CONTENTS

 ISSN 1061-5733

Flexible Mechanical Engineering Guidelines A Maple-based Tool for Cam Design

Burkhard Alpers* *and Stephan Steeb

Abstract: In this article we describe how Maple as a symbolic computer algebra program can be used to make engineering guidelines for cam design accessible in a flexible way. Symbolic computation contained in the guidelines is implemented in Maple such that new symbolic input functions for cam design can be processed and output functions (so-called displacement programs) are provided graphically, tabulated and symbolically for further processing.

Keywords: Mechanical Engineering, Cam Design, VDI-Guidelines, Displacement Programs

Introduction

Cam mechanisms are widely used for machine construction in processing, manufacturing and packaging industries. They enable exact motion of a rigid body along a prescribed planar or spatial curve. Often, these mechanism play an important role with respect to the overall performance of the machine: Improvement of the mechanism allows for more cycles per time unit and hence for a higher throughput.

In order to support engineers in constructing good mechanisms, the Association of German Engineers (Verein Deutscher Ingenieure, VDI) provides guidelines in many areas of construction. For cam design, there are guidelines describing the general procedure of constructing a motion function from a given plan [1, 2]. Moreover, quality measures for motion functions are derived symbolically and several so-called "normal motion laws" as well as their advantages and drawbacks are discussed. The industrial engineer who is looking for a solution to a certain motion problem is expected to check which quality measure is most important to the application under consideration and choose the most adequate one from the set of "motion laws" discussed in the guidelines. The tables provided in the guidelines and some adaptation formulae then serve as input data for further calculation. This way, the design process should be accelerated considerably. Yet, the guidelines suffer from the following restrictions:

- The guidelines are only available in paper form and are hence not part of the usual electronic working environment of an engineer working in industry. Although the motion laws dealt with in the guidelines have been implemented in large and costly CAD/CAM- packages, they are not available as a cheap, self-contained program.

- For space reasons, the guidelines contain quality measures and tabulated values for only very few "motion laws", i.e. functions. For other functions, the reader has to compute the values himself according to the formulae presented in the guidelines. This makes experimentation very tedious, time consuming, and error-prone, and therefore, engineers usually pick one of the functions computed in the guidelines thus giving away the chance of improvement.

- As will be explained in more detail in the next section, the values of the resulting motion function have to be computed piecewise. It is extremely tedious to write up the piecewise defined function symbolically in order to feed it as input function into other routines. But further processing of this function is very interesting: Recent research [6] has shown that the fourier series of this function provides valuable insight in the dynamic behaviour of the mechanism and can be used to reduce oscillatory effects (vibration).

Clearly, the short-comings addressed above call for a program which can deal with arbitrary symbolic input functions and is able to provide symbolic output functions. Since Maple is not only a computer algebra program but also includes a programming language for symbolic computation, it serves very well as a programming environment for the above-mentioned purpose. This was also demonstrated in [3] and [7]. [3] already outlined how Maple can be used for all stages of the cam design. In our work, the emphasis is put on making the VDI-guideline 2143 (suggested functions, quality measures, etc.) available in a user-friendly way for an engineer who is just an occasional Maple user. As does the guideline, we mainly cover kinematic aspects of the cam design process although dynamic aspects are certainly equally important.

The paper is organized as follows: In the next section, the reader is provided with basic terminology and theory of cam design as far as needed for understanding the rest of the paper. Then we present in more detail the requirements on the program which should support the engineer in choosing and experimenting with motion functions. This is followed by the description of how the program was implemented using Maple worksheets and procedures. Thereafter, we present

*Department of Mechanical Engineering, Aalen University of Technology and Business, Beethovenstr. 1, D-73428 Aalen, balper@fh-aalen.de

two user sessions to show how the engineer is assumed to work with the program. Finally, we give an outlook on further developments and draw some conclusions on the value of Maple as a tool for meeting the requirements.

Basic Terminology and Theory in Planar Cam Design

In this section, we present the basic terminology and theory for designing planar cam mechanisms. The outline is based upon [1, 4, 5] but it is restricted to those issues which are necessary for understanding the task and the support requirements. A planar cam mechanism basically consists of three components:

- a cam disk, also called the driving element, which is usually connected to a motor (directly or via a uniformly transforming mechanism);
- a follower, also called the driven element, which is usually connected to the cam via a roller in order to reduce friction;
- a fixed frame.

