mathematics

MDPI

Article

A Lie Group-Based Iterative Algorithm Framework for Numerically Solving Forward Kinematics of Gough–Stewart Platform

Binhai Xie [1,2,*], Shuling Dai [1,2] and Feng Liu [1,2]

[1] The State Key Laboratory of VR Technology & Systems, Beihang University, Beijing 100191, China; sldai@buaa.edu.cn (S.D.); by1503101@buaa.edu.cn (F.L.)
[2] Jiangxi Research Institute, Beihang University, Beijing 100191, China
* Correspondence: xiebinhai@buaa.edu.cn

Abstract: In this work, we began to take forward kinematics of the Gough–Stewart (G-S) platform as an unconstrained optimization problem on the Lie group-structured manifold SE(3) instead of simply relaxing its intrinsic orthogonal constraint when algorithms are updated on six-dimensional local flat Euclidean space or adding extra unit norm constraint when orientation parts are parametrized by a unit quaternion. With this thought in mind, we construct two kinds of iterative problem-solving algorithms (Gauss–Newton (G-N) and Levenberg–Marquardt (L-M)) with mathematical tools from the Lie group and Lie algebra. Finally, a case study for a general G-S platform was carried out to compare these two kinds of algorithms on SE(3) with corresponding algorithms that updated on six-dimensional flat Euclidean space or seven-dimensional quaternion-based parametrization Euclidean space. Experiment results demonstrate that those algorithms on SE(3) behave better than others in convergence performance especially when the initial guess selection is near to branch solutions.

Keywords: Gough–Stewart platform; forward kinematics; lie group; lie algebra; exponential map; Gauss–Newton; Levenberg–Marquardt

check for updates

Citation: Xie, B.; Dai, S.; Liu, F. A Lie Group-Based Iterative Algorithm framework for Numerically Solving Forward Kinematic of Gough–Stewart Platform. *Mathematics* **2021**, *9*, 757. https://doi.org/10.3390/math9070757

Academic Editor: Poul Hjorth

Received: 1 March 2021
Accepted: 29 March 2021
Published: 1 April 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The six degree-of-freedom Gough–Stewart (G-S) type motion platforms have been extensively employed to realize fully operational flight training simulation with high fidelity. This spatial G-S platform had six identical extensible active legs connected with an upper moving platform and a fixed lower platform using six upper passive joints and six lower passive joints. The forward kinematics of the G-S platform allows the platform to configure the pose (position and orientation) of end-effectors on the moving platform when the lengths of the six linear actuators are provided. However, the closed loop kinematic relations between the moving platform and the fixed base platform in 3-D space made forward kinematic problems a challenging issue among all kinematic research of parallel manipulators during the past few decades.

A significant amount of research has been focused on the forward kinematics of the G-S platform, which has lead us to understand the difficulty of this problem. Generally, there exist three main research directions to address the forward kinematics problem, which are the analytic method, numerical iterative method, and auxiliary-sensor-based method. The majority of early research focused on finding all possible closed-form direct solutions for different types of G-S platforms. Since Raghavan [1] and Ronga [2] firstly proved that there existed at most 40 distinct solutions in the complex domain for a general G-S platform, many researchers succeed in determining all 40 direct solutions using algebraic elimination, interval analysis, or numerical continuation methods, etc. Husty [3] obtained a 40-th degree univariate polynomial through complicated algebraic elimination based on a spatial kinematic mapping. For a symmetrical 6-6 G-S platform, the degree of polynomial was ultimately reduced to 14 by Huang [4]. Researcher Ji [5] focused on a special type 6-6 G-S platform in which both the fixed base platform and the moving platform are similar hexagons and only univariate quadratic equations are required to derive eight possible symmetry solutions by introducing a quaternion to represent the orientation matrix. As a general analytic form of direction solutions to forward kinematics of a general G-S platform was luxurious, some scholars resorted to the numerical root-finding method. Both Didrit [6] and Merlet [7] preferred to utilize the interval analysis method to numerically solve the forward kinematic problem for the general G-S platform. Wampler [8] implemented a polynomial continuation scheme to deal with this problem. Besides that, Dietmaier [9] proposed a numerical procedure to change the parameters of the G-S platform and ultimately obtained a particular asymmetry for the G-S platform, which possesses 40 real direct solutions.

As there exist multiple direct solutions, research attempting to solve the forward kinematic problem is further expected to determine a unique actual pose (position and orientation) of the G-S platform. Numerical iterative methods and auxiliary-sensor-based methods are two common schemes to pursue this goal. As the forward kinematic problem was easily boiled down to seeking six dimensional variables, satisfying six dimensional kinematic equations, Newton-type iterative algorithms were extensively put into use because of its simplicity and fast convergence speed [10,11]. However, it had been well known that the Newton-type algorithm usually converges to a particular solution near to initial guess value. Therefore, some variants of the Newton-type algorithm had been discussed to improve its performance in the following studies. Yang [12] modified the traditional Newton–Raphson algorithm using a monotonic descent operator to achieve global convergence regardless of the initial guess selection. Pratik [13] proposed a neural-network-based hybrid strategy that combined with the standard Newton–Raphson algorithm to yield better performance in dealing with uniqueness problem. Furthermore, Innocenti [14], Parenti-Castelli [15], and Chiu [16] suggested adding one or more redundant sensors to determine a unique real solution. Wang [17] designed a incremental iterative algorithm through a series of continuous small changes in leg length to derive a unique forward kinematic solution.

When iterative algorithms are employed to solve optimization that involves the SE(3) configure pose, the problem of which parametrization scheme is beneficial to design its updating framework must be addressed. As we all know, 3D+YPR and 3D+Quat are two common parametrization schemes, as explained in Reference [18]. In the 3D+YPR scheme, the iterative algorithm can be designed on flat Euclidean space $\mathbb{R}^6$ with orthogonal constraints not taken into account. In the 3D+Quat scheme, a unit norm constraint equation is added while state vector is updated on enlarged Euclidean space $\mathbb{R}^7$. In this work, we address the forward kinematic as an unconstrained optimization problem on a Lie group-structured manifold SE(3). In algorithm construction, the updating direction and step size are computed on the associated tangent space $T_eSE(3)$ with differential properties of the Lie group and Lie algebra. The result is then projected back on the manifold SE(3) to update the next configure pose. The advantage in the optimization on manifold SE(3) is that the exponential map that connects the Lie algebra and the Lie group naturally preserves the orthogonal constraint during each iteration. An experimental comparison is expected to show that the ways of simply updating the iteration on the local parameter space make the forward kinematic optimization algorithms more susceptible to being stuck in branch solutions.

The rest of this paper is organized as follows. In Section 2, some preliminary notations and terminology from the theory of Lie group and Lie algebra is reviewed. Section 3 is devoted to formulating the problem of forward kinematics for a general type G-S platform. Two kinds of iterative algorithms framework, the Gauss–Newton (G-N) and Levenberg–Marquardt (L-M) type algorithms are employed in algorithm construction with the mathematical tools from Lie group in Section 4. In Section 5, these two types of Lie group-based iterative algorithms are compared with traditional algorithms on Euclidean space $\mathbb{R}^6$ as well as iterative algorithms using quaternion parametrization through a case study for a general G-S platform.

2. Preliminary Notations and Terminology

2.1. Lie Group and Lie Algebra

We assume that readers are familiar with basic mathematical concepts about group and manifold (see e.g., [19,20] and references therein). The configure pose of the G-S platform end-effector belongs to a special Euclidean group SE(3), which is a semi-direct product of $\mathbb{R}^3$ with special orthogonal group SO(3). An element $g \in SE(3)$ can be presented using a 4×4 homogeneous transformation matrix as follows.

$$g = \begin{bmatrix} R & p \\ 0^T & 1 \end{bmatrix} \tag{1}$$

where $R \in SO(3)$ is a 3×3 orientation matrix, $p \in \mathbb{R}^3$ is a 3×1 translational vector.

