ELSEVIER

Robotics and Autonomous Systems 30 (2000) 249–260

Robotics and Autonomous Systems

www.elsevier.com/locate/robot

Computer support for mechatronic control system design

Job van Amerongen *, Erik Coelingh, Theo J.A. de Vries

Control Laboratory, EE Department, Drebbel Institute for Systems Engineering, University of Twente, EL-RT, P.O. Box 217, 7500 AE Enschede, Netherlands

Received 10 May 1999

Abstract

This paper discusses the demands for proper tools for computer aided control system design of mechatronic systems and identifies a number of tasks in this design process. Real mechatronic design, involving input from specialists from varying disciplines, requires that the system can be represented in multiple views. Several tools are already available but there are still substantial shortcomings. The paper gives indications about the developments needed to come to better design tools in the future. A specific example is worked out in more detail, i.e., automated performance assessment of mechatronic motion systems during the conceptual design stage. ©2000 Elsevier Science B.V. All rights reserved.

Keywords: Mechatronics; CACSD; Controller design; Modeling; Simulation

1. Introduction

A useful definition of *mechatronics* is 'a technology which combines mechanics with electronics and information technology to form both functional interaction and spatial integration in components, modules, products and systems' [5]. This design approach differs from classical patterns, where the design starts with the mechanical subsystems, then the electrical subsystems and finally the controllers. In order to form functional interaction and spatial integration, subsystem designs need to overlap, and hence simultaneous involvement of several disciplines needs to be realized in a coordinated way. The design process of controlled electro-mechanical systems with a mechatronic design approach is therefore generally more complex, but leads to systems with a superior price–performance ratio. A design problem in general is 'ill-structured in the large, but well-structured in the small' [23], i.e., the complex, difficult to tackle design problem can be mastered by splitting it up into small well-structured problems for which a 'local' solution can be found (Fig. 1).

In the same way, the complexity of mechatronic design can be tackled. However, the solutions to the well-structured smaller problems should not — as in the classical approach — be considered just as 'local' solutions. Solutions to domain-specific subproblems should be formulated while taking into account the consequences of the solution in other domains and by considering alternative solutions in these other domains.

When designing systems with this approach, the use of computer tools is indispensable. Such tools facilitate (automated) manipulations of the proposed design, allow to record and browse through relevant knowledge and experience, and document the design process. The rapid technological developments and the

* Corresponding author.
E-mail address: j.vanamerongen@el.utwente.nl (J. van Amerongen).

PII: S0921-8890(99)00090-1

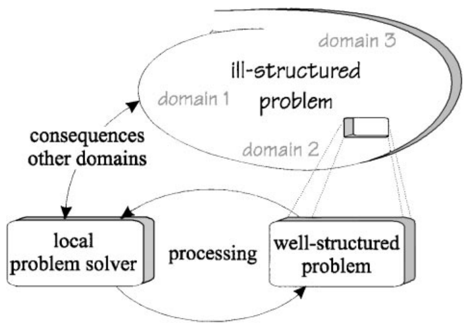


Fig. 1. Schematic structure of solving ill-structured problems (adapted from Simon [23]).

continuing need for reduced times-to-market of new products demand more and more advanced tools in the design stage.

In this paper, we focus on aspects related to computer aided *control system* design (CACSD) of mechatronic systems. We present and discuss some developments in this area. In Section 3, we try to put the contributions in the session on ‘Computer Support for Mechatronic Control System Design’ of ICAM’98 [20] in perspective. After that, we discuss the issues not addressed in the session and give a brief indication of the current state-of-the-art as perceived by us. A specific example, the conceptual design of a plotter, is worked out in more detail. Specific problems that occur during conceptual design of mechatronic systems are [7]:

- the functional interaction between domain-specific subsystems;
- consequences of solutions and alternative solutions in other domains;
- prediction of guaranteed performance of a particular solution.

The example will illustrate the use of a computer-based design tool for conceptual design of mechatronic motion systems that addresses these problems and that will give a feasible design proposal that is likely to meet the desired performance.

2. Design issues

When we consider the design of, e.g., a new A0-plotter, several aspects are important:

- the look of the plotter;
- required accuracy;
- speed;
- functionality, such as color or black and white;
- manufacturing aspects as well as maintenance and recycling issues;
- price, etc.

Although some of these design issues seem to be hardly related, they all influence each other. The total problem is ill-structured, but parts can be solved in a well-organized way. For instance, the influence of the weight of the penholder on the speed and accuracy of the plotter could be investigated by means of simulation tools.

To do simulations of the dynamic performance of the plotter it is not needed to have a detailed description of all the components involved. But it is essential that the weight distribution is well known. This requires support that, for instance, easily enables changes made from an aesthetic point of view to be evaluated in dynamic simulations. On the other hand, the dynamic simulations should give indications about how certain components should be changed or placed in a different location. When design decisions are more closely related, a proper maintenance of relations between different aspects of the design is even more important. This demands for a kind of ‘core language’ or ‘core model’ that represents the design object internally in the computer and that can be viewed from various windows. Only then proper information can be given to all designers involved.

