

E-Chalk

Technical Description

Gerald Friedland, Lars Knipping, Raúl Rojas

Freie Universität Berlin
Institut für Informatik
Takustr. 9
14195 Berlin
Germany
`echalk@inf.fu-berlin.de`

August 28, 2001



Abstract

E-Chalk is a software system for both classroom teaching and distance learning. The idea is to provide a tool which enhances classroom teaching, while having distance teaching as a side effect, i. e. at no extra costs. The basic metaphor that E-Chalk uses is the traditional blackboard that has proven to be an ideal teaching tool since the Ancient Greeks. The software is entirely written in Java, so listeners are not forced to install extra software to be able to get the contents. This article provides an introduction to the ideas that led to our approach, a basic overview of the features of the software system, and a technical description of E-Chalk's design.

Contents

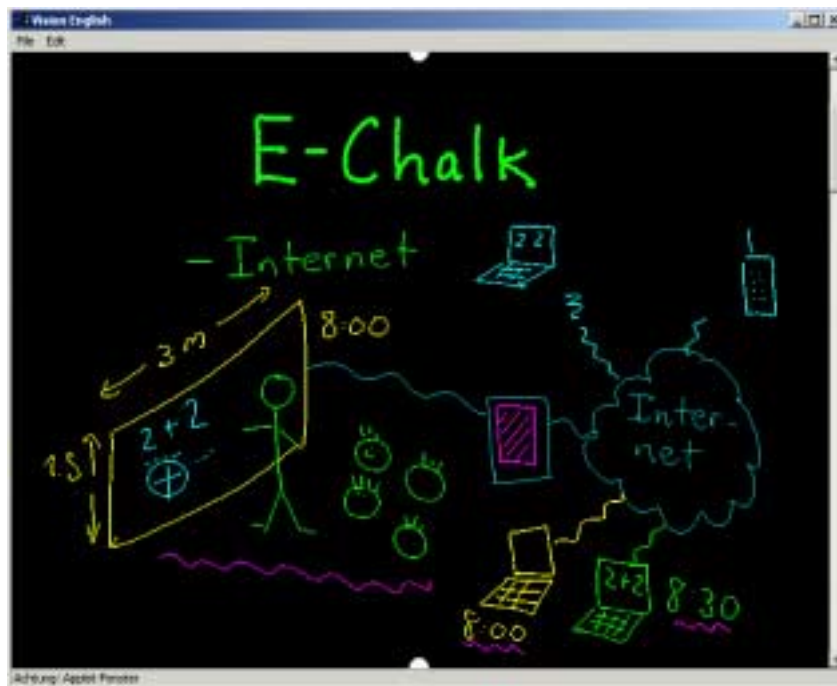
1	Introduction	3
2	The E-Chalk System	4
2.1	From Desktop to Blackboard	4
2.2	Features of the System	4
2.3	The Right Hardware for E-Chalk	6
2.4	Experiences	8
3	Related Projects	8
3.1	Academic Projects	8
3.1.1	“Ubiquitous Computing” at Xerox Parc	8
3.1.2	The “Zombieboard”	9
3.1.3	Transboard	9
3.1.4	Interactive Workspace	10
3.1.5	Extended Whiteboard	10
3.2	Commercial Whiteboard Systems	10
3.2.1	Mimio	11
3.2.2	eBeam	11
3.2.3	Smartboard	12
3.3	Conclusion of the comparison	13
4	Architecture and Implementation of the E-Chalk System	13
4.1	Overview	13
4.1.1	The server system	13
4.1.2	The client system	13
4.2	Database Connectivity	14
4.3	Printing lectures	15
4.4	Board	16
4.4.1	The GUI	16
4.4.2	The network server part	18
4.4.3	The algebraic server interface	18
4.4.4	The applet viewer	18
4.5	Audio	19
4.5.1	CODECs	19
4.5.2	Multicasting	20
4.6	Video	20
4.6.1	The CODEC	21
4.7	The Handwriting Recognition	21
4.7.1	Preprocessing	22
4.7.2	Classification System	22
4.7.3	Postprocessing	23
4.8	APIs	23
4.8.1	Media Applet Synchronization Interface (MASI)	23
4.8.2	Media Application Group Interface (MAGIC)	23
4.8.3	E-Chalk Pluggable Modules (EPM)	24
5	Summary and Perspective	24

