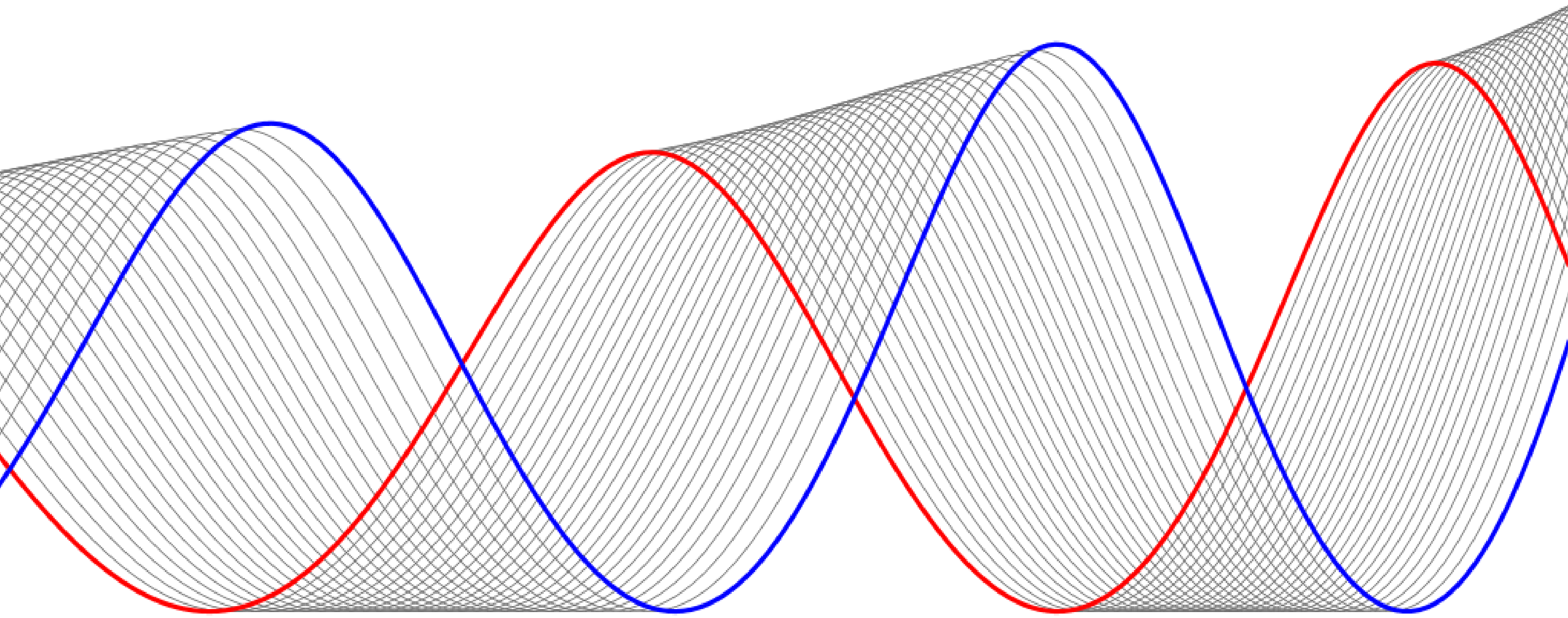
Faculty of Mechanical, Maritime
and Materials Engineering

An intuitive method to design load-displacement characteristics for nonlinear springs in parallelogram linkages

Roel van Ekeren

September 2019



Master of Science Thesis

AN INTUITIVE METHOD TO DESIGN LOAD-DISPLACEMENT CHARACTERISTICS FOR NONLINEAR SPRINGS IN PARALLELOGRAM LINKAGES

by

Roel van Ekeren

in partial fulfillment of the requirements for the degree of

Master of Science
in Mechanical Engineering

at the Delft University of Technology,
to be defended publicly on Monday September 30, 2019 at 14:00 AM.

Supervisors:	Ir. J. Rommers	TU Delft
	Ir. A. Zondervan	Hittech Multin BV
Thesis committee:	Prof. dr. ir. J. L. Herder	TU Delft
	Dr. M.A. Bessa,	TU Delft
	A. Geelkerken,	Hittech Multin BV

Preface

This thesis concludes my master Mechanical Engineering after two years at the faculty Mechanical, Maritime and Materials Engineering at the TU Delft. Formally, this thesis finalizes the master track Bio Mechanical Design I followed in first year. The project itself is carried out at the precision- and micro-systems department, and is partially dedicated to the company Hittech Multin BV, a system supplier in the high-tech industry.

It appeared to me that this is the place to express my gratitude and I am glad to mention the people who made it possible to finish the master Mechanical Engineering.

In the first place, I would like to thank my supervisor, Jelle Rommers, for his incredible positive and helpful feedback at any time. I enjoyed our meetings and time was always short. Besides, I would like to thank Arnold Zondervan for his valuable time, supporting me throughout the thesis on the company side, even when he was super occupied by his own work or just started family.

Also, I would like to mention and thank prof. Just Herder for his valuable time and valuable feedback during our meetings. The interesting lectures at Precision Mechanism Design made me decide to do the graduation project at the PME department and I am very happy I did!

Furthermore, many thanks to Ard Geelkerken for his practical feedback at Hittech and together with Miguel Bessa for being interested, reading my work and taking part in the committee. Also I enjoyed the meetings with Giuseppe and his feedback was very helpful, thank you for that. You have made me very enthusiastic about nonlinear springs by the interesting conversations we had.

A special thanks to my brother Wim, genius and "schoolvoorbeeld student", always willing to listen to my problems. Thank you for your motivating speeches and reviewing my work! Also I would like to mention Arie, my eldest brother for being always so optimistic and proud about my work. I enjoyed playing tennis with you and I always will! Many thanks to my friends and of course I also want to thank Natalie for her confidence in me and her endless support. At last, I am very thankful to my parents who made it possible that I can finish this great study.

Roel van Ekeren
Delft, September 30, 2019

Contents

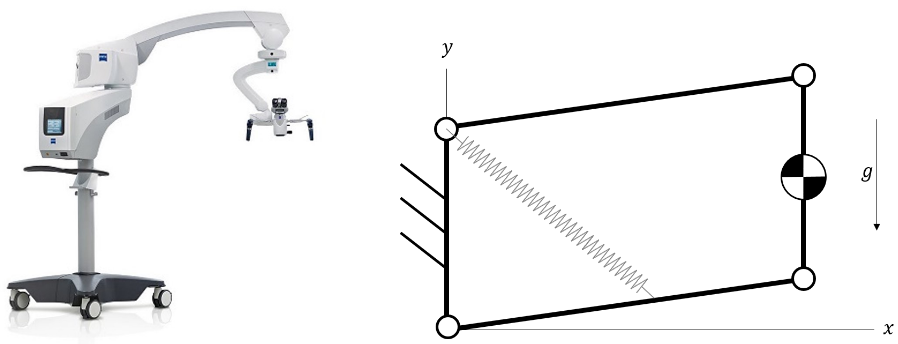
1
Introduction

This thesis includes several independent contributions and is conducted in collaboration with both university and Hittech Multin BV. The reader is especially encouraged to read the paper in chapter 3 with the thesis title, as it introduces the problem compactly and focuses on the presentation of a new design methodology for nonlinear springs to find load-displacement characteristics for the end effector of parallelogram linkages. In this chapter the project background is introduced, to provide a better understanding of the context of this thesis. Next, the scope and problem statement will be given and its relevance discussed. After, thesis goals are stated, followed by a short outline of the report.

1.1. Project Background

Many springs are designed for the purpose of storing potential energy, often with the goal to have a special load-displacement characteristic. The design of very specific nonlinear load displacement functions can be found in various mechanical systems. To introduce the problem, let us consider the following application.

Many surgical procedures require a high level of accuracy and precision. An essential aid is the surgical microscope allowing surgeons to enhance their view on the working area to perform on a specific level of detail. Positioning and adjusting the microscope near the working area by using joysticks is an important procedure, which is performed multiple times per operation, taking up a significant part of the total surgery time [1]. The microscope is suspended to a mobile support system. (figure 1.1a) For most microscope supports a key feature is to statically balance the mass of microscope and the support in such way that the user, in this case the surgeon, will experience the instrument to be weightless.



(a) Microscope mobile support system. The statically balanced parallelogram is incorporated in the top arm to compensate the mass of the microscope at the outer end.

(b) Statically balanced parallelogram for the payload using conventional helical zero-free-length spring.

Figure 1.1: Conventional method to statically balance a parallelogram, implemented in industrial applications such as the microscope support.

To balance the mass of a microscope, a force compensation mechanism is required. There are two methods to compensate forces: active and passive. The method of active force compensation involves external energy, for example a actively controlled actuator. Passive approaches, such as implementation of springs, use potential energy and do not require external energy. Therefore, passive methods are often preferred. For a common industrial application such as the mobile support system a parallelogram linkage is used in combination with a conventional linear helical spring to compensate the mass of the microscope. The parallelogram -applied in many more fields [2]- is particular useful for these type of applications, because it keeps the end-effector in parallel with the reference. The spring and mass are in equilibrium, because the potential energy of the mass is compensated by the potential energy of the spring, meaning the system is statically balanced. This shows the governing principle of static balance: constant potential energy of the total system for the applied range of motion. Statically balanced mechanisms have many advantages, including: compensation of undesired forces, energy free motion, improved performance and inherent safety. Incorporating static balancing from the beginning of the design process will reduce parts and leads to higher performance of the product [3]. The method of force compensation in the mobile support system is frequently seen in mechanical devices and was first introduced by Carwardine in his patents from 1931-1935 [4], and later studied by French et al., [5] and Herder [3]. The nonlinear load-displacement characteristic of the unbalanced system is reached by implementing a zero-free-length linear spring across the links such that is uses the geometry of the parallelogram as shown in figure B.1. In practice this zero-free length is emulated by the use of pulleys or by normal springs.

1.2. SCOPE AND PROBLEM STATEMENT

In designing passive force compensation mechanisms, several challenges arise. The first challenge is to counteract the forces throughout the entire range of motion, such that the system is in equilibrium at any position. In practice, this equilibrium is often not perfectly accurate due to practical limitations, such as emulating a zero-free-length spring. Besides, other external forces than gravity can act on the mechanism such as the elastic forces in compliant mechanisms. Another challenge is to make the force compensator as compact as possible. For the application of a microscope support, compactness of the force compensation mechanism leads to a more lightweight design, making the overall system less bulky. Furthermore, with the same amount of space more payload can be balanced. For many other applications a more compact design is beneficial, especially in the field of exoskeletons. A more challenging aspect is to adjust the spring to a new payload. These problems arise in other applications as well [6] and Extensive research is done in this field by [3],[7].

Helical linear springs are not always the best option. As an alternative, nonlinear springs bring opportunities because of their design freedom [8] and shape. The use of nonlinear springs, however, is a relatively complex and large field with respect to linear springs. More insight in the design of nonlinear springs can help future designers. Translating those insights into design tools will be valuable for designers in the field of nonlinear springs. As a small step towards the understanding of nonlinear spring design, the scope is narrowed down to the implementation of nonlinear springs in parallelogram linkages as potentially compact force compensator for prescribed load-displacement functions. Adaptability of the spring is left out of scope and can be a next step in the design of nonlinear springs.

1.3. RELEVANCE

Many products, devices and other engineering applications benefit from use of static balancing [3]. Limiting the volume occupancy remains a challenging task for designers, especially when it may not violate other requirements. Research to the volume use of spring mechanisms may give engineers understanding for the design of more compact mechanism design. Furthermore this research may contribute to scientific understanding in the design of nonlinear springs.

Society may benefit from the applications of more compact statically balanced devices. In recent years, exoskeletons have drawn more attention. Workers can be relieved from heavy loads and people with muscle diseases may be able to deal with more daily activities because of the support from a force compensator. Reduction of volume of the exoskeleton improves the user interface and experience.

The performance of industrial applications is enhanced by the use of static balance. Not only microscope supports, but many robotic manipulators, pick and place arms, and positioning systems in the high-tech industry can be improved. Passive compensation of forces increases efficiency due to faster operating times and reduction of external power. Besides, more compact force compensation mechanisms lead to more lightweight design due to reduction materials, thereby reducing the product costs.

1.4. THESIS OBJECTIVE

This thesis focusses on the challenge how to make force compensation mechanisms more compact. For that purpose the following research goals are established. The scope of the literature review is narrowed down to spring based force compensation mechanisms. The scope of the paper is narrowed down to the implementation of nonlinear plate springs in parallelogram linkages.

- Provide an overview of the volume occupancy of spring based force compensation mechanisms in literature.
- Investigate implementation of nonlinear plate springs in parallelogram linkages for the design of force compensation mechanisms.

In theory the second research goal is related to the first research goal, with the underlying purpose to investigate if nonlinear plate springs can store the same or even more potential energy in the volume of a parallelogram linkage. Therefore, it is important to find out if nonlinear springs are suitable for force compensation in parallelograms. Subsequently, the possibility to stack multiple nonlinear springs in parallel can be investigated.

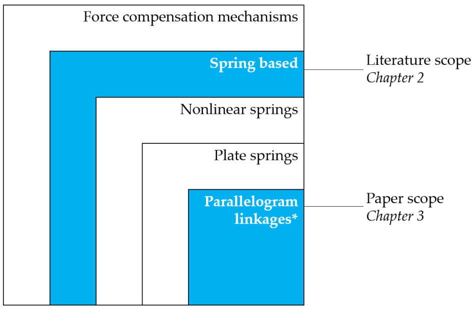


Figure 1.2: Scope of the research: the literature review involves the field of spring based force compensation mechanisms. The paper focusses on the field of nonlinear plate spring implemented in parallelogram linkages. *note: this is not a complete illustration of all fields. For example: parallelogram linkages appear also in non-spring based systems.

1.5. THESIS OUTLINE

This thesis includes 6 chapters and is structured as follows: first, a literature review will be presented in chapter 2, which deals with the first research goal. The second research goal is handled by the paper presented in chapter 3, which is also the main contribution of the thesis. After, the discussion is presented in chapter 4 and main conclusions of the thesis are summarized in chapter 5. Moreover, recommendations for future work are given in chapter 6 and additional work related to the thesis is presented in the appendices A B. In addition, programming code is provided in appendices C D.

2

Literature review - Comparison of spring force compensation mechanisms literature

Comparison of spring force compensation mechanisms in literature

Roel van Ekeren

Delft University of Technology
Department of Precision and Microsystems Engineering, Mekelweg 2, 2628 CD, Delft, Netherlands

January 27, 2019

Nomenclature

A	Metric 1: Accuracy [-]
ED	Metric 3: Energy Density $[J/m^3]$
FCM	Force compensation mechanism
$FCMV$	Force compensation mechanism volume
GCM	Gravity compensation mechanism
R_{VE}	Metric 4: Volume Efficiency Ratio [-]
$RMSE$	Root mean squared error [-]
ROM	Metric 2 : Range of motion divided by diagonal of FCMV[-]

1. Introduction

Force compensation mechanisms (FCM) use the principle of static balancing to relieve an unbalanced system from undesired forces to improve the overall performance. A gravity compensation mechanism (GCM) is a special FCM that only counteracts the forces of gravity, which remain constant. GCMs belong therefore to the category constant force mechanisms. GCMs are widely applied, from industry to society and differ in size, weight, system performance and range of motion. Industrial robotic arms run heavy duty cycles. The GCM relieves the robot arm from gravity forces, resulting in lower energy use and increasing efficiency and performance. In the application field of orthotics and exoskeletons GCMs are used to counteract the gravity forces acting on the human body. Incorporating the GCM into the functional design is challenging due to comfort and volume constraints. Generally, the more volume is occupied, the heavier and more expensive the system becomes. Reduction of the volume of a FCM requires the system to store the same amount of potential energy on a smaller volume. More compact FCM design in terms of higher energy density contributes to smaller machines for industry and consumer products. Insight in the performance of present literature could improve development on FCMs.

1.1. Static Balancing

The governing principle behind static balancing is that the total potential energy within a mechanical system remains constant for a prescribed range of motion. [1] This means that the only required energy to move the system is used to accelerate and decelerate. A gravity equilibrator is designed to statically balance a mass and is in equilibrium if the balancing mechanism counteracts the moment exerted by the mass of the system and its payload, thereby removing any operational energy. Constant potential energy for gravity equilibrators can be established in various ways. The two most common methods are balancing by counterweights and balancing by springs. Other methods involve pneumatic or hydraulic cilinders, or the use of electromagnetic effects.[2] [3].

Spring elements can efficiently store potential energy by compression or extension. These flexible storage elements are advantageous because they are simple, mechanical, passive, relative compact components which are suitable for implementation. In contrast to springs, the use of counterweights is generally not preferred. An important disadvantage in the use of counterweights is the increased mass and inertia of the total system.

Static balance is applied in numerous application fields to counteract the weight, payloads or reaction forces of the system. Examples are robotics [ref], orthotics and assistive devices [4], [5] or the famous Anglepoise desk lamps. [6] Generally, statically balanced systems include the following beneficial features: compensation of undesired forces, energy free motion, full energy exchange, improved information transmission, energy free force control, elimination of backlash, zero stiffness, neutral buoyancy, improved performance and inherent safety. Incorporating static balancing from the beginning of the design process will reduce parts and leads to higher performance of the product. [1]

1.2. Application

The application focused on in this research is a surgical microscope depicted in figure 1. Many surgical procedures require a high level of accuracy and precision. An essential aid is the surgical microscope allowing surgeons to enhance their view on the working area to perform on a specific level of detail.

(a) (b)

Figure 1: *Microscope and balanced microscope stand*

Positioning and adjusting the microscope near the working area by using the joysticks is an important procedure, which is performed multiple times per operation, taking up a significant part of the total surgery time [7]. The microscope is suspended to a mobile support system. For most microscope supports a key feature is to statically balance the weight of microscope and the support in such way that the user, in this case the surgeon, will experience the instrument to be weightless. The floating instrument is now a more manageable device [1b].

1.3. Objectives

This literature review presents an overview of existing methods to compensate forces using springs, but also presents other spring force generators. Furthermore, the existing methods are compared on performance to create insight in the volume occupancy of force compensation mechanisms. To summarise the goals of this literature survey:

- Obtain an overview of spring force generators
- Determine key performance indicators to compare spring force generators
- Compare the literature on performance indicators.

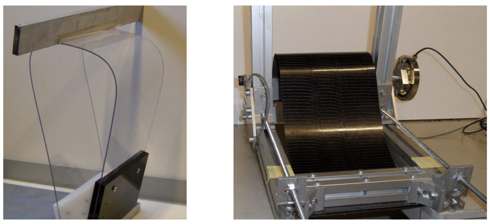
The outline of the literature survey will follow the described goals. First an extensive body of literature will be discussed. In this section also advantages, disadvantages and key features of existing force compensation mechanisms are reviewed. Second, metrics will be defined that can position the reviewed literature in perspective of their performance. Third the found literature will be compared and discussed on the described metrics. Also an overview is given of the collected data and presented in an energy density design chart. A conclusion will be drawn from the state of the art and the classified literature.

2. Force compensation mechanisms in literature

To get a more general picture of the used spring concepts in literature, not FCMs have been taken into account, but also other spring force generators. The body of literature in the field of SFG can be distinguished into four groups. These four groups are formed by combinations of individual parts. An overview is shown in figure 11. Generally, SFG comprises one or multiple spring elements and if necessary a transmission to facilitate the non-linearity. Group one consist of SFG which use one nonlinear spring element without any link or transmission to generate the desired load-displacement function. Group two consists of SFGs which are made of multiple linear or nonlinear spring elements. The third group includes both multiple spring elements as a transmission. The last group includes only a single spring element in combination with a transmission. Linear springs are used more frequently because of their simplicity and availability. In contrast, nonlinear springs are harder to implement, because they are application specific. However, if the nonlinear spring is properly designed, no auxiliary mechanism or transmission is required to balance a system with nonlinear behaviour. First an overview will be given of the systems in literature for each group. A small discussion ends the chapter.

2.1. Class 1: Single Spring

The first group of SFGs are systems that only use a single (non-linear) spring. If a linear spring is used the nonlinear system will only be compensated with high accuracy for a small range of motion. An example is described in a paper by Gopalswamy [24]. He proposed a simple configuration for gravity compensation of a parallelogram linkage using a single linear torsion spring in the main axle. The linear spring limits the range of motion to the linear part of the moment curve and can reduce the systems forces to a specific level. For higher accuracy a better non-linear spring is required. Nonlinear springs are however application-specific. Promising prototypes from recent research by Radaelli [8], [9] show that is possible to compensate nonlinear systems with high accuracy. (figure 2). In his dissertation [10] Radaelli describes several concepts to synthesise non-linear springs. The dissertation is also collection of papers in which new concepts for non-linear springs are prototyped and analysed. The non-linearity can be created by special curves, shapes, widths and preloads.

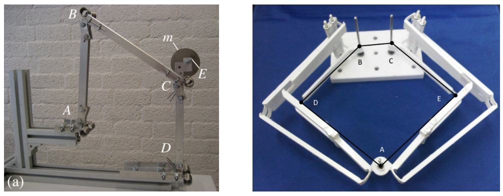


(a) *[11]* **(b)** *[12]*

Figure 2: *Monolithic gravity balancers based on shape optimization*

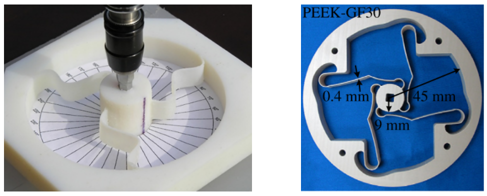
2.2. Class 2: Multiple Springs

Similar to class 1, the second class only uses springs to compensate the mechanism forces. This class is mostly found in compliant mechanisms where hinges are replaced by compliant joints. The flexible members act as springs storing elastic energy. An example of such balanced mechanism is proposed by Radaelli [13] where all joints are replaced by prestressed torsion springs. The springs are designed such that the total mechanism counteracts the torque done on the system by its weight and the payload. The pendulum with the mass requires an auxiliary arm, connected to the pendulum such that the non-linear behaviour can be achieved by the complete set of springs and links. A fully compliant nonlinear variant to this system is a five-bar mechanism by Merriam [14] in the horizontal plane. In this mechanism all the hinges are replaced by lumped compliant joints. After optimizing the dimensions and preloads of the joints the input force required to actuate the device can be eliminated. More examples of constant force mechanisms are presented in figure 4. Here multiple springs are designed to generate a constant torque mechanism. [15], [16]. Constant force linear motion stages are also proposed by Wang [17] and Tolman [18]. Another constant force end effector using two different springs is proposed by Chen [19] where the force can be adapted. Also fully statically balanced compliant grippers are known in literature [20] [21] [22]. Merriam and Radaelli (figure 3 managed to use only the joint space for the spring mechanism. However, the joint space is limited and the space between the links is not used. For the nonlinear springs, energy capacity could be increased by enlarging the width of the springs.



(a) *[13]* **(b)** *[14]*

Figure 3: *Spring force compensation systems based on compliant joints.*

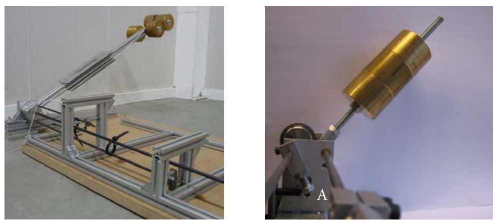


(a) *[16]* **(b)** *[15]*

Figure 4: *Monolithic constant torque mechanisms*

2.3. Class 3: Multiple Springs and Transmission

The third class involves mechanism with multiple springs and a transmission to generate the desired load-displacement function. This transmission mechanism can be a four-bar mechanism, end stops, a combination of links or otherwise. Recent research related to the Holland Container Innovation [23] proposed the use of torsion bars. Since torsion bars are loaded on pure shear, they make efficiently use of the material. Also, the torsion bar could fit with its length easily in the bottom hinge joint of a container. Research is done how to achieve the nonlinear behaviour in combination with the linear torsion bar.



(a) *Prototype by Radaelli [24] balances 25 Nm using multiple torsion bars* **(b)** *Scaled prototype by Claus [23] from thesis using spurs transmission balances 0.4 Nm*

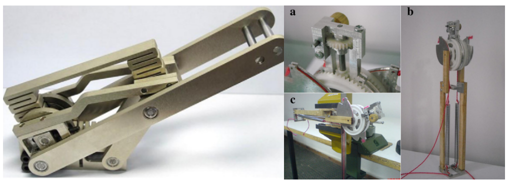
Figure 5: *Gravity balancers using prestressed torsion bars and end stops*

The research conducted by Claus and Radaelli [23] [24], focused on approximate balancing using torsion bars. By positioning preloaded energy storage elements, for instance torsion bars, in series and parallel, end stops can realise a discrete stiffness profile, thus achieving an approximation to counteract the exerted moment. 5 However, such systems include many parts and can accommodate only positive stiffness. A system using the discrete approximation principle achieving negative stiffness is not known.

Another variant to such transmission balancers is the design by van Osch [25]. Torsion bars are used as spring elements and a cam-wire transmission ensures equilibrium for the specified range of 90 degrees. The cam-wire mechanism inherently limits the range of motion. A perfectly static balanced mechanism using torsion bars was not known, but the system still uses a bulky transmission. Kilic [26] proposed an simpler and smaller method for a non-linear spring mechanism using wrapping cams and a pulley connected to linear coil springs. However, comparison is difficult since insufficient data is provided. Another example of a cam mechanism is described by Liu [27]. Two linear springs and a cam mechanism were used to produce a constant force mechanism. These cam roller mechanism are also applicated in statically balanced brakes. [28]. In literature also cam based mechanism designed specifically for gravity balancing were found. [29] [30]

Next to cam-based transmissions various mechanisms in literature make use of linkages to generate the gravity load-displacement function. [31][32]

Within the found body of literature the most compact designed prototype appeared to be a variable gravity equilibrator incorporating a non-linear mechanism using linear compliant compression and extension springs.[31] The extension spring provides stiffness while the compression spring provides negative stiffness through the mechanism links. The pretension can be varied by a screwdriver to adjust the mechanism to different payloads.



(a) *Compact gravity balancer based on folded compliant springs[31]* **(b)** *Cam based mechanism with adjustable cam for other payloads[26]*

Figure 6

2.4. Class 4: Single Spring and Transmission

The fourth class involves single spring mechanisms using a transmission. A frequently seen mechanism is the balanced pendulum or parallelogram having an ideal helical spring [33] [34] attached to the linkage and vertically in line with the hinge, simplified shown in figure 7 [1]. Basically, the linkage provides the non-linear transmission. This method is currently implemented in most of the microscope suspension systems. Pully arrangements can be made to move the springs to desired positions in the mechanism. Additional features were devised to adjust the balancer to various payloads and even energy free adjustment methods are known [35] [36] [37] [38] [39]. If perfect balancing (i.e. with high accuracy) is desired, such springs are very effective. Also in terms of energy stored per unit mass these springs perform well: a helical spring can very efficiently store energy because the whole wire is loaded on shear. [40] Despite the fact that such configurations with efficient helical springs are readily available and affordable, they take up a lot of building space. [23] Furthermore, more space is required during operation because of extension of the springs. Moreover, to emulate a zero-free length spring, cable-pulley configurations are required which take up more space. Space occupancy is disadvantageous since the working area needs to be free for surgeons, tools, the patient and other instruments.

(a) *[6]* **(b)** *[41]*

Figure 7: *Spring balancers using emulated or zero-free-length springs and the geometry of the parallelogram as 'transmission'.*

An interesting relating example is a gravity equilibrator, designed by Bijlsma [**?**], comprising a clever planetary gear allowing unlimited range of motion, i.e. 360 degrees and more. The mechanism uses a torsion bar as well and the transmission system is less bulky. A disadvantage of this systems is the concern of significant hysteresis and wear in the gears, which is a limiting factor for accurate systems.

3. Performance metrics for FCM's

In the preceding literature examples we have seen various advantages and disadvantages which can be assigned to performance indicators. The following four key performance indicators can be distinguished which are used to compare literature. The metrics will be used to evaluate future work as well.

3.1. Accuracy

The accuracy metric can be interpreted as the percentage of the system forces cancelled by the balancing mechanism along its range of motion. Since the system is not perfectly balanced along the displacement interval, the root mean square error between the spring force and the unbalanced system determines the average error.

$$A = 1 - RMSE \; [-] \tag{1}$$

where

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\left(\frac{f_n - F_n}{f_n}\right)^2} \tag{2}$$

where f_n is the spring force and F_n represents the unbalanced system force at interval n. A perfect balanced system has 100% accuracy, meaning that all forces are compensated by the spring.

3.2. Range of Motion

The range of motion metric is a ratio to determine the relative travel distance of the mechanism with respect to its size. The range of motion is described by the upper limit for the given accuracy minus the lower limit divided by the diagonal of the force compensation mechanism volume in meters.

$$ROM = \frac{UL - LL}{D} \; [-] \tag{3}$$

Where LL specifies the lower limit and UL the upper limit. Both are described in height (vertical) meters. If a system is not a gravity equilibrator, but for example a rotating constant force mechanism the upper and lower limits are measured in radians. For these applications a height is calculated by the sine of the angle of the range of motion, multiplied by the link of the system. If the system does not have a link, the length of the link is assumed to be the same as D, where D specifies the diagonal of the force mechanism volume (FCMV) in meters shown in figure 8.

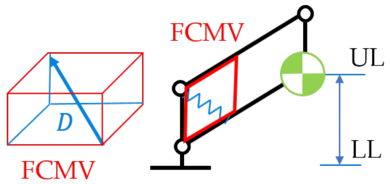


Figure 8

For some systems it is not necessary to achieve a large range of motion, for instance constant force graspers. [20] Other systems require a certain minimum range of motion for example an orthosis to compensate limbs. [42] Therefore it is very situation specific. For designers it is important to specify this key performance indicator early in the design process. Generally a larger range of motion means that the system is wider applicable, and is therefore more relevant.

3.3. Energy density of spring

Various methods are used in literature to quantify the energy density of springs. Cool uses a metric to quantify the maximum amount of energy that is possible to be stored in the spring material before yielding. [40]. A variant on this metric was proposed by Krishnan et al. [43]. Both materials take the material volume of the spring into account. The fraction of material that is maximally used for strain energy we refer to as UMV (used material volume) However, is it also interesting to know what the efficiency is with respect to the occupied mechanism volume.

The combination of the spring mechanism volume and the absolute energy capacity gives useful information for designers about the springs performance in relation to actual occupied volume. Despite a high UMV efficiency, the spring can be inefficient with respect to its FCMV. Therefore the following metric provides a rate of compactness in relation to its energy capacity. This metric will be described in terms of absolute stored strain energy in Joule divided by the FCMV in m^3.

$$EnergyDensity = \frac{Work\ Compensated}{FCMV}\ [J/m^3] \tag{4}$$

where the work compensated is the amount of energy that is possible to be stored by the FCM. A spring compensation system can be implemented more generally if it possesses more energy capacity within a smaller assembly volume. If we look at the conventional coil spring we see an efficiency on material level, but the space occupied by its cylindrical shape is not very efficient.

3.4. Volume Efficiency

The last metric is defined to obtain information about the volume efficiency, or in other words, the amount of volume that is lost to overall design considerations. A high volume efficiency means that more volume of the FCMV is used for energy storage. The metric is defined by the amount of volume used for strain energy, divided by the FCMV.

$$R_{VE} = \frac{UMV}{FCMV}\ [-] \tag{5}$$

where UMV is the used material volume. The UMV can best be illustrated by figure 9. The boxes on top represent volume ratios whereas the volume efficiency ratio is defined by:

$$R_{VE} = R_{material} \cdot R_{unit} \cdot R_{stacking} \cdot R_{system} \tag{6}$$

The grey boxes represent the occupied volumes. The complete system, -the top grey rectangular bar-, represents the total system volume of 100%. The balanced system can be split into two volumes: the FCM volume, and the volume of the inbalanced system. Again, the FCM volume can be split into the volume of the spring system itself and the volume of a potential transmission, if there is one.

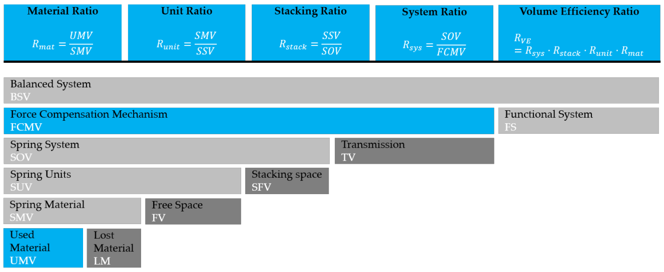


Figure 9: *Overview of volume distribution. The top grey bar represents the volume of the entire system. Below the volumes are split into useful volume and losses. The losses are illustrated in darker grey. On top four ratios, based on the volume bars below, are presented in the blue boxes.*

The spring system includes all energy storage elements, the springs. This could be either one or multiple springs. If there are multiple springs involved, the spring system can be split into spring unit volumes and stacking space, volume that is needed in order to prevent spring collisions. The space that is lost by stacking is referred to as the stacking ratio. The stacking distance is explained by figure 10 where the spring unit cells are defined by the red rectangular blocks.

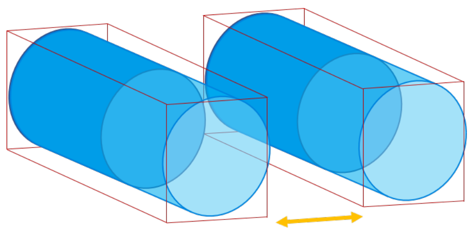


Figure 10: *Illustration of the spring material. On the left the distance between spring material determines the amount of volume that is lost to stacking of springs.*

If we proceed downwards in figure 9, the spring unit volume is split into spring material volume and free space. This can be explained as the spring unit ratio, the amount volume that is lost to free space around the spring material. For instance, the cylindrical space within a coil spring is not used for energy storage. At last the material ratio is defined by the amount of volume within the material that is used for pure energy storage. The illustration on the right of Figure 10 shows a torsion wire that is loaded. The stress distrubution is shown and increases linearly from the inside. As a result only 50 % of the material is used for energy storage. In conclusion only a small part of the FCM will store potential energy. Increasing the Volume Efficiency of the FCM will lead to higher energy densities.

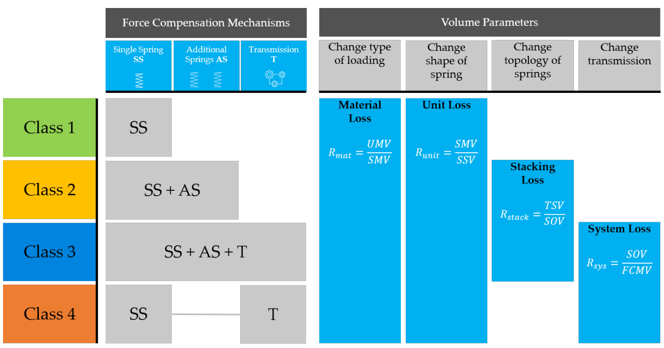


Figure 11: *On the left, four classes are defined by the blue components on top. Class 1 involves spring systems with only 1 spring. Class 2 involves systems with multiple springs. Class 3 involves multiple springs and a transmission. Class 4 involves a single spring with transmission. On the right, an overview is shown for which classes and components volume is lost.*

The volume ratios can be used to identify volume losses. For example this means how much volume is lost by the transmission of the system. Figure 11 shows the four groups. On the right, the volume parameters are shown. Class 1 does not involve stacking and system ratios, because no volume is lost by stacking or transmission. Class 2, however, loses volume to stacking because multiple springs are involved. Class 3 loses volume on all four ratios and class 4 no volume is lost to stacking since only single springs are involved. For example, ways to increase the volume efficiency of class 4, is to change the type of spring, to change the shape or change the volume of the transmission.

4. Performance of Literature

Based on the four performance metrics, literature can be compared and possible opportunities can be extracted. Table 1 shows an overview of several gravity compensation mechanisms and spring force generators, discussed in the previous paragraphs. The table is plotted for the volume metrics, 3 and 4, since we are interested in how volume is used. Metric 1 and 2 is used to quantify the balancing performance. Figure 12 shows an overview of the prototypes. The horizontal axis represents metric 3, the energy density of the systems in $[J/m^3]$. The vertical axis represents metric 4, the volume efficiency. Many papers are excluded because they provide insufficient information about the discussed performance metrics.

The papers that provided sufficient information are listed in the table. For some other papers hold that the provided information was unclear or not shown, so numbers had to be guessed based on other information or pictures. The table gives a rough estimation and calculates the energy density ratios for the enlisted papers. The table is not complete and can be supplemented by new or unseen papers. More importantly, the numbers can be improved by providing more accurate results of the experiments and prototypes. For now it gives an indication where the prototypes can be found on the energy density scale. The figure also provides information about the FCM classes, discussed in section 2. The colors of the prototypes correspond with figure 11.

5. Discussion

The plot from figure 12 provides insight which systems use their volume well, and which systems do not. The lower left corner includes systems which have a low energy density and a low volume efficiency. The upper right corner includes systems with higher volume efficiency and higher energy density. The presented comparison data is based on raw data extracted from literature papers. A lot of papers do not provide sufficient data in order to calculate the performance metrics. Theses papers were therefore not included in the overview. The systems that were compared show very low volume efficiencies. Also the comparison may be not completely fair, because the individual spring systems were not designed for volume efficiency. Most of the systems are prototypes and proof of concepts. The accuracy data was in most case clearly provided. The data of range of motion was most of the times provided, but metric 2 could often not be calculated since the mechanism volume was not known. If more data is available, industrial systems could be compared on volume occupancy by the ratios provided in this paper. From the presented plot no particular relation can be extracted.

