/s00006-016-0647-0
26. Gunn, C.: Geometry, kinematics, and rigid body mechanics in Cayley-Klein geometries. Doctoral thesis, Technische Universität Berlin, Fakultät II - Mathematik und Naturwissenschaften, Berlin (2011). https://doi.org/10.14279/depositonce-3058
27. Hadfield, H., Lasenby, J.: Chap. 39. Constrained dynamics in conformal and projective geometric algebra. In: Computer Graphics International Conference. Lecture Notes in Computer Science, pp. 459–471 (2020). https://doi.org/10.1007/978-3-030-61864-3_39
28. Arnold, V.I.: Mathematical Methods of Classical Mechanics. Graduate Texts in Mathematics, vol. 60. Springer, New York (1978). https://doi.org/10.1007/978-1-4757-1693-1
29. Pottmann, H., Wallner, J.: Computational Line Geometry. Mathematics and Visualization. Springer, Berlin (2001). https://doi.org/10.1007/978-3-642-04018-4
30. Gunn, C.G.: Course notes geometric algebra for computer graphics, SIGGRAPH 2019. In: ACM SIGGRAPH 2019 Courses, pp. 1–140 (2019). https://doi.org/10.1145/3305366.3328099
31. Gaz, C., Cognetti, M., Oliva, A., Robuffo Giordano, P., De Luca, A.: Dynamic identification of the Franka Emika Panda robot with retrieval of feasible parameters using penalty-based optimization. IEEE Robot. Autom. Lett. **4**(4), 4147–4154 (2019). https://doi.org/10.1109/lra.2019.2931248
32. Angeles, J.: Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms. Mechanical Engineering Series, vol. 124. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-01851-5