The inverse of g can be written in the following formulation,

$$g^{-1} = \begin{bmatrix} R^T & -R^T p \\ 0^T & 1 \end{bmatrix} \tag{2}$$

Given a basis $\{E_1, \cdots, E_6\}$ for matrix Lie algebra $\mathfrak{se}(3)$, any arbitrary element X, which is also called a "screw" matrix in the robot community, can be written in the following form,

$$X = \begin{pmatrix} 0 & -x_3 & x_2 & x_4 \\ x_3 & 0 & -x_1 & x_5 \\ -x_2 & x_1 & 0 & x_6 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \sum_{i=1}^{6} x_i E_i \tag{3}$$

where $E_1, E_2, \cdots, E_6$ are listed as Equation (A1) in Appendix A.

Given a rigid body motion $g(t) \in SE(3)$, its associated Lie algebra element shown as Equation (4) denotes a spatial screw matrix S_r in an inertial frame. Meanwhile, Equation (5) denotes a screw matrix S_b in body reference frame.

$$S_r = \dot{g} g^{-1} \tag{4}$$

$$S_b = g^{-1} \dot{g} \tag{5}$$

Accordingly, there exist two 6×1 column vectors, spatial screw vector s_r and body screw vector s_b, which can be derived using following operation.

$$s_r = S_r^{\vee} = \begin{bmatrix} (\dot{R} R^T)^{\vee} \\ -\dot{R} R^T p + \dot{p} \end{bmatrix} = \begin{bmatrix} \omega_r \\ v_r \end{bmatrix} \tag{6}$$

$$s_b = S_b^{\vee} = \begin{bmatrix} (R^T \dot{R})^{\vee} \\ R^T \dot{p} \end{bmatrix} = \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} \tag{7}$$

Here the vee operator $(\cdot)^{\vee} : \mathfrak{se}(3) \to \mathbb{R}^6$ is defined to extract screw coefficients from the screw matrix to form a 6×1 column vector. Conversely, a wedge operator $(\cdot)^{\wedge} : \mathbb{R}^6 \to \mathfrak{se}(3)$ maps the screw vector back to the matrix Lie algebra $\mathfrak{se}(3)$, that is $s_r^{\wedge} = S_r$, $s_b^{\wedge} = S_b$.

2.2. Exponential Map

For each vector field X in tangent space $T_e G$ at group identity e, there exists a smooth one-parameter subgroup $\gamma_X(t)$ of a Lie group G parametrized by a scalar $t \in \mathbb{R}$. There also exists an exponential map defined as follows.

$$exp : T_e G \times \mathbb{R} \to G, (X, t) \mapsto exp(tX). \tag{8}$$

This is a unique homomorphism map from tangent space T_eG around group identity e to one-parameter subgroups of G. The exponential map is related with a one-parameter subgroup by $exp(X) = \gamma(1)$ at $t = 1$.

With these properties of the exponential map, we can define an integral curve of right invariant vector field X^r passing through point $g \in G$ at time zero as $\gamma^r_X : \mathbb{R} \times G \to G, (t, g) \mapsto exp(tX) \circ g$. Similarly, $\gamma^l_X : \mathbb{R} \times G \to G, (t, g) \mapsto g \circ exp(tX)$ denotes an integral curve of left invariant vector field X^l passing through $g \in G$ at time zero.

As for special the Euclidean group SE(3), there exists an associated screw matrix as shown in Equation (3) at its tangent space $T_eSE(3)$. Then, motion around identity I_4 can be obtained by exponentiating these screw matrices. Here, the exponential map $exp : \mathfrak{se}(3) \to SE(3)$ corresponds to matrix exponentiation, which has the closed-form Rodriguez formula as shown in Equation (A2) in the Appendix A. Examples for rotational screw vector x_1 and translational screw vector x_6 are expressed as follows,

$$exp(x_1 E_1) = \begin{pmatrix} cosx_i & -sinx_i & 0 & 0 \\ sinx_i & cosx_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} and \quad exp(x_6 E_6) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & x_6 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{9}$$

2.3. *Taylor Series Expansion of an Analytic Function on Lie Group*

The Taylor series of a function on Euclidean space $\mathbb{R}^n$ can be extended naturally to Lie group-structured non-Euclidean space. In the same way, there exists similar derivatives for group-valued function $f : G \to \mathbb{R}$. Owing to the fact that there exist two invariant vector fields around $g \in G$, The Taylor series expansion around $g \in G$ can be formulated in two ways. The following equation provides the right way.

$$f(g \circ exp(tX)) = f(g) + \frac{d}{dt} f(g \circ exp(tX))|_{t=0} + \frac{1}{2!}\frac{d^2}{dt^2} f(g \circ exp(tX))|_{t=0} + \cdots \tag{10}$$

As discussed in [21], the first and second-order term can be expressed in a different form. Analogously to directional derivative, the first-order term are completely equivalent to the following formulation,

$$(X^r f)(g) \triangleq \frac{d}{dt} f(g \circ exp(tX))|_{t=0} = \lim_{t \to 0} \frac{f(g \circ exp(tX)) - f(g)}{t}. \tag{11}$$

Here, $(X^r f)(g)$ is called right Lie derivative of $f(g)$ with respect to vector filed X. Since E_i is a basis for Lie algebra $\mathfrak{g}$, which can be shown as $X = \sum_{i=1}^n x_i E_i$, then its associated differential operators will be denoted as $E^r_i f$. Thus, the right Lie derivative will be as follows,

$$(X^r f)(g) = \sum_{i=1}^n x_i (E^r_i f)(g) \tag{12}$$

Finally, the "right" Taylor series expansion around $g \in G$ can be written to a second order in the following equation.

$$f(g \circ exp(tX)) = f(g) + t \sum_{k=1}^n (E^r_k f)(g) x_k + \frac{1}{2} t^2 \sum_{i=1}^n \sum_{j=1}^n (E^r_i E^r_j f)(g) x_k x_l + O(\|x\|^3 t^3) \tag{13}$$

A "left" Taylor series expansion can also be derived in an analogous manner.

$$f(exp(-tX) \circ g) = f(g) + t \sum_{k=1}^n (E^l_k f)(g) x_k + \frac{1}{2} t^2 \sum_{i=1}^n \sum_{j=1}^n (E^l_i E^l_j f)(g) x_k x_l + O(\|x\|^3 t^3) \tag{14}$$

3. Problem Formulation of Forward Kinematics for G-S Platform

This section is devoted to formulating the forward kinematic problem of a general type G-S platform as a minimum optimization problem. The ultimate problem formulation depends on which parametrization scheme of configure pose is applied. Here, we take 3D + YPR scheme as an example, where position is in three-dimensional flat Euclidean space $\mathbb{R}^3$ and orientation is parametrized by three angles in Yaw–Pitch–Roll order.

As for a general G-S platform, its geometric structure of a general G-S platform is depicted in the following Figure 1.

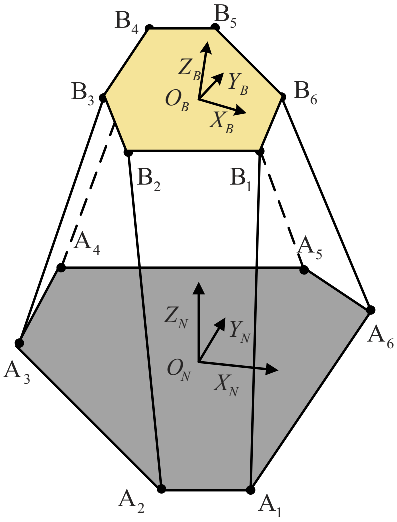


Figure 1. Geometric structure of a general Gough–Stewart (G-S) platform.

