Proceedings of the ASME, DSC-Vol. 71
Dynamic Systems and Control Division – 2003
November 15-21, 2003, Washington D.C.

IMECE2003-42436

INTERNET-BASED COLLABORATION AND LEARNING ENVIRONMENT FOR EFFICIENT SIMULATION AND CONTROL DESIGN

Heřman Mann, Michal Ševčenko
Czech Technical University, Computing and Information Centre
Zikova Street 4, CZ-166 35 Prague 6, Czech Republic
Tel. +420-224352933, Fax. +420-224310271, {mann,sevcenko}@vc.cvut.cz

ABSTRACT

A software system DYNAST for efficient modeling and simulation distributed across the Internet is freely accessible at http://virtual.cvut.cz/dyn/. DYNAST supports collaboration of remote engineering teams as well as hands-on training of Web-based-course learners. The DYNAST Server solves nonlinear algebro-differential equations, and automatically formulates them for multipole models characterizing configuration of real dynamic systems. DYNAST Server is also able to linearize the models and to provide their semisymbolic analysis in time- and frequency-domains. Clients can submit their problems and interpret the simulation results across the Internet using different user environments. The results can be animated in 3D by means of VRML. The DYNAST Server supports also publishing standardized reports on simulation experiments. The accompanying Web-based course with a knowledge-sharing system and interactively resolvable examples exploits innovative didactic approaches. In control design, the modeling efficiency of DYNAST can be combined to a great advantage with the control-design power of the MATLAB toolsets.
Keywords: dynamics, control, simulation, Internet, course

INTRODUCTION

It is well recognized that computer-assisted simulation helps engineers to increase competitiveness of their products and gives students a better insight into system dynamics. Nevertheless, only few engineering schools have introduced modeling and simulation as a distinct topic and have given their students an opportunity to deal with real-life problems and practical tasks in this area. Professors often assume that simulation is just a matter of routine utilizing a ready-made software package. As they consider this uninteresting academically, many of them still have had no personal experience in simulation they could share with their students. Industrial engineers competent in simulation rarely publish about it.

Most of the contemporary machines, instruments and other engineering systems are of multidisciplinary nature, as their design requires integration of several traditional engineering disciplines. To simulate such systems, engineers need simulation tools applicable in a unified way to different

- physical domains (mechanical, electrical, magnetic, fluid, or thermal)
- levels of modeling abstraction and idealization
- model descriptions (various forms of equations, block diagrams, or other graphical representations, etc.)

Introducing such a unified approach into engineering study makes the curriculum more compact, and gives students a more comprehensive view of the real world.

When designing complex contemporary systems, engineers usually proceed from the *conceptual level* of the highest abstraction down to the most concrete manufacturing level. During this, they pass through two important intermediate levels: functional and physical. The *functional level* applies typically to the control-design phase when engineers are concerned about input and output signals, states, or disturbances, either analog or digital.

Physical-level models are less abstract than the conceptual- or functional-level models as they portray energy interactions between real system components. The physical-level models are also called virtual prototypes as they are used to replace to a growing extent the time consuming and costly experimenting with real prototypes.

The conceptual- and functional-level models of dynamic systems are often represented conveniently by block diagrams that can be submitted for simulation to a software package like Simulink, for example. Using such a package for physical-level modeling, however, is a cumbersome and error prone task. It requires an involved manual formulation of the underlying equations and, in addition, manual construction of a block diagram representing the equations.

In the past, efficiency of simulation was evaluated with regard to its demand of computer time only. Nowadays, however, the computer time is so inexpensive that the cost of simulation is dominated by the cost of personnel required to prepare the input data, to supervise the computation and to interpret the results. Therefore, an efficient simulation software tool should minimize these manual operations.

Fortunately, modern computers can be exploited not only to solve equations, but also to formulate them for a given configuration of the modeled real system automatically. Surprisingly, however, some academic community members, elated by the beauty of closed-form equations, reject this idea. They claim that students should formulate the equations manually and then inspect them to get a better insight into the system dynamics. They just overlook the fact that this approach is restricted to small 'textbook' models only. Equations underlying real-life system models are usually too opaque for such an inspection. Students should learn, of course, how to formulate the equations at a certain stage of their study. Even then, however, they can find a software tool capable of automated formulation of equations as very useful for self-checking their results.

MULTIPOLE MODELING

Principles

The automatic formulation of equations for physical-level modeling of multidisciplinary systems can be based on the *multipole modeling approach* well established in the electrical domain. Thanks to the similarity of dynamic effects in different physical domains, this approach can be generalized and applied to other physical domains in a unified way. But it is not just the old idea of modeling mechanical, fluid or thermal systems using electrical analogs.

