Efficient Computation of Large Deformation of Spatial Flexure-Based Mechanisms in Design Optimizations

Koen Dwarshuis
Faculty of Engineering Technology,
University of Twente,
P. O. Box 217,
Enschede AE 7500, The Netherlands
e-mail: k.s.dwarshuis@utwente.nl

Ronald Aarts
Faculty of Engineering Technology,
University of Twente,
P. O. Box 217,
Enschede AE 7500, The Netherlands
e-mail: r.g.k.m.aarts@utwente.nl

Marcel Ellenbroek
Faculty of Engineering Technology,
University of Twente,
P. O. Box 217,
Enschede AE 7500, The Netherlands
e-mail: m.h.m.ellenbroek@utwente.nl

Dannis Brouwer
Faculty of Engineering Technology,
University of Twente,
P. O. Box 217,
Enschede AE 7500, The Netherlands
e-mail: d.m.brouwer@utwente.nl

Design optimizations of flexure-based mechanisms take a lot of computation time, in particular when large deformations are involved. In an optimization procedure, statically deformed configurations of many designs have to be obtained, while finding the statically deformed configuration itself requires tens to hundreds of load step iterations. The kinematically started deformation method (KSD-method) (Dwarshuis, K. S., Aarts, R. G. K. M., Ellenbroek, M. H. M., and Brouwer, D. M., 2020, "Kinematically Started Efficient Position Analysis of Deformed Compliant Mechanisms Utilizing Data of Standard Joints," Mech. Mach. Theory, 152, p. 103911) computes deformed configurations fast by starting the computation from an approximation. This approximation is obtained by allowing the mechanism only to move in the compliant motion-direction, based on kinematic equations, using data of the flexure joints in the mechanism. This is possible as flexure-based mechanisms are typically designed to be kinematically determined in the motion directions. In this paper, the KSD-method is extended such that it can also be applied without joint-data, such that it is not necessary to maintain a database with joint-data. This paper also shows that the method can be used for mechanisms containing joints that allow full spatial motion. Several variants of the KSD-method are presented and evaluated for accuracy and required computation time. One variant, which uses joint-data, is 21 times faster and shows errors in stress and stiffness below 1% compared to a conventional multibody analysis on the same model. Another variant, which does not use joint-data, reduces the computation time by a factor of 14, keeping errors below 1%. The KSD-method is shown to be helpful in design optimizations of complex flexure mechanisms for large range of motion.
[DOI: 10.1115/1.4054730]

Keywords: compliant mechanisms, mechanism design, theoretical kinematics

1 Introduction

Flexure joints have no backlash and friction and therefore allow excellent predictable motion, in contrast to sliding or roller bearings. Therefore, they are often used in precision applications like electron microscopes and lithography equipment [1–7]. A challenge is the limitation in the range of motion of flexure joints, which is mainly limited by stress in the material and by a dramatically drop of the support stiffness and load bearing capacity under deformation [8]. Design optimizations are used to obtain flexure joint designs in which this loss of support stiffness is as low as possible [9,10].

However, these design optimizations are taking a lot of computation time. For example, it takes several hours to optimize five design parameters of a spherical flexure joint [10] (see Fig. 6) that is modeled with 48 spatial beam elements. Although the analysis of a single design for this flexure can be executed in about 10 s, the optimization takes several hours as it requires the analysis of hundreds of designs. Therefore, the number of design parameters that can be optimized is limited. Moreover, as the computation time significantly increases with increasing complexity of the analyzed part, the optimization of a complete flexure-based mechanism that contains multiple flexure joints is still practically unfeasible.

A large part of the computation time is used to obtain the configuration of a mechanism after deformation. A considerable amount of literature has been published on modeling of the deformation of flexure mechanisms, and literature overviews can be found in Refs. [11–13]. Most literature focuses on the modeling of a single leafspring (i.e., a single flexure, which is a thin flexible beamlike element that is the common building block of flexure mechanisms for large range of motion). These leafsprings can be modeled by beam elements using the finite element method, see, e.g., Refs. [14,15], but a single beam element is only accurate for small deformation. An analytical solution for large bending of beams exist which is based on elliptic integrals [16–19]. However, the application of this solution in 3D leads to an infinite series of elliptic integrals [20,21]. A method that works well in 3D is the beam constraint model, which is a model for slender beams that captures nonlinear effects [21–24]. Models for beams with a rectangular cross section of which the width is much larger than the thickness are derived in Refs. [25,26]; these models are more appropriate to model leafsprings. In all the before mentioned techniques, leafsprings can be modeled by multiple serial connected elements [14,19,23,27,28], which can optionally be solved by the chain algorithm. The chain algorithm solves the displacement of the beams individually in a sequence starting from the beams root [2,29]. However, flexure mechanisms for large range of motion tend to be composed of many leafsprings such that all the before mentioned techniques that model each leafspring individually make the analysis still cumbersome.

The most widely used method to reduce the computation time of static computations on general mechanisms is model order reduction technique [30–32]. However, many of these techniques are only accurate for small deformations such that they cannot be used for the optimization of flexure joints for large range of

Contributed by the Mechanisms and Robotics Committee of ASME for publication in the Journal of Mechanisms and Robotics. Manuscript received February 15, 2021; final manuscript received July 21, 2021; published online June 23, 2022. Assoc. Editor: Charles Kim.

motion. The available model order reduction techniques for large deformation typically use a data of the mechanism which have to be available before the actual simulation [33,34]. In a design optimization, these required data are typically not available and obtaining the required data before the actual optimization is unpractical as this requires a lot of computation time. Therefore, model order reduction techniques that require data of the full mechanism are not suitable for design optimizations.

The main reason for the large computation time of conventional methods is that a large deformation cannot be applied in a single computation step. A large deformation has to be applied in multiple small steps and for each step, equilibrium has to be achieved by applying an iterative procedure in order to ensure the mathematical system to converge. In the kinematically started deformation method (KSD-method) [35], we avoided this by starting the computation from a cheap obtained approximation of the deformed configuration instead of the undeformed configuration.

The essence of this approximation is the fact that flexure-based mechanisms for precision mechanisms are typically designed to be compliant in the motion directions and stiff in the other directions [36]. The kinematic behavior of these motion directions is well determined, meaning that its motion is almost independent of the stiffness properties of the mechanism and therefore this motion can be approximated by kinematic equations. The KSD-method approximates this motion and uses the result as a starting point for finding equilibrium of the mechanism. Henceforth, the motion in the motion-direction will be called "intended motion" (in other papers referred to as "degrees-of-freedom (DOFs)") and the motion in the stiff, ideally constraint direction is called "unintended motion" (also referred to as "support-directions" or "off-axis directions" in other papers).

In Ref. [35], the approximation was made using previously obtained data of the flexure joints in the mechanism. These data (henceforth joint-data) describe the kinematic behavior of the joints deforming in the intended directions. It is useful to store joint-data in a database as the required data are almost independent on the stiffness properties of the joint, but mainly depend on the joint-composition (i.e., the way in which the flexures in the joint are positioned and connected with respect to each other). Flexure mechanisms are often built from joints with a standard compositions, examples of such standard joint-compositions are the cross-flexure [37] and the butterfly hinge [38].

Two different variants of the KSD-method were introduced, KSD-full and KSD-reduced. KSD-full computes exactly the same deformed configuration as a conventional method, but more efficient. The computation time required for KSD-reduced was even lower than that of KSD-full by calculating only an approximation of the deformed configuration.

However, two issues are related to the KSD-method. In the first place, there is a strong tradeoff between computation time and the accuracy. On the one hand, the KSD-full requires a lot of computation time with respect to KSD-reduced. On the other hand, KSD-reduced can accidently result in an error that is far over 10%, especially the stress results are unreliable. The second issue is that the KSD-method requires joint-data of the joints in the mechanism. Obtaining these data requires a lot of time and therefore the KSD-method is mainly valuable for mechanisms that consist of flexure joints with compositions of whose data are already available.

In this paper, new variants of the KSD-method are developed to address the two issues mentioned before. These variants are more efficient than KSD-full and more accurate than KSD-reduced. In this way, the user can select a different variant that reduces the computation time significantly in case a lower accuracy is permitted. One of the variants does not require joint-data such that the KSD-method can also be efficiently used for mechanisms containing joints of which no joint-data are available.

Section 2 gives a summary of the existing variants of the KSD-method and Sec. 3 presents the new variants of the method. Section 4 gives results to show the efficiency and accuracy of the different variants of the KSD-method in comparison to a conventional method. In this section, the KSD-method is applied to several mechanisms, of which one contains spatial spherical flexure joints where Ref. [35] only applied the KSD-method to joints for planar motion. Section 4 also shows that the computation time can be reduced significantly by neglecting the geometric part of the stiffness matrix during the computation. One of the prerequisites for using joint-data in the KSD-method in design optimizations is that these data are almost unaffected by changes of dimensions of flexure joints. Section 5 examines this requisite and analyzes the accuracy of the KSD-method. Section 6 performs design optimizations using the KSD-method. The paper ends with the conclusions.

