Mathematics in Virtual Knowledge Spaces: User Adaptation by Intelligent Assistants

Sabina Jeschke, Thomas Richter
Berlin University of Technology

The workplace of scientists and engineers is about to change: even though numerical software and computer algebra systems remove the burden of routine calculation, it becomes more important than ever to get familiar with new concepts and methods quickly. Given the rapid growth of knowledge in today's sciences, traditional "learning on supply" (i.e. defining the learning goal as the accumulation of knowledge) is no longer applicable; instead, adequate learning and teaching methods have to be established to guide learners towards efficient self-controlled learning.

Traditional methods of teaching can only satisfy this demand partially: lectures provide elementary base knowledge, but do not encourage active independent deliberation in the matter. Classical hands-on training in laboratories on the other hand requires additional human resources and is often constrained by the financial possibilities of the involved institutions.

We do not aim to replace the mentioned classical forms of teaching; rather, we want to show how the application of New Media and New Technology presents a turning point in the educational system by providing tools that close the gap between passive document retrieval systems on the one hand and practical courses in laboratories on the other hand.

The value added by the New Technology is the ability to enrich traditional methods of education – teacher centred teaching, literature research, homework training, and laboratory experiments – by some limited form of "intelligence" and by suitable interfaces to allow closer integration of these areas to improve the learning process. Thus, tools are proposed that are not only able to adapt to

the learning process of the student, but are also smart enough to point towards additional background information and thus actively support the learner beyond what has been possible before.

Four areas of the application of New Media are presented: the presentation of mathematical content, intelligent lexicon toolkits that are able to learn from natural language texts, homework training courses that are able to break up assignments into elementary subproblems as needed by the learner, and Virtual Laboratories that are able to provide courses that adapt to the errors of the learner, but are still rich enough to be used in research problems.

Keywords: Computer-Based Training, Course Web Site, Web-Based Courses, Web-Based Teaching, Web-Delivered Education, Virtual Campus, Collaborative Learning, Electronic Learning (E-Learning), Undergraduate Education, Client-Server, Distributed Systems, Media Server, Multimedia, Knowledge Discovery, Semantic matching, Web-Based Applications, Natural Language Processors, Cybernetics, Human-Machine Systems, CORBA, Multimedia

INTRODUCTION: WHY INTELLIGENT ASSISTANTS?

Mathematics is the key technology of the 21st century: besides being a research field of its own, it is the key ingredient for studies in engineering sciences, physics, computer science and many other fields. Teaching mathematics therefore means teaching a very broad, heterogeneous audience with varying fields of interest; teaching at the Berlin University of Technology in particular means having to handle increasing student numbers with decreasing funding. Luckily, mathematics is a highly structured field using a very precise, formalised language. Its internal structure is built on well-developed entities, e.g. fields, vector spaces, linear mappings, all integrated into a well-accepted ontology. Given that more and more of the computational tasks are solved by the computer today, the demand for *understanding the concepts* and *interpreting the results* of the electronically performed operations becomes a major task of the mathematical education. Therefore, the structure and workings of mathematics has to be understood by students – and can thus be exploited to aid the design of electronic tools supporting the learning process.

To achieve this however, we must *go beyond* the first generation of eLearning [Jeschke and Keil-Slawik, 2004] which was rarely more than computer assisted document management. We therefore need technology that provides enough flexibility to adapt to the requirements of the field and the learning process of the student.

If our goal is contributing to multiple learning scenarios *and* providing a scientific broadness at the same time, the complexity of the user interfaces is likely to rise. This is of course undesirable for our purpose; therefore eLearning environments have to evolve from complex toolkits to systems that contain a certain amount of autonomy, enough to support human operation processes by a degree of artificial intelligence, since students should *learn the subject matter, not how to use the software*.

In short: We need *intelligent assistants*. Therefore the concepts which have been recently developed in the field of Artificial Intelligence have to be adapted to Virtual Knowledge Spaces and their components. The impact of intelligent assistants reaches from adaptivity to specific usage patterns up to actively supporting the learning or research process itself by applying the analytic capabilities of computer systems to perform mathematics.

BACKGROUND: ELEARNING AT THE BERLIN UNIVERSITY OF TECHNOLOGY

The Berlin University of Technology is one of the three universities in Berlin, focused mainly on the fields of engineering and applied sciences; it also offers studies in mathematics, physics and even studies in humanities. With over 30,000 enrolled students, it is one of the major universities of Germany. Undergraduate courses in mathematics for engineers typically have around 1,500 to 2,000 participants, taught by up to five lecturers in parallel. These courses are organised and maintained by the Institute of Mathematics and Natural Sciences.

To improve the education for this audience, the institute has established the Multimedia Centre for eLearning, eTeaching & eResearch, or short MuLF [MuLF], which is within the tradition of many earlier third-party funded research projects from the area of electronic learning, e.g. MUMIE [Mumie], MOSES [Moses], NEMESIS [Nemesis], BELEARNING [BeLearning], and the Virtual Laboratories Project located at the DFG research Centre MATHEON [Matheon]. All these projects focus on the mathematics education of engineers, computer scientists, physicists and mathematicians; the teaching level here ranges from undergraduate mathematics up to graduate studies in mathematics and physics.

The research foci of the eLearning group of the faculty range from pedagogical aspects in electronic learning down to technical aspects that need to be solved to implement these goals. The pedagogical aspects cover the development of concepts for intelligent training environments, the design of explorative learning environments and models for user adaptivity. Concepts for distributed teaching and learning in the spirit of eBologna [EU] are also researched.

All mentioned aspects require background knowledge from a broad variety of other disciplines that are also under research; we deal with problems of semantic encoding, analysis of mathematical language and specific matter ontology, with the design of semantic retrieval systems and aspects of automatic validation of mathematical solutions, down to some computer science problems like constructing integration technologies for open eLearning environments and the design of distributed systems for cooperative learning.

CLASSIFICATION OF ELEARNING

In the following, a classification of eLearning support into four categories is presented, always keeping in mind that these have to be understood as intertwined and entangled with each other (cf. Fig. 1): the *content area*, presenting matter in a structured way as it is typically taught in lectures and courses. The *training area* provides a framework for homework assignments and hands-on training, the *semantic retrieval area* parses natural language texts into a knowledge-network and answers individual requests on relations within this network, forming an intelligent encyclopaedia. Last but not least the *Virtual Lab area* allows self-controlled learning by providing the infrastructure for experiments and hands-on training.

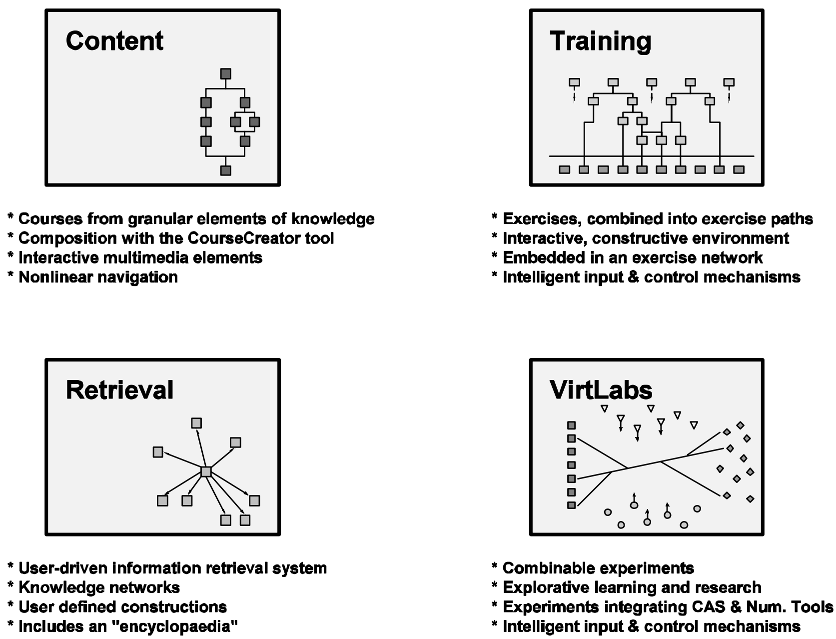


Figure 1: Categories of eLearning and their basic properties.