The multipole modeling procedure starts with decomposition of the modeled systems into disjoint *subsystems*. This idea is similar to free body diagrams in mechanics or control surfaces in thermodynamics, for example. A subsystem multipole model approximates the subsystem mutual energy interactions under the assumptions that

- the interactions take place just in a limited number of *interaction sites* formed by adjacent *energy entries* into the subsystems (like fluid inlets, electrical terminals, translating or rotating mechanical connections, heat-transferring contact surfaces, etc.)
- the energy flow through each such entry can be expressed by a product of two complementary *power variables* (like force – velocity, torque – angular velocity, volume flow rate – pressure, electrical current – voltage, magnetic flux rate – magnetic voltage, or entropy flow – temperature)

Each energy entry into a subsystem is represented in its multipole model by a *pole* associated with a pair of the power variables. In graphical symbols of individual multipoles, the poles are denoted by *pins*, i.e. by short line segments sticking out of the symbol outlines.

Dynamics of a complete system can be easily represented graphically by a *multipole diagram* consisting from symbols of subsystem multipole models. The sites of energy interaction between adjacent energy entries are portrayed in the diagram by its *nodes*. The energy entries interacting at the same site are represented in the diagram by interconnecting the related pins to the same node by line segments called *links.* They can be viewed as interconnections capable of transferring energy in both directions without any dissipation, accumulation or delay.

In each physical domain, the most rudimental multipoles represent *pure twopoles* like pure energy sources, accumulators and dissipators. Energy conversion from one domain to another one can be modeled by *pure transducers.* Only four different pure transducers are needed to model all kinds of electro-magnetic, electro-mechanical, magneto-mechanical, fluid-mechanical, rotary-rectilinear, and other energy conversions. Multipole models of real subsystems like electronic or fluid devices, various mechanisms, motors or sensors, heating or cooling units, etc., can be build up from the pure multipoles. The multipoles can be combined also with *blocks* like integrators, summators, etc., which can be viewed as special cases of multipoles.

Figure 1*a* shows an example of a multipole diagram representing the multipole model of a cold-roll mill. Besides pure sources and blocks, the model is set up from symbols of subsystem multipole models shown separately in Figs. 1*b* through 1*c*. The two DC motors driving coilers are controlled for a constant tension in the metal strip. The DC motor driving milling rollers is controlled for a constant velocity. The model of the milling process is strongly nonlinear, and varying deformation of the rollers and their stand is taken into account in it. The model also respects the considerable transportation lag between the rollers and the distant thickness-measuring gages. Thickness of the incoming metal strip is varied randomly during simulation. The gap between the rollers is controlled by an external digital adaptive controller supplied by data coming from several gages in the model.

Advantages

The most important advantage of multipole diagrams over block diagrams or bond graphs is in the *isomorphism* between their structure and the geometric configuration of the modeled real systems. In fact, multipole diagrams represent a mapping of real system representation from the geometric onto the topological space. The practical consequence of the isomorphism is that the multipole diagram can be set up in a kit-like fashion in the same way in which the real system is assembled from its subsystems. Such a modeling procedure can be based on mere inspection of the modeled real system.

Recollect that a block diagram is just a graphical representation of a set of equations. The line segments interconnecting blocks are associated with just one variable that can propagate in one direction only. Though bond graphs represent energy flow between subsystems too, their structure is not at all isomorphic with the real-system geometric configuration.