2 Summary of the Existing Variants of the Kinematically Started Deformation Method

Goal of the KSD-method is to make static computation more efficient by avoiding the long iterative procedures that are required in conventional methods. All variants of the KSD-method consist of several steps that will be summarized after two introductory notes.

(1) In the KSD-method, the motion of a flexure-based mechanism is split into intended motion and unintended motion. The intended motion is approximated in the first two steps of the KSD-method, mainly based on kinematic relations. The remaining motion is computed in later steps.
(2) A flexure mechanism is considered as a combination of flexure joints and stiff links. Each flexure joint is modeled by a small finite element model. Two nodes of this model are connected to the links; these are called interface nodes and their displacements are interface displacements, see Fig. 1. The other nodes are called internal nodes, and their displacements are called internal displacements. The term displacement is used for the combination of rotations and translational displacements.

Two different variants of the KSD-method were introduced in Ref. [35], KSD-full and KSD-reduced. KSD-full consists of the five steps described below. Steps 1 and 2 both require joint-data that describe the kinematic behavior of the joints. The method to obtain these data is explained after the description of the five steps. The steps are visualized in Fig. 2 for a 2D four-bar mechanism consisting of four cross-flexures and three stiff links. Each flexure is modeled by four rigid and six flexible beam elements as also shown in Fig. 1. The five steps are:

(1) *Estimate the interface displacements* based on a prescribed displacement of the end-effector and by using joint-data to constrain motion in unintended directions. For the 2D cross-flexures, these joint-specific constraint equations can be the x and y positions as a function of the intended motion, i.e., the

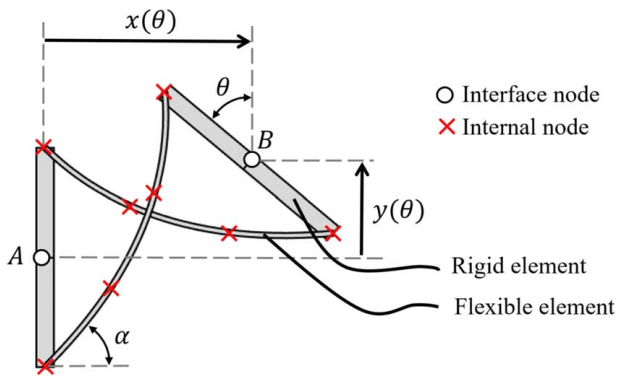

Fig. 1 Two-dimensional cross-flexure modeled by four rigid and six flexible elements in deformed configuration. Functions of $x(\theta)$ and $y(\theta)$ have to be stored in a database to constrain the unintended motion.

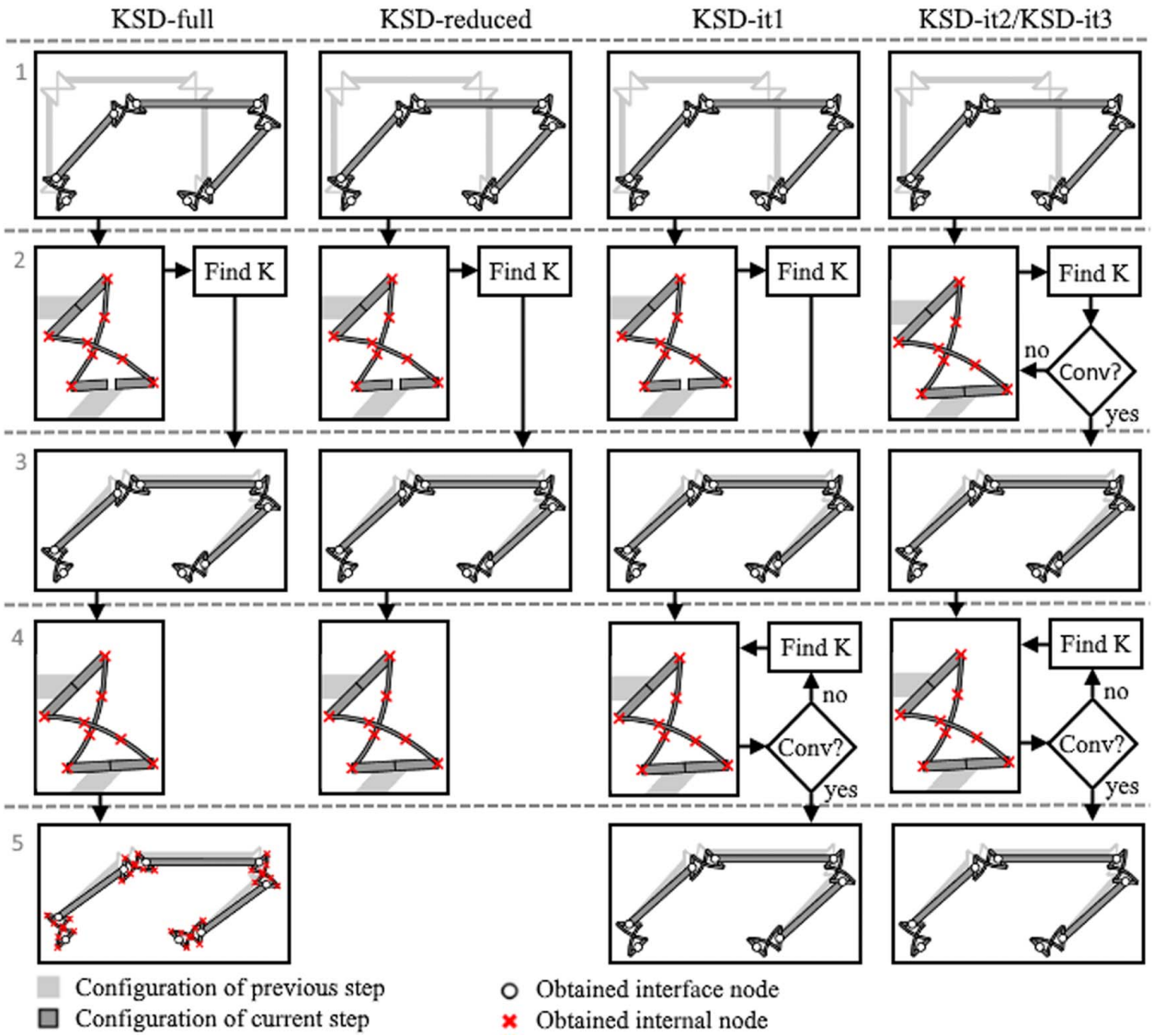

Fig. 2 Overview of the steps in various variants of the KSD-method

rotation θ, see Fig. 1. If the mechanism is kinematically determinate [36], the resulting kinematic equations can be solved uniquely.

(2) *Estimate for each joint internal displacements* based on the interface positions of that joint, and use the internal configuration to obtain the stiffness matrices of the joints. In order to speed up this step, we use the element orientation-based body that is introduced in Ref. [35], which approximates the internal configuration based on joint-data. The main idea is that an estimation of the internal configuration can be obtained efficiently if the orientations of all the elements in the joint are known. This is possible as the orientations are the only variables that make the static equilibrium equation nonlinear. Reference [35] explains the method for planar joints. Appendix A summarizes the method and explains how this technique can be used for joints with spatial intended motion.

(3) *Update the displacements of the interface points* based on static equilibrium of the mechanism. To do this, for each joint the stiffness matrix of step 2 is reduced using the Craig–Bampton boundary modes to obtain the stiffness matrix in terms of its two interface points.

(4) *Update the internal configuration for each joint* based on the new position of the interface points and the stiffness matrices of step 2.

(5) *Update all displacements* by solving the full model for static equilibrium. This step is similar to a conventional method. However, where conventional methods start from the undeformed configuration, the KSD-method starts this step from the positions that are obtained in steps 3 and 4 as initial configuration.

These are the five steps of KSD-full. Step 5 of KSD-full is the most computationally expensive, where the error in the positions after step 4 is already small. Therefore, KSD-reduced was introduced which only performs step 1 till 4 of KSD-full, shown in column 2 of Fig. 2.

The joint-data that is used in steps 1 and 2 are obtained based on static simulations with the finite element model of the joint. One of the interface nodes of the joint is fixed and the other interface node is prescribed in the intended directions. For a finite number of values of the intended motion, the configuration of the joint is obtained based on static equilibrium. Based on the resulting configurations of the joint in the intended deformation, the required data for steps 1 and 2 are approximated based on a least-square fit. Using the cross-flexure in Fig. 1 as an example, we can fix interface point A and prescribe the rotation of interface point B at a finite number of values between -30 and 30 deg. Based on the resulting configurations from static equilibrium, we can fit the positions x and

y as a function of the rotation θ which is the joint-data that are required for step 1.