1 Introduction

Although the chalkboard is one of the earliest teaching tools in history, it is still the preferred instrument in many scientific disciplines. Particularly math teachers appreciate the chalkboard for its naturalness and simplicity. Chalkboards are easy to handle and have good optical features. Even when used in huge lecture halls the chalkboard has enough contrast that its content can be seen from a distance of ten meters or more. Modern whiteboards made of synthetic material, however, do not provide such a good contrast. The most important advantage of the chalkboard is that the lecturer, while writing and talking, is slowed down to a speed where the listeners have a better chance to catch up with the material that is presented. The teacher thinks aloud and writes while the students have the time to understand the substance. This contrasts with overhead slides or computer based slide shows, which for this reason are quite unusable for lectures that deal with rather complicated content. However, slide presentations allow a better visualisation since tables, diagrams, or photos can be directly presented to the audience. Computer generated slides can also be printed out, so that students do not have to copy the content for a later recalling of the lecture. Many students complain about not being capable of copying the board image into their exercise-books while understanding the lecture. Naturally, this also leads to mistakes.

Besides these problems, distance learning gains more and more importance. People want to listen to courses when and wherever they wish to. Most presentation software allows to export generated slides as web pages, but there is no way to record and publish spoken words. Reading published slides is therefore not equivalent to sitting in the classroom and following the lecture. One solution for this problem is to create web pages with a lot more content than you would put on slides. Yet, building web pages that are both appealing and of pedagogic value, usually requires exorbitant efforts. Another solution is to record a video of the entire lecture containing the picture of the board, the lecturer, and an audiotrack. This approach does not improve classroom teaching and since they require a high bandwidth Internet connection the videos usually cannot easily reach every student's home.

The motivation for E-Chalk is to integrate all multimedial features of modern presentation software into the traditional chalkboard and thus enhancing classroom teaching. We want to preserve all good properties of the traditional chalkboard, including easy handling, while providing distance teaching as a pleasing side-effect.



2 The E-Chalk System

2.1 From Desktop to Blackboard

The principal idea of E-Chalk is the simulation of a chalkboard. Right after a first technical part, where a wizard allows the user to configure some properties, the system's user interface metaphor changes from the desktop to a blackboard.

First of all the computer screen becomes a visual output tool for more than one person. Everything that is displayed on the display is watched by many persons and must thus be understood by many persons. Secondly the keyboard loses its importance and needs to be used only seldomly. Our goal is to repeal the keyboard totally. Furthermore the mouse is not the proper input device anymore, since the lecturer is standing in front of an audience rather than sitting in front of a desktop. Finding the right hardware that meets these new requirements is not easy. We tried several devices which are presented in section 2.3. Ideally we think of a really large screen (e.g. 2.5×3.0 meters) that is touch sensitive. This screen must have a huge resolution concerning both, the amount of displayable pixels and the amount of contact sensitive dots per inch. The display must have good contrast, so that the visual quality can be compared to a real chalkboard, e. g. we do not want to darken the room for the lecture.

2.2 Features of the System

The software presents itself as a black screen where you can paint on in different colors and pen thicknesses. The board can be scrolled up and down vertically, providing the lecturer an unlimited surface to write on. Naturally, the user can also use a erazor to delete board content or he¹ can erase the entire screen.

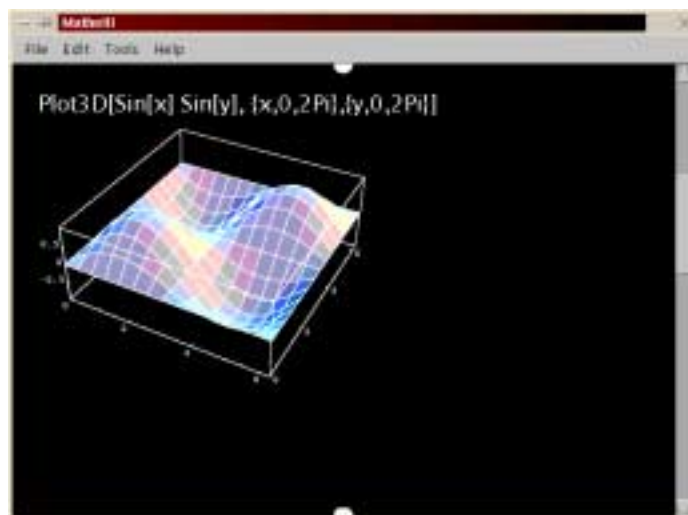
The system allows the user to insert images from the local disc or from the Internet. These images can be loaded directly from the web through entering a URL during the lecture. Even better, they can be loaded using bookmarks that were prepared prior to the lecture. Images inserted into the board can be annotated and edited just as if they were paper pictures stuck to a usual blackboard. This also creates the possibility to insert and annotate legacy slides, so that old slide lectures can be reevaluated. For the use of animations or interactive programmes, E-Chalk allows the user to insert Java Applets into the board. To do this, the lecturer specifies the URL of the web page where the applet is embedded. As with images, this can be done during the lecture or from a prior defined bookmark.