6. Conclusion

Volume occupancy in the design FCM is an important aspect for engineers. Insight where to gain higher volume efficiencies in terms of energy storage can improve the overall compactness of the system. This literature study provides a structured perspective on volume losses in FCMs. Four metrics are presented which compare literature on performance. Apparently the present state of literature shows that a very small amount of volume is effectively used for storing strain energy (<3% of the total FCMV). Future designs can focus on volume improvements based on four classes that are defined by their component levels to create more compact force compensation systems.

References

[1] Herder, J. L., 2001, "Energy-free systems: theory, conception, and design of statically balanced spring mechanisms," Ph.D. thesis, Delft University of Technology, doi:10.13140/RG.2.1.3942.8966, `http://repository.tudelft.nl/view/ir/uuid:8c4240fb-0315-462a-8b3b-efbd0f0e68b6/`

[2] Westerman, S., 2015, "Design of a statically balanced mechanism using magnets and springs," Ph.D. thesis, Delft University of Technology.

[3] Gosselin, C. M. and Wang, J., 2000, "Static balancing of spatial six-degree-of-freedom parallel mechanisms with revolute actuators," Journal of Robotic Systems, **17**(3), pp. 159–170.

[4] Banala, S., Agrawal, S., Fattah, A., Rudolph, K., and Scholz, J., 2004, "A gravity balancing leg orthosis for robotic rehabilitation," IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004, (May 2014), pp. 2474–2479 Vol.3.

[5] te Riele, F. L. and Herder, J. L., 2001, "Perfect Static Balance with Normal Springs," *Perfect Static Balance with Normal Springs*, Figure 9, pp. 1–8.

[6] French, M. J. and Widden, M. B., 2000, "The spring-and-lever balancing mechanism, George Carwardine and the Anglepoise lamp," Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, **214**(3), pp. 501–508.

[7] de Wit, N., 2017, "Vibration dissipation in a surgical microscope support system," Tech. rep., Delft University of Technology.

[8] Radaelli, G. and Herder, J. L., 2016, "Shape optimization and sensitivity of compliant beams for prescribed load-displacement response," Mechanical Sciences, **7**(2), pp. 219–232.

[9] Radaelli, G. and Herder, J. L., 2016, "Shape optimization and sensitivity of compliant beams for prescribed load-displacement response," Mechanical Sciences, **7**(2), pp. 219–232.

[10] Radaelli, G., 2017, "Synthesis of mechanisms with prescribed elastic load-displacement characteristics," Ph.D. thesis, Delft University of Technology, doi:10.4233/uuid.

[11] Radaelli, G. and Herder, J. L., 2016, "A monolithic compliant large-range gravity balancer," Mechanism and Machine Theory, **102**, pp. 55–67.

[12] Radaelli, G. and Herder, J. L., 2014, "Isogeometric Shape Optimization for Compliant Mechanisms With Prescribed Load Paths," *Proceedings of the ASME 2014 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, doi:10.1115/DETC2014-35373, `http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?doi=10.1115/DETC2014-35373`

[13] Radaelli, G. and Intespring, B. V., 2011, "An energy approach to the design of single degree of freedom gravity balancers with compliant joints," *Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2011*, ASME, Washington DC.

[14] Merriam, E. G., Colton, M., Magleby, S., and Howell, L. L., 2013, "The Design of a Fully Compliant Statically Balanced Mechanism," *Proceedings of ASME 2013 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, August, doi:10.1115/DETC2013-13142, `http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?doi=10.1115/DETC2013-13142`

[15] Hou, C. W. and Lan, C. C., 2013, "Functional joint mechanisms with constant-torque outputs," Mechanism and Machine Theory, **62**, pp. 166–181.

[16] Nair Prakashah, H. and Zhou, H., 2016, "Synthesis of Constant Torque Compliant Mechanisms," Journal of Mechanisms and Robotics, **8**(6), p. 064503.

[17] Wang, J.-Y. and Lan, C.-C., 2014, "A Constant-Force Compliant Gripper for Handling Objects of Various Sizes," Journal of Mechanical Design, **136**(7), p. 071008.

[18] Tolman, K. A., Merriam, E. G., and Howell, L. L., 2016, "Compliant constant-force linear-motion mechanism," Mechanism and Machine Theory, **106**, pp. 68–79.

[19] Chen, Y.-H. and Lan, C.-C., 2012, "An Adjustable Constant-Force Mechanism for Adaptive End-Effector Operations," Journal of Mechanical Design, **134**(3), p. 031005.

[20] Lamers, T., 2012, "Design of a Statically Balanced Fully Compliant Grasper Using the Rigid Body Replacement Method," **MSc**.

[21] Stapel, A. and Herder, J. L., 2004, "Feasibility study of a fully compliant statically balanced laparoscopy grasper," Proceedings of the ASME Design Engineering Technical Conference, (January 2004), pp. 1–9.

[22] Steutel, P., Kragten, G. A., and Herder, J. L., 2010, "Design of an Underactuated Finger With a Monolithic Structure and Largely Distributed Compliance," Volume 2: 34th Annual Mechanisms and Robotics Conference, Parts A and B, (December 2015), pp. 355–363.

[23] Claus, M. R., 2008, "Gravity balancing using configurations of torsion bars," Ph.D. thesis, Delft University of Technology, doi:10.1006/jdeq.1996.0111.

[24] Radaelli, G., Buskermolen, R., Barents, R., and Herder, J. L., 2017, "Static balancing of an inverted pendulum with prestressed torsion bars," Mechanism and Machine Theory, **108**(July 2016), pp. 14–26.

[25] Osch, F., 2011, "Design of an adjustable gravity equilibrator using torsion bars," .

[26] Kilic, M., Yazicioglu, Y., and Kurtulus, D. F., 2012, "Synthesis of a torsional spring mechanism with mechanically adjustable stiffness using wrapping cams," Mechanism and Machine Theory, **57**, pp. 27–39.

[27] Liu, Y., Yu, D. P., and Yao, J., 2016, "Design of an adjustable cam based constant force mechanism," Mechanism and Machine Theory, **103**, pp. 85–97.

[28] Van Der Hoeven, T., 2015, "Statically balanced singular-friction locking," Tech. Rep. 1519786, Delft University of Technology.

[29] Koser, K., 2009, "A cam mechanism for gravity-balancing," Mechanics Research Communications, **36**(4), pp. 523–530.

[30] Ulrich, N. and Kumar, V., 1991, "Passive mechanical gravity compensation for robot manipulators.pdf," *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, Sacramento, California.

[31] Yang, Z. W. and Lan, C. C., 2015, "An adjustable gravity-balancing mechanism using planar extension and compression springs," Mechanism and Machine Theory, **92**, pp. 314–329.

Table 1: *Overview of FCM literature which is evaluated on the 4 performance metrics. The data is extracted from the referenced papers. Some ROM valuas are not presented because the ROM could not be calculated.*

Fig. Ref.	Paper Ref.	Author	Year	Class	1. Accuracy [-]	2. ROM [-]	3. Energy Density $[J/m^3]$	4. Volume Efficiency
1	[20]	Lamers	2012	3	99,0%	0,01	206	0,116%
2	[25]	van Osch	2011	3	90,0%		4900	0,144%
3	[24]	Radaelli	2017	3	99,1%		2373	0,281%
4	[12]	Radaelli	2014	1	97,0%	2,00	1460	0,293%
5	[44]	Bijlsma	2017	4	87,0%		3890	0,306%
6	[16]	Prakashah	2016	2	97,4%		1997	0,350%
7	[11]	Radaelli	2016	1	99,0%		2415	0,359%
8	[45]	Berntsen	2014	2	85,0%		2094	0,387%
9	[46]	Jutte	2008	1	85,0%		7486	0,391%
10	[14]	Merriam	2013	3	85,0%	0,82	245	0,478%
11	[47]	Radaelli	2017	1	99,0%	0,98	16	0,563%
12	[48]	Stroo	2014	4	87,0%		8032	0,609%
13	[32]	Dede	2004	3	97,0%	1,05	367	0,900%
14	[31]	Zong-Wei Yang	2015	3	98,0%	0,91	5845	0,105%
15	[15]	Hou	2013	2	88,0%		8407	1,210%
16	[13]	Radaelli	2011	3	93,0%	1,33	611	1,231%
17	[17]	Wang	2014	2	95,0%		12197	1,260%
18	[19]	Yi Ho Chen	2012	2	95,0%	0,08	1974	1,391%
19	[49]	Merriam	2015	2	95,0%		3281	1,912%
20	[23]	Claus	2008	4	96,7%		7679	2,042%

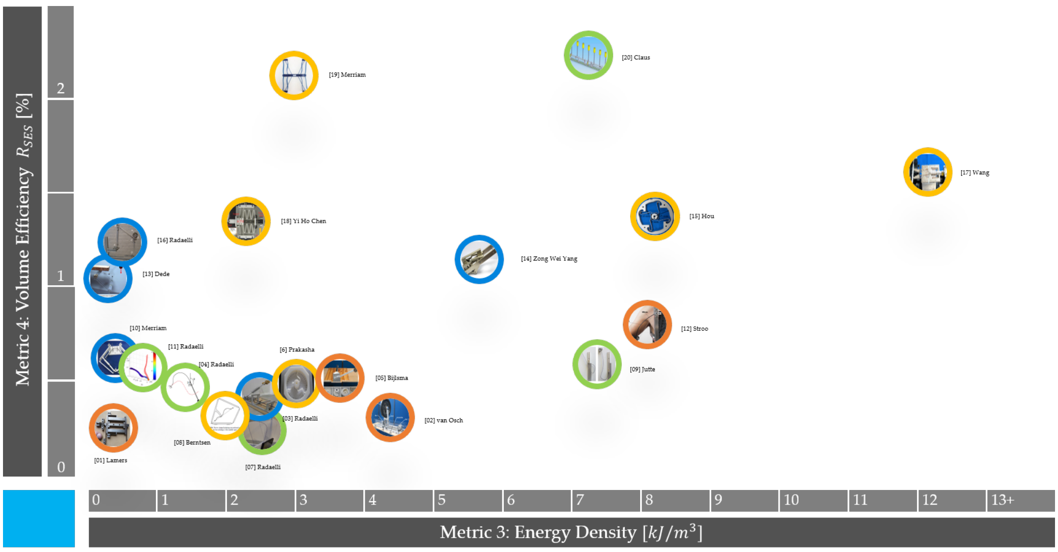


Figure 12: *Plot showing the energy density of literature prototypes from table 1on the horizontal axis and volume efficiency metric R_{VE} on the vertical axis. The classes defined in the previous section are shown in the four colors. Green is class 1, yellow is class 2, blue is class 3, orange is class 4.*

[32] Trease, B. and Dede, E., 2004, "Statically-Balanced Compliant Four-Bar Mechanism for Gravity Compensation," Tech. rep., `http://www-personal.umich.edu/btrease/share/ASME2004/statically-balanced-4bar.pdf`

[33] Arakelian, V. and Ghazaryan, S., 2008, "Improvement of balancing accuracy of robotic systems: Application to leg orthosis for rehabilitation devices," Mechanism and Machine Theory, **43**(5), pp. 565–575.

[34] Rahman, T., Ramanathan, R., Seliktar, R., and Harwin, W., 1995, "A Simple Technique to Passively Gravity-Balance Articulated Mechanisms," Journal of Mechanical Design, **117**(4), p. 655.

[35] Barents, R., Schenk, M., van Dorsser, W. D., Wisse, B. M., and Herder, J. L., 2009, "Spring-to-Spring Balancing as Energy-Free Adjustment Method in Gravity Equilibrators," Volume 7: 33rd Mechanisms and Robotics Conference, Parts A and B, **133**(June 2011), pp. 689–700.

[36] van Dorsser, W. D., Barents, R., Wisse, B. M., and Herder, J. L., 2007, "Gravity-Balanced Arm Support With Energy-Free Adjustment," Journal of Medical Devices, **1**(2), p. 151.

[37] Chu, Y.-L. and Kuo, C.-H., 2017, "A Single-Degree-of-Freedom Self-Regulated Gravity Balancer for Adjustable Payload ¹," Journal of Mechanisms and Robotics, **9**(2), p. 021006.

[38] Briot, S. and Arakelian, V., 2015, "A New Energy-free Gravity-compensation Adaptive System for Balancing of 4-DOF Robot Manipulators with Variable Payloads," Proceedings of the 14th IFToMM World Congress, pp. 179–187.

[39] Wisse, B. M., Van Dorsser, W. D., Barents, R., and Herder, J. L., 2007, "Energy-free adjustment of gravity equilibrators using the virtual spring concept," 2007 IEEE 10th International Conference on Rehabilitation Robotics, ICORR'07, **00**(c), pp. 742–750, arXiv:1011.1669v3.

[40] Cool, J., 1987, *Werktuigkundige systemen*, 3rd ed., Delftse Uitgevers Maatschappij.

[41] Barents, R., 2006, "The space cabinet," Ph.D. thesis.

[42] Esteveny, L., Barbe, L., and Bayle, B., 2014, "A novel actuation technology for safe physical human-robot interactions," Proceedings - IEEE International Conference on Robotics and Automation, pp. 5032–5037.

[43] Krishnan, G., Kim, C., and Kota, S., 2012, "A Metric to Evaluate and Synthesize Distributed Compliant Mechanisms," Journal of Mechanical Design, **135**(1), p. 011004.

[44] Bijlsma, B. G., Radaelli, G., and Herder, J. L., 2017, "Design of a Compact Gravity Equilibrator With an Unlimited Range of Motion," Journal of Mechanisms and Robotics, **9**(6), p. 061003.

[45] Berntsen, L., Gosenshuis, D., and Herder, J., 2014, "Design Of A Compliant Monolithic Internally Statically Balanced Four-Bar Mechanism," *Proceedings of the ASME 2014 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2014*, Buffalo.

[46] Jutte, C. V. and Kota, S., 2008, "Design of Nonlinear Springs for Prescribed Load-Displacement Functions," Journal of Mechanical Design, **130**, arXiv:1011.1669v3.

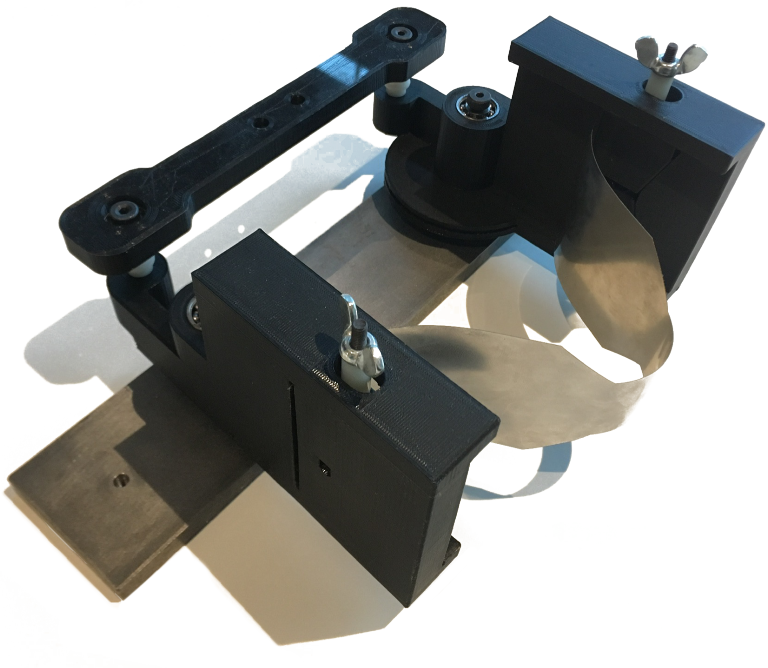
[47] Radaelli, G. and Herder, J. L., 2017, "Gravity balanced compliant shell mechanisms," International Journal of Solids and Structures, **118-119**, pp. 1339–1351.

[48] Stroo, J., 2014, "Feasibility Study of a Balanced Upper Arm Orthosis based on Bending Beams," Tech. rep., Delft University of Technology.

[49] Merriam, E. G. and Howell, L. L., 2015, "Non-dimensional approach for static balancing of rotational flexures," Mechanism and Machine Theory, **84**, pp. 90–98.

3

Paper - An intuitive method to design load-displacement characteristics for nonlinear springs in parallelogram linkages



An intuitive method to design load-displacement characteristics for nonlinear springs in parallelogram linkages

Roel van Ekeren*

Delft University of Technology
Department of Precision and Microsystems Engineering, Mekelweg 2, 2628 CD, Delft, Netherlands
Hittech Multin BV, Laan van Ypenburg 60, 2497 GB, The Hague, Netherlands

September 21, 2019

Abstract

Many mechanical applications involve the use of springs with specifically designed load-displacement characteristics. This paper presents a new mechanical concept, that uses prestressed nonlinear plate springs that can be designed for various load-displacement characteristics for the end effector of parallelogram linkages. An intuitive method is proposed to design the global geometry of the nonlinear plate springs within the given set of boundary conditions from the parallelogram. The results from this method enhance understanding in the design of nonlinear springs and can be used as initial condition for structural shape optimization methods. Three distinct spring characteristics are found using this method to show its applicability. The springs are modelled by a finite element model and validated with a protoype.

Nomenclature

,	
α	Clamp angle of PPS
Ω	Normalized root mean squared error between two curves
ϕ	Orientation angle of PPS
σ	Stress
θ	Angular displacement of parallelogram
ζ	Prestress ratio
EE	End Effector
FE	Finite element
g	Gravity constant
GB	Gravity balancing metric
h	Height of parallelogram
K	Stiffness
L	Initial length of PPS
M	Moment
PPS	Prestressed plate spring
$R1$	Full domain of parallelogram
$R2$	Cropped domain from θ_1 to θ_2
SE	Strain Energy denity metric
U	Work done by PPS
w	Width of PPS

1. Introduction

Parallelogram linkages, a special subset of four-bar mechanisms, are widely used in planar (2-DOF) industrial manipulators. The field of application varies over a large range of scale from micrometers to meters[1] [2] [3] [4] [5]. In various applications a specified force or moment is required along the trajectory of the parallelogram end effectors, for example to statically balance elastic forces in compliant mechanisms or to balance an external load. Statically balanced mechanisms benefit from energy-free force control, inherent safety and improved information transmission [6]. Springs can accommodate the desired load-displacement characteristic for these mechanisms, which is often nonlinear. Typical nonlinear systems in literature are negative stiffness mechanisms [7], constant force mechanisms [8] and bi-stable mechanisms [9], but design of these systems is complicated, since there is no comprehensive method for all nonlinear situations. [10]. Although conventional helical springs are limited to linear responses, techniques are known that use the mechanism geometry or additional transmissions to generate nonlinear load-displacement characteristics for the end-effector. A famous example is the spring-and-lever balancing mechanism from Carwardine [11][12], which is specifically useful to statically balance parallelogram mechanisms, but can also be used for standard rotating pendulums [13]. Furthermore, studies are known where prestressed torsion bars are used to approximate load-displacement functions [14]. Another study uses a special nonlinear gearbox transmission for unlimited range of motion [15]. An energy method is also presented in literature, where linear springs are used as compliant joints to design gravity balancers. [16]

As an alternative to the use of linear springs, which have a fixed spring constant, nonlinear springs can be designed to a prescribed nonlinear load-displacement function. [17]. This makes nonlinear transmissions redundant, which is a clear advantage. However, the possibility to configure many geometric parameters and boundary conditions makes the design process of nonlinear springs challenging. Although, by selecting parameters carefully, shape optimizations can be done to reach desired load-displacement responses. Several successful compliant designs are presented in literature, having constant torque-displacement functions [18], [19]. Also constant-force linear motion mechanisms are presented, but are not directly useful for the application of parallelograms or pendulums [20] [21] [22]. More challenging problems are various gravity balancers, designed by optimizing the initial curvature [23] [24], because such systems can also be employed in parallelograms. Moreover, the use of prestress makes it possible to generate negative stiffness mechanisms. [25] In this study, prestressed nonlinear plate springs are employed in a parallelogram linkage such that the spring is deflected by rotation on both outer ends due to displacement of the paralleogram links, also bringing the possibility to generate negative stiffness and bistable responses. This employment is not earlier seen in literature. In another study by [26], a monolithic, internally statically balanced four-bar was designed, where prestressed nonlinear springs compensate the elastic force of the compliant hinges. Here, the springs were mounted to the reference. Other examples of completely internally statically balanced linkages are optimized without additional prestressed springs [27]. In present designs from literature it is not directly visible how the design can be modified to obtain different load-displacement responses. The optimization procedure should be re-evaluated for a the new load-displacement response. Having a method to design a proper intuitive initial guess of the shape that is to be optimized, assists the optimization procedure and offers understanding in the design of nonlinear springs. The design presented in this paper offers the opportunity to modify the spring shape intuitively to various different load-displacement response. Although the presented method is specifically found for the problem of parallelograms, it is another step in understanding nonlinear spring design.

The goal of this study is to present a novel mechanical conceptual design that uses prestressed nonlinear plate springs. Furthermore, an intuitive method is presented to design the shape of nonlinear plate springs for various load-displacement functions for the end effector of parallelograms.. Results from this approach can be used as proper initial guesses for structural shape optimization methods.

The outline of this paper is as follows: first a detailed problem description will be presented in section 2, followed by additional performance metrics which will be used to evaluate the results. In section 2.3, the spring mechanism concept is proposed. Then, a design method is proposed in section 2.4, and by using a finite element program the plate spring is modelled. Section 2.4.4 presents the constructed and tested prototype to verify the model. In section 3.2, the simulation and measurement results are presented. At last, results will be discussed in section 4, and conclusions given in section 5.

*Graduate student (roelvekeren@outlook.com)

2. Methods

This section formulates the technical research problem and explains the spring mechanism concept. Subsequently a method is presented how springs can be designed for a desired load-displacement objective.

2.1. Problem description

The parallelogram linkage considered in figure 1 consists of four rigid links (1-4). The links are assumed to have infinite stiffness and are connected with standard pin in hole hinges. The most left link is connected to the reference. The linkage system is allowed to rotate by an angle θ in the domain R_2: $[(\frac{\pi}{2} - 0.5)\ (\frac{\pi}{2} + 0.5)]$ rad. Theoretically the parallelogram can move to any angle θ in the domain R_1: $[0\ \pi]$ rad. (dotted line) or even $[0\ 2\pi]$ rad. Since we consider a parallelogram, the opposite links have equal lengths and remain parallel. Also the opposite angles remain equal. For an ideal situation, the friction forces are assumed to be negligible.

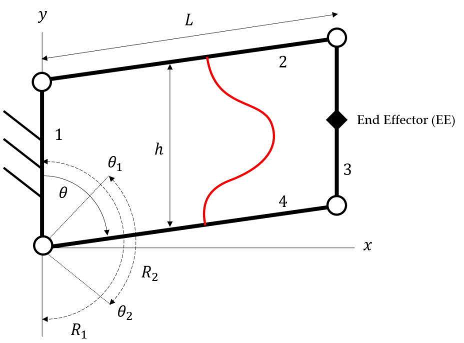


Figure 1: *The parallelogram considered with arm lengths L and h. A moment-displacement characteristic for the end effector is required for the imposed displacement* $[\theta_1\ \theta_2] = [(\frac{\pi}{2} - 0.5)\ (\frac{\pi}{2} + 0.5)]$ *rad. Gravity is considered to act perpendicular to the parallelogram, so this force can be neglected. The links are assumed to have no mass.*

We are interested in a specified moment-displacement function at the end effector, generated by the parallelogram mechanism . Therefore, a prestressed plate spring, hereafter referred to as PPS, is positioned inside the linkage. By displacing the outer ends, the PPS exerts a moment on the parallelogram link 2, link 4 or both. The amount of work done by the resulting moment acting on the parallelogram along the range of motion can be expressed by the energy U:

$$U = \int_{\theta_1}^{\theta_2} M d\theta \tag{1}$$

The work is delivered by the PPS. Examples of three systems with distinct load-displacement characteristics are illustrated in figure 2. The three systems are selected because they are different by their stiffness characteristic (K). Characteristic A is typically used for the application of gravity balancing, which is a difficult nonlinear problem due to a significant negative stiffness range. A gravity balancing spring can be realized by letting the load-displacement curve such that the gravitational force of the mass suspended at the end effector of the parallelogram is balanced by the spring force. It follows that the moment-displacement must be a sine of the angle θ. Characteristic B is the trivial problem, approximating linear springs and characteristic C is typically seen for the application of static balancing elastic forces in compliant mechanisms [26]. The characteristics are normalized, meaning their amplitude is divided by their maximum. In this study, the range of motion and the load-displacement characteristic are considered more important than the load amplitude.

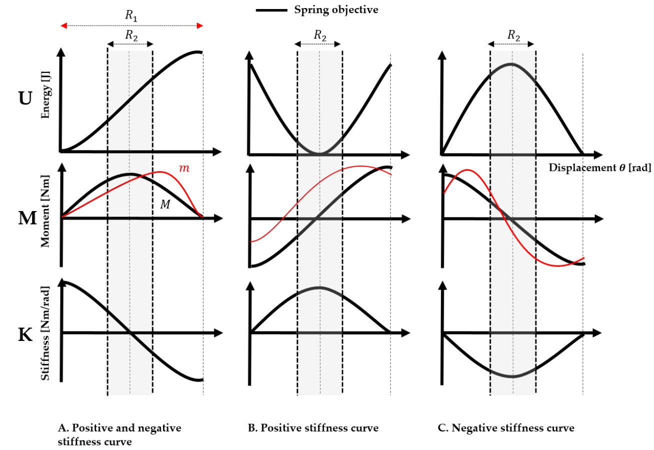


Figure 2: *From left to right three selected energy-displacement curves and below their corresponding moment-displacement* ($\frac{dU}{d\theta}$) *and stiffness-displacement curves.* $\frac{d^2U}{d\theta^2}$). *The hatched area* (R_2) *is the relevant range of motion, in contrast to the full domain* $R_1 = [0\ \pi]$. *Arbitrary designs are illustrated in red as example. The difference between the spring moment in red)*(m) *and the objective moment* (M) *is to be minimized.*

Focusing on the stiffness curves, system A appears to have both positive and negative stiffness. System B only has positive stiffness and system C, has only negative stiffness behaviour. The presented characteristics are highly nonlinear due to the sine-shape and can not directly be created by linear springs without the help of mechanism transmission or without a significant error. A nonlinear spring is desired that is able to approximate the presented objective curves by varying geometry and selecting the right boundary conditions. The target function Ω to be minimized is the normalized root mean squared error between the moment objective function and the actual spring moment for the selected displacement domain R_2. The objective and actual moment are both divided by their maximum, to make comparison independent from scale. Ω can also be used to calculate the error between two other curves. In theory this is calculated by the integral over the complete interval. In practice, the difference between the moments are evaluated at discrete intervals of rotation and the target function is then described numerically:

$$minimize : \Omega(R_2) = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\left(\frac{m_n}{\max(m_n)} - \frac{M_n}{\max(M_n)}\right)^2} \tag{2}$$

In this equation M_n and m_n represent the objective and the designed spring moment respectively for a specific rotation. N is the number of intervals taken for the entire range of motion. This problem definition is subjected to the following constraints:

- $\sigma_{max} < \sigma_{yield}$
- $w_{min} < w < w_{max}$

where w is the stiffness parameter, the width of the PPS. This parameter defines the stiffness by changing the geometry of the spring, which further explained in section 2.3.2.

The selected objective moment-displacement functions are defined by:

$$M_n = sin(\theta_n - \gamma) \tag{3}$$

for the entire interval n and where γ is 0, $\pi/2$ and π for the spring A, spring B and spring C respectively.

2.2. Additional Performance metrics

Two additional performance indicators are used to evaluate the spring designs. The first is used to check the mechanical performance of energy stored into the spring. The second metric is used to evaluate the performance of a spring designed for the gravity balancing objective, which will be discussed in section 4.3.

2.2.1 Beam Energy density

The mechanical efficiency of storing energy into the beam is indicated by the energy density metric as defined by Krishnan et al. [28].

$$\eta_{SE} = \frac{EU}{\sigma_{max}^2 V} \tag{4}$$

Where E is the Young's modulus of the material, U is the work that is put into the system. σ_{max} is the maximum stress and V is the material volume of the beam. The presented metric is the simplified expression of the ratio between the average strain energy density experienced by the entire volume to the local maximum strain energy density. The metric can be interpreted by the fraction of how much material volume is actually used for energy storage.

2.2.2 Gravity Compensation Metric

This metric is used to evaluate the performance of spring designed for the gravity balancing case. The gravity compensation metric (GCM) indicates how much energy that is stored by the springs is effectively used for gravity balancing the payload. Because the springs can be lifted, a part of the spring energy gets lost to lift the weight of the springs. The energy balance is in such cases:

$$U_{stored} = U_{springmass} + U_{payload} + U_{system} \tag{5}$$

where U_{stored} is the energy stored in the spring. $U_{springmass}$ is the potential energy due to the weight of the spring. $U_{payload}$ is the potential energy due to the weight of the payload and the U_{system} represents the potential energy due to the weight of the linkages. We assume from here the linkages to be weightless.

The following definition is used to determine the efficiency of the spring for gravity compensation, which should be smaller than 1 to compensate additional payload:

$$\eta_{gc} = \frac{U_{springmass}}{U_{stored}} = \frac{2\rho V g a}{U_{stored}} = \frac{2\rho g a E}{\eta_{SE}\sigma^2} \tag{6}$$

Here a is the distance from the spring's center of mass to the moment rotation point, which is assumed to be halfway the moment arm of the payload as average. ρ is the density of the spring's material and g is the gravity constant. Furthermore the energy stored in the spring can be expressed in terms of beams energy density metric defined from section 2.2.1. The metric can be interpreted as the percentage of the spring storage that is lost to its own weight. Ideally, this number should be zero. The performance is improved when the COM is displaced towards the rotation point.

2.3. Spring mechanism concept

Prestressed plate springs having specified stiffness over their length can be implemented into a parallelogram linkage to obtain various load-displacement characteristics for the end effector. Stiffness of the PPS can be varied by changing the width parameter along its length. Buckling behaviour of the PPS is forced by axial prestress. The mechanism concept uses this buckling ability to generate negative stiffness and multi-stable behaviour for a large displacement range. First the topology and its boundary conditions are explained. Then, a description is given for the geometry of the PPS. A summary of the design variables for the mechanism concept is presented in table 1.

2.3.1 Topology and boundary conditions

As a first step one unique single spring is investigated. The preloaded spring, illustrated in figure 3, is attached to the parallelogram links with clamp angle α and orientation angle ϕ respectively. Because angle ϕ is kept zero, deformations are only imposed by rotation, which makes the configuration novel, since previously seen concepts that use linear helical springs, were based on change of distance between the clamping points. [11], [6], [29].

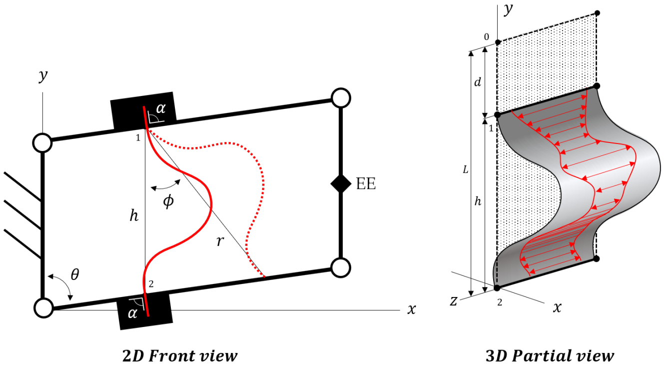


Figure 3: *Schematic 2D representation (left) of the nonlinear spring attachment within the parallelogram linkage. The attachment points (1 and 2) are aligned so angle ϕ is zero. The angle α is 90°. The dotted line spring configuration, located by $\phi \neq 0$, is not used. The 3D partial view shows the spring before prestressing (hatched) by length L, the prestressed spring having constant width (grey) and having an arbitrarily varying width pattern (red).*

In this paper however, the absolute distance d is not changed over rotation because $\phi = 0$. Loading of the spring therefore only occurs as a result of a changing clamping orientation. Axial prestress is imposed by displacement of the clamping points before the PPS is attached to the parallelogram. The distance of prestress is described by prestress ratio ζ, expressed as:

$$\zeta = \frac{d}{L} = \frac{L-h}{L} \tag{7}$$

where d is the absolute prestress distance (from point 0 to 1) and L is the initial length of the spring (0 to 2), as indicated in figure 3. In this paper, the distance h is kept constant. So a change in ζ is done by a change in initial length L.

Attaching the outer ends of the beam to the links can be done in three ways, as displayed in figure 4: Hinged-hinged, Hinged-Fixed or Fixed-Fixed.

Hinged-Hinged - Since we keep the angle ϕ at zero, no change in distance will occur between clamping points 1 and 2. Also no moments can be exerted on the links so this configuration will store no potential energy. A comparable case when ϕ would be nonzero is elaborated by Stroo et al. [30].

Hinged-Fixed - In this configuration the reaction moment is exerted only on one attachment point since the other is free to rotate.

Fixed-Fixed - When both outer ends are clamped to the links, both ends will exert a moment on the links. Depending on the geometry and prestress conditions the resulting moment can be defined.

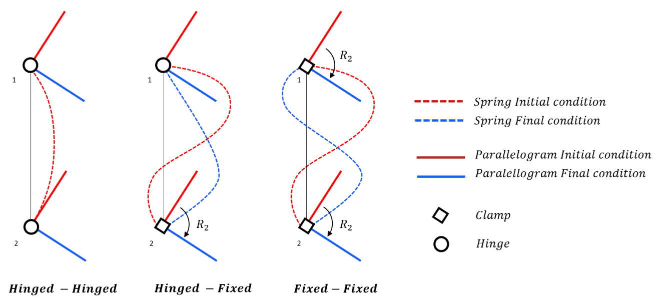


Figure 4: *From left to right three different attachment configurations of the spring within a parallelogram. Hinged-Hinged configuration will apply no moment. Hinged-Fixed results in one applied moment and Fixed-Fixed results in two applied moments.*

For the second and third case, bi-stable behaviour can occur if sufficient displacement is imposed. This behaviour can be used for special load-displacement objectives but is for now not investigated to simplify calculations. Therefore, displacements will be limited to one stable domain during modelling and testing. The fixed-hinged could do the job and could probably store more energy, but is not selected, because it is not practical to create an additional hinge for the spring. The fixed-fixed configuration is therefore selected.

By rotation of the parallelogram links (θ), the outer ends of the plate spring will be rotated as well, resulting in reaction forces and moments on the links. Elastic energy will be stored into the beam. The sum of the reaction moments counteracts the imposed forces on the parallelogram. The reaction forces on the outer ends are cancelled throughout the geometry of the parallelogram as displayed in figure 5.

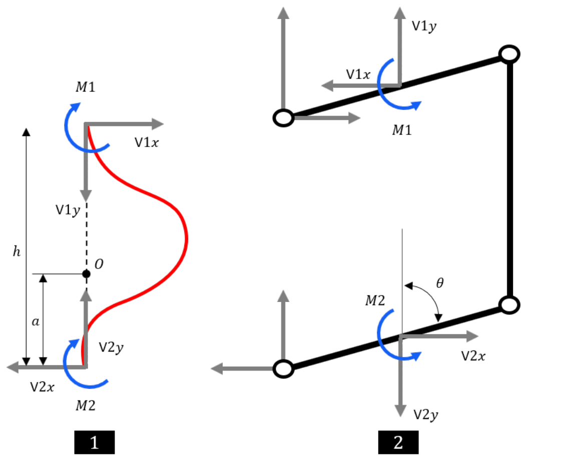


Figure 5: *Reaction forces on the spring are equal and opposite (FBD 1 on the left) Reaction forces of spring cancel out within parallelogram (FBD 2 on the right). Reaction moments are summed within parallelogram if $M_1 \neq M_2$ and contribute to a resulting moment.*

Focusing on the spring itself, static equilibrium equations show that only reaction moments will contribute to compensation of system moments . Consider the free body diagram of the spring from figure 5 on the left. Summing the forces in x-direction gives:

$$\sum F_x = 0 \rightarrow V_{1x} + V_{2x} = 0 \tag{8}$$

Summing the forces in y-direction gives:

$$\sum F_y = 0 \rightarrow V_{1y} + V_{2y} = 0 \tag{9}$$

Summing the moments around an arbitrarily chosen point 0 gives:

$$\sum M_0 = 0 \rightarrow M_1 + M_2 + V_{1x} \cdot (h - a) + V_{2x} \cdot a = 0 \tag{10}$$

We can conclude that the forces V_{1x} and V_{2x}, should be equal and opposite sign. The same holds for the forces V_{1y} and V_{2y}. The moments will be counteracted by the forces in x-direction (V_1x and V_2x) to maintain equilibrium. The resulting moment, the sum of M_1 and M_2 which are not necessarily the same, can be prescribed by changing the geometry or material of the spring.

2.3.2 Geometry and material of spring model

For a single non-spatial spring, its stiffness can be arbitrarily influenced by varying one or more of the following variables over the springs total length s: the initial curvature κ, the Young's modulus E, or the second moment of inertia I, which depends on the width w and thickness t. For spatial problems the in-plane curvature and thickness could also contribute to the beams stiffness.

