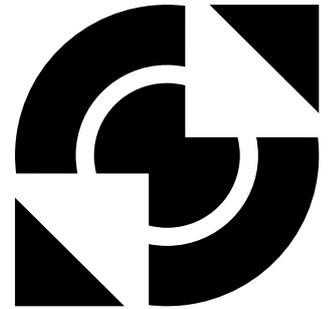


University of Twente

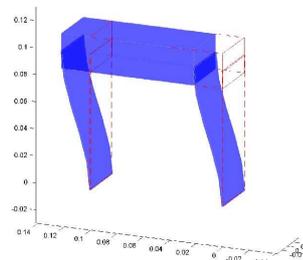
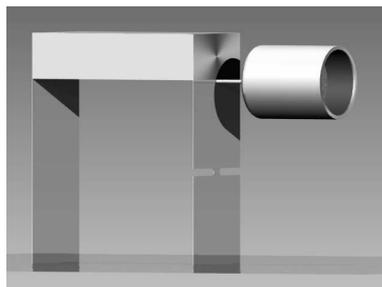


Faculty of  
Engineering Technology

Mechanical Automation and Mechatronics

---

Prototype modelling of  
mechanical systems  
An introduction for Project F



J.B. Jonker, R.G.K.M. Aarts, J. van Dijk

---

edition: 2008  
course: Project F (110325, WB 3.2)  
WA-nr.: 1139



© 2007–2008. Copyright J.B. Jonker, R.G.K.M. Aarts, J. van Dijk, Universiteit Twente, Enschede.

All rights reserved. No part of this work may be reproduced, in any form or by any means, without the explicit and prior permission from the authors.

Acknowledgement: Many of the graphs in this work as well as the data files and some of the text have been created by Jan Bennik and Joost Könemann.

# Contents

<b>Contents</b>	<b>i</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Example system . . . . .	2
1.2 SPACAR . . . . .	3
<b>2 Mass-spring model</b>	<b>5</b>
2.1 First SPACAR model: $x$ degree-of freedom . . . . .	6
2.2 Second SPACAR model: $e$ degree-of freedom . . . . .	13
<b>3 Two-dimensional model</b>	<b>15</b>
3.1 One degree of freedom model . . . . .	17
3.2 Three degrees of freedom model . . . . .	23
<b>4 Three-dimensional model</b>	<b>25</b>
4.1 One degree of freedom model . . . . .	27
4.2 Six degrees of freedom model . . . . .	32
<b>A SPACAR software</b>	<b>37</b>
A.1 Introduction . . . . .	37
A.2 SPACAR & MATLAB . . . . .	37
A.3 SPAVISUAL . . . . .	42
A.4 Download & install . . . . .	42
<b>B SPACAR keywords</b>	<b>45</b>
B.1 Introduction . . . . .	45
B.2 Kinematics . . . . .	45
B.3 Dynamics . . . . .	50
B.4 Linearization . . . . .	54
B.5 Visualization and animation . . . . .	56
<b>Bibliography</b>	<b>59</b>



# 1 Introduction

Modelling and analysis enable designers to test whether design specifications are met. It is important to be able to analyse a system at different levels - high level analysis in the early, conceptual stage when only a few design details are known, and more detailed towards the end of the design process. The use of prototype models significantly shortens the design time and reduces the cost of design. It further provides the designer with immediate feedback on design decisions, which, in turn, promises a more comprehensive exploration of design alternatives and a better performing final design.

For design of mechatronic systems it is essential to make use of simple prototype models with a few degrees of freedom that capture only the relevant system dynamics. In this tutorial an approach will be outlined to obtain so-called *prototype models* of the mechanical part of a mechatronic system. The applied techniques are taken from a more general *multibody system approach* [7, 8], which is a well-suited method to model the dynamic behaviour of the mechanical (sub)systems. In this approach the mechanical components are considered as rigid bodies that interact with each other through a variety of connections such as hinges and flexible coupling elements like trusses and beams (see figure 1.1). A mathematical description of these models is represented by the equations of motion in terms of the system's degrees of freedom (the Lagrange equations [6]) derived from the multibody systems approach).

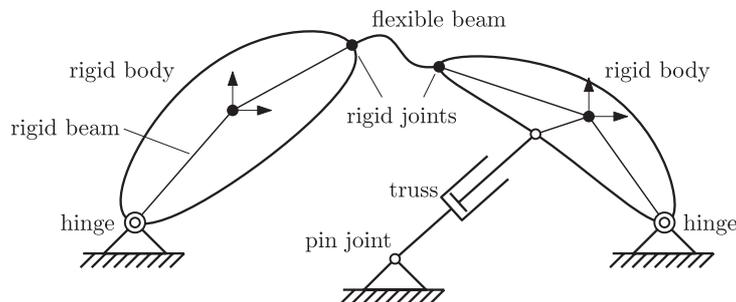


Figure 1.1: Flexible multibody system.

For control synthesis a control engineering based system representation is required. State space formulations are well suited to deal with both Single Input Single Output (SISO) systems as well as with more complicated Multiple Input Multiple Output (MIMO) systems. Linearization of the equations of motion is a well-known technique to obtain the state space matrices of a linearized state-space formulation for flexible multibody systems [1, 9].

A finite element based modelling approach is presented in which a multibody system is modelled as an assembly of rigid body elements interconnected by joint elements such as hinges and flexible elements like trusses and beams. For each element a fixed number of discrete deformation modes is defined that are invariant for rigid body movements of the element. This considerably reduces the number of elements necessary to obtain a sufficiently accurate model which makes the analysis fast and effective. The method is applicable for flexible multibody systems as well as for flexible structures in which the system members experience only small displacement motions and elastic deformations with respect to an equilibrium position. A software package called SPACAR based on this finite element approach is used for dynamic analysis and simulation of flexible multibody systems [7], which has an interface to MATLAB [11]. Subsequently all MATLAB tools for the analysis of (linear) systems are

available including graphical means like Bode plots and  $s$ -plane representations.

Typically for the design of mechatronic systems, the analyses mentioned above, consist of:

- Kinematics
- Natural frequencies and mode shapes
- State space input output formulations
- Static stability (buckling)
- Simulation of the dynamic behaviour

If one considers the following phases in mechatronic system design (not all in chronological order, some are executed in parallel):

- Conceptual design,
- Dimensioning the concepts,
- Computer aided prototyping,
- Final design (fine tuning),

then almost in each phase the dynamics play an important role and therefore modelling the dynamics is of great importance. However, the different phases demand each a more specific analysis type. In the conceptual phase the kinematics are of importance. In this phase it is important to restrict the number of degrees of freedom of the bodies of the concept to the view desired ones, by applying proper constraints. Most of the time the constraints are expressed in terms of stiffness. Thinking in terms of degrees of freedom is crucial in this phase and well supported by the underlying modelling method. Especially in the second phase the dynamics play a more dominant role. Mode-shape analysis and analysis of the (MIMO) transfer functions by state space formulations are crucial and also well supported by the modelling method. Computer models are very suitable for the computer aided prototyping if these models are adequate in the sense of incorporating the relevant dynamics but still fast in simulation and easy to change. In the final design phase much more methods and computer tools are needed than the one described here (i.e. ANSYS), although it is still of value in this phase for analysis of buckling and non-linear behaviour.

## 1.1 Example system

Application of the multibody system approach is illustrated through a detailed model development of an example system, which consists of a one degree of freedom (1-DOF) support mechanism with elastic leaf springs as shown in figure 1.2. The bar on the top has to be positioned horizontally. The support consisting of two leaf springs should confine all other degrees of freedom. Both springs are fixed at the bottom (clamped support). One spring is cut partly to obtain the correct number of degrees of freedom [10, Fig. i4.22]. The cylinder on the right represents a voice coil motor (VCM) that drives the motion of the system. Table 1.1 summarises the dimensions and mechanical properties of the system.

This system will be analysed with an increasing degree of complexity. In the first and most simplest model only a single mass and spring will be considered. This system will be modelled in two distinct ways and the concepts of *coordinates* and *deformations* as used throughout all analyses will be introduced. Next the bending of the leaf springs will be modelled more accurately with a two-dimensional model. Finally a three-dimensional model will be presented. Even such three-dimensional models can be created easily using only a few number of elements types. As the focus remains on the modelling of only the relevant dynamics, insight is obtained into the relations between component properties and dominant system behaviour.

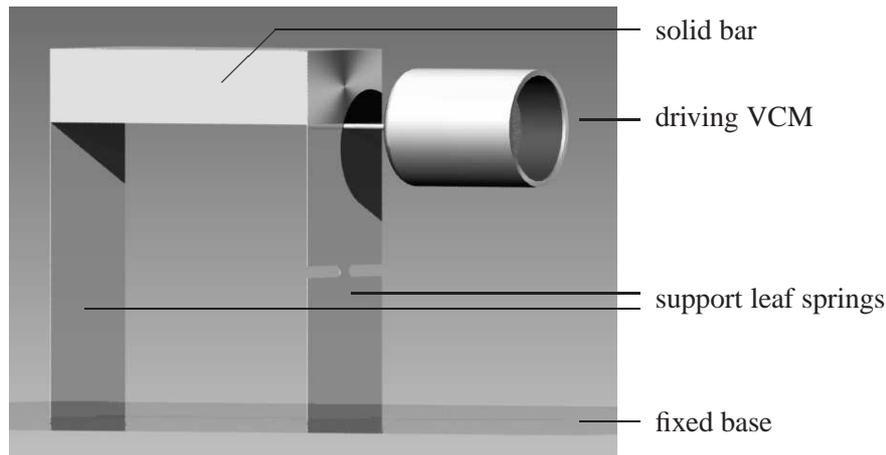


Figure 1.2: Example system driven by a voice coil motor.

<u>Solid bar</u>				
density	$\rho_b$	7690	kg/m <sup>3</sup>	
dimensions	$l_b \times w_b \times h_b$	$0.1 \times 0.02 \times 0.013$	m <sup>3</sup>	
volume	$V_b = l_b w_b h_b$	$2.6 \cdot 10^{-5}$	m <sup>3</sup>	
mass	$m_b = \rho_b V_b$	0.2	kg	
<u>Voice coil motor (VCM)</u>				
moving mass	$m_m$	0.006	kg	
<u>Support leaf springs</u>				
density	$\rho_s$	7850	kg/m <sup>3</sup>	
Young's Modulus	$E_s$	210	GPa	
Shear Modulus	$G_s$	80	GPa	
length	$l_s$	$1.0 \cdot 10^{-1}$	m	
width	$w_s$	$1.8 \cdot 10^{-2}$	m	
thickness	$t_s$	$5.0 \cdot 10^{-4}$	m	
cross-sectional area	$A_s = w_s t_s$	$9.0 \cdot 10^{-6}$	m <sup>2</sup>	
second moment of inertia	$I_s = \frac{1}{12} w_s t_s^3$	$1.88 \cdot 10^{-13}$	m <sup>4</sup>	
mass per unit length	$m_{l,s} = \rho_s A_s$	0.07065	kg/m	

Table 1.1: Dimensions and mechanical properties of the system in figure 1.2.

## 1.2 SPACAR

In all models the SPACAR software package is applied, which is a toolbox to be used in MATLAB. Appendix A provides download and installation instructions and an overview of SPACAR commands. For each system model a SPACAR input file is required that describes the system. Typical *keywords* for this description are introduced when they are needed for an analysis. In this tutorial the input files are split into parts that are explained one after each other. For a better overview of the complete input files and to redo the analyses, the reader is strongly advised to download the input files as mentioned in Appendix A. The most relevant SPACAR keywords are summarised in Appendix B.

It is essential to recognise that systems are defined in SPACAR by means of a number of *finite elements* interconnected in either two or three dimensions. Each element has *nodal points*. *Translational* and *rotational* nodal points are distinguished of which the *coordinates* describe the *position* and *orientation*, respectively, of the element with respect to a fixed global coordinate frame.

A very important property of the elements are the *deformation modes*. These deformations express the change of the “shape” of the element with respect to some undeformed initial “shape”. Formally, the deformations play a role in the mathematical relation between the nodal coordinates of the element.

E.g. an elongation of an element will result in a larger distance between translational nodal points. For each element a fixed number of independent (discrete) deformation modes are defined as functions of the nodal coordinates.

The (finite) elements, the keywords used to define them, and the element properties are introduced in this tutorial when they are used in an example. We will introduce so-called *truss* and *beam* elements. For a more detailed introduction and description, the reader is referred elsewhere [3, 8].

## 2 Mass-spring model

In this first analysis we will model the system of figure 1.2 as a simple one degree-of-freedom mass-spring system, figure 2.1. The voice coil motor gives rise to the input force  $F$  that positions the mass in a single direction denoted by coordinate  $x$ . The stator part of the voice coil is mounted on the support column which is assumed to be rigid and clamped at the bottom. All mass is lumped in a single equivalent mass  $m_{eq}$  and the equivalent stiffness  $k_{eq}$  represents the elastic components in actuation direction.

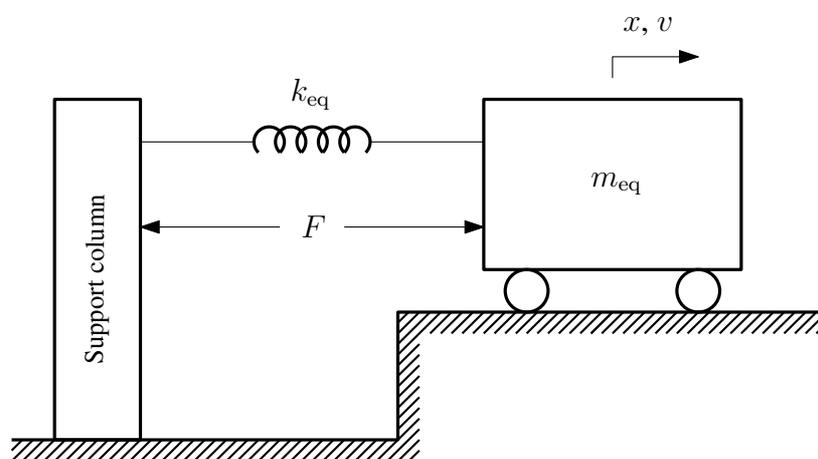


Figure 2.1: Simple 1-DOF mass-spring system.

Using straightforward analysis techniques [2] the input-output relation between the input force  $F$  and the position  $x$  of the mass can be obtained. A state vector can be defined consisting of the single degree of freedom, the position  $x$ , and its time-derivative, the velocity  $v$ . Then we can define the system input  $u$ , system output  $y$  and state vector  $\mathbf{x}$  to be

$$u = F, \quad y = x, \quad \mathbf{x} = \begin{bmatrix} x \\ v \end{bmatrix}. \quad (2.1)$$

As

$$\begin{aligned} \dot{x} &= v, \\ \dot{v} &= \frac{1}{m} (F - k_{eq} x), \end{aligned} \quad (2.2)$$

the state space equations can be written as

$$\begin{aligned} \dot{\mathbf{x}} &= \begin{bmatrix} 0 & 1 \\ -\frac{k_{eq}}{m_{eq}} & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ \frac{1}{m_{eq}} \end{bmatrix} u, \\ y &= [1 \quad 0] \mathbf{x} + [0] u, \end{aligned} \quad (2.3)$$

or in the short hand notation

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}u, \\ y &= \mathbf{C}\mathbf{x} + \mathbf{D}u, \end{aligned} \quad (2.4)$$

where  $A$ ,  $B$ ,  $C$  and  $D$  are the system, input, output and direct feedthrough matrices, respectively.

Alternatively, the input output relation can be expressed by means of a transfer function

$$G(s) = \frac{x}{F} = \frac{1}{m_{\text{eq}} s^2 + k_{\text{eq}}}. \quad (2.5)$$

In either representation numerical values for the mass  $m_{\text{eq}}$  and stiffness  $k_{\text{eq}}$  need to be substituted in order to carry out a numerical analysis. The mass  $m_{\text{eq}}$  should account for all mass that is translating in the  $x$  direction, so e.g. including the moving parts of the VCM. The equivalent bending stiffness of either leaf spring equals

$$k_1 = k_2 = \frac{12E_s I_s}{l_s^3} = \frac{E_s w_s t_s^3}{l_s^3} = 472.5 \text{ N/m}. \quad (2.6)$$

Note that the cuts in the right leaf spring in figure 1.2 do not affect the stiffness in the considered bending direction.

## 2.1 First SPACAR model: $x$ degree-of freedom

The mass-spring system can be modelled as a one-dimensional SPACAR model. For that purpose we need to identify elements and their coordinates and deformations as illustrated in figure 2.2. The numbers  $\vec{1}$  and  $\vec{2}$ , so numbers with an arrow, indicate the two translational nodal points. The number in the circle ① is the single element needed in this model. To model the spring either a *truss* or a *beam* element can be used. The latter is more complicated and will be introduced later.

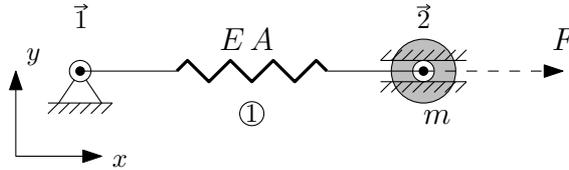


Figure 2.2: Simple 1-DOF mass-spring system with one truss element and two translational nodes.