It consists of one lower fixed platform and one upper moving platform connected through six identical extensible legs. Both two platforms are irregular hexagon structure with their six vertices located at one circumcircle.

In order to describe the relative configure pose between two platform, one coordinate frame $\{O_N - X_N Y_N Z_N\}$ is fixed at geometric center of lower fixed platform O_N, another frame $\{O_B - X_B Y_B Z_B\}$ is fixed at geometric center of upper moving platform O_B. Let $T \in SE(3)$ denote the configure pose (position and orientation) of geometric center O_B of upper moving platform represented in lower fixed frame $\{N\}$.

$$T = \begin{bmatrix} R & p \\ 0^T & 1 \end{bmatrix} \tag{15}$$

where orientation matrix $R \in SO(3)$ is defined by successive rotation in Z-Y-X order.

Let a_i denote the position vector of the lower joint center A_i of i-th leg in frame $\{N\}$, and let b_i denote the position vector of upper joint center B_i of i-th leg in frame $\{B\}$.

The inverse kinematic function of the G-S platform defined as a six dimensional real-valued function $F : SE(3) \to \mathbb{R}^6$ describes a kinematic mapping from configure pose T of upper moving platform to its six leg lengths $L_1, \ldots, L_6$. If T is locally parametrized by 3D+YPR generalized coordinate variable $q \in \mathbb{R}^6$; thus, the ultimate inverse kinematic function $\bar{F}(q)$ is defined as follows.

$$\bar{F}(q) \triangleq F \circ \mu_T : \mathbb{R}^6 \to \mathbb{R}^6, \quad F \circ \mu_T(q) = \|Rb_i + p - a_i\|_2, i = 1, \ldots, 6 \tag{16}$$

The forward kinematic problem of the G-S platform is formulated as a minimum optimization problem related with kinematic mapping residual error $r(T)$. Each component of $r(T)$ is defined as follows.

$$r_i(T) \triangleq \|\bar{F}_i(T)\|_2^2 - L_i^2, \quad i = 1, \ldots, 6 \tag{17}$$

where $L_1, L_2, \cdots, L_6$ are six scalars for current leg length.

Let φ denote the objective function in the following formulation.

$$\varphi : SE(3) \to \mathbb{R}, \quad \varphi(T) = \frac{1}{2}\|r(T)\|_2^2 \tag{18}$$

As shown in Figure 2, local parametrization of objective function φ around configure point T is described as follows,

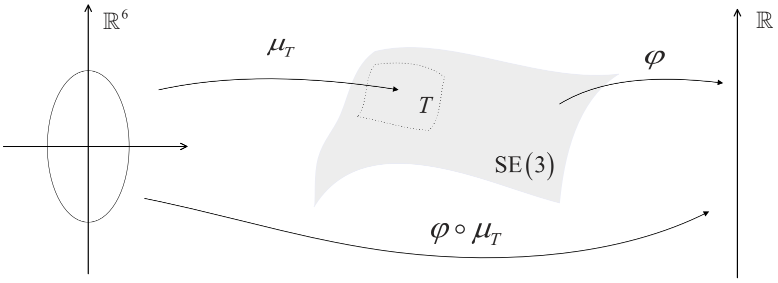


Figure 2. $\varphi \circ \mu_T$ is a local parametrization of function φ around configure pose T expressed in local parameter space $\mathbb{R}^6$.

$$\varphi \circ \mu_T : \mathbb{R}^6 \to \mathbb{R}, \quad \varphi \circ \mu_T(q) = \frac{1}{2}\|r(q)\|_2^2. \tag{19}$$

Finally, forward kinematic problem of the G-S platform can be formulated as a minimum optimization problem as follows.

$$\min_{q \in \mathbb{R}^6} \varphi \circ \mu_T(q), \quad q \in \mathcal{D} \tag{20}$$

where domain $\mathcal{D}$ denotes accessible workspace configure pose of the G-S platform.

4. Iterative Algorithms on SE(3)

As discussed in the introduction section, it will take considerable efforts before we can arrive at an analytical solution to solve the forward kinematic problem in Equation (20). In consideration of this difficulty, many numerical iteration algorithms, e.g., G-N and L-M, have been extensively applied in practical engineering projects.

In order to minimize the quadratic sum of kinematic residual error $r(T)$, the numerically iterative algorithm is updated iteratively by means of a small increment until convergence or the max number of iteration is reached. It should be noticed that all these numerical iteration schemes are originally designed to work on flat Euclidean space, i.e., $\mathbb{R}^n$. If the update process was designed on generalized coordinates $q \in \mathbb{R}^6$ according to Equation (16), there exists a gimbal lock problem as well as three Euler angles that may reach out of their valid ranges. In this section, we begin to build up two elegant iterative algorithms on the Lie group-structured manifold SE(3) with differential properties from theory of the Lie group and Lie algebra.

The core issue in the Newton-type algorithm construction process is to determine the small increments at each step through solving incremental normal equations related with the first-order Taylor series expansion. According to Equation (13), the six-dimensional inverse kinematic residual error function $r(T)$ can be approximated in the neighborhood of $T \in SE(3)$ with first-order "right" Taylor series expansion as follows.

$$r(T \circ exp(ts_b)) \approx r(T) + t(X^r r)(T) \tag{21}$$

As for computing the first order term, right Lie derivative of six-dimensional vector-valued residual function $r(T)$ was defined as $X^r r = [X^r r_1, X^r r_2, \dots, X^r r_6]^T$. Each Lie derivative with respect to body screw matrix S_b can be derived using the following formulation.

$$\begin{aligned}(X^r r_i)(T) &= \sum_{k=1}^{6}(E_k^r r_i)s_b = 2(T\tilde{b}_i - \tilde{a}_i)TS_b\tilde{b}_i \\ &= 2\Big((R^T a_i - R^T p)\times b_i)^T, (b_i - R^T a_i + R^T p)^T\Big)\begin{pmatrix}\omega_b \\ v_b\end{pmatrix}\end{aligned} \tag{22}$$

where tilde on a_i and b_i means a extension by adding an extra element 1, that is $\tilde{a}_i = (a_i^T, 1)^T$ and $\tilde{b}_i = (b_i^T, 1)^T$.