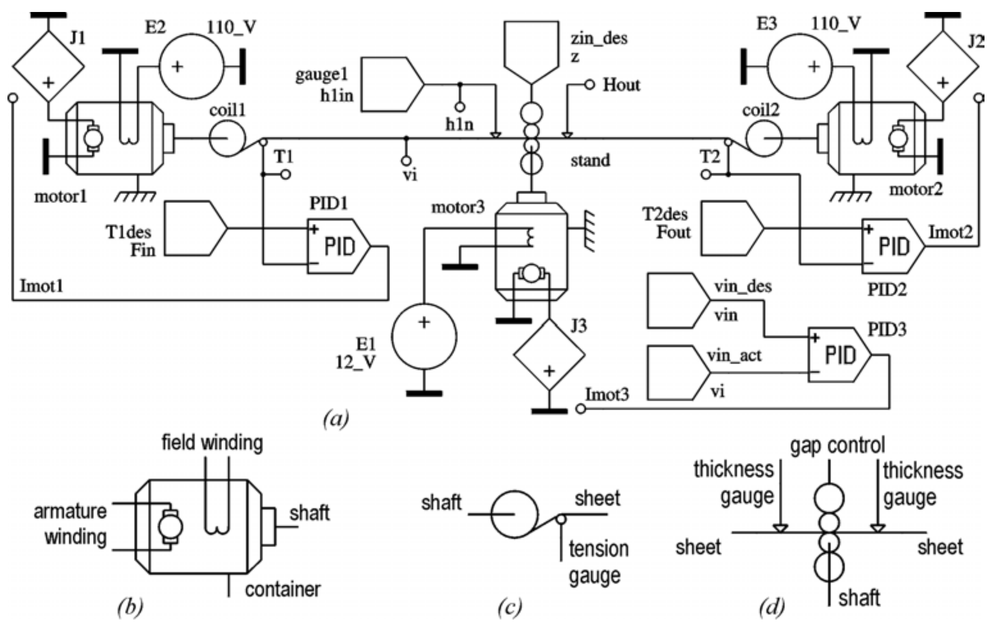


Fig. 1: (a) Roll mill for metal sheets. Multipole models of (b) DC motor, (c) strip coiler, (d) rolling process.

The equations underlying multipole diagrams can be formed by a computer automatically utilizing

- constitutive relations characterizing the individual multipoles
- physical laws governing the balance of energy, mass, electric charge, or magnetic flux in the individual sites of interaction
- laws respecting the geometric connectedness of real systems

Using the multipole approach is also of several other important advantages:

- multipole models can be developed, debugged, tuned up and validated once for ever for the individual subsystems independently of the rest of the system, and once they are formed they can be stored in submodel libraries to be used any time later
- this job can be done for different types of subsystems (e.g., fluid power devices, electronic elements, electrical machines, mechanisms, etc.) by specialists in the field
- inside, the submodels can be represented by different descriptions each of them suiting best to the related engineering discipline or application (lagrangian equations in mechanics, circuit diagrams in fluid power or electronics, block diagrams in control, etc.)
- the submodel refinement or subsystem replacement (e.g., replacement of an electrical motor by a hydraulic one) can be taken into account without interfering with the rest of the system model
- it allows for mixed-level modeling of real systems, i.e. for modeling a system part on the conceptual or functional level by blocks, and at the same time, some other part on the physical level by multipoles

DISTRIBUTED SIMULATION SYSTEM

A system distributed over the Internet has been developed

- to provide an easy access to a powerful simulation engine to engineers even in small enterprises
- to support cooperation of remote teams involved in common research or design projects
- to give to distance or continuous education students a hands-on opportunity to solve real world problems

Simulation Server

The kernel of the distributed simulation system is formed by DYNAST Solver that has been developed for efficient modeling, simulation and analysis of multidisciplinary engineering systems. DYNAST Solver is capable of

- solving implicit sets of nonlinear algebro-differential equations submitted in a natural textual form
- simulating nonlinear systems modeled by multipole and/or block diagrams submitted in a graphical form (the underlying equations are then formed automatically)
- linearizing the diagrams and providing their semisymbolic analysis in time- and frequency-domains

DYNAST is accompanied by libraries of submodels for electronic and fluid-power devices, electromechanical transducers, mechanical parts, control units, etc. The submodel dynamics can be described by a combination of multipoles, blocks, and equations nested in a hierarchical way. The libraries are open for easy addition of user-defined submodels and their symbols. Each DYNAST submodel description is encapsulated in an independent file. The default values of submodel parameters can be overridden by values specified in terms of constants or symbolic expressions.

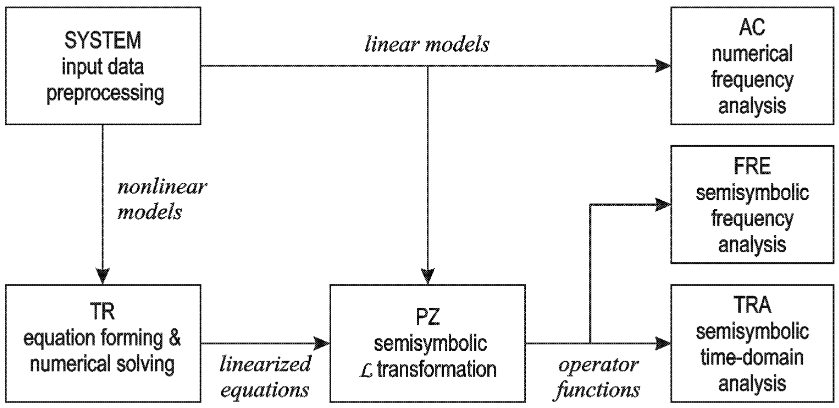


Fig. 2: Data flow in DYNAST Solver

DYNAST Solver is divided into sections shown in Fig. 2. The section SYSTEM reads in input files with problem descriptions in a textual form representing equations and/or diagram netlists. In the latter case, this section formulates equations underlying the diagrams. The input data is directly interpreted, so there is no compilation delay.