When we focus on the design of a control system alone, a representation in the form of transfer functions or a state space description would be appropriate for linear or linearized systems. From such a representation, plot windows with the behavior in the time domain, frequency domain or s-plane can easily be given. In addition, performance criteria can be displayed, indicating whether the system performs optimally. It would be a great advantage if changes made in one domain (window) were reflected immediately in the other windows. Even such simple systems are not yet commonly available.

3. Computer aided control system design (CACSD) in a mechatronic setting

In the light of the characterization of the design process given in Section 1, we can distinguish the following tasks of the mechatronic control system designer:

1. Isolation of a *well-structured control problem* specification from an ill-defined system design specification.
2. Formulation of a *competent plant model* given an incomplete and indefinite system design proposal.
3. Selection of an appropriate control strategy.
4. *Implementation and realization* of a control system according to the selected control strategy.
5. Assessment of the attainable *performance and expected cost* (in terms of time, effort and money) of the controlled system.
6. *Communication* of the consequences, expectations, demands, desires, etc. with respect to the proposed system design from the perspective of the control engineers.

Although presented as separate tasks, the above items have a strong mutual dependence. The important implication of this is that design software that addresses just one of these tasks is not very useful. In general terms, we can state that the more tasks are supported, the more powerful a computer tool will become.

Hereafter, we shortly review some relevant commercial and academic computer tools.

3.1. Matlab/Simulink

In controller design, computer aided design tools are reasonably well developed and used. The well known and widely used package Matlab is now a standard tool for the control engineer and is part of many modern text books on control. Matlab supports the implementation and realization of mainly linear control systems (task 4) by using plant models in the form of state space descriptions and transfer functions, in the Laplace domain, frequency and time domain (optimal control). Nonlinear plant descriptions can be considered in the design process by means of the Simulink toolbox. Simulink is a simulation program based on block diagram input; it can be used to obtain a linear plant model (task 2) and to assess the performance of the controlled nonlinear system (task 5). Many programs provide interfaces to Matlab. Products like dSpace [13] translate controllers designed in Matlab directly into code for a digital signal processor (DSP), enabling fast prototyping.

The papers in the session on ‘Computer Support for Mechatronic Control System Design’ of ICAM’98 [10,16,18] discuss one or more of the above-mentioned tasks and describe tools to handle the related design issues.

3.2. Schemebuilder Mechatronics

Schemebuilder Mechatronics [22] and its knowledge base are developed for handling many types of machine design using feedback control and servo systems. It can invent or evolve an existing design solution and, using its knowledge database, the solution can be built with the help of an expert system, which provides advice for the designer when making design decisions. In their contribution [10], Counsell and Porter describe the design principles for controller design and their representation in the knowledge base. As an illustration of the power of Schemebuilder, its application for controller design of servo-pneumatic machine is shown.

Counsell and Porter [10] focus on task 3 (quite unique). Schemebuilder Mechatronics also addresses task 2, in terms of *physical system models*, which is a more powerful way than Simulink, and addresses task 4 and task 6.

3.3. CSDA

Maekawa [18] presents a user-friendly and comprehensive control system design package called control system design automation (CSDA). The system consists of five main blocks: a requirement interpretation block, a modeling block, an analysis/design block, a database management and knowledge base block, and a verification block. The requirement interpretation block transforms the specifications in terms of the application to those in terms of control. The analysis/design block selects an optimal control structure and determines the controller parameters. In addition to the conventional design methods, CSDA also contains the more recent design methods such as the LMI design approach and the Kessler/Manabe method.

This system is an extension to Matlab and supports tasks 1–5. Quite unique are the modules for task 1 and 3. The system specifically intends to let inexperienced designers complete a satisfactory control system design.

3.4. MATX/RTMATX

Koga and Sampei [16] describe the development of a CAD-tool for control systems, which realizes an integrated environment for simulation and real-time implementation. It enables one to do not only analysis of control systems and design of controllers, but also simulation and real-time implementation with a digital controller in the same environment. By utilizing this software, the control engineer is able to repeat the procedure of design of control systems efficiently to achieve the best performance in much shorter time. It is also shown that a real-time control program is generated from a simulation program with a minimum change, by making two programs with the same structure. The system of Koga and Sampei [16] focuses on task 4, and provides an alternative for the Matlab Real Time Workshop and Simulink. Unique feature is that routines for the design of control systems can be included in the experimental software.

3.5. 20-sim

The example at the end of this paper addresses support for task 5. New in here is that already at an early stage of the design process, when many decisions remain to be taken, a reasonably accurate performance assessment can be made. In related work [7,8], support for tasks 1, 2, and 6 has been discussed. Like Counsell and Porter [10], *physical system models* are used in the form of bond graphs as available in the 20-sim environment. The computer program 20-sim [1,4] (pronounce 'Twente Sim') supports input of models in the form of bond graphs, in addition to model input in the form of iconic diagrams, block diagrams and equations. It supports submodels, organized in a hierarchical way. This makes the program well suited to reuse models developed in earlier projects. Version 3.0 of 20-sim fully supports an object-oriented approach to modeling, in addition to a number of other new features [2,25,26]. 20-sim supports the modeling process itself and has an extremely fast simulator, through its built in compiler and advanced simulation algorithms. The built in optimization feature allows (physical) parameters in different domains to be tuned for optimal performance. As an interface to other environments, a C-code generator automatically converts complete models or submodels (e.g., controllers) into C-code in the form of ANSI-C functions or ANSI-C stand alone code, as well as Simulink S-functions. It can generate linear state-space models from nonlinear physical models, either as a 20-sim equation model or as a Matlab m-file.