Similarly, a left Lie derivative with respect to spatial screw matrix S_r can be used to approximate $r(T)$ in "left" Taylor series expansion.

$$\begin{aligned}(X^l r_i)(T) &= \sum_{k=1}^{6}(E_k^l r_i)s_r = 2(T\tilde{b}_i - \tilde{a}_i)S_r T\tilde{b}_i \\ &= 2\Big((a_i \times (Rb_i + p))^T, (Rb_i + p - a_i)^T\Big)\begin{pmatrix}\omega_r \\ v_r\end{pmatrix}.\end{aligned} \tag{23}$$

All these six right Lie derivatives can be formed into a 6 × 6 Jacobian matrix, which is named right Lie derivative matrix J_b with respect to body screw matrix S_b as follows.

$$J_b(T) = 2\begin{bmatrix}((R^T a_1 - R^T p)\times b_1)^T & (b_1 - R^T a_1 + R^T p)^T \\ \vdots & \vdots \\ ((R^T a_6 - R^T p)\times b_6)^T & (b_6 - R^T a_6 + R^T p)^T\end{bmatrix} \tag{24}$$

Then Equation (21) can be arranged in form of body screw vector s_b as follows.

$$r(T \circ exp(tS_b)) \approx r(T) + J_b(T)ts_b \tag{25}$$

Thus we can obtain body screw error $s_b = -J_b^{-1}(T)r(T)$ by assuming $r(T \circ exp(tS_b)) = 0$. This means that the updating direction in Lie algebra vector space at time step t_k can be given as follows.

$$s_b^{(k)} = -J_b^{-1}(T^{(k)})r(T^{(k)}) \tag{26}$$

After its updating direction has been chosen, a step factor α_k $(0 < \alpha < 1)$ that satisfies the following monotonic descent rule is dedicated to constraint step size at time t_k so as to avoid divergence during iterative process. If initial setting of step factor α_k cannot meet this rule, then multiply by itself $\alpha_k = \alpha_k^m$ until the descent rule is met.

$$\begin{aligned}\|r(T_k \circ exp(\frac{\alpha}{2}S_b^{(k)}))\| &< \|r(T_k)\| \\ \|r(T_k \circ exp(\alpha S_b^{(k)}))\| &< \|r(T \circ exp(\frac{\alpha}{2}S_b^k))\|\end{aligned} \tag{27}$$

Once a descent step size has been determined, the next configure pose can be updated by mapping selected descent screw error in Lie algebra to Lie group using the exponential map.

$$T^{(k+1)} = T^{(k)} \circ exp(\alpha S_b^{(k)}) \tag{28}$$

In the end, a Lie group-based G-N algorithm for the forward kinematics of the G-S platform can be constructed as the Algorithm A1 in the Appendix B. Figure 3 provides an intuitive explanation of this algorithm.

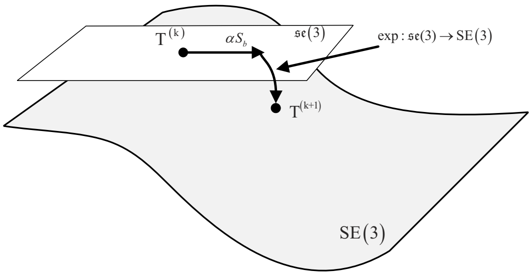


Figure 3. Updating process of G-N type algorithm on Lie group SE(3).

However, it is well known that G-N type algorithms depend heavily on good local linearization around selected expansion points. As to this issue, L-M type algorithms behave better in adjusting to a descent direction within an appropriate dynamically tuned trust region. An L-M type iterative algorithm under the framework of the Lie group is shown in Algorithm A2 in the Appendix B.

5. Implementation of Algorithms and Discussions

In this section, we first compare the aforementioned Lie group-based G-N type iterative algorithm (GN-SE(3)) with corresponding G-N type algorithms (GN-Quat and GN-$\mathbb{R}^6$). Then, it is followed with three L-M type algorithms (LM-SE(3), LM-Quat, and LM-$\mathbb{R}^6$). A general type G-S platform is leveraged to test the performance of these algorithms. The following Table 1 provides those lower joint coordinate vectors on fixed platform $\{N\}$ and upper joint coordinate vectors on moving platform frame $\{B\}$ as referred in [22].

Table 1. Coordinate vectors of the G-S platform's upper and lower joints with respect to their own coordinate frame.

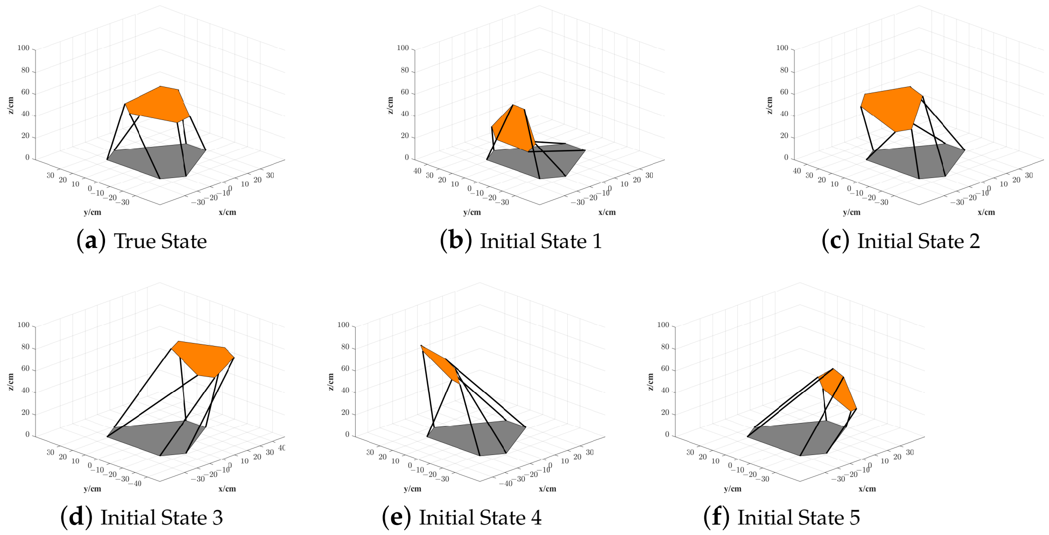
Lower Joint Coordinates on Frame {N} (cm)	Upper Joint Coordinates on Frame {B} (cm)
$a_1 = [28.9778, 7.7646, 0]^T$	$b_1 = [14.1421, 14.1421, 0]^T$
$a_2 = [-7.7646, 28.9778, 0]^T$	$b_2 = [5.1764, 19.3185, 0]^T$
$a_3 = [-21.2132, 21.2132, 0]^T$	$b_3 = [-19.3185, 5.1764, 0]^T$
$a_4 = [-21.2132, -21.2132, 0]^T$	$b_4 = [-19.3185, -5.1764, 0]^T$
$a_5 = [-7.7646, -28.9778, 0]^T$	$b_5 = [5.1764, -19.3185, 0]^T$
$a_6 = [28.9778, -7.7646, 0]^T$	$b_6 = [14.1421, -14.1421, 0]^T$

Take a group of current leg lengths as our destination $\mathcal{L}_d$, which corresponds to the true value of platform pose $q_C = [0, 0, 50, 20°, 0°, -30°]^T$. Figure 4a provides true state of G-S platform in 3D Cartesian coordinate frame.

$$\mathcal{L}_d = [55.8558, 62.5313, 52.7436, 55.1457, 44.7972, 55.9910](cm) \tag{29}$$

Due to the existence of multiple solutions for the G-S platform, we can easily find out another two branch solutions $q_{S1} = [17.58, 10.34, 36.71, -90.72°, -112.2°, 38.00°]^T$ and $q_{S2} = [17.58, 10.34, 36.71, 89.28°, -67.82°, -142.0°]^T$ that correspond to the same lengths group $\mathcal{L}_d$. Although the two branch solutions q_{S1} and q_{S2} are represented with different Euler angles, their orientation parts actually correspond to the same orientation matrix. Therefore, both branch solutions q_{S1} and q_{S2} should be regarded as one configure pose T_s as follows. The following Figure 5 gives its platform state in 3D Cartesian space.

$$T_s = \begin{bmatrix} -0.2975 & 0.7374 & -0.6064 & 17.58 \\ -0.2324 & 0.5602 & 0.7951 & 10.34 \\ 0.9260 & 0.3775 & 0.0047 & 36.71 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{30}$$



(**a**) True State (**b**) Initial State 1 (**c**) Initial State 2

(**d**) Initial State 3 (**e**) Initial State 4 (**f**) Initial State 5

Figure 4. True state and five initial guess states of the G-S platform.

