Mechanism and Machine Theory xxx (xxxx) xxx

Contents lists available at ScienceDirect

Mechanism and Machine Theory

journal homepage: www.elsevier.com/locate/mechmachtheory

Research paper

Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints

Koen Dwarshuis*, Ronald Aarts, Marcel Ellenbroek, Dannis Brouwer

Faculty of Engineering Technology, University of Twente, Enschede, the Netherlands

ARTICLE INFO

Article history:
Received 24 October 2019
Revised 9 April 2020
Accepted 10 April 2020
Available online xxx

Keywords:
Model reduction
Large deflections
Kinematic modeling
Design optimization
Flexures
Compliant mechanism

ABSTRACT

Topology optimization of a flexure-based mechanism requires the properties of the mechanism in several deformed configurations. This paper presents a fast and accurate method to compute these configurations. It is generally applicable on mechanisms with complex standard flexure joints. First kinematic equations of the mechanism are derived by allowing the mechanism to move only in the directions for which it is designed. Secondly the configurations of the joints are approximated based on the rotations of the elements by which the joints are modeled. These orientations are obtained by a parameterization based on a priori knowledge of standard flexure joints. Finally, the resulting approximation is used as initial guess to obtain the configuration accurately, after which relevant properties like stiffness can be derived. For a manipulator with three complex joints the computation time was reduced up to a factor of 65 compared to a conventional method. When for optimization purposes an approximation is acceptable, the computation time can be reduced by a factor of 600, using a linear description of the deformation that remains in the first part of the method.

1. Introduction

Flexure-based mechanisms have become increasingly complex, especially for the cases where a large motion should be realized. These mechanisms typically consist of multiple flexure joints, and each flexure joint can consist of more than thirty leaf springs, see Fig. 1 for an example [1]. Moreover when large deformation is considered in the leaf springs, each leaf spring should be modeled by using multiple elements. In the end many elements are required to obtain an appropriate model of flexure based mechanisms.

Design optimization is common in flexure based design [1–5] as the performance of these mechanisms highly depends on the configuration of the mechanism and the geometry of the flexure joints. These optimizations require an analysis of the mechanical properties of the mechanism, e.g. the support stiffness, the maximum stress, the buckling load and the eigen frequencies of the mechanism. These properties should be evaluated over the full range of motion. Therefore a number of relevant deformed configurations is chosen for which these properties are evaluated in each optimization iteration step. Computing these relevant configurations currently requires the solution of nonlinear equations by iterative methods,

* Corresponding author at: P.O. Box 217, 7500 AE Enschede, the Netherlands.
E-mail address: k.s.dwarshuis@utwente.nl (K. Dwarshuis).

https://doi.org/10.1016/j.mechmachtheory.2020.103911

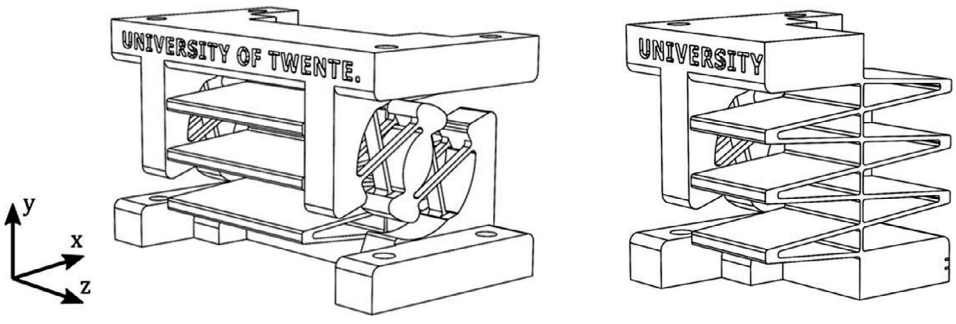

Fig. 1. 3x-Infinity joint: a state of the art flexure joint that facilitates one large rotation of 90° about the z-axis [1].

and with tens to hundreds of design iterations it is clear that these computations are taking a lot of time. Therefore it is advantageous to obtain the deformed configuration of a mechanism containing flexure joints more efficiently.

1.1. Literature

Pseudo rigid body models (PRBMs) [6] offer a simple representation of flexure mechanisms, allowing efficient computation of the configuration. These models consist of rigid parts and pivots. The stiffness of these mechanisms is modeled by torsional springs in each pivot. These models generally only describe the motion for which the mechanism is designed, which makes that the motion of the mechanism can be analyzed very quickly. However this is also limits the applicability of the model as the motion and stiffness in the other directions cannot be analyzed. Another limitation is that it is difficult to obtain accurate models, because it is difficult to tune pivot positions and their torsional stiffness and mass distribution [7–11], balancing between errors in kinematics and stiffness in the models. Appropriate values for the position, stiffness and mass depend on the mechanism configuration and loading conditions and are often obtained by using optimization techniques [10]. All in all, the pseudo rigid body models have not been shown to provide a simple but accurate 3D description of complex flexure based mechanisms.

Reference [12] presents a semi-analytic-deflection-method in which each flexure joint is initially modeled by an ideal hinge, similar to the pseudo rigid body method. This model is used in a kinematic analysis to obtain an initial configuration of the mechanism. This configuration is used to compute the deflection of the flexure joints. The deflections are used to update the kinematics of the mechanism, which is used in new deflection analyses. In this way a loop is built in which the accuracy is improved at each iteration. So similar to the pseudo rigid body models, this method also starts by only allowing the motion for which the mechanism is designed, but in this case the motion in the other directions is added in later iterations. The deflection analysis however is based on semi-analytical approaches which means that the technique cannot be applied for mechanisms containing complex flexure joints like the one in Fig. 1. So in order to reduce computation time of mechanisms with these complex joints, a more general approach is required, i.e. an approach that is applicable to joints of which an analytic solution is difficult to obtain.

Model order reduction techniques are very general approaches to reduce computation time of complex mechanisms. These methods are well established in linear structural mechanics. However, the large deformation of flexure joints cannot be described linearly. Some nonlinear model order reduction methods are available, an overview can be found in [13]. One example is the modal derivatives approach [14]. This approach is accurate in a larger deformation range than the linear methods, but for large deformation this method also lacks accuracy [15], such that it cannot be used to model flexible joints.

There are some data-driven techniques to obtain a nonlinear reduced order model. This requires that training simulations with a model of the mechanism are run. A set of configurations from these simulation (snapshots) is used to obtain a good basis for the reduced order model. For example Proper Orthogonal Decomposition [16,17] uses a singular value decomposition to obtain the most suitable modes to describe the deformed configurations. Design optimizations require knowledge of many designs, i.e. mechanisms with varying dimensions. The reduced order model depends on the dimensions. There are some methods to include such dimension variation in data-driven techniques, an overview can be found in [18]. A disadvantage of these reduced methods is that the accuracy is limited and that the accuracy can be sensitive to variations in the dimensions. Moreover, the data-driven techniques require the a priori computation of deformed configurations for many designs of the mechanism, which takes much computation time.

Global Modal Parameterization (GMP) and similar techniques [19–22] are model order reduction methods that describe the displacement of all nodes as the sum of two contributions. One contribution introduces large (mainly rigid) displacements which are a function of only a few independent parameters. The second contribution consist of small (mainly flexible) displacements. As this contribution is assumed small, it can be approximated by a linear model. A disadvantage of GMP is that the relations between the independent parameters and the large displacements are difficult to obtain. This is currently done in a cumbersome preprocessing step [19,20], or by a data-driven technique [21,22]. The main goal of GMP is to remove

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

the high eigen frequencies in order to increase the allowable time step in dynamic simulations. In this paper a similar technique will be used to improve static computations, i.e. a technique that first solves a large part of the motion and estimates the remaining part by a linear approximation.

All in all, there is no proper reduction technique to obtain a model for flexure based mechanisms that is accurate for large deformation and that does not require a priori obtained knowledge of the complete mechanism.

1.2. Approach

This paper presents a combination of two methods. In the first place a method to kinematically obtain the configuration of a flexure-based mechanism by only allowing the motion for which the mechanism is designed. This configuration is used as a starting point to obtain the real deformed configuration based on static equilibrium. In this way the need for a cumbersome iterative procedure to obtain the deformed configuration is avoided. This method will be referred to as the *Kinematically Started Deformation method* (KSD-method).

The motion for which the mechanism is designed is henceforth referred to as *intended motion/deformation*. In other papers this is referred to as degrees of freedom, despite the fact that the mechanisms can also move slightly in the other directions. This other motion (i.e. the motion in the support-directions) will be referred to as *unintended motion/deformation*. In order to distinguish intended and unintended deformation, it appears to be useful to consider flexure based mechanisms as an assembly of flexure joints and very stiff links. The distinction between intended and unintended deformation will also be made in each flexure joint.

In the KSD-method, each flexure joint is modeled by a small finite element model. The method requires the internal configurations of the deformed flexure joints in the mechanism to be obtained. This requires a time-consuming iterative procedure in general. However, this step can be applied much more efficiently, using the fact that flexure joints are typically designed according to standard assemblies, e.g. the cross flexure [23,24] the cart wheel [24,25], the butterfly hinge [25,26], the infinity hinge [2] or the 3x-Infinity hinge in Fig. 1 [1]. Other standard assemblies can be found in [27]. Joints based on these standard assemblies will be referred to as *standard flexure joints*.

Therefore it is considered to a feasible solution to setup a database that contains information about these standard flexure joints in order to speed up the static calculation of all mechanisms containing these standard flexure joints. In other words, we propose a data-driven technique. In the literature review above, it was stated that using a data-driven reduction technique requires much a priory computation time. However, the proposed data-driven technique only requires data of default parts, which means that the databases does not have to be updated before each design optimization. Another disadvantage of data-driven techniques mentioned above is that the accuracy is sensitive to variations in the dimensions of the model. This disadvantage is resolved by the second method in this paper, which is based on a priory obtained parameters that are valid throughout the full range of commonly encountered dimensions of the flexure joints.

The second method presented in this paper obtains the largely deformed configurations of standard flexure joints, based on a limited number of parameters that are not sensitive to changes in the dimensions of that joint. It appears that the orientations of the elements are a good choice for these parameters as the orientations are independent of most of the dimensions of the flexure joint and provide enough information to obtain a configuration that is close to the deformed configuration. A body that is described by this technique will be referred to as an *Element Orientation based Body* (EOB).

Once the internal configuration in the EOB is obtained, its stiffness matrix can be obtained and this stiffness matrix can be reduced by model order reduction. In this paper, the term *element* is reserved to refer to a modeling part of which the deformation can be described by analytic relations, e.g. beam elements or plate elements. A *body* is a modeling part that is based on a reduced finite element model. So each flexure joint is a body that is described by multiple elements.

The main idea of the *KSD-method* is to prevent the need for a large number of iterations to obtain the deformed configuration of a *mechanism*, where the main idea of the *EOB* is to prevent the need for an iterative approach to find the internal configuration of the *joints*. With this combination the required simulation time is reduced significantly.

The proposed method shares various concepts with the methods in the literature review. The KSD-method first solves for a main part of the deformation, i.e. the intended deformation. This is similar to the PRBMs, the semi-analytic-deflection-method [12] and the GMP-technique. In contrast to PRBMs, flexure joints are not simply modeled as ideal hinges, but parasitic motion is taken into account, and the KSD-method does offer the opportunity to approximate the unintended deformation. In contrast to GMP, the KSD-method separates the large and small contribution to the deformation based on physics, i.e. in intended and unintended motion. Another difference is that GMP is used in dynamics and the KSD-method solves static equations. In contrast to the semi-analytic-deflection-method, the KSD-method can be applied to mechanisms with flexure joints that are arbitrarily complex.

The EOB computes the internal configurations of joints based on a priory simulations, similar to the data-driven reduction techniques. The main novelty of the EOB is the type of data that is used, i.e. the element orientations. Therefore the data is insensitive to changes of most of the dimensions.

Section 2 explains the KSD-method and Section 3 gives details about the EOB. The KSD-method with EOBs is validated in Section 4 by the analysis of a single cross flexure, a fourbar-mechanism with four cross flexure joints and a manipulator with three 3x-infinity joints. The paper ends with the most important conclusions.