The rotation of each individual element can also be fitted as a function of the intended rotation θ which is the data required for step 2. If a third-order polynomial least-square fit is used, the resulting equation for the rotation of element k in the cross-flexure will be

$$\theta_k = c_1\theta + c_2\theta^2 + c_3\theta^3 \tag{1}$$

where the constants c_i are approximated by the least-square fit. The resulting constants will be insensitive to many of the dimensions of the joint like the thickness and the width of the flexures. This is as the kinematic behavior of joints deforming in the intended direction is well determined, i.e., insensitive to the stiffness properties. This means that the same fit functions can be used for cross-flexures with different dimensions making it useful to store the result in a database.

Some dimensions may change the geometry of the flexure joint significantly and therefore influence the element rotations. In that case, the simulation results should be obtained for multiple values of this dimension and it should be taken into account explicitly in the function. As the rotations are sensitive to angle α (see Sec. 5), the function for the rotation of element k of a third-order approximation will be

$$\theta_k = c_{10}\theta + c_{20}\theta^2 + c_{30}\theta^3 + c_{11}\theta\alpha + c_{21}\theta^2\alpha + c_{12}\theta\alpha^2 \tag{2}$$

This ends the summary of the existing variants of the KSD-method, and more details can be found in Ref. [35].

3 New Variants of the Kinematically Started Deformation Method

For complex mechanisms, the error of the stress result of KSD-reduced is very sensitive to the accuracy of the joint-data. It was found that without an iterative procedure to find internal configurations of the flexure joints, the stress result was unreliable. A second observation is that the performance of the KSD-method is sensitive to errors in the unintended displacements of the joints after step 3.

Based on these two observations, a new variant of the KSD-method is introduced, referred to as KSD-it1 (shown in the third column of Fig. 2). KSD-it1 performs the first three steps of the KSD-method similar to KSD-full and KSD-reduced. In step 4, the configuration of the joints is updated with two differences with respect to KSD-full and KSD-reduced. In the first place, the update is iterative. So, where KSD-full and KSD-reduced only update the configuration once based on the stiffness matrix that was obtained in step 2, KSD-it1 recomputes the stiffness matrix after the update of the configuration. This is repeated till certain accuracy is reached. The second difference is related to the boundary conditions. In KSD-full and KSD-reduced, the positions of the interface nodes are used. KSD-it1 uses positions for the intended directions and reaction forces for the unintended directions as boundary conditions. Using the cross-flexure in Fig. 1 as an example, we define the rotation of point B with respect to point A (i.e., the position of B in intended direction) and the reaction forces on point B in horizontal and vertical directions (i.e., the reaction forces in unintended directions) as boundary conditions. The reason for this second change is that small errors in the deformations in the unintended directions have a large influence on the reaction forces and therefore on the resulting stress in the flexure joints where a small variation in the reaction forces does not significantly change the stress, the position, or the stiffness. Because of this second difference in step 4, the position of both interface points with respect to each other may change a little in this step. Therefore, a fifth step is required which updates the interface positions, and this step is performed the same as step 3.

In step 2, the internal configuration is obtained based on joint-data. However, this step can also be applied by an iterative procedure starting from the undeformed configuration of the joint. The resulting procedure is similar to step 4 of KSD-it1, except from the fact that forces in the unintended directions will be set to zero as they are not available in step 2. The resulting approach will be referred to as KSD-it2 and is shown in the fourth column of Fig. 2. The disadvantage of this approach is that step 2 will require significantly more computation time. The advantage of this procedure is that it does not require the joint-data that are otherwise used in step 2.

In KSD-it2, it still requires some joint-data to perform step 1 of the KSD-method, i.e., the data that describe the intended motion. Therefore, KSD-it3 is introduced which does not require joint-data. KSD-it3 is the same as KSD-it2 except from the approximation of the intended motion in step 1. This intended motion is obtained by assuming the hinges to be ideal, i.e., neglecting parasitic motion. For example, the intended motion of the cross-flexure is assumed to be a pure rotation around its initial center. Using such a rough approximation is possible because in step 2 the internal configuration is obtained by assuming the reaction forces in the unintended directions to be zero instead of using the displacements in the unintended directions. Some flexure joints cannot be described as ideal hinges as their intended motion is not a simple combination of ideal rotations or translations. An example is the folded leaf spring which has five intended directions that are not a simple combination of rotations and translations. KSD-it3 does not work for mechanisms containing these joints. However, for these joints, a rough approximation of the intended motion based on a few simulation results can be used.

Many more variants of the KSD-method could be defined. It is also possible to use different update methods for different joints. For example, if a mechanism contains two types of joints and for only one of them joint-data are available, then it is possible to update one of the joint-types based on the joint-data (similar to KSD-it1) and the other joint type can be assumed to be ideal with an iterative procedure in step 2 (similar to KSD-it3). In this paper, we stick to the five introduced methods as their results together give a representative overview of the achievable reduction in computation time and achievable accuracy.

The results described below indicate that KSD-reduced is the most suitable method to use in a design optimization if the accuracy requirements are not high and joint-data for all joints are available, as KSD-reduced is the most efficient variant. Otherwise, KSD-it1 is probably most suitable for the joints of which the joint-data are available and KSD-it3 for the joints of which no data is available.

Five conditions need to be fulfilled in order to obtain an accurate approximation based on kinematic relations in the first two steps of the KSD-method. Although this may seem to limit the applicability of the method significantly, the first four conditions hold for most of the common flexure-based mechanisms and optimization criteria:

- The analyzed mechanism should be built from separable joints and links.
- The analyzed mechanism should be kinematically determinated.
- The large displacement of the mechanism should be prescribed by kinematic relations (i.e., by input displacements) and not by input forces.
- The displacement of the mechanism in the unintended direction should be small, as it is initially approximated linearly.
- Most variants of the KSD-method require data of the flexure joints to be available.

The KSD-method can be applied to all flexure-based mechanisms that fulfill these criteria to obtain a deformed configuration. After this configuration is determined, the results that are required for a design optimization can be evaluated like the stress in the leaf-springs, the reaction forces on the mechanism, the required actuation forces, the stiffness of the mechanism, and its eigen frequencies.

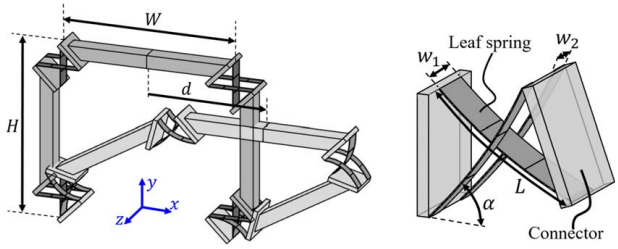

Fig. 3 Four-bar mechanism in deformed and undeformed configuration and cross-flexure modeled with three beam elements per leaf spring

Table 1 Dimensions of four-bar mechanism and cross-flexure

Height mechanism, H	0.4 m
Width mechanism, W	0.4 m
Leaf spring length, L	0.1 m
Angle leaf spring, α	45 deg
Thickness leaf springs, t	1 mm
Width inner leaf spring, w_1	40 mm
Width outer leaf springs, w_2	20 mm

4 Efficiency Results

This section evaluates the performance of the KSD-method based on three different mechanisms:

- Four-bar mechanism with four cross-flexures, see Fig. 3 and Table 1. The top bar is displaced by $d = 0.29$ m. This case was introduced in Ref. [35] with a displacement of 0.3 m. However, in KSD-it3 (which assumes ideal hinges in step 1), a displacement of 0.3 m results in exactly 90 deg rotation of the two vertical bars, which is a singular configuration making this method to fail.
- Manipulator with three 3 × -infinity joints, see Fig. 4 and Table 2. The displacement of the tip of the mechanism is chosen such that the rotation of each of the joints is 45 deg (the first joint in positive z-direction, the other joints in negative y-direction). This case was also introduced in Ref. [35], and the 3 × -infinity joint has been developed in Ref. [9].
- T-flex: a flexure-based hexapod with 12 identical spherical joints, see Figs. 5 and 6 and Table 3. Each spherical joint consists of two serial stacked groups with each group comprising three folded leaf springs in parallel. Their z-axes are aligned with their corresponding upper arm. The six lower spherical joints are combined with a folded leafspring that constrains the rotation around the local z axis of that spherical joint. The six revolute joints at the bottom are assumed to be ideal joints. Two cases are analyzed in which each side of each folded leafspring is modeled with two and with four elements, respectively. In deformed configuration, the rotation of each of the six rotational joints at the bottom is 20 deg, in counterclockwise direction seen from outside (indicated by the dotted arrows in Fig. 5). This mechanism is described in Ref. [39] and the spherical joint in Ref. [10].

The required joint-data for the four joints (cross-flexure, 3 × -infinity, spherical joint, and folded leafspring) are fitted by fourth-order polynomials. KSD-it3 cannot be executed on the T-flex as the intended motion of the folded leafspring cannot be approximated by an ideal joint. Instead a linear expression is used to imprecisely approximate this motion.