4. Task 6: Communication

A major future issue will be the support for task 6. First of all, this will involve appropriate training and habit forming of the members of a mechatronic design team. Computer support can help here if it would enable each member of the design team to work with the tools most suitable for him, but still using up to date system models that consistently describe one and the same proposed design. This 'dream situation' is far beyond the current state-of-the-art. However, promising developments are being made, which can be categorized in two groups:

- A number of 'general purpose' modeling systems, both commercial and academic, is becoming available that allows to build dynamic behavior models using discipline-specific icons. Examples are, e.g., Saber [21], ICAP [15], the before-mentioned 20-sim 3.0 [1], Schemebuilder [22] and Camel [6]. This is important, as functional interaction is generally realized through dynamic interaction; the mentioned software systems hence allow to model this interaction at a system level, while representing discipline-specific parts of the model in discipline-specific ways.
- Domain-specific modeling systems are given the capability to exchange data while doing analyses. For example, the mechanical dynamic modeling environment DADS [11] can be linked to Matlab. In this way it can run simulations of controlled mechanical plants. Typically, these connections make use of techniques like object linking and embedding (OLE2) and dynamic data exchange.

In our view, two problems yet remain to be attended by the scientific community:

1. Typically, models that are competent for aspects important to one discipline, are incompetent in another discipline. That is, a transformation of the models is needed. For example, a DADS model of a mechanical plant is unwieldy and too complex to be used for control system design; it needs to be

simplified and stated in other terms first. The formulation of appropriate transformation algorithms and the (speedy) realization thereof in software is an important issue.

2. When the number of software tools contained in the tool suite (or the abilities of a single tool) grows, there is an increasing need for support in order to guarantee consistent models and to use the tools in a coordinated manner. This requires a proper interface.

5. Interface

A well-defined interface is crucial to enable different representations of the design object, while simultaneously maintaining a consistent model. Control engineers are used to models in the form of transfer functions or state space descriptions. Without special precautions, such models generally do not have a direct relation to the physical parameters in the system. This is a serious disadvantage when not only the structure and parameters of the controller are being considered. If the physical system that has to be controlled is not taken for granted, suggestions made for modification of the physical system during the design of the controller have to be translated into physical parameters. This implies that the parameters in the transfer function should be directly and dynamically related to the parameters of the physical system. The design method described in the next sections is a first step towards the goal of a design system that directly relates modifications in the controlled system to changes in the plant. The parameters in the simplified model have a known — may be complex — relation to physical parameters of the plant. The bond-graph language is used to relate the physical reality to the more simple transfer function based model used for the controller design. Bond graphs are able to capture all the relevant physical characteristics, necessary to do simulations as well as to indicate which physical parameters should be changed in order to achieve the desired performance. The bond-graph representation is in principle domain-independent, but domain-specific models can easily be converted into a bond graph. Bond graphs can be automatically converted into domain-specific models, provided that additional information about the domain as well as geometrical information is stored together with the bond graph [25]. 20-sim allows domain-specific information to be added to the bond graph [1]. The conversion from a model that is closely related to a domain-specific physical system into a more abstract model is relatively easy and is a one to one mapping. But after simplification, the more abstract model can not be converted back by a one to one mapping into the original model domain. This introduces 'additional freedom': one parameter in the more abstract model maps to various combinations of parameters in the physical model and not all combinations may be feasible. The physical parameters will be subject to constraints, which will limit the real freedom of choice.

6. Conceptual design tool

In the conceptual design stage 'a rough idea is developed of how the project will function and what it will look like' [24]. It is an early stage in the design process that is well suited to establish the functional interaction between different subsystems.

In the following we will consider the design of an A0-plotter. A typical result of the first steps in the conceptual design of the plotter may be the sketch of Fig. 2. The plotter has to move a pen (1) across a sheet of paper that is placed in the x–y plane. The pen moves across a shuttle (2) in the y-direction. The shuttle can move in the x-direction, supported by two guidances (3 and 3*) and driven by a motor (12) through a transmission (8, 9 and 10), another transmission (6 and 6*)

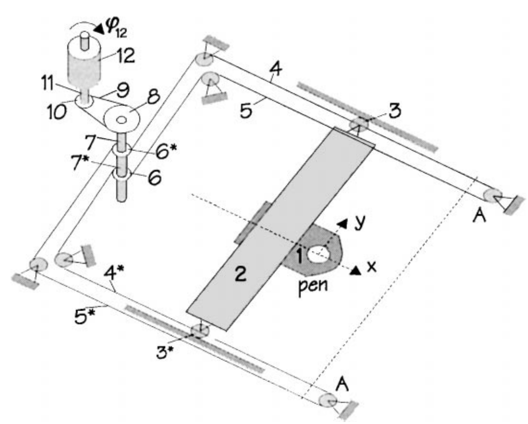


Fig. 2. Sketch of the A0-plotter to be designed.