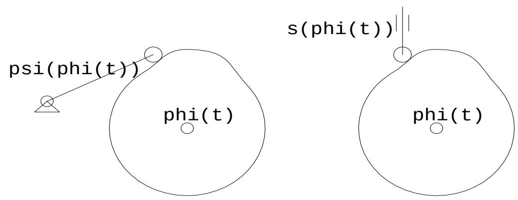

Figure 1: Basic Configurations

The basic configurations for cam design are depicted in Fig. 1. Both parts show a rotating cam disk but in the left configuration the follower performs a rotational (oscillating) motion whereas in the right one it performs a translational motion. Both configurations can be formalized in the same theoretical framework. There are also configurations where the driving element performs a translational motion (e.g. in combustion engines) but for the case of brevity and simplicity we omit these.

After having fixed a starting position (angle), the motion of the cam disk can be described by a function $\phi : [0, \infty] \rightarrow \Re, t \rightarrow \phi(t)$ where the independent variable t denotes time and ϕ denotes the rotation angle of the disk relative to the starting angle. In most practical applications, this function is a linear function since the cam disk is usually connected to a motor operated at constant angular velocity.

The starting position of the follower is determined by the geometric properties of the configuration, and the motion of the follower can be described as a function of the rotation angle of the disk, i.e. $\psi = \psi(\phi)$ in case of rotation and $s = s(\phi)$ in case of translation. Note that the latter functions which are usually called transfer functions are periodic and depend only on the geometry (basically the cam profile). Combining them with the motion function of the driving element yields the time-dependent complete motion function of the follower: $\psi = \psi(\phi(t))$ resp. $s = s(\phi(t))$. From this one can easily compute velocity and acceleration:

$$\dot{\psi} = \frac{d\psi}{d\phi} \cdot \frac{d\phi}{dt} = \psi' \cdot \dot{\phi} \quad and \quad \ddot{\psi} = \psi'' \cdot \dot{\phi}^2 + \psi' \cdot \dot{\phi} \tag{1}$$

$$\dot{s} = \frac{ds}{d\phi} \cdot \frac{d\phi}{dt} = s' \cdot \dot{\phi} \quad and \quad \ddot{s} = s'' \cdot \dot{\phi}^2 + s' \cdot \dot{\phi} \tag{2}$$

where the prime denotes the derivative with respect to ϕ . In the following we assume the angular velocity to be constant (say ω) and thus get the following simplifications:

$$\dot{\psi} = \omega \cdot \psi' \quad and \quad \ddot{\psi} = \omega^2 \cdot \psi'' \tag{3}$$

$$\dot{s} = \omega \cdot s' \quad and \quad \ddot{s} = \omega^2 \cdot s'' \tag{4}$$

Note that in this simpler setting the derivatives are equal up to a constant factor. Hence, when comparing different motion functions with respect to maximal velocities and accelerations it suffices to compare the derivatives of the respective transfer functions. In the sequel, we only consider rotational follower motions since translational ones can simply be derived by replacing ψ by s in the formulae presented below. Usually, the cam designer gets as input a so-called motion plan which consists of those parts of the transfer function which are prescribed by the technological task to be fulfilled. Figure 2 depicts a motion plan for a standard task. The follower has to rest for an interval of a certain length, then it has to be moved to a point where a certain operation is performed (e.g. a part fixed to the follower is compressed), and finally it must be moved back to the starting position where it rests until the next cycle starts.

There is a variety of issues which can be contained in a motion plan (cf. [2]). We restrict ourselves to conditions on points as well as velocities and accelerations at those points. More formally, the plan contains conditions of the following kind of information: For a given sequence of angles