The performance of the KSD-method is compared to the performance of a conventional method. The theoretical background of this reference method is that of the multibody software SPACAR [40], as this software has often been used to model flexure-based mechanisms [9,10,39]. The solver initially tries to solve the full displacement in one step and continues till the error in forces is below a certain lower threshold. If the error in forces is more than a certain upper threshold or the current iteration step cannot be computed, then the step-size in the displacement is reduced by a factor of 2 as explained in more detail in Appendix B of Ref. [35]. The KSD-method is implemented within the same theoretical background. All algorithms have been implemented in Matlab2017b for a fair comparison between the reference method and the KSD-method. The mechanisms are modeled using the beam element that is described in Appendix A of Ref. [35].

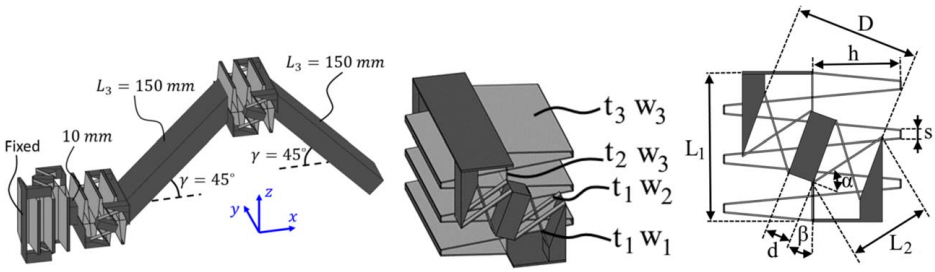

Fig. 4 Manipulator in undeformed configuration with lengths and orientations of the links and 3 × -infinity joint

Table 2 Dimensions of the 3 × -infinity joint

Width inner leaf spring cross flex, w_1	12 mm
Width outer leaf spring cross flex, w_2	6 mm
Width inner leaf springs, w_3	60 mm
Thickness leaf springs, t_1, t_2, t_3	0.45 mm
Length joint, L_1	45 mm
Length side leaf springs, L_2	38.2 mm
Horizontal size inner leaf springs, h	25 mm
Angle leaf springs cross flex, α	53 deg
Angle cross flex, β	20 deg
Length of part between cross flexs, d	8 mm
Total length cross flexs, D	38 mm
Length rigid part between inner leaf springs, s	2.5 mm

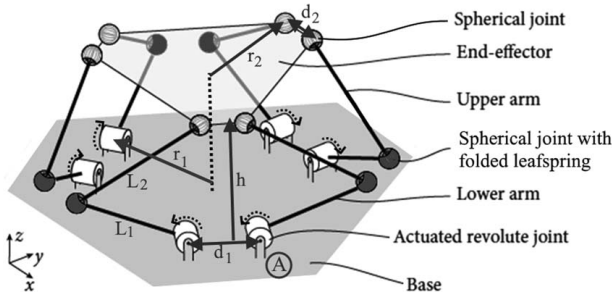

Fig. 5 T-flex

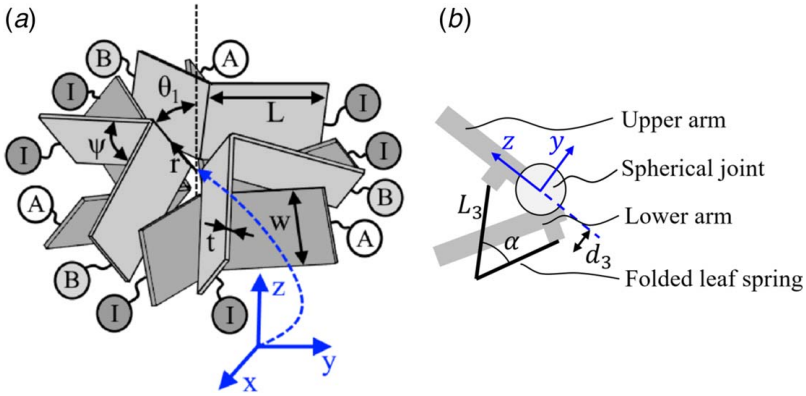

Fig. 6 (*a*) Spherical joint, consisting of two identical serial stacked groups of three folded leafsprings. "A" indicates connections of the dark group to one of the rigid links, "B" indicates the connections of the light group to the other rigid link, and "I" are rigid connections to one intermediate ring. (*b*) Dimensions and position of folded leaf spring.

For all the four cases, it was found that the computation of updates in the configuration could be performed by neglecting the geometric part of the stiffness matrices, i.e., only using the material part of the stiffness matrix. Table 4 shows that the computations without these terms are much more efficient in the framework of SPACAR. The convergence plot in Fig. 7 shows the most important reason for this increase in efficiency: without the geometric term, the full displacement can be added in one step, but with the geometric term the displacement had to be split in four steps for convergence. Another reason is that the computation of the geometric part of the stiffness matrix takes a significant amount of time. In the remaining part of this paper, the geometric part of the stiffness matrix is not used during the update of positions, not for the reference method and not for the iterative updates in the KSD-method.

Figure 8 shows the computation times per step of the KSD-method. It shows that the reduction in computation time is in general more significant for more complex mechanisms, i.e., the mechanisms with a higher number of degrees-of-freedom. The figure also indicates that the computation time of the steps which only update the interface coordinates is almost negligible. KSD-reduced is the most efficient method and reduces the computation time up to a factor of 90 with respect to the conventional method. KSD-full gives a reduction up to a factor 1.8 for the most complex mechanism (T-flex 4). The time reduction for the manipulator is much higher. This has to do with the fact that the 3 × -infinity joints in this manipulator are very stiff in the unintended directions, and therefore the approximation of the configuration is already very accurate after step 4, such that step 5 can be performed in only a few iterations. The computation times by KSD-it2 and KSD-it3 are on average a factor of 1.75 higher than the computation time of KSD-it1 which uses joint-data. This indicates that the computation time in the KSD-method can be reduced by a factor of 1.75 using joint-data. The computation time of KSD-it2 and KSD-it3 is comparable, indicating that the required computation time is insensitive to the accuracy of the estimation in step 1.

Table 3 Dimensions of T-flex

Center to revolute joints, r_1	250 mm
Distance to revolute joints, d_1	75 mm
Distance to spherical joints, d_2	50 mm
Length lower arm, L_1	251 mm
Length upper arm, L_2	304 mm
Undeformed height end-effector, h_2	265 mm
Length side folded leafspring, L_3	40 mm
Angle folded leafspring, α	50 deg
Distance folded leafspring to z axis spherical joint, d_3	15 mm
Length leaf springs, L	27.4 mm
Distance center to center folds, r	11 mm
Angle folded leaf springs, ψ	86 deg
Angle between fold and z axis, θ_1	30 deg
Width leaf spring, w	15 mm
Thickness leaf spring, t	0.4 mm
Counterclockwise rotation of the upper layer	25 deg

Table 4 Computation times in seconds of the reference method with and without geometric stiffness

	With geom.	Without geom.
Four-bar (305 DOF)	4.9	1.0
Manipulator (249 DOF)	48.6	5.0
T-flex 2 (1518 DOF)	333	39.9
T-flex 4 (3246 DOF)	1951	188

Note: The number of degrees-of-freedom is given as a measure for the complexity of the mechanism.

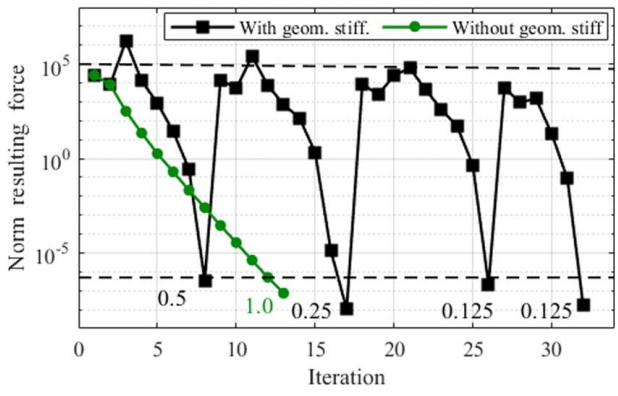

Fig. 7 Convergence of the conventional method for the four-bar mechanism. The numbers indicate the size of the step that is converged. The upper dotted line indicates the upper threshold (if the error is above this norm, the step-size is reduced by a factor of 2) and the lower dotted line indicates the lower threshold (if the error is below this line the current step is converged).