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

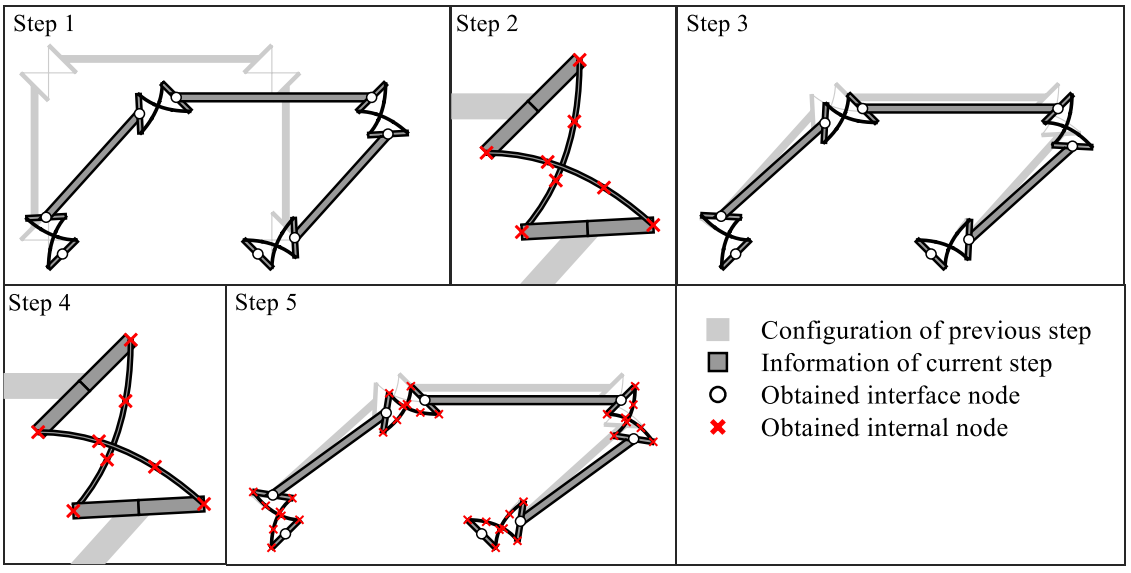

Fig. 2. Schematic overview of the five steps in the KSD-method for a fourbar mechanism. 1) Estimate the interface displacements by kinematics, based on the displacement of the end-effector in actuated directions 2) Estimate the internal displacements by kinematics, based on the information in the database. Based on this step, a stiffness matrix of the joint can be obtained. 3) Update the interface displacements based on static equilibrium 4) Update the internal displacements based on static equilibrium 5) Update all displacements based on static equilibrium.

2. Static deformation starting from a kinematic approximation

The KSD-method computes the static deformed configuration of a mechanism in five steps. Details about the steps are given in the following subsections. In the first two steps the configuration is estimated purely based on kinematic relations. In these steps, only the intended deformation is described and a database with data of the standard flexure joints is used. Based on these two steps, stiffness matrices of the joints can be obtained. In steps 3 and 4 these stiffness matrices are used to update the estimation of the configuration by static equilibrium. In these steps, the deformation in the unintended directions is described linearly. Step 5 computes the configuration based on static equilibrium, taking the configuration after step 4 as an initial guess. Because this initial guess is already quite good, only a few iteration steps are required in this fifth step. Two conditions should be satisfied in order to perform the KSD-method. In the first place, the system should be kinematically determinate, as explained in Section 2.2. Secondly, data of all the joints in the mechanism should be present in a database.

The steps are schematically visualized in Fig. 2 for a two-dimensional fourbar mechanism with four cross flexures. Each cross flexure is modeled by a finite element model with three flexible beam elements per leaf spring and four rigid elements to link the flexible elements. The configuration is described by the positions and rotations of the nodes. The nodes at which the joint is connected to other parts are referred to as *interface nodes* and the nodes inside the joints are *internal nodes*. The displacements related to these nodes are *interface displacements* and *internal displacements* respectively. The term displacement refers to the combination of rotations and translational displacements in this paper.

2.1. Database with information of standard flexure joints

This subsection explains what kind of data of a joint is stored in a database and how this information can be obtained. The joint is modeled by a finite element model. The position and orientation of one of the interface nodes of this model is fixed. The displacement of the other interface node is prescribed in the intended directions, on a finite number of values over the range of motion. The resulting configurations of the flexure joint for each prescribed value are obtained based on static equilibrium, where the forces and moments in the unintended directions are zero. These conditions result in a unique configuration of the flexure joint for each prescribed value of the intended direction.

Fig. 3 shows the finite element model of the cross flexure that is used as an example in this section. It is fixed at interface node A. A cross flexure is meant to allow rotation, so the rotation of interface node B is the intended direction. This intended direction can for example be prescribed on 10 intervals in the range between -30° and 30°, assuming the reaction forces in the x-direction and y-direction on interface node B to be zero.

The resulting configurations are used to derive two kinds of functions that are stored in the database for use in step 1 and 2 of the KSD-method. The functions for step 1 are kinematic relations between both interface nodes of the joint. These are formulated as constraints for the unintended deformations. In case of the cross flexure, these functions can be chosen to be the horizontal and vertical positions of interface node B with respect to interface node A as functions of the intended

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

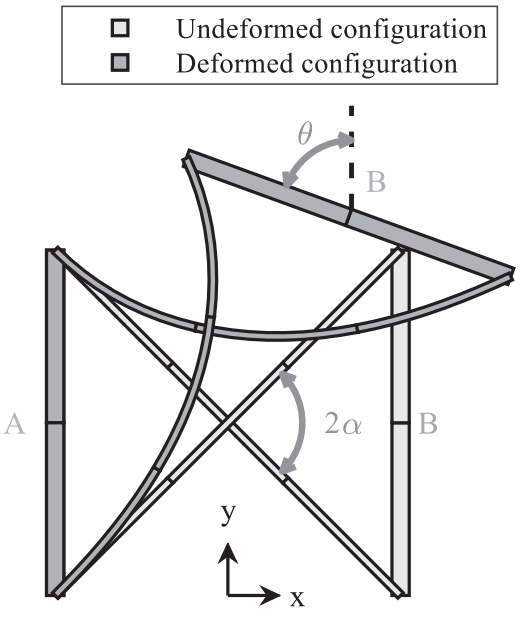

Fig. 3. Two-dimensional cross flexure, modeled by ten elements.

deformation θ, so the functions $x(\theta)$ and $y(\theta)$. These functions can be derived based on the simulation data, for example by a least square polynomial fit.

The second kind of functions that are stored in the database are the rotations of the elements as functions of the intended deformation. These functions are also obtained based on the same simulation data. Section 3.1 gives more details about the way to describe element orientations.

Note that static equilibrium is used to obtain the necessary data for the database, but that the resulting functions in the database are purely kinematic. One of the advantages of using kinematic relations is that they do not depend significantly on most of the dimensions of the joint, e.g. the kinematic relations of the cross flexure joint do not depend on the width and thickness of the flexures.

However, some design parameters may affect the functions in the database. For the cross flexure, the overall length of the flexure (i.e. the distance between both interface nodes) will affect the constraint functions $x(\theta)$ and $y(\theta)$. However, this can be easily avoided by storing dimensionless functions in the database: $\bar{x}(\theta) = x/L$ and $\bar{y}(\theta) = y/L$, where L is the length of the flexure. But also the angle α between both flexures (as indicated in Fig. 3) influences the results. This means that the functions in the database explicitly depend on α:

$$\bar{x}(\theta, \alpha),\ \bar{y}(\theta, \alpha) \tag{2.1}$$

So, although we wanted to obtain data that is insensitive to the dimensions of the joint, still some dimensions may have to be considered explicitly.

2.2. Step 1 – estimate the interface displacements by kinematics

In step 1 the positions and orientations of the interface nodes of the mechanism are approximated by a kinematic approach based on two conditions. In the first place the displacement of the end-effector in the actuated directions, q_{end}, is prescribed. The second condition is that the joints only allow motion in the intended direction. This is done by using the constraint equations for the standard flexures in the database. In this way the flexure joints do not have to be modeled as simple ideal pivot joints, but parasitic motion is also taken into account.

These two conditions are sufficient to obtain the configuration based on kinematic equations. However, this can only be solved uniquely if the mechanism is kinematically determinate (in other literature also referred to as a mechanism that is not underconstrained). This means that the KSD-method does not work for mechanisms that are kinematically indeterminate. On the other hand, if the mechanism is statically indeterminate (overconstrained) the number of unknown displacements is less than the number of equations. This means that some of the constraint equations are redundant. By removing these redundant equations the interface displacements can still be found. This system of nonlinear kinematic equations is solved by a Newton-Raphson iteration. Although this is an iterative process it can be computed very fast with respect to the computation of the solution of the full finite element model of the mechanism.

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

2.3. Step 2 – estimate the internal displacements by kinematics and obtain stiffness matrices of the joints

In step 2 the internal displacements are obtained from the interface displacements of step 1. These displacements will then be used to obtain the stiffness matrices of the joints. This can be done by using the Element Orientation based Body (EOB) description which is explained in Section 3. The result is a relation for each joint like:

$$\boldsymbol{F}_{nodes} = \boldsymbol{K}_{nodes}\boldsymbol{q}_{nodes} + \widehat{\boldsymbol{F}}_{nodes} \tag{2.2}$$

where $\boldsymbol{F}_{nodes}$ and $\boldsymbol{q}_{nodes}$ are the forces on the nodes and the displacements of the nodes of the joint, respectively. $\boldsymbol{K}_{nodes}$ is the stiffness matrix of the joint. The term $\widehat{\boldsymbol{F}}_{nodes}$ is a virtual force that appears because this stiffness relation is linearized around a deformed configuration and not in the undeformed configuration.

Note that the internal configuration of a joint can also be obtained without using the EOB. The most straightforward method to find the internal configuration is by solving the equilibrium equation of the finite element model of the joint, using the displacements of the interface nodes of step 1 as boundary conditions. Once the internal configuration is found, also the stiffness relation of Eq. (2.2) can be found. However, solving the equilibrium equation will require an iterative process that will take a lot of computation time with respect to the solution of the EOB. Therefore in this paper EOBs are used to obtain the stiffness matrices of the joints.

2.4. Step 3 – update the interface displacements based on static equilibrium

Step 3 updates the interface displacements by using the stiffness matrices from step 2. From this step on the unintended deformation is not constrained anymore and force-displacement relations are used, in contrast to step 1 and 2 where kinematic relations were used. This means that also load on constrained directions will have influence on the result of this step. Also the compliance of the stiff connecting links (e.g. the three links in the fourbar mechanism) can be modeled in this step.

In step 3A the stiffness matrices of the joints that specifies the stiffness between the interface nodes are obtained by using the boundary modes of the Craig-Bampton method [28]. The extra force term $\widehat{\boldsymbol{F}}$, has to be taken into account during this reduction. Eq. (2.2) is split in interface coordinates of the joint (normally called boundary coordinates in the Craig-Bampton method, indicated by b) and internal coordinates of the joint (indicated by i):

$$\begin{Bmatrix} \boldsymbol{F}_i \\ \boldsymbol{F}_b \end{Bmatrix} = \begin{bmatrix} \boldsymbol{K}_{ii} & \boldsymbol{K}_{ib} \\ \boldsymbol{K}_{bi} & \boldsymbol{K}_{bb} \end{bmatrix} \begin{Bmatrix} \boldsymbol{q}_i \\ \boldsymbol{q}_b \end{Bmatrix} + \begin{Bmatrix} \widehat{\boldsymbol{F}}_i \\ \widehat{\boldsymbol{F}}_b \end{Bmatrix}. \tag{2.3}$$

By assuming no applied forces on the internal nodes, so $\boldsymbol{F}_i = \boldsymbol{0}$, this equation can be rewritten to:

$$\boldsymbol{F}_b = \boldsymbol{K}_T\boldsymbol{q}_b + \widehat{\boldsymbol{F}}_T, \tag{2.4}$$

where:

$$\boldsymbol{K}_T \equiv \boldsymbol{K}_{bb} - \boldsymbol{K}_{bi}\boldsymbol{K}_{ii}^{-1}\boldsymbol{K}_{ib},\ \widehat{\boldsymbol{F}}_T \equiv \widehat{\boldsymbol{F}}_b - \boldsymbol{K}_{bi}\boldsymbol{K}_{ii}^{-1}\widehat{\boldsymbol{F}}_i. \tag{2.5}$$

$\boldsymbol{K}_T$ is the Craig-Bampton reduced stiffness matrix of the joint. $\widehat{\boldsymbol{F}}_T$ is a virtual force that is applied on the interface nodes of the body.

In step 3B the interface displacements are computed by using these Craig-Bampton reduced stiffness matrices. The interface displacements of step 1 are used as the starting point for this step. Because these displacements are close to the displacements that are obtained in this step, only a few iteration steps are required to solve this static problem.

2.5. Step 4 – estimate the internal displacements based on static equilibrium

In step 4, for each joint the internal displacements are updated by using the interface displacements of step 3 and the joint stiffness matrices of the joints of step 2. An equation for the internal displacements can be derived from Eq. (2.3):

$$\boldsymbol{q}_i = -\boldsymbol{K}_{ii}^{-1}\left(\boldsymbol{K}_{ib}\boldsymbol{q}_b + \widehat{\boldsymbol{F}}_i\right). \tag{2.6}$$

2.6. Step 5 – update all displacements based on static equilibrium

In step 5, the interface and internal displacements are updated simultaneously by solving for static equilibrium by an iterative approach. The interface displacements of step 3 and the internal displacements of step 4 are used as a starting point. Because the starting configuration is close to the deformed configuration, only a few iteration steps are required. The system of equations that is solved in this step is exactly the same as the system of equations that is solved in the conventional approach, i.e. obtaining the deformed configuration by an iterative process starting from the undeformed configuration. This means that step 5 will give the same result as the conventional method where it is generally faster as it will involve fewer iteration steps.