In this research, the stiffness is chosen to vary over the length of the spring by changing its width. The width is a practical parameter for production purposes. Thickness and initial curvature variations are difficult to produce with low tolerances, while a variation in width could be done, if necessary, with high precision laser cutters. The selected width variation makes this a semi-spatial problem, because the spring can still be modelled in two dimensions, having a single parameter varied over one dimension. The variation in width makes the spring three dimensional, but can be modelled as a stiffness parameter. This makes solving less expensive than a normal spatial problem. The assumption is made that the variation in width will not result in 3-dimensional stresses.

The goal is to find the specified width-shape over the beam length L that will produce a desired moment output, the moment-objective. This problem is constrained by the fact that the spring should have a minimum and maximum width. Also the spring is not allowed not exceed its yield strength σ_y. An example of the variables used for the spring having a varying width is illustrated in figure 6.

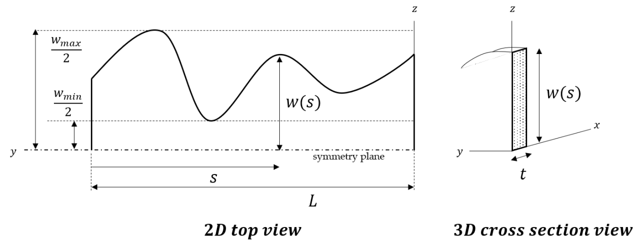


Figure 6: *Top view of half the spring with arbitrary width ($w(s)$) over its length. L is the total non-prestressed length of the spring. The maximum and minimum width is constrained by w_{min} and w_{max}.*

The Bernoulli-Euler equation for the bending moment at any point in the spring is used for modelling the geometry and is expressed by:

$$M = EI\kappa \tag{11}$$

where E is the Young's modulus and curvature can be expressed as the first derivative of the local beam angle:

$$\kappa = ds/d\theta \tag{12}$$

The second moment of inertia for a infinitesimal part of the rectangular cross-section is expressed by the beams width w, and thickness t:

$$I = \frac{wt^3}{12} \tag{13}$$

For practical reasons the material selected for this research is RVS 1.4310. This is a common spring material for industrial applications with a high ultimate tensile strength (1500-1700 MPa) and does not suffer from creep under normal temperatures.

In summary the following variables can be distinguished for the design of the presented balancing concept using the nonlinear spring. The conceptual constraints could be varied for other designs.

Table 1: *List of design variables for the proposed spring mechanism.*

Category	Description	Parameters	This paper
Topology	Nr of nonunique springs	N	1
	Nr of unique springs	n	1
Geometry	Initial Curvature	κ_i	0
	Width	$w(s)$	$w_{min} < w < w_{max}$
	Thickness	t	> 0
	Initial Length	L	> 0
Material			RVS 1.4310
Boundary conditions	Attachment to parallelogram	A	fixed-fixed
	Clamp angle	α	90°
	Prestress ratio	ζ	> 0
	Imposed Rotation	R_I	$R_max > R_I > 0$
	Imposed Translation	T_I	$0\ (\phi = 0)$

2.4. Design Method

The present section will explain how the PPS can be designed for the desired load-displacement characteristic. The PPS in this paper deals with large deflections, so standard equations relating small deflection directly to the spring curvature do not hold. Furthermore, the width of the PPS is not uniform so even more complex analytical models dealing with large deflections can not be used directly [31] [32] [33]. Therefore, a finite element program is used to solve the imposed displacement and boundary conditions of the large-displacement PPS. First, a short overview of the design process will be discussed. Then, the finite element (FE) model set-up will be described. By studying the constant-width PPS, guidelines are given to parametrize the width shape of a non-constant-width PPS to obtain a desired energy objective. The three characteristics from the problem description in section 2 were devised and evaluated by the FE model. Finally, a prototype will be presented that is used to test the three springs and to validate the FE model.

2.4.1 Procedure for beam design

The procedure used to obtain the desired beam geometry for a specified load-displacement objective can be generally described as follows:

1. Select main geometry, boundary conditions and topology;
2. Select geometry parameter for the shape to be optimized;
3. Set initial conditions of the geometry parameter;
4. Calculate load-displacement function for the geometry parameter input;
5. Calculate error of evaluated load-displacement function to the objective function;
6. Configure geometry inputs and iterate untill objective is satisfied;

The first and second step are already discussed in the previous section. The following sections will describe how to find a proper initial guess (step 3) for the geometry parameter to reduce optimization costs. This paper does not discuss the particular optimization routine and calculates only the initial step using the FE program.

2.4.2 Finite element model

The finite element package, ANSYS APDL,is used to calculate and predict the complex PPS shape functions, displacements and resulting forces. A 2-dimensional 188 Bernoulli beam element type is used for this model. In order to model width variations over the length of the complete PPS, multiple 188 beam elements were connected. Each element's width is specified by a shape-vector scalar **S**(i) as shown in equation 14. The FE model is then constructed as displayed in figure 7. The thickness of the PPS is kept constant. No initial curvature is applied.

$$\mathbf{S} = [s(1)s(2)..s(n-1)s(n)] \tag{14}$$

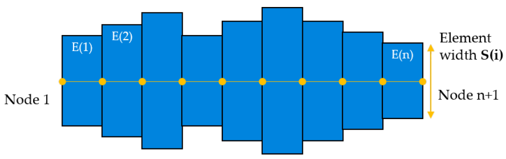


Figure 7: *Top view of the FE model, constructed by multiple 188 Bernoulli beam elements. Every element's width is specified by its corresponding shape-vector item. The displayed shape is arbitrarily chosen for illustrative purpose.*

The FE model is run using 100 elements and a minimum of 50 load steps to make sure the model converged. The material parameters used for the FE model are: density $\rho = 7800\ kg/m^3$, Young's Modulus $E = 200\ GPa$, Poisson ratio $\nu = 0.29$ and $\sigma_{yield} = 1100\ MPa$. To simulate prestress and rotations within the parallelogram, boundary conditions from figure 4a are used:

1. a small perturbation load, a moment on both ends having the same sign, is applied to ensure buckling into the second mode shape;
2. prestress is applied for a selected prestress ratio ζ;
3. the perturbation load is removed;
4. the outer ends are displaced by the same specified rotation $R = R_{left} = R_{right}$;

A MATLAB script defines geometric parameters, runs an ANSYS batch file, and the outputs generated by ANSYS are then loaded and processed in MATLAB again.

2.4.3 Creating shape function

For a specific energy objective function, a corresponding width shape has to be found. The main procedure in finding this width-shape is explained here. By investigating a constant width beam imposed by the described boundary conditions from section 2.3, strain energy waves appear during deflection. The waves occur at points of maximum curvature. Points of zero curvature are called inflection points. The beam sections where peaks in curvature are found, can be used to add or remove material with the goal to reach a desired energy characteristic. From there, an optimization procedure could be initiated to reduce the overall error between the design and objective. First, a single FE model run is done for a spring having uniform width. From this run, an energy diagram and its curvatures were extracted. Figure 8 illustrates the normalized curvature (by its maximum) for the undeformed (red) and deformed (blue) situation. The spring is displaced on both ends by 1.6 radians. The deflection shapes for the initial and final conditions are shown on top, where incremental sub steps are displayed in grey. Strain energy peaks occur at maximum deformation locations in the spring. At deformation points where curvature reaches a local maximum the derivative of the curvature is zero. During deformation the inflection points will move. Since two inflection points appear in this configuration, three energy peaks can be found in total. However, at the initial and final conditions only two peaks are visible.

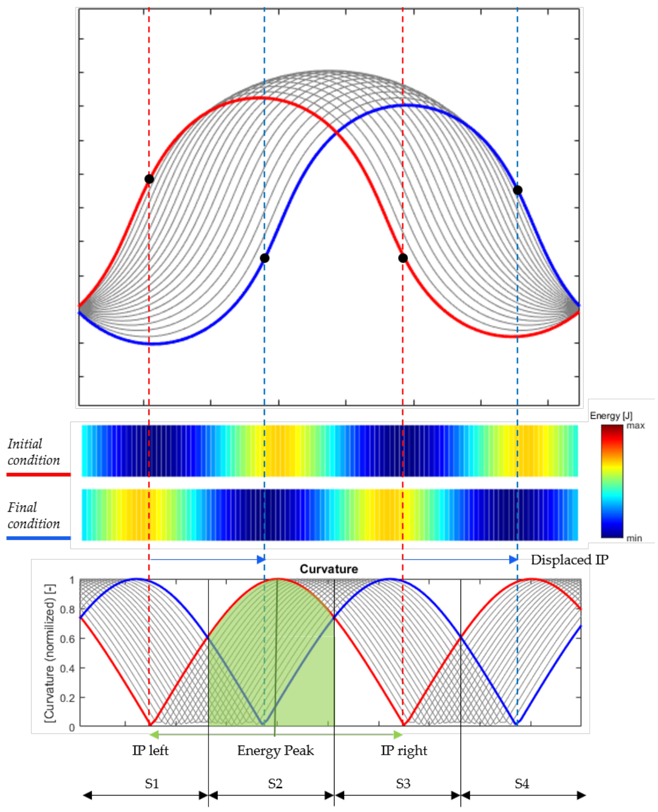


Figure 8: *On top the shape function for initial (red) and final (blue) deformation of the constant width spring. The black lines represent the sub steps. Below the absolute curvature is plotted, normalized by its maximum. In the middle the energy waves for the initial and final condition. Upon deformation strain energy peaks will displace from left to right, caused by the curvature maxima. The inflection points are indicated by the black dots corresponding to zero curvature and zero energy. Energy sections (S1-S4) can be distinguished between the inflection points where curvatures from the initial and final deformation conditions intersect.*

Using the behaviour of the moving energy section, we can create other cross-sections with different width patterns resulting in a changed load-displacement characteristic. For springs with uniform width, the total energy remains constant under current boundary conditions. Springs having a different cross-section, however, yield a slightly similar energy diagram, approaching roughly the uniform width beam. Therefore a width pattern can be selected on basis of the illustrated energy diagram from figure 10. Adding more material to a certain section of the spring will increase its total strain energy if that section is bended. The section bounds will vary when geometry and boundary conditions are changed, meaning that peak stresses can be located at different positions. A simple parametrisation for the width of the spring is devised as illustrated in figure 9. The boundaries where width of the beam varies, is described by a vector **q**. The scalar values of **q** represent the normalized distance in %.

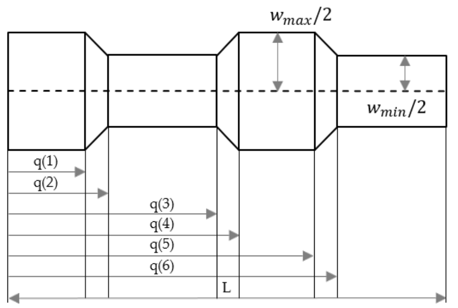


Figure 9: *An arbitrary shape illustrates the spring geometry parameters that is used for the simplified model. Three building blocks were used: wide rectangular block (w_{max}), a narrow rectangular block (w_{min}) and a tapered block to connect the wide with the narrow blocks. The length of the blocks are parametrized by **q** as a fraction of the total length L.*

Since we are interested in the three objectives (A, B and C) from figure 2, three width shapes were constructed semi-arbitrarily, meaning that intuition and a few iterations were done to approach the energy objectives. Their geometry specifications can be found in table 2. The model parameters are presented in table 3.

Table 2: *Spring geometry specifications for the modelled and tested springs.*

Spring Type	Bounds
Spring A_1	$\mathbf{q}$ = [22.7; 31.3; 45.5; 54.1; 68.1; 76.1]
Spring A_2	$\mathbf{q}$ = [15.0; 25.0; 45.0; 55.0; 75.0; 85.0]
Spring B	$\mathbf{q}$ = [31.8; 40.0; 59.0; 67.7]
Spring C	$\mathbf{q}$ = [31.8; 40.0; 59.0; 67.7]

Table 3: *Parameters used for the model and experiment. The prestress ratio and imposed rotation are the only parameters which are varied for the other simulations.*

Category	Description	Parameters	Model input	Unit
Geometry	Width	$w(s)$	$30 < w < 60$	mm
	Thickness	t	0.2	mm
	Initial Length	L	220	mm
Material	Young's Modulus	E	200	GPa
	Poisson ratio	ν	0.29	[−]
	Density	ρ	7800	kg/m^3
	Yield strength	σ_{yield}	1100	MPA
Boundary conditions	Clamp angle	α	90	°
	Prestress ratio	ζ	31.8	%
	Imposed Rotation	R_I	[-1 1]	rad
	Imposed Translation	T_I	0	m

The first objective is only increasing, so width is increased in section S1 and S3. The second objective is initially decreasing and halfway increasing. Therefore width is decreased in the middle, halfway in section B until halfway between in section S3. The third objective is exactly the opposite. Therefore, the width was increased halfway section B and decreased halfway section S3, resulting in the desired energy characteristic that is initially increasing and from halfway decreasing. The final models are presented in figure 11. The simulated spring shapes are most likely not unique solutions

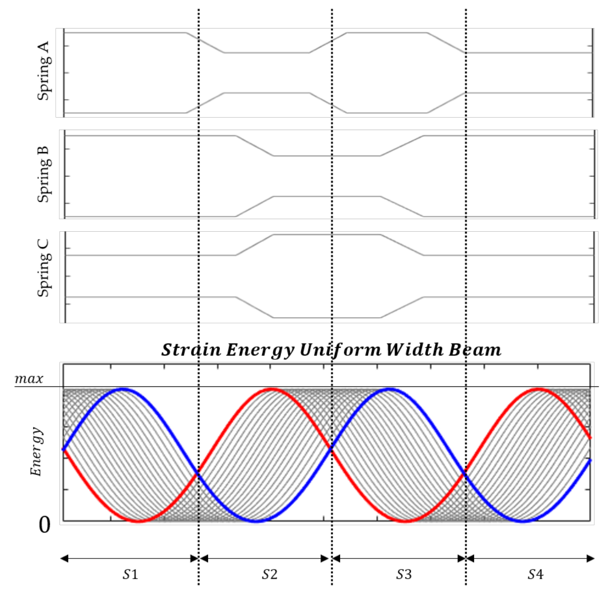


Figure 10: *On top three spring designs are presented parametrized by **q**. Below the strain energy per element is plotted for a uniform width beam for the initial (red) and final (blue) conditions. Sub-steps are indicated in grey. The boundaries are selected where strain energies are equal for both initial and final deflection. The relevant sections appear between the selected boundaries(A-D) for which material is added or removed as guideline to influence the load-displacement objective.*

for the energy characteristics. Other shapes can possibly be found with similar characteristics. However, the method shows that proper intuitive initial guesses can be made for shape optimization of springs for at least three different load-displacement functions. The three spring designs will be simulated for three different prestress ratios: ζ = 20, 40 and 60 % to investigate to what level the objective functions can be approximated.

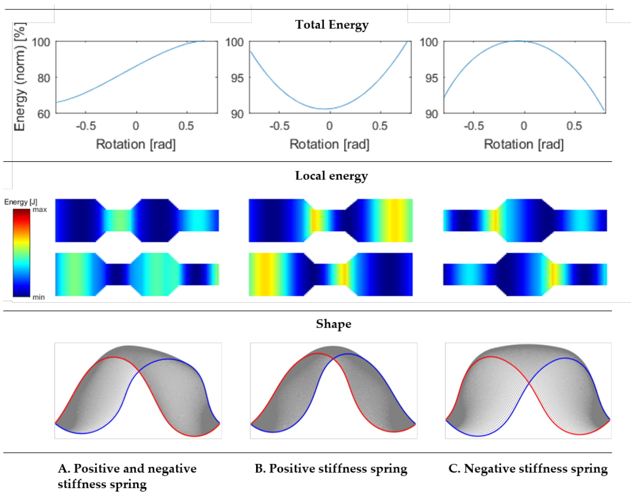


Figure 11: *Three different springs were simulated corresponding to the three objectives (A, B and C) from the problem description. On top the total energy (normalized) is calculated for the entire displacement. Below the initial and end energy distribution in the beam is shown. On the bottom the spring's shape functions are shown.*

2.4.4 Prototype and experiment setup

A prototype was designed, constructed and tested to validate the ANSYS model. The model consists of two arms and a connector, made from 3D printed PLA. Deep groove ball bearings were used to keep friction low at the four hinges.

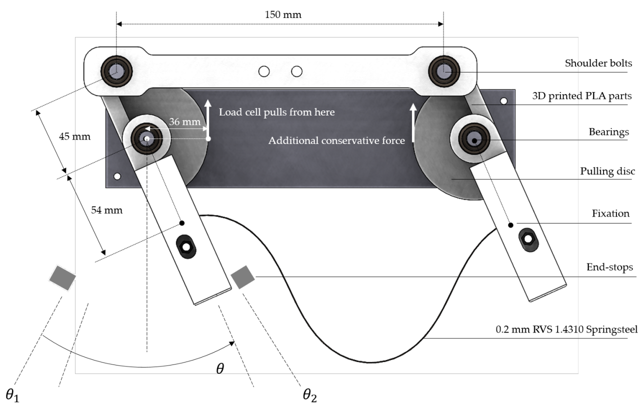


Figure 12: *Snapshot of the parallelogram test set-up prototype. The prototype is made of PLA 3D-printed parts. SKF 306 bearings were used for reduction of friction. Shoulder bolts were used for the pin connections in the hinges. A clamp mechanism was designed to keep the springs in position during rotation. The springs are tested for the range of θ_1 to θ_2. End-stops make sure the range is not violated.*

The prototype is tested on a load-displacement stage, a PI stage (M-505.4DG S/N 107054253). A linear DC motor can displace the load cell in tiny steps of 10 μm with a total of 10.000 steps. The experiment is done in 1200 steps, i.e. displacement intervals of 8.3 μ. The left arm's pulling disc is connected by wire to the loadcell (FUTEK 549178 10lbs). The right arm's pulling disc is connected by wire to additional measured mass of 0.322 kg. The mass was provided to avoid measurements around zero. A clamp mechanism holds the prestressed spring in position. By pulling the wire a moment is exerted on the parallelogram arm resulting in rotation around the hinges. During a single measurement, the force was measured by the load cell over the range of motion backward and forward. The turning point was marked with an endstop. For spring A, C and the set-up this endstop was placed at 1.65 radians. For spring B this endstop was placed at 1.55 radians.

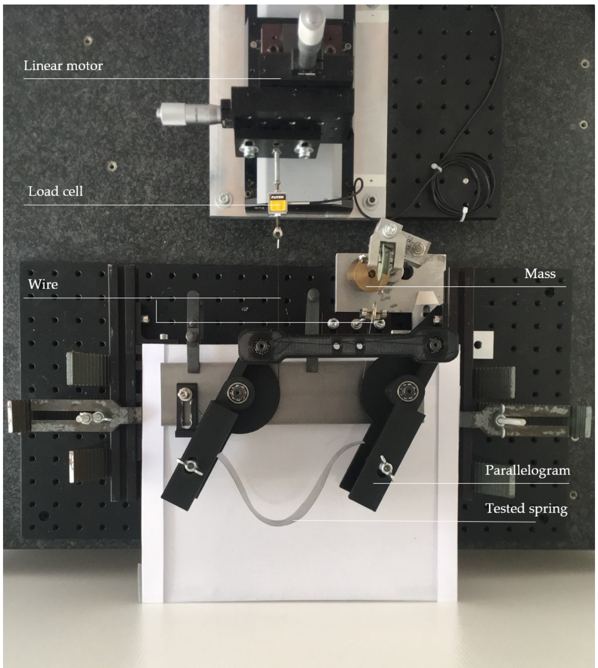


Figure 13: *Top view photograph of the test set-up. The parallelogram is clamped to the ground. The linear motor pulls the left arm. Between the motor and the left arm, a load cell measures the force. A conservative force, gravity acting on a weight, pulls the right arm. The three different springs (A,B and C) are tested on this set-up.*

3. Results

3.1. Simulations

Simulations were done for all spring types A,B and C, to investigate how well the objectives can be reached by the presented method. Three different prestress ratios ζ were simulated (20%, 40% and 60%) for the three spring types and plotted in figure 14. Based on information from the three simulations, a fourth or fifth simulation is done with slightly modified parameters to obtain an improved result. Note that the simulations and objectives are normalized by their maximum and as a consequence the maxima and minima of simulations 5-12 are located at the initial and final point of the simulated range of motion respectively. Except for simulation 10 where the FE model simulated a part of the post-buckling behaviour.

For each simulation the normalized root mean squared error (Ω) between the simulated spring and the objective was calculated twice. The first error,Ω_1 was calculated for the complete range of motion of the objective, π [rad]. The second, Ω_2 was calculated for the narrowed interval of $[-0.5, 0.5]$ $[rad]$. All results are presented in table 4. The best result for spring A is simulation 4 and shows Ω_2 = 2.29%. The best result for spring B is simulation 9 and shows Ω_2 = 3.04% and the best result for spring C is simulation 11 shows Ω_2 = 2.21%.

For all simulations the strain energy density metric (SE) was calculated, as well the absolute amount of energy stored in Joule. However, only for the simulations of spring A, the gravity balancing metric (GB) is calculated since spring A is the only result designed for gravity balancing.

Focussing on spring type A, simulations 1-4, the intervals differ because only positive moments are simulated. The intervals of spring type B are simulated for almost the entire domain (3 rad.) except for simulation 6 which converged only for a slightly smaller interval (2.84 rad.). Simulations 8 and 9 are presented to show that a better approximation reached for the smaller intervals of 2 radians. The last three simulations (10, 11 and 12) are simulated with intervals smaller than 2 radians because larger intervals did not converge.

For spring type A the simulation $\zeta = 60\%$ is repeated with a slightly modification for the spring shape parameter **q** to reach a smaller error Ω. For spring type B the simulations of ζ = 60% and ζ = 20% were repeated for the smaller interval of 2 radians because an increase in performance was expected. The results show that the smaller interval produce smaller errors and for simulation 9 a higher strain energy density. Spring type C showed best results for the prestress ratio of ζ = 40%.

3.2. Measurements

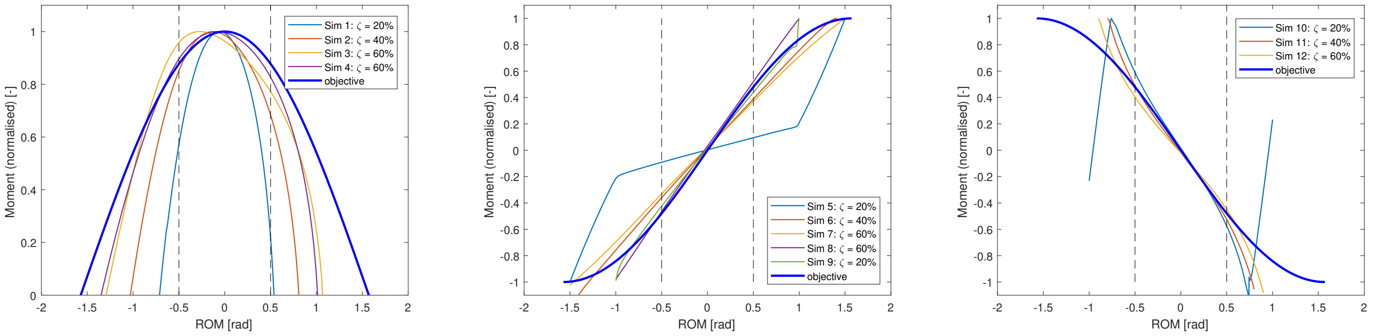
The goal of testing the prototype springs is to validate the FE model. Twelve measurements were done on the previously described test set-up. Three measurements were done for each spring (a total of 9) and three measurements were done with the set-up only, measuring the friction and additional weight to the setup. Results of the measurements are presented in figure 15 and 16. The hysteresis loop is clearly visible. Since every spring is tested three times, three blue loops are slightly visible in the results. The mean of measurements, taken from the three measurement sets per spring, is shown in green. To identify the measured error, the set-up mass is subtracted for all spring measurements (A, B and C) such that comparison is improved. The moments simulated by the FE model are plotted in red. The error between the measurements and the simulated model is calculated and shown in black. Also the normalized root mean squared error (Ω) is calculated between the measurements and the simulated FE model. Finally, to show errors in the set-up, the mass was subtracted from the set-up measurement and shown in blue in figure 16b.

4. Discussion

The presented method seems fast and effective to approximate shapes that can be used as initial guesses for structural shape optimization. The method uses the behaviour of a uniform width PPS to predict strain energy in non-constant width PPS. This implies that

Table 4: *Results calculated by the ANSYS model based on spring A_1 and A_2, BandC. The normalized root mean squared error (Ω), Range of motion (ROM), strain energy density ratio (SE), Gravity balancing metric(GB) and the energy stored in the spring (ES). Higher prestress ratios result in smaller errors larger ROM's for spring A. Ω_1 is the error calculated for the entire ROM. Ω_2 is calculated for a narrowed ROM = [-0.5 0.5] rad. For spring B and C a specific range of motion is to be selected to decrease the error.*

Simulation	Type	ζ [%]	Ω_1 [%]	Ω_2 [%]	Simulated ROM [*rad*]	Energy Stored [*J*]	Metric SE [%]	Metric GB [%]
1	A_1	20	31.94	21.31	1.25	0.103	9.07	23.33
2	A_1	40	22.27	7.36	1.84	0.171	6.87	25.07
3	A_1	60	12.89	6.59	2.36	0.191	4.60	50.35
4	A_2	60	11.63	2.29	2.36	0.179	4.48	54.18
5	B	20	39.79	22.91	3	0.278	19.03	
6	B	40	7.75	6.88	2.84	0.122	4.08	
7	B	60	10.73	8.37	3	0.094	1.87	
8	B	60	4.83	3.04	2	0.094	1.88	
9	B	20	3.04	3.04	2	0.067	4.89	
10	C	20	22.51	4.25	2	0.046	4.72	
11	C	40	7.39	2.21	1.6	0.187	9.68	
12	C	60	6.04	5.03	1.8	0.216	7.46	



(a) *Spring type A* **(b)** *Spring type B* **(c)** *Spring type C*

Figure 14: *Three spring types simulated in comparison with the objective function. The graphs show the normalized moment along the range of motion of rotation in rad. The spring types are simulated for three different prestress ratios $\zeta = 20\%$, 40% and 60%. The narrowed interval for a range of motion [0.5 0.5] rad is indicated with the black dotted lines.*

the method is not perfect and can result in incorrect predictions for more complex load-displacement functions. Therefore, the method is particularly useful for predicting the rough shape of the PPS as approximate to the load-displacement function. The design method has a limited applicability, because it uses the boundary conditions of a parallelogram. Although these boundary conditions appear in other situations as well, there are boundary conditions for which the method cannot be used. Nevertheless, the presented mechanical concept shows that at least three different load-displacement functions can be realised, where two of three exploit the buckling behaviour to generate negative stiffness for a significant range of motion. Creating negative stiffness is a challenging objective for the design of spring mechanisms. Therefore, letting the load displacement characteristic follow a specifically designed load-displacement function, of which a large part has a negative stiffness, is a valuable finding.

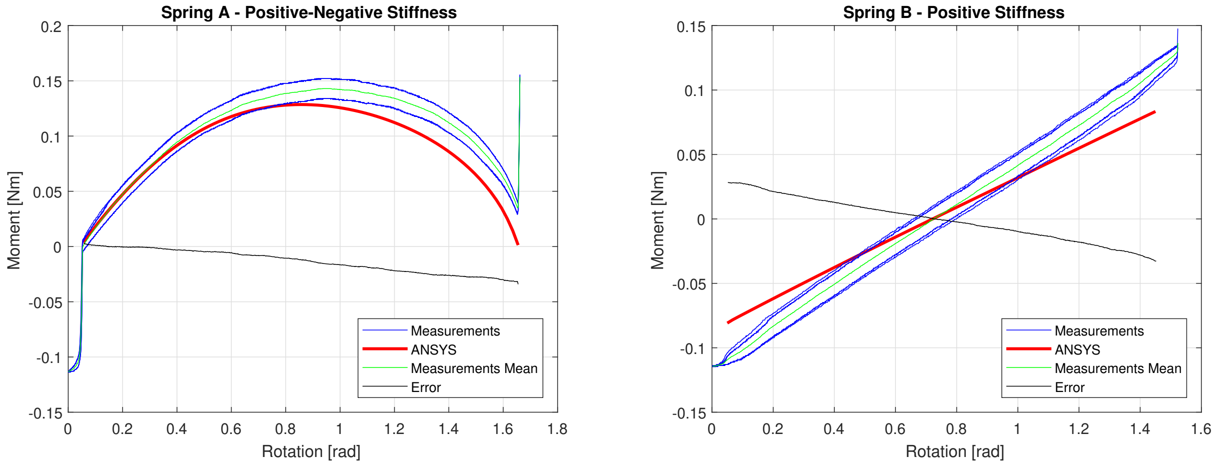
4.1. Simulations

The simulations from figure 14 show good approximations of the objective within the selected interval of $[-0.5, 0.5]$ rad. With the error of Ω = 2.29%, simulation 4 showed the best result. Outside the interval the simulations diverge rapidly from their objective. Focussing on the first four simulations, the increase of a larger prestress distance (higher prestress ratio ζ) leads to better approximations of the objective, because the range of motion is increased. However, by increasing prestress ratio ζ (and so the total length of the PPS) the characteristic does not remain the same. Slight modifications in spring shape parameter $\mathbf{q}$ compensate this, leading to a smaller error Ω in simulation 4. Although Ω is decreased, the gravity balancing metric (GB) increases, meaning that a larger part of the springs energy is lost to it's own mass. For higher prestress ratios the SE shows a lower efficiency. This could be explained by the fact that a higher prestress ratio (and therefore a larger initial volume) results in larger differences between the stress peaks and zones with lower stresses. Also, the zones with lower stresses are relatively larger. Therefore, a lower fraction of the PPS is maximally used for storing energy.

Simulation 5 stands out from the other simulations with spring type B, because a clear discrete stiffness transition is visible. The spring, having a prestress ratio of only $\zeta = 20\%$ shows a stable equilibrium halfway the displacement. The required moment is dramatically increased, because further deflection is constrained by the length of the spring. A fraction of the spring is from this point also axially strained. Therefore, a much higher SE metric can be observed for simulation 5. This effect is also slightly visible in simulation 9, having the same prestress ratio as simulation 5, but a smaller range of motion. In fact, this simulation is a cropped version of simulation 5 and since the moment is normalized, it results in a better approximation of the objective. For the other simulations this effect appears to exist as well, but since the prestress is different, the region of smaller stiffness is increased. An optimization could be performed using the found rough shape as initial condition to find the exact shape that will fit the load-displacement objective.

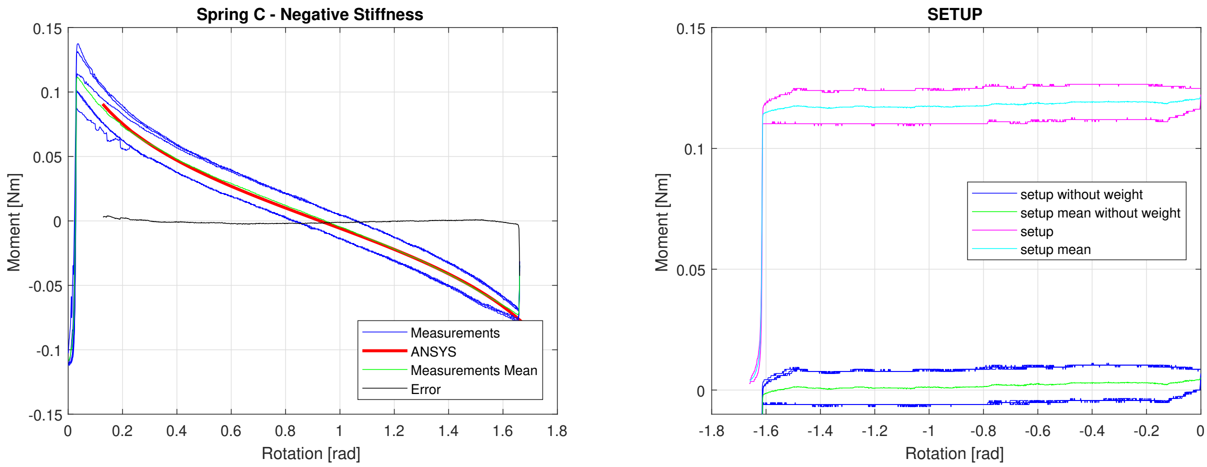
4.2. Measurements

The measurements satisfy the expected and modelled results. The characteristics are qualitatively the same as the modelled springs in ANSYS, although the error between the ANSYS model and the measurements is significant and not the same for every spring element. The possibility exists that errors are caused by elasticity or inaccuracies in the printed PLA, although the springs were lasercutted from the same sheet of spring steel. The thickness tolerance of



(a) Ω = *0.078** **(b)** Ω = *0.107*

Figure 15: *Measurement and model data for spring A having both positive and negative stiffness and for spring B having positive stiffness. Ω is calculated where M_n is the measurement and m_n is the model.*



(a) Ω = *0.021* **(b)**

Figure 16: *Measurement and model data for spring C having negative stiffness in (a) and measurement data of the individual setup to evaluate the hysteresis in (b). Ω is calculated where M_n is the measurement and m_n is the model.*

the sheet according to supplier Jeveka is 3%. The E-modulus is not measured but is assumed to maximally vary 5% within the sheet. The difference in width between the model and the cut sheet is 0.2 mm, giving a maximum width error of 0.6% on both sides. Adding the tolerances would lead to significant maximum error of 8.6%. A realistic measure is to take the root of the sum of the squared tolerances which results in an error of 5.6%. However, for some measurements the error is larger than 10%, so the difference is unlikely to be explained by the inaccuracies of production only. Another possible source for the model differences is the method of clamping the spring to the parallelogram. Errors could occur by clamping the spring not perfectly perpendicular to the plane of the parallelogram, causing complex out of plane bending. After inspection the axle of one parallelogram arm was indeed not perfectly perpendicular. The deviation of the angle was only 1 degree, but this could already have large consequences on the load-displacements. Both the inaccuracies in the parallelogram arm, and possibly a small bearing misalignment could cause the friction variation that is found in the set-up measurements.

4.3. Gravity balancing case

If tuned properly, the load-displacement characteristic obtained by spring A (15a), can be used for gravity balancing. As a first result a spring is designed by simulation 4 for a ROM of $[-0.5, 0.5]$ rad having $\Omega = 2.48\%$. The performance is calculated by the metric GB shown in table 4. For the spring from simulation 4, 54% of the energy that is stored is lost to balance its own weight, assuming that the spring is located in the middle of the parallelogram. Moving the spring toward the hinge results in smaller moment exerted by the spring itself, so less energy is lost to its own weight. For smaller prestress ratios the performance is better: a smaller percentage of the spring energy is lost to balancing its own weight. However, the errors are significantly higher for smaller prestress ratios, so spring balancers with 20% - 40% prestress ratios would not be feasible. The strain energy metric shows that only 2-5 % of the material is used to store energy. For normal helical springs this number is 50%, which is almost ten times higher [34]. The energy capacity of the presented spring system could potentially be increased by using multiple springs in parallel, stacked together, if the deflections are not excessive. Besides that, also the overall width parameter can be increased, or the entire system could be scaled to reach the desired amount of energy.

5. Conclusion

A novel mechanical concept is presented where prestressed nonlinear springs are used to synthesise distinct load-displacement characteristics for parallelogram linkages. An easy-to-use intuitive method is described for the design of nonlinear springs constrained by the boundary conditions of the parallelogram. The width parameter can be manipulated to specify the stiffness of the spring at any location along the length to approximate at least three important load-displacement characteristics. The presented implementation of nonlinear springs with the used boundary conditions is novel and not found earlier in literature. Three different springs were designed, prototyped and compared to their objective function. Based on this intuitive guess the model showed an NMSE with respect to the objective functions of 2.29%, 3.04% and 2.21% respectively, for the selected interval of [-0.5 0.5] rad. Both model and measurements show load-displacement characteristics as expected. The model is validated and shows a good match with the measurements: Ω = 0.083;0.149 and 0.087 for the three springs respectively. The result of one modelled spring load-displacement characteristic can potentially be used for gravity balancing.

References

[1] Wilcox, D. L. and Howell, L. L., 2005, "Fully Compliant Tensural Bistable Micromechanisms (FTBM)," Journal of microelectromechanical systems, **14**(6), pp. 1223–1235.

[2] Hao, G. and Li, H., 2015, "Nonlinear Analytical Modeling and Characteristic Analysis of a Class of Compound Multibeam Parallelogram Mechanisms," Journal of Mechanisms and Robotics, **7**(4), p. 041016.

[3] Arakelian, V. and Ghazaryan, S., 2008, "Improvement of balancing accuracy of robotic systems: Application to leg orthosis for rehabilitation devices," Mechanism and Machine Theory, **43**(5), pp. 565–575.

[4] van Dam, T., Lambert, P., and Herder, J. L., 2011, "Static Balancing of Translational Parallel Mechanisms," *35th Mechanisms and Robotics Conference, Parts A and B* (Vol. 6), pp. 883–889, doi:10.1115/DETC2011-47525, `http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?articleid=1641019`

[5] Briot, S. and Arakelian, V., 2015, "A New Energy-free Gravity-compensation Adaptive System for Balancing of 4-DOF Robot Manipulators with Variable Payloads," Proceedings of the 14th IFToMM World Congress, pp. 179–187.