All four areas are going to be introduced in detail, describing their state of implementation and giving prospects for intelligent assistants within each area.

eLearning Equipment at the Berlin University of Technology

The eLearning initiative of the Berlin University of Technology started with the BMBF (German Ministry for Education and Research)-funded project MUMIE [Mumie; Dahlmann, Jeschke, Seiler, Sinha, 2003] which aims at enriching classical undergraduate math lectures with multimedial content. MUMIE has been designed from ground up to support students in more than one way in their studies: it should not only present the material taught in lectures and make it available for pre- or post-learning by allowing non-linear navigation, it would also contain the exercises and homework assignments for independent study. Thus, the design of the “content area” and the “training area” as two approaches to support learners within one common eLearning framework existed right from the start. The content area currently contains material for an undergraduate course on linear algebra for engineers. It includes references to exercises in the training area that allow students to intensify their knowledge on the topics which were introduced and presented in the course. Vice versa, the homework assignments stated in the training area refer to the corresponding topics of the course. Details are given in the ”Content Area” and ”Training Area” sections.

Not much later, the idea was born to enlarge this concept by an encyclopaedia making the material more accessible. By extending this idea, the conception of a knowledge base [Beierle and Kern-Isberner, 2000; Haun, 2002; Görz et al., 2000] for mathematics was considered, plus the requirement to fill this knowledge base with existing material as found in textbooks. Due to the lack of resources, we had the hope to automatise this process to a good degree. This concept evolved into the “retrieval area” introduced above, and induced the dissertation of one member of our research group [Natho, 2005]. Even though there is no retrieval front-end for its knowledge data base yet, the current codebase is already able to extract the linguistic structure from natural language texts, i.e. syntactical relations between words and their function within a sentence. It has been found [Natho, 2005] that this structure often allows programs to extract mathematical relations between terms, which is mainly due to the very formalised way in which language is used in mathematical context. This project will become part of the Mumie framework soon, allowing students not only to look up words as in a dictionary, but also allowing them to find relations between topics and chapters of a course, see the “Retrieval Area” section. This integration work is currently under development. We plan to integrate this project into the Mumie framework soon.

The “Virtual Lab” area is the youngest member of the Mumie family. Even though the idea of using cellular automata – see the ”Virtual Lab” and ”VideoEasel” sections – to implement models of statistical mechanics is not new [Toffoli and Margolus, 1987]. The author (Th. Richter) has been playing with the idea to exploit this for educational purposes for a while until he had been given the chance to implement the software as project G4 of the MATHEON Research Centre at the Berlin University of Technology. The resulting virtual laboratory VIDEOEASEL, which will be described in its own section in more detail, has now been already deployed for the course Mathematical Physics II - Statistical Mechanics - at the University of Technology, and at the Heinrich Hertz School in Berlin. Although being the youngest component of MUMIE, it is the first project that has been equipped with assistant technology, to be described in ”Intelligent Assistants” section.

Efforts to enrich the training and the content area are currently running, and our concepts for doing so are described in the following sections in more detail.

CONTENT AREA

The content area is the electronic representation of the content of a specific course or lecture. In order to aid the lecturer and the student, the topic of a course is separated into minimal knowledge atoms following a specific matter ontology. A knowledge atom is the smallest presentable unit in a mathematics course, that is: a specific definition of a mathematical object, a theorem describing the properties of an object, a single motivation for a definition, an example out of several examples for a mathematical statement, or an algorithm. The content area may contain several knowledge atoms targeting at presenting the same mathematical property, i.e. several versions of a proof or a theorem might, or even should, exist. The ontology on which this classification is based should not be confused with the ontology of mathematical objects (i.e. sets, tuples, maps, and so forth) but is rather the ontology used to *present* and *organise* mathematics in books and lectures.

These atoms are enriched by interactive applets, cf. Fig. 2, and composed to courses. This concept on the one hand allows efficient re-use of existing atoms and their recomposition to new courses, thus simplify the task of the lecturer, while on the other hand it also allows non-linear navigation for students to either follow courses or refresh their knowledge in exam preparation. It is here again important to note that authors have to be encouraged to provide several equivalent, but different presentations of the underlying mathematical ideas to gain the flexibility required to adapt courses to various audiences.

The course content is presented as a network, shown on the left-hand side of Fig. 2, allowing students to browse freely within the matter while showing the relations and dependencies between the atoms. The atoms themselves are required to be context-free or almost context-free to be read and recombined independently. The ”red thread” (illustrated by a broad line in Fig. 2) indicates the recommended course order. It represents one possible lecture through the material; alternative routes are available simply by selecting the desired atoms.

In its current form, created to a major extend by S. Jeschke, R. Seiler and co-workers of the MUMIE-project [Mumie], it consists of a database providing the knowledge atoms, an application server delivering the contents as HTML data, and a course creator tool that defines the linkage of the atoms – as vertices in a graph – to a complete topic, see Fig. 2.

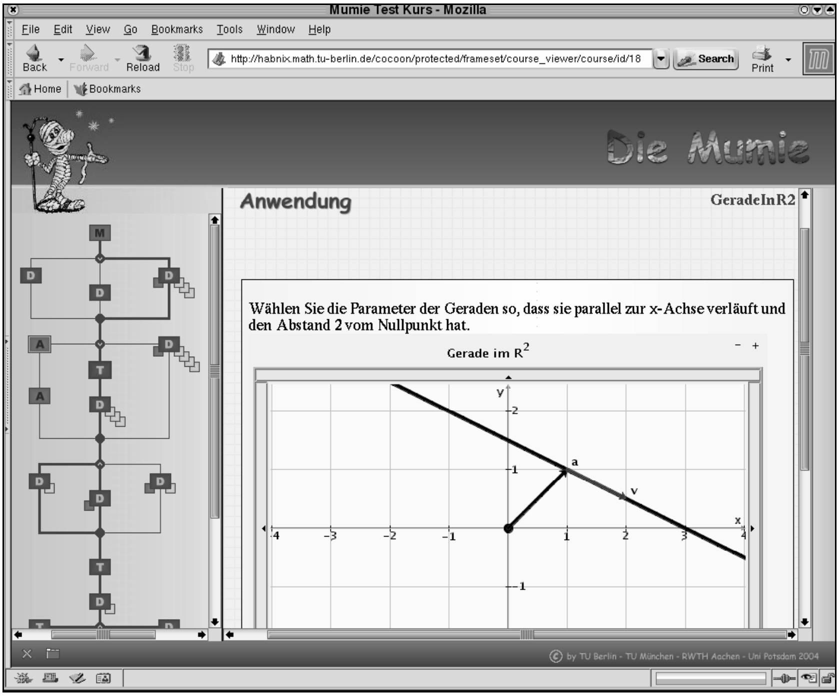


Figure 2: A course in the MUMIE-platform, showing a sample application of linear algebra; in the left frame the course and the corresponding knowledge atoms of this applet are displayed. The "red thread" indicates the recommended course schedule. In the main frame an exercise is shown, which prompts the learner to construct a line parallel to the x-axis in a distance 2 from the origin.

Clearly, this design allows reusing existing material easily to adapt courses to changing requirements. The prospects for intelligent assistants in the content area are plentiful: they would observe the usage patterns of both the lecturer creating and the student using a course and thus would allow adapting the presented material to the corresponding user. Looking at the vertices of the course-graph, an assistant would be able to select the proper presentation style of the content that fits the learner's style best, e.g. textual representation vs. visualisation (see also the "Intelligent Assistants" section for a discussion of user types). In Fig. 2, an assistant would be able to select an alternative route through the material, not following the suggested red thread if the usage patterns of a student indicate that this would be more beneficial for presenting the material.