$$0 \leq \phi_1 \leq \ldots \leq \phi_n \leq 2\pi,$$

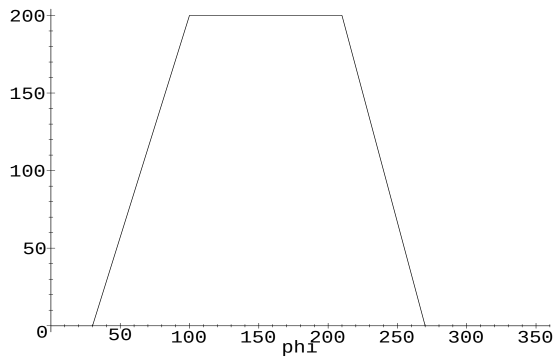

Figure 2: Motion Plan for a Dwell-Rise-Dwell-Return-Dwell Task

the values of $\psi, \dot{\psi}, \ddot{\psi}$ are given (note that from Eq. (3) one immediately gets ψ', ψ'' from $\dot{\psi}, \ddot{\psi}$ when ω is given). Moreover, it is assumed that if $\psi(\phi_i) = \psi(\phi_{i+1})$ and $0 = \dot{\psi}(\phi_i) = \dot{\psi}(\phi_{i+1}) = \ddot{\psi}(\phi_i) = \ddot{\psi}(\phi_{i+1})$ holds then the function should be constant on the interval $[\phi_i, \phi_{i+1}]$ (this is just a practical convention to make specifications of motion plans short and easy to read). Specified points are classified according to whether or not velocity and acceleration (for brevity denoted by v and a respectively in the table below) vanish. The abbreviations in the following table are derived from the German terms:

Class	Motion	Abbreviation
$v = 0, a = 0$	dwell	R
$v \neq 0, a = 0$	constant velocity	G
$v = 0, a \neq 0$	return	U
$v \neq 0, a \neq 0$	general motion	B

Table 1: Classification of points

	$v_{i+1} = 0$ $a_{i+1} = 0$	$v_{i+1} \neq 0$ $a_{i+1} = 0$	$v_{i+1} = 0$ $a_{i+1} \neq 0$	$v_{i+1} \neq 0$ $a_{i+1} \neq 0$
$v_i = 0$ $a_i = 0$	R-R	R-G	R-U	R-B
$v_i \neq 0$ $a_i = 0$	G-R	G-G	G-U	G-B
$v_i = 0$ $a_i \neq 0$	U-R	U-G	U-U	U-B
$v_i \neq 0$ $a_i \neq 0$	B-R	B-G	B-U	B-B

Table 2: Classification of intervals (tasks)

It is the task of the cam designer to add the missing parts to the motion plan such that a complete transfer function is defined. From this transfer function and additional geometric properties of the configuration the cam profile can then be computed (cf. [4]). The completion task can be split up into sub-tasks for each sub-interval where the transfer function is yet to be defined. Depending on which of the above classes a boundary point belongs to, the design task has been classified in [1] as is shown in table 2.

In order to abstract from the specifics of the task and to provide reference solution functions, the tasks have been transformed into a so-called normal form. This is simply done by moving the starting point of the sub-interval under consideration into the origin as well as by scaling abscissa and ordinate such that the starting point is (0,0) and the end-point is (1,1). This is done by performing the following transformations:

$$z = \frac{\phi - \phi_i}{\phi_{i+1} - \phi_i} \quad for \quad \phi_i \leq \phi \leq \phi_{i+1}$$

$$\psi(\phi) - \psi(\phi_i) = (\psi(\phi_{i+1}) - \psi(\phi_i)) \cdot f(z).$$

$f(z)$ is called the normal motion law (or function), and by construction we have $f(0) = 0$ and $f(1) = 1$. Note that the first (and second) derivatives of f wrt. z and ψ wrt. ϕ are equal up to a constant factor. Therefore, instead of comparing two functions for the sub-interval $[\phi_i, \phi_{i+1}]$ with respect to maximal values of their derivatives one can equally compare the corresponding normal motions. In [1], several normal motion functions are investigated and compared (according to criteria discussed below) and for each class of task recommendations are made. The formulae presented above then serve to apply the normal motion laws to the situation under consideration.