[6] Herder, J. L., 2001, "Energy-free systems: theory, conception, and design of statically balanced spring mechanisms," Ph.D. thesis, Delft University of Technology, doi:10.13140/RG.2.1.3942.8966, `http://repository.tudelft.nl/view/ir/uuid:8c4240fb-0315-462a-8b3b-efbd0f0e68b6/`

[7] Hoetmer, K., Woo, G., Kim, C., and Herder, J., 2010, "Negative Stiffness Building Blocks for Statically Balanced Compliant Mechanisms: Design and Testing," Journal of Mechanisms and Robotics, **2**(4), p. 041007.

[8] Tolman, K. A., Merriam, E. G., and Howell, L. L., 2016, "Compliant constant-force linear-motion mechanism," Mechanism and Machine Theory, **106**, pp. 68–79.

[9] Cleary, J. and Su, H.-J., 2015, "Modeling and Experimental Validation of Actuating a Bistable Buckled Beam Via Moment Input," Journal of Applied Mechanics, **82**(5), p. 051005.

[10] Radaelli, G., 2017, "Synthesis of mechanisms with prescribed elastic load-displacement characteristics," Ph.D. thesis, Delft University of Technology, doi:10.4233/uuid.

[11] French, M. J. and Widden, M. B., 2000, "The spring-and-lever balancing mechanism, George Carwardine and the Anglepoise lamp," Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, **214**(3), pp. 501–508.

[12] Carwardine, G., 1935, "Improvements in Equipoising Mechanism," .

[13] Rahman, T., Ramanathan, R., Seliktar, R., and Harwin, W., 1995, "A Simple Technique to Passively Gravity-Balance Articulated Mechanisms," Journal of Mechanical Design, **117**(4), p. 655.

[14] Radaelli, G., Buskermolen, R., Barents, R., and Herder, J. L., 2017, "Static balancing of an inverted pendulum with prestressed torsion bars," Mechanism and Machine Theory, **108**(July 2016), pp. 14–26.

[15] Bijlsma, B. G., Radaelli, G., and Herder, J. L., 2017, "Design of a Compact Gravity Equilibrator With an Unlimited Range of Motion," Journal of Mechanisms and Robotics, **9**(6), p. 061003.

[16] Radaelli, G. and Intespring, B. V., 2011, "An energy approach to the design of single degree of freedom gravity balancers with compliant joints," *Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2011*, ASME, Washington DC.

[17] Jutte, C. V. and Kota, S., 2008, "Design of Nonlinear Springs for Prescribed Load-Displacement Functions," Journal of Mechanical Design, **130**, arXiv:1011.1669v3.

[18] Nair Prakashah, H. and Zhou, H., 2016, "Synthesis of Constant Torque Compliant Mechanisms," Journal of Mechanisms and Robotics, **8**(6), p. 064503.

[19] Hou, C. W. and Lan, C. C., 2013, "Functional joint mechanisms with constant-torque outputs," Mechanism and Machine Theory, **62**, pp. 166–181.

[20] Wang, J.-Y. and Lan, C.-C., 2014, "A Constant-Force Compliant Gripper for Handling Objects of Various Sizes," Journal of Mechanical Design, **136**(7), p. 071008.

[21] Rahman, M. U. and Zhou, H., 2014, "Design of Constant Force Compliant Mechanisms," Internatial Journal of Engineering Research & Technology, **3**(7), pp. 14–19.

[22] Chen, Y.-H. and Lan, C.-C., 2012, "An Adjustable Constant-Force Mechanism for Adaptive End-Effector Operations," Journal of Mechanical Design, **134**(3), p. 031005.

[23] Radaelli, G. and Herder, J. L., 2014, "Isogeometric Shape Optimization for Compliant Mechanisms With Prescribed Load Paths," *Proceedings of the ASME 2014 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, doi:10.1115/DETC2014-35373, `http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?doi=10.1115/DETC2014-35373`

[24] Radaelli, G. and Herder, J. L., 2016, "A monolithic compliant large-range gravity balancer," Mechanism and Machine Theory, **102**, pp. 55–67.

[25] van Eijk, J. and Dijksman, J. F., 1979, "Plate spring mechanism with constant negative stiffness," Mechanism and Machine Theory, **14**(1), pp. 1–9.

[26] Berntsen, L., Gosenshuis, D., and Herder, J., 2014, "Design Of A Compliant Monolithic Internally Statically Balanced Four-Bar Mechanism," *Proceedings of the ASME 2014 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2014*, Buffalo.

[27] Merriam, E. G., Colton, M., Magleby, S., and Howell, L. L., 2013, "The Design of a Fully Compliant Statically Balanced Mechanism," *Proceedings of ASME 2013 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, August, doi:10.1115/DETC2013-13142, `http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?doi=10.1115/DETC2013-13142`

[28] Krishnan, G., Kim, C., and Kota, S., 2012, "A Metric to Evaluate and Synthesize Distributed Compliant Mechanisms," Journal of Mechanical Design, **135**(1), p. 011004.

[29] Cardoso, L. F., TomaÌĄzio, S., and Herder, J. L., 2002, "Conceptual Design of a Passive Arm Orthosis," 27th Biennial Mechanisms and Robotics Conference, **5**(January), pp. 747–756, arXiv:1011.1669v3.

[30] Stroo, J., 2014, "Feasibility Study of a Balanced Upper Arm Orthosis based on Bending Beams," Tech. rep., Delft University of Technology.

[31] Holst, G. L., Teichert, G. H., and Jensen, B. D., 2011, "Modeling and Experiments of Buckling Modes and Deflection of Fixed-Guided Beams in Compliant Mechanisms," Journal of Mechanical Design, **133**(5), p. 051002.

[32] Zhao, J., Jia, J., He, X., and Wang, H., 2008, "Post-buckling and Snap-Through Behavior of Inclined Slender Beams," Journal of Applied Mechanics, **75**(4), p. 041020.

[33] Howell, L. L. S. P. M., 2013, *Handbook of Compliant Mechanisms*, John Wiley & Sons, West Sussex.

[34] Cool, J., 1987, *Werktuigkundige systemen*, 3rd ed., Delftse Uitgevers Maatschappij.

4
DISCUSSION

This section discusses first the literature review from chapter 2. Next, the thesis paper from chapter 3 is discussed. The discussion here is presented more elaborately, but may have overlap with the discussions in the papers.

4.1. LITERATURE REVIEW

The presented classification and metrics in the literature review can be used to identify the accuracy, range of motion and compactness of a spring force mechanism. Two metrics were presented that quantify the performance of the mechanism on error and range of motion. The other metrics quantify a spring mechanism on compactness. By classification of a system in one of the presented classes, it can make designers aware of the possible volume losses in the mechanism. The prototypes from literature however, were designed for the proof of concept, in stead of compactness, so the presented list does not represent the true compactness of these concepts for industrial applications. Nevertheless, the list can serve as inspiration for new opportunities in the design of more compact spring force generators. Volume loss was considered high for the class with multiple springs and transmission. The possibility exists to use space from the spring unit cell for stacking. Moreover, the literature review was a starting point to investigate stacking possibilities of nonlinear plate springs. In contrast to the conventional helical spring, a plate spring stores relatively less energy per unit volume. But, if plate springs are properly stacked, they can provide possibly more energy per unit volume because conventional springs lose also space to their unit volume inside the coil. In this research it appeared that, for the application of gravity balancing and the imposed boundary conditions of the parallelogram, stacking of springs in parallel without contact, becomes very complicated. No feasible option was found for a significant range of motion. Therefore, the focus was set on the design of single unique springs with different load-displacement characteristics for the parallelogram linkage. Nevertheless, stacking is still possible for mechanisms with small displacements. It is left for future research to find out for what conditions stacking of nonlinear plate springs is suitable.

4.2. THESIS PAPER

The discussion of the paper is divided into four sections. The first is a general discussion about the presented method. The second section discusses the parameters and boundary conditions that were used to constrain the problem. Subsequently simulations from the model are discussed in more general terms than in the paper and finally a discussion on the application of gravity balancing is presented.

4.2.1. METHOD FOR LOAD-DISPLACEMENT CHARACTERISTICS

The method explained in the paper can be used to design at least three distinct load-displacement characteristics for nonlinear plate springs that are implemented in a parallelogram. It can also be used for other applications with similar boundary conditions, a few examples of possible applications are presented in appendix A.6.1. The method uses results from an uniform width plate spring to predict load-displacement functions for non-uniform width springs. This implies that the method is not perfect and can result in incorrect predictions. Furthermore, the method only predicts very rough load-displacement characteristics. The three findings are: negative stiffness curve, positive stiffness curve and a curve having both negative and positive stiffness. Other general characteristics that are interesting to reach are: constant force curves or curves that demonstrate more combinations of negative and positive stiffness. A mechanism that exhibits negative stiffness is normally seen as challenging, so the found negative stiffness curves seem to be already a valuable result that can be used for instance to statically balance elastic forces in compliant mechanisms. The results from the paper show that the negative stiffness region of the spring is relative large, and can be increased by enlarging the prestress distance. However, larger prestress distances also result in more volume occupation by the spring. This is more elaborately discussed in the last section 4.2.4. Furthermore, the possibility exists to combine both negative and positive springs by superposition to create a zero-stiffness mechanism, having a constant force output. It is also observed that this mechanism can demonstrate bi-stability. This can be used for a range of different applications, although the focus in this paper was not on the synthesis of bi-stable behaviour.

4.2.2. PARAMETERS AND BOUNDARY CONDITIONS

Many decisions are made for selecting the geometry and boundary conditions to constrain the problem. The width parameter was selected as key parameter to influence the load-displacement characteristic, because of practical advantages in fabrication. Variations in curvature and thickness are relatively more challenging to manufacture with low tolerances. Also the clamp angle α and the orientation angle ϕ (figure 3, paper) can be varied. Furthermore, the hinge-fixed boundary condition could be exploited in order to increase the output moment on the parallelogram. By clamping the outer ends of the spring reaction moments are counteracted, thereby lowering the effective moment exerted on the parallelogram. In the configurations of a hinged-fixed design, the entire internal moment is exerted on the parallelogram resulting in higher forces with respect to the fixed-fixed configuration. However, implementation of a hinge is challenging and can be a source to new problems. If chosen for a hinged configuration, for example a lumped compliant hinge can be used. Opportunities for alternative designs can be found in variations of these boundary conditions. More about this can be found in appendix A.6.1 and A.1.

4.2.3. SIMULATIONS

The simulations showed that for three objectives a sufficient match can be reached. Also other simulations are run for different blocked shapes. Output from these simulations is presented in appendix A.8. For the simulation it was chosen to investigate only blocked shapes to simplify the problem. As a consequence, results from this simplification can be slightly unrealistic for transition zones between narrow and wider widths.

Better predictions can be done if more information is known about the influence of the ratio between the maximum and minimum local width $r_w = w_{min}/w_{max}$ and how the variation in width influences the curvature globally. For now the r_w was set on 0.5 along the entire beam. Also a small ramp was added to avoid extreme stress concentrations at the transition zones. An alternative is to keep a uniform width for the spring and perforate the parts continuously on spots where less material is required. More detail about this idea is explained in in appendix A.6.2.

It is evaluated that for a constant-width spring, the effective moment exerted on the parallelogram is effectively zero, because the reaction moments are opposing (appendix A.8). The reaction moment on an outer end of the spring can be reduced by varying the width along the spring length. Focussing on the strain energy in the spring, three waves can be distinguished. (figure 8, paper). The first wave is initially outside the beam and flows in from the left when the beam is deflected. The second wave, almost halfway, flows from S2 to S3. The third wave on the right outer end (S4) flows outside the beam. The difference between minimum and maximum total energy stored in the spring can be increased by using all waves. Furthermore, the energy wave is distributed over a certain domain of the spring. The shape of this distribution, together with the displacement rate of the wave, determine the global energy-displacement function of the spring. More insight in these complexities can bring new ideas for the design of new load-displacement functions.

4.2.4. GRAVITY BALANCING

The paper showed that a simulation approximated the gravity balancing objective with a normalized root mean squared error of $\Omega = 2.29$ for the range of motion of $[\pi/2 - 0.5; \pi/2 - 0.5]$ rad, and a prestress ratio of $\zeta = 60\%$. This is a relatively low error over a significant range of motion. The prestress ratio is however quite large and as a consequence the spring occupies more volume than expected. Therefore, stacking springs in parallel is compromised by the prestress ratio. The prestress ratio ζ appears to be an important parameter, influencing the range of motion and occupied volume of the mechanism. For example, if a higher prestress ratio is applied, the range of motion is enlarged. However, a larger prestress ratio increases spring occupancy volume as well, which is for some cases not desired. Another consequence is the fast decrease of stack-ability of springs, resulting in lower total energy capacity of the mechanism. As a rough guideline, springs can be stacked next to each other if the prestress ratio is low (< 20%) and the range of motion is sufficiently small (< 0.5 radians).

For the application of gravity balancing it is required that the spring mechanism has sufficient energy capacity to compensate the force exerted by the mass. Furthermore, a minimum prestress ratio is required for the spring to be able to approximate the balancing objective accurately. Since the springs are not stackable for prestress ratios larger than 20%, and a single spring is only able to balance two times its own weight, (GB = 54% in paper, table 4) this mechanism is not very suitable. Another factor that makes the presented spring configuration not suitable for gravity balancing is the maximum allowable stress of the spring. In this research the maximum allowable stress is the yield strength. In practice the value for the yield strength is scaled down by a risk factor. On the other hand, energy capacity could be increased independently from the yield strength by scaling the width, or by scaling the entire spring, thus by scaling the length together with the thickness.

5
Conclusion

The overall goal of the thesis was divided into two parts: the first goal was to provide an overview of the volume occupancy of spring based force compensation mechanisms in literature. The second goals was to investigate the implementation of nonlinear spring in parallelogram linkages for the design of force compensation mechanisms. The following conclusions can be drawn from the research.

5.1. Literature

The literature review shows a classification in four groups of existing force compensation mechanism prototypes from literature. Groups are formed on basic components from the mechanisms: single spring, multiple springs and a transmission or combinations. The classification gives insight on what component level of volume occupancy improvements can be made. Furthermore, four metrics were presented and used to compare literature on accuracy, range of motion, energy density and volume efficiency. For all analysed mechanisms the volume efficiency is below 3% of the total mechanism volume, which is mainly explained by the fact that literature prototypes are not designed for compactness. Nevertheless, the presented overview that shows the compactness of these systems can be a starting point to compare and future designs.

5.2. Paper

The paper presents a mechanical design using prestressed nonlinear plate springs in parallelogram linkages. Boundary conditions of a parallelogram were exploited to impose end rotations on nonlinear springs, which is not earlier seen in literature. The presented method is another step in the understanding of nonlinear springs for designers and future researchers. Three distinct load-displacement characteristics were generated by three spring shapes based on the presented method. The shapes can serve as initial condition for shape optimization methods tot reach smaller errors for the objectives. Moreover, a significant negative stiffness range can be created. For one spring type simulations show that this negative stiffness range was already more than 1 radians, by using a prestress ratio $\zeta = 60\%$. This could be enlarged by increasing the prestress ratio ζ. This ratio is an important parameter that influences applicable range of motion. It also influences the stacking density of spring in parallel. A FE model is used to simulate the springs and is validated using a prototype for three different springs. The presented mechanical concept can be used for the application of gravity balancing but is in this stage not a better alternative with respect to conventional methods. Regarding the energy capacity of the spring a more suitable application is to counteract elastic forces of parallelograms with lumped compliant hinges.

5.3. Appendices

Finally several conclusions regarding the appendices can be drawn:

- The volume occupancy of a maximally stretched conventional helical spring with a spring index of 4 is 30%. Since only 50% of the material is maximally utilized, only 15% of the occupied volume is used for strain energy. This number can serve as an incentive to investigate more efficient methods to store potential energy. This calculation is presented in appendix B.3.

- A list of possible options is presented for adjusting a nonlinear plate spring to new payloads or other load-displacement functions. The list is most likely not complete.
- A list is presented in appendix A.4 that compares possible spring materials on three properties. Based on these properties a suitable material can be selected.
- Several ideas are presented in appendix A.6 showing that the boundary conditions of the parallellogram can also be found in other applications. Also different options are presented for the implementation of a width pattern.
- A GUI is programmed and presented in appendix A.7 to analyse properties of large deflections of non-linear springs.
- An overview is provided in appendix A.5 that shows energy storage efficiency for different types spring shapes and load types.
- Simulations were run for all unique combinations of block-shapes consisting of four blocks, presented in appendix A.8. The simulations shows that with simple building blocks already distinct load-displacement characteristics can be generated. It can serve as a start for a 'building block' library. Furthermore, it can be seen from the simulations that similarities in load-displacement characteristics appear by comparable shapes.
- Another small study in appendix B.4 shows that 216 options are available to create three degrees of freedom for the example of an end effector of a microscope support. This number can be reduced to () feasible options, based on the stated assumptions.
- The study presented in B.2 shows that the spring model with outer end rotations can be converted to a fixed guided beam problem, frequently seen in literature [9]. It also shows analytic equations that are valid to calculate uniform width beams for large displacements.

6

Recommendations

Reflecting on the work done, the following recommendations can be considered. The recommendations are divided into three categories: the first category includes possible improvements on the model. The second category involves recommendations on the prototype and measurements. The final category discusses opportunities for future work related to this thesis. At last a short vision for future development is addressed.

6.1. Improvements on model

- The spring is now modelled with discrete width shapes. As a consequence stresses can accumulate at corners. Moreover, transitions from small to wider widths are modelled as it was a uniform beam, meaning that from one element to the other, the full moment is transferred in the model. In reality, the moment is transferred over the smallest cross-section, resulting in zero stress at the outer corner of the larger element, and additional stresses at the transition zone between the two elements. A smoother variation in width is therefore preferred.
- The model could be extended by an optimization program to find the exact fit to the objective curve. For the optimization it is important to choose the right parameters. The parameters that are now of interest are: the prestress ratio ζ, the building block distances $\mathbf{q}$, and the maximum and minimum widths. However the width can also be defined as a continuous parameter instead of a prescribed minimum and maximum. A spline-based optimization would therefore be more suitable. Spline-based optimizations are for example performed by Radaelli [10].
- Constant parameter in the model, are the clamp angle α and the orientation angle ϕ, could be varied. The parameters influence the spring behaviour and can possibly be used for other load-displacement functions. However, incorporating these parameters into the model is at the expense of computational cost.
- This study is done using a FE model. Analytic solutions to this problem could provide insight to the complexities and open possibilities to new load-displacement characteristics.
- A sensitivity analysis can be performed to find out what parameters have more priority. For example, to find out the influence of an error in the clamp angle α several runs can be compared. The result can be used to find out what influence the calculated error difference from appendix A.11 is on the output. Other important parameters to check are: prestress ration ζ, orientation angle ϕ, thickness t and width w.

6.2. Prototype and measurements

- It is highly recommended to focus on correct alignment in future setups. Improvements on this set-up could be made by ensuring a perfect alignment of the spring with the parallelogram and the ground's surface.
- Clamping the spring with the right angle and distance is challenging and quickly lead to errors in the output. Improvements are possible on the current implementation, since the possibility exists that the

clamp blocks can differ in distance for about a maximum of 1 mm. This is shown in detail in appendix A.9.

- Since pulley disk and wire arrangements can lead to errors, a more accurate approach would be to use a torque-displacement sensor directly on the axle.
- Instead of clamping end-stops to the stage, incorporating end-stops in the mechanism design could increase the accuracy for the initial and final angle.
- In future designs it can be considered to incorporate compliant hinges. However, the elastic forces from these hinges should be accounted for.

6.3. OPPORTUNITIES FOR FUTURE WORK

The following topics can serve as opportunities for future research.

- In the paper from chapter 3 the main focus was on the design of a load-displacement characteristic generated by a single spring. Multiple springs were initially not considered. The implementation of multiple different springs could bring opportunities in the design of more unique load-displacement characteristics in the form of superposition when springs are positioned in parallel in the mechanism.
- Next to the use of multiple different springs, it is also interesting to research how shapes can be stacked in parallel to optimize the space inside the mechanism. Much space is lost due to the curved pre-stressed shape of the spring. If springs could be stacked efficiently, the energy capacity of the mechanism could be increased significantly. Moreover, an overview of the conditions that enable stacking for nonlinear springs could be valuable for designers for example to know which shapes and boundary conditions are suitable for stacking of springs, and which are not.
- The spring is assumed to have both outer ends clamped. It was noted in the paper that a hinged-clamped configuration could be feasible. The hinged-clamped configuration is probably less stable, but can be exploited for bi-stable applications.
- A small study was done to find out what load-displacement functions were obtained for different building blocks. The building blocks were based on a discrete width variation with four blocks as shown in appendix A.8. From this study already arise various load-displacement curves. This study can also be conducted using five or more blocks. However, by increasing the number of blocks, the number of options increase as well. The results from such studies can also form a library to gain insight for the design of load-displacement characteristics.
- A related topic to the design of load-displacement characteristics is the analysis of stability of the spring. The considered spring design was analyzed in its stable region. However, for larger displacements bi-stability occurs. This bi-stable behaviour could be exploited but can also be avoided. In either case it is important to know in which situations and regions the spring is stable or unstable. Having a model that specifies stability properties of the designed beam allows the designer to predict the applicability of the spring.
- The boundary conditions of the parallelogram were used to displace the outer ends of the spring. These imposed rotations can be found in other applications as well, as can be seen in appendix A.6.1. More research can be done on what applications are suitable for which kind of load-displacement functions. The load-displacement characteristics can then be generated by the design method and optimization models.

6.4. VISION

The idea of using nonlinear springs in parallelogram linkages is potentially a solution for optimizing the energy density in a parallelogram. If springs are properly shaped and stacked, the parallelogram can be filled with springs in parallel. The research showed that this is complicated because of the shape of the springs. However, for smaller displacements the method is still feasible. Some ideas are discussed here that can be used for future development:

- The method can be used to statically balance elastic forces of a compliant parallelogram. It can also be used to statically balanced other external forces.
- It can potentially serve as method for compliant four-bars, but since the angular rotations of the end-points will change it is still unknown for what configurations this will hold.
- The balanced parallelogram can be made monolithic. If a monolithic building block can be made, the parallelogram can be extended and scaled, making it a statically balanced meta-material parallelogram.
- Adjustment of the balancing condition is a next step for the nonlinear spring design. However, adjustment could potentially be done by turning springs on or off. This is discussed in appendix A.3.

If a system with the properties from above is proven feasible, it can have the following advantages with respect to the conventional method of a helical spring:

- High redundancy. If multiple springs are used in parallel, the redundancy is increased. For the case that a spring failure, many springs are left to guarantee safety and functionality of the system.
- The system is easily scalable by enlarging the width or by increasing the number of active springs.
- Assembly and maintenance advantages due to monolithic design possiblities. Ways can be found to easily replace failed springs.
- Accuracy can be improved if the entire system is monolithic. No hinges are involved in the connection of springs to the parallelogram.

Bibliography

[1] N. de Wit, *Vibration dissipation in a surgical microscope support system*, Tech. Rep. (Delft University of Technology, 2017).

[2] V. Arakelian and S. Ghazaryan, *Improvement of balancing accuracy of robotic systems: Application to leg orthosis for rehabilitation devices,* Mechanism and Machine Theory **43**, 565 (2008).

[3] J. L. Herder, *Energy Free Systems: Theory, conception and design of statically balanced spring mechanisms*, Ph.D. thesis, Delft University of Technology (2001).

[4] G. Carwardine, *Improvements in Equipoising Mechanism,* (1935).

[5] M. J. French and M. B. Widden, *The spring-and-lever balancing mechanism, George Carwardine and the Anglepoise lamp,* Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science **214**, 501 (2000).

[6] R. Barents, *The space cabinet*, Ph.D. thesis (2006).

[7] R. Barents, M. Schenk, W. D. van Dorsser, B. M. Wisse, and J. L. Herder, *Spring-to-Spring Balancing as Energy-Free Adjustment Method in Gravity Equilibrators,* Volume 7: 33rd Mechanisms and Robotics Conference, Parts A and B **133**, 689 (2009).

[8] G. Radaelli, *TU Delft University*, Ph.D. thesis, Delft University of Technology (2017).

[9] G. L. Holst, G. H. Teichert, and B. D. Jensen, *Modeling and Experiments of Buckling Modes and Deflection of Fixed-Guided Beams in Compliant Mechanisms,* Journal of Mechanical Design **133**, 051002 (2011).

[10] G. Radaelli and J. L. Herder, *Isogeometric Shape Optimization for Compliant Mechanisms With Prescribed Load Paths,* in *Proceedings of the ASME 2014 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference* (2014).

[11] L. Berntsen, D. Gosenshuis, and J. Herder, *Design Of A Compliant Monolithic Internally Statically Balanced Four-Bar Mechanism,* in *Proceedings of the ASME 2014 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2014* (Buffalo, 2014).

[12] J. Cool, *Werktuigkundige systemen*, 3rd ed. (Delftse Uitgevers Maatschappij, 1987).

[13] G. Krishnan, C. Kim, and S. Kota, *A Metric to Evaluate and Synthesize Distributed Compliant Mechanisms,* Journal of Mechanical Design **135**, 011004 (2012).

[14] G. Radaelli, R. Buskermolen, R. Barents, and J. L. Herder, *Static balancing of an inverted pendulum with prestressed torsion bars,* Mechanism and Machine Theory **108**, 14 (2017).

[15] J. L. Herder, N. Vrijlandt, T. Antonides, M. Cloosterman, and P. L. Mastenbroek, *Principle and design of a mobile arm support for people with muscular weakness,* The Journal of Rehabilitation Research and Development **43**, 591 (2006).

[16] W. D. van Dorsser, R. Barents, B. M. Wisse, and J. L. Herder, *Gravity-Balanced Arm Support With Energy-Free Adjustment,* Journal of Medical Devices **1**, 151 (2007).

[17] J. Zhao, J. Jia, X. He, and H. Wang, *Post-buckling and Snap-Through Behavior of Inclined Slender Beams,* Journal of Applied Mechanics **75**, 041020 (2008).

[18] L. L. S. P. M. Howell, *Handbook of Compliant Mechanisms* (John Wiley & Sons, West Sussex, 2013).

A

APPENDIX A

This appendix includes related work to the thesis, which is done during the past year. The work includes individual projects but also supplementary material to the paper that is presented in chapter 3. Programmed code is presented in the next appendices C D.

A.1. PARAMETERS OF SPRING DESIGN

This research focusses on the width parameter of the spring to modifty the springs stiffness. This section shows an overview of the alternatives. Figure A.1 shows an overview of the basic mechanism parameters that can be modified. The overview can be read like an morfological overview where the entire mechanism is formed by the individual components, here denoted by parameters.

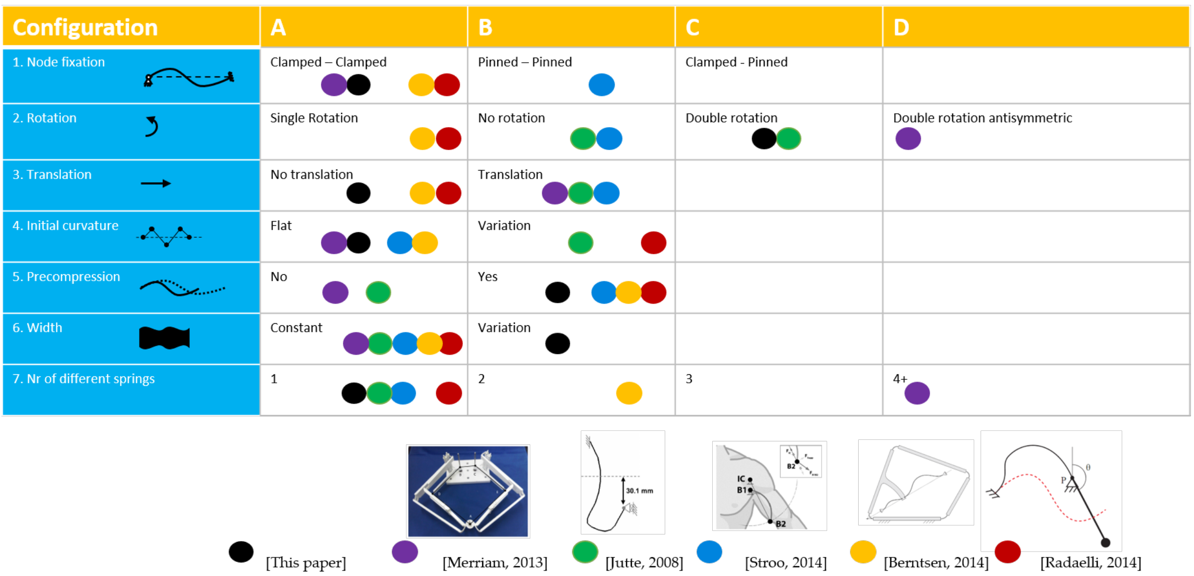


Figure A.1

A.2. STACKING

For the situation of stacking of multiple springs in parallel, sufficient distance is required to avoid contact between adjacent springs. The minimum and maximum distances between the springs are dependent on the shape of spring for each rotational interval. However, before calculating anything, a few things can be stated.

- The stacking distance d, must be chosen with a safety factor. Contact between the springs will lead to stiffness variations.
- Since the spring is prestressed in a s-shape the smallest gaps between the springs will also be in the zones with a low curvature.
- the local angle alpha of the spring determines how much stacking distance is lost to the springs curve. The local angle can be extracted from the model. (elemental z-rotation). If the minimum distance between the springs is required to be k, the minimum stacking distance should be $d_{min} = k \cdot \cos\alpha + s$ where α is the local angle and s is the safety factor distance.
- The maximum local angle of the spring increases by higher prestress ratios as can be seen in figure A.3.
- Next to continuous deflection of the spring itself, the adjacent springs displace relative to the spring with distance: $g = d \cdot \cos\theta$ where θ is the instantatnious angle of the parallelogram, because the springs have different positions with respect to the hinge. This must be taken into account when calculating the minimum distance between two springs, as discussed in the previous bullet-point.

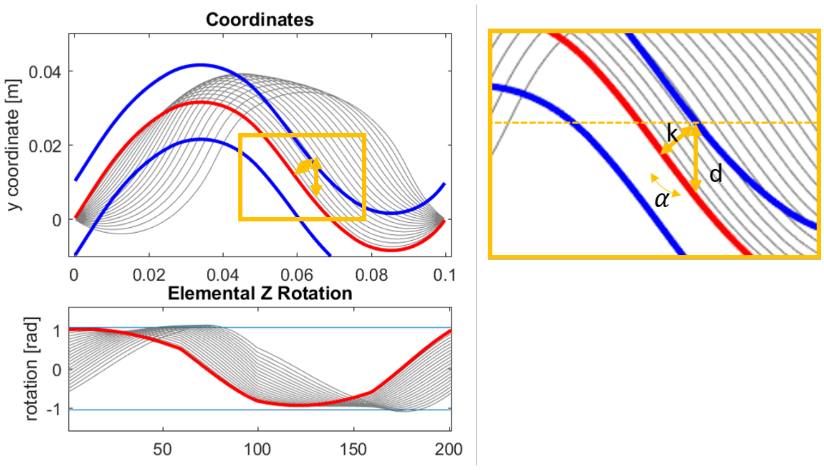


Figure A.2: Three identical springs stacked in parallel. In yellow a close up. Below the local angle per element of one spring.

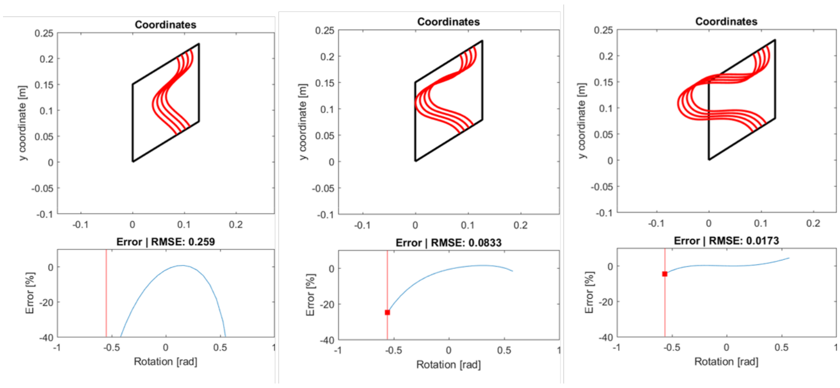


Figure A.3: Three simulations with different prestress ratios (from left to right) $\zeta = 20\%$, $\zeta = 40\%$ and $\zeta = 60\%$ presented for spring type A (see paper). The root mean squared error (not Ω) with respect ot the objective is displayed below. It is observed that for higher prestress ratios stacking is not feasible because contact is made between springs. Depending on the design of the parallelogram the springs can also make contact with its links.

A.3. VARIATION OF PAYLOAD

The following figure A.4 serves as inspiration for future concepts in the design of adjustable balancers using nonlinear plate springs. The list is most likely not complete. First 7 parameters are presented. The 7 parameters can be adjusted to either change the load-displacement curve or to change the amplitude to adjust the balancer to a new payload. It is most likely both load-displacement curve and amplitude change when manipulating one of the parameters. However for some parameters this is not the case, for example the width. Changing the width uniformly will increase the amplitude only.

For parameter 2, 3, 4 and 5 it is hard to think of a solution that can change the parameter without remaking the component. Another trivial solution is to add another subsystem to the mechanism which can be adjusted.

The figures illustrate multiple springs that are stacked in parallel. It is however known from this thesis that stacking is challenging when dealing with large displacement (> 0.5 rad) or large presstress distances (> 10%).

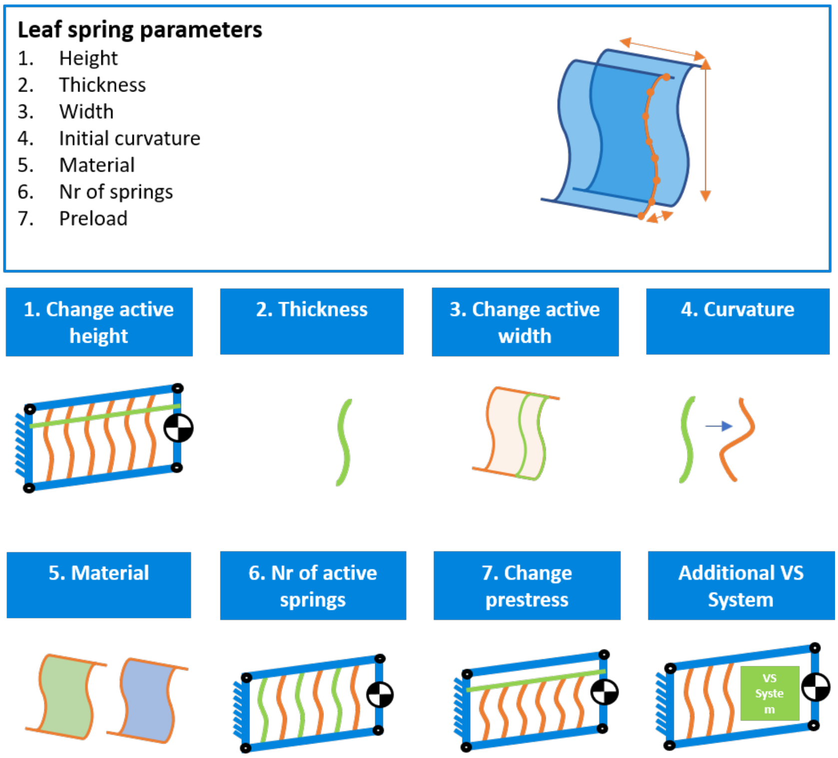


Figure A.4: ways to adjust the load-displacement curve or payload amplitude

A.4. MATERIALS FOR SPRING DESIGN

The following table is constructed using data from CES Edupack and shows minima and maxima of three categories. In his paper, Berntsen [11] uses the metric stress versus stiffness as σ_y/E to quantify the range of motion of the material. To quantify the amount of energy that can be stored into the material, the second metric shows σ_y^2/E. At last the amount of energy is calculated per unit mass $\sigma_y^2/(E\rho)$ as decribed in [12]. Although the selected material, RVS 1.4310, does not perform best on the defined metrics, it is selected as material for the prototype for practical considerations. It was available at suppliers. The material does not suffer from creep and stress relaxation, where polymers and plastic do at room temperature. Also titanium is very expensive. A more complete list could be made by doing more extensive search to suppliers data.

Material	**E (Gpa)**		**Sy (MPa)**		**Density (ρ)**		range of motion **Sy/E**		energy **Sy^2/E**		energy/mass **Sy^2/Eρ**	
	min	max	min	max	min	max	min	max	min	max	min	max
Low alloy steel	205	217	400	1500	7800	7900	2,0	6,9	780	10369	0,10	1,31
Stainless Steel	190	210	170	1000	7600	8100	0,9	4,8	152	4762	0,02	0,59
High Carbon Steel	200	215	400	1150	7800	7900	2,0	5,3	800	6151	0,10	0,78
Titanium Alloys	90	120	250	1250	4400	4800	2,8	10,4	694	**13021**	0,16	2,71
CFRP	70	150	650	1050	1500	1600	9,3	7,0	6036	7350	4,02	**4,59**
PMMA	2,2	3,8	54	72	1160	1220	24,5	18,9	1325	1364	1,14	1,12
Polypropyleen	0,9	1,5	20	37	890	910	22,2	**24,7**	444	913	0,50	1,00
PLA	3,3	3,6	55	72	1240	1240	16,7	20,0	917	1440	0,74	1,16
Paper	3	8,9	15	34	480	860	5,0	3,8	75	130	0,16	0,15
RVS 1.4310	190	200	770	1050	7800	7800	4,1	5,3	3121	5513	0,40	0,71

Figure A.5

A.5. STRAIN ENERGY IN LOADED BEAMS

The following table shows for different crossections of beams and for different types of loading the fraction of volume that is maximally used for storing strain energy. The values are partially based on [12], [13]. For beams that are axially loaded all volume is maximally used. However, for that application strains are very small, which is not practical. It is observed that torsion loading on a shaft has the highest efficiency.