### Planar truss element (PLTRUSS)

The truss element has two translational nodal points, in general denoted  $p$  and  $q$  as shown in figure 2.3. In two dimensions each translational nodal point has two coordinates, so the truss element is described with four coordinates:

$$\mathbf{x}_{\text{truss}}^{(k)} = \begin{bmatrix} \mathbf{x}^p \\ \mathbf{x}^q \end{bmatrix} = [x^p, y^p, x^q, y^q]^T. \quad (2.7)$$

The superscript in brackets is used to indicate the element number  $k$ . For this element a single deformation mode  $e_1$  is defined as the elongation and is written as

$$e_1^{(k)} = l^{(k)} - l_0^{(k)}, \quad (2.8)$$

where the actual length of the element is determined by the instantaneous distance between the nodes  $p$  and  $q$

$$l^{(k)} = \sqrt{(x^p - x^q)^2 + (y^p - y^q)^2}, \quad (2.9)$$

and the length  $l_0^{(k)}$  is some reference element length. For a rigid truss the length can not change and the deformation parameter  $e_1$  is zero. In that case equation (2.9) gives a relation between the four

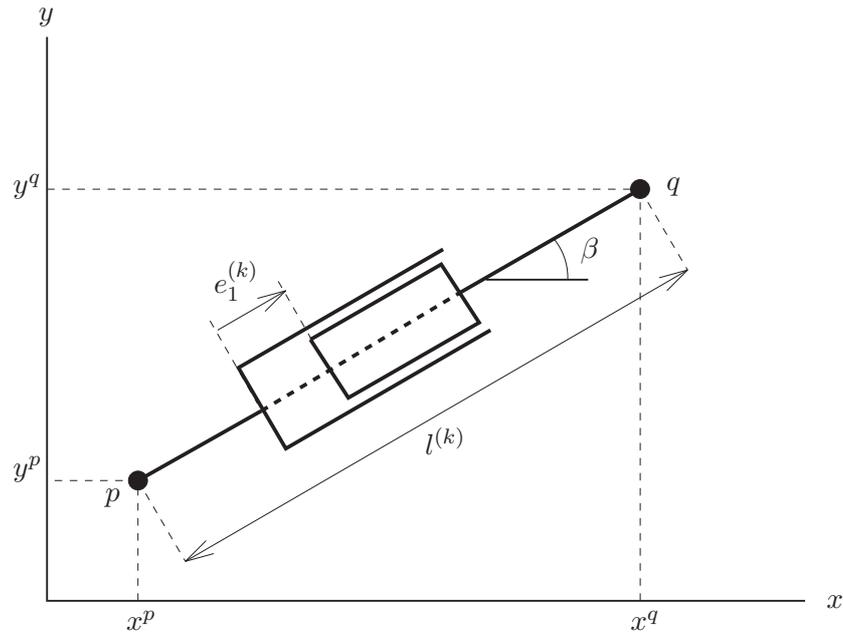


Figure 2.3: Planar slider truss element.

coordinates, so then only three coordinates can be chosen independently. This is exactly the number of degrees of freedom for the rigid truss element as it can translate in two directions and rotate in the  $xy$ -plane. It will appear for all elements that the number of deformation parameters is equal to the number of independent nodal coordinates minus the number of degrees of freedom of the rigid element.

A truss element can account for elasticity and/or damping in which case a normal force will appear that is modelled by

$$\sigma_1^{(k)} = S_1^{(k)} e_1^{(k)} + S_{d,1} \dot{e}_1^{(k)}, \quad (2.10)$$

with stiffness  $S_1^{(k)}$  and damping coefficient  $S_{d,1}$ . For an elastic beam the stiffness equals

$$S_1^{(k)} = \frac{E^{(k)} A^{(k)}}{l_0^{(k)}}, \quad (2.11)$$

where the *axial rigidity*  $E^{(k)} A^{(k)}$  includes the Young's Modulus  $E^{(k)}$  and cross-sectional area  $A^{(k)}$ . Similarly, the damping coefficient can be written as

$$S_{d,1}^{(k)} = \frac{E_d^{(k)} A^{(k)}}{l_0^{(k)}}, \quad (2.12)$$

where coefficient  $E_d^{(k)}$  describes the damping properties of the material as will be outlined in more detail following figure 2.4.

To analyse this system with SPACAR one has to create an input file. SPACAR input files always have the extension or file type `.dat`, so in MATLAB such a file can e.g. be created by typing

```
>> edit pfl1dtrx.dat
```

at the MATLAB command prompt. When the file does not exist, a new one can be created or else the existing file will be opened. The complete input file can be downloaded, see Appendix A.

Each SPACAR input file consists of a number of sections in which specific *keywords* may be used. In these sections the system is defined by considering successively the kinematic properties, the dynamic properties and finally the inputs and outputs. The sections will be described in that order.

### Kinematic section of the input file

The first section gives the *kinematic* description of the system. The system of figure 2.2 can be defined with the following sequence of commands:

```
PLTRUSS 1 1 2
```

The keyword `PLTRUSS` defines a *planar truss*, i.e. a truss element in two dimensions. It is followed by three integer numbers. The first number (1) indicates the element number. It should be taken care of that each element has a unique number. The next numbers indicate the nodal point numbers. As the truss element has two translational nodal points there are indeed two nodal points defined: 1 and 2.

```
X 1 0.0 0.0
X 2 0.1 0.0
```

These `X` keywords specify the initial Cartesian nodal positions. Each `X` is followed by one integer number and two real numbers. The first number indicates the nodal point number. The real numbers are the Cartesian nodal positions in the  $x$  and  $y$  directions, respectively. For the system in figure 2.2 nodal point 1 is positioned in the origin and nodal point 2 is positioned at 0.1 m on the positive  $x$ -axis. Note that in this model the reference length  $l_0^{(k)} = 0.1$  m of the truss element does not have a physical meaning as there is no elongating elastic element in the real system (figure 1.2) and the truss only appears in the simplified model (figure 2.2).

The kinematic description of the system is continued by discriminating the fixed and varying nodal point coordinates and element deformation parameters.

```
FIX 1
FIX 2 2
```

To understand the meaning of these command, one first has to recognise that nodal point coordinates are free unless defined otherwise. For translational nodes, the keyword `FIX` specifies which nodal coordinates are *fixed* to the support. It is followed by one or two integers. The first integer indicates the nodal point number. The (optional) second integer indicates the Cartesian coordinate direction that is fixed. When no second number is specified all coordinate directions of the nodal point are fixed. So the first `FIX` command implies that translational node 1 is fixed in both  $x$  and  $y$  directions. With the second `FIX` command nodal point 2 is fixed in the  $y$  direction.

```
RLSE 1 1
```

In contrast, the element deformations are prescribed zero unless defined otherwise. The `RLSE` command defines a *released* deformation parameter. It is followed by two integers. The first number indicates the element number. The second number indicates the deformation parameter to be released. In this example deformation parameter 1 of element 1 is released, which is the elongation  $e_1^{(1)}$  of the planar truss element representing the spring. Note that in combination with the `FIX` commands the  $x$  coordinate of nodal point 2 can indeed move freely.

The next crucial step is the specification of the degree(s) of freedom. The number of degrees of freedom NDOF has to be equal to

$$\text{NDOF} = \text{NX} - \text{NXO} - \text{NEO}, \quad (2.13)$$

which is the number of nodal coordinates  $\text{NX}$  minus the numbers of *absolute* and *relative constraints*,  $\text{NXO}$  and  $\text{NEO}$  respectively. Absolute constraints are defined by means of the `FIX` commands. The relative constraints are the remaining unreleased element deformation parameters. Violating equation (2.13)

is a frequent cause for an error generated by SPACAR. In the accompanying error message the names of the variables are as in equation (2.13).

For the example of figure 2.2 there are  $NX = 2 \times 2 = 4$  nodal coordinates of which  $NXO = 3$  coordinates are FIX-ed. The single truss element has only one deformation parameter which is released, so  $NEO = 0$  relative constraints remain. Then  $NDOF = 4 - 3 - 0 = 1$  degree of freedom has to be defined. One possibility is:

```
DYNX 2 1
```

The keyword DYNX defines a so-called *absolute degree of freedom*. It is followed by two integer numbers indicating nodal point number and coordinate number, respectively. In this example the first coordinate, the  $x$  coordinate, of nodal point 2 is chosen as degree of freedom.

```
END
HALT
```

The kinematic section of the input file has to be ended with the two keywords END and HALT.

### Dynamic section of the input file

In the second section the *dynamic* properties of the system are specified. For the system of figure 2.2 we proceed with the following sequence of commands:

```
XM 2 0.206
EM 1 0.1413
```

The keyword XM specifies a *lumped* mass attached at a nodal point. It is followed by one integer number and one or more real numbers. The integer number indicates the nodal number. In this example only one real parameter is needed (the second number) to specify the mass ([kg]) attached to the nodal point. The meaning of additional real parameters is e.g. explained in appendix B. The keyword EM specifies a *distributed* mass along an element. It is followed by one integer number which indicates the element number and a real number to specify the mass per length ([kg/m]) of the element. In this example the mass  $m_b$  of the solid bar and the moving mass  $m_m$  of the VCM are attached to nodal point 2. The element mass accounts for both leaf springs.

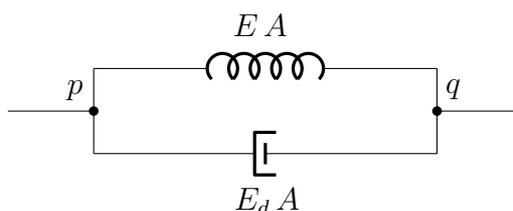


Figure 2.4: Spring damper combination.

The next keywords ESTIFF and EDAMP are used to specify the element's stiffness and damping properties respectively. Essentially a spring damper combination as shown in figure 2.4 can be represented by combining both keywords as in:

```
ESTIFF 1 94.5
EDAMP 1 0.00365
```

The keyword ESTIFF is used to specify the axial rigidity  $EA$  of the truss element that represents the spring. It is followed by one integer number and one or more real numbers. The integer indicates the element number. For the truss element only one real parameter is allowed that should be equal to the

axial rigidity  $EA$ , where  $E$  is the Young's modulus and  $A$  the cross-sectional area of the truss. In the case the elasticity is described by an equivalent longitudinal stiffness  $k_{\text{eq}}$  we recognise that

$$(EA)_{\text{eq}} = k_{\text{eq}}l_0 = 94.5 \text{ N},$$

where  $l_0$  is the reference length of the truss element and, in this example, the equivalent stiffness equals

$$k_{\text{eq}} = k_1 + k_2 = 945 \text{ N/m},$$

making use of the equivalent bending stiffness of both leaf springs described by equation 2.6.

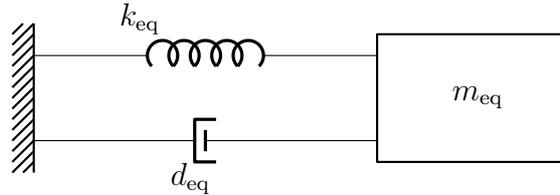


Figure 2.5: Lumped model for internal damping.

Similarly the EDAMP keyword specifies the element's internal viscous damping. The damping of the spring leaves is not exactly known. For metals a typical *relative damping*  $\zeta$  in the range of 0.01 to 0.001 is expected. Then each spring leave can be modelled as the mass-spring-damper system of figure 2.5. For this system the (equivalent) damping  $d_{\text{eq}}$  is related to the relative damping  $\zeta$ , the (equivalent) stiffness  $k_{\text{eq}}$  and (equivalent) mass  $m_{\text{eq}}$  as

$$d_{\text{eq}} = 2\zeta\sqrt{k_{\text{eq}}m_{\text{eq}}}. \quad (2.14)$$

In this tutorial the relative damping is taken:  $\zeta = 0.005$ . For the truss, an equivalent damping for the two leaf springs can be calculated

$$d_{\text{eq}} = 2 \cdot 0.005 \cdot \sqrt{945 \cdot 0.01413} = 0.0365 \text{ Ns/m}. \quad (2.15)$$

Multiplication with reference length  $l_0$  gives the parameter of the EDAMP keyword

$$(E_dA)_{\text{eq}} = d_{\text{eq}}l_0 = 0.00365 \text{ Ns}.$$

XF 2 1. 0.

The keyword XF is used to specify external static forces. For a planar system the keyword XF is followed by one integer number and two real numbers. The integer number represents the node number. The real numbers indicate the force components in  $x$ - and  $y$ -direction applied at a translational node. For the system in figure 2.1 a horizontal force with a magnitude of 1 N is applied at node 2 in the positive  $x$ -direction.

END  
HALT

The dynamic section of the input file has to be ended with the two keywords END and HALT.

### Input-output section of the input file

The third section gives the description of the *inputs* and *outputs* of the system. For the system of figure 2.2 we use the following sequence of commands:

```
INPUTF  1  2  1
OUTX    1  2  1
```

The keywords `INPUTF` and `OUTX` define a single input force  $F$  and a single output coordinate  $X$ , respectively. Both keywords are followed by three integer numbers. The first number indicates the input or output number that corresponds with its position in the input vector  $\mathbf{u}$  or output vector  $\mathbf{y}$  respectively. The inputs and outputs of a system should be numbered consistently, i.e. no numbers should be skipped or used more than once. The second and third integers indicate the nodal point number and the corresponding Cartesian nodal coordinate. So the `OUTX` keyword defines the  $x$  coordinate of nodal point 2 to be the first (and only) component of the output vector  $\mathbf{y}$ . Similarly the force on this point is the first (and only) component of the input vector  $\mathbf{u}$ . The third number 1 indicates that the component of the force in the  $x$  direction is considered.

```
END
END
```

The final section of the input file has to be ended using twice the keyword `END`.

### SPACAR results

The goal of a typical SPACAR analysis could be a static analysis and the computation of the natural frequencies of the system. For this purpose mode 7 is well-suited. It is executed from the MATLAB command prompt

```
>> spacar(7,'pf1dtrx')
```

The output of this command is a MATLAB plot window that shows the system in its equilibrium configuration, where a green line indicates the truss element. The generated log file `pf1dtrx.log` is a plain text file in which the processing of the input file and some of the analysis results can be read. It is advisable to read this file always after a SPACAR run to check for any unexpected messages that can indicate mistakes in the input file. Data is written to the binary files `pf1dtrx.sbd` and `pf1dtrx.sbm`. Immediately after the SPACAR run, the variables stored in these files are also available in MATLAB. We can check the actual horizontal position of node 2 with the command:

```
>> x(lnp(2,1))
ans =
    0.1011
```

Location matrix `lnp(i,j)` denotes the location of the  $j$ th coordinate, or associated force component, of node  $i$ , see Appendix A. The normal force in the truss element is obtained by:

```
>> sig(le(1,1))
ans =
    1.0000
```

Location matrix `le(i,j)` denotes the location of the  $j$ th deformation, or associated stress resultant component, of element  $i$ , see Appendix A.

We can e.g. check the mass and stiffness “felt” by the degree of freedom that we defined:

```
>> m0
m0 =    0.2107

>> k0
k0 =    945
```

From these values the natural frequency (in Hz) follows as:

```
>> sqrt(eig(k0,m0))/2/pi
ans =
    10.6584
```

With mode 9 the state space matrices can be computed

```
>> spacar(9,'pflldtrx')
```

As a result of this run an additional data file `pflldtrx.ltv` is written in which the matrices  $A$ ,  $B$ ,  $C$ , and  $D$  are stored. These are not immediately available in MATLAB, but can be read with the `getss` command:

```
>> getss('pflldtrx')

a =
           x1           x2
x1         0           1
x2    -4485   -0.1732

b =
           u1
x1         0
x2    4.746

c =
           x1  x2
y1    1      0

d =
           u1
y1    0
```

Continuous-time model.

Note that in this (unmodified) MATLAB output the variables  $a$ ,  $b$ ,  $c$  and  $d$  denote the state space matrices  $A$ ,  $B$ ,  $C$  and  $D$  as in equation (2.4) with a small extra term due to the damping. Also the names for the states  $x_1$  and  $x_2$ , the input  $u_1$  and output  $y_1$  are generated by MATLAB.

The state space representation can be easily transformed into a transfer function (equation (2.5)) using MATLAB's `tf` command:

```
>> G=tf(getss('pflldtrx'))
```

```
Transfer function:
      4.746
-----
s^2 + 0.1732 s + 4485
```

```
>> bode(G)
```

where the latter command is used to create the Bode plot in figure 2.6.

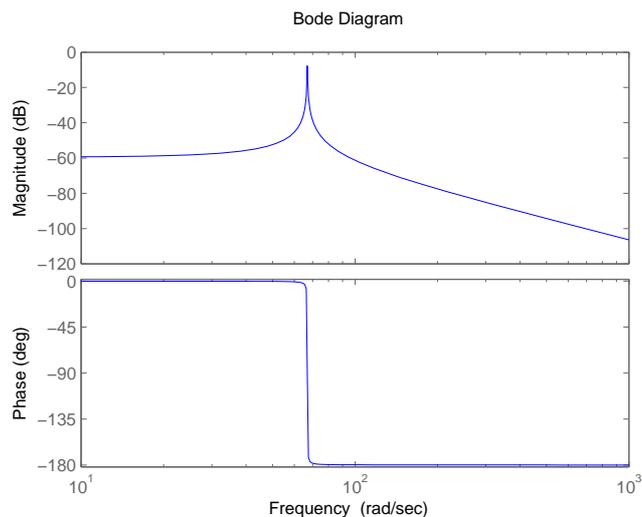


Figure 2.6: Bode plot of the 1-DOF mass-spring system.

## 2.2 Second SPACAR model: $e$ degree-of freedom

As was mentioned in the previous section, the selection of the degree of freedom is not unique. Another possibility is to define the deformation parameter  $e_1$  of the truss element, the elongation, as the degree of freedom. This can be done by copying the previous input file `pfl1dtrx.dat` to `pfl1dtre.dat` and replacing the kinematic part of the input file with

```
PLTRUSS 1 1 2

X 1 0.0 0.0
X 2 0.1 0.0

FIX 1
FIX 2 2

DYNE 1 1

END
HALT
```

The only difference with the previous input file is the use of the keyword `DYNE` that defines a *relative* degree of freedom. It is followed by two integer numbers. The first number indicates the element number and the second number indicates the element's deformation parameter. As the truss element number 1 has only one deformation parameter, this is the only relative degree of freedom that can be chosen. Note that the `RLSE` command should not be used to release the deformation when it is defined as relative degree of freedom (`DYNE`). The rest of the input file is unaltered and it can be used for similar SPACAR analyses as in the previous section.



### 3 Two-dimensional model

Next, a more complicated model will be described in which the bending of the leaf springs will be modelled more accurately using a so-called planar beam element. Figure 3.1 shows a planar frame model of the example system of figure 1.2. It is modelled using three beam elements. The solid bar is modelled by a rigid beam element ②. Both support leaf springs are modelled with the flexible beam elements ① and ③ as will be explained in more detail below.