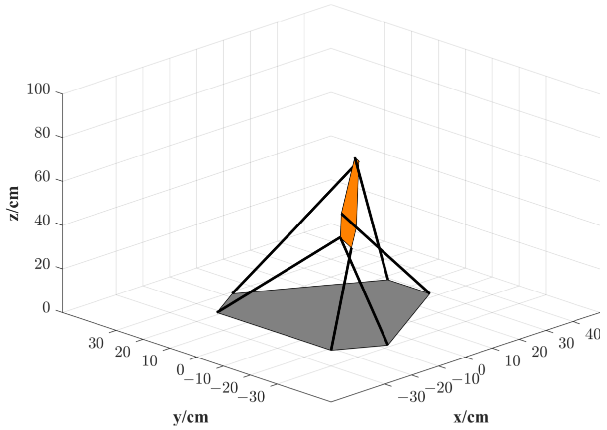


Figure 5. Platform state T_s corresponding to branch solutions q_{S1} and q_{S2}.

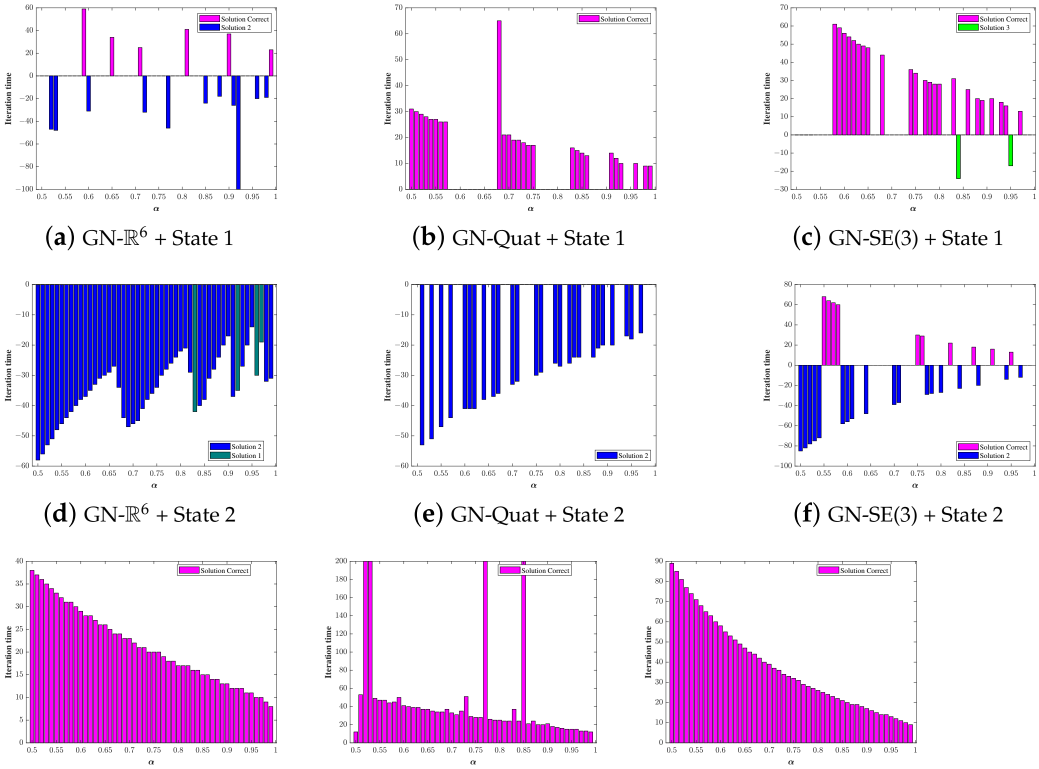
In algorithm initialization phase, five configure poses as shown in Table 2 are provided as initial guess states to testify to three kinds of G-N type forward kinematic algorithms. All these initial states are chosen to be far away from the true value q_C. The choice of these five configure poses are based on the performance that G-N and L-M algorithms on $\mathbb{R}^6$ (GN-$\mathbb{R}^6$ and LM-$\mathbb{R}^6$) act at these initial guess states. State 1, state 2, and state 4 represent a group of initial guess values that GN-$\mathbb{R}^6$ and LM-$\mathbb{R}^6$ deteriorated quickly, while state 3 and state 5 represent the opposite type of initial guess value, these two algorithms converged to the true value normally.

The other algorithms that were constructed on quaternion-based Euclidean space $\mathbb{R}^7$ and SE(3) are expected to improve the convergence performance when starting from state 1, state 2, and state 4 because both of them have kept the orthogonal constraint at each iteration process. Meanwhile, state 3 and state 5 are necessary moderate choices to verify the reliability of the other two algorithms. In order to evaluate the influence of the step factor α on algorithm performance, the step factor α is selected from 0.50 to 0.99 with an equal interval of 0.01. The reason to choose a relatively large value for the step factor is to make sure at least half of the derived updating direction could be exploited at each iteration. All tested G-N type algorithms share the same accuracy threshold ($\epsilon_1 = 1.0 \times 10^{-14}$, $\epsilon_2 = 1.0 \times 10^{-14}$, $\epsilon_3 = 1.0 \times 10^{-14}$), and max iteration time $k_{max} = 200$ to end their iteration. Figure 4b–f corresponds to these five initial guess states of the G-S platform, respectively.

Table 2. Five initial configure poses of the G-S platform and performance comparison of G-N type algorithms.

[x,y,z] (cm)	$[\phi, \theta, \psi]$	Percentage That Converges to True Value		
		GN-$\mathbb{R}^6$	GN-Quat	GN-SE(3)
$[0, 20, 20]$	$[10°, 100°, 5°]$	10%	52 %	52%
$[0, 20, 40]$	$[0°, -50°, 70°]$	0.0%	0.0%	20%
$[20, -15, 70]$	$[20°, -20°, 50°]$	100%	100%	100%
$[-20, 10, 70]$	$[50°, -20°, 70°]$	0.0%	4%	54%
$[20, -10, 40]$	$[60°, 70°, 50°]$	100%	100%	100%

In this experiment, two criteria were employed to evaluate algorithm performance, one is the range of step factor that converges to true value q_C, the other is convergence speed. The algorithm performance reflects significant difference among these five initial states. Their performance difference comparisons are shown in Figures 6 and 7, which take step factor α and iteration time as their coordinate axes. Let the positive direction of longitudinal axis indicate the iteration time that an algorithm reaches final true value q_C within given accuracy threshold while negative direction indicates the iteration time that it takes to converge to other branch solutions. This comparison analyst yields the following conclusions.