Application	**Type of loading**	**Square**	**Circular**
Axial strain energy	Compression or extension	100%	100%
Bending strain energy	end point loading	11%	8%
	distributed loading	7%	5%
	moment loading	33%	25%
transverse strain energy (for A = 1, L = 5A)	end point loading	0,4%	0,3%
	distributed loading	0,5%	0,4%
	moment loading	0,0%	0,0%
Torsion strain energy	torsion moment loading	26%	50%
Special	Spiral leaf spring	33%	
	Linear coil spring		50%
	Torsion coil spring		25%

Figure A.6: The numbers show how much material volume is maximally used for storing strain energy. In most cases this is not 100%, because stresses are not evenly distributed.

A.6. IDEAS FOR FUTURE RESEARCH

Ideas that have come up during the research that could be valuable for future work are denoted here.

A.6.1. BOUNDARY CONDITIONS OF THE PARALLELOGRAM

The boundary conditions of the parallelogram were exploited to serve as imposed rotations for the spring. In other words, the outer ends were displaced over an equal finite rotation. These boundary conditions can also be applied in other situations. Figure A.7 shows a few other possible applications. Note that the boundary condition of equal finite rotation on both ends, can be seen in other perspective if the reference frame is fixed. This is explained in B.2. Therefore, the spring can also be applied in concentric axles for finite displacements.

The conformal transmission shown in the figure shows three grey gears. The left and right gear rotate with same angular rate. The spring is fixed to the gears. Therefore, equal angular displacements are imposed on the spring, just like in the parallelogram. This could be used to apply specific load-displacement characteristics on the transmission, for example for the purpose of static balancing.

Two concentric axles are shown in the right figure. The outer axles rotates around the inner axle. The spring is fixed between the two axles, but the outer end attached to the outer axle remains the same orientation (in this figure horizontal). By rotation of the outer axle, the outer end of the spring follows a sinusoidal path. By rotating the reference frame with respect to the inner axle, the boundary conditions can be seen similar to the parallelogram linkage. Therefore, this spring configuration can be used to create specific load displacement characteristics for concentric axles.

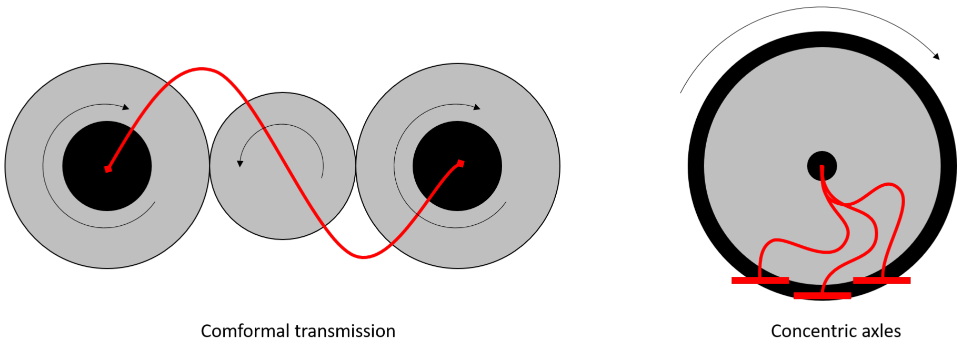


Figure A.7: Two possible applications that use the same boundary conditions to impose finite rotations on the spring as imposed in the parallelogram linkage.

A.6.2. WIDTH PATTERN IMPLEMENTATION

In the previous section is assumed that the width of the beam just a fixed parameter dependent on the beam's length. The width was assumed to vary from its center line from inside out. However, if we consider the beam as real spatial object, the total material used per width increment can be varied along the depth of the beam as well. For example, when making the beam smaller in width, material can be removed from the sides, but can also be removed from the center, as displayed in figure A.8. This could benefit the beams behavior, especially when the the beam gets wider and spatial effects come into play.

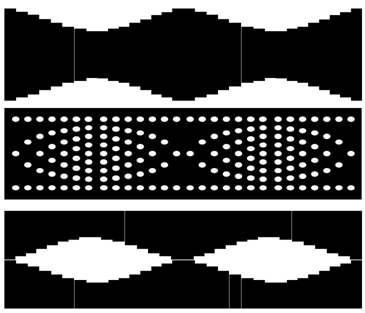


Figure A.8: Three different springs having the same width variation, because the amount of material along the length (from left to right) is for all the same.

A.6.3. TORSION BARS

Torsion bars and tubes in series can be folded to limit the maximum length of the mechanism. The system is then compressed to smaller length. The compactness of storing energy can be increased. Methods to create negative stiffness is however not known. It is possible to use end stops to create degressive behaviour [14].

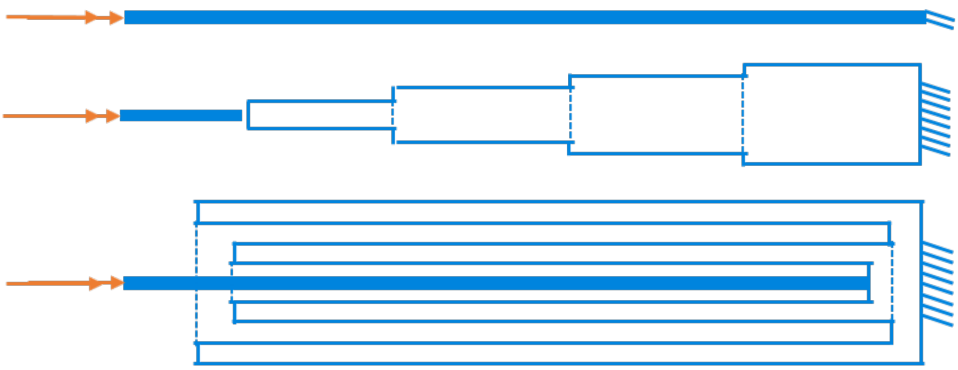


Figure A.9: Folded torsion bars

A.7. GUI

A GUI was used to quickly analyse simulations. The code of the gui is provided in the appendix A.7 and can be used for further analysis. The provided figure is an arbitrary snapshot.

The GUI shows in red the result of the selected rotation interval. In the most left column: on top the springs, in blue parallel identical springs. Below the top view of the spring. Colors indicate stresses. Below the elemental local rotation of the spring. On bottom the elemental stress. The second column shows from top to bottom per element: axial force, shear force, internal moment, curvature, relative strain energy. The third column shows the same quantities of the second column, only now for the endpoint nodes. The last column illustrates the movement of the springs within the parallelogram.

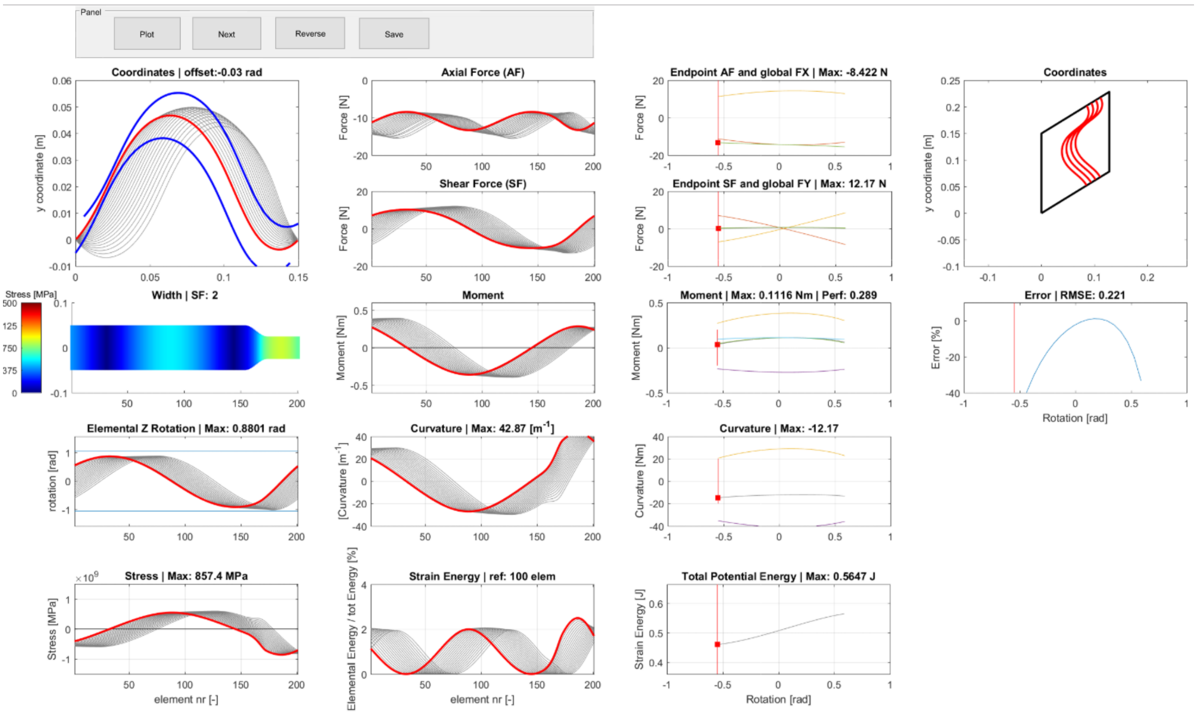


Figure A.10: GUI

A.8. BUILDING BLOCKS

This section shows the results of the all possible configurations for the spring constructed by 4 building blocks.

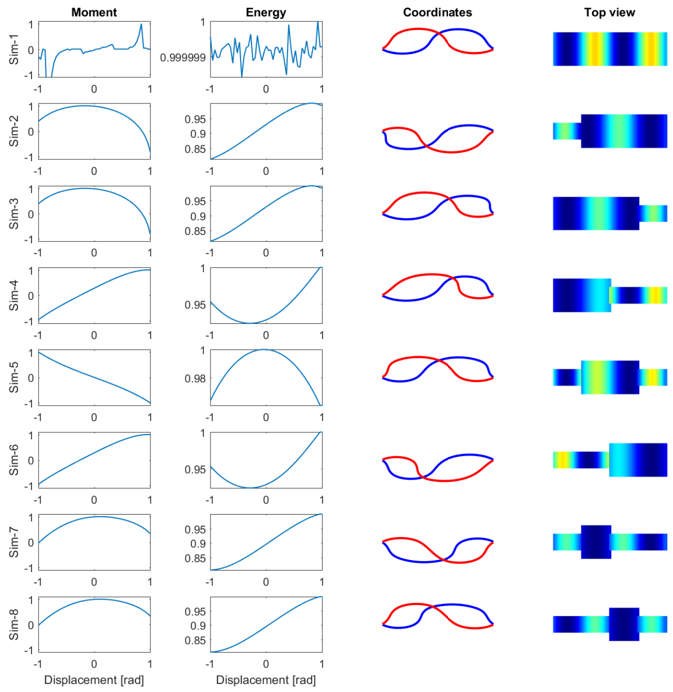


Figure A.11

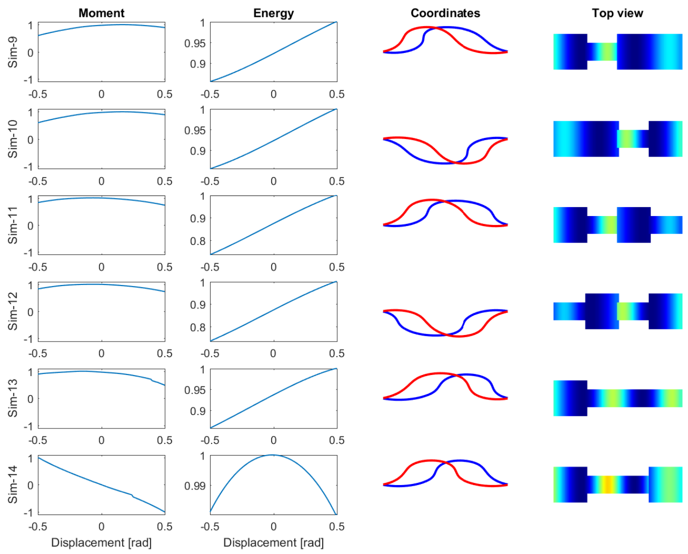


Figure A.12

A.9. PROTOTYPE AND MEASUREMENTS

The parallelogram constructed from PLA printed parts performed sufficiently accurate to test the springs. The connector, connecting the two arms of the parallellogram, is placed on the opposite side to create more space for the spring. The spring is not allowed to make contact with the parallelogram arms or the connector. The focus was on testing the springs so hinges were kept simple by using standard bearings. Compliant hinges can be used in improved designs to avoid friction to improve accuracy. The springs were designed to deflect to stresses up to 90% of the Yield strength. Reducing the load limit will strongly increase the lifetime of the springs. For industrial purposes a safety factor of at least 1/4 times the yield strength is required.

A.9.1. CAD MODEL AND CONSTRUCTION

The CAD model shows the design of the prototype. Clamps are used to make the spring easily removable. The prototype arms and connector (shown in white) are 3D-printed from PLA on standard settings on a Ultimaker 3 printer, having 0.4mm nozzle. 2 SKF 306 bearings were used per arm, separated by a distancer to avoid alignment problems. The bearings rotate around a f6 tolerance shoulder bolt. The spring is made from RVS 1.4310 spring steel ordered at JEVEKA: FOBLADA20200903, 0.2x305x1000mm. Brand: H+S. The shapes created using a lasercutter machine from the faculty of 3ME at TU Delft with a tolerance of 0.2mm.

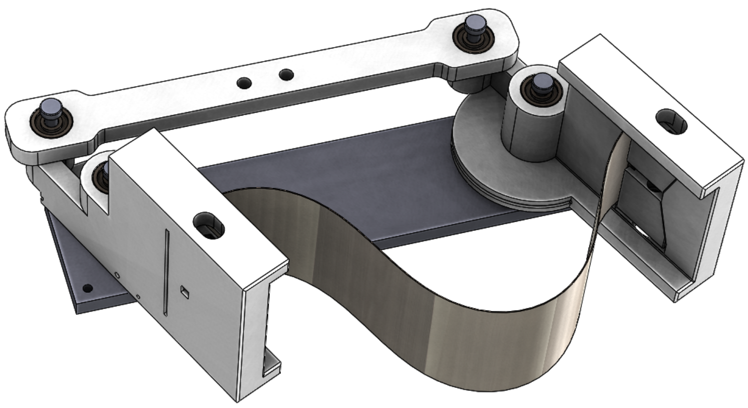


Figure A.13: CAD model (solidworks) from prototype used for measurements. The spring can be substituted for a different plate spring.

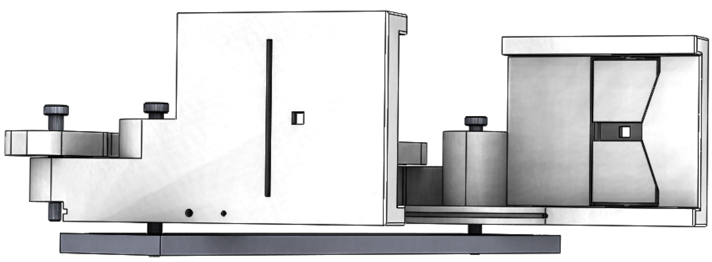


Figure A.14: Side view of the prototype - the clamp blocks can be pulled together by a bolt and nut. When moving to the middle the plate spring will be clamped.

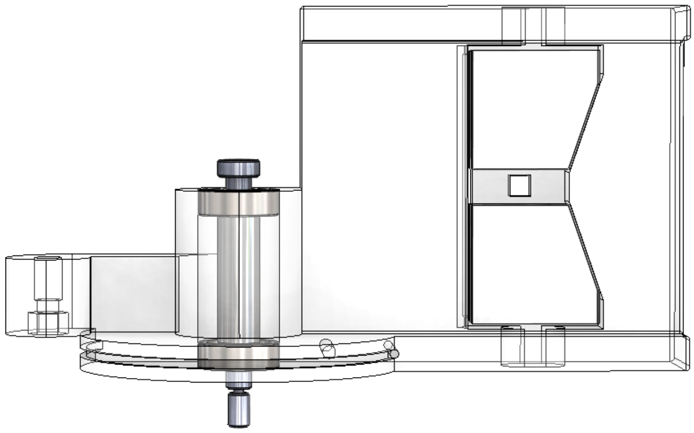


Figure A.15: Side view showing the bolt and bearings inside the parallelogram arm. The pulling disk is also clear visible below. Around this disk a wire pulls the arm to exert a moment on the parallelogram.

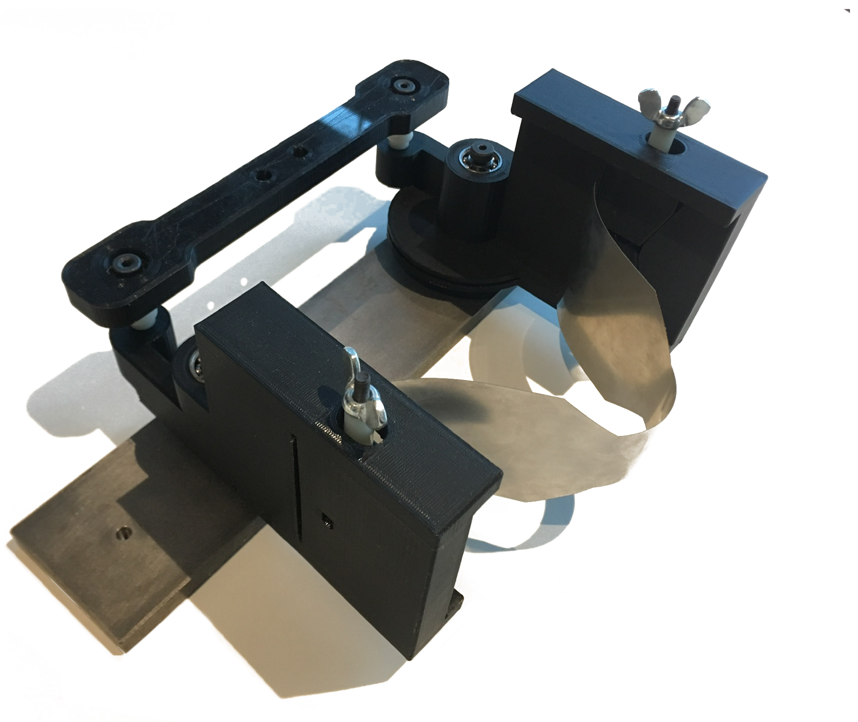


Figure A.16: Photograph of assembled prototype.

A.9.2. MEASUREMENT SETUP

Supplementary material about the testing stage is provided here.

List of possible sources for errors:

- Very small slip in attachment point of wire to prototype.
- Strain in wire
- Parallelogram not perfectly parallel
- Small plastic deformations in the spring from scratches, transportation or earlier measurements
- Friction and backlash of bearings

- Friction of pulleys from the load
- Elasticitiy of printed PLA
- clamping blocks not perfectly aligned with surface of parallelogram arm
- Circumference of pulling disk not perfectly round
- Height of pulling disk not perfectly aligned with height of point of application of wire to loadcell
- Tolerance of spring steel: E-modulus, thickness (3%)

Figure A.17: Photograph of the measurement stage.

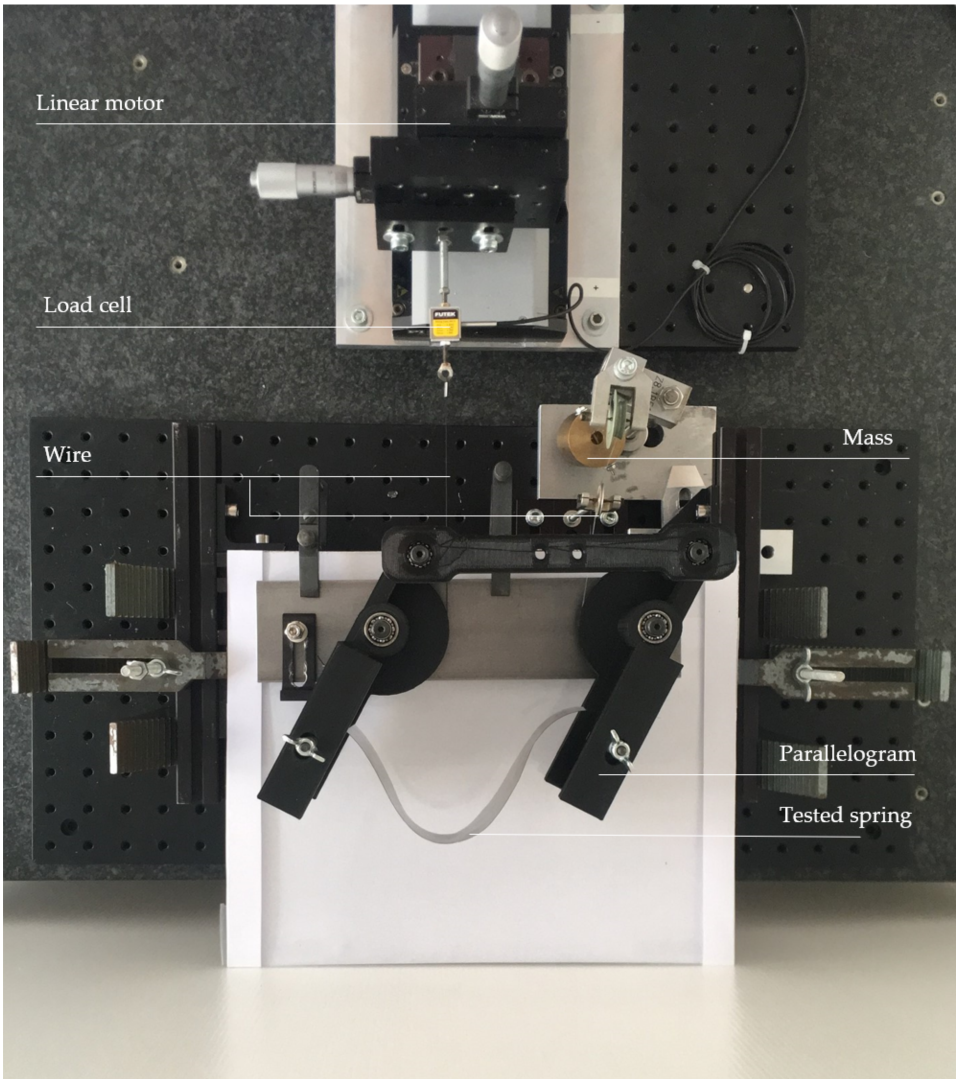


Figure A.18: Photograph of the measurement stage.

Figure A.19: Photograph of used wiring. The left wire is used for the mass. The right wire is used for the loadcell.

A.10. ANSYS MODEL

This section presents the setup of the ANSYS model. The goal of the ANSYS model is to simulate the spring behaviour for the presented boundary conditions as described in the paper, because analytical methods would be very tedious or maybe even not possible.

A.10.1. SETUP

The simulation setup is as follows. The main MATLAB script A1_LSW defines a spring width shape. The boundary conditions and parameters of the model are defined in a separate script A4_parameters. It runs a ANSYS APDL script from D which performs the actsual finite element simulation of the spring. The parameters are read and run and the ansys apdl program produces results to a batch run dataset. This dataset is loaded into the main script and processed to visual results. The results can be read by Matlab gui A7_LSWGUI. Using this ANSYS apdl batch file an optimization could be performed. The force and moment results from a single run can be processed and compared to a desired objective fucntion. The calculated error can then be send to the Matlab optimizer, for instance *fmincon*. The optimization script is then able to configure the initial shape conditions of the spring to improve the springs behavior.

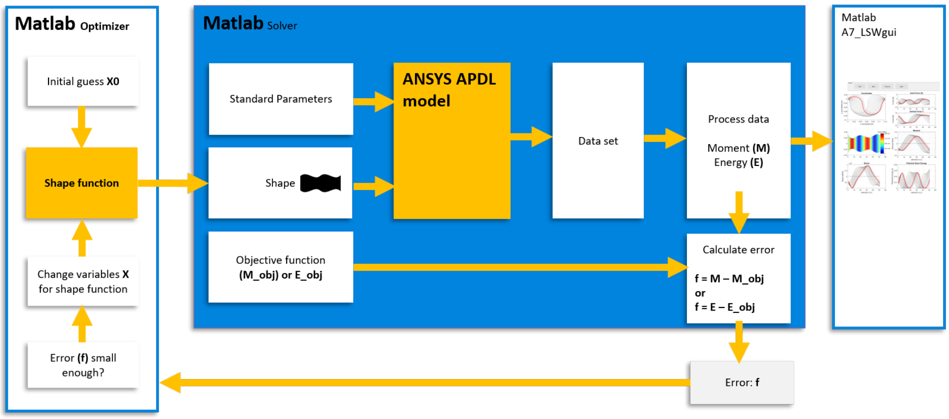


Figure A.20

A.10.2. APDL SCRIPT

This section describes the ANSYS APDL script used to model the springs in this research. The ANSYS APDL script first loads the parameters from ansys. Two files are loaded: C1_Parameters, which are the boundary conditon parameters and C2_shapedata, which is the vector describing the shape of the spring. The script continues by constructing the spring simulation using the parameters. A Bernoulli beam 188 element is used with rectangular crossection. The beam is constructed with inc amount of keypoints. Lines connect the keypoints and by meshing the lines and keypoints are converted to elements and nodes. Each line is a beam 188 element with a specified width from the width vector S.

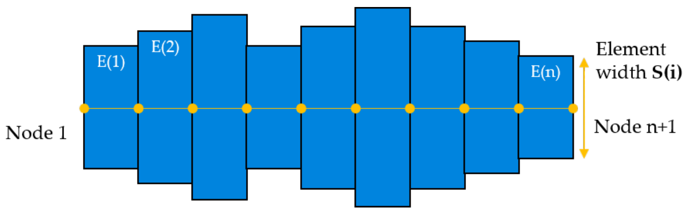


Figure A.21

The entire beam is initially constrained for all degrees of freedom to the most outer nodes. The boundary conditions are schematically displayed in figure B.6. Step zero is shows the contraining the outer nodes. During step 2 a small pertubation is performed to force the beam into an s-shape. Step 2 displaces the right right nodes to a specified compression. Step 3 removes the pertubation from step 2. Step 4 - Step 6 rotate the outer ends of the beam to a specified start condition. It can be seen as the initial condition for the actual parallelogram rotation. Step 7 is the imposed rotation on the outer ends which should be performed by the parallelogram. Step 4-6 can be be used to specify arbitrary initial conditions for the spring within the parallelgram. For instance the spring can be clamped with outer ends under different angles. In the paper and this report the outer end angles are kept equal.

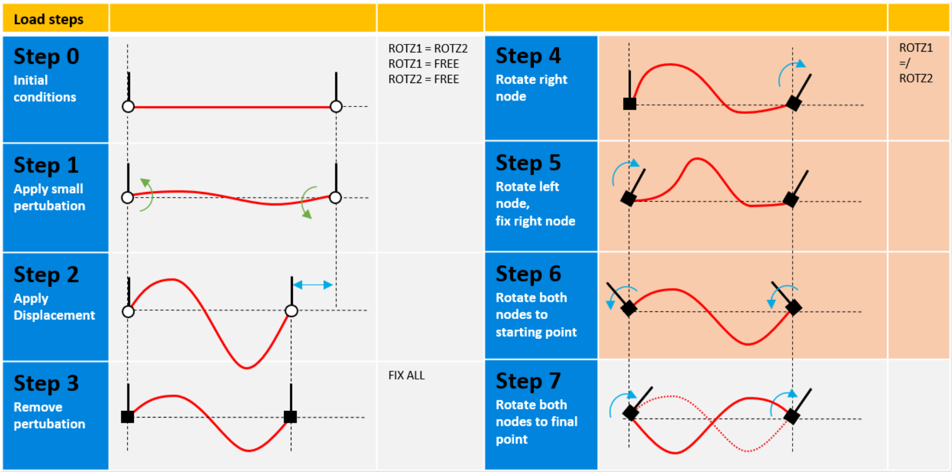


Figure A.22

The solution section produces results which are written to text files. The following nodal results from the outer nodes were extracted using the *RFORCE* command.

Nodal results	APDL command
Force x-direction node 1	RFORCE,11,ID_left ,F,X,FX1
Force y-direction node 1	RFORCE,12,ID_left ,F,Y,FY2
Moment z-direction node 1	RFORCE,13,ID_left ,M,Z,M1
Force x-direction node 2	RFORCE,14,ID_right,F,X,FX2
Force y-direction node 2	RFORCE,15,ID_right,F,Y,FY2
Moment z-direction node 2	RFORCE,16,ID_right,M,Z,M2

A.10.3. PRESTRESS OPTIONS

Figure A.23 shows an overview of the available options to apply presstress to the spring. The horizontal axis shows the imposed displacement to apply prestress. The vertical axis shows three possible options as boundary conditions for the outer ends of the spring. The fixed-fixed option means that both ends are clamped to a point that may or may not displace (translate or rotate). The fixed-hinged option means that one outer end may displace (both translate and rotate) but the hinged outer end may only displace, since the no rotation can be imposed. For the hinged-hinged option only prestress displacement can be imposed, because both outer ends are free to rotate, and not rotations can be imposed.

Focusing on the top level horizontal axis (yellow), two sections were created. Coupled imposed rotations and uncoupled imposed rotations. The coupled rotations are actually a subcategory of the uncoupled rotations, for the case where the rotation of left outer end is the the same as the right outer end.

In theory, all options are subcategories of option 10, where the prestress of the spring is fully defined by both outer end rotations and a translation. For example, option 11 is one of the solutions from option 10, only the right outer is now free to rotate, meaning that its angle can not be prescribed but depends on the other displacements. Focusing on option 6, rotations can not be imposed since the outer ends are hinged and are both free to rotate. Option 9 (and 12) are special, because of a translation the shape of the spring will form itself to its lowest energy shape. The spring can therefore not be formed into an s-shape.

Option 10 is used in the model of this thesis, to have full control on the imposed prestress and initial conditions of the spring. However, additional boundary conditions were required for implementation of this prescribed prestress freedom, so this effects the time to construct and solve the model.

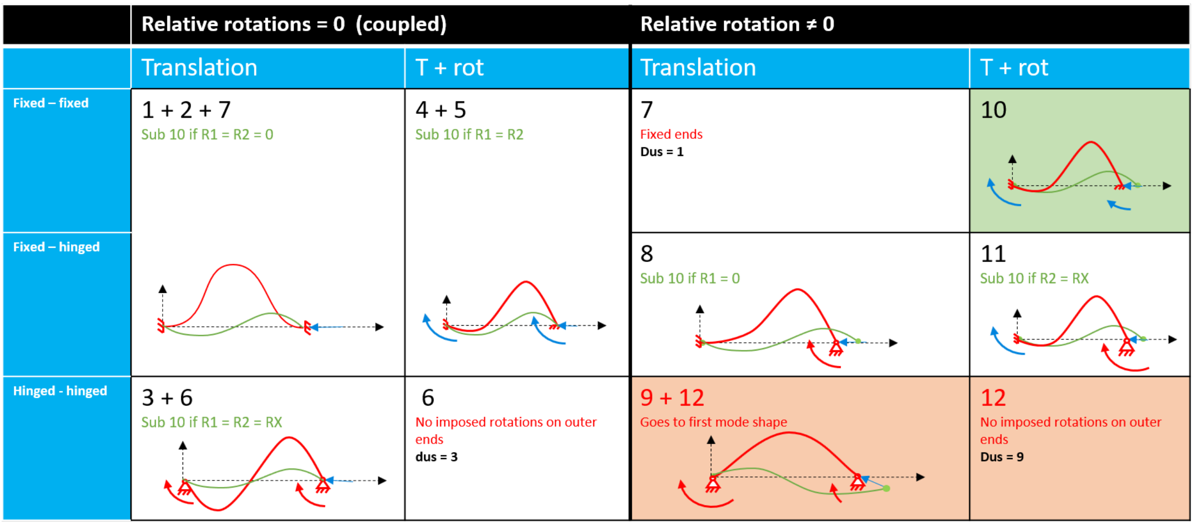


Figure A.23: Options to apply prestress.

A.11. TOLERANCES PARALLELOGRAM

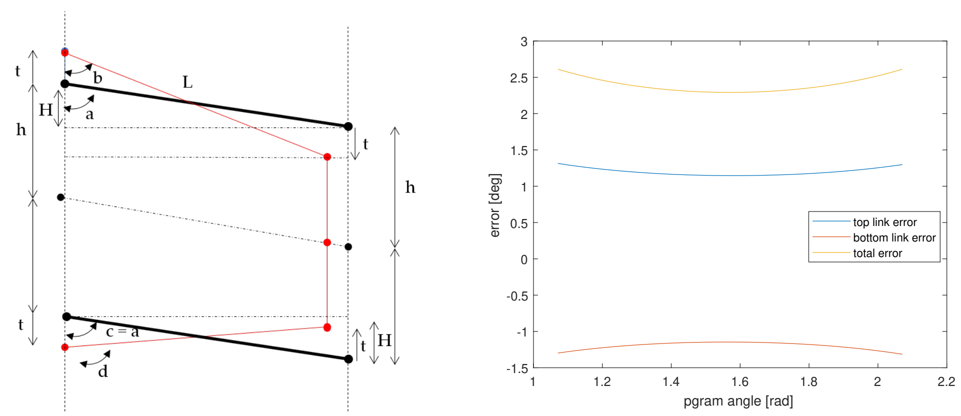
A small study is done to find out what error tolerance is on variations in the parallelogram linkage. The parallelogram is parametrized as illustrated in figure A.24. The figure illustrates a maximum offset t. The illustrated vertical red link in the figure is not necessarily 100% vertical in reality. The following angles can be expressed:

$$b = arccos\left(\frac{Lcos(a)+2t}{L}\right) \tag{A.1}$$

$$d = arccos\left(\frac{Lcos(a)-2t}{L}\right) \tag{A.2}$$

$$H = Lcos(a) \tag{A.3}$$

A plot shows the error in degrees for a parallelogram having arms with L = 1000 mm and an error t = 10 mm. Thus the vertical error t, selected at 1% (10/1000), gives a total maximum error around 3 degrees. Thus, the matlab script can be used to calculate the angular error along the range of motion, given a tolerance t. The angular error can be used for the spring design, which depends on the clamp angle (to the links) for the outer ends of the plate springs.



(a) (b)

Figure A.24: Dimensions of parallelogram (a) and error analysis (b) for L = 1000 mm and t = 10 mm along the range of motion from a= $pi/2 - 0.5$ rad to a = $\pi/2 + 0.5$ rad.

B

Appendix B - Additional projects

B.1. Static balancing parallelogram linkage

The parallelogram linkage with a mass can be statically balanced using a zero-free-length spring across the links. To explain this concept in more detail, the following equations are introduced. A zero-free-length helical spring with extension length s is positioned between the two links with length L. The spring will compensate the potential energy of the mass.

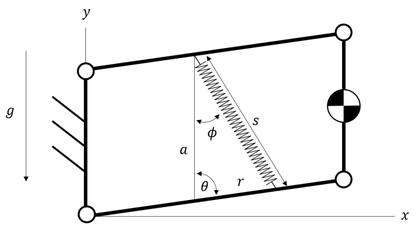


Figure B.1: Parallelogram linkage using ideal (zero-free-length)linear helical spring with extended length s to counteract gravity force for any angle θ.

The total potential energy of this system is described by the following energy balance

$$V_{total} = V_{mass} + V_{spring} = constant \tag{B.1}$$

The energy corresponding to the mass is described by

$$V_{mass} = mgLcos\theta \tag{B.2}$$

The potential energy of the spring depends on the geometry of the system and is described by the distance of the springs attachment point using the cosine rule.

$$s = \sqrt{a^2 + r^2 - 2ar \cdot cos\theta} \tag{B.3}$$

and the energy of the zero-free-length (ideal) spring would then be:

$$V_{spring} = ks^2 = k(a^2 + r^2 - 2ar \cdot cos\theta) \tag{B.4}$$

According to equation B.1 the energy should be constant for all angles so its derivative to θ should be zero, leading to:

$$mgL = akr \tag{B.5}$$

This equation only holds for implementations where $\phi \neq 0$. Many variations are done on this basic concept [15], [16]. However an ideal or emulated spring is required. Furthermore the concept works only for the gravity balancing objective of equation B.3, where a constant force is acting on the end effector, rotating around the hinge. The energy objective is illustrated in figure B.2.

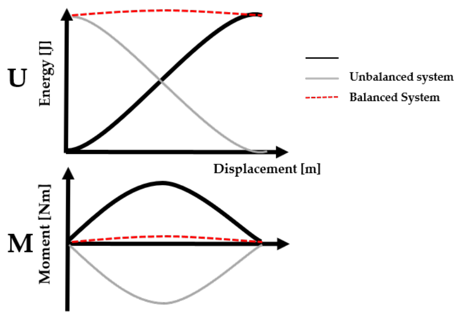


Figure B.2: Energy and moment plot of a balanced system, comprising of an unbalanced mechanism and a force compensation mechanism. In practice a system can not be perfectly balanced so a slight error is visible.