But assistants could also help lecturers to combine the atoms to courses given the demands of the audience and the author at hand, and thus act on the edges of the course-graph as well; the tool developed for this process is the so-called *Course Creator*. Even though the Course Creator is currently just a graphical composition toolkit without own intelligence, given the dependencies between the atoms and the preferences and demands of the lecturer and the audience an assistant would be able to propose a full course.

SEMANTIC RETRIEVAL AREA

While the content area requires that the content provided to students and teachers is already integrated into a network of knowledge atoms, the main focus of the semantic retrieval area is to construct these networks from mathematical texts formulated in natural language. The mentioned texts could be taken from a mathematical textbook or lecture notes; while we do not believe that it is possible to entirely extract the contents automatically from these raw materials, we are confident that a semi-automatic approach requiring manual interaction gains enough information from the source to make the usage of the tool worthwhile [Natho, 2005].

The resulting knowledge networks represent connections between terms as well as dependencies between mathematical statements. Even though this front-end is not implemented yet, the resulting network could be visualised by a retrieval system, making the connections between terms and statements accessible and attractive to the student – providing a valuable tool for exam preparation or homework assignments. Alternatively, the network would prove useful for the authors of lectures as well as it helps to build up an electronic lecture from printed ”traditional” material.

In its current form, mainly developed by N. Natho as part of the MUMIE-environment [Natho, 2005; Mumie], the software ”mArachna” implements a semi-automatic natural language parser that analyses mathematical texts for their linguistic structure; that is, starting from the syntactical linkage of words into sentences, mArachna builds up graphs of objects and relations between these objects.

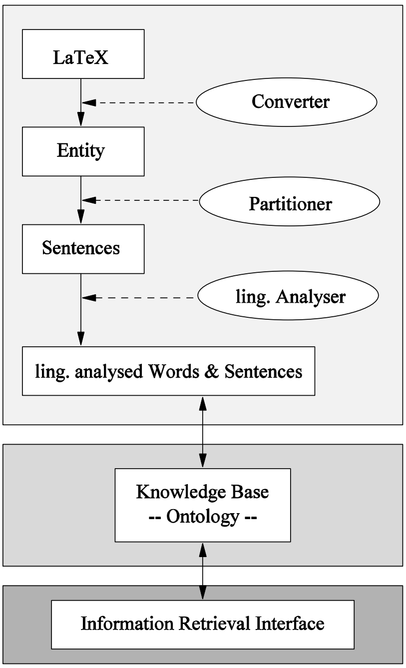


Figure 3: Architecture of the semantic retrieval tool, showing the information flow through its modules, from top to bottom.

Fig. 3 shows the architecture of the tool: a *converter* analyses the mathematical content given in LATEX form, identifying theorems, definitions, and other mathematical entities and separating them from non-parsable contents, e.g. images, applets, graphs and similar non-textual sources of information (Rather: hard to parse content. While technology might be in development to allow even semantic analysis of images or applet code [Dahlmann, Jeschke, Seiler, Wilke, 2005], those attempts are not considered here; the emphasis of this project aims at parsing of natural language texts.). Since LATEX requires or at least encourages authors to already tag mathematical entities, this parsing process is greatly simplified.

The *partitioner* breaks up these entities to identify their internal structure by means of a simple pattern matching algorithm that already requires a basic morphological analysis of the sentences.

A full *linguistic analysis* of the sentences based on Chomsky's transformational grammar is applied in the next step, segmenting sentences into phrases and identifying their syntactic function within them.

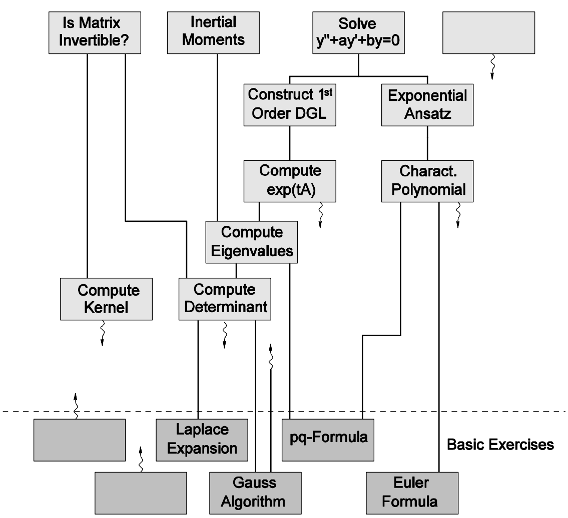
The *semantic analysis* detects relations between these phrases, groups them into objects and builds triples of object pairs and relations, thus assigning a semantic to the phrases. These object triples are used to build up semantic networks within the *knowledge base*, allowing an *information retrieval* front-end to gain access to it. A knowledge base is a data base system that contains graphs whose nodes are terms, in our case mathematical entities like "group" or "determinant", and whose edges represent relations between these terms, e.g. whether one term is a specialisation of another or contains another, etc.

Semantic analysis of natural language is an active research field and as such many problems are still open. Restriction to mathematical texts simplifies this task to a major degree: the texts are to be supplied in LaTeX, already providing a very simple form of semantic annotation. Additionally, mathematical language is highly formalised and uses well-defined phrases, simplifying the process of identifying the semantics by identifying these phrases.

Even though this system already qualifies as an intelligent assistant by itself, especially its retrieval component could be enhanced by making use of assistants that try to find a representation of the contents suitable for a given user profile, and thus to adapt the answers of the system to its user. Clearly, another application of the semantic retrieval system would be the initial step of reverse-fitting an existing lecture in paper form into the content area, thus aiding the lecturer and the student. A third, inner-mathematical application of the semantic analyser might be to identify similarities within the internal structure, i.e. requirements and conclusions of theorems to build-up cross-relations between ad-hoc unrelated research fields, thus supporting mathematical research to gain insight into the meta-structure of mathematics. For example, given the definition of the gcd in elementary number theory, an assistant might point out that its requirements, namely a ring with division algorithm and without zero divisors, apply also to polynomial rings, and thus a gcd can be defined there as well.

TRAINING AREA

The training area provides students with highly structured exercises to delve into the subject of a course to a higher degree than by just following the lecture. The keywords here as in all other fields discussed so far are *granularity* & *structure* of the exercises. Thus, a given assignment is structured into sub-problems to be solved by the student so that a corresponding easier exercise can be given to focus on trouble points in case the student gets stuck. In other words, the training area is designed around a hierarchically structured graph of exercises providing individual learning units.



This type of adjustment enables learners to gradually enhance their competencies in self-directed problem solving.

Figure 4: Training network of exercises, here an excerpt from a course on linear algebra, showing graph-like dependencies. Basic exercises in the bottom row.

Fig. 4 shows an excerpt of a training network containing exercises of a course on linear algebra. For example, to solve a linear differential equation with constant coefficients, one could allow the user to pick either the exponential ansatz, i.e. insert $y(t) := \exp(\lambda t)$ or to transform this equation into a vector-valued first-order equation and solve this system by means of the matrix exponential function. As one can observe already in this tiny example, training networks are typically no trees. They are true graphs and may contain same nodes even on different levels in the hierarchical network, as the position of a node depends on the context where the exercise represented by it is needed. In the given example, the Gauss Algorithm as a basic exercise could also be stated as an initial problem to solve for.

Cross-links to the corresponding chapters in a lecture, available in the content area provide the necessary background to solve the assignments; providing several ways to master an exercise allows the student to keep his personal style or allows the teacher to setup an exercise network which sets relevant foci for the course.

The training area concept in its current state, developed chiefly by S. Jeschke and R. Seiler [Mumie], is mainly based on Java applets that state classical exercises in the field of undergraduate linear algebra, allowing the student to gain the required score to be admitted to the final exams. Figure 5 demonstrates a typical homework assignment within the training network. In the example shown, the user is asked to answer an assignment on matrix multiplication, kernel, and image of a linear mapping.