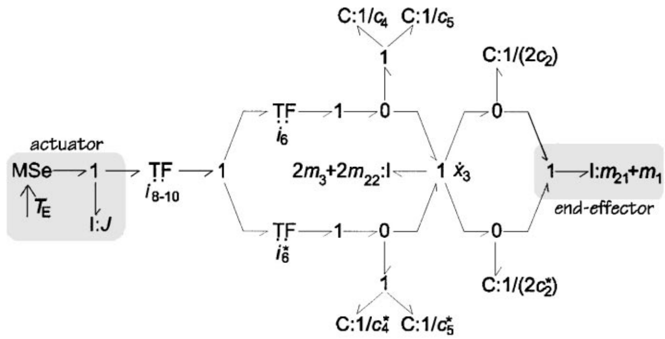


Fig. 3. Initial bond-graph model of A0-plotter.

and four timing belts (4, 4*, 5 and 5*). The motion of interest is the motion in the *x*-direction. The technical specification (task) for this motion is that the pen should be able to move over a distance $h_{\mathrm{m}} = 0.5\,\mathrm{m}$ within the motion time $t_{\mathrm{m}} = 1\,\mathrm{s}$. The maximum allowable positional error after the motion time is $u_0 = 0.1\,\mathrm{mm}$.

To verify whether this is a feasible specification, an indication of the dynamic behavior of the controlled system is required. Controllers are designed on the basis of a model of the plant. In the conceptual design stage these models should have the following characteristics [19]:

- simple, low order;
- small number of parameters,

such that the model of the controlled system provides:

- reliable estimates of the dominant dynamic behavior;
- reliable estimates of the attainable bandwidth of the controlled system.

Models with these characteristics will provide an easy to understand, sufficiently accurate and efficient framework for continuation of the design.

A bond-graph model of the design proposal of Fig. 2 is shown in Fig. 3. Note: it is assumed that the shuttle cannot rotate, such that the velocities of guidances 3 and 3* are equal. The definition and initial estimates of the parameter values in the model are given in the Appendix A. In a later stage of the design process more accurate and more definitive parameter values have to be determined.

The specifications have to be met for the worst-case situation, i.e., the situation where the shuttle (2) is in position AA and the penholder is located at the center of the shuttle.

For fourth-order electro-mechanical system models, Groenhuis [14] developed a design method for the minimization of the positional error after a change in the input function (point-to-point motion) when using a PD controller. This design method provides recommended values for the combined parameter set of the controller, path generator, plant and motion specification. Hence, this method takes into account the interaction between the domain-specific subsystems. It is a powerful method, which can be applied in several ways, as it advocates a true mechatronic design approach [7].

In the conceptual design stage one generally comes up with a model with too many parameters and too little knowledge to estimate appropriate parameter values. Model simplification and reduction techniques can be applied iteratively to reduce the number of parameters and the model order. Subsequently, the Groenhuis design method can be used to find parameter values for the controlled system, as shown in Fig. 4.

To allow fast and correct model reduction, computer-based support has been developed that can reduce models of mechatronic plants to fourth-order models. The outcome is a model where the representations of the subsystems are generally reduced to a mass for the end-effector, a compliance for the transmission and a mass with an applied force for the actuator (mass-spring-mass model). The parameters in this reduced-order model are a combination of the parameters of the original model.

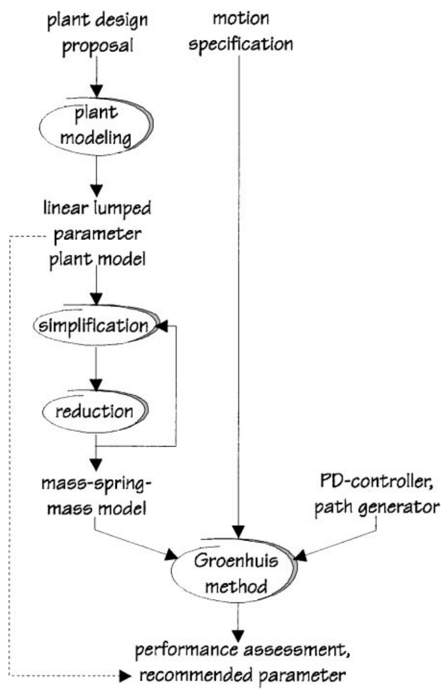


Fig. 4. The automated conceptual design process.

6.1. Automated model simplification

The simplification algorithm minimizes the number of elements in a model by eliminating transformations and by joining elements. Bond graph models are being used, because simplification rules have been formally described and are applicable in any energetic domain [3]. For a short list of common simplification rules and the procedure implementing these rules one is referred to the original ICAM'98 publication [9]. Here, the results of automated model simplification of the model of the A0-plotter are illustrated.

Fig. 5 shows a possible intermediate step of the simplification where the transmissions are collected just after the actuator, to allow the choice of transmission ratios according to inertial match [17].

Some parameters in Fig. 5 are a composition of parameters in the initial bond graph of Fig. 3. The dependency between these parameters is maintained in the software by means of parameter relations. The parameter relations for Fig. 5, including their values, are:

$$c_{45} = c_4 + c_5 = 1.0 \times 10^4\,\mathrm{N\,m^{-1}}, \tag{1}$$

$$c^*_{45} = c^*_4 + c^*_5 = 1.0 \times 10^4\,\mathrm{N\,m^{-1}}, \tag{2}$$

$$m_{23} = 2m_3 + 2m_{22} = 0.474\,\mathrm{kg}, \tag{3}$$

$$m_{\mathrm{end}} = m_1 + m_{21} = 0.47\,\mathrm{kg}. \tag{4}$$

If the model does not contain power loops, further propagation and composition of transmissions will lead to a model without transmissions. The final plotter model after completion of the simplification algorithm is shown in Fig. 6.