B.2. Bernoulli-Euler Beam theory

The problem of the large-displacement fixed guided beam having a constant cross-section can be analytically solved. Several analytical models have been synthesised to accurately predict the beams behaviour and shape functions. [9] [17] [18] It may not be directly visible that the problem as explained in the previous sections is a modified fixed guided beam problem. Figure B.3 shows that our problem is similar but the coordinate system is fixed to the parallelogram. In future work this may be helpful for modelling.

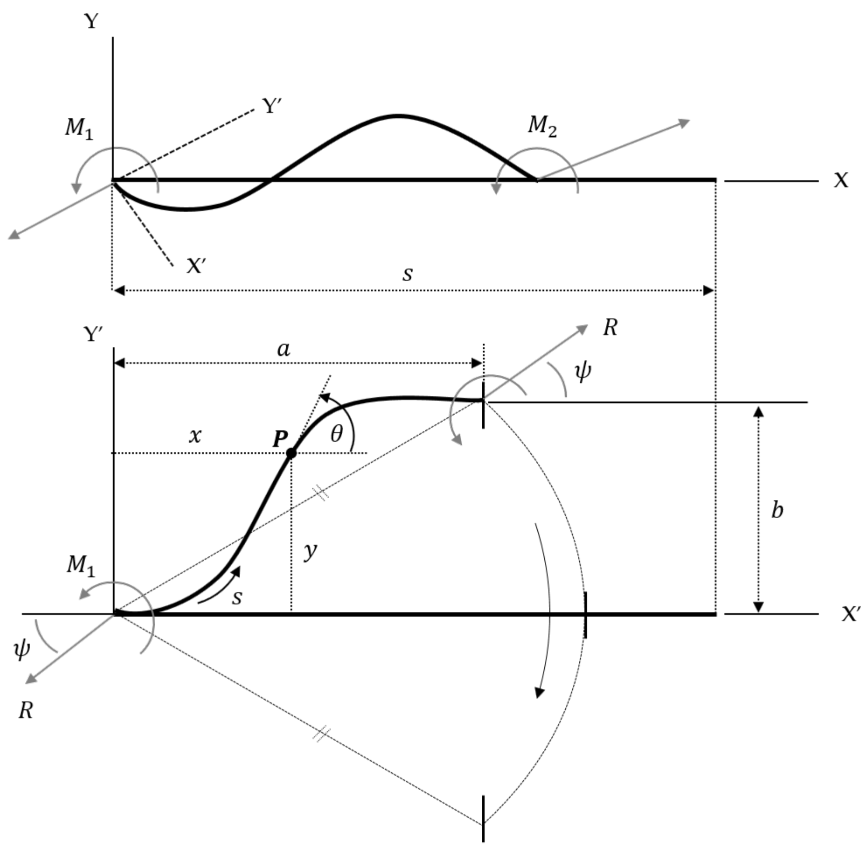


Figure B.3: Similar representations of the beam under large displacement. The first representation keeps the coordinate system (X,Y) fixed with respect to the beam. The second representation has a coordinate system (X', Y') fixed to the left outer end.

To make the similarity complete for its imposed boundary conditions, the fixed-guided beam should follow a sinusoidal path, because the distance between the outer ends is not changed. Finally, the clamp angle should remain zero since the rotating links remain parallel. The Bernoulli-Euler beam theory states that the relation between moment and curvature is linear. The Bernoulli-Euler equation for bending moment holds for any point in the beam and is described by equation **??**. The bending moment can then be expressed as

$$EI(s)\frac{ds}{d\theta} = M_1 - Rx\sin(\psi) + Ry\cos(\psi) \tag{B.6}$$

where M_1 is the reaction moment on the left side, R is the imposed force under an angle ψ and x and y are the coordinates of the considered point P of bending in the beam. $EI(s)$ represents the beam's stiffness by its Young's modulus and second moment of inertia, specifically indicated as a fucntion of the position s along the beam. For any point P along the beam's length s, the following geometric relations hold:

$$\frac{dy_a}{ds} = \sin(\theta) \qquad \frac{dx_a}{ds} = \cos(\theta) \tag{B.7}$$

For a beam with constant EI the procedure from Holst et al. [9] can be followed where elliptical integrals can be obtained for the end displacements:

$$\begin{aligned} \frac{b}{L} = \frac{-1}{\sqrt{\alpha}}\{&\sin\psi\left(2E(k,\phi_2) - 2E(k,\phi_1) - 2F(k,\phi_2) + 2F(k,\phi_1)\right) \\ &+2k\cos\psi(\cos\phi_1 - \cos\phi_2)\} \end{aligned} \tag{B.8}$$

$$\begin{aligned} \frac{a}{L} = \frac{-1}{\sqrt{\alpha}}\{&\sin\psi\left(2E(k,\phi_2) - 2E(k,\phi_1) - 2F(k,\phi_2) + 2F(k,\phi_1)\right) \\ &+2k\cos\psi(\cos\phi_2 - \cos\phi_1)\} \end{aligned} \tag{B.9}$$

F is the incomplete elliptical integral of the first kind and E is the incomplete elliptical integral of the second kind with ϕ is the amplitude k the modulus, defined by:

$$k\sin\phi = \cos\frac{\psi - \theta}{2} \tag{B.10}$$

At last the end moments can be calculated by:

$$M_{1,2} = 2k\sqrt{EIR}\cos\phi_{1,2} \tag{B.11}$$

The presented equations hold only for beams with constant E and I and without initial curvature, whereas we are now interested in a beam with varying width, thus a non-constant second moment of inertia. The presented equations can therefore not be directly used. However, the equations can still be used as reference of the constant width beams. Analytical expressions for large displacement beams having a varying width are left for future research.

B.3. VOLUME OCCUPANCY OF HELICAL SPRINGS

A small study is done to calculate the volume occupancy of conventional helical springs. First, a calculation is done for a unstretched spring. Only 39% is used for spring material for the unstretched spring. Second, another calculation is done for a stretched coil spring. Only 30% is used for spring material for the stretched spring, if stretched to the maximum allowable stress, the yield strength. A maximum spring index is selected to occupy as much space as possible. For a compression spring the volume occupation is the other way around. Since only 50% of the material is maximally utilized, only 15% of the occupied volume is used for strain energy. This number can serve as an incentive to investigate more efficient methods to store potential energy.

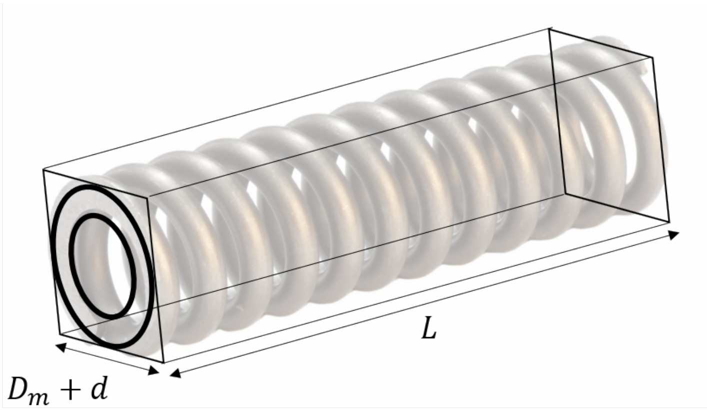


Figure B.4: A helical spring uses only a fraction of the unit cell box that is actually occupied in space. The space that is lost is inside the spring and between the coils. For the assumption of a rectangular box, also space is lost on the corners.

B.3.1. UNSTRETCHED SPRING

This calculation provides an estimation of the maximum volume efficiency that can be achieved for strain energy storage when using a conventional coil spring by extension or compression. No linkage configuration is provided. For this calculation we will only look a the space in use by the spring itself and its direct unit cell. The volume of a coil spring is:

$$V_{spring} = \pi n D_m \cdot \frac{\pi d^2}{4} \tag{B.12}$$

where Dm is the coil mean diameter, n the number of coils and d the wire diameter. The number of coils is defined by the free length L_0 divided by the wire diameter d. In this case the coils are in contact with the next coils when unstressed.

$$n = \frac{L_0}{d} \tag{B.13}$$

We assume a maximum spring index for maximum volume occupation. Therefore

$$Index = \frac{D_m}{d} = 4 \tag{B.14}$$

The volume of the unit cell (rectangular box) that is occupied by the coil spring is defined by the mean diameter:

$$V_{cell} = (D_m + d)^2 \cdot L_0 \tag{B.15}$$

The ratio of volume used by the coil spring for its free length is:

$$R = \frac{V_{spring}}{V_{cell}} = \frac{\pi n D_m \cdot \frac{\pi d^2}{4}}{(D_m + d)^2 \cdot L_0} = \frac{\pi^2 D_m \cdot d}{4(D_m + d)^2} = 0.39 \tag{B.16}$$

B.3.2. STRETCHED SPRING

We can also calculate the volume that is used by the spring while operating. So we calculate the extension of the spring and use the final length as length for the unit cell.

The maximum force the spring can endure is:

$$F = \frac{\pi d^3 \tau}{16r}; \tag{B.17}$$

where tau is the maximum stress The extension of the spring is then calculated by: [ref: http://werktuigbouw.nl/sub17.htm]

$$u = \frac{64 \cdot n \cdot r^3 \cdot F}{d^4 G}; \tag{B.18}$$

The unit cell volume becomes:

$$V_{cell,ext} = L \cdot (Dm + d)^2; \tag{B.19}$$

Where $L = L_0 + u$

The volume of the spring material stays the same. The volume ratio of the operating spring with respect to its unit cell then results in:

$$R_{ext} = \frac{V_{spring}}{V_{cell,ext}} = 0.30 \tag{B.20}$$

B.4. KINEMATIC OPTIONS MICROSCOPE STAND

A small study is done to investigate the kinematic options that are available to create three degrees of freedom for the microscope stand. The goal is to find out if there are other feasible configurationsto reach three degrees of freedom for the end effector. The available kinematic options are:

- Rotational joint: X(R_x)
- Rotational joint: Y(R_y)
- Rotational joint: Z (R_z)
- Translational joint: X (U_x)
- Translational joint: Y (U_y)
- Translational joint: Z (U_z)

Rotational joints can be thought of as hinges. Translational joints can be thought of a telescopic motion. An example is shown in figure B.5. The example comprises the following joints: R_z, U_z and R_x or R_y, since a rotation the first joint Rz can make the last joint Rx or Ry in the global coordinate system.

The total available options N, including mirrors and non-unique solutions due to the shown effect is calculated by:

$$N_{total} = 6 \cdot 6 \cdot 6 = 216 \tag{B.21}$$

Looking carefully to the base joint (joint 1), Ux, Uy, Rx and Ry are considered not feasible options as base joint, because the occupied volume increases significantly. This reduces the amount of options to: $N = 2 \cdot 6 \cdot 6 = 72$; Leaving only Rz or Uz as base joint options. (36 options for base joint Rz and 36 options for base joint Uz). These options are shown in the figure. The figure is ordered on basis of feasibility. Concept 43-72 are not feasible because the Uz joint is not followed by the Rz joint. Therefore, these configurations are considered inpractical. Concept 34-41 are considered non-feasible because the reach in the x-y plane or y-z plane is inpractical. This leaves us with 33 feasible options, from which 14 options are mirror versions. This leaves us with 18 unique options.

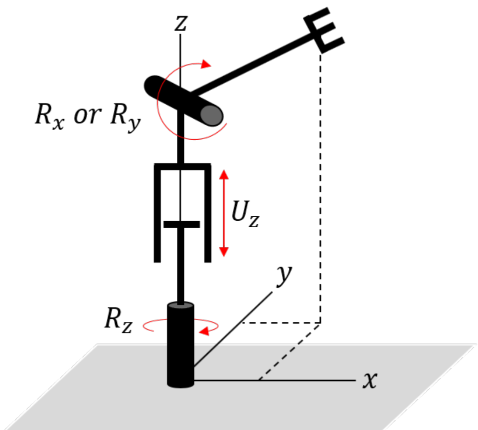


Figure B.5: Example of a 3DOF manipulator with end effector. This type is called Rz-Uz-Rx or Rz-Uz-Ry

Feasible concepts

Concept	DOF 1	DOF 2	DOF 3	Remark
1	Rz	Uz	Rx	
2	Rz	Rz	Rx	
3	Rz	Rx	Rx	
4	Rz	Rx	Rz	
5	Rz	Rz	Uz	
6	Rz	Uz	Rz	
7	Uz	Rz	Rz	
8	Rz	Rx	Uy	
9	Rz	Ux	Uz	
10	Rz	Uz	Ux	
11	Uz	Rz	Ux	
12	Rz	Rx	Ux	
13	Rz	Rx	Uz	
14	Rz	Ux	Rx	
15	Rz	Ux	Ry	
16	Rz	Ux	Rz	
17	Uz	Rz	Rx	
18	Rz	Rx	Ry	
19	Uz	Rz	Uy	mirrorversion
20	Rz	Ry	Ux	mirrorversion
21	Rz	Ry	Ry	mirrorversion
22	Rz	Ry	Rz	mirrorversion
23	Rz	Rz	Ry	mirrorversion
24	Rz	Uy	Uz	mirrorversion
25	Rz	Uz	Uy	mirrorversion
26	Rz	Ry	Uy	mirrorversion
27	Rz	Ry	Uz	mirrorversion
28	Rz	Uy	Rx	mirrorversion
29	Rz	Uy	Ry	mirrorversion
30	Rz	Uy	Rz	mirrorversion
31	Uz	Rz	Ry	mirrorversion
32	Rz	Uz	Ry	mirrorversion
33	Rz	Ry	Rx	mirrorversion

Non-feasible concepts

Concept	DOF 1	DOF 2	DOF 3	Remark
34	Rz	Rz	Rz	no height
35	Rz	Rz	Ux	no height
36	Rz	Rz	Uy	no height
37	Rz	Ux	Ux	no height
38	Rz	Ux	Uy	no height
39	Rz	Uy	Ux	no height
40	Rz	Uy	Uy	no height
41	Rz	Uz	Uz	no reach
42	Uz	Rz	Uz	no reach
43	Uz	Rx	Rx	u z not followed by r z
44	Uz	Rx	Ry	u z not followed by r z
45	Uz	Rx	Rz	u z not followed by r z
46	Uz	Rx	Ux	u z not followed by r z
47	Uz	Rx	Uy	u z not followed by r z
48	Uz	Rx	Uz	u z not followed by r z
49	Uz	Ry	Rx	u z not followed by r z
50	Uz	Ry	Ry	u z not followed by r z
51	Uz	Ry	Rz	u z not followed by r z
52	Uz	Ry	Ux	u z not followed by r z
53	Uz	Ry	Uy	u z not followed by r z
54	Uz	Ry	Uz	u z not followed by r z
55	Uz	Ux	Rx	u z not followed by r z
56	Uz	Ux	Ry	u z not followed by r z
57	Uz	Ux	Rz	u z not followed by r z
58	Uz	Ux	Ux	u z not followed by r z
59	Uz	Ux	Uy	u z not followed by r z
60	Uz	Ux	Uz	u z not followed by r z
61	Uz	Uy	Rx	u z not followed by r z
62	Uz	Uy	Ry	u z not followed by r z
63	Uz	Uy	Rz	u z not followed by r z
64	Uz	Uy	Ux	u z not followed by r z
65	Uz	Uy	Uy	u z not followed by r z
66	Uz	Uy	Uz	u z not followed by r z
67	Uz	Uz	Rx	u z not followed by r z
68	Uz	Uz	Ry	u z not followed by r z
69	Uz	Uz	Rz	u z not followed by r z
70	Uz	Uz	Ux	u z not followed by r z
71	Uz	Uz	Uy	u z not followed by r z
72	Uz	Uz	Uz	u z not followed by r z

Kinematics
Rx
Ry
Rz
Ux
Uy
Uz

Figure B.6: The right column is referred to as non-feasible because of the reasons stated in the remark box.

C

Appendix C - MATLAB code

C.1. Structure of MATLAB files

Figure C.1: Overview of matlab code

Filenr	Main files	Description	Filenr	Executed files	Description
01_00	**A9_runner**	Main executive file	01_01	**A1_LSW**	Runs the ANSYS batch file
			01_02	**A4_LSW_parameters**	Runs parameters
			01_03	**A7_LSWGUI**	Runs dashboard for analysis
			01_04	**A8_LSW_globals**	Runs global parameters
02_00	**A13_Measure**	Processes measured data	02_01	**A14_plotmeasurements**	Function file
03_00	**A15_Relations**	Processes simulations			

C.2. File 01_00

```
%% Run simulation

clear all
close all
clc

for iteration = 1:1
clearvars -except iii iteration compressionvector
run('A8_LSW_globals.m');
iii = iteration;
assignin('base','iii',iii)
assignin('base','compression',compression)
run('A1_LSW.m');
end
disp('done')

%% PLOT GUI MULTIPLE TIMES

% load('E2_output.mat');
%
% for i = 1:10
%
%      E1output = E2(end+1-i);
%      E1output = cell2mat(E1output);
%      save('E1_output.mat','E1output');
%
```

```
%       A7_LSWGUI;
%       hGuiFig = findobj('Tag','Guifig1','Type','figure');          %find figure
%       handles = guidata(hGuiFig);                                  %get handles
%       A7_LSWGUI('pushbutton1_Callback',handles.pushbutton1,[],handles); %push plot
%       A7_LSWGUI('pushbutton2_Callback',handles.pushbutton2,[],handles); %push next
%       A7_LSWGUI('Save_Callback',handles.Save,[],handles);          %push save
%       close(A7_LSWGUI)
%
% end

%% ELEMENT CONTROL
% This part checks the minimum amount of elemnets required for having an
% accurate solver. The solver compares different element sets. When the
% next larger element set has an offset smaller than 1% the amount of
% elements is sufficient.

% for now 81 elements is sufficient. Turned off for convenience and speed.

controller = [10 20 50 100 400]; %(inc-1) = deelbaar door 4,
clen       = length(controller);

for jj = 1:clen
clearvars -except jj controller clen; close all; clc;
run('A8_LSW_globals.m');
elementcontrol = controller(jj);
run('A1_LSW.m');
end

load('E2_output.mat');
CompareE2 = E2((end-(clen-1)):end);

for jjj = 1:clen
cdata = cell2mat(CompareE2(jjj));
M1node(:,jjj) = cell2mat(cdata.M1node);
M2node(:,jjj) = cell2mat(cdata.M2node);
Kelem        = cell2mat(cdata.Kurv);
Sene         = cell2mat(cdata.Energy);
for jjjj = 1:50
SumSene(jjjj,jjj) = sum(Sene(:,jjjj));
K1elem(jjjj,jjj) = Kelem(1,jjjj);
end
end

jjj = 1;
for jjj = 1:(clen-1)
check1(:,jjj) = M1node(:,jjj)./M1node(:,jjj+1);
check2(:,jjj) = M2node(:,jjj)./M2node(:,jjj+1);
check3(:,jjj) = SumSene(:,jjj)./(SumSene(:,jjj+1));
check4(:,jjj) = K1elem(:,jjj)./K1elem(:,jjj+1);

d1(:,jjj) = M1node(:,jjj)./M1node(:,clen);
d2(:,jjj) = M2node(:,jjj)./M2node(:,clen);
d3(:,jjj) = SumSene(:,jjj)./(SumSene(:,clen));
d4(:,jjj) = K1elem(:,jjj)./K1elem(:,clen);
end

for jjj = 1:(clen-1)
check1total(jjj) = max(abs(check1(:,jjj)-1))*100;
check2total(jjj) = max(abs(check2(:,jjj)-1))*100;
check3total(jjj) = max(abs(check3(:,jjj)-1))*100;
```

```
check4total(jjj) = max(abs(check4(:,jjj)-1))*100;

d1total(jjj) = max(abs(d1(:,jjj)-1))*100;
d2total(jjj) = max(abs(d2(:,jjj)-1))*100;
d3total(jjj) = max(abs(d3(:,jjj)-1))*100;
d4total(jjj) = max(abs(d4(:,jjj)-1))*100;
end

checkvalues = [check1total; check2total; check3total; check4total]
dvalues = [d1total; d2total; d3total; d4total]
```

C.3. FILE 02_00

```
%% Measurements
% processes the measurements and generates plots for figure 15 and 16.

clear all
clc
close all

directory = 'C:\Users\Roel van Ekeren\OneDrive\Afstuderen\Ansys\09 LSW\';
datefolder = 'Meting 2019_07_31\';
filetype1 = '.csv';

files1    = { '19 07 31 15 48 28 Roel GP real w0_1'
'19 07 31 15 51 36 Roel GP real w0_1'
'19 07 31 16 02 11 Roel GP real w0_2'
'19 07 31 16 22 40 Roel GP real w0_2'
'19 07 31 16 29 45 Roel GP real w0_3'
'19 07 31 16 36 53 Roel GP real w0_3'
'19 07 31 15 08 46 Roel GP real rm1_1'
'19 07 31 15 11 33 Roel GP real rm1_1'
'19 07 31 15 14 48 Roel GP real rm1_2'
'19 07 31 15 17 44 Roel GP real rm1_2'
'19 07 31 15 20 31 Roel GP real rm1_3'
'19 07 31 15 23 19 Roel GP real rm1_3'
'19 07 31 17 13 24 Roel GP real rm2_1'
'19 07 31 17 26 32 Roel GP real rm2_1'
'19 07 31 17 29 52 Roel GP real rm2_2'
'19 07 31 17 32 37 Roel GP real rm2_2'
'19 07 31 17 35 22 Roel GP real rm2_3'
'19 07 31 17 38 02 Roel GP real rm2_3'
'19 07 31 17 44 38 Roel GP real rm3_1'
'19 08 01 10 26 48 Roel GP real rm3_1'
'19 08 01 10 40 10 Roel GP real rm3_2'
'19 08 01 10 43 16 Roel GP real rm3_2'
'19 08 01 10 48 05 Roel GP real rm3_3'
'19 08 01 10 53 33 Roel GP real rm3_3'

};

% skim datasets
dataset1 = {};
for i = 1:length(files1(:,1))
rawdata = xlsread(strcat(directory,datefolder,files1{i},filetype1));
dataset1{i} = rawdata(:,[2,4]);
end

```

```

% Plot all datasets
for ii = 1:length(files1(:,1))
m  = cell2mat(dataset1(ii));
d   = m(:,1);
f   = m(:,2);


plot(d,f); hold on

end
legvec = string([1:1:ii]);
legend(legvec)

% Save all datasets to mat-file
dataset1 = table2struct(cell2table(dataset1));
save('rmdata1.mat','dataset1');


%% LOAD DATASETS

global radius mass0 mass1 mass2 g
close all;
clear all
clc

load('rmdata1.mat')

% Create variables
radius= 0.0361;
mass0 = 0;
mass1 = 0.050;
mass2 = 0.36;
g     = 9.81;


%% PLOT DATA --------------------------------------------------------------------------------

% weight
w0left_force = dataset1.dataset12(50:end,2);
w0right_force = dataset1.dataset11(50:end,2);
weight1      = mean(w0left_force);
weight2      = mean(w0right_force);
weight       = mean([weight1 weight2]);

close all;


%% WEIGHTS
A14_plotmeasurements(dataset1.dataset12, dataset1.dataset11,radius, mass0, g, 0,'b');
A14_plotmeasurements(dataset1.dataset14, dataset1.dataset13,radius, mass0, g, 0,'b');
A14_plotmeasurements(dataset1.dataset16, dataset1.dataset15,radius, mass0, g, 0,'b');


%% SPRING 1 SPRING A - POSITVIE AND NEGATIVE STIFFNESS
close all
figure
% load('E1_output.mat')
% load('RM1_0.mat') % ANSYS spring 1
```

```
% load('RM1_1.mat') % ANSYS spring 1
load('RM1_5.mat') % ANSYS spring 1
[MN1i, pm3  ]       = A16_plotansys(E1output,0.755); %0.755
[mo1, di1, pm1, pm2] = A14_plotmeasurements(dataset1.dataset18, dataset1.dataset17,
    radius, mass0, g, weight,'b'); %(0.9 - 0.7 rad)
[mo2, di2, pm1, pm2] = A14_plotmeasurements(dataset1.dataset110, dataset1.dataset19,
    radius, mass0, g, weight,'b');
[mo3, di3 ,pm1, pm2] = A14_plotmeasurements(dataset1.dataset112,
    dataset1.dataset111, radius, mass0, g, weight,'b');
for i = 1:length(mo1.mean)
mo_mean(i) = mean([mo1.mean(i) mo2.mean(i) mo3.mean(i)]);
end
for i = 1:length(MN1i);
SE(i) = (MN1i(i)-mo_mean(i))^2;
SEN(i) = (MN1i(i)/max(MN1i)-mo_mean(i)/max(mo_mean(300:800)))^2;
Err(i) = MN1i(i)-mo_mean(i);
Err_n(i) = 100*abs(Err(i))/abs(MN1i(i));
end
RMSE1 = sqrt(nansum(SE)/length(SE))
NMSE1 = sqrt(nansum(SEN)/length(SEN))
rho_cA = corrcoef(MN1i,mo_mean,'rows','complete')
pm4 = plot(10.^-6.*di3.mright./radius,mo_mean','g');
pm5 = plot(10.^-6.*di3.mright./radius,Err,'k');
legend([pm2 pm3 pm4 pm5],'Measurements','ANSYS','Measurements
    Mean','Error','location','southeast')
title('Spring A - Positive-Negative Stiffness')

% norm error
figure
plot(10.^-6.*di3.mright./radius,sqrt(SEN),'k')
ylim([0 1])

%% SPRING C - NEGATIVE STIFFNESS

figure
load('RM2_1.mat') % ANSYS spring 2
[MN1i  ,pm3] = A16_plotansys(E1output,0.9);
[mo1, di1,pm1, pm2] = A14_plotmeasurements(dataset1.dataset114, dataset1.dataset113,
    radius, mass0, g, weight,'b'); %(0.9 - 0.7 rad)
[mo2, di2,pm1, pm2] = A14_plotmeasurements(dataset1.dataset116, dataset1.dataset115,
    radius, mass0, g, weight,'b');
[mo3, di3,pm1, pm2] = A14_plotmeasurements(dataset1.dataset118, dataset1.dataset117,
    radius, mass0, g, weight,'b');
for i = 1:length(mo1.mean)
mo_mean(i) = mean([mo1.mean(i) mo2.mean(i) mo3.mean(i)]);
end
for i = 1:length(MN1i);
SE(i) = (MN1i(i)-mo_mean(i))^2;
SEN(i) = (MN1i(i)/max(MN1i)-mo_mean(i)/max(mo_mean(96:1245)))^2;
Err(i) = MN1i(i)-mo_mean(i);
Err_n(i) = 100*abs(Err(i))/abs(MN1i(i));
end
RMSE2 = sqrt(nansum(SE)/length(SE))
NMSE2 = sqrt(nansum(SEN(1:1235))/(length(SEN(1:1235))))
rho_cC = corrcoef(MN1i,mo_mean,'rows','complete')
pm4 = plot(10.^-6.*di3.mright./radius,mo_mean','g');
pm5 = plot(10.^-6.*di3.mright./radius,Err,'k');
legend([pm2 pm3 pm4 pm5],'Measurements','ANSYS','Measurements
    Mean','Error','location','southeast')
title('Spring C - Negative Stiffness')
```

```
figure
plot(10.^-6.*di3.mright./radius,sqrt(SEN),'k')
ylim([0 1])

%% SPRING B - POSITIVE STIFFNESS
figure
load('RM3_3.mat') % ANSYS spring 3
[MN1i  , pm3 ] = A16_plotansys(E1output,0.75);
[mo1, di1] = A14_plotmeasurements(dataset1.dataset119, dataset1.dataset120, radius, mass0, g, weight,'b');%(0.9 - 0.7 rad)
[mo2, di2] = A14_plotmeasurements(dataset1.dataset121, dataset1.dataset122, radius, mass0, g, weight,'b');
[mo3, di3,pm1, pm2] = A14_plotmeasurements(dataset1.dataset123, dataset1.dataset124, radius, mass0, g, weight,'b');
for i = 1:length(mo1.mean)
mo_mean(i) = mean([mo1.mean(i) mo2.mean(i) mo3.mean(i)]);
end
for i = 1:length(MN1i);
SE(i) = (MN1i(i)-mo_mean(i))^2;
SEN(i) =(MN1i(i)/max(MN1i)-mo_mean(i)/max(mo_mean(40:1090)))^2;
Err(i) = MN1i(i)-mo_mean(i);
end
RMSE3 = sqrt(nansum(SE)/length(SE))
NMSE3 = sqrt(nansum(SEN)/length(SEN))
rho_cB = corrcoef(MN1i,mo_mean,'rows','complete')
pm4 = plot(10.^-6.*di3.mright./radius,mo_mean','g');
pm5 = plot(10.^-6.*di3.mright./radius,Err,'k');
legend([pm2 pm3 pm4 pm5],'Measurements','ANSYS','Measurements Mean','Error','location','southeast')
title('Spring B - Positive Stiffness')

figure
plot(10.^-6.*di3.mright./radius,sqrt(SEN),'k')
ylim([0 1])


%% SETUP - WEIGHT ONLY
figure
weight = 3.22;
[mo1, di1] = A14_plotmeasurements(dataset1.dataset11, dataset1.dataset12, radius, mass0, g, weight,'b');%(0.9 - 0.7 rad)
[mo2, di2] = A14_plotmeasurements(dataset1.dataset13, dataset1.dataset14, radius, mass0, g, weight,'b');
[mo3, di3,pm1, pm2] = A14_plotmeasurements(dataset1.dataset15, dataset1.dataset16, radius, mass0, g, weight,'b');
for i = 1:length(mo1.mean)
mo_mean(i) = mean([mo1.mean(i) mo2.mean(i) mo3.mean(i)]);
end
pm4 = plot(10.^-6.*di3.mright./radius,mo_mean','g'); hold on

weight = 0;
[mo3 , di4, pm6,pm7] = A14_plotmeasurements(dataset1.dataset15, dataset1.dataset16, radius, mass0, g, weight,'m');
pm8 = plot(10.^-6.*di3.mright./radius,mo_mean+3.22*radius','c'); hold on
legend([pm2 pm4 pm7 pm8],'setup without weight','setup mean without weight','setup','setup mean', 'location','east')
title('SETUP')
ylim([-0.01 0.15])
grid on
```