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

2.7. Reduced KSD-method and full KSD-method

In the kinematic iterative process in step 1 and in the static equilibrium iterative process in step 3 a mechanism is considered in which each joint is considered as a single body and where only the interface displacements were obtained. This mechanism will be referred to as the *reduced mechanism*. In contrast, in step 5 the *full mechanism* is considered. In the full mechanism each element in the finite element models of the joints is considered as a separate modeling part and the interface displacements and internal displacements are computed simultaneously. The result after step 4 is a good approximation of the displacements. Using this as final result will be referred to as *KSD-reduced*. Using all 5 steps will be referred to as *KSD-full*.

Step 3 may seem unnecessary for KSD-full, as step 3 updates the displacements in unintended directions which are generally small. However, using step 3 to update the displacements of the interface nodes gives the advantage of performing some more iteration steps with the reduced mechanism instead of the full mechanism, which is much faster. Moreover, the inaccuracy of the obtained interface displacements in step 1 can be significant, such that a static equilibrium iteration on the full mechanism in step 5 can become instable.

3. Element orientation based approach to obtain the internal configuration in standard joints

This section shows the approach to obtain the internal configuration and the stiffness relations of an EOB, which is used to apply step 2 of the KSD-method. The position and orientation of the interface nodes should be known on beforehand. So the EOB is valuable in combination with the KSD-method, as the interface displacements of the joints are computed in step 1 of the KSD-method. Based on the interface positions and rotations and a database, the element orientations are obtained (Section 3.1). Using these orientations a configuration is derived that is close to the deformed configuration, referred to as the *near configuration* (Section 3.2). In this near configuration the stiffness matrix of the body is derived (Section 3.3).

3.1. Obtain the element orientations

The orientations of the elements are described by functions of the intended deformation that are stored in the database. These functions may depend on some of the design parameters. In case of the cross flexure, the angle α (see Fig. 3) appears to have a significant influence on the rotation of the elements, as noted in Section 2.1. The intended deformation is already known after step 1 of the KSD-method, the element orientations can therefore be obtained by simply evaluating the functions that are stored in the database:

$$\hat{\beta}_k = f_k(\mathit{Intended\ deformation},\ \mathit{Design\ Param}), \tag{3.1}$$

where $\hat{\beta}_k$ are the parameters that are used to describe the rotation of element k. The meaning of the hat on β is explained below. If the intended deformation is only in one plane, for example the deformation of a 2D or 3D cross flexure, the rotation of each element can be described by the rotation about one axis. If the elements rotate about multiple axis other techniques are required to describe the orientation. In that case $\hat{\beta}_k$ can be for example a vector with Euler parameters or Euler angles.

Instead of using a database with functions to obtain the element orientations, we could also define functions that define other properties of the joint by which the internal configuration and stiffness matrix of the joint could be obtained. One other option is to store some relevant terms of the stiffness matrix. However, these terms will depend on the material properties of the joint and are very sensitive to the thickness and width of all the individual flexures. That means that the database should provide different functions for all the relevant designs of the flexure joint. The configuration of the joint is less sensitive to these design-parameter variations, therefore storing kinematic functions is preferred.

Another option is to store functions for the displacements (rotational and translational) of all the internal nodes in the body. This would immediately describe the internal configuration and provide enough information to obtain the stiffness matrix. However, the translational displacements are more sensitive to variations in the dimensions of the joint than the element rotations. All in all, the element orientations are probably the most dimension independent parameters that provide enough information to obtain the stiffness matrices of the elements and the internal configuration of the body as the remaining part of this section will show.

3.2. Obtain the near configuration

Fig. 4(a) shows the near configuration. The displacements of the nodes in the near configuration are defined with respect to one of the interface nodes, in this case with respect to node A. The orientations of the elements in the near configuration are equal to the orientations that are described by $\hat{\beta}$. The orientations of two elements that are connected to each other are not necessarily the same and therefore the orientation of the intervening node cannot be uniquely defined. Therefore each element is described by its own nodes in the near configuration, these nodes will be referred to as *markers*. So the near configuration of the joint in Fig. 4(a) is described by 24 markers as it consist of 12 elements that all have two markers.

To obtain the translational displacements of the markers, the condition is set that the elements are undeformed in the near configuration. The translational displacement of the markers associated with node *A* equal zero by definition. The other

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

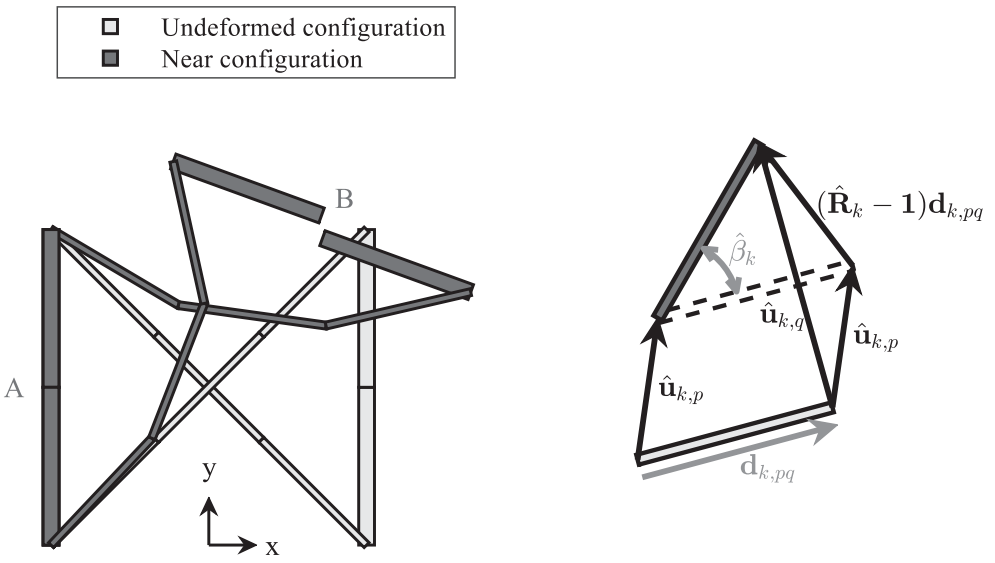

Fig. 4. Near configuration. (a) Near configuration of the complete body (b) Displacement of one element.

translational displacements are obtained by considering the joint to consist of multiple chains of connected elements that start at node *A*. Once the translational displacement of one of the markers of an element is found based on this condition, the translational displacements of the other markers can be found by the rotation of the element with respect to the undeformed configuration, see also Fig. 4(b):

$$\hat{\boldsymbol{u}}_{k,q} = \hat{\boldsymbol{u}}_{k,p} + \left(\hat{\boldsymbol{R}}_k - \mathbf{1}\right)\boldsymbol{d}_{k,pq}, \tag{3.2}$$

where $\hat{\boldsymbol{u}}_{k,q}$ and $\hat{\boldsymbol{u}}_{k,p}$ are the translational displacement of marker q and p respectively on element k. The hat indicates that the related variable describes a displacement from the undeformed configuration to the near configuration, these displacements will be referred to as *rigid displacements*. The vector $\boldsymbol{d}_{k,\,pq}$ is the distance from marker p to marker q in the undeformed configuration. The matrix $\hat{\boldsymbol{R}}_k$ is the rotation matrix that defines the rotational part of the rigid displacement is only a function of $\hat{\beta}_k$. Note that in case of one or more closed loops of elements, there will be points at which the markers of two coupled elements are not exactly at the same location because they are related to node *A* by a different chain of elements. This is also the case in the cross flexure, therefore the locations of both markers at node *B* are not exactly the same. For clarity this effect is exaggerated in Fig. 4(a).

3.3. Obtain stiffness matrix in the near configuration

This section describes how the stiffness matrix can be obtained in the near configuration by which a better approximation of the deformed configuration can be obtained. Because the orientation of each element in the near configuration should be close to the orientation of the deformed configuration, the displacements between both configurations can be described linear. Note that there can be deviation in the locations of both configurations, but large translations do not make the force-displacement relation nonlinear. The force-displacement relation for an element k is

$$\boldsymbol{F}_k = \boldsymbol{K}_k \bar{\boldsymbol{q}}_k, \tag{3.3}$$

where $\boldsymbol{F}_k$ is the vector with the forces and force moments on the markers of element k expressed in absolute coordinates. So this vector consist of six numbers for each marker of a three dimensional element k. $\boldsymbol{K}_k$ is the global stiffness matrix of element k which can be obtained by rotating the local stiffness matrix, using the rotation matrix $\hat{\boldsymbol{R}}_k$ of the element. The vector $\bar{\boldsymbol{q}}_k$ contains the translational and rotational displacements on the markers of element k. The bar indicates that these are displacements from the near configuration to the deformed configuration, which will be referred to as the *flexible displacements*. Note that there is no force required for the rigid displacements, this means that the force in Eq. (3.3) is the total force that is required to have element k in its deformed configuration. The relations for all elements can be combined in one equation:

$$\boldsymbol{F}_{all} = \boldsymbol{K}_{all} \bar{\boldsymbol{q}}_{all}, \tag{3.4}$$

where:

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

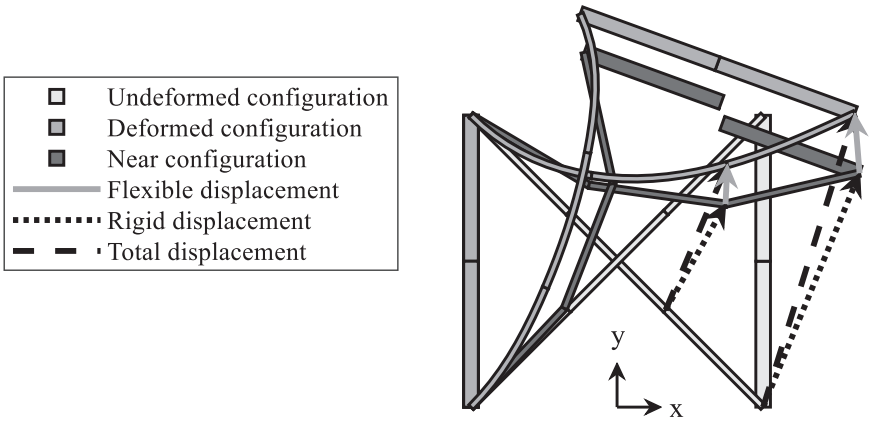

Fig. 5. The total displacement is the sum of the rigid displacement and the flexible displacement, indicated for one element.

$$\boldsymbol{F}_{all} \equiv \begin{Bmatrix} \boldsymbol{F}_1 \\ \vdots \\ \boldsymbol{F}_N \end{Bmatrix}, \ \boldsymbol{K}_{all} \equiv \begin{bmatrix} \boldsymbol{K}_1 & & \\ & \ddots & \\ & & \boldsymbol{K}_N \end{bmatrix}, \ \bar{\boldsymbol{q}}_{all} \equiv \begin{Bmatrix} \bar{\boldsymbol{q}}_1 \\ \vdots \\ \bar{\boldsymbol{q}}_N \end{Bmatrix}, \tag{3.5}$$

where N is the number of elements by which the joint is modeled.

The total displacement is the sum of the rigid and the flexible displacement. This is shown for the two markers of one element in Fig. 5. So, the flexible displacement is the total displacement minus the rigid displacement:

$$\bar{\boldsymbol{q}}_{all} = \boldsymbol{q}_{all} - \hat{\boldsymbol{q}}_{all}. \tag{3.6}$$

Note that, in case of large rotations about multiple axis, the rotational parts in these displacement vectors cannot be added directly due to the non-vectorial nature of rotations. Therefore the rotations in $\boldsymbol{q}_{all}$ and $\hat{\boldsymbol{q}}_{all}$ should be defined in such way that their difference equals the finite rotations in $\bar{\boldsymbol{q}}_{all}$. In this paper only large planar deformation is considered, such that the rotation in $\boldsymbol{q}_{all}$ and $\hat{\boldsymbol{q}}_{all}$ can be described by the rotation about the axis perpendicular to this deformation plane. Substituting Eq. (3.6) in Eq. (3.4) gives:

$$\boldsymbol{F}_{all} = \boldsymbol{K}_{all}(\boldsymbol{q}_{all} - \hat{\boldsymbol{q}}_{all}) = \boldsymbol{K}_{all}\boldsymbol{q}_{all} + \hat{\boldsymbol{F}}_{all}, \tag{3.7}$$

in which the term:

$$\hat{\boldsymbol{F}}_{all} = -\boldsymbol{K}_{all}\hat{\boldsymbol{q}}_{all}, \tag{3.8}$$

is constant. This term compensates for the fact that the rigid motions in the first term on the right hand side of the equation are computed linear. This term can be computed directly as the rigid displacements $\hat{\boldsymbol{q}}_{all}$ are the displacements from the undeformed to the near configuration that was obtained in Section 3.2.