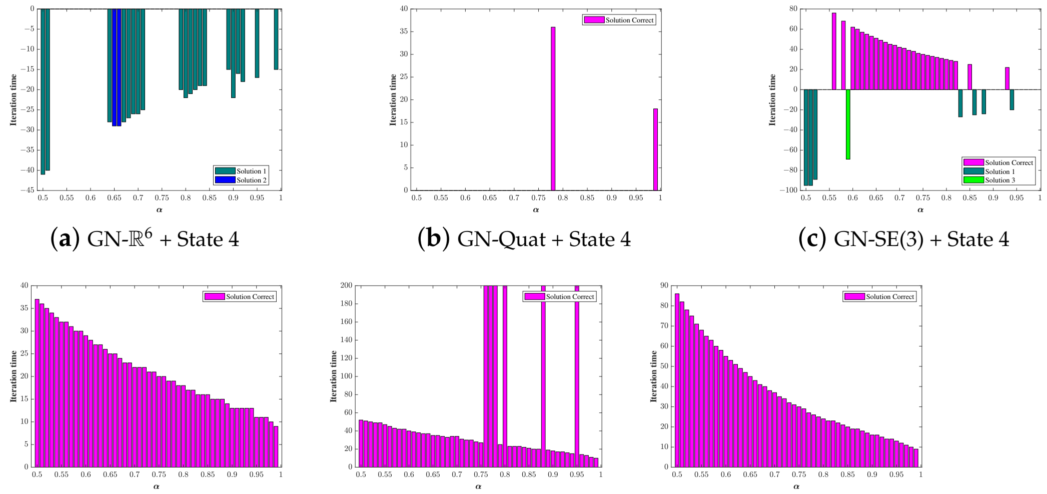


(**a**) GN-$\mathbb{R}^6$ + State 1 (**b**) GN-Quat + State 1 (**c**) GN-SE(3) + State 1

(**d**) GN-$\mathbb{R}^6$ + State 2 (**e**) GN-Quat + State 2 (**f**) GN-SE(3) + State 2

(**g**) GN-$\mathbb{R}^6$ + State 3 (**h**) GN-Quat + State 3 (**i**) GN-SE(3) + State 3

Figure 6. G-N algorithm performance comparison for initial guess state 1, state 2, and state 3.



(**a**) GN-$\mathbb{R}^6$ + State 4 (**b**) GN-Quat + State 4 (**c**) GN-SE(3) + State 4

(**d**) GN-$\mathbb{R}^6$ + State 5 (**e**) GN-Quat + State 5 (**f**) GN-SE(3) + State 5

Figure 7. G-N type algorithm performance comparison for initial guess state 4 and state 5.

1. For GN-$\mathbb{R}^6$, it achieves fast convergence when it starts from state 2, state 3, and state 5. However, it is more likely to reach one branch solution q_{S2} when state 2 is selected. For state 4, the algorithm is inclined to another branch solution q_{S1} within some isolated range. As to state 1, due to its divergence under the large range of the step factor, it means that state 1 is not an appropriate choice for the GN-$\mathbb{R}^6$ initial guess value.
2. For GN-Quat, it ends up with the true value q_C among some isolated range of step factor when starting from state 1. For state 2, it always converges to q_{S2}. For state 3 and state 5, although it achieves convergence to the true value q_C, but it could possibly behave worse if the convergence speed is slowed down at some specific larger step factors. The state 4 is definitely no longer a suitable choice for GN-Quat algorithm initialization.
3. For GN-SE(3), when it starts from state 3 and state 5, it always converges to the true value and the larger step factor would be helpful to reduce iteration time. When state 1 is provided as the initial guess state, it converges to the true value q_C with a probability of more than fifty percent. For state 4, it converges to the true value with a consecutive range from 0.60 to 0.84.

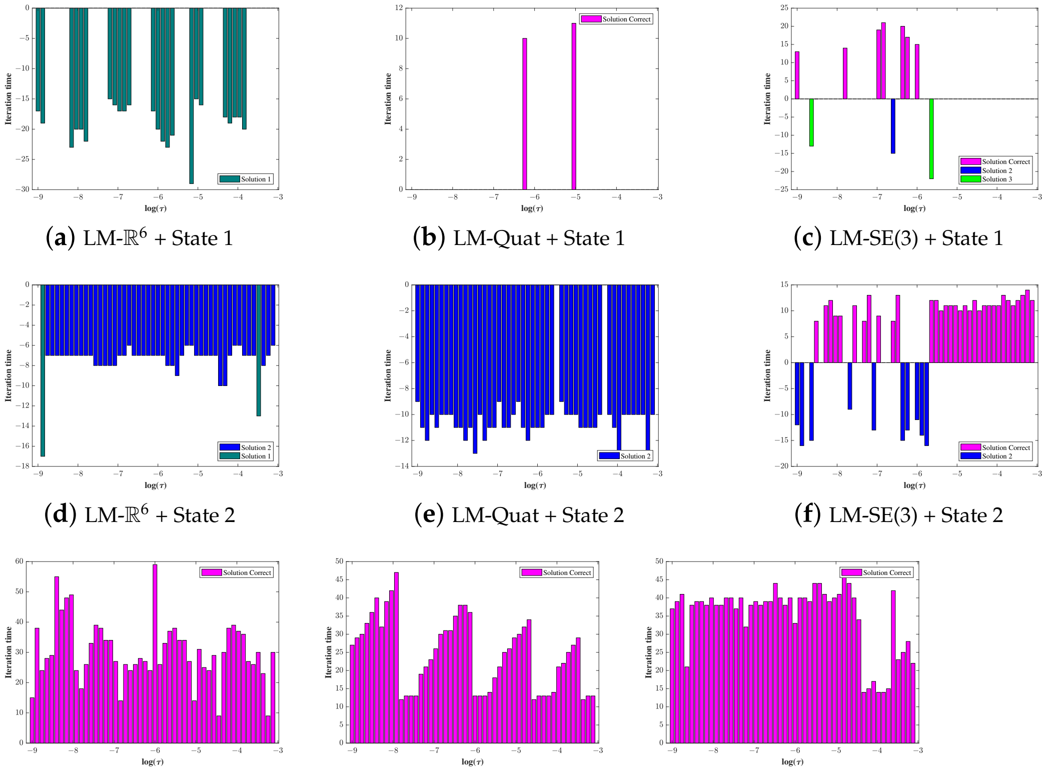
In the second comparison experiment, these five configure poses were provided as input to L-M type algorithms (LM-$\mathbb{R}^6$, LM-Quat and LM-SE(3)). Their performance comparison with respect to the damping ratio τ is given in Table 3. The logarithmic of the damping ratio τ that was taken as horizontal coordinate is assigned to a range between -9 and -3.12 with an equal interval of 0.12. Its iteration time is taken as its longitudinal coordinate. All these tested L-M type algorithms share the same accuracy threshold ($\epsilon_1 = 1.0 \times 10^{-14}$, $\epsilon_2 = 1.0 \times 10^{-144}$) and max iteration time $k_{max} = 200$ to terminate their iteration.

As seen in Table 3 and Figure 8, if state 3 is employed for algorithm initialization, all three kinds of L-M type algorithms converge to the true value q_C at an acceptable speed in the whole range of the damping ratio τ. However, there exists obvious performance differences between these L-M type algorithms when the four other states are used to initialize these algorithms. Their performance comparisons are arranged in Figures 8 and 9. From these comparisons, we can draw the following conclusions,

1. For LM-$\mathbb{R}^6$, when its is initialized from state 5, it achieves fast convergence toward true value q_C in the whole range of damping factor. However, it is more likely to converge to branch solution q_{S2} when state 2 is employed. If state 4 is taken into consideration, it reaches the true value q_C within some isolated regions for damping

factor selection. As to state 1, it has a higher chance to converge to the branch solution q_{S1} within some isolated regions.

2. For LM-Quat, it would converge to the true value q_C when it starts form state 5. From state 2, it also achieves almost total convergence towards branch solution q_{S2}. For state 1, it is definitely not suitable for initialization as it cannot converge to any solution within the whole selection range. Finally, when the algorithm initialized from state 4, it can achieve some convergence to the true value q_C within some isolated regions for damping factor τ.
3. For LM-SE(3), it can converge to the true value q_C with almost the whole range of damping factor selection except some large values. For state 2, it behaved better in convergence to the true value q_C when the logarithm value of the damping factor is selected from -5.5 to -3.12. For state 1 and state 4, its convergence probability towards the true value q_C is quite small, which means that the two states cannot be appropriate for LM-SE(3) algorithm initialization.