There are several criteria or quality measures used for comparing normal motion laws. First of all, the functions should satisfy the boundary conditions since otherwise there will be discontinuities in the velocity function leading to (theoretically) infinite jumps in the acceleration function or discontinuities in the acceleration function leading to so-called jumps which can cause damage in the long run or even make the mechanism completely unusable because of vibrations. It depends on the application situation and has hence to be decided by the engineer whether or not such a behavior can be tolerated. Besides these basic requirements, the maximum absolute values of the first, second and third derivatives, as well as of the product of the first and second derivatives are used as criteria for the following reasons for a more detailed exposition cf. [2]):

- a low maximum absolute value of $f'(z)$ (denoted by C_v) means that the required maximum momentum at the driving element caused by static loads on the driven element is low (in fact, both values are proportional);
- a low maximum absolute value of $f''(z)$ (denoted by C_a) means that the mass force at the driven element

is low which is of significance when the mass of the driven element dominates;

- a low maximum absolute value of $f'''(z)$ (denoted by C_j) means that the mass force at the driven element changes slowly and hence does not cause considerable vibrations;
- a low maximum absolute value of $f'(z) * f''(z)$ (denoted by C_{Mdyn}) means that the effect of the mass force of the driven element on the driving element is low.

From the above it is clear that the engineer has to decide on the importance of each quality measure for the application under consideration. Sometimes, it is also important where the maximum occurs such that the engineer should be presented the whole graph. The criteria stated above mainly cover kinematic aspects. Dynamic criteria which are particularly important for high-speed cams (cf. [3]) are only roughly outlined in [1, 2], so we did not include them in our tool.

Recently, so-called electronic cam disks have been introduced where motion control is done by varying the velocity of the driving motor (cf. [8]). This can be described within a framework quite similar to the one presented above; for space and clarity reasons, we omit the description of such a configuration but we included it in the implementation of the Maple tool described in the sections below.

Requirements for a Support Tool in Maple

The requirements listed below result from discussions with engineers who construct and produce packaging machines. They are divided into user interface and functional requirements. The former are quite important since real use of the tool by engineers depends strongly on the quality of its user interface.

User Interface Requirements

- The Maple-Tool should be as easy to use as spreadsheet programs which are quite familiar to engineers; no deeper knowledge of Maple commands and procedures should be required. In particular, error messages (in general resulting from wrong or unsuitable input) should be clear to a user not familiar with Maple. Since engineers do not work with Maple constantly they cannot be assumed to have this familiarity.
- Users must be provided with available choices when input is required; e.g. when they have to tell the program which motion law they want to investigate, the available laws should be presented.
- The user interface of the tool should be clearly structured such that the user can get quickly to those parts which are of current interest for him; e.g., if the user is only interested in investigating new normal motion functions he should not be bothered with going through other parts of the tool.

Functional Requirements

- The user should be enabled to choose floating-point precision and number of function evaluations for producing tables and graphs.
- The user should be enabled to get data on all normal motion laws investigated in the guidelines [1, 2] such that the tool serves as electronic guideline (yet, this does not include all the explanatory material in the guidelines; the engineer is assumed to be familiar with the theory of cam design). The data comprise tables, graphs and quality measures of the normal motion laws.
- The user should be enabled to experiment with new motion laws. I.e., all the data presented in the guidelines for specific functions should be computed and presented for arbitrary functions provided by the user. This can later be used in an optimization procedure. In order to support users in constructing new functions, interpolating polynomials and spline functions should be computed based on a list of points provided by the user.
- The tool should provide the (time-dependent) complete motion function and its derivatives in symbolic, tabulated and graphical form. For this, the user has to provide the motion plan as well as his choice of normal motion laws for each section of the plan including the necessary boundary values. Here, the user should be able to specify functions provided in the guidelines as well as other functions (probably resulting from experimentation). When the user inputs points of the plan, the graph of the linear interpolation should be shown such that he can check his input.

  Besides this, the user must specify the rate of revolutions such that velocity and acceleration can be computed. The tool should also present maximum absolute values of velocity and acceleration for the complete motion function. Since the complete motion function is available in symbolic form, it can serve as input for further processing (in particular: computation of fourier series and polynomials).
- The tool should compute the motion of a mechanical cam disk as well as that of an electronic cam disk but for space reasons we only consider the mechanical situation in this paper.

Design and Implementation of the Maple Program

Maple offers worksheets for communication with users. In order to offer the user a clear, task-oriented structure, three worksheets have been set up:

- First, a worksheet called "Electronic_norm.mws" contains all commands and information for providing properties of normal motion laws investigated in the guidelines as is stated in the requirements above. If the user only wants to get information on functions otherwise found in the guidelines, the use of this worksheet is sufficient.
- Secondly, a worksheet called "New_functions.mws" allows to get information on new (wrt. the guidelines), user-specified functions as stated in the requirements section. If the user wants to experiment with new functions, this is the worksheet to use.
- Thirdly, a worksheet called "Complete_motion_functions.mws" allows to construct a complete motion function from a plan, based on laws and boundary values provided by the user. In addition, this worksheet offers further processing of the motion function for computing motion functions of sliding crank mechanisms.