In order to obtain the deformed configuration, the constraints between the elements should be applied. These constraints are applied by defining a Boolean matrix $\boldsymbol{L}$ such that:

$$\boldsymbol{q}_{all} = \boldsymbol{L}\boldsymbol{q}_{nodes}, \tag{3.9}$$

where $\boldsymbol{q}_{nodes}$ is the matrix with the total displacements of the nodes in the joint. The forces on the nodes can then be related like:

$$\boldsymbol{F}_{nodes} = \boldsymbol{L}^T\boldsymbol{F}_{all}, \ \hat{\boldsymbol{F}}_{nodes} = \boldsymbol{L}^T\hat{\boldsymbol{F}}_{all}. \tag{3.10}$$

This means that the relation between the forces and the displacements on the nodes can be written like:

$$\boldsymbol{F}_{nodes} = \boldsymbol{K}_{nodes}\boldsymbol{q}_{nodes} + \hat{\boldsymbol{F}}_{nodes}, \ \boldsymbol{K}_{nodes} \equiv \boldsymbol{L}^T\boldsymbol{K}_{all}\boldsymbol{L}. \tag{3.11}$$

This is the static equilibrium equation, which is valid around the near configuration. The equation is identical to eq. (2.2) that should be obtained in step 2 of the KSD-method.

4. Numerical validation

The EOB is validated by an analysis of a 3D cross flexure (Section 4.2) and the KSD-method with EOBs is validated by a fourbar mechanism with four cross flexure joints (Section 4.3). Section 4.1 describes the cross flexure model that is used in these sections. The KSD-method is further analyzed by a spatial manipulator with more complex joints in Section 4.4.

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

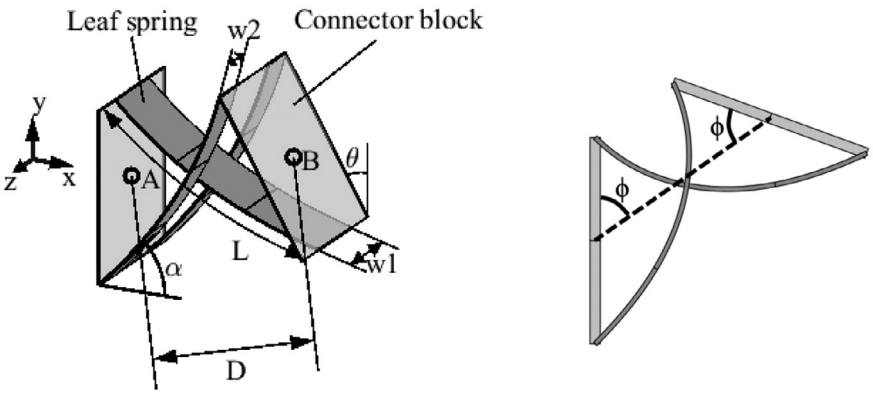

Fig. 6. Finite element model of the cross flexure with 3 beam elements per leaf spring. The right figure explains the fourth constraint on unintended deformation.

Table 1
Properties of the cross flexure.

Leaf spring length L	0.1 m
Angle α	45°
Thickness leaf springs t	$1 \cdot 10^{-3}$ m
Width inner leaf spring w_1	0.04 m
Width outer leaf springs w_2	0.02 m
Elasticity E	$200 \cdot 10^9$ Pa
Shear modulus G	$76.9 \cdot 10^9$ Pa

The used stiffness matrices of the flexible 3D beam elements are derived in Appendix A. The performance of the KSD-method is compared to that of a conventional method that is described in Appendix B. Both, the KSD-method and the conventional method are implemented in MATLAB and both methods solve the same finite element models of the mechanisms.

4.1. Cross flexure model

The finite element model of the cross flexure that was used in the simulations is shown in Fig. 6. The relevant properties of the flexure are given in Table 1. Both connector blocks are modeled by rigid elements and the interface nodes are exactly in the center of these connector blocks. The leaf springs were modeled with 1 to 5 serial connected equally sized elements.

The intended rotation is described by the angle between node A and B around the z-axis. It is denoted by θ and referred to as the (intended) rotation. For each of the models with 1 to 5 elements per leaf spring, simulations were run where the intended rotation was varied from 0 to 75° in steps of 5° and the displacements of all nodes were obtained. These displacements are used as simulation data to obtain the two required types of functions.

In order to perform step 1 of the KSD-method, five constraints for the unintended deformation are required. Four of these constraints can be defined without using data of a priory simulations (the chosen constraints are different from the constraints for the 2-dimensional cross flexure, introduced in Section 2.1):

- The displacement in the z-direction of node B with respect to node A should be zero.
- The rotation around the x-axis of node B with resprect to node A should be zero
- The rotation around the y-axis of node B with resprect to node A should be zero
- The rotation around the z-axis of node A and B with respect to the line through both nodes should be equal in size and opposite, as shown in Fig. 6.

The fifth constraint is the length of the cross flexure (the distance between both interface nodes) as a function of the intended deformation. This length was parameterized as a function of the intended rotation by a sixth order polynomial least squares fit to the simulation data. The odd terms in this polynomial are zero. The obtained relation is found to be almost independent of the cross flexure dimensions, except for the angles at which the leaf springs are positioned with respect to the local x-axis, which is denoted by α in Fig. 6. In this paper, only results are obtained with an angle $\alpha = 45°$. In summary, the constants g_k in the following relation are obtained:

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

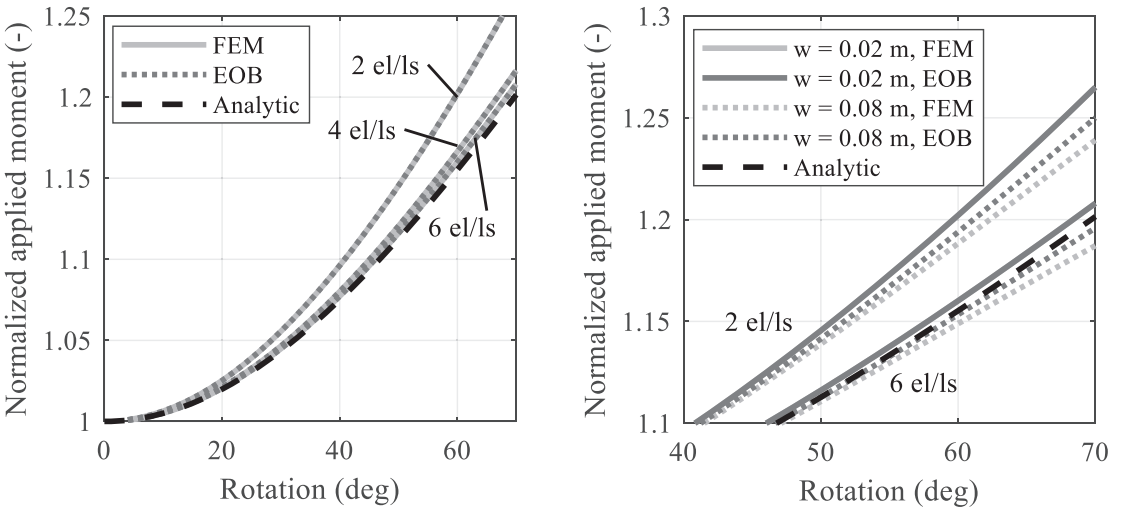

Fig. 7. Normalized applied moment on the cross flexure as a function of the intended rotation (a) For the default model with 2, 4 and 6 elements per leaf spring (b) for two different widths of the outer leaf springs with 2 elements per leaf spring (upper lines) and with 6 elements per leaf spring (lower lines).

$$\frac{D}{D_0} \approx 1 + g_1 \cdot \theta^2 + g_2 \cdot \theta^4 + g_3 \cdot \theta^6, \tag{4.1}$$

where D is the distance between both interface nodes, and D_0 is this distance in undeformed configuration.

The rotations of the elements around the z-axis were fitted by third order polynomials as a function of θ. So for the rotation of each element k a relation is obtained like:

$$\hat{\beta}_k \approx c_{k1} \cdot \theta + c_{k2} \cdot \theta^2 + c_{k3} \cdot \theta^3. \tag{4.2}$$

The global stiffness matrices of the beam elements as described in Appendix A depend on the orientations of the nodes, therefore also the orientation of the nodes are parameterized by a third order polynomial, similar to Eq. (4.2).

4.2. Results for single cross flexure

The accuracy of the results of the linearized equilibrium in the EOB, Eq. (3.11) was analyzed, which is representative for the accuracy of KSD-reduced. Fig. 7 shows the force moment that has to be applied on the cross flexure as a function of the intended rotation. The force moment is normalized by dividing it by the linear approximation [23], given by:

$$\bar{M}_{norm} = \frac{M}{M_{lin}}, \; M_{lin}(\theta) = \frac{2EI}{L}\theta, \tag{4.3}$$

where M is the applied moment on node B around the z-axis and I is sum of the second moment of area for all the three leaf springs:

$$I = \frac{1}{12}(w_1 + 2w_2)t^3. \tag{4.4}$$

An analytic reference for the single cross flexure, which is based on the assumptions of infinite axial stiffness of the flexures and no shear, was adapted from [23]. Fig. 7(a) shows that in the default case, the results of the EOB are exactly similar to the results of the finite element model of the joint: The line of the FEM result in this figure is exactly behind the line of the result of the EOB. Also the stiffness in the unintended directions of the EOB was found to match the result of the finite element method perfectly.

The applicability of the EOB is found to be bounded by two kinds of limitations. The first kind of limitation is that the difference in stiffness between the inner flexure and the two outer flexures may not vary too much. Fig. 7(b) shows the effect of wider outer leaf springs. In the default case, $w_2 = 0.02$ m, the line of the reference case is hidden behind the line of the EOB. If the width of the outer flexures is four times as large, the standardized moment is a little lower for both cases, but this effect for the EOB is smaller than for the reference case. So there is some variation between the EOB and the reference, which is caused by the fact that the parameterizations of the rotations $\hat{\beta}$ are not accurate for a large variation in the ratio between these stiffness of the inner and the outer flexures. The result indicates that it should be carefully considered for which ranges in the dimensions the parametrization holds.

The second kind of limitation is on the amount of unintended deformation. The results become inaccurate if this deformation becomes so large that the assumption of linear unintended deformation is not valid anymore. Fig. 8 shows the effect of a disturbance force in the z-direction on node B. The resulting displacement mainly depends on the stiffness in the

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

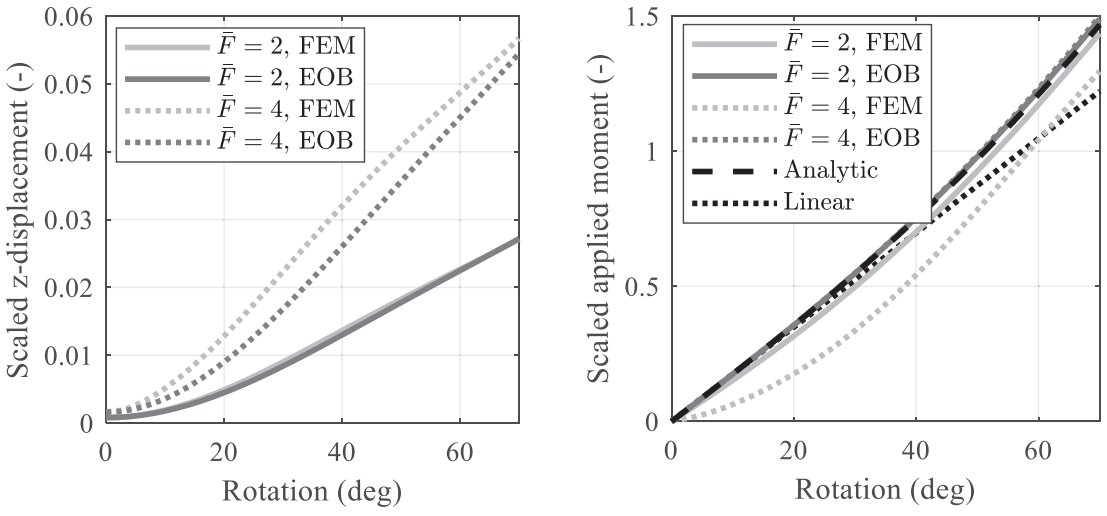

Fig. 8. Effect of large force in z-direction on node B of the cross flexure, modeled with four elements per leaf spring (a) Displacement of node B in the z-direction (b) Effect on the applied moment to rotate the cross flexure.

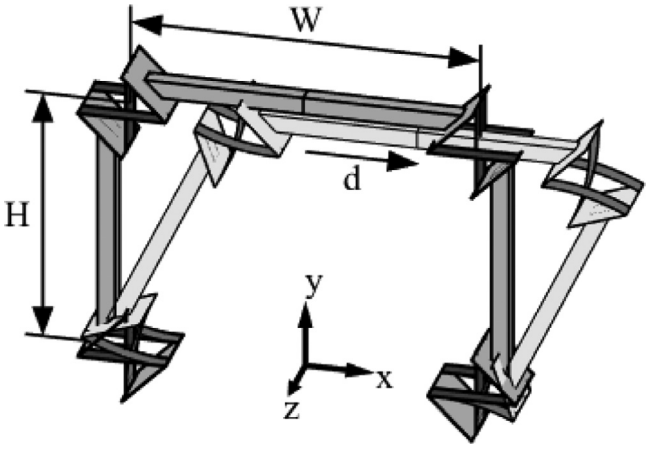

Fig. 9. Fourbar mechanism in undeformed and deformed configuration. Flexible parts are dark and rigid parts are light.

intended direction for large intended deformation. Therefore the applied force is scaled by dividing it by the stiffness in this direction:

$$\bar{F} = \frac{L^2}{2EI} \cdot F_z, \tag{4.5}$$

where F_z is the applied force on node B in the z-direction. The z-displacement is normalized by dividing it by the length of the leaf springs, L. The force moment that is required for the intended deformation is scaled by dividing it by the linear stiffness:

$$\bar{M}_{scale} = \frac{L}{2EI} \cdot M. \tag{4.6}$$

Note that this scaled force moment is different from the normalized force moment that was introduced in Eq. (4.3). The results in Fig. 8 indicate that the deformation in the z-direction is approximated quite accurately with the EOB, but that it shows almost no effect of the disturbance force on the applied force moment, where this effect can be quite significant.