(**a**) LM-$\mathbb{R}^6$ + State 1 (**b**) LM-Quat + State 1 (**c**) LM-SE(3) + State 1

(**d**) LM-$\mathbb{R}^6$ + State 2 (**e**) LM-Quat + State 2 (**f**) LM-SE(3) + State 2

(**g**) LM-$\mathbb{R}^6$ + State 3 (**h**) LM-Quat + State 3 (**i**) LM-SE(3) + State 3

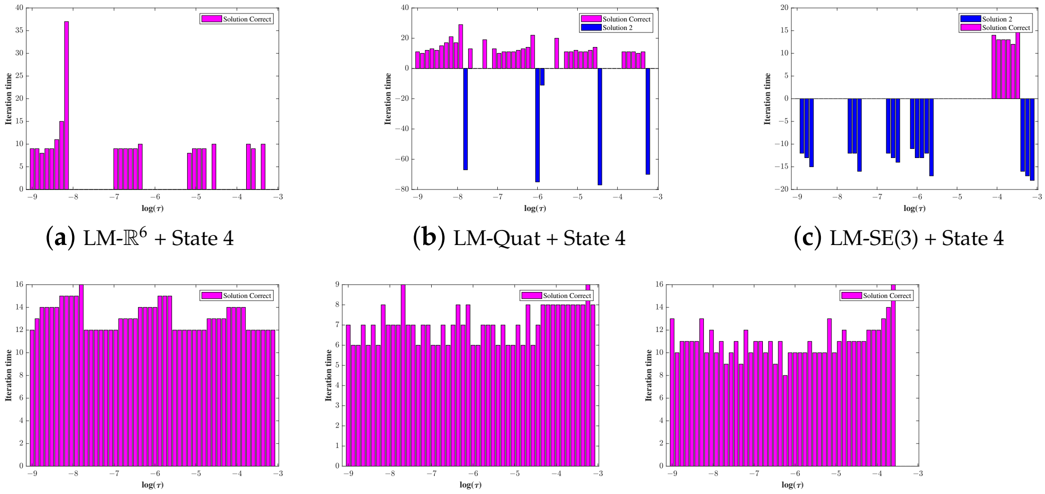
Figure 8. L-M type algorithm performance comparison for initial guess state 1, state 2, and state 3.

Table 3. Five initial configure poses of the G-S platform and performance comparison for L-M type algorithms.

[x,y,z] (cm)	**[ϕ, θ, ψ]**	**Percentage to True Value**		
		LM-$\mathbb{R}^6$	**LM-Quat**	**LM-SE(3)**
[0, 20, 20]	[10°, 100°, 5°]	0.0%	4.0%	14%
[0, 20, 40]	[0°, −50°, 70°]	0.0%	0.0%	66%
[20, −15, 70]	[20°, −20°, 50°]	100%	100%	100%
[−20, 10, 70]	[50°, −20°, 70°]	44%	68%	12%
[20, −10, 40]	[60°, 70°, 50°]	100%	100%	92%

Finally, we consider two evaluation indicators for the initial guess state selection, convergence probability toward the true value q_C and consecutive region for influence factors' selection. For GN type algorithms, state 1, state 4, and state 5 are totally or conditionally suitable for LM-SE(3). State 5 is the only suitable initial guess state for GN-$\mathbb{R}^6$ and conditionally acceptable for GN-Quat. For LM type algorithms, state 5 is totally suitable initial guess state choice for all three algorithms. Meanwhile, state 2 is conditionally acceptable for LM-SE(3) algorithm.

In summary, we can arrive at some conclusions through the above comparative analysis. Firstly, the iterative algorithms constructed on SE(3) would leave more space for initial guess state selection in the workspace of the G-S platform. Secondly, the algorithm on SE(3) has few chances to be stuck in other branch solutions even though the initial guess value is near to the branch solution.



(**a**) LM-$\mathbb{R}^6$ + State 4 (**b**) LM-Quat + State 4 (**c**) LM-SE(3) + State 4

(**d**) LM-$\mathbb{R}^6$ + State 5 (**e**) LM-Quat + State 5 (**f**) LM-SE(3) + State 5

Figure 9. L-M algorithm performance comparison for initial guess state 4 and state 5.

6. Conclusions

In this paper, we discussed how to construct G-N and L-M type iterative algorithms (GN-SE(3) and LM-SE(3)) to solve the forward kinematics problem of a G-S platform with mathematical tools from the Lie group. The key part of the Lie group-based iterative algorithm was to determine an updating direction in Lie algebra $\mathfrak{se}(3)$ and was then projected back to the Lie group SE(3) with exponential map, which differentiates the algorithm from traditional iterative algorithms based on locally Euclidean space $\mathbb{R}^6$ or quaternion-based parametrization space $\mathbb{R}^7$. Five different initial configure poses of a general G-S platform were used to compare Lie group-based algorithms (GN-SE(3) and LM-SE(3)) with two other kinds of iterative algorithms. Experiment results demonstrated that Lie group-based iterative algorithms behave better in converging to the true solution while the two other algorithms leave more chance in converging to another branch solutions. In the future, more work is needed to study how geometric parameters of the G-S platform can influence the choice of the iterative forward kinematic algorithm.

Author Contributions: Conceptualization, B.X.; investigation, B.X.; methodology, B.X.; project administration, S.D.; resources, B.X.; software, F.L.; supervision, S.D.; validation, B.X.; visualization, B.X.; writing—original draft, B.X.; writing—review and editing, S.D. and F.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Key Research and Development Plan of China (No. 2018AAA0102902).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Appendix A.1

The following $E_1, E_2, \cdots, E_6$ are six basis components for matrix Lie algebra $\mathfrak{se}(3)$.

$$E_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad E_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad E_3 = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$
$$E_4 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad E_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad E_6 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{A1}$$

Appendix A.2

The Rodriguez formula for screw matrix S in $\mathfrak{se}(3)$ is as follows,

$$e^S = I_4 + S + \frac{1}{\theta^2}(1 - cos\theta)S^2 + \frac{1}{\theta^3}(\theta - sin\theta)S^3 \tag{A2}$$

where $\theta^2 = ||\omega||^2$.

Appendix B

The G-N algorithm that constructed on Lie group-structured manifold SE(3) are shown as follows. Here, s_{gn} denotes the updating direction in Lie algebra vector space $\mathbb{R}^6$ during each iteration.

Algorithm A1 G-N Type Algorithm on SE(3) for solving the forward kinematics problem of a G-S platform.

```
Input: Coordinate vectors of upper and lower joints with respect to their own coordinate
    frames: $b_1, \cdots, b_6$; $a_1, \cdots, a_6$; Current leg lengths: $L_1, \cdots, L_6$; Initial Configuration
    pose: $T_0$; Accuracy threshold: $\epsilon_1$, $\epsilon_2$, $\epsilon_3$; Max iteration time $k_{max}$.
Output: Final configuration pose: $T_f$.
$k \leftarrow 0, T \leftarrow T_0$;
$A \leftarrow J_b(T)^T J_b(T), g \leftarrow J_b(T)^T r(T)$;
$found \leftarrow (\|g\|_\infty \leq \epsilon_1)$
while (not found) and $(k < k_{max})$ do
    $k \leftarrow k + 1$;
    Solve normal equation: $A \cdot s_{gn} = -g$;
    if $\|s_{gn}\| \leq \epsilon_2$ then
        $found \leftarrow true$;
        $\alpha \leftarrow \alpha_0$
        $accept \leftarrow (\|r(T \circ exp(0.5\alpha S_{gn}))\| \leq \|r(T)\|)\&\&(\|r(T \circ exp(\alpha S_{gn})) \leq \|r(T \circ exp(0.5\alpha S_{gn}))\|)$;
        while (accept or $\alpha \leq \epsilon_3$) do
            $accept \leftarrow (\|r(T \circ exp(0.5\alpha S_{gn}))\| \leq \|r(T)\|)$ && $(\|r(T \circ exp(\alpha S_{gn})) \leq \|r(T \circ exp(0.5\alpha S_{gn}))\|)$;
            $\alpha \leftarrow \alpha \cdot \alpha$;
        end while
        if $\alpha \leq \epsilon_3$ then
            $found \leftarrow true$; //No appropriate descent step
        else
            $T \leftarrow T \circ exp(\alpha S_{gn})$; //Update configuration pose
            $A \leftarrow J_b(T)^T J_b(T), g \leftarrow J_b(T)^T r(T)$;
            $found \leftarrow (\|g\|_\infty \leq \epsilon_1)$;
        end if
    end if
end while
return Final configure pose $T_f \leftarrow T$.
```