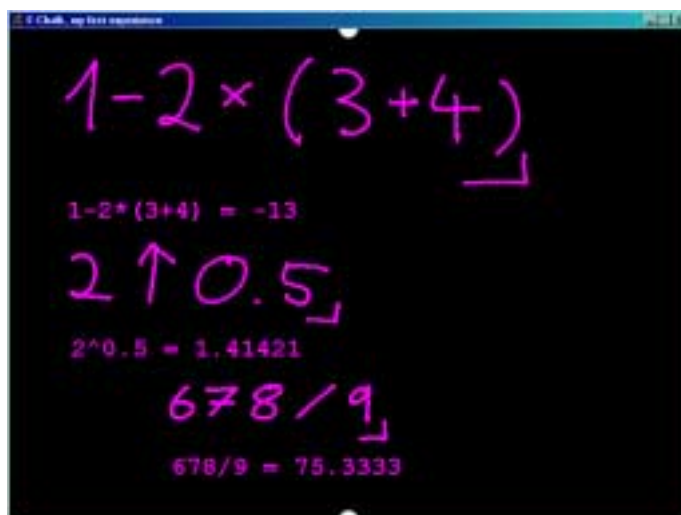
E-Chalk also allows the integration of an algebraic server system, such as Mathematica [W2].

¹“He” should be read as “she or he” throughout the whole article.

The user can evaluate expressions and plot functions directly on the board. Function plots are treated like images: They can be placed on the board anywhere and in any color. After a function plotting has been inserted it can be annotated and edited. Expressions are evaluated as text objects.



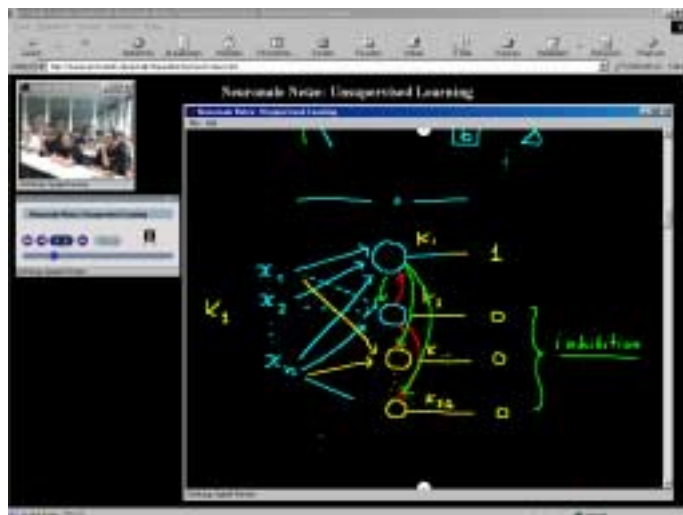
In the preliminary versions of E-Chalk the lecturer was forced to use the keyboard to enter commands for the algebraic server. However, this was not ergonomic and did not reflect the idea of E-Chalk. To provide a better way of interaction we implemented a handwriting recognition. So far it recognizes only digits, the basic mathematical operators (plus, minus, product, divide, and power), the decimal point, and parenthesis.



In order to tell the system that the symbols written on the board should be recognized, one color can be defined as the handwriting recognition color. The lecturer writes naturally as he would do with a mechanical blackboard. If he makes a mistake, he can erase the digits just as he would do on a mechanical chalkboard. To finish the teacher has to write a special ending symbol: A bottom left corner symbol, which reminds one of the emblem printed on the enter key of standard PC keyboards.