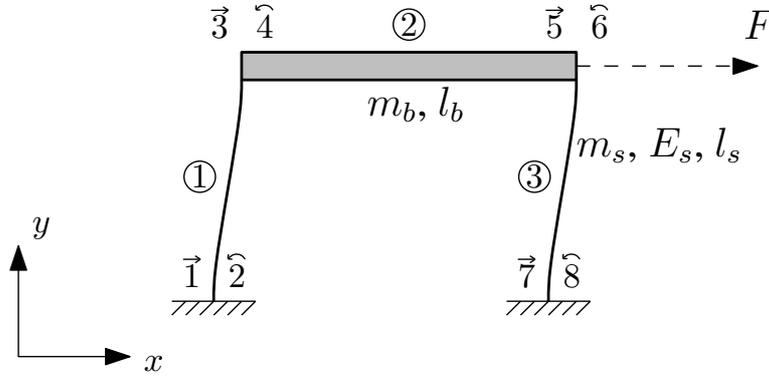


Figure 3.1: Planar frame model with elastic beams.

Like the truss element on either side of a beam a translational node is attached. In figure 3.1 the nodes 1, 3, 5 and 7 represent these translational nodes, where nodes 3 and 5 are common nodes between two beams to indicate that these nodes are shared. In addition, the nodes 2, 4, 6 and 8 indicate *rotational* nodes. These are of importance in the description of the bending deformations of the flexible beam elements. Note that also the rotational nodes 4 and 6 are common between the two beams on either side. Sharing a translational node between two or more beams implies that these beams are connected to each other in that point, but they can still rotate relative to each other. If these beams also share a rotational node, then relative rotations are no longer possible and the beams are completely fixed.

#### Planar beam element (PLBEAM)

The configuration of the beam element is described by the position vectors  $x^p$  and  $x^q$  of the end nodes  $p$  and  $q$  and the angular orientation of orthonormal triads rigidly attached to the element nodes, as shown in figure 3.2. The rotation part of the motion of the beam element is described by the rotation of the triads which are described by planar rotation matrices  $R^p$  and  $R^q$ , defined by

$$R^p \equiv \begin{bmatrix} \cos \phi^p & -\sin \phi^p \\ \sin \phi^p & \cos \phi^p \end{bmatrix} \quad \text{and} \quad R^q \equiv \begin{bmatrix} \cos \phi^q & -\sin \phi^q \\ \sin \phi^q & \cos \phi^q \end{bmatrix} . \quad (3.1)$$

As nodal coordinates for the beam element we have four Cartesian coordinates  $(x^p, y^p)$ ,  $(x^q, y^q)$  describing the position of the beam in the  $(x, y)$ -coordinate system and two rotation angles  $\phi^p$  and  $\phi^q$  representing the angular orientation of the triads  $(R^p n_x, R^p n_y)$  and  $(R^q n_x, R^q n_y)$  at the nodes  $p$

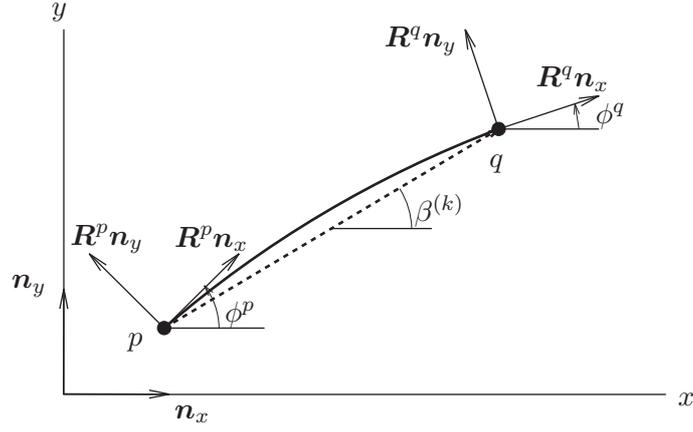
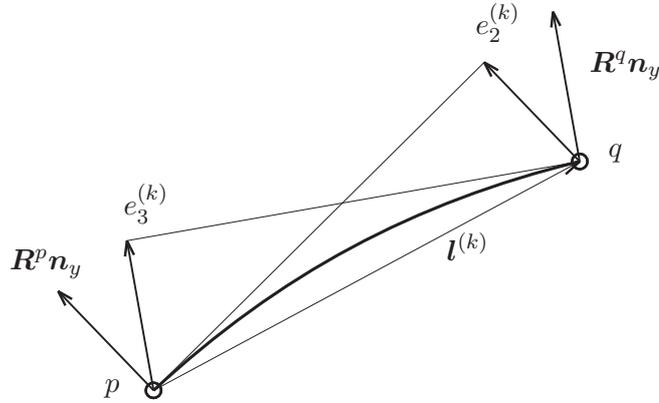


Figure 3.2: Planar flexible beam element.

and  $q$  respectively. Hence the vector of nodal coordinates is given by

$$\mathbf{x}_{\text{beam}}^{(k)} = \begin{bmatrix} \mathbf{x}^p \\ \phi^p \\ \mathbf{x}^q \\ \phi^q \end{bmatrix} = [x^p, y^p, \phi^p, x^q, y^q, \phi^q]^T, \quad (3.2)$$

where  $\mathbf{x}^p, \mathbf{x}^q$  and  $\phi^p, \phi^q$  are the translational and rotational nodal points respectively. In the undeflected state the vectors  $\mathbf{R}^p \mathbf{n}_x$  and  $\mathbf{R}^q \mathbf{n}_x$  are directed along the beam axis  $pq$  and the vectors  $\mathbf{R}^p \mathbf{n}_y$  and  $\mathbf{R}^q \mathbf{n}_y$  are perpendicular to the beam axis.

Figure 3.3: Visualisation of bending deformations  $e_2^{(k)}$  and  $e_3^{(k)}$  of the planar beam element.

In addition to the deformation  $e_1^{(k)}$  representing the elongation of the element, two deformation parameters  $e_2^{(k)}$  and  $e_3^{(k)}$ , associated with the flexible deformation of the beam element, can be defined:

$$\text{elongation:} \quad e_1^{(k)} = l^{(k)} - l_0^{(k)}, \quad (3.3)$$

$$\text{bending:} \quad e_2^{(k)} = -(\mathbf{R}^p \mathbf{n}_y, \mathbf{l}^{(k)}), \quad (3.4)$$

$$e_3^{(k)} = (\mathbf{R}^q \mathbf{n}_y, \mathbf{l}^{(k)}), \quad (3.5)$$

where  $\mathbf{n}_y = [0, 1]^T$  and the vector  $\mathbf{l}^{(k)}$  is defined by

$$\mathbf{l}^{(k)} \equiv \mathbf{x}^q - \mathbf{x}^p = [(x^q - x^p), (y^q - y^p)]^T. \quad (3.6)$$

The geometric meaning of the bending deformations  $e_2^{(k)}$  and  $e_3^{(k)}$  is visualised in figure 3.3.

Note that the deformations are independent for rigid body movements of the element. Furthermore, they have a clear physical meaning. If the deformations remain sufficiently small, then in the elastic range they are linearly related to known beam quantities as normal force ( $\sigma_1^{(k)}$ ) and bending moments ( $\sigma_2^{(k)}l^{(k)}, \sigma_3^{(k)}l^{(k)}$ ) by the element stiffness equations:

$$\begin{bmatrix} \sigma_1^{(k)} \\ \sigma_2^{(k)} \\ \sigma_3^{(k)} \end{bmatrix} = \begin{bmatrix} \frac{E^{(k)}A^{(k)}}{l^{(k)}} & 0 & 0 \\ 0 & 4\frac{E^{(k)}I^{(k)}}{(l^{(k)})^3} & -2\frac{E^{(k)}I^{(k)}}{(l^{(k)})^3} \\ 0 & -2\frac{E^{(k)}I^{(k)}}{(l^{(k)})^3} & 4\frac{E^{(k)}I^{(k)}}{(l^{(k)})^3} \end{bmatrix} \begin{bmatrix} e_1^{(k)} \\ e_2^{(k)} \\ e_3^{(k)} \end{bmatrix}, \quad (3.7)$$

where  $E^{(k)}$  is the modulus of elasticity,  $A^{(k)}$  the cross section area and  $I^{(k)}$  the second moment of inertia. In figure 3.4 the stress resultant components are visualised.

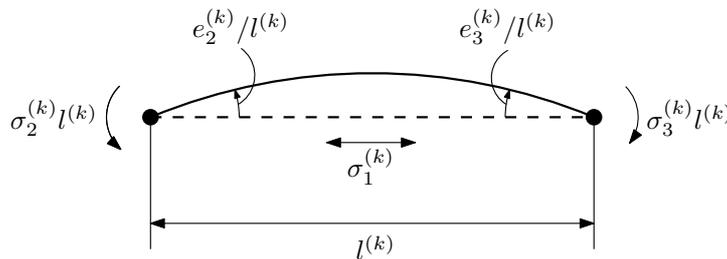


Figure 3.4: Normal force and bending moments of the planar beam element.

## 3.1 One degree of freedom model

### Kinematic section of the input file

Before describing the SPACAR input file for the planar frame model in figure 3.1, the computation of the number of degrees of freedom according to equation (2.13) is evaluated. The figure shows that there are in total four translational nodes (1, 3, 5, 7) with two coordinates each and four rotational nodes (2, 4, 6, 8) with one coordinate each. So in total there are  $NX = 12$  coordinates. The two translational nodes 1 and 7 and the rotational nodes 2 and 8 are clamped, so the number of absolute constraints is  $NXO = 6$ . The three beams each have three deformation parameters, so there are  $NE = 9$  deformations. If one degree of freedom ( $NDOF = 1$ ) is desired, then according to equation (2.13) the number of relative constraints  $NEO = 5$ . This is realised by prescribing the longitudinal deformation of beams 1 and 3 and all deformations of beam 2 equal to zero.

In this example the horizontal position of translational node 3 is chosen as degree of freedom. Then the following commands for the kinematic section of the input file can be used:

```
PLBEAM 1 1 2 3 4
PLBEAM 2 3 4 5 6
PLBEAM 3 5 6 7 8
```

Each PLBEAM keyword defines a planar beam. It is followed by five integers. The first indicates the element number. Next the translational nodal point number and the rotational nodal point number of end node  $p$  are specified. Finally these nodal point numbers for the end node  $q$  are given. Note that the only way to distinguish between a translational and rotational nodal point number arises from its (implicit) definition as part of the element definition. Note also that the keyword PLRBEAM defines a planar *rigid* beam and may be used for the solid bar. This element is described in the SPACAR manual [3].

```
X 1 0.0 0.0
X 3 0.0 0.1
X 5 0.1 0.1
X 7 0.1 0.0
```

The initial positions of the translational nodes are defined with four X keywords.

```
FIX 1
FIX 2
FIX 7
FIX 8
```

The four FIX keywords impose the  $NXO = 6$  constraints for the clamped nodes.

```
RLSE 1 2 3
RLSE 3 2 3
```

The four bending modes ( $e_2^{(1)}$ ,  $e_3^{(1)}$ ,  $e_2^{(3)}$ , and  $e_3^{(3)}$ ) of the beams representing the leaf springs are released with two RLSE keywords. As shown in this example more deformations of an element can be released with a single RLSE keyword.

```
DYNX 3 1
```

Finally the degree of freedom is specified and the kinematic section of the input file is concluded:

```
END
HALT
```

### Dynamic section of the input file

For a dynamic analysis the *distributed* inertia of the solid bar ② has to be modelled. The simplest way to accomplish this is by means of a lumped mass representation. In this idealisation rigid bodies with equivalent mass and rotational inertia properties are attached to the end nodes of the element. Three conditions have to be satisfied to obtain the equivalent masses and rotational inertias [13]:

1. The mass of the element should be equal to the sum of the lumped masses.
2. The center of mass of the element and that of the discrete mass model should coincide.
3. The rotational inertia of the element and that of the lumped system should be equal.

For the solid bar in the left graph of figure 3.5 with mass  $m$  and length  $l$  the rotational inertia relative to the centre of mass  $c$  equals  $J_c = \frac{1}{12}ml^2$ . The equivalent lumped masses and rotational inertias are then shown in the right graph and can be written as

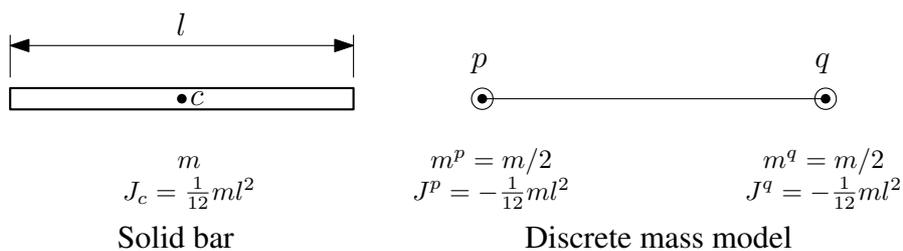


Figure 3.5: Solid bar and equivalent lumped mass system.

$$m^p = m^q = \frac{1}{2}m, \quad (3.8)$$

and

$$J^p = J^q = -\frac{1}{12}ml^2. \quad (3.9)$$

In the SPACAR input file the lumped masses are defined with the following commands:

```
XM 3  0.103
XM 4 -0.0001716
XM 5  0.103
XM 6 -0.0001716
```

For the translational nodes 3 and 5 the definition of the (lumped) mass with the XM keyword is as before. For the rotational nodes the rotational inertia is given. Note that in this example the solid bar, element 2, experiences a translation only and hence the rotational inertia will not affect the analysis.

The inertia properties of the *flexible* beams representing the leaf springs are defined in terms of a distributed mass using the commands:

```
EM 1  0.07065
EM 3  0.07065
```

As before, the keyword EM is followed by one integer number and one real number. The integer number indicates the element number. The real number represents the mass per length ([kg/m]).

```
ESTIFF 1  1890000.  0.039375
ESTIFF 3  1890000.  0.039375

EDAMP  1      0.3654  0.0000053
EDAMP  3      0.3654  0.0000053
```

The ESTIFF and EDAMP define stiffness and damping. For the planar beam element two real parameters are specified. The first real parameter represents the axial rigidity  $EA$ , where  $E$  is the modulus of elasticity and  $A$  is the cross-sectional area of the beam. The second real parameter represents the flexural rigidity  $EI$ , where  $I$  is the second moment of inertia of the cross-section with respect to the neutral axis. For the leaf springs this moment of inertia equals

$$I_s = \frac{w_s t_s^3}{12} = 1.88 \cdot 10^{-13} \text{ m}^4,$$

as was used previously in equation 2.6.

The damping values can be calculated with equation 2.14. As was shown before, first the longitudinal stiffness  $EA/l_0$  is computed from the axial rigidity  $EA$ . This stiffness and the total mass of the spring are substituted in equation 2.14 to obtain the damping which is related to the required parameter  $E_d A$ . The procedure for the flexural rigidity  $EI$  is similar, except that the cube of the length  $l_0^3$  appears in the expressions, e.g. to compute the bending stiffness  $EI/l_0^3$ .

Note that when the longitudinal deformation is not released the specifications for the longitudinal stiffness and damping are of no account.

The dynamic part of the input file is concluded with:

```
END
HALT
```

### Input-output section of the input file

```
INPUTF 1 5 1
OUTX 1 3 1
OUTX 2 3 2
OUTX 3 5 1
OUTX 4 5 2
OUTX 5 4 1
OUTX 6 6 1
```

In addition to the desired position of nodal point 3 in the  $x$ -direction, five other translational and rotational nodal coordinates are defined in the output vector.

```
END
END
```

### Optional visualisation section of the input file

```
VISUALIZATION
VIBRATIONMODE 1
VIBREND 7.854
ENLARGEFACTOR 0.005
```

SPACAR reads up to the the last two END keywords. Hereafter custom settings for SPAVISUAL can be declared. SPAVISUAL reads all input after VISUALIZATION. Explanation of the keywords can be found in Appendix B.

### SPACAR results for natural frequency and vibration mode

As with the one-dimensional model a mode 7 analysis can be executed to find the natural frequency of the mechanism.

```
>> spacar(7,'pf2d');
>> sqrt(eig(k0,m0))/2/pi
```

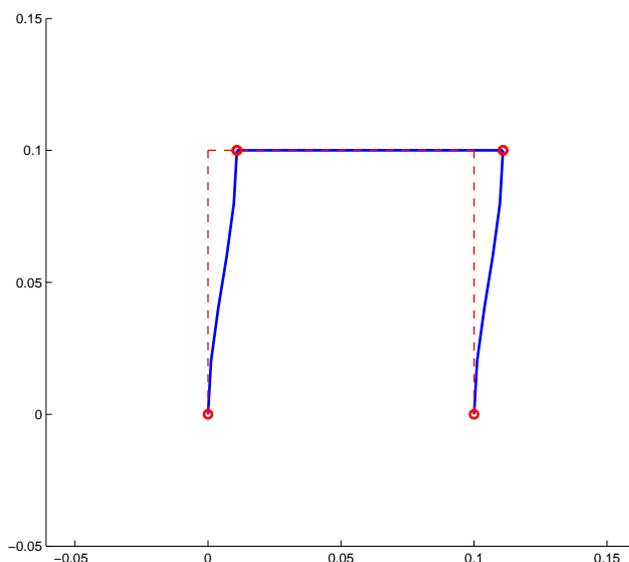


Figure 3.6: First vibration mode of the two-dimensional model (10.6448 Hz).

```
ans =
    10.6448
```

The vibration mode is visualised using SPAVISUAL, see figure 3.6.

```
>> spavisual('pf2d');
```

### SPACAR results of a buckling analysis

By means of mode 8 a linear buckling analysis can be carried out for a static equilibrium configuration. A buckling load  $f_i = \lambda_i f_0$  is computed, where  $\lambda_i$  is a critical loading parameter and  $f_0$  represents a static reference loading vector of nodal forces. An external force component is applied adding the command:

```
XF 3 0. -1.
```

to the previous SPACAR input file and saving it as pf2db.dat.

For the system in figure 3.1 a vertical force with a magnitude of 1 N is applied at node 3 in the negative  $y$ -direction. A linear buckling analysis is executed using the MATLAB command prompt

```
>> spacar(8,'pf2db')
```

Before the buckling analysis is executed, the equilibrium configuration is determined. The critical loading parameter  $\lambda_1$  is computed using the command:

```
>> eig(-k0,g0)
ans =
    78.7500
```

The buckling mode is visualised using SPAVISUAL, in figure 3.7 this mode is shown.