One of the major challenges in developing multimedia-based learning and teaching platforms is to drive dynamical validation forwards, i.e. using internal or external tools that are able to validate the correctness of a statement *dynamically* and *on the fly*. As far as the field of mathematics is concerned, computer algebra systems, automatic proving – if available – as well as specific numerical software have been used. The possibilities made available by linking external tools to training environments are currently not well exploited. A central reason for this unnecessary restriction is the lack of proper interface definitions – or even the total lack of any interface at all – at both the training tools as well as the external software.

Intelligent validation tools are characterised by their tolerance towards various notations and formulations of the same subject. To achieve this tolerance, it is not only necessary to encode exercises semantically instead of just providing a textual formulation, but also to use and interpret the same semantic correctly within the validation toolkit. The semantic coding of scientific content is therefore an important research challenge (cf. [W3C; Caprotti and Carlisle; Saarland Universität, TU Eindhoven, RISC Linz]).

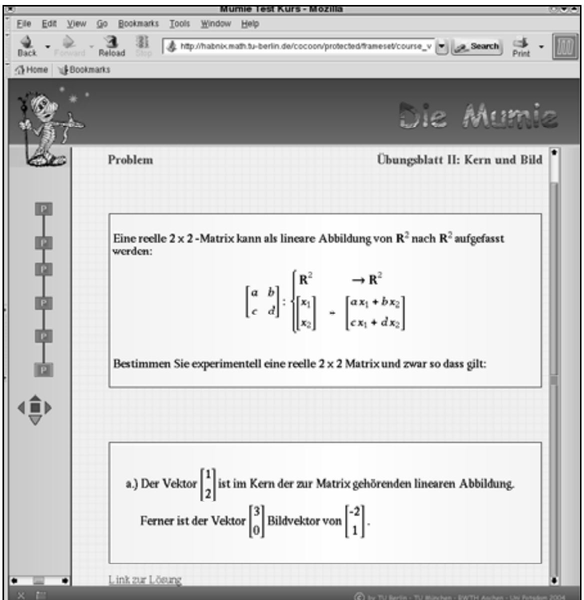


Figure 5: A typical homework assignment in the MUMIE-Platform. Here the reader is required to compute a 2×2 matrix such that $(1,2)^T$ is in its kernel and $(3,0)^T$ is the image of $(-2,1)^T$. Matrix multiplication is defined atop. On the left, the navigator through the exercise graph, which is a straight line here, is shown.

The prospects for intelligent assistants in this field are manifold, and the functionality provided by them is in most cases imperative to make learning within the training area fruitful. As discussed above, user input must be validated and qualified – let it be automatic or semi-automatic – and if weaknesses show up, intelligent feedback in the form of hints or reinforcing exercises must be generated. Additionally, intelligent feedback must be given to the learner as to why his exercise might have failed and which matter to study or repeat. Intelligent tutoring will help the training area tools to provide exercises that both adapt to the learning process of the student as well as to the preferred presentation form of the contents – for example students might either prefer a mathematically exact definition or a hand waving demonstration that appeals to intuition. By observing the user's behaviour, an intelligent tutoring system would then be able to offer a suitable exercise to train the student in the subtasks in which significant deficiencies have been identified.

VIRTUAL LABORATORY AREA

Virtual Laboratories use the metaphor of a scientific laboratory as guiding line for the design of the software. That is, similar to real laboratories, they provide the framework and equipment to setup and run experiments, i.e. qualitative and quantitative explorations of physical or mathematical phenomena of the model under examination. Similar to a real laboratory, it is not the primary goal of a Virtual Laboratory to define the experiments or the learning goals: it is left to the lecturer or organiser of a course to do so.

Applications of laboratories range from experiments for traditional lectures, homework assignments and practical training for students up to aiding researchers in experimentation and visualisation.

That is, a Virtual Laboratory has to deal with several classes of user types and deployment areas:

Demonstration setup: To address lecturers, it must be easy to setup and perform experiments, optimally by graphically combining the required components or loading readily-setup experiments from a data pool.

Training and tutorials: The main field of application for Virtual Laboratories as eLearning Software of the second generation [Jeschke and Keil-Slawik, 2004] lies in the implementation of autonomous exercises, projects, and training that are supported by a member of teaching staff. Sophisticated user's guidance is decisive in shortening the adjustment period to the laboratory's interface. Partially pre-prepared experiments can circumvent the access obstacles and encourage further experiments. However, the interface has to provide full access to all functionalities.

An important aspect for practical training courses is that a given problem has to be solved within the interaction of a group of learners. It is reasonable to demand the support for cooperative learning scenarios as an important feature of this concept of Virtual Laboratories. Last but not least, to make a Virtual Laboratory applicable to real-life research problems, its flexibility and proper integration of well-accepted computer algebra systems is imperative. The pedagogical profile for Virtual Laboratories is defined in detail in section ”Towards a pedagogical profile”.

Self-study, homework, preparation for exams: These fields of application require completely autonomous work from the students in order to train self-guided learning. Opposite to the previously mentioned introduction in training and tutorials, there is no support from the teaching staff, working hours and place of work are totally independent of the university. However, in this application, it is similarly desirable for the laboratory to be available to multiple users, thus supporting cooperation within homework groups, even outside the physical framework of the university.

Implementation in research: Here is where flexibility and, more importantly, the integration and interconnection of the laboratories with other software elements and existing infra-structures are highly required. While a good tutorial and intuitive user interface is indeed always helpful and highly desirable, this scenario can require a higher degree of previous knowledge and – where applicable – a certain adjustment period. The complexity and specific quality of the algorithms is decisive, as they define how appropriate the given laboratory is to the modelling of relevant problems.

A Virtual Laboratory should ideally address all the above-mentioned purposes. The given demands have consequences on the architecture of the software design under consideration: to offer interfaces that are adapted to the corresponding group of users, Virtual Laboratories are separated into kernels, user interfaces and a third interconnecting layer allowing the combination of various labs into larger experiments, see the subsection on ”Consequences for the Software Design”.

In its current form, mainly developed by Th. Richter, the existing Virtual Lab VIDEOEASEL [Richter] focuses on the field of statistical physics and statistical mechanics, described in detail in the VIDEOEASEL section. This field was chosen because it ideally combines mathematical research and its applications to important problems of natural science, engineering and economy: elementary mathematics is often sufficient to construct interesting models in this area, and thus the construction of models is very accessible to students. However, highly non-trivial phenomena can be observed and measured due to the interaction of the atomic components that are, even today, only partially understood and still under research. These components might be the molecules of a gas, spins of a ferromagnet, etc. To give just one example: phase transitions as an everyday phenomenon in large physical systems are complex phenomena, however exact proofs exist only for the most elementary models, e.g. the Ising model [Ising, 1925; Onsager, 1944].

Within VIDEOEASEL, probability, analysis, dynamical systems, and cellular automata are the important mathematical disciplines which can be brought to action in an environment of interesting applications like image data compression and de-noising, phase transitions, transmission from microscopic reversibility to macroscopic irreversibility for large numbers of atomic components [Jeschke, Richter, Seiler, 2005].

Towards a Pedagogical Profile

One of the defining principles of a Virtual Laboratory is that it does not define “learning units” – instead, it defines “learning spaces” for virtual experiments. The definition of learning goals remains to be given by the lecturer or the leader of the seminar or training course. Virtual Laboratories are tools to achieve a high-quality education *within* a course by providing virtual devices, algorithms, etc. The goal is rather to allow students to develop skills in problem solving and self-controlled learning that are required for their future professional work. Especially laboratories should provide enough freedom for individual and/or unusual experiments outside the limitations set by the curriculum [Hampel, Keil-Slawik, Ferber, 1999]. Thus, a laboratory should allow an explorative learning style and should *encourage self-guided learning* rather than just present concepts. The freedom to set up and combine experiments by themselves trains students to solve problems by themselves rather than adapting given solutions. This also addresses creativity, thus increasing the learners’ motivation.