Nonlinear systems of equations are solved in the TR section. Besides *transient responses*, this section computes also *system steady states* after converting the differential equations into algebraic ones. The algebraic equations can be solved also for a parameter swept through an interval. The transient responses can start either from initial conditions specified by the user, or from those corresponding to the system steady state. Even if the user-specified initial conditions are inconsistent, DYNAST is able to find the nearest consistent initial conditions just in few iterations. The fast Fourier transformation is avai-lable in this section for *frequency-spectrum analysis* of steady-state periodic solutions of nonlinear systems.

The DYNAST nonlinear *equation-solving routine* is based on a stiff-stable implicit multistep backward-differentiation integration formula. The length of integration steps and, at the same time, the order of the formula is continuously optimized continuously to minimize the computation time while respecting the admissible computational error. Jacobians of the equations are evaluated using a symbolic differentiation procedure. Consi-derable savings of computational time and memory are also achieved by exploiting the jacobian sparsity. The integration provides *automatic lineralization* of the analyzed nonlinear equations. Thus, the nonlinear system models can be subjected to *small-excitation analysis* within the vicinity of a user-specified or computed quiescent operating point.

Thanks to the implicit form of the equations formulated by DYNAST, and thanks to their simultaneous solving, DYNAST imposes no artificial restrictions on the simulated system structure. There is no need for equation sorting to solve the cau-sality, algebraic loop and other problems associated with software exploiting the explicit form of equations.

Operator functions representing transfer functions and transforms of initial-state responses for lineralized system models can be computed by the PZ section. The operator functions are provided in a semisymbolic form with the Laplace operator *s* as a symbol and polynomial roots or coefficients as numbers. For such operator functions, DYNAST can compute semisymbolic-form time-domain characteristics in the TRA section. The FRE section evaluates the corresponding frequency characteristics numerically. Linear frequency-dependent models (like those with parameter-distributed submodels) DYNAST analyses in the AC section.

DYNAST Solver runs under MS Windows and Unix. It is distributed in different single-user as well as in server versions suitable for engineering teams and computer classrooms. Its student version is available free of charge for downloading [3].

Working Environments

The server-based DYNAST Solver can be accessed across the Internet in a Web-based, on-line and e-mail modes [1]. Setting up the multipole and block diagrams directly on a Web page is enabled by the schematic editor DYNCAD formed by a Java applet. DYNCAD converts diagrams into the DYNAST input language and sends the data to the DYNAST Solver across the Internet. After the computational results are sent back, they can be plotted on the client-computer screen.

Even more comfortable and user-friendly mode of access to DYNAST Solver provides DYNAST Shell as shown in Fig. 3. This mode requires, however, downloading and installing this software on client computers with MS Windows. DYNAST Shell has been designed for a wide variety of tasks in a way suitable to users of different levels of qualification and experience. All operations are supported by a context-sensitive help system. A built-in syntax analyzer is continuously checking the submitted data. DYNAST input language is object oriented, yet it is very natural. Dialog windows (wizards) in DYNAST Shell allow, however, for submitting data without any knowledge of the input language.