Likewise, the L-M type algorithm on SE(3) is constructed as follows. Here, s_{lm} denotes the updating direction in Lie algebra vector space $\mathbb{R}^6$ during each iteration process.

Algorithm A2 L-M Type Algorithm on SE(3) for solving the forward kinematics problem of G-S platform.

Input: Coordinate vectors of upper and lower joints with respect to their own coordinate frames: $b_1, \cdots, b_6$; $a_1, \cdots, a_6$; Current leg lengths: $L_1, \cdots, L_6$; Initial Configuration pose: T_0; Accuracy threshold: ϵ_1, ϵ_2; Max iteration time k_{max}.

Output: Final configuration pose: T_f.

```
$k \leftarrow 0, T \leftarrow T_0$;
$A \leftarrow J_b(T)^T J_b(T), \quad g = J_b(T)^T r(T)$;
$found \leftarrow (\|g\|_\infty \leq \epsilon_1), \quad \mu = \tau \cdot max\{a_{ii}\}$;
while $(notfound)$ and $(k < k_{max})$ do
    $k \leftarrow k + 1.$;
    Solve: $(A + \mu I)s_{lm} = -g$;
    if $\|s_{lm}\| \leq \epsilon_2$ then
        $found \leftarrow true$;
    else
        $T_{new} \leftarrow T \circ exp(S_{lm})$;
        $\varrho = \frac{F(T)-F(T_{new})}{L(T)-L(T_{new})}$;
        if $\varrho > 0$ then
            $T \leftarrow T_{new}$;
            $A \leftarrow J_b(T)^T J_b(T), \quad g \leftarrow J_b(T)^T r(T)$;
            $found \leftarrow (\|g\|_\infty) \leq \epsilon_1$;
            $\mu \leftarrow \mu \cdot max\{\frac{1}{3}, 1 - (2\varrho - 1)^3\}, \quad \nu \leftarrow 2$;
        else
            $\mu \leftarrow \mu \cdot \nu, \quad \nu \leftarrow 2 \cdot \nu$;
        end if
    end if
end while
return Final configure pose $T_f \leftarrow T$.
```

References

1. Raghavan, M. The Stewart platform of general geometry has 40 configurations. *J. Mech. Des.* **1993**, *115*, 277–282. [CrossRef]
2. Ronga, F.; Vust, T. Stewart platforms without computer. In Proceedings of the Conference Real Analytic and Algebraic Geometry, Trento, Italy, 21–25 September 1992; De Gruyter: Berlin, Germany, 1995; pp. 197–212.

3. Husty, M.L. An algorithm for solving the direct kinematics of general Stewart-Gough platforms. *Mech. Mach. Theory* **1996**, *31*, 365–379. [CrossRef]
4. Huang, X.; Liao, Q.; Wei, S. Closed-form forward kinematics for a symmetrical 6-6 Stewart platform using algebraic elimination. *Mech. Mach. Theory* **2010**, *45*, 327–334. [CrossRef]
5. Ji, P.; Wu, H. A closed-form forward kinematics solution for the 6-6/sup p/Stewart platform. *IEEE Trans. Robot. Autom.* **2001**, *17*, 522–526.
6. Didrit, O.; Petitot, M.; Walter, E. Guaranteed solution of direct kinematic problems for general configurations of parallel manipulators. *IEEE Trans. Robot. Autom.* **1998**, *14*, 259–266. [CrossRef]
7. Merlet, J.P. Solving the forward kinematics of a Gough-type parallel manipulator with interval analysis. *Int. J. Robot. Res.* **2004**, *23*, 221–235. [CrossRef]
8. Wampler, C.W. Forward displacement analysis of general six-in-parallel SPS (Stewart) platform manipulators using soma coordinates. *Mech. Mach. Theory* **1996**, *31*, 331–337. [CrossRef]
9. Dietmaier, P. The Stewart-Gough platform of general geometry can have 40 real postures. In *Advances in Robot Kinematics: Analysis and Control*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 7–16.
10. Dieudonne, J.E.; Parrish, R.V.; Bardusch, R.E. *An Actuator Extension Transformation for a Motion Simulator and an Inverse Transformation Applying Newton-Raphson's Method*; National Aeronautics and Space Administration: Columbia, WA, USA, 1972.
11. Koekebakker, S. Coordinate reconstruction in model based control of a Stewart platform. *IFAC Proc. Vol.* **1998**, *31*, 103–108. [CrossRef]
12. Yang, C.; He, J.; Han, J.; Liu, X. Real-time state estimation for spatial six-degree-of-freedom linearly actuated parallel robots. *Mechatronics* **2009**, *19*, 1026–1033. [CrossRef]
13. Parikh, P.J.; Lam, S.S. A hybrid strategy to solve the forward kinematics problem in parallel manipulators. *IEEE Trans. Robot.* **2005**, *21*, 18–25. [CrossRef]
14. Innocenti, C. Closed-form determination of the location of a rigid body by seven in-parallel linear transducers. *J. Mech. Des.* **1998**, *120*, 293–298. [CrossRef]
15. Parenti-Castelli, V.; Di Gregorio, R. A new algorithm based on two extra-sensors for real-time computation of the actual configuration of the generalized stewart-gough manipulator. *J. Mech. Des.* **2000**, *122*, 294–298. [CrossRef]
16. Chiu, Y.J.; Perng, M.H. Forward kinematics of a general fully parallel manipulator with auxiliary sensors. *Int. J. Robot. Res.* **2001**, *20*, 401–414. [CrossRef]
17. Wang, Y. A direct numerical solution to forward kinematics of general Stewart-Gough platforms. *Robotica* **2007**, *25*, 121. [CrossRef]
18. Blanco, J.L. A tutorial on se (3) transformation parameterizations and on-manifold optimization. *Univ. Malaga Tech. Rep.* **2010**, *3*, 6.
19. Kamedulski, B. Product property of equivariant degree under the action of a compact abelian Lie group. In *Bulletin of the Brazilian Mathematical Society*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2020; pp. 1–13.
20. Fermanian-Kammerer, C.; Fischer, V. Defect measures on graded lie groups. In *Annali della Scuola Normale Superiore di Pisa—Classe di Scienze 21 Special Issue*; Scoula Normale Superiore: Pisa, Italy, 2020; pp. 207–291.
21. Chirikjian, G.S. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011; Volume 2.
22. Cardona, M. Kinematics and Jacobian analysis of a 6UPS Stewart-Gough platform. In Proceedings of the 2016 IEEE 36th Central American and Panama Convention (CONCAPAN XXXVI), San Jose, Costa Rica, 9–11 November 2016; pp. 1–6.