Another aspect of Virtual Laboratories is that they address a broader audience than traditional eLearning solutions; rather than to convey scientific material to students as in a lecture, a Virtual Laboratory is – like its “real” counterpart – a *tool* whose primary purpose is to run experiments. This means that even though the main application of the Virtual Laboratories discussed here is in the training and educational field, applications in research as in a scientific workplace are possible and should be considered.

Due to the larger audience, Virtual Laboratories ought to be able to adapt themselves to the style of the individual users. The necessity to handle multiple application targets with partially diverging goals has several impacts on the software design, which will be discussed in the next section, and is one of the driving forces towards intelligent assistants.

Due to the high degree of specialisation we find today, scientific results are more and more accomplishments of cooperations between individuals and the outcome of teamwork; this is due to the high complexity of today's problems that requires the cooperation of experts from various fields. Therefore, *teamwork* has to be actively promoted by Virtual Laboratories as well. By using networked applications, even cooperation across borders should be made available.

Last but not least, a laboratory should integrate into existing software infrastructure by using standard components from the everyday environment of the working scientist or engineer; this includes products like Maple, Mathematica, or Matlab. In first place these tools provide numerical algorithms to analyse measured data, but using them also allows the students to familiarise themselves with software required for their professional life.

Consequences for the Software Design

The desired pedagogical profile formulated in previous section shows its consequences in the design of a Virtual Laboratory. Since laboratories have to address various targets and interest groups, see the "Virtual Lab Area", a highly granular software design is required which provides laboratory modules that are individually adapted to their target audience. This results in the separation of the laboratory into a kernel and several user interfaces that each fit their respective task and audience best. Furthermore, once the components are equipped with *open* interfaces using *accepted open standards*, the possibilities to combine these components freely to experiments beyond their initial application target and to reuse them outside their initial operational area are gained.

For that reason a three-tiered design for components is promoted in the following, cf. Fig. 6:

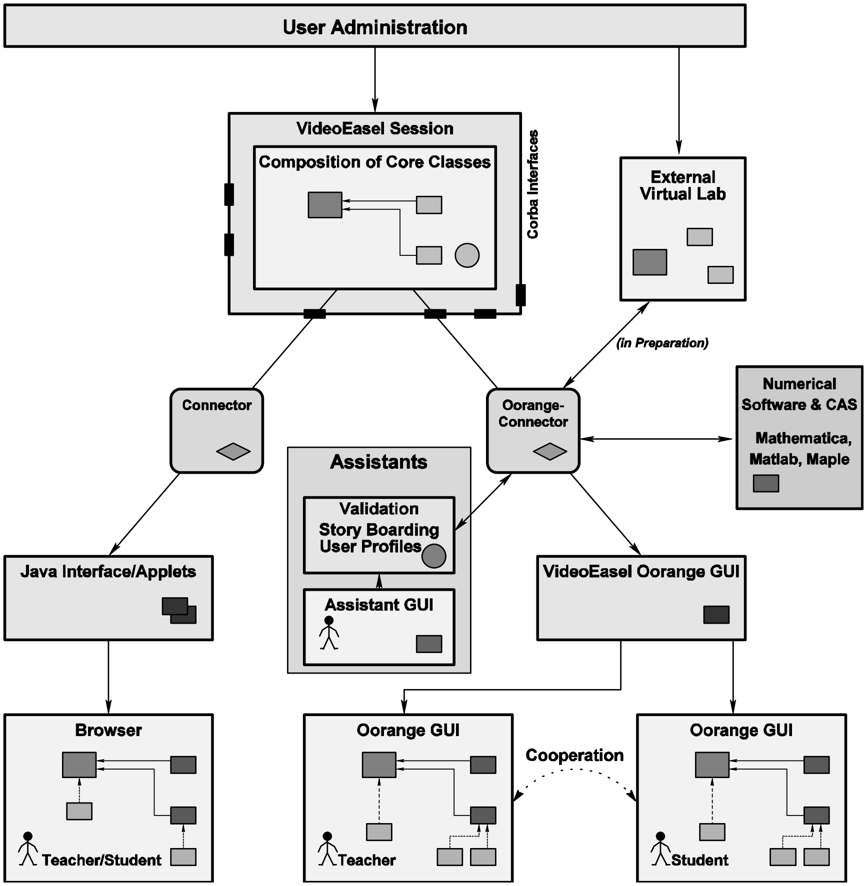


Figure 6: Software Architecture of a Virtual Lab, here VIDEOEASEL. In the top-middle the kernel which is controlled by the connector layer (rounded boxes) through CORBA interfaces [Scallan] (small black boxes). The connectors are linked to various front-ends providing their individual GUIs (light grey, bottom). On the right-hand side, an external component, e.g. a CAS, is talking to a connector. Assistant technology observes user behaviour and controls the kernel (grey box in the middle) by its own GUI (light grey box within). A user administration (topmost box) controls access to running experiments.

Simulation and computation components: These components implement the number crunchers in Virtual Laboratories; they do not provide any visible user interface, but rather implement the physical modelling of the entities to be measured. The only kind of interface they provide is one that defines the parameters for the experiment they emulate and that allows to extract the experimental results. In Fig. 6 this is the kernel.

Connectors: Connectors are software components that aid the user to combine and link the components of a laboratory to an experiment. In the simplest case, they could be realised by means of a script that extracts measurements from a laboratory kernel as described in the previous point, and feeds this data back into a numerical algebra program. Ideally, this kind of linkage should be carried out with minimum effort by means of a graphical user interface. The second task of the connectors is to translate and adapt the languages and interfaces between distinct laboratories. Even though we should enforce a unique interface definition for all laboratory components, it seems to be unrealistic to achieve this goal in practice, especially when having to deal with components from various sources. In Fig. 6, the rounded boxes in the middle depict these.

User interfaces: User interfaces address the needs and goals of the user group experimenting with a Virtual Laboratory. Depending on the application target, a user interface may present a readily set-up experiment for demonstration purposes in a lecture, an applet in an internet browser or an experiment in a GUI showing an experiment for students in practical training. A computer algebra system talking to the laboratory kernel through a connector might also provide a user interface of its own that is better suited to numerical analysis. Thus, in general more than one user interface will be required.

Once the separation into the above classes is understood, it is natural how to address the requirement of supporting cooperative learning scenarios: if we allow data streams between components to cross machine-boundaries, thus allowing the exchange of data in a network, one simulation component running on a server can be observed and investigated by more than one student at once – each experimenting and measuring from a client at a possibly remote location. Side channels would then allow students to communicate and exchange their experiences. A peer-to-peer network would be an alternative architecture for a distributed Virtual Laboratory. Nevertheless, in our understanding *cooperative learning requires networked applications* [Hampel, Keil-Slawik, 2001].

VIDEOEASEL – a Virtual Lab for Statistical Mechanics

An implementation of a Virtual Laboratory prototype in the above sense, demonstrating the impacts of the above demands on pedagogics and software design is the Virtual Laboratory VIDEOEASEL [Richter], developed at the DFG research centre MATHEON [Matheon] of the Berlin universities. VIDEOEASEL focuses on statistical mechanics, usually lectured in the second semester of the "Mathematical Physics" course. Typically, the audience is a mixture of mathematics and physics students that participate in the course "Mathematical Physics II" which is taught in the 6th semester for mathematicians and physicists.

VIDEOEASEL implements the microscopic rules of physical systems that are of interest in this area, for example the classical Ising Model or Lattice Gas Models, by using so-called *Cellular Automata* [Toffoli and Margolus, 1987]. A cellular automaton defines a time evolution on a matrix, often visualised by coloured cells. The next state of a matrix element is hereby computed solely by the so-called microscopic rule from the state of the element and its surrounding neighbours. Cellular automata are simple enough to be described by elementary math while being capable of demonstrating a rich set of complex phenomena. Thus, they invite to experiment without building up barriers when entering the field.