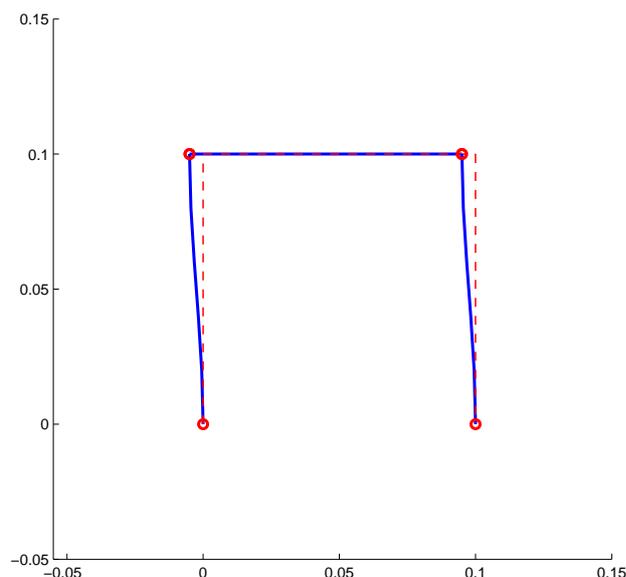


Figure 3.7: First buckling mode of the two-dimensional model ( $f_1 = 78.75$  N).

**SPACAR results for a state space model**

Next, the state space matrices are computed (reverting to the earlier `pf3d.dat`-file without the static reference load):

```
>> spacar(9,'pf2d');
```

Again, the matrices must be read from the `ltv`-file with the `getss` command (where small numbers in the actual MATLAB output have been replaced by zeros for readability):

```
>> getss('pf2d')
```

```
a =
      x1      x2
x1      0      1
x2  -4473  -0.6021
```

```
b =
      u1
x1      0
x2  4.734
```

```
c =
      x1  x2
y1      1  0
y2      0  0
y3      1  0
y4      0  0
y5      0  0
y6      0  0
```

```
d =
      u1
y1      0
y2      0
y3      0
y4      0
y5      0
y6      0
```

Continuous-time model.

For each input - output combination a transfer function can be generated:

```
>> tf(getss('pf2d'))
```

Transfer function from input to output...

```

          4.734
#1:  -----
      s^2 + 0.6021 s + 4473

#2:  0
```

```

                4.734
#3:  -----
      s^2 + 0.6021 s + 4473

#4:  0

#5:  0

#6:  0

```

The Bode plot of the first transfer function can be found in figure 3.8.

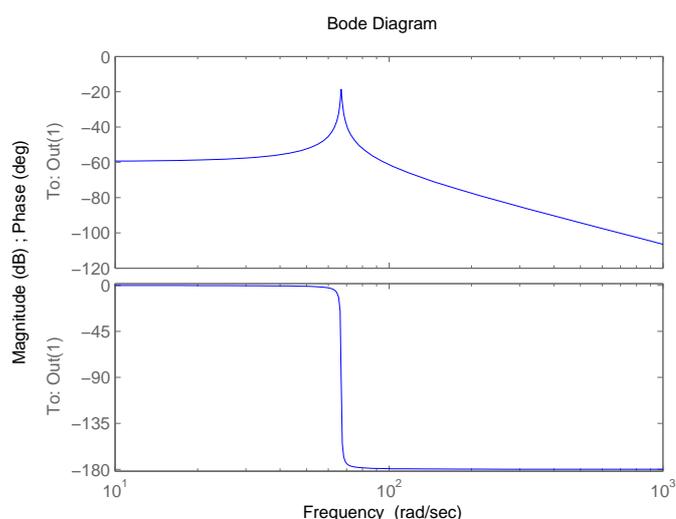


Figure 3.8: Bode plot of transfer from input to first output of the two-dimensional model.

## 3.2 Three degrees of freedom model

It is the responsibility of the user to select the proper degrees of freedom and release deformations. Releasing all deformations of the two leaf springs results in two extra degrees of freedom as can be calculated with equation (2.13):

$$\text{NDOF} = \text{NX} - \text{NXO} - \text{NEO} = 12 - 6 - 3 = 3. \quad (3.10)$$

The coordinates of translational node 3 and the single coordinate of rotational node 4 are chosen as dynamic degrees of freedom. The input file changes into

```

RLSE 1 1 2 3
RLSE 3 1 2 3

DYNX 3
DYNX 4

```

The extra degrees of freedom result in mass and stiffness matrices of dimension  $3 \times 3$  and a total of three natural frequencies. The matrices are available in the MATLAB workspace as row vectors. With MATLAB's `reshape` command they can be presented as square matrices:

```
>> reshape(k0,3,3)
```

```

1.0e+007 *

    0.0001    -0.0000    0.0000
   -0.0000    3.7800    0.1890
    0.0000    0.1890    0.0189

>> reshape(m0,3,3)

    0.2112         0    0.0001
         0    0.2112    0.0106
    0.0001    0.0106    0.0007

```

The natural frequencies are:

```

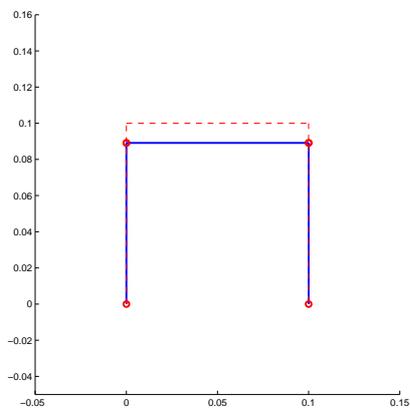
>> sort(sqrt(eig(reshape(k0,3,3),reshape(m0,3,3)))/2/pi)

1.0e+003 *

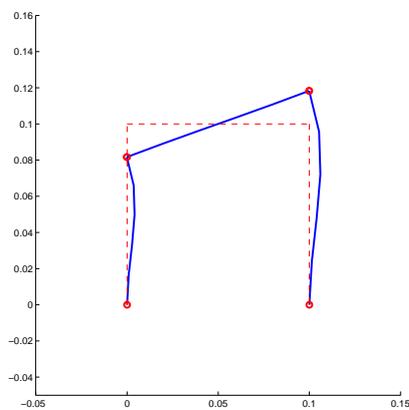
    0.0106
    2.1290
    3.5851

```

In figure 3.9 the mode shapes of the second (2129 Hz) and third (3585 Hz) natural frequency are displayed. These natural frequencies are much higher than the first natural frequency, so in most cases the 1-DOF approximation of section 3.1 is justified.



(a) Second natural frequency (2129 Hz).



(b) Third natural frequency (3585 Hz).

Figure 3.9: Mode shapes of the second and third natural frequency.

## 4 Three-dimensional model

Finally, a three-dimensional model will be described in which the mass-spring system will be modelled using spatial beam elements. Figure 4.1 shows a spatial frame model of the sample system of figure 1.2. It is modelled using three spatial beam elements. The element numbers and the nodal numbers correspond with the two-dimensional frame model in figure 3.1.

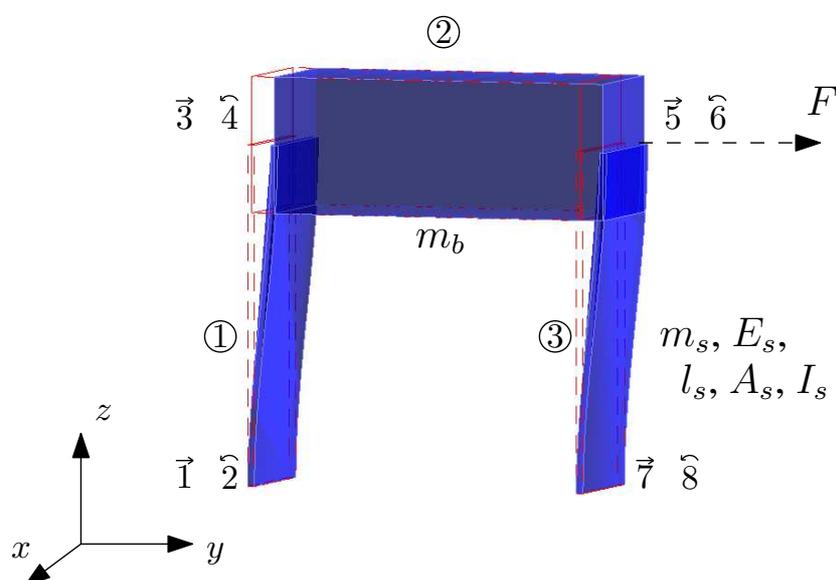


Figure 4.1: Spatial frame model with elastic beams.

### Spatial beam element (BEAM)

Figure 4.2 shows a spatial beam element in an  $(x, y, z)$  inertial coordinate system. The configuration of the element is determined by the position vectors  $\mathbf{x}^p$  and  $\mathbf{x}^q$  of the end nodes and the angular orientation of orthogonal triads  $(\mathbf{n}_{x'}, \mathbf{n}_{y'}, \mathbf{n}_{z'})$  rigidly attached to each end point. In the undeflected state the triads coincide with the axis  $pq$  and the principle axes of its cross section. The rotation part of the motion of the (flexible) beam is described by the rotation of the triads  $(\mathbf{n}_{x'}, \mathbf{n}_{y'}, \mathbf{n}_{z'})$  which are determined by rotation matrices  $\mathbf{R}^p$  and  $\mathbf{R}^q$ . If the beam is rigid then the rotation matrices are identical and in the initial undeflected state they are equal to the identity matrix.

The nodal coordinates for the spatial beam element consist of translational and rotational coordinates. The six translational coordinates are from two position vectors  $\mathbf{x}^p$  and  $\mathbf{x}^q$  describing the position of the undeflected beam in the fixed inertial coordinate system. There are also six independent rotational coordinates as the orientation of each rotational node in three dimensions is given by three independent rotation coordinates collected in the vectors  $\boldsymbol{\lambda}^p$  and  $\boldsymbol{\lambda}^q$  respectively. These coordinates describe the angular orientation of the triads  $(\mathbf{n}_{x'}, \mathbf{n}_{y'}, \mathbf{n}_{z'})$  at the nodes  $p$  and  $q$ . Hence the spatial beam has a total of twelve nodal coordinates. As a rigid body the spatial beam has six degrees of freedom, so we can define  $12 - 6 = 6$  independent deformation parameters. In addition to the deformation mode coordinate  $e_1^{(k)}$

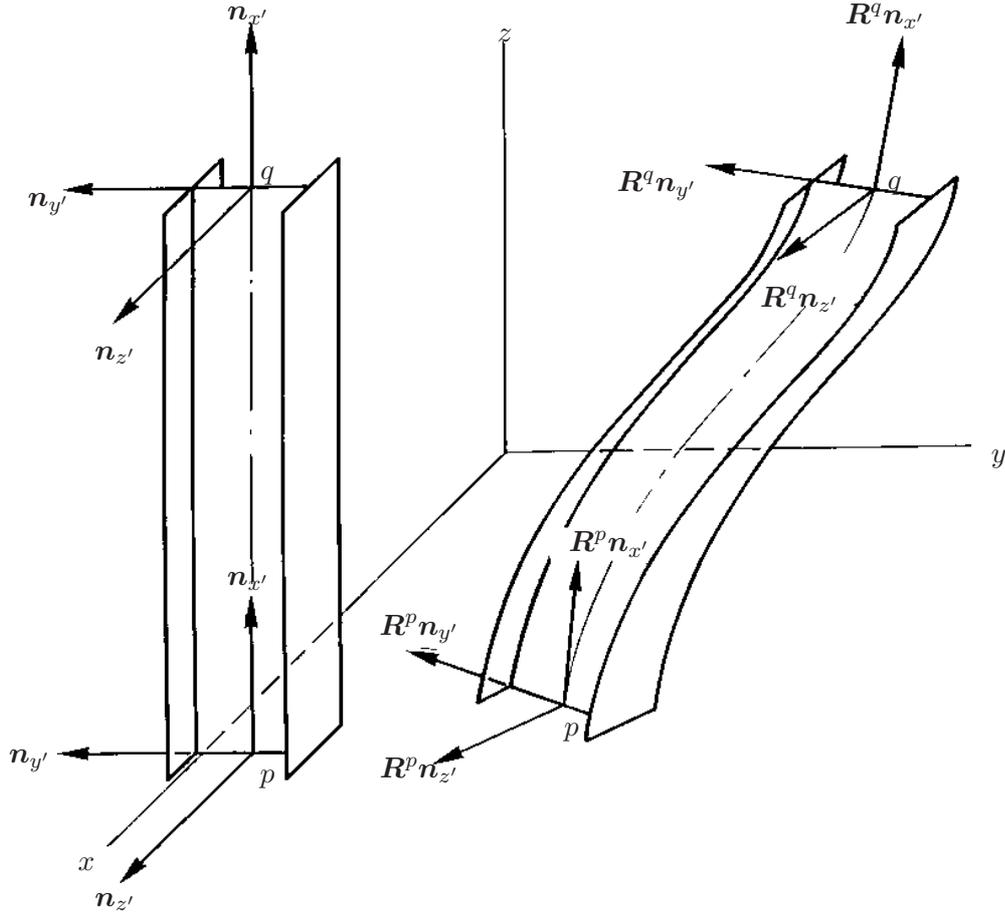


Figure 4.2: Beam element, initial and deformed state.

representing the elongation, five deformation parameters can be defined for the beam element in terms of the relative rotations at the end nodes  $p$  and  $q$ :

$$\text{elongation:} \quad e_1^{(k)} = l^{(k)} - l_0^{(k)}, \quad (4.1)$$

$$\text{torsion:} \quad e_2^{(k)} = \frac{1}{2} l_0^{(k)} [(\mathbf{R}^p \mathbf{n}_{z'}, \mathbf{R}^q \mathbf{n}_{y'}) - (\mathbf{R}^p \mathbf{n}_{y'}, \mathbf{R}^q \mathbf{n}_{z'})], \quad (4.2)$$

$$\text{bending:} \quad e_3^{(k)} = -(\mathbf{R}^p \mathbf{n}_{z'}, \mathbf{l}^{(k)}), \quad (4.3)$$

$$e_4^{(k)} = (\mathbf{R}^q \mathbf{n}_{z'}, \mathbf{l}^{(k)}), \quad (4.4)$$

$$e_5^{(k)} = (\mathbf{R}^p \mathbf{n}_{y'}, \mathbf{l}^{(k)}), \quad (4.5)$$

$$e_6^{(k)} = -(\mathbf{R}^q \mathbf{n}_{y'}, \mathbf{l}^{(k)}), \quad (4.6)$$

where the vector  $\mathbf{l}^{(k)}$  is defined as

$$\mathbf{l}^{(k)} = \mathbf{x}^q - \mathbf{x}^p = [x^q - x^p, y^q - y^p, z^q - z^p]. \quad (4.7)$$

The geometric meaning of the bending deformations  $e_3^{(k)}$ ,  $e_4^{(k)}$ ,  $e_5^{(k)}$ , and  $e_6^{(k)}$  is visualised in Fig. 4.3. Note that the deformations are invariant with respect to rigid body movements of the element.

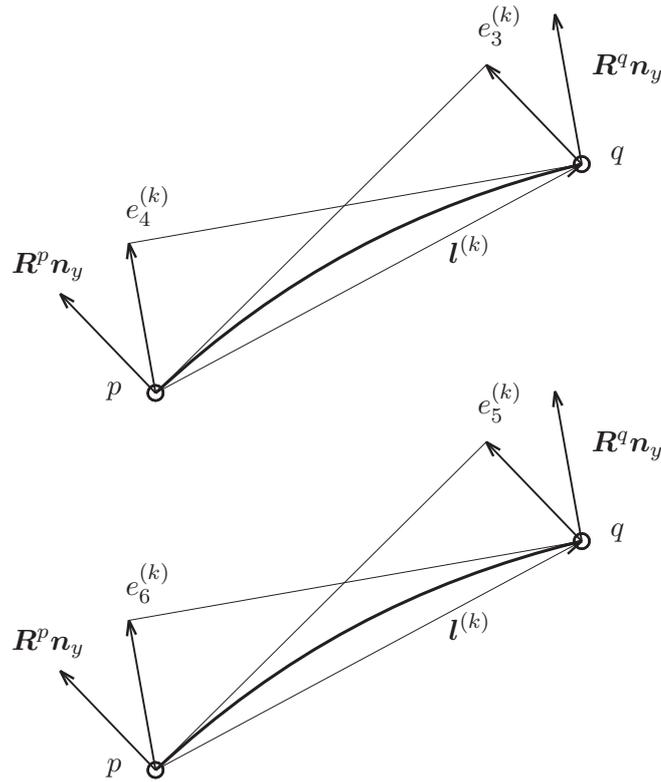


Figure 4.3: Visualisation of bending deformations  $e_3$ ,  $e_4$ ,  $e_5$ , and  $e_6$  of the spatial beam element.

In the description of the spatial beam it was mentioned that each rotational node is described by three independent coordinates. However, SPACAR uses internally 4 orientational coordinates (called Euler parameters denoted by  $\lambda_i$ ) for each rotational node which are related by  $\sum_{i=0}^3 (\lambda_i)^2 = 1$ . From this point of view each rotational node adds 4 nodal coordinates (instead of 3) to NX and one (extra) relative constraint to NEO. Effectively, this does not affect the outcome of equation 2.13 for NDOF as both NX and NEO are incremented with the number of rotational nodes. However, to avoid misunderstanding it has to be specified in which way the nodal coordinates and deformations are counted. In the SPACAR log file both values can be found, i.e. 4 nodal coordinates and 1 relative constraint or just 3 nodal coordinates per rotational node. In this tutorial the latter approach is preferred.

## 4.1 One degree of freedom model

### Kinematic section of the input file

For the spatial frame model in figure 4.1 the number of degrees of freedom has to satisfy equation (2.13). Similar to the two-dimensional model there are four translational and four rotational nodes. As each translational and rotational node has three coordinates, the total number of coordinates is  $NX = 28$ . Again two translational and two rotational nodes are clamped, so the number of absolute constraints is  $NXO = 12$ . The goal is to create a one degree of freedom system, so NEO has to be  $24 - 12 - 1 = 11$ . Six prescribed deformations are contributed by beam 2, so 5 deformations have to be prescribed for beam 1 and 3. They have a total of 12 deformations, which means 7 deformations need to be released.