Sample sessions with the first and third worksheet are presented in the next section such that we do not have to go into detail here. Beside these worksheets providing the required functionality there is another help worksheet where explanations are given on the meaning of certain steps in the other worksheets. There are hyperlinks between these steps and the respective parts of the help worksheet.

The user is only required to input values of certain variables and parameters of procedures and to start procedures by simply returning on the line where the procedure occurs. Thus, no further knowledge of Maple is required. Originally, we tried to use the readstat command for input but there are problems when a user not familiar with Maple makes a mistake, e.g. forgets the semi-colon at the end. Moreover, readstat-output is not removed from the worksheet when clicking the Remove Output menu item. So, we decided to stick to the traditional approach of letting the user make changes in existing Maple lines.

In order to offer a clear structure within a worksheet, the latter is split up into sections for partial tasks and the user has to go through only those sections which are of interest to him. The user can go back and forth within the worksheet but it is clear that changes made at a certain line affect the lines above and below only if these are returned again. If the user is in doubt about the state, he can always start again with the first line containing the restart command.

The tools outlined above make use of Maple procedures in order to perform the computation required. These computations include the following:

- The computation of derivatives and expressions containing them. Here, Maple's capability of symbolic computation pays off.
- The computation of quality measures, e.g. the maximum absolute values of the derivatives of normal and complete motion functions. Since the fsolve command does not always work and since the functions under consideration might have discontinuities we decided to simply compute maxima by comparing values: search ranges are very restricted here such that for our purposes this approach is sufficient.
- The construction of functions based on information on the laws and boundary values to be used. Here, Maple's capability to deal with piecewise functions is extremely helpful: As was described in the second section, complete motion functions might consist of several parts for the different phases where tasks have to be performed or synchronized. Since the complete motion function is available in symbolic form, it can further be used in other computations. It can also be stored in a file for further processing.
- For an electronic cam mechanism, the motion function of a connected sliding crank mechanism is computed. The interesting point here is that the symbolic function describing the motion of the cam is used as input for describing the time-dependent behavior of connected mechanisms (simply by substituting an angle variable by a time- dependent angle function in the equation modeling the geometry of the mechanism).

The normal motion laws investigated in the guidelines [1, 2] are stored in data files. Data and procedures which are also currently stored in specific files are read into those worksheets which make use of them. For the next version it is planned to provide a Maple package for cam computations.

Sample Sessions

In this section we present application scenarios for two of the worksheets described above, namely for using the tool as substitute for the computational part of the VDI-guidelines and for computation of complete motion functions.

Computing Normal Motion Laws According To VDI-Guidelines

This worksheet allows the computation of normal motion laws according to VDI 2143. It consists of the following sub-sections:

+ Choice of Motion Task
+ Boundary Values and Displacement of Inflection Point
+ Computation
+ Output

The user has to input first the motion task for which he wants a normal motion law to be presented:

```
> motion_task:=[R,R]:
```

R stands for "dwell". The user has to replace R by another letter according to table 1 if he is interested in a different situation. After this, he can get information on the normal motion laws the VDI-guideline investigates for the task under consideration:

```
> choice_of_laws(motion_task);
```

$$straight\ line\ -> \ 1$$

$$quadratic\ polynomial\ -> \ 2$$

$$simple\ sinus\ -> \ 3$$

$$polynomial\ of\ degree\ 5\ -> \ 4$$

$$inclined\ sinus\ -> \ 5$$

$$modified\ trapezoid\ -> \ 6$$

$$modified\ sinus\ -> \ 7$$

He then has to choose a law (here: the inclined sinus):

```
> chosen_law:=5;
```

The user can then check in the subsequent section whether or not any boundary values are required for computation:

```
> boundary_values(yes,chosen_law);
```

$$no\ boundary\ values\ required$$

For the [R,R]-task, no values are required since the first and second derivatives are assumed to be zero at 0 and 1. If the user knows from earlier experience that no such values are necessary, he can simply skip the section (for brevity, we do not discuss inflection points here; cf. [1] for their meaning). The yes/no parameter allows to simply return the line with no effect when it is set to no. We use this way to allow the user to procede by simply returning without mouse click at several places.