According to the demands mentioned above, VIDEOEASEL is based on a three-tiered software design separated into a computation kernel implementing the microscopic dynamics of a physical system, an interface/connector layer and several GUI front-ends that allow users to observe and manipulate the experiment. The interface between the kernel and the connector is realised by means of the well-established CORBA middleware [Scallan], thus making the laboratory a networked application. The kernel is designed to handle several connections at once, even from several users observing the same experiment. VIDEOEASEL therefore supports cooperative and remote learning scenarios right away.

The microscopic rules are written in a C-like programming language that is compiled and linked to the kernel at runtime; a set of predefined programs implementing various experiments are available on the server, though the user is always invited to modify and change these rules locally if desired. Similar to the experiments, measurement tools exist as microscopic rules defining the physical entities to be observed and measured. Following the granular design philosophy, they can be plugged into each experiment as long as the objects referred to by the measurement tool exist in the object to be measured on.

Several user front-ends are provided, compare Figure 7: the simplest one is a Java applet which is able to display an experiment along with some of its parameters in a web browser, thus making it applicable for on-line experiments or quick demonstrations in a lecture. A more complete stand-alone Java GUI hides most of the complexity while making a large subset of the possibilities available to its users, including the attachment of measurement tools and the editing of the microscopic rules. This interface was mainly designed to be used in practical training since it makes readily set-up experiments quickly available for the students.

An interface to the Oorange toolkit, a Java programming tool also developed at the Berlin University of Technology [Oorange], is provided to make VIDEOEASEL applicable to research problems. It represents the objects of VIDEOEASEL, i.e. algorithms and measurement tools as well as parameters controlling its operation as boxes which can be linked together by means of "drag and drop". Oorange can be used to have VIDEOEASEL talk to external applications, given these applications provide a Java interface themselves. Oorange is therefore a classical connector in the sense of the above section.

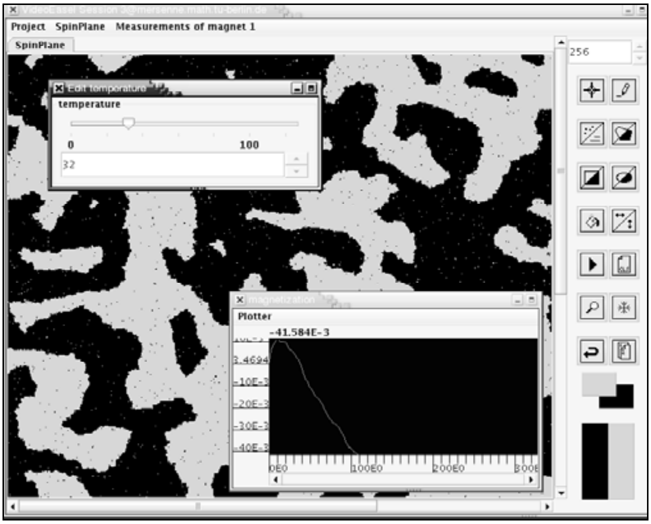


Figure 7a: Java front-end for a Virtual Laboratory.

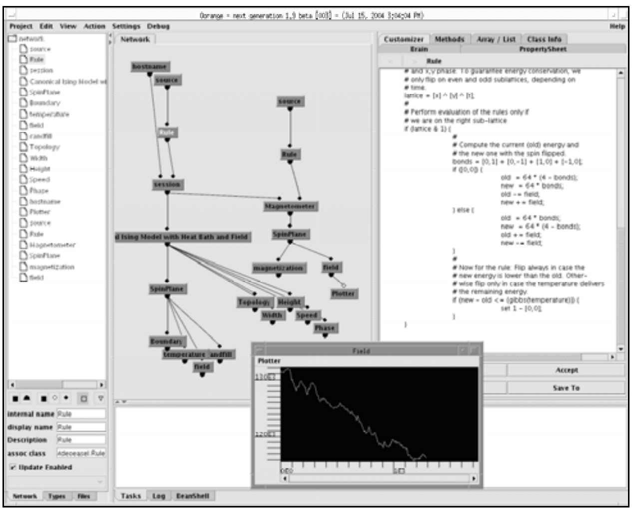


Figure 7b: Oorange interface for a Virtual Laboratory.

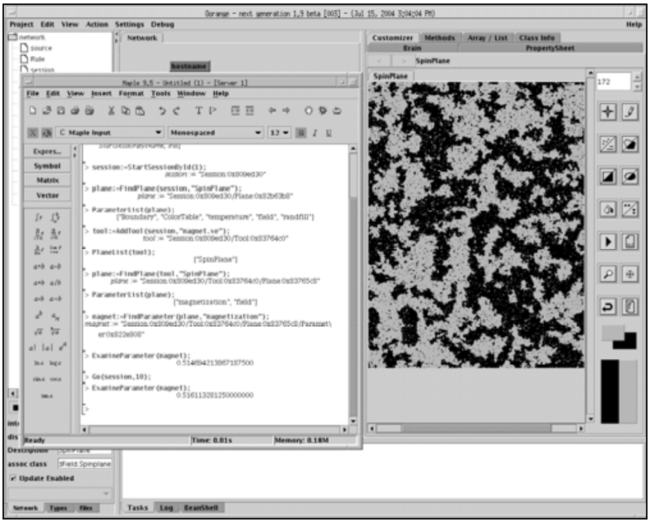


Figure 7c: Oorange interface communicating with Maple.

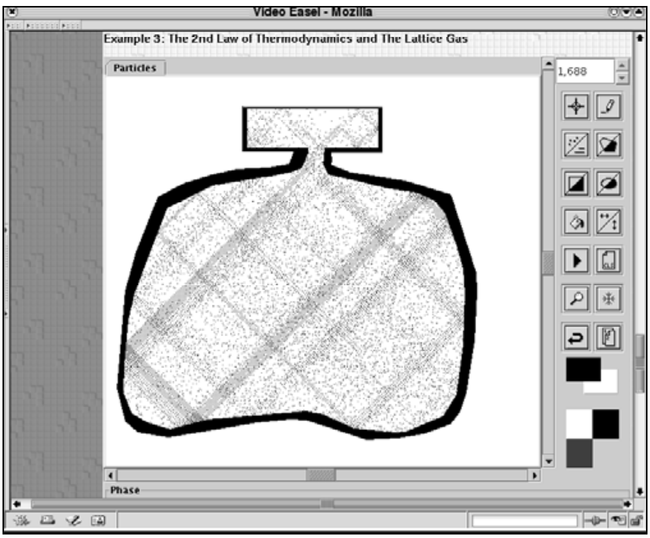


Figure 7d: Applet front-end for a Virtual Laboratory, showing an experiment on the role of reversibility in thermodynamics in lattice gases, the remaining three all visualise the Ising model, measuring the magnetisation over time.

Last but not least, an interface to the computer algebra system "Maple" to analyse the measured data and to control the laboratory from there in order to run even more complex measurement tasks is supplied.

From Microscopic Dynamics to Algorithms

To shed some light on the way VIDEOEASEL works, the classical model of statistical mechanics, the Ising Model [Ising, 1925], is briefly discussed. The automaton operates here on a lattice whose elements are called "spins". Each spin can be in one of only two possible states, classically called "spin up" and "spin down". Within the VIDEOEASEL world, the spin configuration is visualised by a two-dimensional drawing canvas, with spins pointing up shown in white and spins pointing down depicted in black. The offered user front-ends enable the user to manipulate the spin-configuration in a GUI that looks and feels very much like a painting program. The Ising Model and a typical spin configuration for this model is seen in Figure 7a.