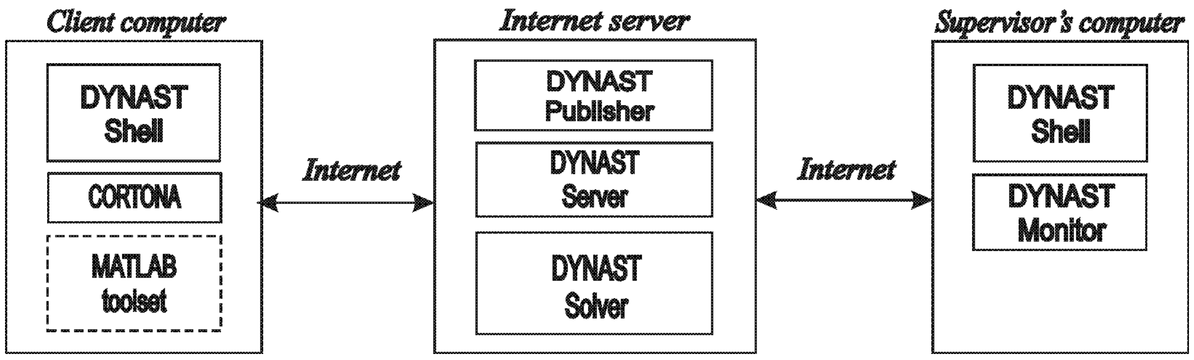


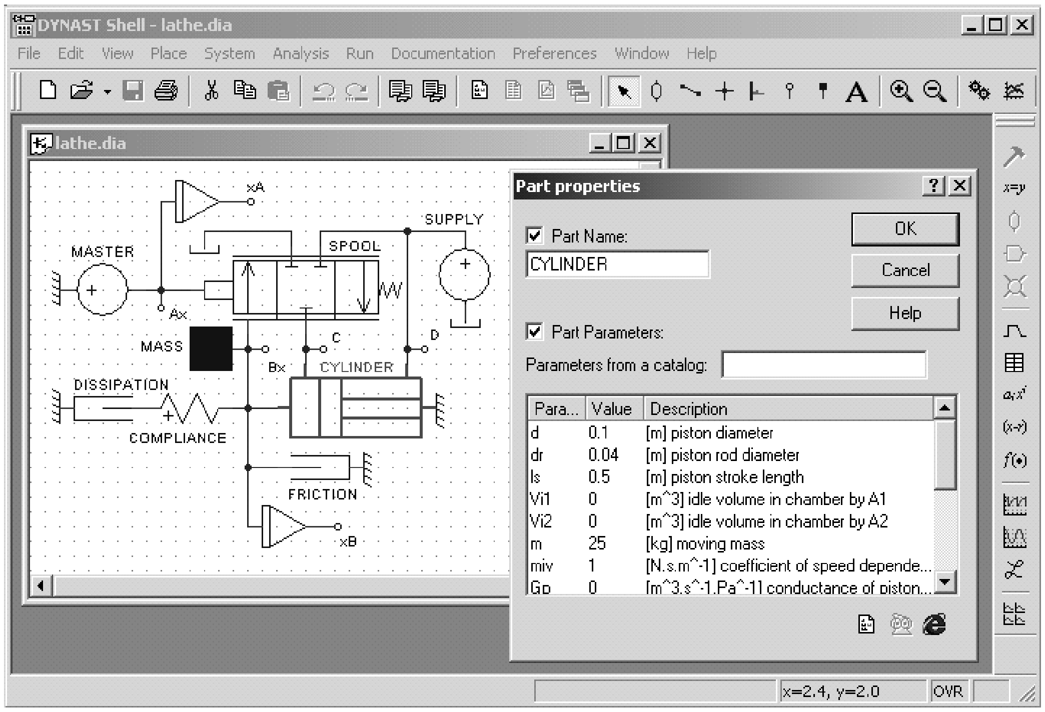
Fig.3: Environment for modeling, simulation and virtual experiments.

DYNAST Shell includes graphical editors for multipole and block diagrams as well as for submodel symbols. A special dialog box for each new submodel is formed automatically as shown in the screenshot of the DYNAST Shell graphical interface in Fig. 4.

DYNAST Shell can also communicate with the server-based DYNAST Publisher. It is a documentation system for automated publishing of reports on simulation experiments and descriptions of library submodels using LaTeX. The systems extracts automatically the relevant parts of the input data and captures the submitted multipole or block diagrams as well as the resulting output plots and includes them into the documents. The documents can be converted by the server into PostScript, PDF and HTML formats.

Simulation results can be used for animation of 3D-space models formed using VRML- and Java-based visualization tools developed for the purpose. The motion scripts for the virtual experiments are generated by DYNAST Solver. The users need to download and install only the free CORTONA software on their PCs. Figure. 6a shows an example of a robot movable 3D model.

Another very useful tool communicating with the server is the software package called DYNAST Monitor shown in Fig. 3. It allows design managers or tutors to observe from any site on the Internet the data files and diagrams the users are submitting to DYNAST Solver from their client computers. The supervisor can communicate with the users across the Internet and assist them in solving their problems.



☑ Part Name: CYLINDER

☑ Part Parameters:

Parameters from a catalog:

Para...	Value	Description
d	0.1	[m] piston diameter
dr	0.04	[m] piston rod diameter
ls	0.5	[m] piston stroke length
Vi1	0	[m^3] idle volume in chamber by A1
Vi2	0	[m^3] idle volume in chamber by A2
m	25	[kg] moving mass
miv	1	[N.s.m^-1] coefficient of speed depende...
Gp	0	[m^3.s^-1.Pa^-1] conductance of piston...

Fig. 4: Parameters for the CYLINDER submodel of a copying lathe.

MODELING TOOLBOX FOR MATLAB

In control design, the modeling efficiency of DYNAST can be combined to a great advantage with the control-design power of the MATLAB toolsets. Using either DYNCAD or DYNAST Shell, a model of the plant to be controlled can be easily set up in a graphical form and then used to validate the open-loop model. At the same time, DYNAST is able to compute the required plant transfer-function poles and zeros and to export them to MATLAB in an M-file. Communication between the server-based DYNAST and MATLAB installed on a user's computer across the Internet is available at [1].