Now, the user can start the computation by returning on the line:

```
> computation(chosen_law,lambda);
```

The user has to go to this line again if he makes any changes concerning laws or other values. The final section provides the output. The user can get function graphs, tables and quality measures:

```
> graph(yes);
```

Here, one gets - among others - the graph of $f'(x)*f''(x)$ shown in Figure 3.

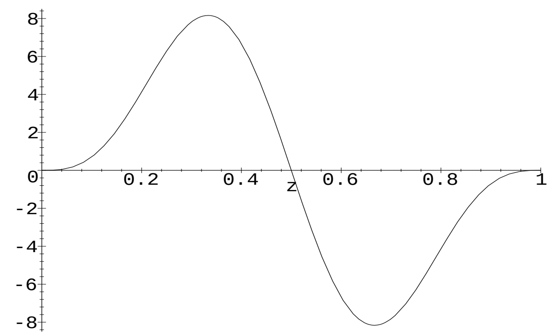

Figure 3: Graph of $f'(x) * f''(x)$

The next command computes a function table where the user can state how many evaluations he wants to have (for space reasons, we don't present the output here):

```
> table(yes,10);
```

Finally, data for the quality measures are provided:

```
> quality_measures(yes);
```

$$\left[\begin{array}{lcl} Cv & = & 2.0000 \\ Ca & = & 6.2832 \\ Cj & = & 39.479 \\ Cm & = & 8.1588 \end{array}\right]$$

Computing Complete Motion Functions

The worksheet for computing complete motion functions is structured as follows:

+ Input of Motion Plan
+ Input of Boundary Values and Motion Laws
+ Input of Data for Constructing the Complete Motion Function
+ Output for Mechanical Cam Disks
+ Output for Electronic Cam Disks

First, the user has to specify the motion plan. For this, he provides the values of the driving element (independent variable) and those for the driven element (dependent variable) as is shown below (we assume the driven element to be rotating):

```
> values_for_driving_element:=
>  [0,30,100,140,180,270,360];
> values_for_driven_element:=
>  [0,100,100,360,300,300,0];
```

From this, the motion plan is constructed and depicted such that the user can check his input:

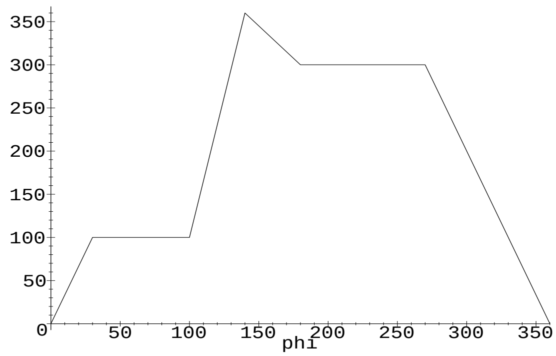

Figure 4: Motion Plan

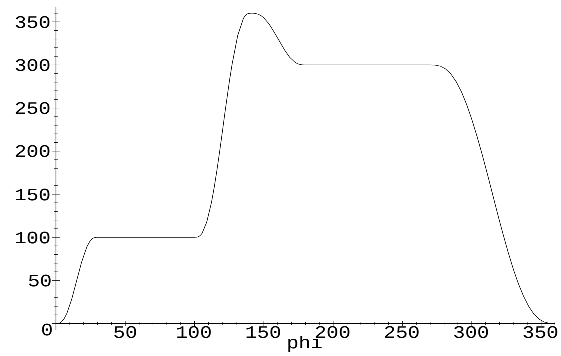

Figure 5: Angular Displacement [Degree]

At the beginning of the next section, the user has to specify the motion tasks at the points provided according to table 1:

```
> Motion_tasks:=[R,R,R,U,R,R,R];
```

In our example, we want to have a dwell at 100 degrees (e.g., in order to perform a certain packaging task); moreover, there is a return point at 360 degrees and another dwell at 300 degrees before the driven element returns to its starting position.