Everything that appears on the board is automatically transmitted into the Internet. Remote users can connect to the computer where the E-Chalk system is running and can view everything as it can be seen in the classroom. Additionally remote listeners get audio content to hear the

lecturer and optionally a small video of the teacher or classroom. The small video is not provided to give content related information, but to procure an impression of what is going on in the classroom in order to achieve psychological closeness to the lecturer.



As video streams consume CPU and connection resources, the client can close the video stream at any time to save bandwidth and processor time. The connection speed needed for a complete lecture with video is roughly 128kbit/s (that is the speed of 2 B-channels ISDN). Without the video stream the connection speed needs do not exceed 64kbit/s. Users do not need anything but a Java enabled web browser to be able to follow the lectures. There is no need to manually install any plug-in or client software. The E-Chalk system automatically creates all resources that are needed to follow lectures remotely. That includes the mirroring of all inserted Applets and images, the creation of webpages containing author and lecture name, and the preparation of a Java Applet client software that is able to receive and playback audio, video, and board contents. Lectures that were held with the E-Chalk system cannot only be transmitted live, they are also automatically stored for on demand access. As with live lectures, all resources are installed automatically and the remote user only has to open the created webpages using a web browser. In archived lectures the remote user is presented a console, that enables him to control the content flow similarly to the remote control of a VCR, i. e. pausing, fast-forwarding, and rewinding of the lecture. One advantage of this simple directory based storage of lectures is, that they can easily be packed into one file for download - a feature which many students use to save connection costs, when wanting to watch a lecture at home. However, the simple directory based storage of lectures also led to some problems which are discussed in section 4.2. For this reason we decided to also experiment with the storage of lectures in databases. The pros and cons of this approach are discussed in section 4.2, too.

Last but not least, archived lectures are automatically converted to Adobe's well known PDF format. This enables remote users to print all or partial contents of the board for later review.

2.3 The Right Hardware for E-Chalk

As mentioned above, the ideal hardware for E-Chalk is one that perfectly simulates the good characteristics of a chalkboard. Our first idea was to use a large plasma screen with a touch sensitive overlay. Plasma screens have a good contrast and luminosity, are relatively large, and can display sufficient resolutions. Unfortunately, plasma screens also have several disadvantages. At the time being, plasma screens are heavy and expensive. Furthermore they require a sophisticated cooling system, which makes a disturbing noise. Besides that, they are not available with a sufficiently large surface diagonal for big lecture rooms. Several manufacturers, such as NEC Inc., provide systems, that enable you to combine four plasma displays to one large screen [W3], but

they do not yet provide a touch screen overlay, since these systems are mainly designed for trade fair presentations. Finally, plasma displays are quite susceptible to screen burn-in, which rather restricts their usage to animated presentations. However, our first approach to use the E-Chalk system with a 43" plasma display from Fujitsu was unsatisfying because the touch screen overlay's resolution was insufficient for drawing. It was only usable for standard desktop operations, such as clicking on icons or dragging objects from one window to another.

The second idea was to use the Mimio system (see section 3.2.1) in connection with the plasma display. But instead of writing on a whiteboard, we wrote on the plasma screen (using a neutral pen). The Mimio system detects the position of the pen and emulates a mouse device. Nevertheless, the latency between actual drawing on the board and realization of this line on the screen was so big, that this Mimio plasma combination had to be considered unusable. Besides that, the overall problem that a 43" surface is too small for bigger classrooms, remained.

A first nice and cheap solution to the problem was to use a regular LCD projector to project the content of the board onto a white canvas or wall and then use a digitizer tablet as input device, such as the ones provided by Wacom, Inc. [W4]. Digitizer tablets have several advantages: They are cheap, mobile, and quite easy to handle. The user writes with a magnetic pen on a tablet. The tablet contains a grid of wires. The stylus induces a voltage at the point were it is currently writing. The position is tracked and used reported to the computer system. The user can write as he would do on a sheet of paper and the LCD projector shows the image.