The parameter relations in this figure are:

$$c_{\mathrm{belts}} = c_{45} + c^*_{45} = 2.0 \times 10^4\,\mathrm{N\,m^{-1}}, \tag{5}$$

$$i = i_{8-10} \times i_6 = 1.65 \times 10^{-3}\,\mathrm{m}, \tag{6}$$

$$m_{\mathrm{act}} = \frac{J_{\mathrm{m}}}{i^2} = 0.92\ \mathrm{kg}. \tag{7}$$

Fig. 6 shows that the model simplification algorithm decreases the complexity of the model structure. Simultaneously, the complexity of the parameter relations in the model increases, as shown by the equations. What is gained in this procedure is that the composite parameters are more easily interpreted and related to the controller, path generator and motion specification.

6.2. Automated model reduction

The resulting model after application of the simplification procedure is generally not in the form of

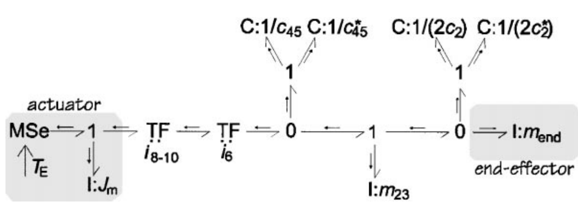


Fig. 5. Intermediate step in simplification procedure.

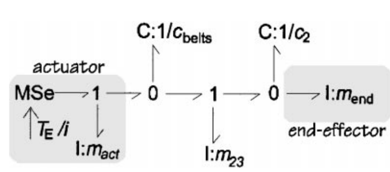


Fig. 6. Simplified bond graph model.

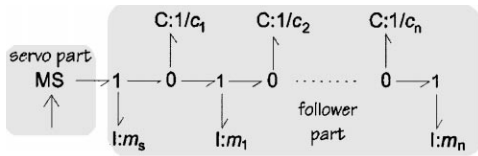


Fig. 7. Chain structure.

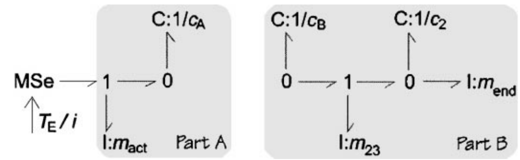


Fig. 8. Split plotter model.

the mass-spring-mass system required by the design method of Groenhuis. To reduce the order of the model and convert it to the required form, a reduction algorithm has been developed, in which the rigid body mode and the lowest mode of vibration in the model are preserved. This reduction algorithm is applicable to two common types of model structures: the *chain* structure (Fig. 7) and the *fork* structure. The fork structure consists of three chain structures connected by a 0-junction. Reduction of fork structures is similar to reduction of chain structures, therefore only the latter is described. Reduction of the chain structure is performed by dividing the follower part (see Fig. 7) into two subchains that have equal stiffness. The masses in both subchains reflect the mass ratio of the system. A simple search algorithm is used for this purpose. The two subchains are reduced to a mass-spring model and finally combined to a mass-spring-mass model. Two possible approaches for reducing subchains, i.e., intuitive reduction and Rayleigh's reduction method [12], are shortly described in the original ICAM'98 publication [9]. Here, only the application of the numerical reduction algorithm [9] to the plotter model is shown.

The lowest mode of vibration of the plotter model can be calculated directly from the bond graph model. Once the total stiffness c_c of the model has been determined, the equivalent mass m_{eq} can be calculated, such that the lowest mode of vibration is obtained. No approximation error will be made, but the same restrictions as in Rayleigh's method [12] exist to ensure the lowest mode of vibration to be dominant. The reduction algorithm will be applied to the 3-DOF model that is obtained after simplification of the plotter model. This model has a chain structure. First the follower part is split up in two parts (A and B) that have equal stiffness.

The overall stiffness of the model is:

$$c_{tot} = \left(\frac{1}{c_{belts}} + \frac{1}{c_2}\right)^{-1} = 1.93 \times 10^4 \;\; \mathrm{N\,m^{-1}}, \qquad (8)$$

so the stiffness c_A and c_B are:

$$c_A = \left(\frac{1}{2}\frac{1}{c_{tot}}\right)^{-1} = 2c_{tot} = 3.86 \times 10^4 \;\; \mathrm{N\,m^{-1}}, \quad (9)$$

$$c_B = \left(\frac{1}{2c_{tot}} - \frac{1}{c_2}\right)^{-1} = 4.15 \times 10^4 \;\; \mathrm{N\,m^{-1}}. \qquad (10)$$

Part A is a mass-spring model and needs no reduction. Part B has to be reduced. The natural frequencies of this part are:

$$\omega_1 = 2.08 \times 10^2 \;\; \mathrm{rad\ s^{-1}}, \quad \omega_2 = 1.54 \times 10^3 \;\; \mathrm{rad\ s^{-1}}. \qquad (11)$$

Modeling the lowest mode of vibration results in the following values for the stiffness and equivalent mass:

$$c_c = 2c_{tot} = 3.86 \times 10^4 \;\; \mathrm{N\,m^{-1}}, \quad m_{eq} = \frac{2c_{tot}}{\omega_1^2} = 0.89 \;\; \mathrm{kg}. \qquad (12)$$

Connecting part A and the reduced part B results in the reduced-order model of Figs. 8 and 9.