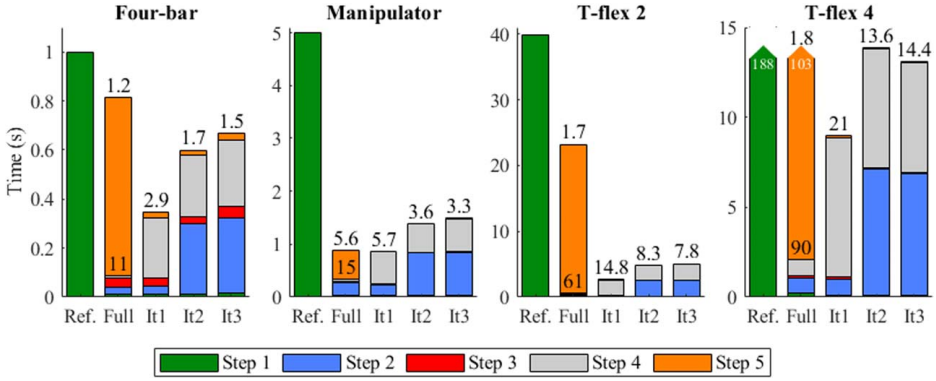

Fig. 8 Computation time per step for the reference without geometric stiffness and the KSD-method (KSD-reduced is equivalent to the first four steps of KSD-full). The black numbers indicate the time reduction factor with respect to the reference. The plot of T-flex 4 is zoomed in, and the times of the reference and KSD-full are 188 and 103 s, respectively.

Table 5 Influence of dimension parameters on rotation of elements during intended motion

Joint	Range of motion	Significant influence	Negligible influence
Cross-flexure	[−30 deg, 30 deg]	α, [30 deg, 60 deg]	t/L, [0.001, 0.05] w_1/L, [0.1, 1] w_2/w_1, [0.25, 1]
3 × -infinity	[−45 deg, 45 deg]	β, [30 deg, 120 deg] D/L, [0.5, 1]	d/D, [0, 0.3] h/L, [0.2, 0.8] s/L, [0, 0.1] α, [30 deg, 60 deg] t_1/L, [0.001, 0.05] t_i/t_1, [0.5, 2], $i = 2, 3$ w_1/L, [0.2, 4] w_2/w_1, [0.05, 1] w_3/w_2, [0.25, 1]
Spherical joint	Tip-tilt:[a] [−30 deg, 30 deg] Pan: [−15 deg, 15 deg]	θ_1, [25 deg, 45 deg] ψ, [70 deg, 100 deg] r/L, [0.1, 0.5]	w/L, [0.1, 0.5] t/w, [0.01, 0.1]

Note: Significant influence means that element rotations have more than 10% deviation. For each dimension parameter, the range for common values is given.
[a]Tip-tilt is rotation around the x and y axes. Pan is rotation around the z axis.

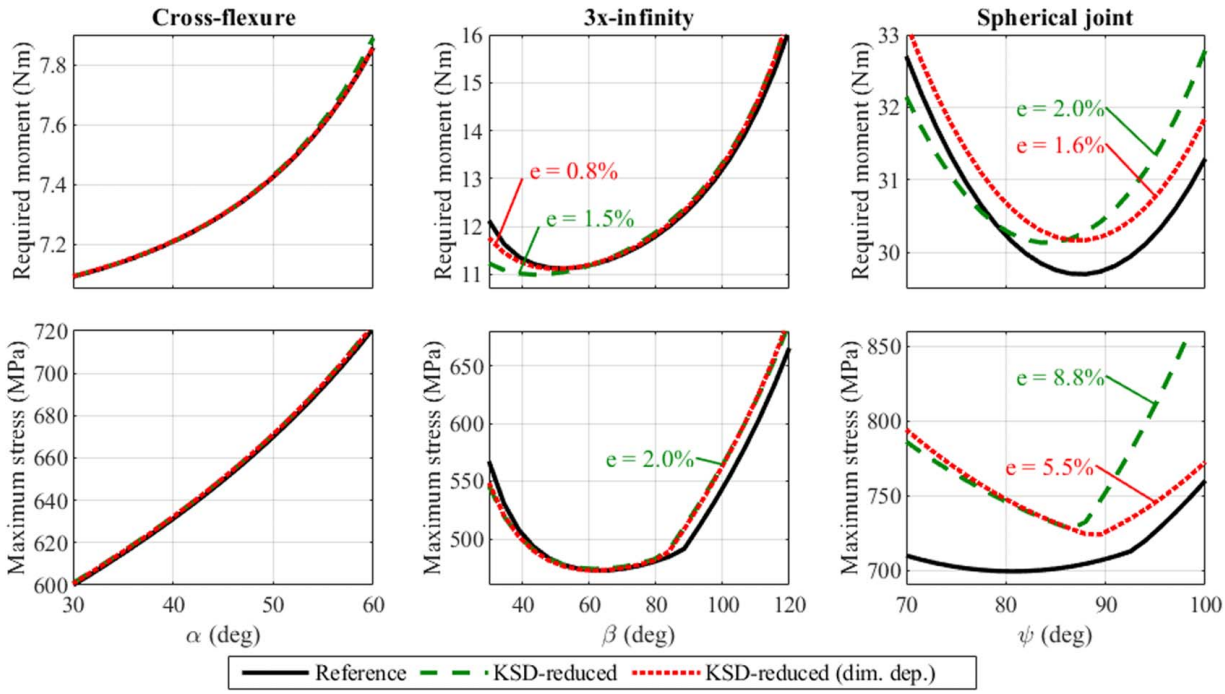

Fig. 9 Analysis results for three flexure joints in which one dimension is varied

5 Accuracy Results

This section evaluates the accuracy of the KSD-method and its dependency on variations of dimensions of the flexure joints. In step 2 of KSD-full, KSD-reduced, and KSD-it1, the element configuration is computed by element orientation-based bodies. In this method, the element rotations are obtained as a function of the intended deformation based on a least-square fit on simulation results as explained in Appendix A. For each of the three joints, it has been analyzed which dimension parameters have a significant influence on the rotations of the elements, and the results are given in Table 5. A significant influence means more than 10% deviation of the rotations for common values of the dimension parameters and common values of motion. In general, the width and thickness of flexures have a negligible influence. For the two joints with a simple motion (cross-flexure and 3×-infinity), only a limited number of dimension parameters have influence. For the spherical joint with a complex three-dimensional motion, more dimensions have influence.

Figure 9 shows the results for three introduced flexure joints on which one of these dimensions is varied. It shows the required applied moment and the maximum stress for a certain intended deformation:

- The cross-flexure is rotated 30 deg.
- The 3×-infinity is rotated 45 deg.
- The spherical joint is rotated 10 deg around the z axis and then 20 deg around the y axis, and the graph only shows the required moment around the z axis.

The results are given for the reference method, for KSD-reduced which uses joint-data obtained with the default dimensions and for KSD-reduced which uses joint-data that is explicitly made dependent on the dimension parameters with significant influence based on fourth-order polynomials.

The results are accurate as there is almost no deformation in the unintended directions in these cases. The only significant error is the stress computed for the spherical joint with KSD-reduced, but this error is reduced by making the joint-data dependent on the dimension parameter. However, to obtain this accuracy, a fourth-order polynomial was required for the joint-data, and using a third-order polynomial gives similar results to the case where the data did not depend on the dimension parameters. This indicates that the stress result of KSD-reduced can be quite sensitive to the accuracy of the joint-data.

Figure 10 shows the results for the mechanisms with the deformations as described in Sec. 4, in which dimension parameters are varied. The forces are:

- For the four-bar: the required force applied at the center of the top bar.
- For the manipulator: the total required force on the tip.