Now, the user gets an overview which VDI-laws can be chosen for each sub-interval. Based on this information, he makes his choice. He can also specify his own laws which have to be provided in normal form. Next, the user gets information on which boundary values are required. The following line shows the user input for our example: no values for the first derivative are required but values for the second derivative for the right boundary point of the third sub-interval and the left boundary point of the forth sub-interval are needed (the first derivative is zero since the point is specified as a return point):

```
> values_for_first_derivative:=[];
> values_for_second_derivative:=[-2,-1];
```

In the next section, the user has to provide information on the specific configuration: the rate of revolutions (for the driving element); the kind of cam disk: mechanical or electronical; the kind of motion of the driven element: rotation or translation. Then, the complete motion function is computed.

In the final sections, the user gets the graphical as well as tabulated output he wants. We just show the complete motion function as graph and as piecewise defined function. The symbolic function is not presented in the worksheet since it is only important for programmers who want to use it as input for further processing.

$$
\begin{cases}
\frac{1}{27}\,\phi^3 - \frac{1}{540}\,\phi^4 + \frac{1}{40500}\,\phi^5 & \phi \le 30 \\
100 & \phi \le 100 \\
100 + 2340.0\,(\frac{1}{40}\,\phi - \frac{5}{2})^3 & \\
-3380.\,(\frac{1}{40}\,\phi - \frac{5}{2})^4 + 1300.0\,(\frac{1}{40}\,\phi - \frac{5}{2})^5 & \phi \le 140 \\
360 + 30.0\,(\frac{1}{40}\,\phi - \frac{7}{2})^2 - 690.0\,(\frac{1}{40}\,\phi - \frac{7}{2})^3 & \\
+990.0\,(\frac{1}{40}\,\phi - \frac{7}{2})^4 - 390.0\,(\frac{1}{40}\,\phi - \frac{7}{2})^5 & \phi \le 180 \\
300 & \phi \le 270 \\
300 - 3000\,(\frac{1}{90}\,\phi - 3)^3 & \\
+4500\,(\frac{1}{90}\,\phi - 3)^4 - 1800\,(\frac{1}{90}\,\phi - 3)^5 & \phi \le 360
\end{cases}
$$

The user can also obtain the maximal absolute values of (angular) velocity and acceleration.

Having the symbolic motion function the user can easily compute the profile of a cam disk realizing the motion if in addition the necessary geometry data are provided. For illustration, we present a procedure for a rotational follower. A systematic treatment including other types and shapes of followers is contained in the VDI-guideline 2142 which could be implemented in the same way as was done for 2143. The formula for computing the curve of the centre of the roller can be found in [5].

```
> computeProfileRotation:=
> proc(motion, rr, br,d, l)
> # motion:motion function as expression
> #       in the variable phi_r in radians
> # rr: radius of the roller
> # br: radius of the base circle of the cam
> # d: distance between centres of follower
> #    and cam disc
> # l: length of the follower (from
> #    centre to centre of the roller)
> local psi_g,x,y,xdot,ydot,normal_abs,
>     x_cam,y_cam;
```

```
> # compute start angle of the follower
> psi_g:= evalf(-arccos(((br+rr)^2-d^2-l^2)
>     /(-2*l*d))):
> # compute x-coordinate function of the
> # centre of the roller rotating inversely
> # around the cam disk
> x:=unapply(d*cos(phi_r)-l*cos(phi_r
>     -motion(phi_r)+psi_g),phi_r):
> # same for y-coordinate
> y:=unapply(d*sin(phi_r)-l*sin(phi_r
>     -motion(phi_r)+psi_g),phi_r):
> # compute normal vector
> # for each curve point
>
> xdot:=unapply(diff(x(phi_r),phi_r),phi_r):
> ydot:=unapply(diff(y(phi_r),phi_r),phi_r):
> normal_abs:=unapply(sqrt(xdot(phi_r)^2
>     +ydot(phi_r)^2),phi_r):
>
> #compute cam profile coordinate functions
> #by "adding" the inbound normal vector
>
> x_cam:=unapply(x(phi_r)-rr*ydot(phi_r)
>     /normal_abs(phi_r),phi_r):
> y_cam:=unapply(y(phi_r)+rr*xdot(phi_r)
>     /normal_abs(phi_r),phi_r):
>
> [x_cam(phi_r),y_cam(phi_r)];
> end;
```

We used the above procedure for constructing the profile of the cam disk depicted in Fig. 1.