C.4. FILE 03_00

```
%% Process simulations for three spring types
% This file processes simualations 1-12 and creates the plot and tabledata
% presented in table 4.


clear all
clc
close all

load('E2_output.mat');
% E4 = cell2mat(E2([1 4 6]));
% E6 = cell2mat(E2([77 76 75 79])); %spring A
% E7 = cell2mat(E2([83 86 87 89 93])) %spring B
% E8 = cell2mat(E2([90 91 92]));    %spring C
% E9 = cell2mat(E2([96:101,103:104])); %blocksimulation
% E10 = cell2mat(E2([105:110]));    %blocksimulation
run('A4_LSW_parameters.m');

%% SPRING A (E2 79 75 76 77)
% Different compressions are simulated

xaxisx = [-2 2];
xaxisy = [0 0] ;

load('E6.mat');
Sim = E6;

figure
for i = 1:length(Sim)
Rotation       = cell2mat(Sim(i).RotN1);
Sim(i).Msum    = -(cell2mat(Sim(i).M1node) + cell2mat(Sim(i).M2node));      % resulting absolute moment
Sim(i).Mnorm   = Sim(i).Msum/max((Sim(i).Msum));                            % normalized moment
RotZ           = Sim(i).RotZ{1,1};
Stressset      = Sim(i).Stress{1,1};
Maxrotation(i) = max(abs(RotZ(:)));
Maxstress(i)   = max(abs(Stressset(:)));
[zerox(i,:), zeroy(i,:)] = intersections(xaxisx, xaxisy, -Rotation, Sim(i).Mnorm); % find intersection with xaxis
ROM(i)                  = zerox(i,2)-zerox(i,1);                            % calculate range of motion
%________________________


% Load Energy Data
Shape      = cell2mat(Sim(i).Shape);
Energy     = cell2mat(Sim(i).Energy);
for iv = 1:length(Energy(1,:))
Sumenergy(iv) = sum(Energy(:,iv));
end


ef        = [0.2 0.4 0.6 0.6];
len       = totlen/(1-ef(i));
```

```
% STRAIN ENERGY
Uspring(i) = max(Sumenergy)-min(Sumenergy);
Unorm(i)  = min(Sumenergy)/max(Sumenergy);
Volume    = len*sum(Shape)*(len/inc)*thickness;
Umass     = rho*Volume*g*2*0.5;

% Metrics
eta_SE(i) = E*Uspring(i)/(sigma^2*Volume);
eta_SE2(i) = E*Uspring(i)/(Maxstress(i)^2*Volume);
eta_GB(i) = 2*rho*g*E*0.5/(eta_SE(i)*sigma^2);
eta_GB2(i) = 2*rho*g*E*0.5/(eta_SE2(i)*Maxstress(i)^2);

%________________________

intnr = 200; % interpolation nr
for iii = 1:3

% create interpolation interval
newint(1,:) = linspace(zerox(i,1), zerox(i,2), intnr);

% try other smaller interpolation intervals
newint(2,:) = linspace(-0.5,0.5,intnr);
newint(3,:) = linspace(-pi/2,pi/2,intnr);

% interpolate over ROM
Sim(i).Mnormint = interp1(-Rotation,Sim(i).Mnorm,newint(iii,:));

% Objective function
Mobj(iii,:) = sin(newint(iii,:)+pi/2);

for ii = 1:length(Sim(i).Mnormint)
Sim(i).Err(ii) = ((Sim(i).Mnormint(ii))-Mobj(iii,ii));
Sim(i).SE(ii)  = ((Sim(i).Mnormint(ii)-Mobj(iii,ii)))^2;
end

Sim(i).RMSE(iii) = sqrt(nansum(Sim(i).SE)/length(~isnan(Sim(i).SE)));
Sim(i).MaxE(iii) = max(abs(Sim(i).Err));
end

plot(-Rotation,Sim(i).Mnorm,'-','LineWidth',1); hold on
xlabel('ROM [rad]')
ylabel('Moment (normalised) [-]')
ylim([0 1.1]);
end

plot(newint(3,:),Mobj(3,:),'LineWidth',2,'color','b'); hold on;
plot([0.5 0.5],[0 1.1],'k--')
plot([-0.5 -0.5],[0 1.1],'k--')

legend( 'Sim 1: \zeta = 20%',...
'Sim 2: \zeta = 40%',...
'Sim 3: \zeta = 60%',...
'Sim 4: \zeta = 60%',...
'objective',...
'location','northeast');

varNames    = {'zeta','RMSE1','RMSE2','ROM','Us','Un','eta_SE','eta_GB'};
Compressions = {'20'; '40'; '60';'60'};
RMSE1       = round(100*[Sim(1).RMSE(1); Sim(2).RMSE(1); Sim(3).RMSE(1);
    Sim(4).RMSE(1)],2);
```

```
RMSE2       = round(100*[Sim(1).RMSE(2); Sim(2).RMSE(2); Sim(3).RMSE(2); Sim(4).RMSE(2)],2);
ROM         = round(ROM,2);
eta_SE      = round(100*eta_SE ,2);
eta_GB      = round(100*eta_GB ,2);
Uspring     = round(Uspring,3);
Unorm       = round(100*Unorm,2);

T1 = table(Compressions,RMSE1,RMSE2,ROM',Uspring',Unorm',eta_SE',eta_GB','VariableNames',varNames)




%% SPRING B

load('E7.mat');
Sim = E7;

figure
for i = 1:length(Sim)

Rotation       = cell2mat(Sim(i).RotN1);
Sim(i).Msum    = -(cell2mat(Sim(i).M1node) + cell2mat(Sim(i).M2node));      % resulting absolute moment
Sim(i).Mnorm   = Sim(i).Msum/max((Sim(i).Msum));                           % normalized moment
RotZ           = Sim(i).RotZ{1,1};
Stressset      = Sim(i).Stress{1,1};
Maxrotation(i) = max(abs(RotZ(:)));
Maxstress(i)   = max(abs(Stressset(:)));
ROM(i)         = Rotation(1)-Rotation(end);


ef       = [0.2 0.4 0.6 0.6 0.2];
len      = totlen/(1-ef(i));
%________________________
% Load Energy Data
Shape      = cell2mat(Sim(i).Shape);
Energy     = cell2mat(Sim(i).Energy);
for iv = 1:length(Energy(1,:))
Sumenergy(iv) = sum(Energy(:,iv));
end

% STRAIN ENERGY
Uspring(i) = max(Sumenergy)-min(Sumenergy);
Unorm(i)   = min(Sumenergy)/max(Sumenergy);
Volume     = len*sum(Shape)*(len/inc)*thickness;
Umass      = rho*Volume*g*2*0.5;

% Metrics
eta_SE(i) = E*Uspring(i)/(sigma^2*Volume);
eta_SE2(i) = E*Uspring(i)/(Maxstress(i)^2*Volume);
eta_GB(i) = 2*rho*g*E*0.5/(eta_SE(i)*sigma^2);

%________________________



intnr = 200; % interpolation nr
```

```
for iii = 1:2

% try interpolation intervals
newint(1,:) = linspace(-pi/2,pi/2,intnr);
newint(2,:) = linspace(-0.5,0.5,intnr);

% interpolate over ROM
Sim(i).Mnormint = interp1(-Rotation,Sim(i).Mnorm,newint(iii,:));

% Objective function
Mobj(iii,:) = sin(newint(iii,:));

for ii = 1:length(Sim(i).Mnormint)
Sim(i).Err(ii) = ((Sim(i).Mnormint(ii))-Mobj(iii,ii));
Sim(i).SE(ii)  = ((Sim(i).Mnormint(ii)-Mobj(iii,ii)))^2;
end

Sim(i).RMSE(iii) = sqrt(nansum(Sim(i).SE)/length(~isnan(Sim(i).SE)));
Sim(i).MaxE(iii) = max(abs(Sim(i).Err));
end


plot(-Rotation,Sim(i).Mnorm,'-','LineWidth',1); hold on
xlabel('ROM [rad]')
ylabel('Moment (normalised) [-]')
ylim([-1.1 1.1]);
end

plot(newint(1,:),Mobj(1,:),'LineWidth',2,'color','b'); hold on;
plot([0.5 0.5],[-1 1],'k--')
plot([-0.5 -0.5],[-1 1],'k--')

legend( 'Sim 5: \zeta = 20%',...
'Sim 6: \zeta = 40%',...
'Sim 7: \zeta = 60%',...
'Sim 8: \zeta = 60%',...
'Sim 9: \zeta = 20%',...
'objective',...
'location','southeast');

varNames    = {'zeta','RMSE1','RMSE2','ROM','Us','Un','eta_SE','eta_GB'};
Compressions = {'20'; '40'; '60';'60'; '20'};
RMSE1       = round(100*[Sim(1).RMSE(1); Sim(2).RMSE(1); Sim(3).RMSE(1);
    Sim(4).RMSE(1); Sim(5).RMSE(1)],2);
RMSE2       = round(100*[Sim(1).RMSE(2); Sim(2).RMSE(2); Sim(3).RMSE(2);
    Sim(4).RMSE(2); Sim(5).RMSE(1)],2);
ROM         = round(ROM,2);
eta_SE      = round(100*eta_SE ,2);
eta_GB      = round(100*eta_GB ,2);
Uspring     = round(Uspring,3);
Unorm       = round(100*Unorm,2);

T2 =
    table(Compressions,RMSE1,RMSE2,ROM',Uspring',Unorm',eta_SE',eta_GB','VariableNames',varNames)


%% SPRING C

load('E8.mat');
Sim = E8;
```

```
figure
for i = 1:length(Sim)

Rotation        = cell2mat(Sim(i).RotN1);
Sim(i).Msum     = -(cell2mat(Sim(i).M1node) + cell2mat(Sim(i).M2node));       % resulting absolute moment
Sim(i).Mnorm    = Sim(i).Msum/max((Sim(i).Msum));                             % normalized moment
RotZ            = Sim(i).RotZ{1,1};
Stressset       = Sim(i).Stress{1,1};
Maxrotation(i) = max(abs(RotZ(:)));
Maxstress(i)    = max(abs(Stressset(:)));
ROM(i) = Rotation(1)-Rotation(end)

ef       = [0.2 0.4 0.6];
len      = totlen/(1-ef(i));
%________________________
% Load Energy Data
Shape      = cell2mat(Sim(i).Shape);
Energy     = cell2mat(Sim(i).Energy);
for iv = 1:length(Energy(1,:))
Sumenergy(iv) = sum(Energy(:,iv));
end

% STRAIN ENERGY
Uspring(i) = max(Sumenergy)-min(Sumenergy);
Unorm(i)   = min(Sumenergy)/max(Sumenergy);
Volume     = len*sum(Shape)*(len/inc)*thickness;
Umass      = rho*Volume*g*2*0.5;

% Metrics
eta_SE(i) = E*Uspring(i)/(sigma^2*Volume);
eta_SE2(i) = E*Uspring(i)/(Maxstress(i)^2*Volume);
eta_GB(i) = 2*rho*g*E*0.5/(eta_SE(i)*sigma^2);


%________________________

intnr = 200; % interpolation nr
for iii = 1:2

% create interpolation interval
newint(1,:) = linspace(-pi/2,pi/2,intnr);
newint(2,:) = linspace(-0.5,0.5,intnr);

% interpolate over ROM
Sim(i).Mnormint = interp1(-Rotation,Sim(i).Mnorm,newint(iii,:));

% Objective function
Mobj(iii,:) = sin(newint(iii,:)+pi);

for ii = 1:length(Sim(i).Mnormint)
Sim(i).Err(ii) = ((Sim(i).Mnormint(ii))-Mobj(iii,ii));
Sim(i).SE(ii)  = ((Sim(i).Mnormint(ii)-Mobj(iii,ii)))^2;
end

Sim(i).RMSE(iii) = sqrt(nansum(Sim(i).SE)/length(~isnan(Sim(i).SE)));
Sim(i).MaxE(iii) = max(abs(Sim(i).Err));
end
```

```
plot(-Rotation,Sim(i).Mnorm,'-','LineWidth',1); hold on
xlabel('ROM [rad]')
ylabel('Moment (normalised) [-]')
ylim([-1.1 1.1]);
end

plot(newint(1,:),Mobj(1,:),'LineWidth',2,'color','b'); hold on;
plot([0.5 0.5],[-1 1.1],'k--')
plot([-0.5 -0.5],[-1 1.1],'k--')

legend('Sim 10: \zeta = 20%',...
'Sim 11: \zeta = 40%',...
'Sim 12: \zeta = 60%',...
'objective','location','southeast');

varNames    = {'zeta','RMSE1','RMSE2','ROM','Us','Un','eta_SE','eta_GB'};
Compressions = {'20'; '40'; '60'};
RMSE1       = round(100*[Sim(1).RMSE(1); Sim(2).RMSE(1); Sim(3).RMSE(1)],2);
RMSE2       = round(100*[Sim(1).RMSE(2); Sim(2).RMSE(2); Sim(3).RMSE(2)],2);
ROM         = round(ROM,2);
eta_SE      = round(100*eta_SE ,2);
eta_GB      = round(100*eta_GB ,2);
Uspring     = round(Uspring,3);
Unorm       = round(100*Unorm,2);

T2 =
    table(Compressions,RMSE1,RMSE2,ROM',Uspring',Unorm',eta_SE',eta_GB','VariableNames',varNames)

%% GRAPHS

% load('E2_output.mat');
% E5 = cell2mat(E2([12 11 10 8 9]));
% save('E5.mat','E5')

load('E5.mat')
Sim = E5;

for iiv = 1:length(Sim)


Energyset = Sim(iiv).Energy{1,1};
Stressset = Sim(iiv).Stress{1,1};
par = Sim(iiv).parameters{1,1};

for iv = 1:length(Energyset(1,:))
Energy(iv) = sum(Energyset(:,iv));
end

Energystored(iiv) = abs(Energy(end)-Energy(1));
Maxstress(iiv)  = 1e-6*max(abs(Stressset(:)));

LT(iiv)         = par(2)/par(1);


end

Volume = (par(1)*par(2)*par(3));
Energystoredvolume = Energystored/Volume;
```

```
% Stacking
nrsprings = 1/(par(1)+5*par(1));


figure
plot(LT,Energystored,'s-')
xlabel('L/t [-]')
ylabel('Energy stored [J]')
title('Energy Geometry relation | Compression 20%')
figure
plot(LT,Maxstress,'s-')
xlabel('L/t [-]')
ylabel('Max Stress [MPa]')
title('Stress Geometry relation | Compression 20%')
figure
plot(LT,Energystoredvolume,'s-')
xlabel('L/t [-]')
ylabel('Energy stored [J/m^3]')
title('Energy Stored per volume | Compression 20%')
figure
yyaxis left
plot(Maxstress,Energystoredvolume,'s-')
xlabel('max stress [MPa]')
ylabel('Energy stored per volume [J/m^3]')
yyaxis right
plot(Maxstress,LT,'s-')
ylabel('L/t [-]')
title('Stress Energy relation | Compression 20%')

%% Blocksimulation

m = 6;
n  = 4;
xaxisx = [-2 2];
xaxisy = [0 0] ;

load('E9.mat');
load('E10.mat');
% Sim = E9;
Sim = E10;


h = figure
for i = 1:length(Sim)
clear Sumenergy

Rotation        = cell2mat(Sim(i).RotN1);
Sim(i).Msum     = -(cell2mat(Sim(i).M1node) + cell2mat(Sim(i).M2node));       % resulting
    absolute moment
Sim(i).Mnorm    = Sim(i).Msum/max((Sim(i).Msum));                              %
    normalized moment
RotZ            = Sim(i).RotZ{1,1};
Stressset       = Sim(i).Stress{1,1};
Maxrotation(i)  = max(abs(RotZ(:)));
Maxstress(i)    = 1e-6*max(abs(Stressset(:)));
Xpos            = cell2mat(Sim(i).Xpos);
Ypos            = cell2mat(Sim(i).Ypos);

% Load Energy Data
Shape       = cell2mat(Sim(i).Shape);
```

```
Energy     = cell2mat(Sim(i).Energy);
for iv = 1:length(Energy(1,:))
Sumenergy(iv) = sum(Energy(:,iv));
end
Sumenergy(end+1) = Sumenergy(iv);
Energynorm = Sumenergy/max(Sumenergy);

Sabs    = 1.5*max(Energy(:));
perc    = abs(Energy(:,1))/Sabs;
new     = ceil(perc*256);
scaling = jet(256);

ef        = 0.4;
len       = totlen/(1-ef);
% STRAIN ENERGY
Uspring(i) = max(Sumenergy)-min(Sumenergy);
%          Unorm(i)  = min(Sumenergy(i,:))/max(Sumenergy(i,:));

%________________________


h1(i) = subplot(m,n,n*i-3);
plot(-Rotation,Sim(i).Mnorm,'-','LineWidth',1); hold on
%xlabel('ROM [rad]')
ylabel(strcat('Sim-',sprintf('%d',i+8)))
ylim([-1.1 1.1]);

h2(i) = subplot(m,n,n*i-2);
plot(-Rotation,Energynorm,'-','LineWidth',1); hold on
%xlabel('ROM [rad]')
%ylabel('Moment (normalised) [-]')
ylim([min(Energynorm) 1]);

h3(i) = subplot(m,n,n*i-1);

animatie1 = plot(Xpos(:,end),Ypos(:,end),'color','b','LineWidth',2); hold on
animatie2 = plot(Xpos(:,1) ,Ypos(:,1) ,'color','r','LineWidth',2);
axis off

h4(i) = subplot(m,n,n*i);

for k = 1:length(Shape)
animatie9a = line([k k],[0 0.5*Shape(k)],'color',scaling(new(k),:),'LineWidth',3);
animatie9b = line([k k],[0 -0.5*Shape(k)],'color',scaling(new(k),:),'LineWidth',3);
end
colormap(jet(256));
axis off




end
h.Name = 'Moment, Energy and Shape';
set(h1(1).Title,'String','Moment');
set(h1(6).XLabel,'String','Displacement [rad]');
set(h2(1).Title,'String','Energy');
set(h2(6).XLabel,'String','Displacement [rad]');
set(h3(1).Title,'String','Coordinates');
set(h4(1).Title,'String','Top view');
```

C.5. FILE 01_01

```
%% ANSYS SIMULATION PRESTRESSED LEAF SPRING
% This simulation prestresses a leaf spring and rotates both outer ends.
% The model describes the situation of a leaf spring being compressed and
% rotated by a parallellogram four bar linkage.
% On bottom a small script can calculate the closest distance to a nearby
% spring.

% ANSYS WORKING DIR: /CWD,'C:\Users\Roel van Ekeren\OneDrive\Afstuderen\Ansys\09 LSW'
% ANSYS READ FILE : /input,B5_LSW,txt

% clear all
% clc
% close all
% rng('shuffle');


%% Directory names

global dir_ansys fileafs_apdl fileafs_simout parameters shapedata curvedata cadcoor
    cadcoorspring

dir_ansys      = 'C:\Program Files\ANSYS Inc\v190\ansys\bin\winx64\ANSYS190.exe';
fileafs_apdl   = 'B5_LSW.txt';
fileafs_simout = 'fileafs.simout';
parameters     = 'C1_parameters.macro';
shapedata      = 'C2_shapedata.txt';
cadcoor        = 'C3_cadcoordinates.txt';
cadcoorspring  = 'C4_cadcoorspring.txt';
% curvedata    = 'C3_curvedata.txt';


%% PARAMETERS
run('A4_LSW_parameters.m');
run('A8_LSW_globals.m');

%% CHECK FILES

files = {
'D1_anpar.txt';...
'D2_coordinates.txt';...
'D3_results.txt';...
'D4_elementtable.txt';...
'D5_energies1.txt';...
'D6_energies2.txt';...
'D7_moment1.txt';...
'D8_stress1.txt';...
'D9_translationx.txt';...
'D10_translationy.txt';...
'D11_forcex.txt';...
'D12_forcey.txt';...
'D13_rotationz.txt';...
'D14_kurvature.txt';...
'D15_kurvaturej.txt';...
'D16_ntx.txt';...
'D17_nty.txt';...

'fileafs.db';...
```

```
'fileafs.DSP';...
'fileafs.err';...
'fileafs.esav';...
'fileafs.full';...
'fileafs.ldhi';...
'fileafs.log';...
'fileafs.mntr';...
'fileafs.rdb';...
'fileafs.rst';...
'fileafs.simout';...

'E1_output.mat';...
};

for i = 1:length(files)
if exist(char(files(i)),'file')
delete(char(files(i)))
end
fid = fopen(char(files(i)),'w');
fclose(fid);
end
clear i

%% CROSSECTION
close all;

X1      = linspace(0,len,inc);

q    = [22.7 31.3 45.5 54.1 68.1 76.1]; % graph 1
q    = [31.8 40  59   67.7 100     ];  % for graph 2 and 3
%     q    = [30 40 60 70 100];
%     q    = [15 25 45 55 75 85];
%     q    = [5 20 30 45 60 70 80 95]

par1 = q(1);
par2 = q(2);
par3 = q(3);
par4 = q(4);
par5 = q(5);
par6 = q(6);

maxwidth = 1*(maxw+minw);
minwidth = 1*(minw-maxw);
shape    = ones(1,inc)*maxwidth;

t1 = round(inc*par1/100);
t2 = round(inc*par2/100);
t3 = round(inc*par3/100);
t4 = round(inc*par4/100);
t5 = round(inc*par5/100);
t6 = round(inc*par6/100);

ramp1 = linspace(maxwidth,minwidth,(t2-t1));
ramp2 = linspace(minwidth,maxwidth,(t4-t3));
ramp3 = linspace(maxwidth,minwidth,(t6-t5));

shape(1:t1)      = maxwidth;
shape(t1+1:t2)   = ramp1;
shape(t2+1:t3)   = minwidth;
shape(t3+1:t4)   = ramp2;
```

```
shape(t4+1:t5)   = maxwidth;
shape(t5+1:t6)   = ramp3;
shape(t6+1:end) = minwidth;

%_____________________________________________________________________
%%    BUILDING BLOCKS
%    blocks   = 4;                    %amount of blocks
%    blen     = floor(inc/blocks);    %elements per block
%    blenp    = 100/blocks;           %percentage of beam
%    blen     = (inc)*blenp/100;
%    maxwidth = 1*(maxw+minw);
%    minwidth = 1*(minw-maxw);
%
%    B1 = ones(1,blen)*maxwidth;
%    B0 = ones(1,blen)*minwidth;
%
%    iii = 1; % alleen aanzetten voor de elementcontrol
%    combinations = [
%                   B1 B1 B1 B1
%                   B0 B1 B1 B1
%                   B1 B0 B1 B1
%                   B1 B1 B0 B1
%                   B1 B1 B1 B0
%                   B1 B1 B0 B0
%                   B0 B1 B1 B0
%                   B0 B0 B1 B1
%                   B1 B0 B1 B0
%                   B0 B1 B0 B1
%                   B1 B0 B0 B1
%                   B1 B0 B0 B0
%                   B0 B1 B0 B0
%                   B0 B0 B1 B0
%                   B0 B0 B0 B1
%                  ];
%
%    randomform = round(rand(blocks,1));
%    formation = [combinations(iii,:)];
%    shape      = [formation];
%
%_____________________________________________________________________
% RANDOM SPLINE

%                X       = abs(rand(7,1));
%                X       = ones(7,1);
%                X       = [1; 1; 1; 0; 1; 0; 0; 0];
%                X       = [0.658;0.083;0.590;0.454;0.170;0.654;0.670];
%                X       = flip(X);
%                ary     = 1*minw+(2*maxw)*X;
%                arx     = linspace(0,len,length(ary));
%                shape   = spline(arx,ary,X1);

%_____________________________________________________________________
% COSINE SHAPE BUILDING BLOCKS

%                   close all;
%                   ratio      = 15;             % percentage of length where pi/2 fits
    in.
%                   period     = 100/ratio*pi;   % period
%                   plen       = period/4;       % lengte van rise
%
```

```
%                   t0 = 0;
%                   t1 = 70; %35
%                   t2 = 65;
%                   t3 = 30;
%
%                   ps0         = period*(1-t0/100)-pi;
%                   ps1         = period*(1-t1/100);            % phase shift
%                   ps2         = period*(1-t2/100)-pi;
%                   ps3         = period*(1-t3/100);
%
%                   shape0      = 2*minw+2*maxw*cos(period/len*X1-ps0);
%                   shape1      = 2*minw+2*maxw*cos(period/len*X1-ps1);
%                   shape2      = 2*minw+2*maxw*cos(period/len*X1-ps2);
%                   shape3      = 2*minw+2*maxw*cos(period/len*X1-ps3);
%
% %                 e01            = (telem-1)/100*(100-t0);        %weghalen bij
    normaal
%                   e0             =  (telem-1)/100*(100-t0+ratio); %weghalen bij
    normaal
%                   e1             =  (telem-1)/100*(100-t1);
%                   e2             =  (telem-1)/100*(100-t1+ratio);
%                   e3             =  (telem-1)/100*(100-t2);
%                   e4             =  (telem-1)/100*(100-t2+ratio);
%                   e5             =  (telem-1)/100*(100-t3);
%                   e6             =  (telem-1)/100*(100-t3+ratio);
%
%                   shape1(1:e01)   = min(shape1);
%                   shape1(e01:e0)  = shape0(e01:e0); %weghalen bij normaal
%                   shape1(e0:e1)   = max(shape1);
%                   shape1(e1:e2)   = shape1(e1:e2);
%                   shape1(e2:e3)    = min(shape0);
%                   shape1(e3:e4)   = shape2(e3:e4);
%                   shape1(e4:e5)   = max(shape0);
%                   shape1(e5:e6)   = shape3(e5:e6);
%                   shape1(e6:end)  = min(shape0);


%_____________________________________________________________________
% STRAIGHT SHAPE

% check for different widths: 15, 20, 25, 30
% 50, 40, 30, 20,
%     percentage = (telem-1)/100;
%
%     b1 = 30*percentage;
%     b2 = 20*percentage;
%     b3 = 30*percentage;
%
%     shape(1:b1)          = max(shape);
%     shape(b1:b1+b2)      = min(shape);
%     shape(b1+b2:b1+b2+b3) = max(shape);
%     shape(b1+b2+b3:end) = min(shape);
%_____________________________________________________________________

%_____________________________________________________________________
% Curvature of beam
% ampl      = 0.01;       % [m] amplitude of curvature
% curve     = ampl*sin(X1*(2*pi/len));
%_____________________________________________________________________
```

```
%______FLIP
%   shape5      =  flip(shape1);

%______MIRROR
%   shapegem    = (min(shape)+max(shape)/2);
%   shapemir    =  ((shape - shapegem)*-1)+shapegem/2;
%   shape       = shapemir;

%______STRAIGHT
%   shape(1:end) = max(shape);

%_____________________________________________________________________
% PLOT
trueshape1 = 0.5*shape;
trueshape2 = -0.5*shape;
figure
plot(X1,[trueshape1; trueshape2]); hold on
axis('equal')
title('top view of the spring with varying width')
%_____________________________________________________________________


%% WRITE PARAMETERS IN MACRO AND TXT FILES

% Vectors to be written to ANSYS
vars     = {'Ey','nu','rho','len','thickness','section','inc','minw','maxw','nelem','incr','loadf','steps','telem','
x        = [ E;  nu;  rho;  len;  thickness;  section;  inc;  minw;  maxw;  nelem;  incr;  loadf;  steps;  telem;  nnodes;  rot;  preload;  prerot1;  prerot2;  prerot3;  prerot4];

% write parameters
fid_out = fopen(parameters, 'w');
for k = 1:length(vars)
fprintf(fid_out, '%s=%10.8f \r\n', vars{k}, x(k)) ;
end
fclose(fid_out);

% write shapedata
fid_out = fopen(shapedata, 'w');
for k1 = 1:length(shape)
fprintf(fid_out, '%1.6f \r\n', shape(k1)) ;
end
fclose(fid_out);

% nodal coordinates txtfile
cadcoordinates = [trueshape1;X1;zeros(1,length(X1))];
fid_out = fopen(cadcoor, 'w');
for k = 1:length(X1)
fprintf(fid_out, '%10.8f %10.8f %10.8f \r\n', cadcoordinates(1,k),cadcoordinates(2,k),cadcoordinates(3,k) ) ;
end
fclose(fid_out);

% Write curvedata
% fid_out = fopen(curvedata, 'w');
% for k2 = 1:length(shape)
%     fprintf(fid_out, '%1.6f \r\n', curve(k2)) ;
% end
% fclose(fid_out);
```

```
run = 'ready';

%% RUN SIMULATION
% Delete previous simulation file
if exist('fileafs.lock', 'file')
delete('fileafs.lock');
end

tic
% Run program
cmd = strcat('SET KMP_STACKSIZE=2048k & "', dir_ansys, '" -b -j fileafs -dir "', pwd
    ,'" -i "', pwd, '\', fileafs_apdl, '" -o "', pwd, '\', fileafs_simout, '"');
status = dos(cmd);
toc

%% LOAD DATA FROM ANSYS FILES

% data parameters
data1 = load('D1_anpar.txt');
data2 = load('D2_coordinates.txt');
data3 = load('D3_results.txt');
data4 = load('D4_elementtable.txt');
data5 = load('D5_energies1.txt');
data6 = load('D6_energies2.txt');
data7 = load('D7_moment1.txt');
data8 = load('D8_stress1.txt');
data9 = load('D9_translationx.txt');
data10 = load('D10_translationy.txt');
data11 = load('D11_forcex.txt');
data12 = load('D12_forcey.txt');
data13 = load('D13_rotationz.txt');
data14 = load('D14_kurvature.txt');
data15 = load('D15_kurvaturej.txt');

data16 = load('D16_ntx.txt');
data17 = load('D17_nty.txt');


if isempty(data2) == true
% type fileafs.err
disp('------file not solved------')
RMSE_shift_int = 1;
else


%% STRUCTURE DATA
n1 = data1(1);
n2 = data1(2);
n3 = data1(3);
n4 = data1(4);
n5 = data1(5);
n6 = data1(6);
n7 = data1(7);
n8 = data1(8)+1;

ntotal = n1+n2+n3+n4+n5+n6+n7+n8;
npre   = ntotal-n8;

% Relevant load step (only last step)
data2b = data3([1:n8]+npre-1,:);
```

```
% Rotations Node 1 and Node 2
rotN1 = data2b(:,1)-data2b(1,1); %from zero
rotN2 = data2b(:,2)-data2b(1,2); %from zero
RotN1 = data2b(:,1);
RotN2 = data2b(:,2);

% Structural moments and forces Node 1 and 2
FX1 = data2b(:,3);
FY1 = data2b(:,4);
MN1 = data2b(:,5);
FX2 = data2b(:,6);
FY2 = data2b(:,7);
MN2 = data2b(:,8);

% Stresses
S   = abs(data4(:,1))*10^-6;

% Initial Positions
% Nodal coordinates undeformed
X0n = [data2(1,1);data2(3:end,1);data2(2,1)];
Y0n = [data2(1,2);data2(3:end,2);data2(2,2)];

% Element Coordinates undeformed
X0e = data2(1:end-1,4);
Y0e = data2(1:end-1,5);

% Displacements and Deformations
X_1 = data4(:,4);
Y_1 = data4(:,5);
X_2 = data4(:,7);
Y_2 = data4(:,8);
X_3 = data4(:,9);
Y_3 = data4(:,10);
X_4 = data4(:,11);
Y_4 = data4(:,12);
X_5 = data4(:,2);
Y_5 = data4(:,3);
MZ  = data4(:,6);

% Elemental deformation
% load step 1
Xdisp1 = X0e + X_1;
Ydisp1 = Y0e + Y_1;
% load step 2
Xdisp2 = X0e + X_2;
Ydisp2 = Y0e + Y_2;
% load step 3
Xdisp3 = X0e + X_3;
Ydisp3 = Y0e + Y_3;
% load step 4
Xdisp4 = X0e + X_4;
Ydisp4 = Y0e + Y_4;
% load step 5
Xdisp5 = X0e + X_5;
Ydisp5 = Y0e + Y_5;


%% STRUCTURE STEP 5 DATA FOR ANIMATION

```

```
MZ_5     = data7;
X_5all   = data9;
Y_5all   = data10;
Xpos     = X0e+X_5all;
Ypos     = Y0e+Y_5all;
Stress1 = data8;
Ene1     = data5;
Ene2     = data6;
Ene3     = data6-data5(:,end);
ForceX   = data11;
ForceY   = data12;
RotZ     = data13;
Kurv     = data14;

parvector = [thickness len maxw+minw];


%% SAVE AND EXPORT DATA

% elemental data
E1output.Xpos  = {Xpos};
E1output.Ypos  = {Ypos};
E1output.Stress = {Stress1};
E1output.Moment = {MZ_5};
E1output.Energy = {Ene2};
E1output.ForceX = {ForceX};
E1output.ForceY = {ForceY};
E1output.RotZ  = {RotZ};
E1output.Kurv  = {Kurv};
E1output.Shape = {shape};

% nodal data
E1output.RotN1  = {RotN1};
E1output.RotN2  = {RotN2};
E1output.Fx1node = {FX1};
E1output.Fy1node = {FY1};
E1output.M1node = {MN1};
E1output.Fx2node = {FX2};
E1output.Fy2node = {FY2};
E1output.M2node = {MN2};

% parameter data
E1output.parameters = {parvector};



save('E1_output.mat','E1output');

%% Open the GUI and save file *** turn off when running optimization***
%      A7_LSWGUI;
%      hGuiFig = findobj('Tag','Guifig1','Type','figure');          %find figure
%      handles = guidata(hGuiFig);                                  %get handles
%      A7_LSWGUI('pushbutton1_Callback',handles.pushbutton1,[],handles); %push plot
%      A7_LSWGUI('pushbutton2_Callback',handles.pushbutton2,[],handles); %push next
%      A7_LSWGUI('Save_Callback',handles.Save,[],handles);          %push save
%      close(A7_LSWGUI)
%
%% Save new data to larger struct
E2  = {E1output};
if exist('E2_output.mat','file')
```

```
fin = load('E2_output.mat');
fin.E2(end+1,:) = E2;
E2              = fin.E2;
end
save('E2_output.mat','E2');

%% Save coordinates of spring to file *** turn off when running optimization ***

%      Springcoor = [Xpos(:,1) Ypos(:,1) zeros(length(Xpos(:,1)),1)]';
%
%
%      % nodal coordinates txtfile
%
%     fid_out = fopen(cadcoorspring, 'w');
%         for k = 1:length(Xpos(:,1))
%             fprintf(fid_out, '%10.8f %10.8f %10.8f \r\n',
    Springcoor(1,k),Springcoor(2,k),Springcoor(3,k)) ;
%         end
%     fclose(fid_out);

%% calculate closest distance to next spring
%
%     Xpos2 = Xpos;
%     Ypos2 = Ypos+offs;
%
%     for iv = 1:length(Xpos)
%
%         for iii = 1:length(Xpos)
%         vectorX = Xpos(iv,:) - Xpos2(iii,:);
%         vectorY = Ypos(iv,:) - Ypos2(iii,:);
%         dist(iii,:) = sqrt(vectorX.^2+vectorY.^2);
%         end
%         mindist1(iv) = min(dist(:));
%         dist = [];
%
%     end
%      mindist = min(mindist1(:));
%      surfdist = mindist-thickness;


end
%% Print Errors
type 'fileafs.err'
```

C.6. FILE 01_02

```
%% FIXED PARAMETERS

% Finite elements
steps     = 50;              % nr of load steps
inc       = 100;              % nr of lines (odd) % 81 minimum amount of elements
nelem     = 1;                % nr of elements per line
nnodes    = nelem*inc+1;      % total nr of nodes
telem     = nnodes-1;         % total nr of elements

% Geometry
thickness = 0.000200;       % [m] thickness of beam
totlen    = 0.15;           % [m] resulting length after prestress
```

```
ef        = 0.4;           % [*100%] prestress factor
len       = totlen/(1-ef); % [m] initial length of beam
maxw      = 1.2*0.0125;    % [m] maximum width = 0.6 m
minw      = 1.2*0.0375;    % [m] minimum width = 0.3 m
offs      = 0.01;          % [m] offset for next stacked spring

% Loads
preload  = len-totlen;   % [m]  distance of prestress
rot       = 1 ;          % [rad] endpoints rotation     % change for final angle
prerot1  = 0.8;
prerot2  = 0.1;          % initial position rigth node
prerot3  = 0.1;          % intiial position left node
prerot4  = 1;            % rotate to start position     % change of initial angle
loadf     = 0.01;        % [Nm] small pertubation load

% Steel Material RVS 1.4310
E         = 200*10^9 ;   % [Pa] Young's modulus
nu        = 0.29;        % [ ]Poisson ratio
rho       = 7800;        % [kg/m^3] Density
sigma     = 1100e6;      % MPa

%Constants
g         = 9.81;

% Crossection
section  = len/mode;
incr      = round(inc/(mode/2+1));
```