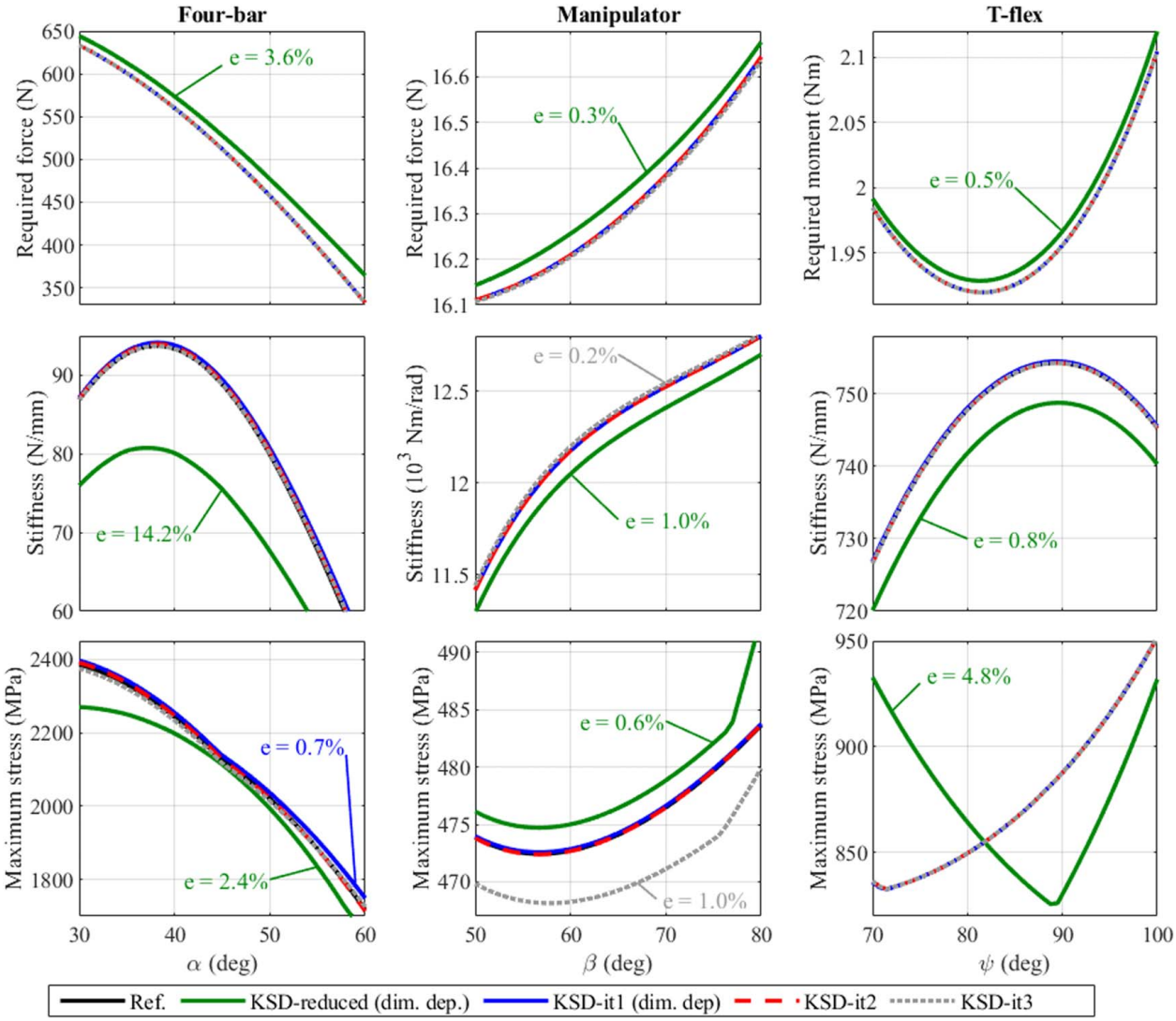

Fig. 10 Analysis results in which one dimension of the flexure joints is varied. For some lines the average error is given. The reference line is often hidden behind the line of KSD-it1 (dim. dep).

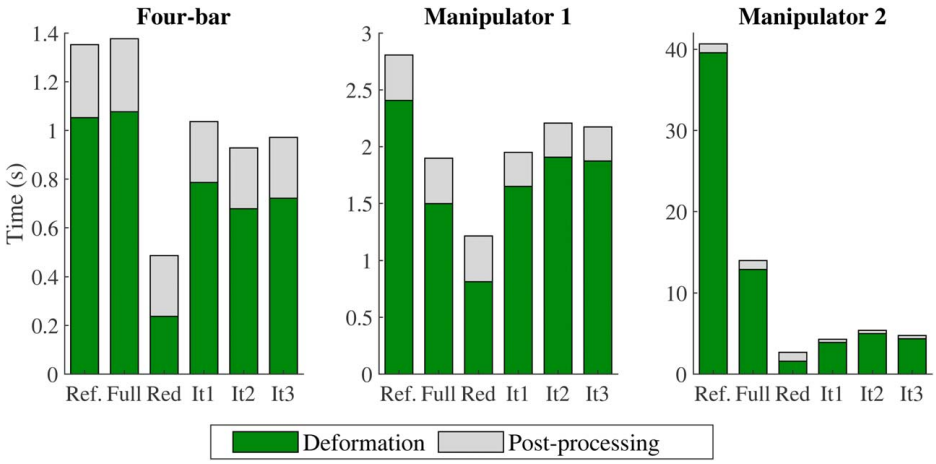

Fig. 11 Average computation time of one function evaluation during the optimizations for the different methods, split in the time for the computation of the deformation and the post-processing

- For the T-flex: the required moment applied by one of the revolute joints, indicated by "A" in Fig. 5.

The stiffness results are in unintended directions where the motion in the intended direction of the mechanism is constrained at the displacements given in Sec. 4:

- For the four-bar: the stiffness at the center of the top bar in the vertical direction.
- For the manipulator: the rotational stiffness of the tip around the y axis.
- For the T-flex: the vertical stiffness at the center of the end-effector.

In these stiffness results, the geometric stiffness terms are taken into account although it was not used to find the deformed configuration. The computation of the geometric stiffness and the stresses is not part of the KSD-method and it is therefore not included in the computation times in Fig. 8.

6 Optimizations

The performance of the KSD-method in a design optimization is tested with the four-bar and the manipulator.

In the four-bar, the flexure thickness (t) and the length of the flexures (L) were used as design variables. The other dimensions were the same as in Secs. 4 and 5, see Table 1. The four-bar was optimized for stiffness in the z-direction at the center of the top bar under 0.29 m displacement. The stress in the leafsprings was limited to 600 MPa.

In the manipulator, the thickness of the inner flexures (t_1) and the angle of both rigid elements (γ) were used as design variables. The length of both diagonal rigid links (L_3) was adjusted in each design to keep the undeformed end-position at coordinate (0.35, 0, 0). The other dimensions were the same as in Secs. 4 and 5, see Table 2. The manipulator was optimized for rotational stiffness around the z axis when the end-effector is displaced to the coordinate: (0.2, 0.2, 0.1). The stress in the leafsprings is limited to 600 MPa. The optimization is executed twice for each method, once with each flexure modeled by one beam element (similar to the case of the previous sections) and once with each flexure modeled by two beam elements.

Both cases are optimized using the covariance matrix adaptation evolution strategy (CMA-ES) algorithm [41], a genetic algorithm, with a population size of 10. The optimizations required 400–600 function evaluations for convergence, independent of the method. Each function evaluation includes the computation of one deformed configuration. Figure 11 shows the average computation times per function evaluation. The time reduction of the KSD-method with respect to the reference method is lower than the time reduction shown in Fig. 8. One reason is that the displacement of the manipulator is smaller than the case defined in Secs. 4 and 5 such that it can be computed faster by the reference method. Another reason is that the KSD-method performs especially well on mechanisms with common flexure dimensions. This is because the kinematic approximation of the indented motion becomes worse if the intended motion is not much more compliant than the unintended motion. This happens for example if the flexures are very thick or very short, which will be the case in some trial-designs during the optimization. This is the reason that the KSD-full method is even slower than the reference method in case of the four-bar. This also explains that the difference between the computation time of KSD-it1 is sometimes slower than KSD-it2 and KSD-it3.

Table 6 shows the resulting optimized design parameters and support stiffness. All optimizations converged to the same global optimum although there are some slightly deviations. KSD-it3 gives relatively large errors on the four-bar which is probably due to the large parasitic error motion of cross-flexures. KSD-reduced

Table 6 Accuracy of the optimizations, optimized design parameters, and support stiffness

	Four-bar			Manipulator 1			Manipulator 2		
	t(mm)	L(mm)	K(N/m)	t_1(mm)	γ(deg)	K(kNm)	t_1(*mm*)	γ(deg)	K(kNm)
Ref./KSD-full	0.272	83.4	100.7	0.55	48.1	15.4	0.59	52.8	6.47
KSD-red	0.270 (1.0%)	82.2 (1.5%)	96.0 (4.7%)	0.55 (0.6%)	48.2 (0.2%)	15.4 (0.0%)	0.58 (1.1%)	51.9 (1.7%)	6.94 (7.2%)
KSD-it1	0.270 (0.7%)	82.6 (1.0%)	100.7 (0.4%)	0.55 (0.0%)	48.3 (0.5%)	15.4 (0.1%)	0.59 (0.0%)	52.0 (1.5%)	6.80 (5.1%)
KSD-it2	0.270 (0.6%)	82.7 (0.9%)	101.1 (0.4%)	0.53 (3.9%)	47.8 (0.6%)	14.9 (3.4%)	0.57 (3.5%)	53.0 (0.4%)	6.43 (0.7%)
KSD-it3	0.254 (6.8%)	76.6 (8.3%)	91.5 (9.2%)	0.53 (4.0%)	49.0 (2.0%)	14.9 (3.5%)	0.57 (3.5%)	52.8 (0.0%)	6.43 (0.6%)

Note: The numbers between brackets give the accuracy with respect to the reference method.

results in a significant error in the support stiffness, but still gives a good result for the design parameters. The errors of the different KSD-methods for the manipulator are significantly smaller than the error that is made by modeling all the flexures by only one beam element instead of two beam elements.

7 Conclusions

The KSD-method, introduced in Ref. [35], efficiently computes large deformed configurations of flexure-based mechanisms. Existing variants of the KSD-method utilize a priori obtained joint-data of flexure joints to efficiently obtain its internal deformed configuration. This paper shows that the KSD-method can also be used without these joint-data, and this increases the required computation time but it is still significantly faster than a conventional method.

The computational efficiency and accuracy are compared to a reference method. This reference method is an implementation of the conventional method SPACAR in MATLAB. It was found that most static computations on flexure-based mechanisms can be performed without using the geometric part of the stiffness matrix in the updates of the configuration. This reduces the computation time of the conventional method SPACAR significantly.