The dynamics of the system is launched by a button in the GUI; this dynamics is defined by a "microscopic rule" that computes the next state of a spin – thus the colour of a pixel on the screen – by its own state, the states of its neighbouring pixels and the set of external parameters that can also be defined in the GUI. For the model at hand, the dynamics is given by the Metropolis Algorithm [Metropolis et al., 1953], implemented in the internal language provided by the Lab. Each neighbouring pair of spins is assigned a number, called its "energy" where parallel spins contribute low and anti-parallel spins high energy. Leaving some technicalities aside, the Metropolis algorithm computes the energy for all spin pairs and compares this energy with that of a configuration where a randomly picked spin state is replaced by its opposite, i.e. with the spin "flipped around". If the energy of the flipped configuration is smaller than that of the current one, the spin is flipped. Otherwise, if the energy difference between new and old configuration is smaller than the energy a “heat bath” contributes, the spin flip is performed as well. In all other cases, the spin remains in its current state. External parameters control the temperature of the heat bath as well as an additional energy source representing an external magnetic field.

Measurement tools operate on the very same spin configuration, carrying out the measurement process by applying their defining algorithm once for each available spin. In the simplest case of measuring the overall magnetisation of the Ising Model, this algorithm would simply consist of adding one unit to the magnetisation for each spin pointing up and subtracting one for each spin pointing down. Similar, though more complex rules can be formulated for other macroscopic observables, e.g. entropy, internal energy or Helmholtz free energy. Both, parameters as well as measurements can be manipulated by the CORBA interface.

Measurement tools and local dynamics need to match each other to make the measuring process meaningful, though. In the case of VIDEOEASEL, this identification is done in the simplest possible way: an automaton assigns names to its defining elements, and the measurement process identifies the configuration to measure on by their names.

Intelligent Assistants for Virtual Labs

To support users, the laboratory also provides *software assistants* as compiled Java code. Currently two types of assistants are implemented: First, so-called *wizards* provide simplified and streamlined user interfaces; the wizards do not add any functionality to the user interface, they rather bundle existing features and present them in a way that makes the problem at hand more accessible. E.g. the configuration wizard of the Ising model allows to set up the interaction energies of all possible spin-neighbourhood configurations. The added value lies in the assignment of an interpretation for each of the parameters by the wizard (otherwise tation for each of the parameters by the wizard (otherwise only represented by a labelled slider in the regular GUI).

Besides the static adaptation towards users available by selecting an appropriate front-end (see the VIDEOEASEL section) and enriching this front-end by wizards, the second kind of assistants provide user adaptivity through direct feedback from the user input: the so-called *tutors* present a guided tour through a predefined set of experiments to introduce students to a specific matter. That is, their purpose is to lower the entry barrier for students to run experiments on their own. Therefore, the assistants observe user behaviour, try to provide intelligent feedback and redirect the user to additional experiments to gain further insight. Figure 8 presents a prototypical tutoring system that demonstrates how matrix convolution operations act as filters on natural images and how to perform some basic image manipulations, here image smoothing.

The tutoring system uses the idea of *storyboards* [Jantke and Knauf, 2005]: a storyboard, as it is used here, is a network of assignments similar to the networks presented in “Training Area” section, though the nodes of this graph do not represent complete exercises, but smaller working units; each unit leads the user one step further into a possibly complex topic. Navigation on the network is driven by the assistant itself and not under direct control of the user. Figure 9 presents the network for the image convolution filter example introduced above:

The tutoring system observes user behaviour, validates and rates the answers and redirects the user to the suitable next step to perform. It is furthermore important to note that this tutoring system only makes suggestions on what the user might want to look at next. A student is always free to leave the tutoring system alone and experiment on his own.

Even though the current implementation uses only the most recent user input to select a path through the storyboard graph, the idea is naturally extended by user profiles, which is the next step to be taken in the development plan: namely, to select custom-tailored exercises by observed usage patterns, and by other side information gained on the user from other sources, e.g. the courses a user is participating in.

To perform user adaptation, modelling a learner according to the following "coordinate system" is proposed:

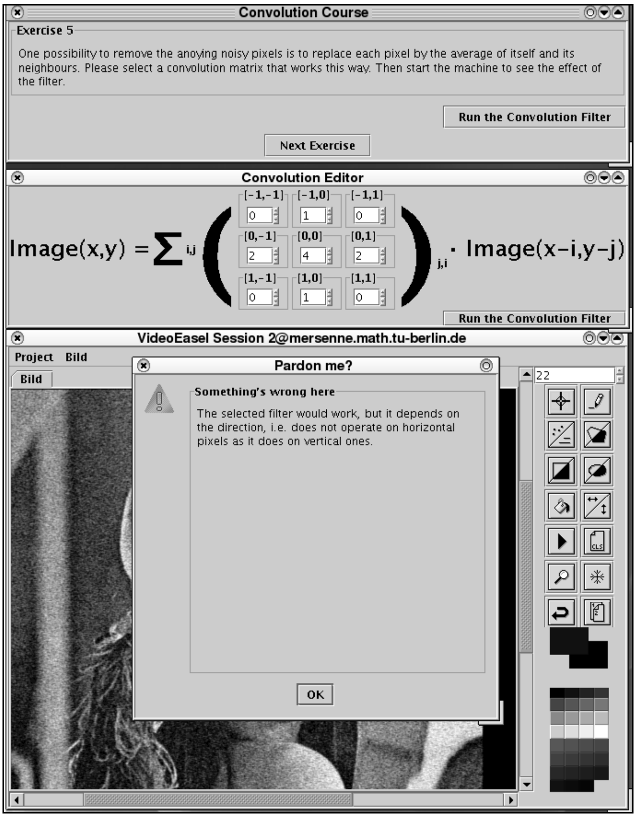


Figure 8: A tutoring assistant in a laboratory on matrix convolution. On top the current assignment to be performed; the middle window shows the configuration wizard for the convolution automaton, the target image is shown below. On top of the target image some feedback given by the assistant.

Graphical vs. Textual Orientation: To introduce a mathematical or physical phenomenon, users might either prefer a precise mathematical description, e.g. as textual formula, or a graphical/figurative description of the same content. A possible indicator on which presentation to choose might be the subject the learner is studying: mathematicians might prefer textual definitions whereas engineers might prefer graphical demonstrations. Note that a graphical representation is not necessarily imprecise or less powerful than a textual presentation of the same content.

Serialistic vs. Holistic Learner: Depending on preference or learning goal, a learner might first want to get a broad overview over a given field, or might want to delve deeply into the matter first, preferring to learn the material step by step [Scott, 2001; Pangaro, 2001]. This type of information is in our understanding best gained by observing the learner on navigation through the user interface, and by that selecting the nodes and learning units that fit best to the navigation style the user preferred so far. Clearly, user interfaces will have to be equipped by software that is capable of collecting this type of information.

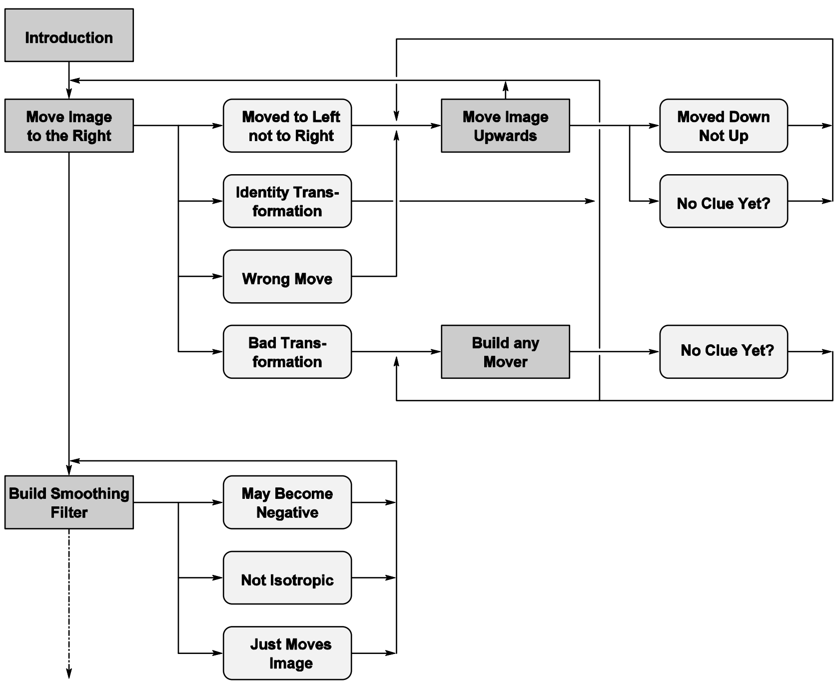


Figure 9: Excerpt from the storyboard on image filtering techniques. Working units in boxes, possible evaluation re-

sults in rounded boxes show the conditions for which assistant branches off to related experiments.