The disadvantage of this approach is the handling: It differs remarkably from the handling of a mechanic blackboard. This is especially noticeable for there is no visual feedback on the tablet. The lecturer has to change repeatedly between looking on the screen and writing on the tablet. On a first try this seems annoying but a little bit of practice helps. Wacom, Inc. also provides digitizer tablets with build-in LCD display [W5], which is a try to get around the problem. However, there is a short visual distance between the display and the stylus (parallax error). Resultingly this tablet needs a short period of accostumization, too. After all, this approach is an acceptable solution, especially because the digitizing tablets are supported in most operating systems.

Numonics, Inc [W6] provides a front projection electronic whiteboard system called "Interactive Presentation Manager". It is very similar to the one described in section 3.2.3.



The main virtue of this system is that its handling is very close to the traditional blackboard. The lecturer stands in front of the audience and writes directly on the board while the audience watches the lecture looking at it. The tablet has a resolution of more than 300dpi and uses a magnetic stylus with a small rechargeable battery. We are currently using this front projection whiteboard as this solution comes closest to the basic idea. Yet, this approach has downsides, too. Although its 77" diagonal is sufficient for small lecture rooms (up to 15 students), this solution is not adequate for great lecture halls. Although the contrast is good (finally it depends on the LCD projector that is used), it does not reach the contrast of plasma displays and the lecture room has to be darkened. The main disadvantage is that the writing hand shades the position where the stylus is currently writing thus the lecturer does not see what he is actually doing.

2.4 Experiences

The first real application of the E-Chalk system was a lecture given by Professor Raúl Rojas in a computer architecture class in the winter term 2000. The lecture hall was filled up with 200 students and Professor Rojas used an LCD projector in connection with a digitizer tablet. The lecture has been archived and can be watched at our homepage [W7]. During the summer term 2001 several courses at the computer science department of the Freie Universität Berlin used the E-Chalk system. One example is the lecture about neural networks, which can be watched at the website of our institute [W1]. Roughly 20 students attended the course and we used Numonics' system described in section 2.3. The students' reaction was by all means positive. For solving the assignments or before the finals, most of the students favored to download the packed versions of the archived lectures to review the material at home. We also used the E-Chalk system for the undergraduate seminar "History of Computing" in the summer term 2001. In this course, as it is a seminar, the students had to prepare material and present it to the class using the E-Chalk system. Interestingly enough, none of the students had problems to reorientate from the mechanical chalkboard to the E-Chalk system.

3 Related Projects

In this section we will discuss some related projects and compare them to our approach.

3.1 Academic Projects

3.1.1 "Ubiquitous Computing" at Xerox Parc

At the end of the 1980s, computer scientists at Xerox Parc already thought of the possibility to use wide a LCD monitor as a kind of whiteboard. Their so called "live board" was targeted to be connected to the Internet to support collaborative work. The final product was a retroprojection system that was commercialized. Until 1998, several hundreds of these systems had been sold [3]. The main usage of this project was computer supported collaborative work via small notepads

where the participants could write their comments on, not classroom teaching. The idea survived in today's collaborative whiteboards.

3.1.2 The “Zombieboard”

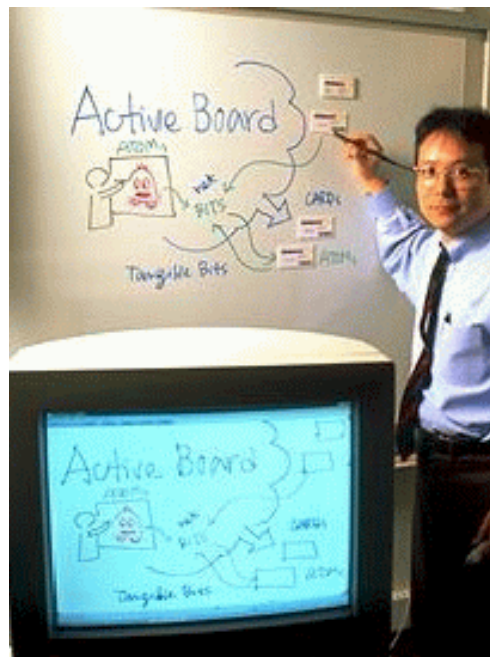
Another project at Xerox Parc is the so called “Zombieboard” [4]. It consists of a synthetic whiteboard that is scanned by a camera system. The images are analyzed by the software to cut out the lecturer. Only the contents of the board is transmitted through the Internet.