In this example the horizontal position, in the  $y$ -direction, of translational node 3 is chosen as degree of freedom. Then the following commands for the kinematic section of the input file can be used:

```
BEAM 1 1 2 3 4 0.0 1.0 0.0
BEAM 2 3 4 5 6 0.0 0.0 1.0
BEAM 3 5 6 7 8 0.0 1.0 0.0
```

The definition of the spatial beam is partly similar to the definition of the planar beam: The first integer parameter is the element number and next four integers specify the translational and rotational node points at either end of the beam. The extra three real numbers may be used to define the initial direction of the principal  $y'$ -axis of the beam's cross-section (see figure 4.2). The beam's principal  $x'$ -axis coincides with the undeformed element axis and is directed from end point  $p$  to end point  $q$ , so it is defined from the initial position of the beam. When the principal  $y'$ -axis is defined, the principal  $z'$ -axis is also known.

```
X 1 0.0 0.0 0.0
X 3 0.0 0.0 0.1
X 5 0.0 0.1 0.1
X 7 0.0 0.1 0.0
```

The definition of the initial coordinates is as before, except that three Cartesian node coordinates in the  $x$ ,  $y$ , and  $z$ -directions have to be specified.

```
FIX 1
FIX 2
FIX 7
FIX 8
```

The four FIX keywords impose the  $NXO = 12$  constraints for the clamped nodes.

```
RLSE 1 5 6
RLSE 3 2 3 4 5 6
```

The two RLSE keywords completely release the bending modes of the leaf springs that are associated with the bending illustrated in figure 4.1, i.e. the deformation mode coordinates  $e_5^{(1)}$ ,  $e_6^{(1)}$ ,  $e_5^{(3)}$ , and  $e_6^{(3)}$ . In addition the torsion and remaining bending modes of element 3 are released as well in accordance with the local cut in the corresponding leaf spring. Finally the degree of freedom is specified and the kinematic section of the input file is concluded:

```
DYNX 3 2

END
HALT
```

### Dynamic section of the input file

The dynamic properties of the spatial frame model start with the inertia properties:

```
XM 3 0.103
XM 5 0.103
EM 1 0.07065
EM 3 0.07065
```

Note that the rotational inertia are not included here. If rotational inertia are specified in three dimensions, the XM command is followed by up to six real numbers to specify all inertia components  $J_{xx}$ ,  $J_{xy}$ ,  $J_{xz}$ ,  $J_{yy}$ ,  $J_{yz}$ , and  $J_{zz}$  [3, Sect. 2.3].

```
ESTIFF 1 1890000. 0.05895 51.030 0.039375
ESTIFF 3 1890000. 0. 0. 0.039375
```

For the spatial beam element the ESTIFF keyword has four real parameters that specify the axial rigidity  $EA$ , the torsional rigidity  $GI_t$  and the flexural rigidities  $EI_{y'}$  and  $EI_{z'}$  [3, Sect. 2.3].

Torsional stiffness  $S_t$  is not straightforward for beams with a non-circular cross section. The torsional stiffness in SPACAR is defined as

$$S_t = \frac{GI_t}{l^3}, \quad (4.8)$$

where  $G$  is the shear modulus of the material and  $I_t$  is Saint-Venant's torsion constant which is the polar moment of inertia for a circular cross section. For elements having a rectangular cross section it can be determined by [14]

$$I_t = wt^3 \left( \frac{1}{3} - 0.21 \frac{t}{w} \left( 1 - \frac{(t/w)^4}{12} \right) \right), \quad (4.9)$$

where  $w$  is the width and  $t$  is the thickness of the cross section. This approximation only holds for a beam with both ends free. The twisting angle  $\theta_l$  of a prismatic beam clamped at one end can be described with

$$\theta_l = \frac{T}{GI_t} \left( l - \frac{1}{\alpha} \right), \quad (4.10)$$

where  $T$  is the twisting moment,  $GI_t$  is the torsional rigidity,  $l$  represents the length of the prismatic beam and  $\frac{1}{\alpha}$  represents the influence of the clamped end on the twisting angle. For a narrow rectangular cross section where the width  $w$  is large in comparison to the thickness  $t$ , it can be written as

$$\frac{1}{\alpha} = \frac{w}{\sqrt{24(1-\nu)}}. \quad (4.11)$$

Here  $\nu$  denotes the Poisson's ratio of the material, which is approximately 0.3 for steel. Now,  $\frac{1}{\alpha}$  can be written as a fraction of the width  $w$  of the beam

$$\frac{1}{\alpha} = 0.244w. \quad (4.12)$$

Equation 4.10 can be rewritten with an equivalent torsional rigidity  $(GI_t)_{\text{eq}}$  as

$$\theta_l = \frac{T}{(GI_t)_{\text{eq}}}, \quad (4.13)$$

with

$$(GI_t)_{\text{eq}} = \frac{1}{1 - \frac{1}{\alpha l}} GI_t = \frac{1}{1 - 0.244 \frac{w}{l}} GI_t. \quad (4.14)$$

If  $l = 0.244w$ , the denominator of the fraction in the righthand side of equation 4.14 becomes zero, implying an infinitely large torsional rigidity. Equation 4.14 is valid for a beam which is clamped at one end and subjected to a torque  $T$  at the free end. From the theory of elasticity it is known that warping is only constrained near the clamped end. A good estimate is obtained if the adjoining element is modelled rigidly for torsion over a length  $l = 0.244w$ .

In the current model torsion does not play a dominant role and only one element is used for each leaf spring. Note that the cut in the second leaf spring should decrease the torsional stiffness and  $z'$ -bending stiffness as much as possible. In this analysis, the stiffnesses of these deformation modes are taken zero.

```
EDAMP 1 0.3654 0.0000065 0.00019 0.0000053
EDAMP 3 0.3654 0. 0. 0.0000053
```

The longitudinal, torsional and bending damping values (EDAMP) can be calculated analogously as for the planar beam with equation 2.14.

The dynamic part of the input file is concluded with:

```
END
HALT
```

**Input-output section of the input file**

```
INPUTF 1 5 2
```

```
OUTX 1 3 1
```

```
OUTX 2 3 2
```

```
OUTX 3 3 3
```

```
OUTX 4 5 1
```

```
OUTX 5 5 2
```

```
OUTX 6 5 3
```

```
END
```

```
END
```

As outputs of the system, all coordinates of the translational nodes 3 and 5 are chosen.

**Optional visualisation section of the input file**

```
VISUALIZATION
```

```
BEAMVIS 1 0.0005 0.02
```

```
BEAMVIS 2 0.013 0.02
```

```
BEAMVIS 3 0.0005 0.02
```

```
VIBRATIONMODE 1
```

```
VIBREND 7.8540
```

```
ENLARGFACTOR 0.01
```

```
TRANSPARENCY 0.6
```

**SPACAR results**

For this three-dimensional model the natural frequency, vibration mode (figure 4.4), state space matrices, transfer functions and the Bode plot (figure 4.5) from input force to  $y$ -displacement of node 3 are generated.

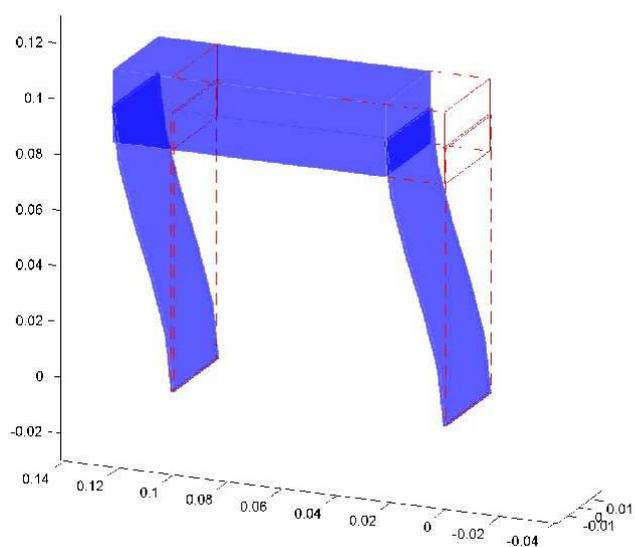


Figure 4.4: First vibration mode of the three-dimensional model.

```
>> spacar(7,'pf3d');
>> sqrt(eig(k0,m0))/2/pi
```

```
10.6448
```

```
>> spavisual('pf3d');
```

```
>> spacar(9,'pf3d');
>> getss('pf3d')
```

```
a =
      x1      x2
x1      0      1
x2  -4473  -0.6021
```

```
b =
      u1
x1      0
x2  4.734
```

```
c =
      x1  x2
y1      0  0
y2      1  0
y3      0  0
y4      0  0
y5      1  0
y6      0  0
```

```
d =
      u1
y1      0
y2      0
y3      0
y4      0
y5      0
y6      0
```

Continuous-time model.

```
>> G=tf(getss('pf3d'))
```

Transfer function from input to output...

```
#1: 0
```

```
#2: -----
      4.734
      s^2 + 0.6021 s + 4473
```

```
#3: 0
```

```

#4: 0

#5: -----
      4.734
      s^2 + 0.6021 s + 4473

#6: 0

```

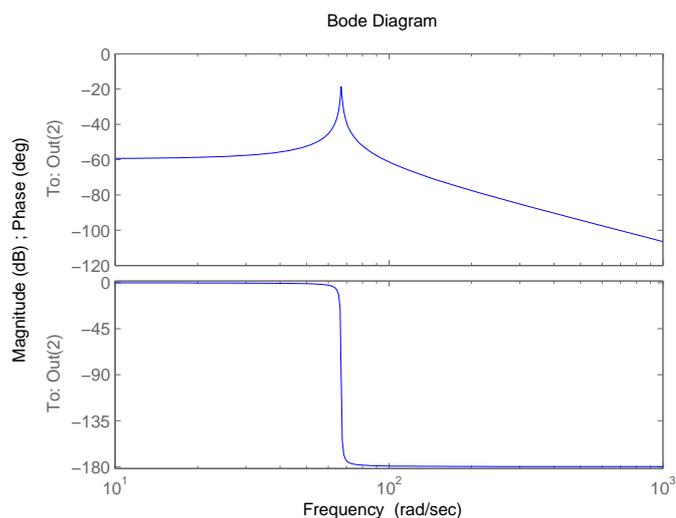


Figure 4.5: Bode plot of transfer from input force to  $y$ -displacement of node 3 of the three-dimensional model.

## 4.2 Six degrees of freedom model

It is the responsibility of the user to select the degrees of freedom carefully. In the previous example only one degree of freedom was defined and hence only one vibrational mode (with one natural frequency) will be found. Suppose that e.g. all 12 deformation modes of the spring leaves are released. Then there are only  $NEO = 18 - 12 = 6$  relative constraints and the number of degrees of freedom according to equation (2.13) is

$$NDOF = NX - NXO - NEO = 24 - 12 - 6 = 6. \quad (4.15)$$

The previous RLSE and DYNX commands have to be replaced with:

```

RLSE 1
RLSE 3 2 5 6

DYNX 3
DYNE 3 1 3 4

```

where a keyword with only the element number defaults to all deformation parameters of that element. So all coordinates of translational node 3 and three deformations of element 3 are chosen as dynamic degrees of freedom. Choosing the 3 independent coordinates of rotational node 4 is also possible, but it would result in an  $A$ -matrix without physical meaning.

**SPACAR results**

The result of the extra degrees of freedom is that more natural frequencies and their vibration modes can be found. After a mode 7 run, the natural frequencies can be obtained similarly as before, but the matrices  $k_0$  and  $m_0$  have to be reshaped first.

```
>> sqrt(eig(reshape(k0,nddof,nddof),reshape(m0,nddof,nddof)))/2/pi

1.0e+003 *

0.0038
0.0106
0.1924
2.1263
2.1290
6.2350
```

In figure 4.6 on page 34 the six mode shapes of the 6-DOF system can be found. As the figure as well as the list of frequencies show, a rather unphysical first mode at 3.8 Hz appears. It is left as an exercise to the reader to investigate the occurrence of this mode in more detail.

Using mode 9, the state space matrices and consequently the transfer functions for the system can be determined. Only the transfer function from input force to  $y$ -displacement of node 3 is given, in figure 4.7 the Bode plot of this transfer function is displayed.

```
>> spacar(9,'pf3d6dof');
>> tf(getss('pf3d6dof'))
```

Transfer function:

```
4.734 s^4 + 330.9 s^3 + 1.692e9 s^2 + 5.914e10 s + 1.512e17
-----
s^6 + 70.5 s^5 + 3.574e8 s^4 + 1.271e10 s^3 + 3.194e16 s^2 + 1.929e16 s + 1.429e20
```

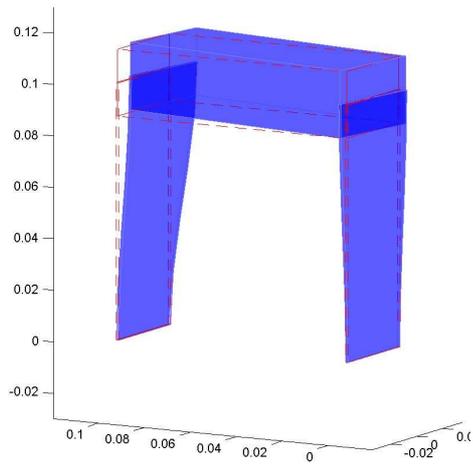
The Bode plot looks like one from a 2<sup>th</sup> order system, even though the transfer function is of 6<sup>th</sup> order. So probably there are some pole/zero pairs in this transfer which cancel out. You can try to simplify this transfer by using MATLAB's `minreal` command:

```
>> minreal(tf(getss('pf3d6dof')), 1e-4)
```

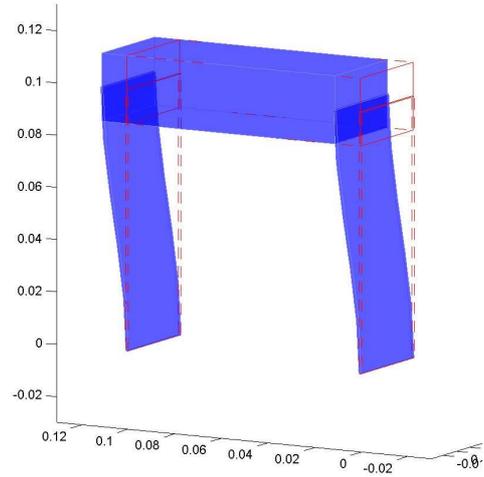
Transfer function:

```
4.734
-----
s^2 + 0.6021 s + 4473
```

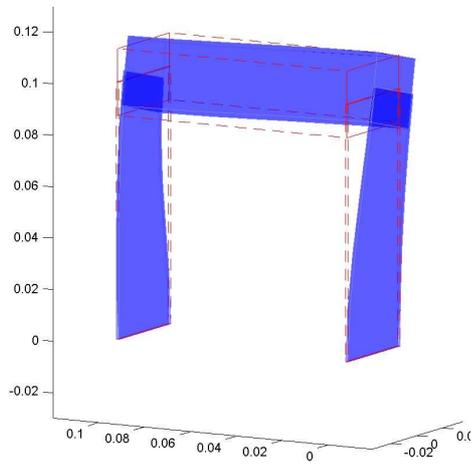
When the “tolerance” parameter  $10^{-4}$  is set sufficiently large, indeed a second order transfer function is obtained that equals the corresponding transfer function from the 1-DOF model in the previous section.



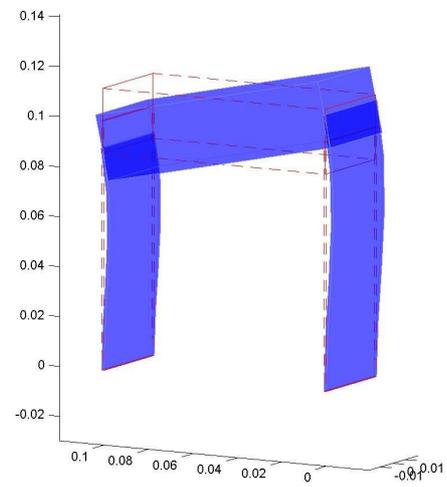
(a) First natural frequency (3.8 Hz).



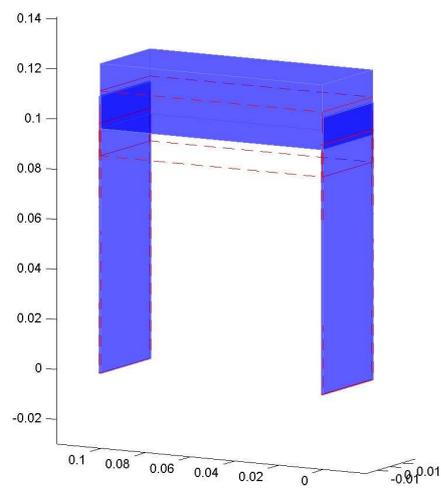
(b) Second natural frequency (10.6 Hz).



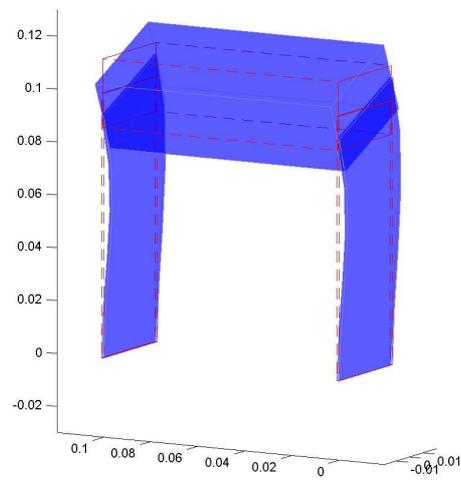
(c) Third natural frequency (192.4 Hz).



(d) Fourth natural frequency (2126 Hz).



(e) Fifth natural frequency (2129 Hz).



(f) Sixth natural frequency (6235 Hz).

Figure 4.6: Mode shapes of the 6-DOF three dimensional model.