The performances of different variants of the KSD-method that vary in accuracy and computation time were verified. This is verified by computations on three mechanisms and three joints. One of these joints is a spatial spherical joint proving that the KSD-method also works for full spatial joints.

The variant "KSD-full" is as accurate as the reference method and reduces the computation time up to a factor of 5.6. The variant "KSD-reduced" is up to 90 times faster than the reference method but can result in errors up to 15%. The variant "KSD-it1" reduces the computation time up to a factor of 21 and it gives errors below 1%. The variants "KSD-it2" and "KSD-it3" which uses less and no joint-data, respectively, are up to 14 times faster than the reference method, resulting in errors below 1%.

The KSD-method is especially helpful in design optimizations as it does not use data that are sensitive to the dimensions of the analyzed mechanism, which is shown by several design optimizations.

Funding Data

- This work is part of the research program HTSM 2017 with project number 16210, which is partly financed by the Netherlands Organisation for Scientific Research (NWO).

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request. The authors attest that all data for this study are included in the paper.

Appendix A: Spatial Element Orientation-Based Body

This appendix summarizes how the internal configuration of a spatial element orientation-based body is found. Reference [35] gives more details about this method for planar joints. This appendix shows that for bodies with a full spatial intended motion, not only the orientations of the elements but also the orientations of the nodes are required.

In the element orientation-based body, the deformed configuration is approximated based on the rotations of the elements. This approximation is referred to as near configuration, see Fig. 12. It is the configuration that is obtained in step 2 of the KSD-method

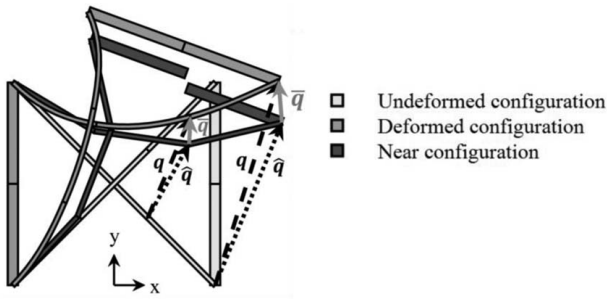

Fig. 12 Element orientation-based body indicating the three different types of displacement for one element

(for the variants that use element orientation-based bodies: KSD-full, KSD-reduced, and KSD-it1). The near configuration is obtained as follows:

- The intended motion of the element orientation-based body is obtained in step 1 of the KSD-method.
- The rotations of the elements are obtained based on this intended motion and the least-square fit that is a priori obtained by simulation-data.
- The locations of the elements are obtained starting from one of the interface points. The local displacement of this interface point is defined to be zero which defines the positions of the elements that are attached to this interface point. The locations of the other elements are obtained by placing them in chains to the previous elements, assuming all elements to be undeformed, so it is only rotated and displaced as shown in Fig. 12.

As the orientations of the elements are known in the near configuration, the stiffness matrices of the elements can be defined by rotating the undeformed stiffness matrices to obtain the equation:

$$\boldsymbol{F}_{\text{all}} = \begin{Bmatrix} \boldsymbol{F}_1 \\ \vdots \\ \boldsymbol{F}_N \end{Bmatrix} = \begin{bmatrix} \boldsymbol{R}_1\boldsymbol{K}_1^{(\text{undef})}\boldsymbol{R}_1^T & 0 & 0 \\ & \ddots & \\ 0 & 0 & \boldsymbol{R}_N\boldsymbol{K}_N^{(\text{undef})}\boldsymbol{R}_N^T \end{bmatrix} \begin{Bmatrix} \bar{\boldsymbol{q}}_1 \\ \vdots \\ \bar{\boldsymbol{q}}_N \end{Bmatrix} = \boldsymbol{K}_{\text{all}}\bar{\boldsymbol{q}}_{\text{all}} \tag{A1}$$

where $\boldsymbol{F}_k$ are the 12 boundary-forces and moments on element k, $\boldsymbol{K}_k^{(\text{undef})}$ is its stiffness matrix in the undeformed configuration, $\boldsymbol{R}_k$ is the rotation matrix of element k that defines the rotation to the near configuration, and $\bar{\boldsymbol{q}}_k$ are the 12 displacements with respect to the near configuration (see Fig. 12). As $\bar{\boldsymbol{q}}_{\text{all}}$ is the difference between the displacement to the deformed configuration ($\boldsymbol{q}_{\text{all}}$) and the displacement to the near configuration ($\hat{\boldsymbol{q}}_{\text{all}}$), we can rewrite Eq. (A1) to

$$\boldsymbol{F}_{\text{all}} = \boldsymbol{K}_{\text{all}}\boldsymbol{q}_{\text{all}} - \boldsymbol{K}_{\text{all}}\hat{\boldsymbol{q}}_{\text{all}} \tag{A2}$$

This equation is in terms of 12 displacements per element (six displacements for both sides). However, elements that are connected share a node. Therefore, a Boolean matrix $\boldsymbol{L}$ is defined to express the displacements of the sides of the elements in terms of the displacements of the nodes:

$$\boldsymbol{q}_{\text{all}} = \boldsymbol{L}\boldsymbol{q}_{\text{nodes}}, \boldsymbol{F}_{\text{nodes}} = \boldsymbol{L}^T\boldsymbol{F}_{\text{all}} \tag{A3}$$

where $\boldsymbol{q}_{\text{nodes}}$ is composed of the displacements of the nodes and $\boldsymbol{F}_{\text{nodes}}$ is composed of the forces on the nodes. Substituting Eq. (A3) into Eq. (A2) gives

$$\boldsymbol{F}_{\text{nodes}} = \boldsymbol{K}_{\text{nodes}}\boldsymbol{q}_{\text{nodes}} + \hat{\boldsymbol{F}}_{\text{nodes}} \tag{A4}$$

where

$$\boldsymbol{K}_{\text{nodes}} = \boldsymbol{L}^T\boldsymbol{K}_{\text{all}}\boldsymbol{L}, \hat{\boldsymbol{F}}_{\text{nodes}} = -\boldsymbol{L}^T\boldsymbol{K}_{\text{all}}\hat{\boldsymbol{q}}_{\text{all}} \tag{A5}$$

This is the stiffness relation that is used to update the interface positions in step 3 of the KSD-method and to update the internal configuration in step 4 of the KSD-method.

Equation (A4) can be obtained for joints with planar intended deformation, but if the motion is spatial the large rotations in $\boldsymbol{q}_{\text{all}}$, $\boldsymbol{q}_{\text{nodes}}$, and $\hat{\boldsymbol{q}}_{\text{all}}$ should be described by parameters that can describe large rotations, e.g., Euler parameters or Euler angles. Using Euler parameters as an example, this means that the flexible displacement $\bar{\boldsymbol{q}}_{\text{all}}$ is a small difference in Euler parameters. However, the stiffness matrix $\boldsymbol{K}_{\text{all}}$ relates forces to rotations around the x, y, and z axes. This means that for each node, the small difference in Euler parameters should be rewritten to small differences in rotations around the x, y and z axes. The required relations to do this do exist, see, for example, Ref. [42]. The relation between variations in Euler parameters $\{\delta\lambda_0, \delta\boldsymbol{\lambda}^T\}^T = \{\delta\lambda_0, \delta\lambda_1, \delta\lambda_2, \delta\lambda_3\}^T$ and variations in these rotations $\delta\boldsymbol{\theta} = \{\delta\theta_x, \delta\theta_y, \delta\theta_z\}^T$ are

$$\delta\boldsymbol{\theta} = 2\begin{bmatrix} -\boldsymbol{\lambda} & \lambda_0 1 + \tilde{\boldsymbol{\lambda}} \end{bmatrix} \begin{Bmatrix} \delta\lambda_0 \\ \delta\boldsymbol{\lambda} \end{Bmatrix} = 2\boldsymbol{\Lambda} \begin{Bmatrix} \delta\lambda_0 \\ \delta\boldsymbol{\lambda} \end{Bmatrix}, \quad \tilde{\boldsymbol{\lambda}} = \begin{bmatrix} 0 & -\lambda_3 & \lambda_2 \\ \lambda_3 & 0 & -\lambda_1 \\ -\lambda_2 & \lambda_1 & 0 \end{bmatrix} \tag{A6}$$

However, the matrix $\boldsymbol{\Lambda}$ depends on the orientation of the node. Therefore, this orientation has to be estimated on beforehand. This means that the orientations of the nodes should be obtained similar to the orientations of the elements, i.e., based on a least-square polynomial fit on the simulation results.

References

[1] Smith, S. T., 2000, *Flexures: Elements of Elastic Mechanisms*, CRC Press, Boca Raton, FL.

[2] Howell, L. L., 2001, *Compliant Mechanisms*, John Wiley & Sons, New York.

[3] Richard, M., and Clavel, R., 2011, "Concept of Modular Flexure-Based Mechanisms for Ultra-High Precision Robot Design," Mech. Sci., **2**(1), pp. 99–107.