6.3. Design

A special editor has been developed that allows explorational design of controlled fourth-order systems. Constraints are used to represent the dependencies between variables, design parameters and diagrams. If a constraint variable changes, other variables are immediately updated by constraint satisfaction techniques,

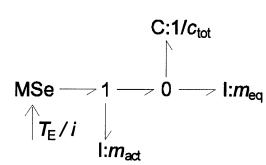


Fig. 9. Reduced-order model of the plotter.

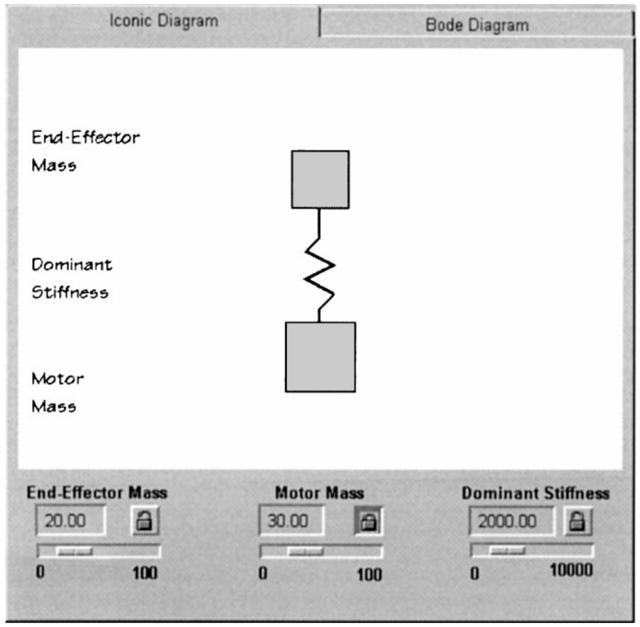


Fig. 10. Iconic diagram with variables.

such that the relations defined by Groenhuis are always valid [7].

If, for example, the motor mass is increased by dragging the slider, the icon of the motor mass in Fig. 10 will also increase, the natural frequency will decrease and the error will increase for a given reference path. These changes are represented numerically and graphically in an open- and closed-loop Bode diagram and a time response.

In either of the reduction methods the total mass of the reduced model is usually lower than the total mass of the original model. The servo parameters in the Groenhuis design tool are made proportional to the total of the mass of the non-reduced model, to allow the application of Groenhuis' results to the original model.

A possible application of this editor to the reduced-order model of the A0-plotter may consist of the following (partially automated) steps:

1. The parameter values of the masses and dominant stiffness of Fig. 9 are automatically entered in the editor of Fig. 10.
2. The specifications (task), describing the desired point-to-point motion in terms of the movement h_m and motion time t_m are entered in an editor similar to Fig. 10. This editor graphically shows the reference path.
3. The location of the position sensor is indicated. For the A0-plotter it is located on the motor axis.
4. A second-order reference path is selected, i.e., a parabolic path.
5. The design tool now automatically indicates the values for the PD controller, using constraint satisfaction techniques on the rules proposed by Groenhuis [14].
6. An estimate of the positional error for the specified task is indicated. For the A0-plotter this error u_0 equals 0.18 mm, which is larger than the desired positional error of 0.1 mm.
7. The designer may use sliders and locks (as in Fig. 10) to find out the consequences of changes in the physical parameters or the design parameters in an explorational way.
8. It is possible to determine what the value of the dominant stiffness c_tot has to be when the maximum positional error is 0.1 mm. This stiffness equals $3.5\times10^4\,\mathrm{N\,m^{-1}}$.

Simulations of the PD-controlled plotter will show that the specifications are met in case the plant is represented by the model of Fig. 9, with the new value for the dominant stiffness c_{tot}. However, the specifications have to be met when using the model of Fig. 3 for the plant and the new value of c_{tot} has to be mapped onto stiffnesses in this initial model. A possible continuation of the design consists of the following steps:

9. The dominant stiffness in the reduced-order model consists of the sum of the stiffnesses of the four timing belts and the shuttle. These stiffnesses can be assigned a new value in an explorational way using sliders and locks as in Fig. 10, using constraint satisfaction techniques.
10. Changing the stiffness of the shuttle c_2 possibly requires modifications to the proposed construction. It is easier to select a different type of timing belt. The required stiffness per timing belt equals $9.4\times10^3\,\mathrm{N\,m^{-1}}$, therefore commercially available belts of type 3 (Appendix A) are selected.

6.4. Evaluation

Now we have a feasible design proposal for the A0-plotter that is likely to meet the desired performance. A simulation of the controlled system, with the plant model of Fig. 3, is shown in Fig. 11. This simulation shows:

- A — the reference path,
- B — the position of the penholder (1),
- C — the error between the reference path and position of the penholder.