CONCLUSION AND OUTLOOK

VIDEOEASEL is currently still in a prototypical state, only limited practical experiences have been gained so far. A school project at the Heinrich Hertz School in Berlin revealed that the technology developed for cooperative learning and teaching is also very useful to help the administrator in providing individual support to students and to demonstrate individual achievements to the class. In contrast, unrestricted access of each student to the workplace of every other student as it was available in a preliminary version causes a lot of turbulence in the classroom. Therefore a minimal user administration was added. A second experiment using VIDEOEASEL as part of the Mathematical Physics course at the Berlin University of Technology is in progress at the time of writing; so far user feedback looks promising.

The process of gaining more experience with the content and training areas is going on, though the focus here is on undergraduate mathematics, specifically on the course on linear algebra for engineers. Besides purely technical aspects, a second problem specific to overcrowded undergraduate courses has to be faced: namely that of properly motivating abstract concepts and explaining their relevance to the audience's field of interest. Concrete exercises showing how abstract mathematics solve concrete problems is a major goal of our system.

To conclude, let us remark that we are not aiming at replacing frontal lectures or training courses by electronic media; we rather impose a *blended learning* approach: electronic media will *enrich* traditional courses by providing learning experiences that have not been possible before and that are more necessary than ever due to the changing demands of education. intelligent assistants are a valuable concept to reach this goal since they allow tailoring the software to the individual needs of the very broad audience we face at the Berlin University of Technology. The field of mathematics might act as a toy model to drive development of intelligent assistants in other fields further; even though construction of this technology for teaching mathematics might be simpler due to the highly developed internal structure and strongly formalised language hardly found anywhere else, we still believe that concepts and experiences gained here carry over to other fields.

References

Beans. JavaBeans. `http://java.sun.com/products/ejb/`

BeLearning, *Barrierefreies, Barrierefreies eLearning in Natur- und Ingenieurwissenschaften*. `http://www.belearning.de`.

Beierle, C. and Kern-Isberner, G. (2000). *Methoden wissensbasierter Systeme*. Vieweg, Braunschweig, Wiesbaden.

Dahlmann, N., Jeschke, S., Seiler, R., and Sinha, U. (2003). *MOSES meets MUMIE: Multimedia-based Education in Mathematics*. In Malpica, F., Tremante, A., and Sala, N., editors, International Conference on Education and Information Systems: Technologies and Applications, pages 370-375, Orlando, Fl. International Institute of Informatics and Systemics.

Dahlmann N., Jeschke S., Seiler R., Wilke, M. (2005), *Semantische Kodierung fachwissenschaftlicher Applets*, To appear in Lecture Notes in Informatics, Bonner Köllen Verlag, Bonn.

EU. *The Bologna Declaration* - Joint declaration of the European Ministers of Education, 1999. `http://www.bologna-berlin2003.de/pdf/bologna_declaration.pdf`

Görz, G., Rollinger, C.-R., and Schneeberger, J., editors (2000). *Handbuch der Künstliche Intelligenz*. Oldenbourg, München, Wien.

Hampel, R. Keil-Slawik, T. (2001). sTeam - *Designing an Integrative Infrastructure for Web-Based Computer Supported Cooperative Learning*. Proceedings of the 10th International World Wide Web Conference, May 1-5, 2001, pages 76-85, Hong Kong.

Explorations - *A New Form of Highly Interactive Learning Materials*. Proceedings of WebNet99 - World Conference on the WWW and Internet, pages 463-

468, Charlottesville: Association for the Advancement of Computing in Education. P. de Bra, J. Leggett (eds.).

Haun, M. (2002). *Handbuch Wissensmanagement*. Springer, Berlin, Heidelberg.

Ising, E. (1925). *Beitrag zur Theorie des Ferromagnetismus*. Zeitschrift für Physik, 31:253-258.

Jantke, K. P. and Knauf, R. (2005). *Didactic Design through Storyboarding: Standard Concepts for Standard Tools*. First Intl. Workshop on Dissemination of E-Learning Systems and Applications (DELTA 2005). Proc. of ACM Press.

Jeschke, S. (2004). *Mathematik in Virtuellen Wissensräumen - IuK-Strukturen und IT-Technologien in Lehre und Forschung*. PhD thesis, Technische Universität Berlin, Berlin.

Jeschke, S. and Keil-Slawik, R. (2004). *Next Generation in eLearning Technology - Die Elektrifizierung des Nürnberger Trichters und die Alternativen*. Informationsgesellschaft. Alcatel SEL Stiftung.

Jeschke, S., Kohlhase, M., and Seiler, R. (2004). *eLearning-, eTeaching- & eResearch-Technologien - Chancen und Potentiale für die Mathematik*. DMV-Nachrichten.

Jeschke, T. Richter, R. Seiler, S. (to appear June 2005). *VideoEasel: Architecture of Virtual Laboratories on Mathematics and Natural Sciences*. Proceedings of the 3rd International Conference on Multimedia and ICTs in Education, June 7-10, 2005, Careres/Spain.

Matheon. *DFG Research Center Matheon*, `http://www.matheon.de` .

Metropolis, N., Rosenbluth, A., Teller, M., and Teller, E. (1953). *Equations of state calculations by fast computing machines.* J. Chem. Phys, (21):1087-1091.

Moses. *Mobile Services for Students* at the Berlin University of Technology. `http://www.moses.tu-berlin.de`

MuLF. *Multimedia Centre for eLearning, eTeaching & eResearch* at the Berlin University of Technology. `http://www.mulf.tu-berlin.de`

Mumie community. `http://www.mumie.net`.

Natho, N. (2005). *mArachna: Eine semantische Analyse der mathematischen Sprache für ein computergestütztes Information Retrieval*. PhD thesis, Technische Universität Berlin, Berlin.

Nemesis. *New Media Support and Infrastructure* at the Berlin University of Technology. `http://www.nemesis.tu-berlin.de.`

O. Caprotti and D. Carlisle. OpenMath and MathML: *Semantic Mark Up for MathematicsOpenMath*. `http://www.openmath.org.`

Onsager, L. (1944). *A two-dimensional model with an order-disorder transformation*. Phys. Rev., 65:117-149.

Oorange. Oorange: *The Oorange development environment*. http://www.oorange.de.

Pangaro, P. (2001). THOUGHTSTICKER 1986: *A Personal History of Conversation Theory in Software, and its Progenitor*, Gordon Pask. *Kybernetes*, 30(5/6):790-806.

Richter, T. *VideoEasel*. http://www.math.tu-berlin.de/~thor/videoeasel.

Saarland Universität, TU Eindhoven, RISC Linz. *OMDoc*. http://www.mathweb.org/omdoc/

Scallan, T. *a corba primer*. http://www.omg.org.

Scott, B. (2001). *Conversational Theory: A constructivist, Dialogical Approach to Educational Technology*. Cybernetics & Human Knowning, 5(4).

Toffoli, T. and Margolus, N. (1987). *Cellular Automata Machines*. MIT Press Cambridge.

W3C, Math Working Group, W3C. *MathML*. http://www.w3.org/Math/