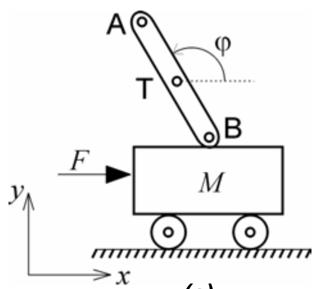
Example of Analog Control

Let us consider analog-PID control design for the plant in the form of the inverted pendulum given in Fig 5*a*. As shown at [2], a considerable number of manual operations are necessary before MATLAB can be exploited for the plant simulation. DYNAST allows for avoiding all these tedious manual operations. Fig. 5*b* shows a multipole model of the pendulum, which has been set up using the Web-based schematic editor DYNCAD. The motion of the cart, identified with the *x*-motion of point B, is represented by node B_x in the multipole model. Using the DYNCAD menu for the translational mechanical domain, dynamics of the cart is represented by symbols for the cart-mass inertia and for cart damping due to the cart wheel friction. The external force *F* acting on the cart is represented by the source of force symbol.

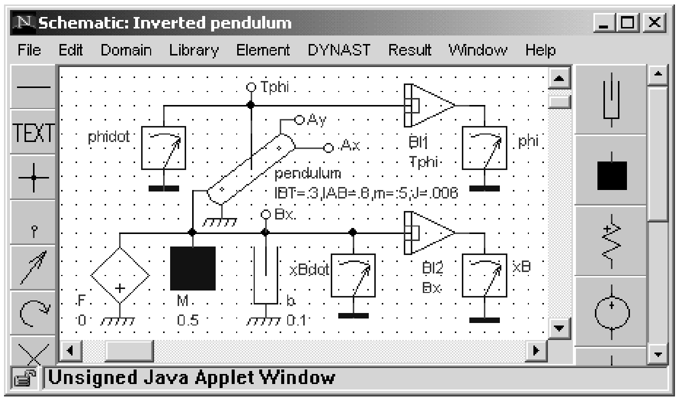
Dynamics of the pendulum is modeled by the submodel for planar motion of a body with the rotational joint B. The *x*- and *y*-motions of the pendulum end-point A are represented by nodes A_X and A_Y in the multipole model, while the rotational motion of the pendulum represents node A_φ. Poles representing a common motion, either rectilinear or angular, are interconnected by a line segment. Integrating blocks are added to compute positions from node velocities, and the symbols of ideal measuring instruments indicate which variables should be exported from DYNAST to MATLAB in an M-file. The plant PID control design by means of MATLAB can proceed as described in [2]. Then the resulting feedback loop is added to the plant model in DYNCAD as shown in Fig. 5*c*. Finally, the complete nonlinear control system is verified in DYNCAD

Example of Digital Control

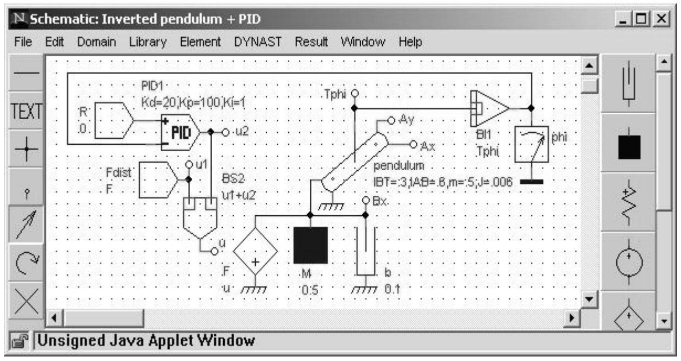
Also in the case of a digital-PID control design the M-file with transfer-function data of the plant model is first exported from DYNAST to MATLAB. To verify the design in this case, the digital feedback loop is implemented in SIMULINK as shown in Fig. 5*d*. The large square block represents there the plant multipole model in DYNAST shown already in Fig. 6*b*. The communication across the Internet between this model remaining in DYNAST and the rest of the diagram implemented in SIMULINK is enabled by the SIMULINK S-function available for downloading at [1].



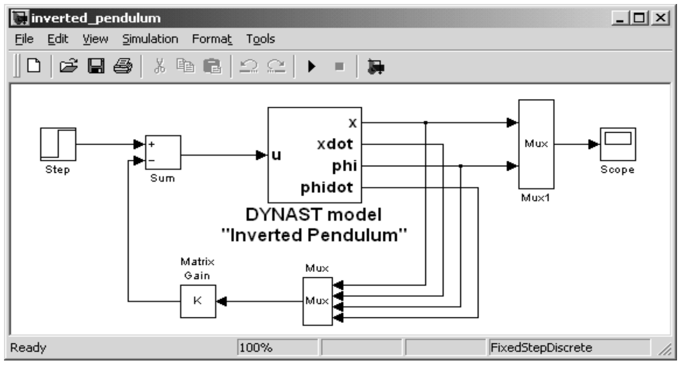
(a)