4.3. Fourbar-mechanism

Fig. 9 shows the fourbar mechanism. The cross flexures are modeled with the properties given in Table 1, and the other relevant dimensions are indicated in the figure. The width W is 0.4 m and the height H is 0.3 m. The links between the cross

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

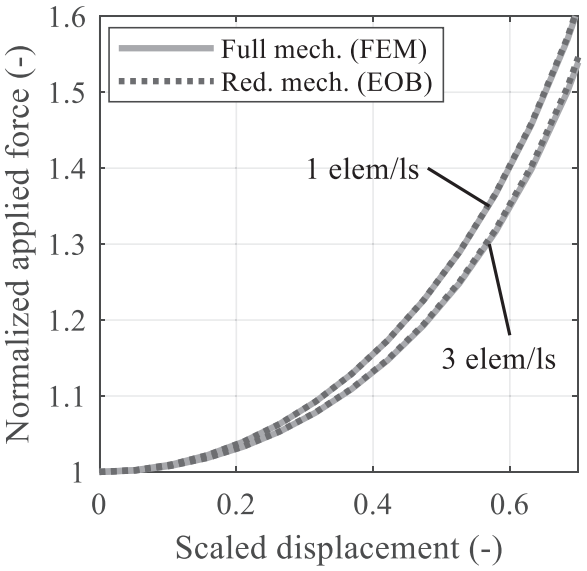

Fig. 10. Normalized applied force to displace the end effector of the fourbar mechanism with 1 and 3 elements per leaf spring.

flexures are assumed to be infinitely stiff. The desired motion of the fourbar is described by the displacement of the center of the upper link in the *x*-direction. The mechanism is actuated by a force on this point in the *x*-direction.

Fig. 10 shows the normalized applied force as a function of the normalized displacement for the full mechanism and for the reduced mechanism. The displacement, d, is scaled by dividing it by the height H of the mechanism, the force is normalized by dividing it by the linear approximation, given by:

$$\bar{F}_{norm} = \frac{F}{F_{lin}},\ F_{lin}(d) = \frac{4EI}{LH^2} \cdot d, \tag{4.7}$$

where F is the force that is applied on the center of the upper link. The result shows that KSD-reduced gives accurate results.

The computational efficiency of the KSD-method is examined by computing the deformed configuration of the fourbar mechanism where the endpoint is displaced by $d = 0.3$ m. The computation stops if the norm of the vector with resulting forces and force moments on the nodes is smaller than $5 \cdot 10^{-7}$, in which the forces are expressed in N and the moments in Nm.

Fig. 11(a) shows the number of iteration steps that are required for the conventional approach, KSD-reduced and KSD-full. In each iteration step the independent coordinates are updated one. Inside each iteration step, another iterative algorithm updates a set of dependent coordinates. For the full approach, the number of iterations is split in the number of iterations on the reduced mechanism for step 3 of the KSD-method (which is of course identical to the number of iterations for KSD-reduced) and the number of iterations on the full mechanism for step 5. The conventional approach requires more iterations than the KSD-method, especially in case of a large number of elements per leaf spring. This is because of the higher number of independent parameters that describe the configuration, increasing the nonlinearity of the system of equations.

Fig. 11(b) shows the average time per iteration. The solution time per iteration applied on the reduced mechanism is almost independently of the number of elements per leaf spring. In the case of the fourbar each iteration applied on the reduced mechanism was solved in approximately 30 ms. The time per iteration on the full mechanism is higher. The average time per iteration on the full mechanism in step 5 of the full approach is shorter than the time per iteration of the conventional approach. It is reduced by a factor of about 2 for the cases with 4 or 5 elements per leaf spring. The reason for this is that the displacements in these iterations are much smaller, and therefore it takes shorter to update the dependent coordinates in each iteration.

In order to study the effect of forces in the unintended directions, an external moment around the z-axis is applied on the end-effector, i.e. the center of the top-bar. Fig. 12(a) shows the resulting rotation of the upper link for the full model and for the reduced model. This error is result of the linearization of the unintended motion of the cross flexures. Even without external moment, there is a difference of about 20% in the rotation. This unintended motion is present as the cross flexures are not excited to a pure moment. Fig. 12(b) shows the required number of iterations and the total simulation time for the conventional method and step 5 of KSD-full. It indicates that for both methods the required number of iterations increases with an increasing force in the unintended directions. The KSD-method is about 5 times faster over the full range of the external moment in unintended direction. It was observed that in simulations with a force out of plane (e.g. a force in the z-direction on the center of the top link) the computation of the KSD-method does not converge in step 5.

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

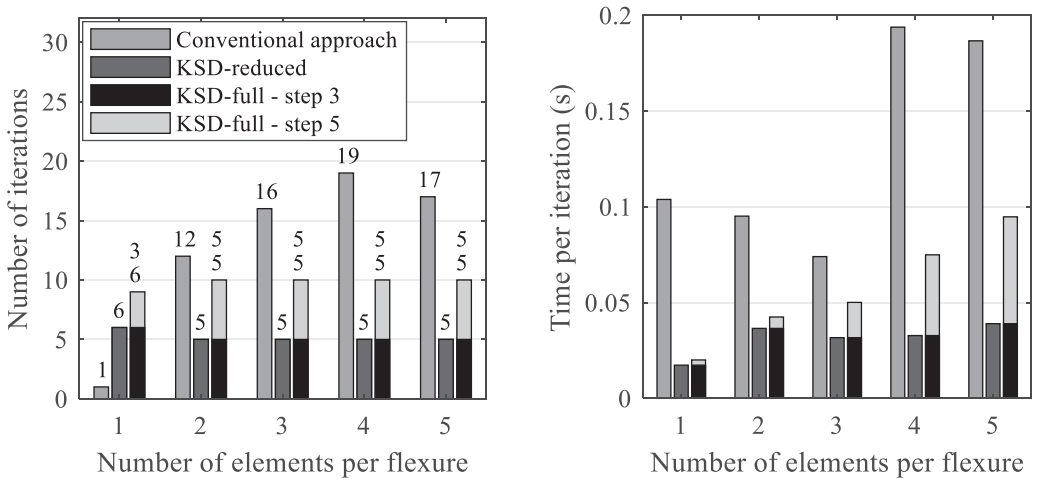

Fig. 11. Efficiency of the KSD-method and the conventional approach for the fourbar problem (a) Number of iterations that was required. (b) Average time per iteration, for the full approach, this is split in the time per iteration on the reduced mechanism (step 3) and the time per iteration on the full mechanism (step 5).

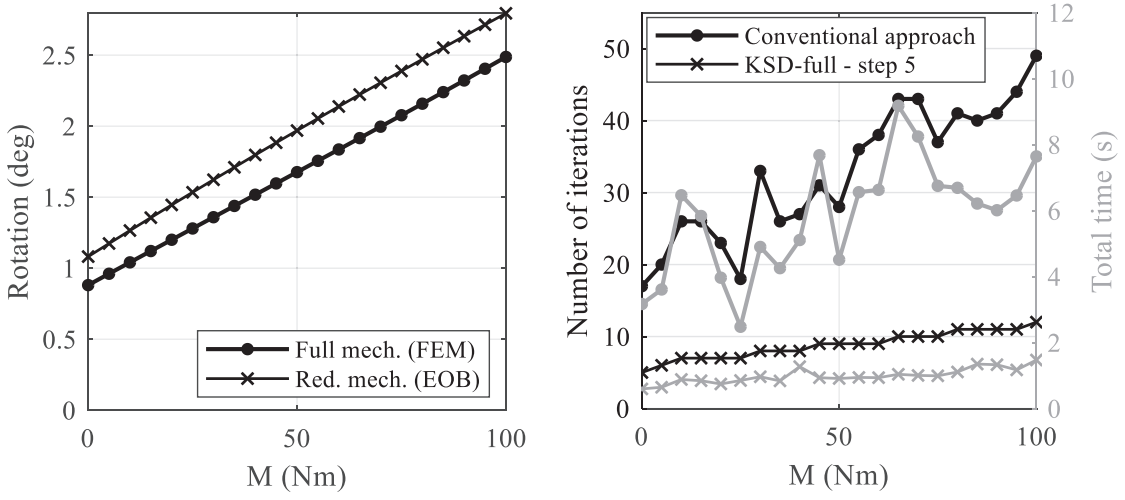

Fig. 12. Fourbar mechanism with an external applied force moment, computed for 21 different values of the applied force. (a) Resulting rotation of the upper link around the z-axis for the full mechanism and the reduced mechanism (b) Required number of iterations and simulation time to compute the configuration for the conventional approach and step 5 of KSD-full.

Overall KSD-reduced is about 20 times faster than the conventional method for 4 or 5 elements per leaf spring, respectively. KSD-full is about 5 times faster than the conventional method for 4 or 5 elements per leaf spring. The main reason for this increase in efficiency is the reduced number of required iterations.

4.4. Spatial three-link manipulator

The KSD-method is applied to the manipulator with three 3x-infinity joints and three links, as shown in Fig. 13. The first joint can rotate around the global z-axis and the other two rotate initially around the global y-axis. Fig. 14 shows the dimensions of the 3x-infinity joints. The material properties are similar to the material properties of the cross flexure: the elasticity is 200 GPa and the shear modulus 76.9 GPa. The joints are modeled by 30 flexible beam elements and 36 rigid beam elements. To obtain data of these joints for the database, training simulations were run to compute the resulting configurations of 19 different values of the intended rotation. These 19 values for the intended rotation are chosen in the range between -45 and $45°$, with a step size of $5°$.

Five constraints on the unintended deformation are required for step 1 of the KSD-method. Three of these constraints are identical to the first three constraints of the cross flexure, described in Section 4.1. The other two constraints are the displacements in the x and y-direction of interface node B with respect to node A as a function of the intended rotation.

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

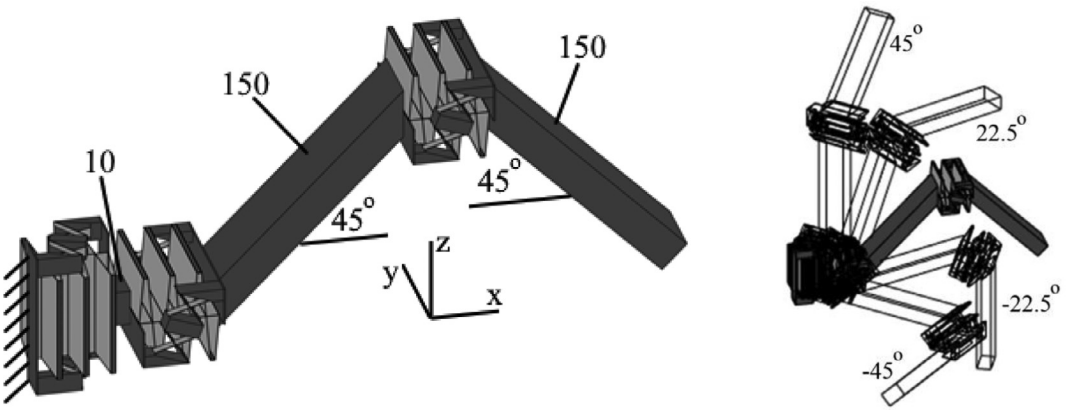

Fig. 13. Manipulator, fixed to the ground at the left side. Flexible parts are light and rigid parts are dark. (a) Undeformed configuration showing length and angles of the links (b) Undeformed configuration and some deformed configurations.