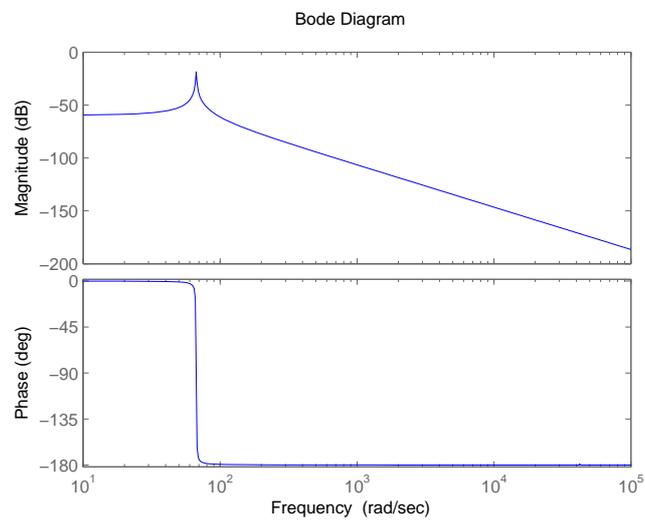


Figure 4.7: Bode plot of transfer from input force to  $y$ -displacement of node 3 of the 6-DOF three-dimensional model.



# A SPACAR software

## A.1 Introduction

The computer program SPACAR is based on the non-linear finite element theory for multi-degree of freedom mechanisms as described in Jonker's lecture notes on the Dynamics of Machines and Mechanisms [8]. The program is capable of analysing the dynamics of planar and spatial mechanisms and manipulators with flexible links and treats the general case of coupled large displacement motion and small elastic deformation. The motion can be simulated by solving the complete set of non-linear equations of motion or by using the so-called perturbation method. The computational efficiency of the latter method can be improved further by applying modal techniques.

In this chapter an outline of the SPACAR package for use with MATLAB is given in the next section. A special visualisation tool, SPAVISUAL, is described in Section A.3. Installation notes for SPACAR are given in Section A.4.

## A.2 SPACAR & MATLAB

SPACAR can run different types of analysis called modes. Only mode 7, 8 and 9 will be used in this tutorial. For information about the other modes, see the SPACAR manual [3].

In mode=7 eigenvalues (frequencies) and corresponding eigenvectors are computed for a static equilibrium configuration. The associated frequency equation of the undamped system is given by

$$\det \left( -\omega_i^2 \bar{M}_0^{dd} + \mathbf{K}_0^{dd} + \mathbf{N}_0^{dd} + \mathbf{G}_0^{dd} \right) = 0, \quad (\text{A.1})$$

where the quantities  $\omega_i$  are the natural frequencies of the system.  $M_0^{dd}$  is the mass matrix and  $\mathbf{K}_0^{dd}$ ,  $\mathbf{N}_0^{dd}$  and  $\mathbf{G}_0^{dd}$  are the structural, dynamic and geometric stiffness matrices respectively.

In mode=8 a linear buckling analysis is carried out for a static equilibrium configuration. Critical load parameters  $\lambda_i$  are determined by solving the eigenvalue problem.

$$\det(\mathbf{K}_0^{dd} + \lambda_i \mathbf{G}_0^{dd}) = 0, \quad (\text{A.2})$$

where,

$$\lambda_i = \mathbf{f}_i / \mathbf{f}_0. \quad (\text{A.3})$$

Here,  $\mathbf{K}_0^{dd}$  is the structural stiffness matrix and  $\mathbf{G}_0^{dd}$  is the geometric stiffness matrix due to the reference load  $\mathbf{f}_0$  giving rise to the reference stresses  $\boldsymbol{\sigma}_0$ .  $\mathbf{f}_i$  represents the buckling load that corresponds with  $\lambda_i$ .

In mode=9 locally linearized equations for control system analysis are computed for a static equilibrium configuration. The linearized equations can be transformed into the linearized state space form:

$$\begin{aligned} \delta \dot{\mathbf{z}} &= \mathbf{A} \delta \mathbf{z} + \mathbf{B} \delta \mathbf{u}, \\ \delta \mathbf{y} &= \mathbf{C} \delta \mathbf{z} + \mathbf{D} \delta \mathbf{u}, \end{aligned} \quad (\text{A.4})$$

where  $\mathbf{A}$  is the state matrix,  $\mathbf{B}$  the input matrix,  $\mathbf{C}$  the output matrix and  $\mathbf{D}$  the feed through matrix. The state vector  $\delta \mathbf{z}$  is defined by  $\delta \mathbf{z} = [\delta \mathbf{q}^{dT}, \delta \dot{\mathbf{q}}^{dT}]^T$ , where  $\delta \mathbf{q}^d$  is the vector of dynamic degrees of freedom. The matrices  $\mathbf{B}$ ,  $\mathbf{C}$  and  $\mathbf{D}$  depend on the chosen input vector  $\delta \mathbf{u}$  and the output vector  $\delta \mathbf{y}$ . Details of the linearization are discussed in chapter 12 of [8].

## Definition of a mechanism model

A model of a mechanism must be defined in an input file of file type (or file name extension) `.dat`. This input file consists of a number of keywords with essential and optional parameters. The input file can be generated with any text editor. In Appendix B the meaning of the keywords and their parameters is discussed in detail.

## Running SPACAR in the MATLAB environment

Once the mechanism is defined and this information is saved to a `.dat` input file, SPACAR can be activated with the MATLAB command

```
>> spacar(mode, 'filename')
```

Here, `filename` is the name of the input file, without the extension `.dat`. The `filename` is limited to 20 characters from the set “0”–“9”, “a”–“z”, “A”–“Z” and “\_”, so it can not include drive or path specifications.

During the computation a plot of the mechanism is shown in a separate window. While the simulation is running an `Abort` button is activated in the plot area. Pressing this button will terminate the simulation (possibly after some delay). To speed up the computation, the plot can be disabled by specifying the mode with a minus sign, e.g. `mode=-9`. The plotting utility `spadraw` can also be used after the simulation to visualise the results, see page 41.

During the computations the results are stored in one or more data files and in MATLAB arrays. A `log` file is always created when SPACAR starts processing the input `.dat` file. This `log` file contains an analysis of the input and possible errors and warnings. It is described in more detail on page 40. Many errors in the input file do not lead to an early termination of the SPACAR computation, but nevertheless give unusable results. Therefore it is advisable to check the `log` file for unexpected messages.

All other data files are so-called SPACAR Binary data Files (SBF), which implies that these are in a binary format and can not be read easily by a user. Therefore, utilities are provided to read and modify data in these files, see page 41. Depending on the mode up to three binary output files may be created.

For all modes a SPACAR Binary Data file with filename identical to the input file and extension `sbd` is written. The contents of this file is also stored in MATLAB arrays, that are of course immediately available in the MATLAB workspace e.g. to be visualised with the standard MATLAB graphics commands, such as `plot`. The following variables are created or overwritten:

<code>mode</code>	SPACAR mode number	
<code>ndof</code>	number of kinematic DOFs	
<code>nddof</code>	number of dynamic DOFs	
<code>nx</code>	number of coordinates	
<code>ne</code>	number of deformation parameters	
<code>nxp</code>	number of fixed, calculable, input and dynamic coordinates	
<code>nep</code>	number of fixed, calculable, input and dynamic deformation parameters	
<code>lnp</code>	location matrix for the nodes	*1
<code>le</code>	location matrix for the elements	*1
<code>ln</code>	connection matrix for the nodes in the elements	*2
<code>it</code>	list of element types	*2
<code>time</code>	time column vector	
<code>x</code>	coordinates (nodal displacements)	
<code>xd</code>	nodal velocities	
<code>xdd</code>	nodal accelerations	
<code>fx</code>	prescribed nodal forces/moments	
<code>fxtot</code>	reaction forces/moments	
<code>e</code>	generalized deformations	

ed	velocities of generalized deformations	
edd	accelerations of generalized deformations	
sig	generalized stress resultants	
de	first order geometric transfer function for the deformations $\mathbf{D}\mathcal{F}^{(e)}$	*3
dx	first order geometric transfer function for the coordinates $\mathbf{D}\mathcal{F}^{(x)}$	*3
d2e	second order geometric transfer function for the deformations $\mathbf{D}^2\mathcal{F}^{(e)}$	*3
d2x	second order geometric transfer function for the coordinates $\mathbf{D}^2\mathcal{F}^{(x)}$	*3
xcompl	location vector for directional nodal compliances	*4
rxyz	initial orientations of elements	*2

Notes:

- \*1 The two location matrices provide information to find the location of a specific quantity in the data matrices:

lnp	location matrix for the nodes. The matrix element $lnp(i, j)$ denotes the location of the $j^{\text{th}}$ coordinate ( $j=1..4$ ) of node $i$ .
le	location matrix for the elements. The matrix element $le(i, j)$ denotes the location of the $j^{\text{th}}$ generalised deformation ( $j=1..6$ ) of element $i$ .

The locations of undefined or unused coordinates and deformations equal zero.

For example, the  $x$ - and  $y$ -coordinates of node 7 can be shown as function of time in a graph by typing

```
>> plot(time, x(:, lnp(7, 1:2)))
```

and the first generalised stresses in elements 1, 2 and 3 can be plotted by typing

```
>> plot(time, sig(:, le(1:3, 1)))
```

Obviously, storage in the  $x$ ,  $xd$ ,  $xdd$ ,  $fx$ ,  $e$ ,  $ed$ ,  $edd$  and  $sig$  matrices is like  $x(t, k)$  where  $t$  is the time step and  $k$  ranges from 1 to  $nx$  for  $x$ ,  $xd$ ,  $xdd$  and  $fx$ ,  $fxtot$  and from 1 to  $ne$  for  $e$ ,  $ed$ ,  $edd$  and  $sig$ , respectively.

- \*2 The variables  $ln$ ,  $it$  and  $rxyz$  are mainly intended for internal use in the drawing tool `spadraw`. More user-friendly information is available in the `log` file, page 40.
- \*3 The (large) variables  $de$ ,  $dx$ ,  $d2e$  and  $d2x$  are only created if the parameters of the `LEVELLOG` are set accordingly, Sect. B.2.
- \*4 After a linearisation run (`mode=8`) directional nodal compliances (inverse stiffnesses) are computed. Using the location matrix,  $xcompl(lnp(i, j))$  gives this quantity for the  $j^{\text{th}}$  coordinate ( $j=1..4$ ) of node  $i$ .

After a linearization run (`mode=3, 4, 7, 8` or `9`) the coefficient matrices are stored in a SPACAR Binary Matrix file with extension `sbm`. The accompanying MATLAB matrices are:

m0	reduced mass matrix $M_0$	*5
b0	input matrix $B_0$	*5, *6
c0	velocity sensitivity matrix $C_0$	*5
d0	damping matrix $D_0$	*5
k0	structural stiffness matrix $K_0$	*5
n0	geometric stiffness matrix $N_0$	*5
g0	geometric stiffness matrix $G_0$	*5

Notes:

\*5 Storage of the time-varying matrices is in a row for each time step, so in  $m0(t, k)$  index  $t$  is the time step and  $k$  ranges from 1 to  $ndof \times ndof$ . To restore the matrix structure at some time step type e.g. `reshape(m0(t, :), ndof, ndof)`.

\*6 Only available for `mode=4` and `9`.

In `mode=2, 3, 4` and `9` a so-called `ltv` file is created. The contents of this file varies and is not automatically imported to the MATLAB workspace. From a `mode=2` run the following data is available (the names indicate the identities of the data used in the file; identities marked with “\*” are available at each time step):

NNOM	number of (actuator) inputs	
NY	number of outputs	
T	time	*
U0	nominal input for the desired motion	*
Y0	reference output of the desired motion	*

In the addition the linearization runs yield additional setpoints, state space matrices and other data in the `ltv` file (not all identities are always present):

NNOM	number of (actuator) inputs	
NX	number of states ( $2 \times ndof$ )	
NU	number of inputs (length of U0)	
NY	number of outputs (length of Y0)	
NRBM	number of rigid body DOF's	
NYS	number of outputs with 2 <sup>nd</sup> order expression	
DF'T	direct feedthrough flag ( $D \neq 0$ )	
X0	initial state vector	
T	time	*
A	state space system matrix	*
B	state space input matrix	*
C	state space output matrix	*
D	state space direct feedthrough matrix	*
G	second order output tensor	*
M0	mass matrix $M_0$	*
C0B	combined damping matrix $C_0 + D_0$	*
K0B	combined stiffness matrix $K_0 + N_0 + G_0$	*
SIG0	generalized stress resultants	*

The `getss` tool can be used to read the state space matrices from the `ltv` file, see page 41. Other utilities are available to use parts of these data in a SIMULINK environment, e.g. to read setpoints or to simulate a Linear Time-Varying (LTV) system, as is described in the manual.

## The log file

The `log` file contains an analysis of the input and possible errors and warnings that are encountered. The error and warning messages are explained in more detail in the SPACAR manual. The other output can be separated into a number of blocks.

The first lines indicate the version and release date of the software and a copyright note.

Next the lines from the input file read by the KIN module are shown (not showing comments present in the input file), see also Sect. B.2. From the analysis is written:

- The elements used in this model. The deformations of all elements are shown with the internal numbers according to the `le` array and the classification of each deformation: O = fixed, C = calculable and M = DOF.
- The nodal point information with the internal numbers of the coordinates according to the `lnp` array and the classification as above.

- Finally a list shows the degrees of freedom. Dynamic degrees of freedom are indicated.

The DYN module reads the next data block and processed input lines are shown. From the analysis we get

- The numbers NEO, NEMM, NEM and NEC indicate the numbers of deformations in each class as explained in the lecture notes [8].
- The numbers NXO, NXC, NXMM and NXM indicate the numbers of position coordinates in each class as explained in the lecture notes [8].
- The stiffness, damping and mass of the elements.
- The nodal point forces, mass and gyroscopic terms.
- The total mass of the system.

The zeroth, first, second and third order transfer functions are shown next, each for the position parameters and deformation parameters, respectively. The amount of output can be controlled by the keyword OUTLEVEL in the input file.

Next for a forward analysis (`mode=1` and `mode=4`) the name of the integrator and accuracy settings are shown. Finally a list with all time steps and the number of internal iterations are given. For an inverse dynamics analysis the trajectories and input/output definitions are read and analysed. In case of `mode=3` the name of the data file of the previous `mode=2` is shown. In case of `mode=7` the eigen values (frequencies) and normalized eigenvectors of the state system matrix are shown. In case of `mode=8` load multipliers and normalized buckling modes are presented. In addition the vector of directional nodal compliances is shown.

## SPACAR Binary data Files

Some utilities are available to show, check, load or replace the data in SPACAR Binary data Files (SBF) files. These are files with extensions `sbd`, `sbm` and `ltv`.

`checksbf` check and shows the contents of a SPACAR Binary data File. The output for each variable is the name (“Id”), the type (1 for integer, 2 for real, 3 for text) and the size (number of rows and columns). First the “header” variables are shown with their value. Long vectors may be truncated. Between TDEF and TDAT the time-varying data is given. The number of time steps equals the number of rows specified for TDEF.

`getfrsbf` extract a variable from a SPACAR Binary data File. The “Id” must be specified and for time-varying data the time step as well.

`getss` extracts the state space formulation from a SPACAR `ltv`-file.

`repinsbf` replaces the value of a variable in a SPACAR Binary data File. The “Id” must be specified and for time-varying data the time step as well.

`loadsbd` loads all data from a SPACAR Binary Data (`sbd`) file into MATLAB’s workspace.

`loadsbm` loads all data from a SPACAR Binary Matrix data (`sbm`) file into MATLAB’s workspace.

`combsbd` combines data from two or more SPACAR Binary Data (`sbd`) files into a single output file. The specified output file is overwritten without a warning.

`spadraw` is the plotting utility used internally by SPACAR. It can also be used to visualize results after a simulation has been completed.

For all utilities additional online help is available by typing `help` command at the MATLAB prompt.

## Limitations

The SPACAR package has some built-in limitations on the size of the manipulators that can be analysed. Table A.1 shows the limits for the so-called “Student version” that can be downloaded as describes in Appendix A.4. In case your requirements are larger, you need to contact the authors. The licence for the freely downloadable software is time limited.

Maximum number of coordinates/deformations	175
Maximum number of DOFs	20
Maximum number of elements/nodal points	50
Maximum number of inputs	12
Maximum number of outputs	25

Table A.1: Built-in limitations of the “Student version” of the SPACAR package.

## A.3 SPAVISUAL

SPAVISUAL is the visualization tool for SPACAR. It can visualize deformation, vibration and buckling modes. SPAVISUAL shows beams, trusses, and hinges in 2D as well as in 3D. It works with default settings which can be adjusted by the user. The only input of SPAVISUAL is a filename. This file has to be a `.dat` file which has been analysed with SPACAR. This is necessary because SPAVISUAL needs the `.sbd` files for the deformation modes and also the `.sbm` files for the vibration and the buckling modes. There are a couple of keywords that can adjust the default settings. These keywords are listed in section B.5.

SPAVISUAL is a standalone function in MATLAB. To run SPAVISUAL the user has to type the next command.

```
>> spavisual(filename)
```

Here `filename` refers to the `.dat`-file that is executed by SPACAR.

## A.4 Download & install

### Prerequisites

Before installing SPACAR on a computer system it is advisable to check that the system is suitable of running the software and to have MATLAB installed. This SPACAR version has been developed and tested with MATLAB 7.0.4 and SIMULINK 6.2 (Release 14SP2). It is expected to work with any modern version of MATLAB/SIMULINK since R12, but in case of problems we can offer only limited support. The system requirements depend heavily on the version of MATLAB you are using. Consult the accompanying Installation Guide or check The Mathworks. You may expect that SPACAR will run on any Microsoft 32-bit Windows PC on which MATLAB/SIMULINK are running. Only the base systems of MATLAB and SIMULINK are required to run SPACAR, but additional toolboxes like the Control System Toolbox may be helpful to develop and analyse control systems. The installation of SPACAR uses less than 4 MB extra disk space. The SPACAR files are stored in ZIP-archives or, in Microsoft Windows XP, a compressed folder. In Windows XP you can easily open such archives, but of course you may chose to use your favourite unzipper. The ZIP-archives can be downloaded from <http://www.wa.ctw.utwente.nl/Software/SPACAR/>. In addition to the software there is a ZIP-archive with the data files that are used in this tutorial available on the SPACAR-site.