Schematic: Inverted pendulum
File Edit Domain Library Element DYNAST Result Window Help
TEXT
Tphi
Ay
Ax
phidot
BI1
Tphi
phi
pendulum
IBT=.3,IAB=.8,m=.5,J=.006
Bx
xBdot
BI2
Bx
xB
F
0
M
0.5
b
0.1
Unsigned Java Applet Window

(b)


Schematic: Inverted pendulum + PID
File Edit Domain Library Element DYNAST Result Window Help
TEXT
PID1
Kd=20,Kp=100,Ki=1
R
0
PID
u2
Tphi
Ay
Ax
BI1
Tphi
phi
Fdist
F
u1
BS2
u1+u2
u
pendulum
IBT=.3,IAB=.8,m=.5,J=.006
Bx
F
u
M
0.5
b
0.1
Unsigned Java Applet Window

(c)



(d)

Fig. 5: (a) Inverted pendulum, (b) multipole model, (c) analog PID control, (d) digital PID control.

WEB-BASED DYNAMICS AND CONTROL COURSE

A project called DynLAB is in development by an international consortium [3]. The project aims at motivating young people to engineering study, and at improving engineering training using innovative didactic approaches and tools.

The emphasis and style of the course differs from most of the existing courses by

- exposing learners to a novel systematic and efficient methodology for realistic modeling of multidisciplinary system dynamics applicable to electrical, magnetic, thermal, fluid, acoustic and mechanical dynamic effects in a unified way
- introducing learners to the methodology through simple, yet practical, examples to stimulate their interest in engineering before exposing them to rigor math
- giving learners a better 'feel' for the topic by problem graphical visualizations and interactive virtual experiments
- allowing different target groups to select individual paths through the course tailor-made to their actual needs and respecting their background
- allowing both for self-study and remote tutoring with investigative and collaborative modes of learning
- integrating computers into the course curriculum consistently and giving learners a hands-on opportunity to acquire the necessary skills
- exploiting the computers not only for equation solving, but also for their formulation minimizing thus learners' distraction from their study objectives
- giving learners the opportunity to benefit from 'organisational learning', i.e. from utilizing knowledge recorded during previous problem solving both in academia and industry

The dynlab course is delivered within a web-based learning environment supporting learners' mutual collaboration and their communication with a tutor. Investigative learning is encouraged by open problems and virtual experiments.

Self-study is supported by interlacing the course with self-assessed quizzes, tests and other motivation elements. For the remote tutoring mode, there is a collection of tutor-marked assignments in each module. The course is accompanied by a large collection of examples of various problems solved both in academia and industry to imitate knowledge sharing and informal learning typical for large organizations. It is supported by a computer-assisted ontology-based process in which knowledge gained during solution of problems is captured, recorded and later made available to learners 'just in time' when it is relevant to the problems they are supposed to solve. The examples can be resolved and modified in an interactive way across the Internet. This gives the learners a hands-on opportunity to acquire the necessary skills in solving real-life problems.

The course flexibility is achieved by its modular arrangement with a number of different entry points. In each module, the area and level of prerequisite knowledge as well as the degree of motivation required for its study is clearly specified. Novices are motivated to a more involved engineering investigation of system dynamic behavior by 'playing' with movable 3D virtual-reality models of various systems. They can change the model parameters and excitation while observing the model behavior not only qualitatively, but also quantitatively using virtual measuring instruments. As an example of such a virtual system, Fig. 6a shows the movable 3D geometric model of a robot.

In the next step, students can model, simulate and analyze dynamic behavior of given systems. A multipole dynamic model of the robot and an example of its simulation result is in Fig. 6*b* and 6*c*, respectively. For example, sufficiently motivated students with a mechanical engineering background, but of an introductory level only, can start from the module on multipole modeling of linear one-dimensional mechanical systems. From there, they can move to nonlinear mechanical systems or to linear electrical and then electro-mechanical systems, for example. More advanced students can enter directly the module on linear multidisciplinary systems, etc.