Conclusions and Further Developments

In this paper we have shown how Maple can be used to implement a very flexible tool supporting the industrial engineer in constructing efficient cam mechanisms. The flexibility and thus usefulness of the tool results essentially from the ability to perform symbolic computation which allows the engineer to experiment with new functions. There are no hard mathematics contained in the tool but the user is freed from the tedious and error- prone task to perform large symbolic algebra computation. Maple provides symbolic, tabulated and graphic output giving the engineer a complete mathematical view on the problem under consideration. Since more than a fixed set of functions must be handled, the well-known spreadsheet programs are not sufficient for implementation. In Maple, we found the possibility to define and manipulate piecewise functions extremely helpful in programming the tool and providing the complete motion function for further processing.

With respect to the user interface, one has to take into account that engineers working with many programs (CAD, FEM, etc.) can certainly not be assumed to be familiar with Maple. Here, in Maple V R4 the facilities are rather limited and spreadsheet programs offer far more to construct easy-to-use interfaces (event-driven input boxes etc.). Note however, that for Maple V R5 a spreadsheet facility is announced.

There are several directions of further research on the topics presented in this paper:

- In some situations, motion plans might be more complex than the ones our tool takes as input: For example, it could be required that the motion of the cam be synchronized with the motion of another part of the overall mechanism. Then the motion plan does not only contain points and derivatives but function pieces for sub-intervals where synchronization is needed.
- Having a tool to compute quality measures for arbitrary normal input functions, the next step is to use this for optimization purposes. A promising way to do this in a Matlab environment (containing Maple as a so-called extended symbolic toolbox) was already described in the work of [9].
- The symbolic output of the Maple tool can be used for computing fourier series and polynomials important for vibration reduction (cf. [6]).

Finally, our approach is applicable to other VDI-guidelines on cam design (as well as on other topics in engineering) which also contain a lot of symbolic algebra.

References

[1] VDI-Richtlinie 2143 Blatt 1: *Bewegungsgesetze für Kurvengetriebe. Theoretische Grundlagen*, VDI-Verlag, Düsseldorf, (1980). (also available in English, cf: http://www.vdi.de)

[2] VDI-Richtlinie 2143 Blatt 2: *Bewegungsgesetze für Kurvengetriebe. Praktische Anwendung*, VDI-Verlag, Düsseldorf, (1987). (also available in English, cf: http://www.vdi.de)

[3] E. Pennestri and V. Falasca: Design of Cam Mechanisms, *MapleTech*, **3-1**, pp. 74–78 (1996).

[4] J. Angeles and C.S. Lopez-Cajun: *Optimization of Cam Mechanisms*, Kluwer, Dordrecht, (1991).

[5] J. Vollmer: *Kurvengetriebe*, 2nd. Edition, Hüthig, Heidelberg, (1989).

[6] R. Braune: HS-Profile mit vielen Harmonischen Wirkungsvolle Schwingungsreduzierung in Kurvengetrieben bei extremen Bewegungsanforderungen, *VDI- Berichte No. 1111* , pp. 127–154, (1994).

[7] A.M. Cohen and A. Heck: Applied Computer Algebra: Experience from Cam Design, J. Fleischer et al. (ed.): *Computer algebra in science and engineering*, World Scientific, Singapore, (1995).

[8] J. Eidam: *Beurteilung und Simulation des Betriebsverhaltens von lagegeregelten elektrischen Direktantrieben als "Elektronische Kurvenscheibe"*, VDI-Verlag, Düsseldorf, (1997).

[9] R. Bacher: Combining Symbolic and Numeric Tools for Power System Network Optimization, *MapleTech*, **4–2**, pp. 41–51, (1997).

Biographies

Burkhard Alpers received his PhD in mathematics from the University of Hamburg. For several years, he worked at the Corporate R&D Division of Siemens on topics in network management. He is currently professor of mathematics and computer science in the Department of Mechanical Engineering at the Aalen University of Technology and Business. He is interested in educational and industrial applications of computer algebra, optimization and databases.

Stephan Steeb received a master's degree in mechanical engineering from the Aalen University of Technology and Business. He is currently working with an engineering company offering consulting and computational services.