C.7. FILE 01_03

```
function varargout = A7_LSWGUI(varargin)
% A7_LSWGUI MATLAB code for A7_LSWGUI.fig
%      A7_LSWGUI, by itself, creates a new A7_LSWGUI or raises the existing
%      singleton*.
%
%      H = A7_LSWGUI returns the handle to a new A7_LSWGUI or the handle to
%      the existing singleton*.
%
%      A7_LSWGUI('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in A7_LSWGUI.M with the given input arguments.
%
%      A7_LSWGUI('Property','Value',...) creates a new A7_LSWGUI or raises the
%      existing singleton*. Starting from the left, property value pairs are
%      applied to the GUI before A7_LSWGUI_OpeningFcn gets called. An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to A7_LSWGUI_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help A7_LSWGUI

% Last Modified by GUIDE v2.5 07-Jun-2019 11:35:04

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
```

```
gui_State = struct('gui_Name',    mfilename, ...
'gui_Singleton', gui_Singleton, ...
'gui_OpeningFcn', @A7_LSWGUI_OpeningFcn, ...
'gui_OutputFcn', @A7_LSWGUI_OutputFcn, ...
'gui_LayoutFcn', [] , ...
'gui_Callback', []);
if nargin && ischar(varargin{1})
gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
[varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before A7_LSWGUI is made visible.
function A7_LSWGUI_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject   handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)
% varargin  command line arguments to A7_LSWGUI (see VARARGIN)

% Choose default command line output for A7_LSWGUI
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes A7_LSWGUI wait for user response (see UIRESUME)
% uiwait(handles.Guifig1);

% Create the data to plot.
%     run('A1_LSW.m');

%     assignin('base','handles',handles)
% assignin('base','hObject',hObject)

% Axes onzichtbaar maken bij openen
axes(handles.axes_position)
set(gca, 'visible', 'off');
axes(handles.axes_moment)
set(gca, 'visible', 'off');
axes(handles.axes_stress)
set(gca, 'visible', 'off');
axes(handles.axes_energy)
set(gca, 'visible', 'off');
axes(handles.axes_shape)
set(gca, 'visible', 'off');
axes(handles.axes_sumenergy)
set(gca, 'visible', 'off');
axes(handles.axes_resultmoment)
set(gca, 'visible', 'off');
axes(handles.axes_forcex)
set(gca, 'visible', 'off');
axes(handles.axes_sumforcex)
set(gca, 'visible', 'off');
```

```
axes(handles.axes_forcey)
set(gca, 'visible', 'off');
axes(handles.axes_sumforcey)
set(gca, 'visible', 'off');
axes(handles.axes_rotation)
set(gca, 'visible', 'off');
axes(handles.axes_pgram)
set(gca, 'visible', 'off');
axes(handles.axes_error)
set(gca, 'visible', 'off');
axes(handles.axes_curvature)
set(gca, 'visible', 'off');
axes(handles.axes_resultkurv)
set(gca, 'visible', 'off');

% --- Outputs from this function are returned to the command line.
function varargout = A7_LSWGUI_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject   handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject   handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

run('A8_LSW_globals')
run('A4_LSW_parameters.m');
load('C2_shapedata.txt');
load('D1_anpar.txt');
load('E1_output.mat');
%    load('E2_output.mat');
%    E1output = cell2mat(E2([83]));
%_________________________________________
Stress     = cell2mat(E1output.Stress);
Moment     = cell2mat(E1output.Moment);
Energy     = cell2mat(E1output.Energy);
Xpos       = cell2mat(E1output.Xpos);
Ypos       = cell2mat(E1output.Ypos);
Shape      = cell2mat(E1output.Shape);
ForceX     = cell2mat(E1output.ForceX);
ForceY     = cell2mat(E1output.ForceY);
Steps      = D1_anpar(8);
RotZ       = cell2mat(E1output.RotZ);
Kurv       = cell2mat(E1output.Kurv);

% nodal data
RotN1      = cell2mat(E1output.RotN1);
RotN2      = cell2mat(E1output.RotN2);
Fx1node    =cell2mat(E1output.Fx1node);
Fy1node    =cell2mat(E1output.Fy1node);
M1node     =cell2mat(E1output.M1node);
Fx2node    =cell2mat(E1output.Fx2node);
Fy2node    =cell2mat(E1output.Fy2node);
```

```
M2node      =cell2mat(E1output.M2node);

%_________________________________________


r1          = RotZ(1,:);
r2          = RotZ(end,:);
Rotation    = (RotZ(1,:));
angleoff    = mean(r2-r1);
Elements    = length(Xpos);

for i = 1:Steps
Sumenergy(i) = sum(Energy(:,i));
Summoment(i) = Moment(1,i)-Moment(end,i);
Sumkurv(i)  = Kurv(1,i) -Kurv(end,i);
Sumforcex(i) = ForceX(1,i)-ForceX(end,i);
Sumforcey(i) = ForceY(1,i)-ForceY(end,i);

%         StrainEnergy(:,i) = (Energy(:,i));
end


for i = 1:Steps
StrainEnergy(:,i) = 100*(Energy(:,i)/Sumenergy(end));
end

Summomentnodes = abs(M1node+M2node);
Summomentnodes2 = -(M1node+M2node);

% Convert shear force and axial force to global coordinate system
FY1 = ForceY(1,:).*cos(r1)+ForceX(1,:).*sin(r1);
FX1 = ForceX(1,:).*cos(-r1)+ForceY(1,:).*sin(-r1);

FY2 = ForceY(end,:).*cos(r1)+ForceX(end,:).*sin(r1);
FX2 = ForceX(end,:).*cos(-r1)+ForceY(end,:).*sin(-r1);

% Calculate Error
Mobj = max(Summomentnodes2)*sin(RotN1+pi/2);

for i = 1:Steps+1
Err(i)  = 100*(Summomentnodes(i)-Mobj(i))/Mobj(i);
SE(i)   = ((Summomentnodes(i)-Mobj(i))/Mobj(i))^2;
end

newint  = linspace(-RotN1(1),-RotN1(end),100);
Err_int = interp1(-RotN1,Err,newint);
SE_int  = interp1(-RotN1,SE,newint);


% Perfomance Units
Perf_moment = max(Summomentnodes(:))/max(Moment(1,:));
Perf_error = sqrt(sum(SE_int)/length(SE_int));
%     Perf_error2 = sqrt(sum(SE)/length(SE));
Perf_total = Perf_moment*(1-Perf_error);

% Standard units
Smin    = abs(min(Stress(:)));
Smax    = max(Stress(:));
Sabsreal= max([Smax Smin])/10^6;
Sabs    = 2000000000;
```

```
next    = 0;
nextone = 1;
spacing = 0.010;
spacing2 = 0.010;

assignin('base','Shape',Shape)
assignin('base','Kurv',Kurv)
assignin('base','Err_int',Err_int)
assignin('base','SE',SE)
assignin('base','SE_int',SE_int)
assignin('base','dist',spacing)
assignin('base','Err',Err)
assignin('base','Steps',Steps)
assignin('base','FX2',FX2)
assignin('base','FY2',FY2)
assignin('base','FX1',FX1)
assignin('base','FY1',FY1)
assignin('base','r1',r1)
assignin('base','r2',r2)
assignin('base','RotN1',RotN1)
assignin('base','RotN2',RotN2)
assignin('base','RotZ',RotZ)
assignin('base','Rotation',Rotation)
assignin('base','next',next)
assignin('base','nextone',nextone)
assignin('base','Stress',Stress)
assignin('base','Moment',Moment)
assignin('base','Energy',Energy)
assignin('base','StrainEnergy',StrainEnergy)
assignin('base','Xpos',Xpos)
assignin('base','Ypos',Ypos)
assignin('base','Shape',Shape)
assignin('base','Sumenergy',Sumenergy)
assignin('base','Summoment',Summoment)
assignin('base','Summomentnodes',Summomentnodes)
assignin('base','Summomentnodes2',Summomentnodes2)
assignin('base','ForceX',ForceX)
assignin('base','ForceY',ForceY)
assignin('base','Sumforcex',Sumforcex)
assignin('base','Sumforcey',Sumforcey)
assignin('base','Elements',Elements)

assignin('base','Fx1node',Fx1node)
assignin('base','Fy1node',Fy1node)
assignin('base','M1node',M1node)
assignin('base','Fx2node',Fx2node)
assignin('base','Fy2node',Fy2node)
assignin('base','M2node',M2node)


axes(handles.axes_position)
set(gca, 'visible', 'on');
axes(handles.axes_moment)
set(gca, 'visible', 'on');
axes(handles.axes_stress)
set(gca, 'visible', 'on');
axes(handles.axes_energy)
set(gca, 'visible', 'on');
axes(handles.axes_shape)
set(gca, 'visible', 'on');
```

```
axes(handles.axes_sumenergy)
set(gca, 'visible', 'on');
axes(handles.axes_resultmoment)
set(gca, 'visible', 'on');
axes(handles.axes_forcex)
set(gca, 'visible', 'on');
axes(handles.axes_sumforcex)
set(gca, 'visible', 'on');
axes(handles.axes_forcey)
set(gca, 'visible', 'on');
axes(handles.axes_sumforcey)
set(gca, 'visible', 'on');
axes(handles.axes_rotation)
set(gca, 'visible', 'on');
axes(handles.axes_pgram)
set(gca, 'visible', 'on');
axes(handles.axes_error)
set(gca, 'visible', 'on');
axes(handles.axes_curvature)
set(gca, 'visible', 'on');
axes(handles.axes_resultkurv)
set(gca, 'visible', 'on');

limx   = 1.5;       %rotation limit
elimx = Elements; %element limit
mlimy = 0.3;      %moment
flimy = 20;       %force
elimy = 1;        %energy
selimy = 0.020;  %strain energy per element limit
strlim = 2000*10^6;

stringformat = 4;
arm    = 0.295;


% graph 1
axes(handles.axes_position)
base1     = plot(Xpos,Ypos,'color',[0,0,0]+0.5); hold on
animatie1 = plot(Xpos,Ypos,'color',[0,0,0]+0.5);
%                animatie1c =
    plot(Xpos+spacing*sin(Rotation(1)),Ypos+spacing*cos(Rotation(1)),'color',[0,0,0]+1);
%                animatie1e =
    plot(Xpos*spacing*sin(Rotation(1)),Ypos-spacing*cos(Rotation(1)),'color',[0,0,0]+1);
%                xlabel('x coordinate [m]')
ylabel('y coordinate [m]')
%                ylim([-0.01 0.06])
title(['Coordinates | offset:',num2str(angleoff,1),' rad | ef: ', num2str(ef,2)])
hold on;
axis equal
%                xlim([0 totlen]);


% graph 2
axes(handles.axes_moment)
base2     = plot(Moment,'color',[0,0,0]+0.5); hold on
zero1     = plot(zeros(length(Stress),1),'k');
animatie2 = plot(Moment,'color',[0,0,0]+0.5);
ylabel('Moment [Nm]');
title('Moment');
ylim([-0.6 0.6]);
```

```
xlim([1 Elements]);
grid on


% graph 3
axes(handles.axes_stress)
base3      = plot(Stress,'color',[0,0,0]+0.5); hold on
zero1      = plot(zeros(length(Stress),1),'k');
animatie3  = plot(Stress,'color',[0,0,0]+0.5);
xlabel('element nr [-]')
ylabel('Stress [MPa]')
title(['Stress | Max: ', num2str(Sabsreal,stringformat),' MPa' ])
grid on
ylim([-strlim strlim])
xlim([1 Elements]);

% graph 4
axes(handles.axes_energy)
base4      = plot(StrainEnergy,'color',[0,0,0]+0.5); hold on
animatie4  = plot(StrainEnergy,'color',[0,0,0]+0.5);
xlabel('element nr [-]')
ylabel('Elemental Energy / tot Energy [%]')
title('Strain Energy | ref: 100 elem')
%                ylim([0 4]);
xlim([1 Elements]);
grid on

% graph 5
axes(handles.axes_sumenergy)
base5      = plot(-Rotation,Sumenergy,'color',[0,0,0]+0.5); hold on
animatie5  = plot(-Rotation,Sumenergy,'color',[0,0,0]+0.5);
line5a     = line([next next],[0 1],'color','red','LineStyle','-');
xlabel('Rotation [rad]')
ylabel('Strain Energy [J]')
title(['Total Potential Energy | Max: ', num2str(max(Sumenergy(:)),stringformat),' J'
    ])
ylim([min(Sumenergy(:))-0.1 max(Sumenergy(:))+0.1])
xlim([-limx limx])
grid on

% graph 6
axes(handles.axes_resultmoment)
base6      = plot(-RotN1,Summomentnodes2,'color',[0,0,0]+0.5); hold on
line6a     = line([next next],[-mlimy mlimy],'color','red','LineStyle','-');
animatie6  = plot(-RotN1,Summomentnodes2,'color',[0,0,0]+0.5);
%                animatie6  = plot(-Rotation,Summoment,'color',[0,0,0]+0.5);
%                animatie6b = plot(-Rotation,Moment(1,:));
%                animatie6c = plot(-Rotation,-Moment(end,:));
%                summoment1 = plot(-Rotation,(FY1+FY2)/2*totlen);
objective  = plot(-RotN1,Mobj);

%                nodemoment1  = plot(-Rotation,M1node);
%                nodemoment2  = plot(-Rotation,M2node);
ylabel('Moment [Nm]')
title(['Moment | Max: ',num2str(max(Summoment(:)),stringformat),' Nm'])
ylim([-0.2 0.2])
xlim([-limx limx])
grid on
%              text(-0.9,0.2,{'Mmax =' num2str(max(Summoment))})
%                legend('Eresult','Nresult','live');
```

```
% graph 7
axes(handles.axes_forcex)
base7      = plot(ForceX,'color',[0,0,0]+0.5); hold on
animatie7 = plot(ForceX,'color',[0,0,0]+0.5);
ylabel('Force [N]')
title('Axial Force (AF)')
xlim([1 Elements]);
ylim([-flimy 0])
grid on

% graph 7b
axes(handles.axes_forcey)
base7b      = plot(ForceY,'color',[0,0,0]+0.5); hold on
animatie7b = plot(ForceY,'color',[0,0,0]+0.5);
ylabel('Force [N]')
title('Shear Force (SF)')
xlim([1 Elements]);
ylim([-flimy flimy])
grid on


% graph 8
axes(handles.axes_sumforcex)
base8 = plot(-Rotation,Sumforcex,'color',[0,0,0]+0.5); hold on
animatie8 = plot(-Rotation,Sumforcex,'color',[0,0,0]+0.5);
%               base8    = plot(-Rotation,FX1,'color',[0,0,0]+0.5); hold on;
animatie81 = plot(-Rotation,ForceX(1,:));
animatie82 = plot(-Rotation,-ForceX(end,:));
line8a      = line([next next],[-max(abs(ForceX(:)))
    max(abs(ForceX(:)))],'color','red','LineStyle','-');
%               globalfx1    = plot(-Rotation,FX1);
%               globalfx2    = plot(-Rotation,FX2);
ylabel('Force [N]')
title(['Endpoint AF| Max: ', num2str(max(ForceX(:)),stringformat),' N' ])
grid on
ylim([-15 15])
xlim([-limx limx])

% graph 8b
axes(handles.axes_sumforcey)
base8b = plot(-Rotation,Sumforcey,'color',[0,0,0]+0.5); hold on
animatie8b = plot(-Rotation,Sumforcey,'color',[0,0,0]+0.5);
%               base8b      = plot(-Rotation,FY1,'color',[0,0,0]+0.5); hold on;
animatie83 = plot(-Rotation,ForceY(1,:));
animatie84 = plot(-Rotation,-ForceY(end,:));
line8b      = line([next next],[min(ForceY(:))
    max(ForceY(:))],'color','red','LineStyle','-');
%               globalfy1  = plot(-Rotation,FY1);
%               globalfy2  = plot(-Rotation,FY2);
ylabel('Force [N]')
title(['Endpoint SF | Max: ', num2str(max(ForceY(:)),stringformat),' N' ])
grid on
ylim([-10 10])
xlim([-limx limx])


global sequence
sequence = 1:1:length(RotN1);
```

```
% graph 9
axes(handles.axes_shape)
for i = 1:length(Shape)
animatie9a = line([i i],[0 0.5*Shape(i)]);
animatie9b = line([i i],[0 -0.5*Shape(i)]);
end
colormap(jet(256));
title(['Width | SF: ',num2str(Shape(1)/Shape(end),3)]);
h = colorbar('westoutside');
h.YAxisLocation='left';
title(h, 'Stress [MPa]')
h.Ticks = linspace(0, 1, 5) ;                    %Create ticks from zero to 1
h.TickLabels = num2cell([0 0.25 0.5 0.75 1]*Sabs*10^-6) ;
xlim([1 Elements]);
ylim([-0.1 0.1])

% graph 10
axes(handles.axes_rotation)
base10     = plot(RotZ,'color',[0,0,0]+0.5); hold on
animatie10 = plot(RotZ,'color',[0,0,0]+0.5); hold on
limit1 = line([0 Elements],[pi/3 pi/3]);
limit2 = line([0 Elements],[-pi/3 -pi/3]);
title(['Elemental Z Rotation | Max: ', num2str(max(RotZ(:)),stringformat),' rad' ]);
ylabel('rotation [rad]');
ylim([-pi/2 pi/2])
xlim([1 Elements]);




% graph 11
axes(handles.axes_pgram)
pgram     = plot([0 0 arm*cos(Rotation(1)) arm*cos(Rotation(1)) 0],...
[0 totlen totlen+arm*sin(Rotation(1)) arm*sin(Rotation(1)) 0 ]); hold on;
%               pgramveer1 = plot(-Ypos,Xpos,'color',[0,0,0]+0.5);
pgramveer2 =
    plot(Ypos+0.1*cos(Rotation(1)),totlen-Xpos+0.1*sin(Rotation(1)),'color',[0,0,0]+0.5);
%               pgramveer3 =
    plot(Ypos+(0.1+1*spacing2)*cos(Rotation(1)),totlen-Xpos+(0.1+1*spacing2)*sin(Rotation(1)),'color',
%               pgramveer4 =
    plot(Ypos+(0.1+2*spacing2)*cos(Rotation(1)),totlen-Xpos+(0.1+2*spacing2)*sin(Rotation(1)),'color',
%               pgramveer5 =
    plot(-Ypos+(0.1+3*spacing)*cos(Rotation(1)),Xpos+(0.1+3*spacing)*sin(Rotation(1)),'color',[0,0,0]+(
ylabel('y coordinate [m]')
ylim([-0.10 0.25])
title('Coordinates')
hold on;
xlim([-0.05 0.15]);
axis equal


% graph 12
axes(handles.axes_error)
base12     = plot(-Rotation,Err); hold on;
animatie12 = plot(-Rotation,Err);
line12     = line([-1 -1],[-40 10],'color','red','LineStyle','-');
title(['Error | RMSE: ',num2str(Perf_error,3)]);
ylabel('Error [%]');
xlabel('Rotation [rad]');
xlim([-limx limx]);
```

```
ylim([-40 10])

% graph 13
axes(handles.axes_curvature)
base13     = plot(abs(Kurv),'color',[0,0,0]+0.5); hold on
animatie13 = plot(abs(Kurv),'color',[0,0,0]+0.5); hold on
title(['Curvature | Max: ', num2str(max(abs(Kurv(:))),stringformat),' [m^{-1}]' ]);
ylabel('[Curvature [m^{-1}]');
ylim([0 70])
xlim([1 Elements]);
grid on



% graph 14
axes(handles.axes_resultkurv)
base14       = plot(-Rotation,Sumkurv,'color',[0,0,0]+0.5); hold on
animatie14   = plot(-Rotation,Sumkurv,'color',[0,0,0]+0.5);
animatie14b = plot(-Rotation,Kurv(1,:));
animatie14c = plot(-Rotation,-Kurv(end,:));
line14       = line([next next],[-max(abs(Kurv(:)))
    max(abs(Kurv(:)))],'color','red','LineStyle','-');
ylabel('Curvature [Nm]')
title(['Curvature | Max: ',num2str(max(Sumkurv(:)),stringformat)])
ylim([-25 25])
xlim([-limx limx])
grid on


% graph 15
axes(handles.axes_forcesGC)
%                base8    = plot(-Rotation,FX1,'color',[0,0,0]+0.5); hold on;
%                animatie81 = plot(-Rotation,ForceX(1,:));
%                animatie82 = plot(-Rotation,-ForceX(end,:));
%                line8a      = line([next next],[-max(abs(ForceX(:)))
    max(abs(ForceX(:)))],'color','red','LineStyle','-');
globalfy1   = plot(-Rotation,FY1); hold on;
globalfy2   = plot(-Rotation,FY2);
nodeforce2  = plot(-Rotation,-Fy1node);
nodeforce4  = plot(-Rotation,Fy2node);
ylabel('Force [N]')
title(['Global Y Forces' ])
grid on
legend('eFY1','eFY2','nfy1','nfy2')
%                legend('FX1','FX2','FY1','FY2')
%                ylim([-flimy flimy])
%                xlim([-limx limx])

axes(handles.axes_forcesXGC)
globalfx1   = plot(-Rotation,-FX1); hold on;
globalfx2   = plot(-Rotation,-FX2);
nodeforce1  = plot(-Rotation,Fx1node);
nodeforce3  = plot(-Rotation,-Fx2node);
ylabel('Force [N]')
title(['Global X Forces' ])
grid on
legend('eFX1','eFX2','nfx1','nfx2')
% graph 16
% axes(handles.axes_forcesDIFF)
% %                base8    = plot(-Rotation,FX1,'color',[0,0,0]+0.5); hold on;
```

```
% %                 animatie81 = plot(-Rotation,ForceX(1,:));
% %                 animatie82 = plot(-Rotation,-ForceX(end,:));
% %                 line8a     = line([next next],[-max(abs(ForceX(:))) max(abs(ForceX(:)))],'color','red','LineStyle','-');
%                 localfxdiff = plot(-Rotation,ForceX(1,:)-ForceX(end,:));hold on;
%                 localfydiff = plot(-Rotation,ForceY(1,:)-ForceY(end,:));
%                 globalfxdiff   = plot(-Rotation,FX1-FX2);
%                 globalfydiff   = plot(-Rotation,FY1-FY2);
%                 globalfxdiffnodes  = plot(-Rotation,Fx1node+Fx2node,'+-');
%                 globalfydiffnodes  = plot(-Rotation,Fy1node+Fy2node);
%
%    legend('localFx','localFy','Globalfx','Globalfy','globalnodex','globalnodey');
% %                 xlim([-limx limx])
%

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

run('A8_LSW_globals');

next     = next + nextone;
perc     = abs(Stress(:,next))/Sabs;
new      = ceil(perc*256);
scaling = jet(256);

assignin('base','next',next)
assignin('base','scaling',scaling)
assignin('base','new',new)
assignin('base','perc',perc)

% graph 1
axes(handles.axes_position)
set(animatie1, 'Xdata', Xpos(:,next), 'Ydata', Ypos(:,next),'color','r','LineWidth',2); hold on;
set(animatie1c, 'Xdata', Xpos(:,next)+0.010*sin(Rotation(next)), 'Ydata', Ypos(:,next)+0.010*cos(Rotation(next)),'color','b','LineWidth',2);
set(animatie1e, 'Xdata', Xpos(:,next)-0.010*sin(Rotation(next)), 'Ydata', Ypos(:,next)-0.010*cos(Rotation(next)),'color','b','LineWidth',2);

% graph 2
axes(handles.axes_moment)
set(animatie2, 'Ydata', Moment(:,next),'color','r','LineWidth',2);

% graph 3
axes(handles.axes_stress)
set(animatie3, 'Ydata', Stress(:,next),'color','r','LineWidth',2);

% graph 4
axes(handles.axes_energy)
set(animatie4, 'Ydata', StrainEnergy(:,next),'color','r','LineWidth',2);

% graph 5
axes(handles.axes_sumenergy)
set(animatie5,'XData',-Rotation(next),'YData',Sumenergy(next),'color','r','Marker','square','MarkerFace
set(line5a, 'Xdata',[-Rotation(next) -Rotation(next)],'Ydata',[-max(Sumenergy(:))-0.1 max(Sumenergy(:))+0.1],'LineWidth',0.3);
```

```
% graph 6
axes(handles.axes_resultmoment)
set(animatie6, 'Ydata',
    Summomentnodes2(next),'XData',-Rotation(next),'color','r','Marker','square','MarkerFaceColor','r');
set(line6a, 'Xdata',[-Rotation(next) -Rotation(next)],'Ydata',[-mlimy
    mlimy],'LineWidth',0.3);

% graph 7
axes(handles.axes_forcex)
set(animatie7, 'Ydata', ForceX(:,next),'color','r','LineWidth',2);


% graph 7b
axes(handles.axes_forcey)
set(animatie7b, 'Ydata', ForceY(:,next),'color','r','LineWidth',2);


% graph 8
axes(handles.axes_sumforcex)
set(animatie8, 'Ydata',
    Sumforcex(next),'XData',-Rotation(next),'color','r','Marker','square','MarkerFaceColor','r');
set(line8a, 'Xdata',[-Rotation(next) -Rotation(next)],'Ydata',[-flimy
    flimy],'LineWidth',0.3);

% graph 8b
axes(handles.axes_sumforcey)
set(animatie8b, 'Ydata',
    Sumforcey(next),'XData',-Rotation(next),'color','r','Marker','square','MarkerFaceColor','r');
set(line8b, 'Xdata',[-Rotation(next) -Rotation(next)],'Ydata',[-flimy
    flimy],'LineWidth',0.3);

% % graph 9
axes(handles.axes_shape)
for i = 1:length(Shape)
animatie9a = line([i i],[0 0.5*Shape(i)],'color',scaling(new(i),:),'LineWidth',3);
    hold on
animatie9b = line([i i],[0 -0.5*Shape(i)],'color',scaling(new(i),:),'LineWidth',3);
    hold on
end

% graph 10
axes(handles.axes_rotation)
set(animatie10,'Ydata', RotZ(:,next),'color','r','LineWidth',2); hold on;


% graph 11
axes(handles.axes_pgram)
set(pgram,'Ydata', [0 totlen totlen+arm*sin(Rotation(next)) arm*sin(Rotation(next)) 0
    ],...
'Xdata', [0 0 arm*cos(Rotation(next)) arm*cos(Rotation(next)) 0 ],...
'color','k','LineWidth',2); hold on;
%                set(pgramveer1,'Xdata', -Ypos(:,next), 'Ydata',
    Xpos(:,next),'color','r','LineWidth',2); hold on;
set(pgramveer2,'Xdata', Ypos(:,next)+0.1*cos(Rotation(next))  , 'Ydata',
    totlen-Xpos(:,next)+0.1*sin(Rotation(next)),'color','r','LineWidth',2); hold on;
%                set(pgramveer3,'Xdata',
    Ypos(:,next)+(0.1+spacing2)*cos(Rotation(next)), 'Ydata',
    totlen-Xpos(:,next)+(0.1+spacing2)*sin(Rotation(next)),'color','r','LineWidth',2);
    hold on;
```

```
%               set(pgramveer4,'Xdata',
    Ypos(:,next)+(0.1+2*spacing2)*cos(Rotation(next)), 'Ydata',
    totlen-Xpos(:,next)+(0.1+2*spacing2)*sin(Rotation(next)),'color','r','LineWidth',2);
    hold on;
%               set(pgramveer5,'Xdata',
    -Ypos(:,next)+(0.1+3*spacing2)*cos(Rotation(next)), 'Ydata',
    Xpos(:,next)+(0.1+3*spacing)*sin(Rotation(next)),'color','r','LineWidth',2); hold
    on;


% graph 12
axes(handles.axes_error)
set(animatie12,'Xdata',-Rotation(next),'Ydata',
    Err(next),'color','r','Marker','square','MarkerFaceColor','r');hold on;
set(line12, 'Xdata',[-Rotation(next) -Rotation(next)],'Ydata',[-40
    10],'LineWidth',0.3);

% graph 13
axes(handles.axes_error)
set(animatie13, 'Ydata', abs(Kurv(:,next)),'color','r','LineWidth',2);

% graph 14
axes(handles.axes_resultkurv)
set(animatie14, 'Ydata',
    Sumkurv(:,next),'XData',-Rotation(next),'color','r','Marker','square','MarkerFaceColor','r');
set(line14, 'Xdata',[-Rotation(next) -Rotation(next)],'Ydata',[-20
    20],'LineWidth',0.3);

% DRAW NEW POINTS
drawnow

% go to next point
if next == Steps && nextone == 1
next = 0;
end

if nextone == -1 && next == 1
next = Steps;
end



% --- Executes on button press in Reverse.
function Reverse_Callback(hObject, eventdata, handles)
% hObject   handle to Reverse (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)


global nextone
nextone = nextone*-1;


% --- Executes on button press in Save.
function Save_Callback(hObject, eventdata, handles)
% hObject   handle to Save (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)
ctime = (datetime('now'));
ctime.Format = 'yyMMdd_HHmmss';
```

```
ctime = char(ctime);
plotname = strcat(ctime,'_LSWgui');
fig = gcf;
fig.PaperPositionMode = 'auto';

saveas(fig,plotname,'svg')
```

C.8. FILE 01_04

```
%% all globals

global elementcontrol compression
global E1output Stress1 MZ_5 Ene3 Ene2 rotN1 rotN2
global trueshape1 trueshape2 Shape
global par1 par2 par3 par4 par5 par6
global scaling new perc Smax Sabs
global next nextone Steps spacing spacing2
global limx mlimy flimy elimy selimy
global totlen arm Rotation sequence stringformat Elements
global t1 t2 t3 t4 t5 t6 q
global animatie1 Xpos Ypos
global animatie1b
global animatie1c
global animatie1d
global animatie1e
global animatie2 Moment moment1 moment2
global animatie3 Stress
global animatie4 Energy StrainEnergy
global animatie5 Sumenergy line5a
global animatie6 Summoment line6a Summomentnodes Summomentnodes2
global animatie7 ForceX
global animatie7b ForceY
global animatie8 line8a Sumforcex globalfx1 FX1
global animatie8b line8b Sumforcey globalfy1 FY1
global animatie9a
global animatie9b Shape RotN1 RotN2
global animatie10 RotZ base10a line10
global animatie12 Err line12
global animatie13 Kurv
global animatie14 line14 Sumkurv
global M1node M2node Fx1node Fx2node Fy1node Fy2node
global pgram pgramveer1 pgramveer2 pgramveer3 pgramveer4 pgramveer5
```

C.9. FILE 02_01

```
function [mo, di, pm1, pm2] = A14_plotmeasurements(fileleft, fileright, radius, mass,
    g, weight,color)



m.left =  fileleft;
m.right = fileright;

%distances
di.mleft   = m.left(:,1);
```

```
di.mright  = m.right(:,1);

di.mleft   = flip(di.mleft);
di.mright  = -di.mright;

%forces
fo.mleft   = m.left(:,2);
fo.mright  = m.right(:,2);

fo.mleft2  = interp1(di.mleft,fo.mleft,di.mright); %interpolated data
fo.mdiff   = - fo.mleft2 + fo.mright;

%plot absolute measurments
% figure
% plot(di.mleft,fo.mleft); hold on
% plot(di.mright,fo.mright); hold on;
% plot(di.mright,fo.mdiff); hold on;
% legend('left','right','difference')

% moments minus weight
mo.mleft = (fo.mleft2-weight)*radius;
mo.mright = (fo.mright-weight)*radius;

for i = 1:length(mo.mleft)
mo.mean(i)  = mean([mo.mleft(i) mo.mright(i)]);
end


%plot real and comparison
% figure;
pm1 = plot(10.^-6.*di.mright./radius,mo.mleft,color); hold on
pm2 = plot(10.^-6.*di.mright./radius,mo.mright,color); hold on;
% plot(10.^-6.*di.mright./radius,mo.mean','g');
% legend('ANSYS','Measurements','mean')
xlabel('Rotation [rad]');
ylabel('Moment [Nm]');
grid on;

end
```

D

APPENDIX D - ANSYS APDL CODE

The code presented in this appendix is used to model a 188 Bernoulli beam for large deflections. The code is run using the MATLAB script from appendix C. More information about the model is presented in appendix A.

```

!___________________________________
FINISH
/CLEAR,START
/FILNAME,fileafs,1

!___________________________________
!setparameters

*USE, 'C1_parameters.macro'

!___________________________________
!load shape data
*DIM,crshape,ARRAY,inc,1
*VREAD, crshape(1,1),C2_shapedata,txt,,IJK,inc,1
(1F8.4)

!___________________________________
/PREP7
!element selection
ET, 1, BEAM188
ET, 2, BEAM188

!___________________________________
!define crosssections
*DO, i, 1, inc
SECTYPE,  i, BEAM, RECT, , 0
SECOFFSET, CENT
SECDATA, thickness, crshape(i,1) !thickness,width
*ENDDO

!___________________________________
! Material properties
MPTEMP, 1   , 0
MPDATA, EX  , 1, , Ey
MPDATA, PRXY , 1, , nu
MPDATA, DENS , 1, , rho

```

```
!__________________________________
!Define keypoints
K,1,0,0
*DO,i,1,inc
K, i+1   ,(i/inc)*len  ,0
*ENDDO

!__________________________________
!Define Lines, index on sections
*DO,i,1,inc
L,i,i+1
*ENDDO

!__________________________________
! Meshing
TYPE,1

*DO,i,1,inc
SECNUM,i
LSEL,S,LINE, ,i
LESIZE,ALL, , ,nelem
LMESH,ALL
*ENDDO

!__________________________________
!get node number under keypoints 1 and inc+1
!then we can put bc on nodes instead of kp
KSEL,S,KP,,1
NSLK,S
*GET, ID_left,NODE,,NUM,MIN

KSEL,S,KP,,inc+1
NSLK,S
*GET, ID_right,NODE,,NUM,MIN

! Create Dummy node
N,500,0,0.11,0
N,501,0,0.1,0
TYPE,2
REAL,2
EN,500,500,501

!__________________________________
!BOUNDARY CONDITIONS
ALLSEL,ALL

!Constrain n1 n2
D,ID_left,ALL
D,ID_right,ALL

!Hinge n1 n2
DDELE,ID_right,ROTZ
DDELE,ID_left,ROTZ

!Free Ux n2
DDELE,ID_right,UX

!Imperfection End moment n1 n2
F,ID_left ,MZ,loadf
F,ID_right,MZ,loadf
```

```
!Uniform pressure on beam
!SFBEAM,ALL,2,PRES,1,1

!Dummy node contrain to hinged
D,500,ALL
DDELE,500,ROTZ

! Couple DOFS
CE, 1, 0.0, ID_left, ROTZ, 1, ID_right, ROTZ, -1,500,ROTZ,-1

/ESHAPE,1
/PBC,ALL,,2 !plot the BC, just to check
eplot

!_________________________________
/SOLU
ANTYPE, 0
NLGEOM,ON
OUTRES,ALL,ALL
NSUBST,steps,,steps


!Load step 1
TIME,1
D,500,ROTZ,%_FIX%          !Make n1 n2 dependent
LSWRITE, 1

!Load step 2
TIME,2
D,ID_right,UX,-preload       !load displacement
LSWRITE, 2

!Load step 3
TIME,3
D,ID_left,ROTZ,%_FIX%
FDELE,ALL,ALL              !Remove imperfection
LSWRITE, 3

! Load step 4: make independent
TIME,4
DDELE,500,ROTZ             !Make n1 n2 independent
D,ID_left ,ROTZ,%_FIX%        !Fix n1 (and n2) in current rotational position
D,ID_right ,ROTZ,%_FIX%
LSWRITE, 4

!Load step 5 !-->rotate right node
TIME,5
D,ID_right ,ROTZ,prerot2
LSWRITE, 5

!Load step 6 !-->rotate left node
TIME,6
D,ID_right ,ROTZ,%_FIX%      !Fix n1 (and n2) in current rotational position
D,ID_left ,ROTZ,prerot3
LSWRITE, 6

!Load step 7 !-->rotate both nodes backwards
TIME,7
!D,500,ROTZ,%_FIX%   !C      !Make n1 n2 dependent
```

```
D,ID_left ,ROTZ,prerot4       !Rotate n1 and n2
D,ID_right,ROTZ,prerot4
!DDELE,ID_right,ROTZ
LSWRITE, 7

LSSOLVE,1,7

!Load step 8
TIME,8
!ARCLEN, ON, 1
!DDELE,500,ROTZ               !Make n1 n2 independent
D,ID_left ,ROTZ,-rot          !Rotate n1 (and n2)
D,ID_right ,ROTZ,-rot         !Rotate n1 (and n2)
!DDELE,ID_right,ROTZ
!LSWRITE, 8
SOLVE






!__________________________________
/POST1

!SET,1
!PLDISP,1
!ANTIME,50,0.1,1,0,1,1,5
!/ANFILE, SAVE, LSW5,avi

! CREATE ELEMENT TABLE
*DIM,out_s,ARRAY,telem,12

SET,1
NSEL,ALL
*GET,nsteps1,ACTIVE,0,SET,SBST

etable,translationx1,U,X
*vget,out_s(1,4),elem,,etab,translationx1
etable,translationy1,U,Y
*vget,out_s(1,5),elem,,etab,translationy1

!__________________________________
! acquire locations of every node
*DIM, out_list, ARRAY, nnodes, 6
*VGET,out_list(1,1),NODE,1,LOC,X
*VGET,out_list(1,2),NODE,1,LOC,Y
*VGET,out_list(1,3),NODE,1,LOC,Z
*VGET,out_list(1,4),ELEM,1,CENT,X
*VGET,out_list(1,5),ELEM,1,CENT,Y
*VGET,out_list(1,6),ELEM,1,CENT,Z

!WRITE TABLE TO FILE
*MWRITE,out_list(1,1),D2_coordinates,txt,,JIK,6,1000,1
(6E15.6)


SET,2
NSEL,ALL
*GET,nsteps2,ACTIVE,0,SET,SBST
```

```

etable,translationx2,U,X
*vget,out_s(1,7),elem,,etab,translationx2
etable,translationy2,U,Y
*vget,out_s(1,8),elem,,etab,translationy2

SET,3
NSEL,ALL
*GET,nsteps3,ACTIVE,0,SET,SBST

etable,translationx3,U,X
*vget,out_s(1,9),elem,,etab,translationx3
etable,translationy3,U,Y
*vget,out_s(1,10),elem,,etab,translationy3

SET,4
NSEL,ALL
*GET,nsteps4,ACTIVE,0,SET,SBST


etable,translationx4,U,X
*vget,out_s(1,11),elem,,etab,translationx4
etable,translationy4,U,Y
*vget,out_s(1,12),elem,,etab,translationy4

SET,5
NSEL,ALL
*GET,nsteps5,ACTIVE,0,SET,SBST

SET,6
NSEL,ALL
*GET,nsteps6,ACTIVE,0,SET,SBST

SET,7
NSEL,ALL
*GET,nsteps7,ACTIVE,0,SET,SBST

SET,8
NSEL,ALL
*GET,nsteps8,ACTIVE,0,SET,SBST

! FILL ETABLE
etable,bendingstress,LS,1
*vget,out_s(1,1),elem,,etab,bendingstress ! ALL STRESSES
etable,translationx,U,X
*vget,out_s(1,2),elem,,etab,translationx ! X TRANSLATION
etable,translationy,U,Y
*vget,out_s(1,3),elem,,etab,translationy ! Y TRANSLATION
etable,momentz,M,Z
*vget,out_s(1,6),elem,,etab,momentz   ! Z MOMENT

!WRITE TABLE TO FILE
*MWRITE,out_s(1,1),D4_elementtable,txt,,JIK,40,1000,1
(40E15.6)

! READ DATA
ESEL,ALL

*DIM, out_ene1 , ARRAY, telem,nsteps2

```

```
*DIM, out_ene2 , ARRAY, telem,nsteps8
*DIM, out_momz1, ARRAY, telem,nsteps8
*DIM, out_str1 , ARRAY, telem,nsteps8
*DIM, out_tx  , ARRAY, telem,nsteps8
*DIM, out_ty  , ARRAY, telem,nsteps8
*DIM, out_fx  , ARRAY, telem,nsteps8
*DIM, out_fy  , ARRAY, telem,nsteps8
*DIM, out_rotz , ARRAY, telem,nsteps8
*DIM, out_kurv , ARRAY, telem,nsteps8
*DIM, out_kurvj , ARRAY, telem,nsteps8

*DIM, out_nodetx , ARRAY, nnodes,nsteps8
*DIM, out_nodety , ARRAY, nnodes,nsteps8

! calculate potential energy for the preload step (1)
*DO,i,1,nsteps2
SET,2,i
etable,potentialenergy,SENE
*vget,out_ene1(1,i),elem,,etab,potentialenergy
*ENDDO

! calculate potential energy for final step (2)
*DO,i,1,nsteps8
SET,8,i
etable,potentialenergy,SENE
*vget,out_ene2(1,i) ,elem,,etab,potentialenergy ! POTENTIAL ENERGY
etable,momz1,SMISC,3
*vget,out_momz1(1,i),elem,,etab,momz1 ! Z MOMENT
etable,str1,SMISC,32
*vget,out_str1(1,i) ,elem,,etab,str1 ! ALL STRESSES
etable,transx,U,X
*vget,out_tx(1,i) ,elem,,etab,transx ! X TRANSLATION
etable,transy,U,Y
*vget,out_ty(1,i) ,elem,,etab,transy ! Y TRANSLATION
etable,forcex,SMISC,14
*vget,out_fx(1,i) ,elem,,etab,forcex ! X FORCES
etable,forcey,SMISC,19
*vget,out_fy(1,i) ,elem,,etab,forcey ! Y FORCES
etable,rotz,ROT,Z
*vget,out_rotz(1,i) ,elem,,etab,rotz ! Z ROTATION
etable,kurv,SMISC,9
*vget,out_kurv(1,i) ,elem,,etab,kurv ! Z CURVATURE
etable,kurvj,SMISC,22
*vget,out_kurvj(1,i) ,elem,,etab,kurvj ! Z CURVATURE

*VGET,out_nodetx(1,i),NODE,1,U,X
*VGET,out_nodety(1,i),NODE,1,U,Y
*ENDDO

*MWRITE,out_ene1(1,1),D5_energies1,txt,  ,JIK,nsteps2,telem,1
(200E15.6)
*MWRITE,out_ene2(1,1),D6_energies2,txt,  ,JIK,nsteps8,telem,1
(200E15.6)
*MWRITE,out_momz1(1,1),D7_moment1,txt,  ,JIK,nsteps8,telem,1
(200E15.6)
*MWRITE,out_str1(1,1),D8_stress1,txt,  ,JIK,nsteps8,telem,1
(200E15.6)
*MWRITE,out_tx(1,1),D9_translationx,txt, ,JIK,nsteps8,telem,1
(200E15.6)
*MWRITE,out_ty(1,1),D10_translationy,txt, ,JIK,nsteps8,telem,1
```

```
(200E15.6)
*MWRITE,out_fx(1,1),D11_forcex,txt,   ,JIK,nsteps8,telem,1
(200E15.6)
*MWRITE,out_fy(1,1),D12_forcey,txt,   ,JIK,nsteps8,telem,1
(200E15.6)
*MWRITE,out_rotz(1,1),D13_rotationz,txt, ,JIK,nsteps8,telem,1
(200E15.6)
*MWRITE,out_kurv(1,1),D14_kurvature,txt, ,JIK,nsteps8,telem,1
(200E15.6)
*MWRITE,out_kurvj(1,1),D15_kurvaturej,txt, ,JIK,nsteps8,telem,1
(200E15.6)

*MWRITE,out_nodetx(1,1),D16_ntx,txt,   ,JIK,nsteps8,nnodes,1
(200E15.6)
*MWRITE,out_nodetx(1,1),D17_nty,txt,   ,JIK,nsteps8,nnodes,1
(200E15.6)

!__________________________________
/POST26
TIMERANGE
NUMVAR,200

!__________________________________STORE FORCES AND DISP

NSOL  ,2 ,ID_left ,ROT,Z,phi1
NSOL  ,3 ,ID_right,ROT,Z,phi2

*DIM, out_disp, ARRAY, 1000, 2
VGET,out_disp(1,1),2
VGET,out_disp(1,2),3

RFORCE,11,ID_left ,F,X,FX1
RFORCE,12,ID_left ,F,Y,FY2
RFORCE,13,ID_left ,M,Z,M1
RFORCE,14,ID_right,F,X,FX2
RFORCE,15,ID_right,F,Y,FY2
RFORCE,16,ID_right,M,Z,M2

*DIM, out_forces, ARRAY, 1000, 6
VGET,out_forces(1,1),11
VGET,out_forces(1,2),12
VGET,out_forces(1,3),13
VGET,out_forces(1,4),14
VGET,out_forces(1,5),15
VGET,out_forces(1,6),16


*CFOPEN,D3_results,txt
*VWRITE,out_disp(1,1),out_disp(1,2),out_forces(1,1),out_forces(1,2),out_forces(1,3),out_forces(1,4),out_force
(8(E15.6))
*CFCLOS

!__________________________________ WRite parameters to file
*CFOPEN,D1_anpar,txt
*VWRITE,nsteps1,nsteps2,nsteps3,nsteps4,nsteps5,nsteps6,nsteps7,nsteps8,rotnr
(1(E15.6))
*CFCLOS


!__________________________________
```

```
FINISH
!_________________________________
```

Faculty of Mechanical, Maritime and Materials Engineering

Master Mechanical Engineering

Track Bio Mechanical Design

TU Delft