## Installation

First of all, you should create a subdirectory e.g. `\Matlab\Toolbox\Spacar`. Next, you extract the files from the SPACAR software ZIP-archive `spacar2008_bin.zip` into this subdirectory. There are three types of files:

- Files with the extension `.dll` are the actual executables of the SPACAR package. The original SPACAR-code (not provided) is written in C and FORTRAN77, compiled and linked into so-called MEX-modules, that are executables for use within the MATLAB-environment. The following files must exist:

```

checksbf.dll      combsbd.dll      getfrsbf.dll     loadsbd.dll
loadsbf.dll      ltv.dll          mrltv.dll        repinsbf.dll
spacar.dll       spacntrl.dll     spasm.dll

```

- Files with extension `.m` are the MATLAB-files necessary to use the SPACAR and SPAVISUAL program. The following files must exist:

```

getss.m
spadraw.m
spavisual.m

```

Other `.m`-files provide help text for the MEX-modules. These files are:

```

checksbf.m      combsbd.m      getfrsbf.m      loadsbd.m
loadsbf.m      ltv.m          mrltv.m          repinsbf.m
spacar.m       spacntrl.m     spasm.m

```

- Files with extension `.mdl` are SIMULINK models. There is only one file which is actually a library from which the SPACAR modules for use in SIMULINK can be copied:

```

spacar_lib.mdl

```

- A folder with the name `private` with a lot of `.p`-files that are used by SPAVISUAL.

The (optional) data files can be extracted in a separate working directory.

The files in the SPACAR subdirectory should be in the MATLAB path when MATLAB is running. There are two ways to accomplish this:

1. Make sure that the SPACAR subdirectory is the local directory. You can verify this by typing `pwd`. If necessary, change your local directory by typing

```

cd \Matlab\Toolbox\Spacar

```

or whatever directory you chose to store your files.

2. Another possibility is to change the settings of the MATLAB environment by adding the SPACAR subdirectory to the MATLAB path. This modification is either temporary or permanent. The path can be modified from the pulldown menu with `File|Set Path...`, or by using the MATLAB commands `path` or `addpath`.

Now you are ready to run SPACAR in MATLAB and SIMULINK.



# B SPACAR keywords

## B.1 Introduction

In this appendix the user is informed about the creation of correct input data for the software package SPACAR. The input must have a specific form. Behind a number of permitted keywords the user supplies a list of arguments. The arguments behind a keyword are well defined. Each module of SPACAR, except `mode=4` of LINEAR, has its own list of available keywords. They form blocks that are separated by the following pair of keywords:

```
END  
HALT
```

The final closure of the input is effected by:

```
END  
END
```

The first block contains the kinematic data. The input of the mechanism model (by means of keywords) is treated in the “Kinematics” section B.2. A second block of input is reserved for the dynamics module. The keywords for this block are presented in the “Dynamics” section B.3. The keywords for the linearization of `mode=9` are given in the “Linearization” section B.4. At the end of the file custom settings for SPAVISUAL can be added, the keywords can be found in section B.5.

Some general remarks:

- Keywords and arguments can be separated by one or more spaces, tabs or line breaks.
- Lines must not contain more than 160 characters.
- Any text in a line following a #, % or ; is treated as a comment.
- All input is case insensitive.
- Data read from the input file are echoed in the `log` file, *after* the comments have been removed and all text is transformed into upper case (capitals).
- Angles are always specified in radians.
- For commands like XF and STARTDE not all arguments have to be specified. Default values are zero unless otherwise specified.

## B.2 Kinematics

A kinematic mechanism model can be built up with finite elements by letting them have nodal points in common. The nodal coordinates of the finite elements are described by position and orientation coordinates. Therefore, two types of nodes are distinguished: *position* or *translational nodes*, denoted

keyword	type	end node $p$		end node $q$		deformation parameters
		$\vec{x}$	$\hat{\phi}$	$\vec{x}$	$\hat{\phi}$	
PLBEAM	planar beam	$\mathbf{x}^p$	$\hat{\phi}^p$	$\mathbf{x}^q$	$\hat{\phi}^q$	$e_1, e_2, e_3$
PLTRUSS	planar truss	$\mathbf{x}^p$	–	$\mathbf{x}^q$	–	$e_1$
PLTOR	planar hinge	–	$\hat{\phi}^p$	–	$\hat{\phi}^q$	$e_1$
PLPINBOD	planar pinbody	$\mathbf{x}^p$	$\hat{\phi}^p$	$\mathbf{x}^q$	–	$e_1, e_2$
PLRBEAM	planar rigid beam	$\mathbf{x}^p$	$\hat{\phi}^p$	$\mathbf{x}^q$	–	$e_1, e_2$
BEAM	spatial beam	$\mathbf{x}^p$	$\lambda^p$	$\mathbf{x}^q$	$\lambda^q$	$e_1, e_2, e_3, e_4, e_5, e_6$
TRUSS	spatial truss	$\mathbf{x}^p$	–	$\mathbf{x}^q$	–	$e_1$
HINGE	spatial hinge	–	$\lambda^p$	–	$\lambda^q$	$e_1, e_2, e_3$
PINBODY	spatial pinbody	$\mathbf{x}^p$	$\lambda^p$	$\mathbf{x}^q$	–	$e_1, e_2, e_3$
RBEAM	spatial rigid beam	$\mathbf{x}^p$	$\lambda^p$	$\mathbf{x}^q$	–	$e_1, e_2, e_3$

Table B.1: Nodes, nodal coordinates, and deformation parameters for the planar and spatial truss, beam, bearing, hinge and pinbody elements.

by  $\vec{p}$  for node  $p$ , and *orientation* or *rotational nodes* denoted by  $\hat{p}$ . The nodes, nodal coordinates, and deformation parameters for the truss, beam, planar bearing, hinge and pinbody (rigid beam) element are summarized in Table B.1.

Usually, the convention is made that node  $p$  of an element is assigned to the lower number of the element nodes, and that node  $q$  is assigned to the higher node number. The interconnections between the elements are accomplished by indicating common nodes between the elements. For instance, with a pin-joint connection only the translational nodes are shared. In case of a hinge-joint connection only the rotational nodes are shared whereas translational coordinates can either be shared or unshared. When elements are rigidly connected to each other, both the translational and rotational nodes are shared, see Fig. B.1. It can be observed from Table B.1 that a truss element and a hinge element do not have common nodal types and therefore cannot be connected to each other.

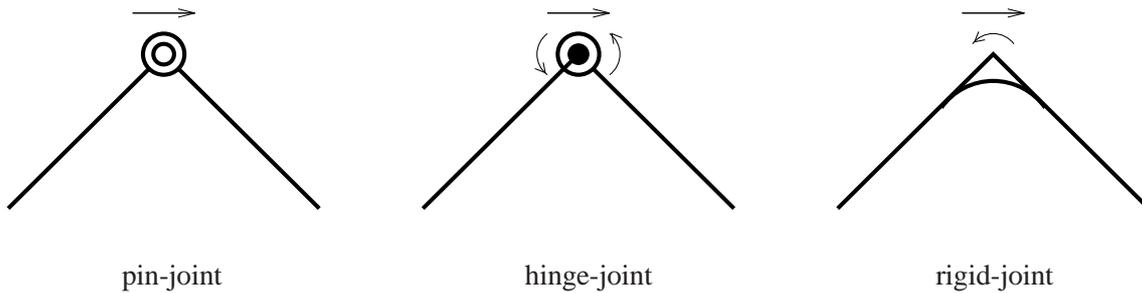


Figure B.1: Joint connections between finite elements.

In the first block of the kinematics module either two-dimensional (planar) or three-dimensional (spatial) elements can be specified. In the second block the initial configuration of the mechanism is specified. In the third block the coordinates and generalized deformations are divided into four groups, depending on the boundary conditions:

1. fixed prescribed coordinates (supports)
2. dependent, calculable deformations
3. prescribed, time-dependent coordinates
4. dynamic degrees of freedom

For the keywords in the third block it is important to remark that there are no keywords to fix a deformation or to release a coordinate. These are the default settings. So a deformation is fixed unless a RLSE, INPUTE or DYNE keyword specifies otherwise. Similarly, a coordinate is calculable unless a FIX, INPUTX or DYNX keyword specifies otherwise.

<b>KEYWORDS KINEMATICS</b>
----------------------------

1	<table border="1"> <tr><td>PLBEAM</td><td>Planar beam element.</td></tr> <tr><td>PLTRUSS</td><td>Planar truss element.</td></tr> <tr><td>PLTOR</td><td>Planar hinge element.</td></tr> <tr><td>PLPINBOD</td><td>Planar pinbody element.</td></tr> <tr><td>PLRBEAM</td><td>Planar rigid beam element.</td></tr> <tr><td>BEAM</td><td>Beam element.</td></tr> <tr><td>TRUSS</td><td>Truss element.</td></tr> <tr><td>HINGE</td><td>Hinge element.</td></tr> <tr><td>PINBODY</td><td>Spatial pinbody element.</td></tr> <tr><td>RBEAM</td><td>Spatial rigid beam element.</td></tr> </table>	PLBEAM	Planar beam element.	PLTRUSS	Planar truss element.	PLTOR	Planar hinge element.	PLPINBOD	Planar pinbody element.	PLRBEAM	Planar rigid beam element.	BEAM	Beam element.	TRUSS	Truss element.	HINGE	Hinge element.	PINBODY	Spatial pinbody element.	RBEAM	Spatial rigid beam element.
PLBEAM	Planar beam element.																				
PLTRUSS	Planar truss element.																				
PLTOR	Planar hinge element.																				
PLPINBOD	Planar pinbody element.																				
PLRBEAM	Planar rigid beam element.																				
BEAM	Beam element.																				
TRUSS	Truss element.																				
HINGE	Hinge element.																				
PINBODY	Spatial pinbody element.																				
RBEAM	Spatial rigid beam element.																				
2	<table border="1"> <tr><td>X</td><td>Specification of the initial Cartesian nodal positions.</td></tr> </table>	X	Specification of the initial Cartesian nodal positions.																		
X	Specification of the initial Cartesian nodal positions.																				
3	<table border="1"> <tr><td>FIX</td><td>Support coordinates <math>\boldsymbol{x}^0</math>.</td></tr> <tr><td>RLSE</td><td>Calculable deformations <math>\boldsymbol{e}^c</math>.</td></tr> <tr><td>INPUTX</td><td>Prescribed DOF <math>\boldsymbol{x}^m</math>.</td></tr> <tr><td>INPUTE</td><td>Prescribed DOF <math>\boldsymbol{e}^m</math>.</td></tr> <tr><td>DYNX</td><td>Dynamic DOF <math>\boldsymbol{x}^m</math>.</td></tr> <tr><td>DYNE</td><td>Dynamic DOF <math>\boldsymbol{e}^m</math>.</td></tr> </table>	FIX	Support coordinates $\boldsymbol{x}^0$ .	RLSE	Calculable deformations $\boldsymbol{e}^c$ .	INPUTX	Prescribed DOF $\boldsymbol{x}^m$ .	INPUTE	Prescribed DOF $\boldsymbol{e}^m$ .	DYNX	Dynamic DOF $\boldsymbol{x}^m$ .	DYNE	Dynamic DOF $\boldsymbol{e}^m$ .								
FIX	Support coordinates $\boldsymbol{x}^0$ .																				
RLSE	Calculable deformations $\boldsymbol{e}^c$ .																				
INPUTX	Prescribed DOF $\boldsymbol{x}^m$ .																				
INPUTE	Prescribed DOF $\boldsymbol{e}^m$ .																				
DYNX	Dynamic DOF $\boldsymbol{x}^m$ .																				
DYNE	Dynamic DOF $\boldsymbol{e}^m$ .																				

The parameters for these keywords are listed below.  $\{*i\}$  refers to note  $i$  listed at the end of the keywords.

PLBEAM	1 2 3 4 5	element number first position node first orientation node second position node second orientation node
PLTRUSS	1 2 3	element number first position node second position node
PLTOR	1 2 3	element number first orientation node second orientation node
PLPINBOD	1 2 3 4	element number first position node first orientation node second position node
PLRBEAM	1 2 3 4	element number first position node first orientation node second position node
BEAM	1 2 3 4 5 [ 6–8	element number first position node first orientation node second position node second orientation node initial direction of the principal $y'$ -axis of the beam cross-section ] $\{*1\}$
TRUSS	1 2 3	element number first position node second position node
HINGE	1 2 3 4–6	element number first orientation node second orientation node initial direction of the $x'$ -axis of rotation $\{*2\}$
PINBODY	1 2 3 4 [ 5–7	element number first position node first orientation node second position node initial direction of the principal $y'$ -axis of the beam cross-section ] $\{*3\}$
RBEAM	1 2 3 4 [ 5–7	element number first position node first orientation node second position node initial direction of the principal $y'$ -axis of the beam cross-section ] $\{*1\}$

X	1	position node number
	2	$x_1$ -coordinate
	3	$x_2$ -coordinate
	[ 4	$x_3$ -coordinate ] { *4 }

FIX	1	node number
	[ 2–	coordinate number (1, 2, 3 or 4) ] { *5 }
RLSE	1	element number
	[ 2–	deformation mode coordinate number (1, 2, 3, 4, 5 or 6) ] { *6 }
INPUTX	1	node number
	[ 2–	coordinate number (1, 2, 3 or 4) ] { *5 }
INPUTE	1	element number
	[ 2–	deformation mode coordinate number (1, 2, 3, 4, 5 or 6) ] { *6 }
DYNX	1	node number
	[ 2–	coordinate number (1, 2, 3 or 4) ] { *5 }
DYNE	1	element number
	[ 2–	deformation mode coordinate number (1, 2, 3, 4, 5 or 6) ] { *6 }

## NOTES:

- \*1 The direction vector lies in the local  $x'y'$ -plane of the beam element. If no direction is specified, the local direction vector is chosen as the standard basis vector that makes the largest angle with axis of the beam; in case of a draw, the vector with the highest index is chosen.
- \*2 The local  $y'$  and  $z'$  unit vectors are chosen as follows. First, the standard basis vector with the largest angle with the hinge axis is chosen; in case of a draw, the vector with the highest index is chosen. Then the local  $y'$  is chosen in the direction of the cross product of the local  $x'$ -direction with this basis vector. The local  $z'$ -direction is chosen so as to complete an orthogonal right-handed coordinate system.
- \*3 If no direction is specified, directions initially aligned with the global coordinate axes are chosen; otherwise the line connecting the translational node is chosen as the local  $x'$ -direction, the specified vector is in the local  $x'y'$ -plane. The directions used are made orthonormal.
- \*4 The specification of the initial positions with the keyword X is only required for non-zero position-coordinates. The initial orientations cannot be chosen freely.
- \*5 If the keywords INPUTX, DYNX and FIX are used without an explicit specification of the coordinate, all (independent) coordinates will be marked as degrees of freedom or supports. This means that  $x_1$ ,  $x_2$  (and  $x_3$ ) are marked for position nodes and  $\beta$  or  $\lambda_1$ ,  $\lambda_2$  and  $\lambda_3$  for orientation nodes. If more than one coordinate is specified, each of the specified coordinates is chosen as a degree of freedom or a support.
- \*6 If the keywords INPUTE, DYNE and RLSE are used without an explicit specification of the deformation mode coordinate, all deformation mode coordinates will be marked as degrees of freedom or released. If more than one deformation mode coordinate is specified, each of the specified coordinates is chosen as a degree of freedom or as released.

### B.3 Dynamics

With the keywords of the dynamics module the following blocks of information can be supplied. Blocks 1 and 2 are optional. If deformable elements have been defined in the kinematics, block 3 has to be filled, lest the stiffness and damping are zero. If the motion is not prescribed by trajectories, block 4 has to be used to define the input motion.

<b>KEYWORDS DYNAMICS</b>
--------------------------

1	<table border="1"> <tr> <td style="padding: 2px;">XM</td> <td>Inertia specification of lumped masses.</td> </tr> <tr> <td style="padding: 2px;">EM</td> <td>Inertia specification of distributed element masses.</td> </tr> </table>	XM	Inertia specification of lumped masses.	EM	Inertia specification of distributed element masses.						
XM	Inertia specification of lumped masses.										
EM	Inertia specification of distributed element masses.										
2	<table border="1"> <tr> <td style="padding: 2px;">XF</td> <td>External force specification of the mechanism in nodes.</td> </tr> </table>	XF	External force specification of the mechanism in nodes.								
XF	External force specification of the mechanism in nodes.										
3	<table border="1"> <tr> <td style="padding: 2px;">ESTIFF</td> <td>Specification of elastic constants.</td> </tr> <tr> <td style="padding: 2px;">ESIG</td> <td>Specification of preloaded state.</td> </tr> <tr> <td style="padding: 2px;">EDAMP</td> <td>Specification of viscous damping coefficients.</td> </tr> </table>	ESTIFF	Specification of elastic constants.	ESIG	Specification of preloaded state.	EDAMP	Specification of viscous damping coefficients.				
ESTIFF	Specification of elastic constants.										
ESIG	Specification of preloaded state.										
EDAMP	Specification of viscous damping coefficients.										
4	<table border="1"> <tr> <td style="padding: 2px;">TIMESTEP</td> <td>Duration and number of time steps.</td> </tr> <tr> <td style="padding: 2px;">INPUTX</td> <td>Specification of simple time functions for the prescribed degrees of freedom.</td> </tr> <tr> <td style="padding: 2px;">INPUTE</td> <td>Specification of initial values for the dynamic degrees of freedom.</td> </tr> <tr> <td style="padding: 2px;">STARTDX</td> <td>Specification of initial values for the dynamic degrees of freedom.</td> </tr> <tr> <td style="padding: 2px;">STARTDE</td> <td>Specification of initial values for the dynamic degrees of freedom.</td> </tr> </table>	TIMESTEP	Duration and number of time steps.	INPUTX	Specification of simple time functions for the prescribed degrees of freedom.	INPUTE	Specification of initial values for the dynamic degrees of freedom.	STARTDX	Specification of initial values for the dynamic degrees of freedom.	STARTDE	Specification of initial values for the dynamic degrees of freedom.
TIMESTEP	Duration and number of time steps.										
INPUTX	Specification of simple time functions for the prescribed degrees of freedom.										
INPUTE	Specification of initial values for the dynamic degrees of freedom.										
STARTDX	Specification of initial values for the dynamic degrees of freedom.										
STARTDE	Specification of initial values for the dynamic degrees of freedom.										