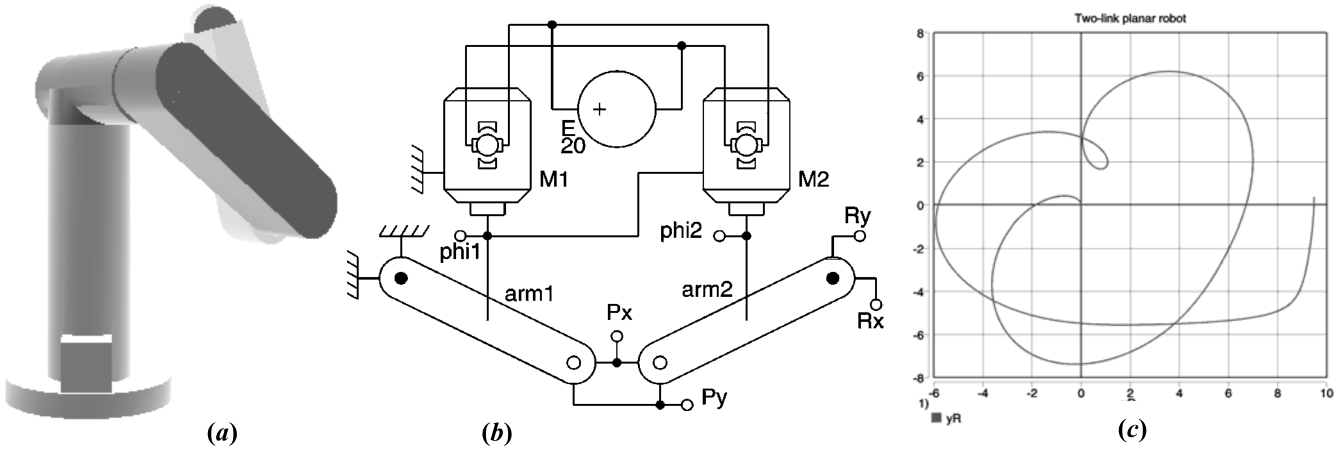


Fig. 6: Robot: (a) 3D model, (b) multipole diagram, (c) plot of a robot-arm trajectory.

Table 1: Learning modes in the DynLAB course.

Learning objective	Prerequisites	Course assignment	
		Given	Task
stirring up interest in dynamics	high-school math and physics	3D virtual model of a real system	to modify system parameters or excitation and to observe changes in the system dynamic behavior
introduction to dynamic modeling	high-school math and physics	configuration of a real system	to set up the corresponding multipole diagram and to simulate its dynamic behavior
more advanced dynamic modeling	fundamentals of system dynamics	configuration of a real system	to set up the multipole diagram from user-made submodels and to analyze the real system dynamic behavior
formulation of system equations	introduction to dynamic modeling	configuration of a real system	to form system equations, to solve them, to set up the multipole diagram, and to compare the solution with the simulation results
introduction to control design	formulation of system equations	plant specification & control objectives	to reduce the model, to design control, and to verify the design using the plant unreduced model
introduction to system design	introduction to control design	system specification	to design the plant as well as its control, then to verify and optimize the overall design
design of virtual experiments	advanced dynamic modeling	experiment specification	to design the virtual 3D geometric model, to set up the dynamic model, and to write the simulation script

The most advanced students are supposed to design controlled dynamic systems, to verify the design and to optimize it. A short survey of different learning modes supported in DynLAB environment is given in Table 1.

Teachers interested in utilizing some of the course modules, usually as a complement to their face-to-face courses, assign their students the module open problems and often add some problems of their own. If the modules are chosen properly, most of the students are able to follow the modules and to solve the problems quite independently. For them, this activity usually becomes a kind of a computer game. At the very start, however, tutoring speeds up overcoming the module initial learning barrier of the modules though it is fairly low. This is where DYNAST Monitor takes over an indispensable role as it shows exactly what the students are doing and allows the tutor to assist them in the most suitable way.

CONCLUSIONS

Capabilities of the simulation system DYNAST [4], available now in a distributed form, have been well proven by many of its academic and industrial users already. Automated analysis of the monthly access to the DynLAB server [3] indicates a considerable interest in utilizing the online system by people from all over the world each month. Many of them also download the free DYNAST Student version that can be used off-line. The Professional and Server versions are available on a commercial basis for very affordable prices.

ACKNOWLEDGMENTS

The authors would like to gratefully acknowledge the support for this project by the Leonardo da Vinci and Socrates Minerva Programmes of the European Union.

REFERENCES

[1] Website of the *Virtual Action Group for Multidisciplinary System Simulation*, a part of the Technical Committee on Computer Aided Control System Design of the IEEE Control Systems Society at http://virtual.cvut.cz/cacsd/dyn/

[2] Messner, B. and D. Tilbury, “Control Tutorials for MATLAB”, University of Michigan/ Prentice Hall 2000, http://www.engin.umich.edu/group/ctm/

[3] Website of DynLAB, Pilot Project of the EU Leonardo da Vinci Vocational Training Programme, at http://virtual.cvut.cz/dynlab/

[4] Mann, H., “Support for efficient modeling of multidisciplinary systems”, Proc. of the ASME, DSC-Vol. 67, Nashville 1999.