[4] Tolou, N., Henneken, V. A., and Herder, J. L., 2010, "Statically Balanced Compliant Micro Mechanisms (SB-MEMS): Concepts and Simulation," International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Montreal, Canada, Aug. 15–18.

[5] Wan, S., and Xu, Q., 2016, "Design and Analysis of a New Compliant XY Micropositioning Stage Based on Roberts Mechanism," Mech. Mach. Theory, **95**, pp. 125–139.

[6] Le Chau, N., Dang, V. A., Le, H. G., and Dao, T.-P., 2017, "Robust Parameter Design and Analysis of a Leaf Compliant Joint for Micropositioning Systems," Arab. J. Sci. Eng., **42**(11), pp. 4811–4823.

[7] Wu, Z., and Xu, Q., 2018, "Survey on Recent Designs of Compliant Micro-/Nano-Positioning Stages," Actuators, **7**(1), p. 5.

[8] Wiersma, D., Boer, S., Aarts, R. G., and Brouwer, D. M., 2014, "Design and Performance Optimization of Large Stroke Spatial Flexures," ASME J. Comput. Nonlinear Dyn., **9**(1), p. 011016.

[9] Naves, M., Brouwer, D. M., and Aarts, R. G. K. M., 2017, "Building Block-Based Spatial Topology Synthesis Method for Large-Stroke Flexure Hinges," ASME J. Mech. Rob., **9**(4), p. 041006.

[10] Naves, M., Aarts, R. G. K. M., and Brouwer, D. M., 2019, "Large Stroke High Off-Axis Stiffness Three Degree of Freedom Spherical Flexure Joint," Precis. Eng., **56**, pp. 422–431.

[11] Hao, G., Yu, J., and Li, H., 2016, "A Brief Review on Nonlinear Modeling Methods and Applications of Compliant Mechanisms," Front. Mech. Eng., **11**(2), pp. 119–128.

[12] Ling, M., Howell, L. L., Cao, J., and Chen, G., 2020, "Kinetostatic and Dynamic Modeling of Flexure-Based Compliant Mechanisms: A Survey," ASME Appl. Mech. Rev., **72**(3), p. 030802.

[13] Bilancia, P., and Berselli, G., 2021, "An Overview of Procedures and Tools for Designing Nonstandard Beam-Based Compliant Mechanisms," Comput. Aided Des., **134**, p. 103001.

[14] Bathe, K. J., and Bolourchi, S., 1979, "Large Displacement Analysis of Three-Dimensional Beam Structures," Int. J. Numer. Methods Eng., **14**(7), pp. 961–986.

[15] Pai, P., and Palazotto, A., 1996, "Large-Deformation Analysis of Flexible Beams," Int. J. Solids Struct., **33**(9), pp. 1335–1353.

[16] Bisshopp, K., and Drucker, D., 1945, "Large Deflection of Cantilever Beams," Q. Appl. Math., **3**(3), pp. 272–275.

[17] Zhang, A., and Chen, G., 2013, "A Comprehensive Elliptic Integral Solution to the Large Deflection Problems of Thin Beams in Compliant Mechanisms," ASME J. Mech. Rob., **5**(2), p. 021006.

[18] Cammarata, A., Lacagnina, M., and Sequenzia, G., 2019, "Alternative Elliptic Integral Solution to the Beam Deflection Equations for the Design of Compliant Mechanisms," Int. J. Interact. Des. Manuf., **13**(2), pp. 499–505.

[19] Xu, K., Liu, H., and Xiao, J., 2021, "Static Deflection Modeling of Combined Flexible Beams Using Elliptic Integral Solution," Int. J. Non-Linear Mech., **129**, p. 103637.

[20] Frisch-Fay, R., 1962, *Flexible Bars*, Butterworths, London.

[21] Sen, S., and Awtar, S., 2013, "A Closed-Form Nonlinear Model for the Constraint Characteristics of Symmetric Spatial Beams," ASME J. Mech. Des., **135**(3), p. 031003.

[22] Sen, S., 2013, "Beam Constraint Model: Generalized Nonlinear Closed-form Modeling of Beam Flexures for Flexure Mechanism Design," Ph.D. thesis, University of Michigan, Ann Arbor, MI.

[23] Turkkan, O. A., and Su, H.-J., 2017, "A General and Efficient Multiple Segment Method for Kinetostatic Analysis of Planar Compliant Mechanisms," Mech. Mach. Theory, **112**, pp. 205–217.

[24] Chen, G., Ma, F., Hao, G., and Zhu, W., 2019, "Modeling Large Deflections of Initially Curved Beams in Compliant Mechanisms Using Chained Beam Constraint Model," ASME J. Mech. Rob., **11**(1), p. 011002.

[25] Nijenhuis, M., Meijaard, J. P., and Brouwer, D. M., 2020, "A Spatial Closed-Form Nonlinear Stiffness Model for Sheet Flexures Based on a Mixed Variational Principle Including Third-Order Effects," Precis. Eng., **66**, pp. 429–444.

[26] Bai, R., Chen, G., and Awtar, S., 2021, "Closed-Form Solution for Nonlinear Spatial Deflections of Strip Flexures of Large Aspect Ratio Considering Second Order Load-Stiffening," Mech. Mach. Theory, **161**, p. 104324.

[27] Chen, G., Zhang, Z., and Wang, H., 2018, "A General Approach to the Large Deflection Problems of Spatial Flexible Rods Using Principal Axes Decomposition of Compliance Matrices," ASME J. Mech. Rob., **10**(3), p. 031012.

[28] Bai, R., and Chen, G., 2021, "Modeling Large Spatial Deflections of Slender Beams of Rectangular Cross Sections in Compliant Mechanisms," ASME J. Mech. Rob., **13**(1), p. 011021.

[29] Chase Jr., R. P., Todd, R. H., Howell, L. L., and Magleby, S. P., 2011, "A 3-D Chain Algorithm With Pseudo-Rigid-Body Model Elements," Mech. Based Des. Struct. Mach., **39**(1), pp. 142–156.

[30] Liang, Y., Lee, H., Lim, S., Lin, W., Lee, K., and Wu, C., 2002, "Proper Orthogonal Decomposition and Its Applications—Part I: Theory," J. Sound Vib., **252**(3), pp. 527–544.

[31] Mignolet, M. P., Przekop, A., Rizzi, S. A., and Spottswood, S. M., 2013, "A Review of Indirect/Non-Intrusive Reduced Order Modeling of Nonlinear Geometric Structures," J. Sound Vib., **332**(10), pp. 2437–2460.

[32] Baur, U., Benner, P., and Feng, L., 2014, "Model Order Reduction for Linear and Nonlinear Systems: A System-Theoretic Perspective," Arch. Comput. Methods, **21**(4), pp. 331–358.

[33] Degroote, J., Vierendeels, J., and Willcox, K., 2010, "Interpolation Among Reduced-Order Matrices to Obtain Parameterized Models for Design, Optimization and Probabilistic Analysis," Int. J. Numer. Methods Fluids, **63**(2), pp. 207–230.

[34] Kuether, R. J., and Allen, M. S., 2014, "Craig-Bampton Substructuring for Geometrically Nonlinear Subcomponents," *Dynamics of Coupled Structures*, M. Allen, R. Mayes, and D. Rixen, eds., Springer, New York, pp. 167–178.

[35] Dwarshuis, K. S., Aarts, R. G. K. M., Ellenbroek, M. H. M., and Brouwer, D. M., 2020, "Kinematically Started Efficient Position Analysis of Deformed Compliant Mechanisms Utilizing Data of Standard Joints," Mech. Mach. Theory, **152**, p. 103911.

[36] Blanding, D. L., 1999, *Exact Constraint: Machine Design Using Kinematic Principles*, American Society of Mechanical Engineers, New York.

[37] Haringx, J., 1949, "The Cross-Spring Pivot as a Constructional Element," Flow Turbul. Combust., **1**(1), p. 313.

[38] Xu, P., Jingjun, Y., Guanghua, Z., Shusheng, B., and Zhiwei, Y., 2008, "Analysis of Rotational Precision for an Isosceles-Trapezoidal Flexural Pivot," ASME J. Mech. Des., **130**(5), p. 052302.

[39] Naves, M., Hakvoort, W. B. J., Nijenhuis, M., and Brouwer, D. M., 2020, "T-Flex: A Large Range of Motion Fully Flexure-Based 6-DOF Hexapod," Euspen's 20th International Conference & Exhibition, Geneva, Switzerland, June 8–12.

[40] Jonker, J., and Meijaard, J., 1990, "SPACAR—Computer program for dynamic analysis of flexible spatial mechanisms and manipulators," *Multibody Systems Handbook*, W. Schiehlen, ed., Springer, New York, pp. 123–143.

[41] Hansen, N., 2006, "The CMA Evolution Strategy: A Comparing Review," Towards Evol. Comput., **192**, pp. 75–102.

[42] Schwab, A. L., and Meijaard, J. P., 2006, "How to Draw Euler Angles and Utilize Euler Parameters," International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Philadelphia, PA, Sept. 10–13.