The positional error, i.e., the maximum error after the motion time $t_m = 1$ s, is $u_0 = 0.06$ mm for $h_m = 0.5$ m.

The specific problems that occur during conceptual design of mechatronic systems are addressed by the presented design tool. Functional interaction between domain-specific subsystems and consequences of solutions and alternative solutions in other domains, are dealt with by the machinery of the Groenhuis design tool. The computer support provides the designer with transparency in the relations between the design parameters; sliders and locks can be used to (not) change the parameters. If one parameter is changed, others will change automatically according the underlying constraints, so the designer can evaluate the interaction between different subsystems in an explorational design mode. Local design goals can easily be changed, while information about the consequences of this change is readily available.

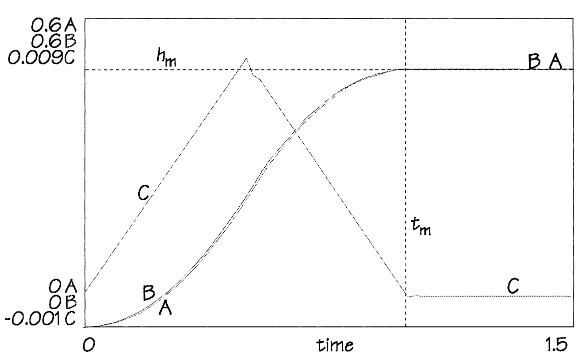


Fig. 11. Simulation of the controlled plotter model.

7. Conclusions

In this paper it has been shown that the design of a mechatronic system can be split into a number of distinct tasks. Tools to solve these tasks are becoming available. A single design environment able to cover all these tasks is not (yet) available. From the perspective of control system design, Matlab is a de facto standard for the design of all the elements of the system that are directly related to the controller. With additional toolboxes such as the Simulink toolbox (for non-linear simulations), State flow (for the design of the computer-based controller), as well as additional hardware such as dSpace (for the realization of controllers in DSPs), prototypes of control systems can relatively easily be realized. But from a mechatronic design perspective, the standard Matlab tools are not sufficient. The tools presented in this session are examples. For instance, modeling of mechatronic systems may be much easier with a modeling and simulation program such as 20-sim, that supports modeling in terms of physical components, than with a block diagram based tool like Simulink. The need for a proper interface between the different tools has been identified as one of the key issues to come to a real mechatronic design approach. Multiple views on the design including the possibility to let changes in one view, simultaneously and instantly, influence the views in another domain are essential for a real mechatronic design environment.

An example has been given of an interactive computer-based support tool, developed for conceptual design of mechatronic systems, using constraints, such that it:

- supports the complete conceptual design stage for mechatronic systems;
- supplies design automatons for fast and correct model simplification and order reduction;
- provides transparency in the relations between different design parameters;
- supports application of the Groenhuis design tool in an explorational design mode;
- can apply the results of the Groenhuis design tool to the initial model in an explorational way;
- puts emphasis on the interpretation of the results instead of the application of procedures.

The principal benefits are that it quickly provides insight into the design problem and that feasible goals and required design efforts can be estimated at an early stage.

Appendix A. Properties of the A0-plotter

Mass of the penholder (1)	$m_1 = 0.2\,\mathrm{kg}$
Mass of the shuttle at (1)	$m_{21} = 0.27\,\mathrm{kg}$
Stiffness of the shuttle (2)	$c_2 = 5.5\times10^5\ \mathrm{N\,m^{-1}}$
Mass of the shuttle at (3 and 3*)	$m_{22} = 0.137\,\mathrm{kg}$
Mass of the guidances (3 and 3*)	$m_3 = 0.1\,\mathrm{kg}$
Stiffness per timing belt	
Type 1:	$c = 0.5\times10^4\,\mathrm{N\,m^{-1}}$
Type 2:	$c = 1.0\times10^4\,\mathrm{N\,m^{-1}}$
Type 3:	$c = 1.5\times10^4\,\mathrm{N\,m^{-1}}$
Radius of pulleys (6 and 6*)	$r_6 = 5\times10^{-3}\,\mathrm{m}$
Transmission	$i_{8\text{–}10} = 0.33$
Motor inertia	$J_\mathrm{m} = 2.5\times10^{-6}\,\mathrm{kg\,m^2}$

References

[1] 20-sim, http://www.rt.el.utwente.nl/20sim

[2] A.P.J. Breunese, Automated support in mechatronic systems modelling, Ph.D. Thesis, University of Twente, 1996.

[3] P.C. Breedveld, J. van Amerongen, Dynamische systemen: Modelvorming en simulatie met bondgrafen, Dictaat Open University, Heerlen, The Netherlands, 1994 (in Dutch).

[4] J.F. Broenink, Modelling, simulation and analysis with 20-sim, Journal A 38 (3) (1997) 22–25.

[5] J. Buur, A theoretical approach to mechatronics design, Ph.D. Thesis, Institute for Engineering Design, Technical University of Denmark, 1990.

[6] Camel, http://MLaP.Uni-Paderborn.de

[7] H.J. Coelingh, T.J.A. de Vries, J. van Amerongen, Automated conceptual design of controllers for mechatronic systems, in: Proceedings of the International Workshop on Engineering Design, CACD'97, Lancaster, UK, 1997, pp. 27–45.