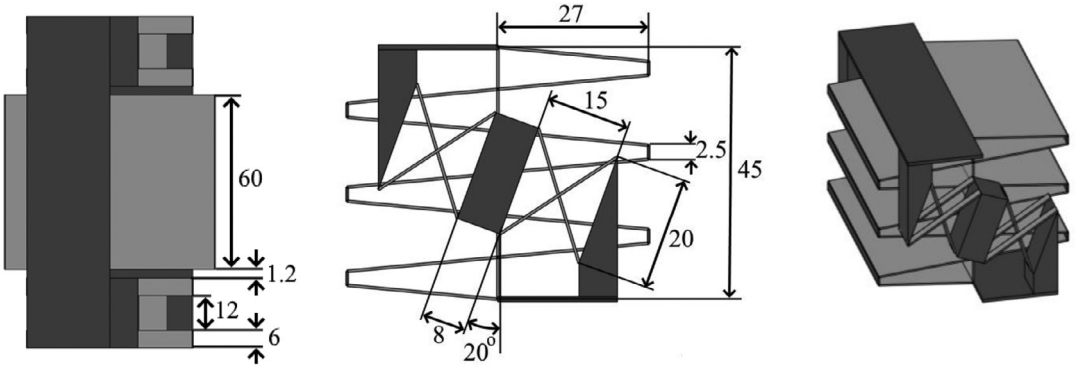

Fig. 14. Dimensions of the 3x-infinity joint in mm. The thickness of the flexible elements is 0.45 mm. Flexible parts are light and rigid parts are dark. (a) Top view (b) Side view (c) 3D-view. The figure of the top view is rescaled to visualize the small dimensions.

These constraints are expressed as a fourth order polynomial of the intended rotation which is a least squares approximation of the simulation data. The rotations of all the elements and nodes are also approximated by a fourth order polynomial in the intended rotation, in order to perform step 2 of the KSD-method.

The manipulator has three intended degrees of freedom, therefore three parameters are required to define the position of the end-effector. Two different parameter sets are analyzed to describe the position:

- Joint-case: the rotational motion of each of the three joints;
- Location-case: the location of the end-point of the third link in *x, y*, and *z*-direction.

Both cases are essentially different in terms of the reaction forces. The joint-case assumes three reaction moments around the joints, the location-case assumes three linear forces on the end-point of the third link. This means that the motion in the unintended directions is different. Actually there is no unintended motion in the joint-case as all joints are actuated by a pure force moment. Therefore the KSD-method is expected to perform better in the this case.

Simulations are run in which the rotation of all three joints is simultaneously varied in the range from minus 45 to 45°. Fig. 13(b) shows some of the resulting configurations. In a first run the joint-case was performed. In a second run, the location-case was performed, using the locations for the end-point of the third link obtained from the joint-case. Fig. 15 shows the required number of iterations and the total simulation time. It shows that the KSD-method indeed requires more iterations for the location-case. This holds surprisingly also for the conventional method. Fig. 15(d) shows that using KSD-full instead of the conventional method reduces the simulation time up to a factor of 65. KSD-reduced reduces the total simulation time by a factor up to 600. The joint-case has almost 100% accuracy (except from some numerical and interpolation errors). The error on the estimated reaction force of the location-case is less than 2% and the error in displacements less than 0.3%.

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

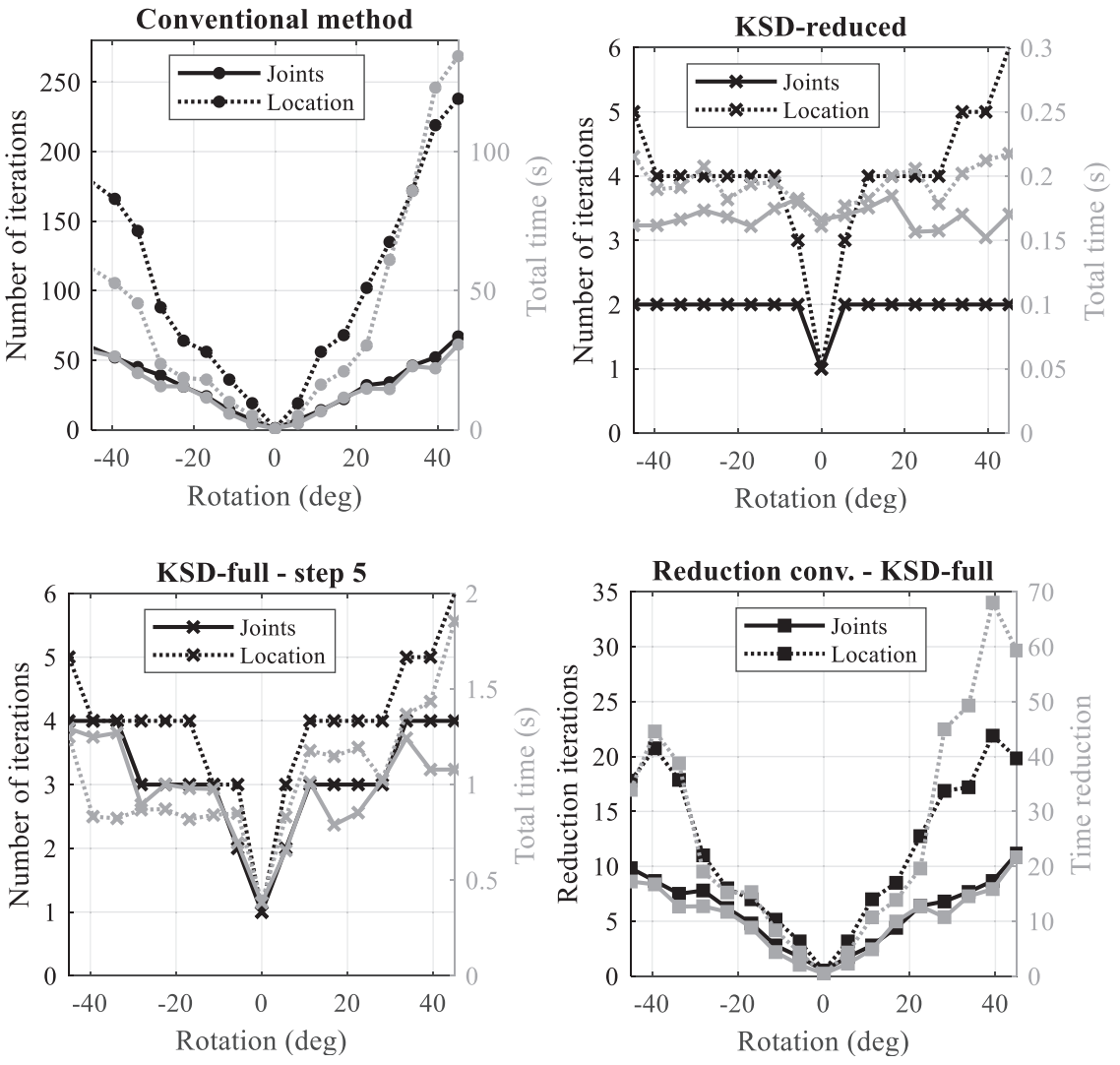

Fig. 15. Computational efficiency for manipulator with 3x-infinity joints for two different descriptions for the position of the end-effector, computed for 19 different positions of the end-effector. (a) Conventional method (b) KSD-reduced (this is the same as step 3 of KSD-full) (c) KSD-full – step 5 (d) Factor by which the number of iterations and simulation time are reduced by using KSD-full in comparison to the conventional method.

5. Conclusions

In this paper we presented a Kinematically Started Deformation method (KSD-method) to obtain static equilibrium of flexure mechanisms efficiently, by exploiting prior knowledge on the intended degrees of freedom. The relatively large deformation in the intended degrees of freedom is kinematically approximated as a first step in the iteration process.

The flexure joints in the mechanism were modeled with Element Orientation based Bodies (EOBs). The configuration of an EOB is approximated based on a parameterization of the rotations of the elements and this approximation is refined using static equilibrium. Using element orientations makes the parameterization suitable for joints with a large range of dimensions and therefore applicable in design optimizations.

The KSD-method with EOBs provides a way to use a database with information of standard joints. As a consequence, the application of the method is limited to mechanisms that consist of joints of which information exist in the database. This required information consists of relations between the interface points during intended deformation and the rotations of the elements as function of the intended deformation. Another limitation is that the mechanism should be kinematically determined in order to perform the first step of the KSD-method, but this is typical the case for flexure based mechanisms. A last limitation is that unintended deformation (deformation in supporting directions) should be low. This is not a large limitation as the purpose of a flexure based designs are related to high support stiffness. Although in this paper the method

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

was only applied to mechanisms that were modeled by beam-elements, the method is not fundamentally limited to this modeling approach.

The efficiency of the KSD-method is compared to that of a conventional method. For a fourbar mechanism the required computation time was decreased up to a factor of 5 to obtain the accurate deformed configuration. For a manipulator with three 3x-infinity joints the computation time reduced up to a factor of 65. The computed configuration of the KSD-method is exactly the same as the configuration found by the conventional method as the KSD-method solves in the end exactly the same equations. However, the KSD-method can also be used to obtain the approximated deformed configuration in which the deformation in the unintended direction of the flexure joint is assumed to be linear. The time reduction to find this approximation with respect to the conventional method is a factor up to 20 for the fourbar-mechanism and a factor up to 600 for the manipulator. The proposed method potentially saves orders of magnitude of valuable time during the optimization of flexure mechanisms in the conceptual phase.

Acknowledgement

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).

Appendix A. Stiffness beam elements

In this appendix the stiffness matrix of a beam element is derived that is used in this paper. The used beam formulation and its derivation are relatively similar to the formulation described in [29].

A.1. Element configuration

A.1.1. Nodal coordinates

Fig. 16 shows a beam element in its initial configuration and its deformed configuration. The configuration of the beam is defined by twelve independent nodal coordinates:

$$\boldsymbol{x} = \left\{ \begin{array}{cccc} \boldsymbol{r}^{pT} & \boldsymbol{\lambda}^{pT} & \boldsymbol{r}^{qT} & \boldsymbol{\lambda}^{qT} \end{array} \right\}^T, \tag{A.1}$$

where $\boldsymbol{r}^p$ and $\boldsymbol{r}^q$ define the locations of node p and q respectively with respect to the global fixed frame. The rotation matrices that define the orientations of the nodes, $\boldsymbol{R}^p$ and $\boldsymbol{R}^q$, are parameterized by $\boldsymbol{\lambda}^p$ and $\boldsymbol{\lambda}^q$ respectively. The initial orientation of the element is given by $[\ \boldsymbol{n}_x \quad \boldsymbol{n}_y \quad \boldsymbol{n}_z\ ]$. The unit vectors at the nodes of the elements, as visualized in Fig. 16 can therefore computed by:

$$\begin{aligned} \boldsymbol{n}_y^p &= \boldsymbol{R}^p\left(\boldsymbol{\lambda}^p\right)\boldsymbol{n}_y,\ \boldsymbol{n}_y^q = \boldsymbol{R}^q\left(\boldsymbol{\lambda}^q\right)\boldsymbol{n}_y, \\ \boldsymbol{n}_z^p &= \boldsymbol{R}^p\left(\boldsymbol{\lambda}^p\right)\boldsymbol{n}_z,\ \boldsymbol{n}_z^q = \boldsymbol{R}^q\left(\boldsymbol{\lambda}^q\right)\boldsymbol{n}_z. \end{aligned} \tag{A.2}$$

A.1.2. Deformation coordinates

The deformation of the beam is described by six mode shapes. These modes are specified by deformation coordinates, $\boldsymbol{\varepsilon}$, that are an explicit function of the nodal coordinates:

$$\boldsymbol{\varepsilon} = \mathcal{D}(\boldsymbol{x}). \tag{A.3}$$

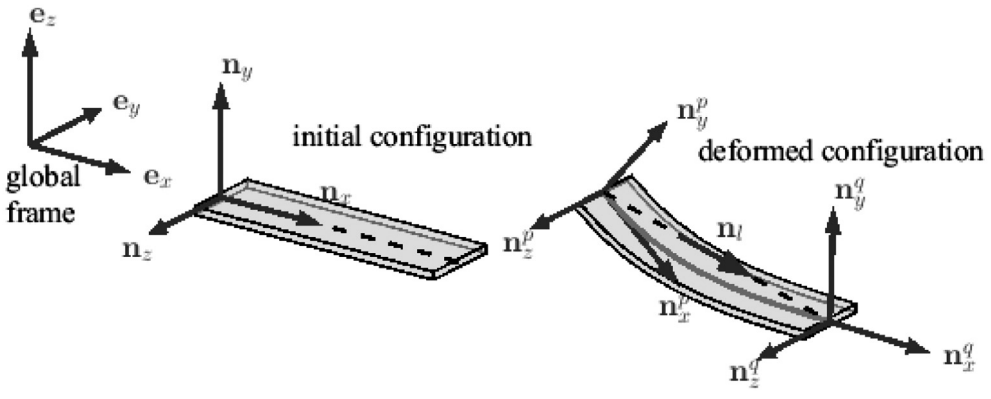

Fig. 16. Beam element in initial configuration and deformed configuration.