The parameters required with these keywords are listed below. {\*i} refers to note i listed at the end of the keywords.

XM	1	node number
	2	concentrated mass for position nodes, rotational inertia $I$ for plane orientation nodes, for spatial orientation nodes, the inertia components $J_{xx}$ {*1}
	3	$J_{xy}$ {*1}
	4	$J_{xz}$ {*1}
	5	$J_{yy}$ {*1}
	6	$J_{yz}$ {*1}
	7	$J_{zz}$ {*1}
EM	1	element number
	2	mass per unit of length
	[ 3	rotational inertia $J_{x'x'}$ per unit of length for spatial beam rotational inertia $J$ per unit of length for planar beam ]
	[ 4	rotational inertia $J_{y'y'}$ per unit of length for spatial beam ]
	[ 5	rotational inertia $J_{z'z'}$ per unit of length for spatial beam ]
	[ 6	rotational inertia $J_{y'z'}$ per unit of length for spatial beam ]
XF	1	node number
	2	forces dual with the 1 <sup>st</sup> node coordinate
	[ 3-5	forces dual with the 2 <sup>nd</sup> , 3 <sup>rd</sup> and 4 <sup>th</sup> node coordinate ]
ESTIFF	1	element number
	2	$EA$ for beam and truss elements $S_1 = S_t$ for hinge elements $S_1$ , first stiffness coefficient for pinbody and cognates
	[ 3	$GI_t$ for spatial beam $EI$ for planar beam $S_2$ , second stiffness coefficient for pinbody and cognates ] {*4}
	[ 4	$EI_{y'}$ for spatial beam $EI/(GAk)$ for planar beam $S_3$ , third stiffness coefficient for pinbody and cognates ] {*4}
	[ 5	$EI_{z'}$ for spatial beam ] {*4}
	[ 6	$EI_{y'}/(GAk_{z'})$ for spatial beam ] {*4}
	[ 7	$EI_{z'}/(GAk_{y'})$ for spatial beam ] {*4}
ESIG	1	element number
	2	preloaded force in beam, pinbody and truss elements and torque in hinge elements
EDAMP	1	element number
	2	$E_dA$ , longitudinal damping for beam and truss elements $S_{d1}$ , torsional damping for hinge elements $S_{d1}$ , first damping coefficient for pinbody and cognates
	[ 3	$G_dI_t$ , torsional damping for beam elements $E_dI$ , bending damping for planar beams $S_{d2}$ , second damping coefficient for pinbody and cognates ] {*4}
	[ 4	$E_dI_{y'}$ , bending damping in $y'$ -direction for spatial beams $S_{d3}$ , third damping coefficient for pinbody and cognates ] {*4}
	[ 5	$E_dI_{z'}$ , bending damping in $z'$ -direction for spatial beam ] {*4}

TIMESTEP	1	length of time period
	2	number of time steps
INPUTX	1	node number (position or orientation node) {*5}
	2	coordinate number (1, 2, 3 or 4)
	3	start value
	4	start rate
	5	acceleration (constant)
INPUTE	1	element number {*6}
	2	deformation mode coordinate number (1, 2, 3, 4, 5 or 6) {*7}
	3	start value {*8}
	4	start rate
	5	acceleration (constant)
STARTDX	1	node number
	2	coordinate number (1, 2, 3 or 4)
	3	start value
	4	start rate
STARTDE	1	element number
	2	deformation mode coordinate number (1, 2, 3, 4, 5 or 6)
	3	start value {*8}
	4	start rate

## NOTES:

- \*4 Unspecified values for the stiffness and damping are assumed to be zero by default. The meaning of the variables is:  $E$ , elasticity modulus (Young's modulus);  $G = E/(2 + 2\nu)$ , shear modulus;  $\nu$ , Poisson's ratio;  $E_d$ , damping modulus in Kelvin–Voigt model;  $G_d$ , shear damping modulus in Kelvin–Voigt model;  $A$ , cross-sectional area;  $I$  ( $I_{y'}$ ,  $I_{z'}$ ), second area moment (about  $y'$ -axis and  $z'$ -axis);  $I_t$ , Saint-Venant's torsion constant;  $k$  ( $k_{y'}$  and  $k_{z'}$ ), shear correction factor (in  $y'$ -direction and  $z'$ -direction). The shear correction factors are about 0.85; a table of values for various cross-sections can be found in [?].

The generalized stresses are calculated according to the Kelvin–Voigt model as follows. All first stresses are calculated as  $\sigma_1 = S_1\varepsilon_1 + S_{d1}\dot{\varepsilon}_1 + \sigma_0$ , where  $S_1 = EA/l_0$  and  $S_{d1} = E_d A/l_0$  for the truss and beam elements, where  $l_0$  is the undeformed length of the element, and the first stiffness and damping coefficients as defined in the input for the other types of elements.  $\sigma_0$  is the preload defined by the keyword ESIG. For hinge and pinbody elements, the other stresses are calculated in an analogous way, but without the preload. For a planar beam element, the bending stresses are calculated as

$$\begin{bmatrix} \sigma_2 \\ \sigma_3 \end{bmatrix} = \frac{S_2}{1 + \Phi} \begin{bmatrix} 4 + \Phi & -2 + \Phi \\ -2 + \Phi & 4 + \Phi \end{bmatrix} \begin{bmatrix} \varepsilon_2 \\ \varepsilon_3 \end{bmatrix} + \frac{S_{d2}}{1 + \Phi} \begin{bmatrix} 4 + \Phi & -2 + \Phi \\ -2 + \Phi & 4 + \Phi \end{bmatrix} \begin{bmatrix} \dot{\varepsilon}_2 \\ \dot{\varepsilon}_3 \end{bmatrix},$$

where  $S_2 = EI/l_0^3$ ,  $\Phi = 12EI/(GAKl_0^2)$  and  $S_{d2} = E_d I/l_0^3$ . For a spatial beam element, the torsional stress is calculated as  $\sigma_2 = S_2\varepsilon_2 + S_{d2}\dot{\varepsilon}_2 + \sigma_0$ , where  $S_2 = GI_t/l_0^3$  and  $S_{d2} = G_d I_t/l_0^3$ ; note the wacky occurrence of  $\sigma_0$  here. For bending along the local  $y'$ - and  $z'$ -axes, the stresses are, analogous to the planar case,

$$\begin{bmatrix} \sigma_3 \\ \sigma_4 \end{bmatrix} = \frac{S_3}{1 + \Phi_z} \begin{bmatrix} 4 + \Phi_z & -2 + \Phi_z \\ -2 + \Phi_z & 4 + \Phi_z \end{bmatrix} \begin{bmatrix} \varepsilon_3 \\ \varepsilon_4 \end{bmatrix} + \frac{S_{d3}}{1 + \Phi_z} \begin{bmatrix} 4 + \Phi_z & -2 + \Phi_z \\ -2 + \Phi_z & 4 + \Phi_z \end{bmatrix} \begin{bmatrix} \dot{\varepsilon}_3 \\ \dot{\varepsilon}_4 \end{bmatrix}$$

and

$$\begin{bmatrix} \sigma_5 \\ \sigma_6 \end{bmatrix} = \frac{S_4}{1 + \Phi_y} \begin{bmatrix} 4 + \Phi_y & -2 + \Phi_y \\ -2 + \Phi_y & 4 + \Phi_y \end{bmatrix} \begin{bmatrix} \varepsilon_5 \\ \varepsilon_6 \end{bmatrix} + \frac{S_{d4}}{1 + \Phi_y} \begin{bmatrix} 4 + \Phi_y & -2 + \Phi_y \\ -2 + \Phi_y & 4 + \Phi_y \end{bmatrix} \begin{bmatrix} \dot{\varepsilon}_5 \\ \dot{\varepsilon}_6 \end{bmatrix},$$

where  $S_3 = EI_{y'}/l_0^3$ ,  $\Phi_z = 12EI_{y'}/(GAk_{z'}l_0^2)$ ,  $S_{d3} = E_dI_{y'}/l_0^3$ ,  $S_4 = EI_{z'}/l_0^3$ ,  $\Phi_y = 12EI_{z'}/(GAk_{y'}l_0^2)$ , and  $S_{d4} = E_dI_{z'}/l_0^3$ .

- \*5 In a mode=7, 8 or 9 run a (deformed) mechanism configuration is computed which corresponds with the specified nodal position.
- \*6 Stiffness and damping properties of the corresponding element are not used for the dynamic computations.  
In a mode=7, 8 or 9 run a (deformed) mechanism configuration is computed which corresponds with the specified element deformation.
- \*7 Rotational deformations are defined in radians.
- \*8 Note that the keyword X defines an initial configuration in which the deformations are zero. A start value defined with INPUTE or STARTDE defines a deformation with respect to the initial configuration.

## B.4 Linearization

Linearization in mode=7 and 9 is around a pre-computed static equilibrium configuration, or a state of steady motion. In addition in mode=9 the state space matrix  $A$ , the input matrices  $B_0$  and  $B$ , the output matrix  $C$  and the feed through matrix  $D$  are calculated. Obviously, the matrices  $B_0$ ,  $B$ ,  $C$  and  $D$  depend on the chosen input and output vectors  $\delta u$  and  $\delta y$  respectively. These vectors are defined in the blocks below. These blocks are optional, but as omitting one or both blocks means that no input and/or output vectors are defined and hence no state space matrices can be generated and written to the Itv-file

### KEYWORDS INPUT VECTOR $\delta u$ (mode=9)

1

INPUTS	Specification of input stresses.
INPUTF	Specification of input forces.
INE	Specification of input deformation parameters.
INEP	<i>Ibid.</i> first time derivative.
INEDP	<i>Ibid.</i> second time derivative.
INX	Specification of input nodal coordinates.
INXP	<i>Ibid.</i> first time derivative.
INXDP	<i>Ibid.</i> second time derivative.

### KEYWORDS OUTPUT VECTOR $\delta y$ (mode=9)

2

OUTS	Specification of output stresses.
OUTF	Specification of output forces.
OUTE	Specification of output deformation parameters.
OUTEP	<i>Ibid.</i> first time derivative.
OUTEDP	<i>Ibid.</i> second time derivative.
OUTX	Specification of output nodal coordinates.
OUTXP	<i>Ibid.</i> first time derivative.
OUTXDP	<i>Ibid.</i> second time derivative (see note).

The parameters for these keywords are listed below.  $\{*i\}$  refers to note  $i$  listed at the end of the keywords.

INPUTS $\{*2\}$	1	input number $\{*1\}$
INE	2	element number
INEP	3	deformation parameter number (1, 2, 3, 4, 5 or 6)
INEDP		
INPUTF $\{*4\}$	1	input number $\{*1\}$
INX	2	node number
INXP	3	coordinate number (1, 2, 3, or 4)
INXDP		

OUTS $\{*6\}$	1	output number $\{*1\}$
OUTE	2	element number
OUTEP	3	deformation parameter number (1, 2, 3, 4, 5 or 6)
OUTEDP		
OUTF $\{*8\}$	1	output number $\{*1\}$
OUTX	2	node number
OUTXP	3	coordinate number (1, 2, 3, or 4)
OUTXDP		

## NOTES:

- \*1 The input numbers and output numbers are the positions of the specified inputs or outputs in the input and output vectors, respectively.
- \*2 Associated with dynamic DOF's  $e^{(m,d)}$ .
- \*3 Associated with prescribed deformations  $e^{(m,r)}$ .
- \*4 Associated with calculable coordinates  $x^{(c)}$  and/or dynamic DOF's  $x^{(m,d)}$ .
- \*5 Associated with prescribed nodal coordinates  $x^{(m,r)}$ .
- \*6 Associated with prescribed deformations  $e^{(0)}$  and/or  $e^{(m,r)}$ .
- \*7 Associated with calculable deformations  $e^{(c)}$  and/or dynamic DOF's  $e^{(m,d)}$ .
- \*8 Associated with prescribed nodal coordinates  $x^{(0)}$  and/or  $x^{(m,r)}$ .

## B.5 Visualization and animation

To adjust the default settings of SPAVISUAL the user can type VISUALIZATION after the last two END commands in the .dat file. All commands after the command VISUALIZATION are read by SPAVISUAL as an adjustment on the default settings.

BEAMVIS	Adjusts the height and the width of a beam element.
HINGEVIS	Adjusts the radius and the length of the hinge element.
TRUSSVIS	Sets the visualization of the truss element on or off.
TRANSPARENCY	Adjusts the transparency of the elements.
VIBRATIONMODE	Selects the vibration modes.
BUCKLINGMODE	Selects the buckling modes.
ENLARGFACTOR	Sets the amplitude of the vibration or buckling modes.
RECORDMOVIE	Sets recordmovie on or off, the movie is saved as an .avi file in the workspace.
MOVIENAME	Sets the name of the recorded movie.
UNDEFORMED	Sets the visualization of the un-deformed mechanism on or off.
VIBREND	Sets the period of the sine function for the vibration mode.
STEPLINE	Sets the size of the line elements that are used to draw the elements.
STEPVIBRATION	Sets the number of steps in the vibration visualization.
LIGHT	Sets the light on or off.
JOINTS	Sets the joints on or off.
TRAJECTVIS	Sets the trajectory on or off.
TRAJECTNODE	Selects the node for the trajectory.

The parameters for these keywords are listed below.

KEYWORD	DESCRIPTION	DEFAULT SETTINGS
BEAMVIS {*1}	1 size of all beam elements in the local y' direction 2 size of all beam elements in the local z' direction	0.006 0.006
BEAMVIS {*1}	1 element number 2 size of the element for the local y' direction 3 size of the element for the local z' direction	0.006 0.006
HINGEVIS {*2}	1 element number 2 length of the hinge 3 radius of the hinge	0.003 0.009
TRUSSVIS	1 sets the visualization of the truss elements on (1) or off (0)	on
TRANSPARENCY	1 adjusts the transparency between (0) and (1)	1
VIBRATIONMODE	1 switch between the 10 lowest vibration modes {*3}	1
BUCKLINGMODE	1 switch between the 10 buckling modes {*3}	1
ENLARGFACTOR	1 amplitude of the vibration or buckling modes	1
RECORDMOVIE	1 sets record movie on (1) or off (0)	off
MOVIEFILENAME	1 filename for the recorded movie	filename
UNDEFORMED	1 sets the visualization of the initial (un-deformed) mechanism configuration on (1) or off (0)	on
VIBREND	1 sets the period of the sin function for the vibration mode	$2\pi$ {*4}
STEPLINE	1 sets the size of the line elements that are used to draw the elements	0.2
STEPVIBRATION	1 sets the number of intermediate steps in the vibration visualization	$\frac{\pi}{10}$ {*4}
LIGHT	1 sets the light source on (1) or off (0)	off
JOINTS	1 sets the joints on (1) or off (0)	on
TRAJECTVIS	1 sets the trajectory on (1) or off (0)	off
TRAJETCNODE	1 Select the node for the trajectory	1

### Notes

- \*1 The BEAMVIS command has two variations. The one with only two parameters adjusts all beam elements. The variant with three parameters can be used to adjust only a single beam element.
- \*2 If multiple hinges are used in series, they should be numbered sequentially and the hinge with the lowest element number should be connected to the beam.
- \*3 Only the lowest vibration and buckling modes are available with a maximum of 10 modes.
- \*4 Only numerical values are allowed, no symbols or functions.



# Bibliography

- [1] R.G.K.M. Aarts and J.B. Jonker, *Dynamic simulation of planar flexible link manipulators using adaptive modal integration*, *Multibody Systems Dynamics*, **7**, pp. 3150, 2002.
- [2] R.G.K.M. Aarts and J. van Dijk, *Inleiding Systeem- en Regeltechniek* (in Dutch), Lecture notes University of Twente, 2007.
- [3] R.G.K.M. Aarts, J.P. Meijaard, J.B. Jonker *SPACAR manual*, University of Twente, 2008.
- [4] J. van Dijk, *Systeem- en Regeltechniek 2* (in Dutch), Lecture notes University of Twente, 2008.
- [5] Gene F. Franklin, J. David Powell and Abbas Emami-Naeini, *Feedback control of dynamic systems*, 5<sup>th</sup> edition, Pearson Education, ISBN 0-13-149930-0, 2006.
- [6] P.J.M. van der Hoogt, *Dynamica 2*, Lecture notes University of Twente, 2006.
- [7] J.B. Jonker and J.P. Meijaard. *SPACAR-computer program for dynamic analysis of flexible spational mechanisms and manipulators*. In *Multibody Systems Handbook*, W. Schiehlen (ed.), 123–143. Springer-Verlag, Berlin, 1990.
- [8] J.B. Jonker, *Dynamics of machines: A finite element approach*, Lecture notes University of Twente, 2007.
- [9] J.B. Jonker, R.G.K.M. Aarts and J. van Dijk *An extended input-output representation for control synthesis in multibody system dynamics*, to be published on the ECCOMAS Thematic Conference “Multibody Dynamics 2007”, Milano, Italy, 25–28 June 2007.
- [10] M.P. Koster, *Constructieprincipes voor het nauwkeurig bewegen en positioneren*, 3<sup>e</sup> druk, Twente University Press, ISBN: 90-365-1456-8 or 90-365-1455-X, 2000.
- [11] The Math Works Inc., *Getting Started with MATLAB*, version 7, Revised for MATLAB 7.3 (Release 2006b), September 2006.  
WWW\*: <http://www.mathworks.com/.../matlab/getstart.pdf>
- [12] The Math Works Inc., *SIMULINK — Getting Started*, version 6, Revised for SIMULINK 6.5 (Release 2006b), September 2006.  
WWW\*: [http://www.mathworks.com/.../simulink/sl\\_gs.pdf](http://www.mathworks.com/.../simulink/sl_gs.pdf)
- [13] J.L. Meriam and L.G. Kraige *Engineering Mechanics, Dynamics*, 4<sup>th</sup> edition, SI version, John Wiley & Sons, ISBN 0-471-24167-9, 1998
- [14] Robert L. Mott, *Machine elements in mechanical design*, Fourth edition in SI units, Pearson Education, ISBN 0-13-197644-3, 2005.

\* The full path to the PDF documents of The Math Works starts with  
[http://www.mathworks.com/access/helpdesk/help/pdf\\_doc/](http://www.mathworks.com/access/helpdesk/help/pdf_doc/)