[8] H.J. Coelingh, T.J.A. de Vries, J. van Amerongen, Automated conceptual design of mechatronic systems, Journal A 38 (3) (1997) 26–29.

[9] H.J. Coelingh, T.J.A. de Vries, J. van Amerongen, Automated performance assessment of mechatronic motion systems during the conceptual design stage, in: Proceedings of the International Conference on Advanced Mechatronics, Okayama, Japan, 1998, pp. 472–477.

[10] J.M. Counsell, I. Porter, Schemebuilder mechatronics: Design principles for controller design, in: Proceedings of the International Conference on Advanced Mechatronics, Okayama, Japan, 1998, pp. 454–459.

[11] DADS, http://www.cadsi.com; http://www.cadsi.com/dadsplant.html

[12] J.P. den Hartog, Mechanical Vibrations, McGraw Hill, New York, 1956.

[13] dSpace: http://www.dspace.de

[14] H. Groenhuis, A design tool for electromechanical servo systems, Ph.D. Thesis, University of Twente, Enschede, The Netherlands, 1991.

[15] ICAP: http://www.softsim.com/eanal/icap4o.htm

[16] M. Koga, M. Sampei, An integrated software environment for design and real-time control of mechatronic systems, in: Proceedings of the International Conference on Advanced Mechatronics, Okayama, Japan, 1998, pp. 466–471.

[17] M.P. Koster, W.T.C. van Luenen, T.J.A. de Vries, Mechatronica, Course Material, University of Twente, Enschede, The Netherlands (in Dutch).

[18] K. Maekawa, Control system design automation for mechanical system — Machine tool module and control parameters calculation module, in: Proceedings of the International Conference on Advanced Mechatronics, Okayama, Japan, 1998, pp. 460–465.

[19] W. Oelen, Modelling as a tool for design of mechatronic systems, Ph.D. Thesis, University of Twente, Enschede, The Netherlands, 1995.

[20] Proceedings of the International Conference on Advanced Mechatronics, ICAM'98, August 1998,.

[21] Saber, http://www.analogy.com; http: //www.analogy.com/products/prod.htm# ControlSystems

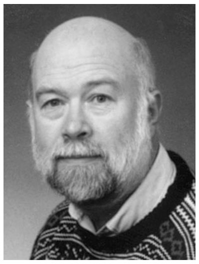
[22] Schemebuilder, http://www.comp.lancs.ac.uk/edc/schemebuilder.html

[23] H.A. Simon, The structure of ill-structured problems, Artificial Intelligence 4 (1973) 181–201.

[24] D.G. Ullman, The Mechanical Design Process, McGraw- Hill, New York, 1992.

[25] T.J.A. de Vries, Conceptual design of controlled electro-mechanical systems, Ph.D. Thesis, University of Twente, 1994.

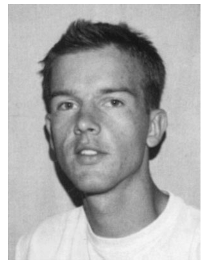
[26] P.B.T. Weustink, T.J.A. de Vries, P.C. Breedveld, Object oriented modeling and simulation of mechatronic systems with 20-Sim 3.0, in: Proceedings of the Sixth UK Mechatronics Forum International Conference, Mechatronics'98, Skovde, Sweden, 1998, pp. 873–878.



Job van Amerongen obtained his Master's degree in Electrical Engineering at Delft University of Technology in 1971 and obtained a Ph.D. degree in 1982. From 1971 to 1973 he did his military service as officer in the Royal Netherlands Navy. From 1973 to 1987 he was Assistant and Associate Professor at the Control Laboratory of the Department of Electrical Engineering of Delft University of Technology, where he worked on applications of modern control theory, especially model reference adaptive control, in ship control systems and electrical power production systems.

Since 1987 he is Professor in Control Engineering in the Faculty of Electrical Engineering at the University of Twente and Head of the Control Laboratory. His current research interests are applications of modern control theory, especially intelligent control, in mechatronic systems. He is Scientific Director of the Cornelis J. Drebbel Institute for Systems Engineering, a co-operation of four departements of the University of Twente. From 1994 to 1998 he was Dean of the Faculty of Electrical Engineering.

He is (co-)author of many papers on adaptive and intelligent control systems, mechatronics and automatic steering of ships, co-author of a book on adaptive control systems and author of three courses on systems and control of the Dutch Open University.



Erik Coelingh was born in Hengelo, the Netherlands, in 1971. He studied at the Faculty of Electrical Engineering of the University of Twente, where he obtained his Master's degree in 1995. His research concerned bond graph modeling of three phase asynchronous machines. Currently he works as a Ph.D. student in mechatronics at the Control Laboratory of the University of Twente. His main interest is the development of design principles and computer-based support for controllers of electromechanical motion systems.

Theo J.A. de Vries is Associate Professor in Intelligent Control and Mechatronics at the Control Laboratory, Faculty of Electrical Engineering, University of Twente, Enschede, Netherlands. His main interest is the development of controlled electro-mechanical systems, in particular using learning controllers. He received a master's degree in Electrical Engineering from the University of Twente in 1990, and a Ph.D. from the same university in 1994 for research on computer support for conceptual design of controlled electro-mechanical systems. He is co-editor of the Dutch Handbook of Electrical Engineering.