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

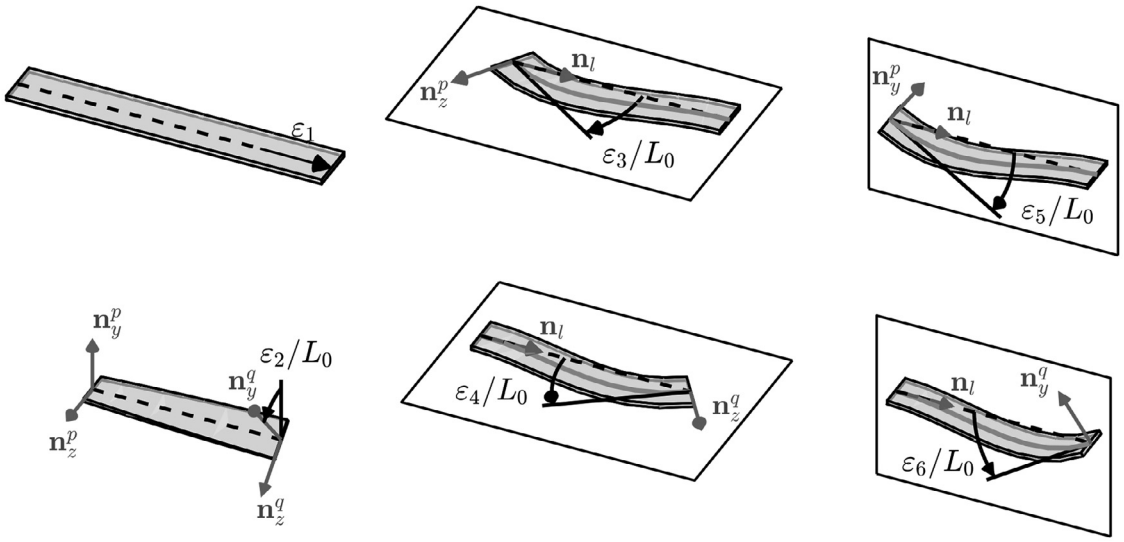

Fig. 17. Mode shapes beam element.

The chosen functions for these deformation coordinates are:

$$\begin{aligned}
&\varepsilon_1 = |\boldsymbol{L}| - L_0 && \text{(axial elongation)}\\
&\varepsilon_2 = \tfrac{1}{2}L_0\left(\text{asin}\left(\boldsymbol{n}_z^p \cdot \boldsymbol{n}_y^q\right) - \text{asin}\left(\boldsymbol{n}_y^p \cdot \boldsymbol{n}_z^q\right)\right) && \text{(torsion)}\\
&\left.\begin{aligned}
\varepsilon_3 &= -L_0\text{asin}\left(\boldsymbol{n}_z^p \cdot \boldsymbol{n}_l\right)\\
\varepsilon_4 &= \ L_0\text{asin}\left(\boldsymbol{n}_z^q \cdot \boldsymbol{n}_l\right)\\
\varepsilon_5 &= \ L_0\text{asin}\left(\boldsymbol{n}_y^p \cdot \boldsymbol{n}_l\right)\\
\varepsilon_6 &= -L_0\text{asin}\left(\boldsymbol{n}_y^q \cdot \boldsymbol{n}_l\right)
\end{aligned}\right\} && \text{(bending)}
\end{aligned} \tag{A.4}$$

where:

$$\boldsymbol{n}_l = \frac{\boldsymbol{L}}{|\boldsymbol{L}|},\ \boldsymbol{L} = \boldsymbol{r}^q - \boldsymbol{r}^p. \tag{A.5}$$

The resulting mode shapes are visualized in Fig. 17. The difference between these deformation functions and the deformation functions introduced in [29], are the arcsine-terms in the second till the sixth function. The arcsine-terms are included to make the deformation functions linear dependent on the angle between the unit vectors. These arcsine-terms were also introduced in the 3D beam formulation in [30] and correspond to most of the 2D corotational beam formulations like [31,32].

A.1.3. Relation between change of nodal and deformation coordinates

The change of displacements of the nodes is given by:

$$\delta\boldsymbol{q} = \left\{ \begin{array}{cccc} \delta\boldsymbol{r}_p^T & \delta\boldsymbol{\phi}^{pT} & \delta\boldsymbol{r}_q^T & \delta\boldsymbol{\phi}^{qT} \end{array} \right\}^T, \tag{A.6}$$

where $\delta\boldsymbol{\phi}^p$ and $\delta\boldsymbol{\phi}^q$ are the virtual rotations of node p and node q respectively. Note that the orientations are defined different in the nodal coordinates. However, there exists relations between these virtual rotations and the change of $\delta\boldsymbol{\lambda}^p$ and $\delta\boldsymbol{\lambda}^q$. These relations depend on the choice of parametrization of the rotations and is therefore not discussed in this appendix. The relation between the change of the deformation and the virtual displacements is given by a matrix $\boldsymbol{D}$:

$$\delta\boldsymbol{\varepsilon} = \boldsymbol{D}\delta\boldsymbol{q}, \tag{A.7}$$

In order to derive an expression for $\boldsymbol{D}$, based on the definition of the deformation coordinates in Eq. (A.4), we note that the change of a rotation matrix can be expressed by the virtual rotations like:

$$\delta\boldsymbol{R}^p = \delta\tilde{\boldsymbol{\phi}}^p\boldsymbol{R}^p,\ \delta\boldsymbol{R}^q = \delta\tilde{\boldsymbol{\phi}}^q\boldsymbol{R}^q,\ \text{where}\ \tilde{\boldsymbol{\phi}} = \begin{bmatrix} 0 & -\phi_z & \phi_y \\ \phi_z & 0 & -\phi_x \\ -\phi_y & \phi_x & 0 \end{bmatrix}, \tag{A.8}$$

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

Using these relations, the derivatives of the unit vectors in Eq. (A.2) with respect to the virtual rotations can be expressed like:

$$\begin{aligned} \frac{\partial \boldsymbol{n}_y^p}{\partial \boldsymbol{\phi}^p} &= -\widetilde{\boldsymbol{n}}_y^p, \quad \frac{\partial \boldsymbol{n}_y^q}{\partial \boldsymbol{\phi}^q} = -\widetilde{\boldsymbol{n}}_y^q, \\ \frac{\partial \boldsymbol{n}_z^p}{\partial \boldsymbol{\phi}^p} &= -\widetilde{\boldsymbol{n}}_z^p, \quad \frac{\partial \boldsymbol{n}_z^q}{\partial \boldsymbol{\phi}^q} = -\widetilde{\boldsymbol{n}}_z^q. \end{aligned} \tag{A.9}$$

The derivative of the unit vector $\boldsymbol{n}_l$ with respect to the locations of the nodes can be expressed like:

$$\frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{r}^p} = -\frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{L}}, \quad \frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{r}^q} = \frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{L}}, \quad \frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{L}} = \frac{1}{|\boldsymbol{L}|}\left(\mathbf{1} - \boldsymbol{n}_l\ \boldsymbol{n}_l^T\right), \tag{A.10}$$

The matrix $\boldsymbol{D}$ can be determined from the definitions in Eq. (A.4), using Eqs. (A.9) and (A.10):

$$\boldsymbol{D} = [\boldsymbol{D}_1\ \boldsymbol{D}_2], \tag{A.11a}$$

$$\boldsymbol{D}_1 = \begin{bmatrix} -\boldsymbol{n}_l^T & \mathbf{0}^T \\ \mathbf{0}^T & \frac{L_0}{2B_{2a}}\left[\widetilde{\boldsymbol{n}}_z^p \boldsymbol{n}_y^q\right]^T - \frac{L_0}{2B_{2b}}\left[\widetilde{\boldsymbol{n}}_y^p \boldsymbol{n}_z^q\right]^T \\ \frac{L_0}{B_3}\left[\frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{L}}\boldsymbol{n}_z^p\right]^T & -\frac{L_0}{B_3}\left[\widetilde{\boldsymbol{n}}_z^p \boldsymbol{n}_l\right]^T \\ -\left[\frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{L}}\boldsymbol{n}_z^q\right]^T & \mathbf{0}^T \\ -\frac{L_0}{B_5}\left[\frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{L}}\boldsymbol{n}_y^p\right]^T & \frac{L_0}{B_5}\left[\widetilde{\boldsymbol{n}}_y^p \boldsymbol{n}_l\right]^T \\ \frac{L_0}{B_6}\left[\frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{L}}\boldsymbol{n}_y^q\right]^T & \mathbf{0}^T \end{bmatrix}, \tag{A.11b}$$

$$D_2 = \begin{bmatrix} \boldsymbol{n}_l^T & \mathbf{0}^T \\ \mathbf{0}^T & \frac{L_0}{2B_{2a}}\left[\widetilde{\boldsymbol{n}}_y^q \boldsymbol{n}_z^p\right]^T - \frac{L_0}{2B_{2b}}\left[\widetilde{\boldsymbol{n}}_z^q \boldsymbol{n}_y^p\right]^T \\ -\frac{L_0}{B_3}\left[\frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{L}}\boldsymbol{n}_z^p\right]^T & \mathbf{0}^T \\ \frac{L_0}{B_4}\left[\frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{L}}\boldsymbol{n}_z^q\right]^T & \frac{L_0}{B_4}\left[\widetilde{\boldsymbol{n}}_z^q \boldsymbol{n}_l\right]^T \\ \frac{L_0}{B_5}\left[\frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{L}}\boldsymbol{n}_y^p\right]^T & \mathbf{0}^T \\ -\frac{L_0}{B_6}\left[\frac{\partial \boldsymbol{n}_l}{\partial \boldsymbol{L}}\boldsymbol{n}_y^q\right]^T & -\frac{L_0}{B_6}\left[\widetilde{\boldsymbol{n}}_y^q \boldsymbol{n}_l\right]^T \end{bmatrix}, \tag{A.11c}$$

where:

$$\begin{aligned} B_{2a} &= \sqrt{1 - \left(\boldsymbol{n}_z^p \cdot \boldsymbol{n}_y^q\right)^2}, & B_{2b} &= \sqrt{1 - \left(\boldsymbol{n}_y^p \cdot \boldsymbol{n}_z^q\right)^2}, \\ B_3 &= \sqrt{1 - \left(\boldsymbol{n}_z^p \cdot \boldsymbol{n}_l\right)^2}, & B_4 &= \sqrt{1 - \left(\boldsymbol{n}_z^q \cdot \boldsymbol{n}_l\right)^2}, \\ B_5 &= \sqrt{1 - \left(\boldsymbol{n}_y^p \cdot \boldsymbol{n}_l\right)^2}, & B_6 &= \sqrt{1 - \left(\boldsymbol{n}_y^q \cdot \boldsymbol{n}_l\right)^2}. \end{aligned} \tag{A.12}$$

A.2. Stiffness properties

A.2.1. Local stiffness matrix. The relation between the elastic load on the mode shapes and the deformation of the mode shapes is represented by a matrix $\boldsymbol{S}$:

$$\boldsymbol{\sigma} = \boldsymbol{S}\boldsymbol{\varepsilon}. \tag{A.13}$$

The local stiffness matrix $\boldsymbol{S}$ can be expressed like:

$$\boldsymbol{S} = \mathrm{diag}(S_1, S_2, \boldsymbol{S}_3, \boldsymbol{S}_4), \tag{A.14}$$

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

with:

$$
\begin{aligned}
S_1 &= \frac{EA}{L_0} \\
S_2 &= \frac{k_{\bar{x}} G I_p}{L_0^3} \\
\boldsymbol{S}_3 &= \frac{EI_{\bar{y}}}{(1+\Phi_{\bar{z}})L_0^3}\begin{bmatrix} 4+\Phi_{\bar{z}} & -2+\Phi_{\bar{z}} \\ -2+\Phi_{\bar{z}} & 4+\Phi_{\bar{z}} \end{bmatrix} \\
\boldsymbol{S}_4 &= \frac{EI_{\bar{z}}}{(1+\Phi_{\bar{y}})L_0^3}\begin{bmatrix} 4+\Phi_{\bar{y}} & -2+\Phi_{\bar{y}} \\ -2+\Phi_{\bar{y}} & 4+\Phi_{\bar{y}} \end{bmatrix},
\end{aligned}
\tag{A.15}
$$

Here, E is the modulus of elasticity and G is the shear modulus. A is the area of the cross section, I_p is the polar moment of area of the cross section and $I_{\bar{y}}$ and $I_{\bar{z}}$ are the second moments of area of the cross section with respect to the principal y and z-axis respectively. $k_{\bar{x}}$ is the torsion correction factor according to Saint-Venant's theory. The shear factors are given by

$$
\Phi_{\bar{y}} = \frac{12EI_{\bar{y}}}{k_{\bar{z}}GAL_0^2}, \quad \Phi_{\bar{z}} = \frac{12EI_{\bar{z}}}{k_{\bar{y}}GAL_0^2},
\tag{A.16}
$$

in which $k_{\bar{y}}$ and $k_{\bar{z}}$ are shear correction coefficients [33]. The derivation of this stiffness matrix is based on the exact solution of the equilibrium equations of the Timoshenko beam for the case that only load is applied on the nodes of the beam element. For further details the reader is referred to [29].

A.2.2. Global stiffness matrix

The forces at the nodes, $\boldsymbol{F}^p$ and $\boldsymbol{F}^q$ and the force moments $\boldsymbol{T}^p$ and $\boldsymbol{T}^q$ are combined in one vector with nodal forces:

$$
\boldsymbol{F} = \begin{Bmatrix} \boldsymbol{F}^{pT} & \boldsymbol{T}^{pT} & \boldsymbol{F}^{qT} & \boldsymbol{T}^{qT} \end{Bmatrix}^T,
\tag{A.17}
$$

These forces are in the same direction as the nodal displacements, $\boldsymbol{q}$, of the element. Therefore, the relation between the nodal forces, $\boldsymbol{F}$, and the elastic load, $\boldsymbol{\sigma}$, can be derived by the principle of virtual work using Eqs. (A.7):

$$
\delta\boldsymbol{q}^T\,\boldsymbol{F} = \delta\boldsymbol{\varepsilon}^T\boldsymbol{\sigma} = \left(\delta\boldsymbol{q}^T\boldsymbol{D}^T\right)\boldsymbol{\sigma}\ \forall\ \delta\boldsymbol{u} \ \Rightarrow\ \boldsymbol{F} = \boldsymbol{D}^T\boldsymbol{\sigma}.
\tag{A.18}
$$

Using this equation and Eqs. (A.7) and (A.13) the relation between the forces and displacements can be derived:

$$
\boldsymbol{F} = \boldsymbol{K}\boldsymbol{q},\ \boldsymbol{K} \equiv \boldsymbol{D}^T\boldsymbol{S}\boldsymbol{D},
\tag{A.19}
$$

where the displacements, $\boldsymbol{q}$, are assumed to be small. $\boldsymbol{K}$ is the global stiffness matrix of the beam element that was used in the simulations in this paper.

Appendix B. Conventional method

To analyze the performance, the KSD-method is compared to a conventional method which solves the full finite element model of the mechanisms. As this conventional method uses exactly the same model that was used in the KSD-method, the final results of both methods are exactly the same, but the required simulation time may be different.

The theoretical background of this conventional method is the same as the theoretical background that has been implemented in SPACAR [34], as this software has often been used to model flexure based mechanisms [1–5]. It is particularly suited to obtain the relevant mechanism properties that are required in a design optimization, like maximum stress and eigen frequencies. The configuration is described by a set of independent coordinates and a set of dependent coordinates that are kinematically related to the independent coordinates. This theoretical background of SPACAR is implemented MATLAB, similar to the KSD-method, in order to have a fair comparison between both methods.

The number of iterations in the conventional methods is automatically determined as shown in Fig. 18. The total displacement of the end-effector, d_{tot} is first tried to be solved in one step. One iteration is performed, which means that the independent coordinates are updated based on the linearized equilibrium equations and the dependent coordinates are updated based on the underlying kinematic relations. After an iteration an error, ε, is computed based on the resulting forces on the nodes. The next step is decided based on this error, which has three options:

- The error is larger than a threshold, ε_{max}: in this case the iterative process will probably not converge and the step size in terms of displacement of the end-effector, d_{step}, is reduced by a factor of two.
- The error is smaller than ε_{max}, but above a certain accuracy-threshold, ε_{acc}: the equilibrium equations are linearized again and a new iteration is performed to increase the accuracy.
- The error is smaller than ε_{min}: the computation for the current step, d_{step}, is completed, therefore either the full process is completed, i.e. the total displacement d_{tot} has been reached, or a new step on the displacement of the end-effector is applied.

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

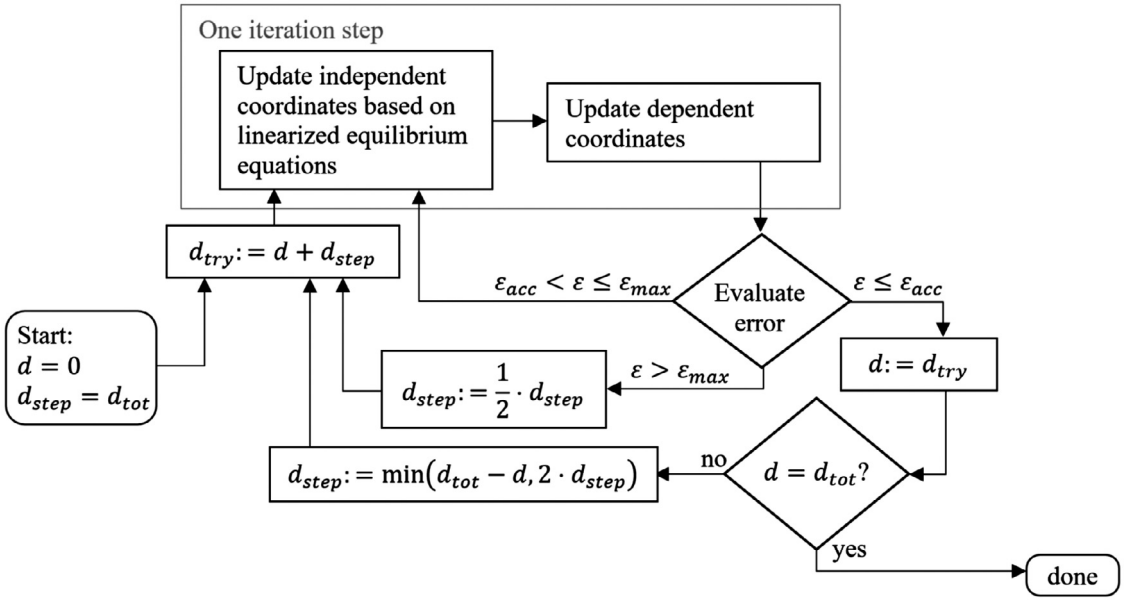

Fig. 18. Schematic overview of the conventional method that is used as a reference for the KSD-method.

References

[1] M. Naves, D.M. Brouwer, R.G.K.M. Aarts, Building block-based spatial topology synthesis method for large-stroke flexure hinges, J. Mech. Robot. 9 (4) (2017) 041006, doi:10.1115/1.4036223.
[2] D. Wiersma, S. Boer, R.G. Aarts, D.M. Brouwer, Design and performance optimization of large stroke spatial flexures, J. Comput. Nonlinear Dyn. 9 (1) (2014) 011016, doi:10.1115/1.4025669.
[3] B.P. Trease, Y.M. Moon, S. Kota, Design of large-displacement compliant joints, J. Mech. Des. 127 (4) (2005) 788–798, doi:10.1115/1.1900149.
[4] D.M. Brouwer, J.P. Meijaard, J.B. Jonker, Elastic element showing low stiffness loss at large deflection, in: Proceedings of the 24th Annual Meeting of the American Society of Precision Engineering, Monterey, CA, 2009.
[5] Folkersma, K., Boer, S., Brouwer, D., Herder, J., Soemers, H., A 2-DOF large stroke flexure based positioning mechanism. *ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* 2012. American Society of Mechanical Engineers. doi: 10.1115/DETC2012-70377.
[6] L.L. Howell, Compliant Mechanisms, John Wiley & Sons, 2001.
[7] B.D. Jensen, L.L. Howell, The modeling of cross-axis flexural pivots, Mech. Mach. Theory 37 (5) (2002) 461–476, doi:10.1016/S0094-114X(02)00007-1.
[8] X. Pei, J. Yu, G. Zong, S. Bi, An effective pseudo-rigid-body method for beam-based compliant mechanisms, Precis. Eng. 34 (3) (2010) 634–639., doi:10.1016/j.precisioneng.2009.10.001.
[9] X. Pei, J. Yu, G. Zong, S. Bi, H. Su, The modeling of cartwheel flexural hinges, Mech. Mach. Theory 44 (10) (2009) 1900–1909, doi:10.1016/j.mechmachtheory.2009.04.006.
[10] T.J. Teo, I.-M. Chen, G. Yang, W. Lin, A generic approximation model for analyzing large nonlinear deflection of beam-based flexure joints, Precis. Eng. 34 (3) (2010) 607–618, doi:10.1016/j.precisioneng.2010.03.003.
[11] N. Li, H.J. Su, X.P. Zhang, Accuracy assessment of pseudo-rigid-body model for dynamic analysis of compliant mechanisms, J. Mech. Robot. 9 (5) (2017) 054503, doi:10.1115/1.4037186.
[12] S. Venanzi, P. Giesen, V. Parenti-Castelli, A novel technique for position analysis of planar compliant mechanisms, Mech. Mach. Theory 40 (11) (2005) 1224–1239, doi:10.1016/j.mechmachtheory.2005.01.009.
[13] J. Rutzmoser, Model Order Reduction for Nonlinear Structural Dynamics, Technische Universität München, 2018.
[14] L. Wu, P. Tiso, Modal derivatives based reduction method for finite deflections in floating frame, World Congress on Computational Mechanics, 2014, doi:10.13140/2.1.4041.8244.
[15] J.P. Schilder, F.M. Segeth, M.H.M. Ellenbroek, M. van den Belt, A. de Boer, Model order reduction of large stroke flexure hinges using modal derivatives, in: Proceedings of the 28th International Conference on Noise and Vibration Engineering, 2018.
[16] Y. Liang, H. Lee, S. Lim, W. Lin, K. Lee, C. Wu, Proper orthogonal decomposition and its applications—part I: theory, J. Sound Vib. 252 (3) (2002) 527–544, doi:10.1006/jsvi.2001.4041.
[17] G. Kerschen, J.-c. Golinval, A.F. Vakakis, L.A. Bergman, The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: an overview, Nonlinear Dyn. 41 (1–3) (2005) 147–169, doi:10.1007/s11071-005-2803-2.
[18] P. Benner, S. Gugercin, K. Willcox, A survey of projection-based model reduction methods for parametric dynamical systems, SIAM Rev. 57 (4) (2015) 483–531, doi:10.1137/130932715.
[19] R.G.K.M. Aarts, J.B. Jonker, Dynamic simulation of planar flexible link manipulators using adaptive modal integration, Multibody Syst. Dyn. 7 (1) (2002) 31–50, doi:10.1023/A:1015271000518.
[20] F. Naets, W. Desmet, Super-element global modal parameterization for efficient inclusion of highly nonlinear components in multibody simulation, Multibody Syst. Dyn. 31 (1) (2014) 3–25, doi:10.1007/s11044-013-9342-2.
[21] O. Brüls, P. Duysinx, J.C. Golinval, The global modal parameterization for non-linear model-order reduction in flexible multibody dynamics, Int. J. Numer. Methods Eng. 69 (5) (2007) 948–977, doi:10.1002/nme.1795.
[22] G.H. Heirman, F. Naets, W. Desmet, A system-level model reduction technique for the efficient simulation of flexible multibody systems, Int. J. Numer. Methods Eng. 85 (3) (2011) 330–354, doi:10.1002/nme.2971.
[23] J. Haringx, The cross-spring pivot as a constructional element, Flow Turbul. Combust. 1 (1) (1949) 313, doi:10.1007/BF02120338.
[24] S.T. Smith, Flexures: Elements of Elastic Mechanisms, CRC Press, 2014, doi:10.1201/9781482282962.

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911

[25] P. Xu, Y. Jingjun, Z. Guanghua, B. Shusheng, Y. Zhiwei, Analysis of rotational precision for an isosceles-trapezoidal flexural pivot, J. Mech. Des. 130 (5) (2008) 052302, doi:10.1115/1.2885507.
[26] S. Henein, P. Spanoudakis, S. Droz, L.I. Myklebust, E. Onillon, Flexure pivot for aerospace mechanisms, 10th European Space Mechanisms and Tribology Symposium, San Sebastian, Spain, 2003.
[27] D.F. Machekposhti, N. Tolou, J. Herder, A review on compliant joints and rigid-body constant velocity universal joints toward the design of compliant homokinetic couplings, J. Mech. Des. 137 (3) (2015) 032301.
[28] R. Craig, M. Bampton, Coupling of substructures for dynamic analyses, AIAA J. 6 (7) (1968) 1313–1319, doi:10.1115/1.4029318.
[29] J.B. Jonker, J.P. Meijaard, A geometrically non-linear formulation of a three-dimensional beam element for solving large deflection multibody system problems, Int. J. Non Linear Mech. 53 (2013) 63–74, doi:10.1016/j.ijnonlinmec.2013.01.012.
[30] M.A. Crisfield, A consistent co-rotational formulation for non-linear, three-dimensional, beam-elements, Comput. Methods Appl. Mech. Eng. 81 (2) (1990) 131–150, doi:10.1016/0045-7825(90)90106-V.
[31] T.N. Le, J.M. Battini, M. Hjiaj, Efficient formulation for dynamics of corotational 2D beams, Comput. Mech. 48 (2) (2011) 153–161, doi:10.1007/s00466-011-0585-6.
[32] M. Iura, S. Atluri, Dynamic analysis of planar flexible beams with finite rotations by using inertial and rotating frames, Comput. Struct. 55 (3) (1995) 453–462, doi:10.1016/0045-7949(95)98871-M.
[33] G. Cowper, The shear coefficient in Timoshenko's beam theory, J. Appl. Mech. 33 (2) (1966) 335–340, doi:10.1115/1.3625046.
[34] J. Jonker, J. Meijaard, SPACAR—computer program for dynamic analysis of flexible spatial mechanisms and manipulators, in: Multibody Systems Handbook, Springer, 1990, pp. 123–143.

Please cite this article as: K. Dwarshuis, R. Aarts and M. Ellenbroek et al., Kinematically started efficient position analysis of deformed compliant mechanisms utilizing data of standard joints, Mechanism and Machine Theory, https://doi.org/10.1016/j.mechmachtheory.2020.103911