This interesting approach could as well be used for classroom teaching. However, in most cases touch screen technology will be a cheaper solution, especially if more than one board is needed.

3.1.3 Transboard

Transboard is a project at the MIT Media Lab [5].



This approach is similar to the E-Chalk system, but it is mainly targeted for collaborative work rather than for teaching. The main idea of Transboard is to store objects (sets of linestrokes) in physical maps which can be read and interpreted by the computer. The basic usage of the system is to hold brain storming sessions in front of the whiteboard. Remote users are able to participate in these discussions. Handwriting recognition or the use of interactive software does not play a prominent role in this system.

3.1.4 Interactive Workspace

The so called Interactive Workspace developed under the direction of Terry Winograd at Stanford University [6] is a multi-device, multi-user environment based on a new architecture that makes it easy to create and add new display and input devices, to move work of all kinds from one computing device to another, and to support and facilitate group interactions.



Their idea is to construct a “higher level operating system for the world of ubiquitous computing”. They are focusing their work on a meeting room where people want to do task-oriented work, rather than teaching. The project is heavily based on Microsoft operating systems.

3.1.5 Extended Whiteboard

The ideas of Professor Thomas Ottmann at the University of Freiburg (Germany) led to a so called “Extended Whiteboard for Authoring and Teleteaching on the fly” which was developed in cooperation with the University of Mannheim (Germany) [7]. This project also uses an electronic whiteboard which can be used for classroom teaching as well as for distance learning. Slides in the form of colored Postscript files, animations, and simulation programs have to be prepared by the lecturer before the presentation. Audio data, video data, and whiteboard events are captured and stored, so that they can be combined to one multimedia document after the lecture. These multimedia documents can be replayed using a receiver program. The receiver program can jump forward and backward inside the lectures and can directly position the stream to a certain index containing a certain topic.

The universities provide receiver software for the operating systems Irix (SGI), Solaris (SUN), and Linux (x86). If, for some reason, a lecture requires greater amounts of data, they are stored on an FTP server and have to be downloaded by the user prior to the replay of the presentation. For live transmissions the system uses the MBONE [W8] together with so called “Scalable Multicast Protocol” (smp), which is a development of the University of Mannheim.

Though the initial idea to enhance classroom teaching through an electronic board and store the lecture for later use in distance learning applications is the same, both systems are rather different. First, E-Chalk does not require the lecturer to prepare postscript slides and it also does not require the user to install anything prior to listen a lecture. Secondly, E-Chalk stresses on low bandwidth connections, while Professor Ottman’s approach uses the MBONE, which is mostly only available in academic institutions.

3.2 Commercial Whiteboard Systems

There are several commercial whiteboard systems which are in some way related to the E-Chalk project.

3.2.1 Mimio

Virtual Ink Cooperation [W9] provides a hardware software combination called Mimio. It captures whiteboard images and transmits them over the net.



Mimio uses a high-resolution ultrasonic position capture system consisting of a capture bar, color-coded marker sleeves and an electronic eraser. The electronic marker sleeves transmit ultrasonic and infrared signals to the capture bar, which triangulates the pen's position on the board as the user writes. The triangulated position relative to the whiteboards upper left corner is then transmitted via serial port. Virtual Ink provides a software package that is able to record and store the board content in different image formats and can emulate a mouse device. They also provide software that is able to transmit the "dynamics of the collaboration process" through the Internet. The great advantage of the system is its compactness and its low weight. It is hence ideal to be connected to a notebook for mobile use. However, the software package does not record audio or video data and all software is only available for Macintosh computers and Windows based systems. Another disadvantage is that the data returned by the positioning system is not accurate [W10]. Therefore the processing computer has to apply a filter on the data which results in a small delay between the time when a line is written on the whiteboard and when it appears on the computer screen. This is especially disturbing when using the mouse device emulator.

3.2.2 eBeam

eBeam by Electronics for Imaging, Inc [W11] is very similar to Mimio.



However, they provide software, which can record audio together with board events for synchronous and asynchronous transmission through the Internet. They also claim that their product captures more precisely and can follow the pen moving more quickly. This is achieved through the sensors are not fixed to a vertical capture bar. They are two separate sensors, one of them is attached to the upper left corner of the board and one of them is attached to the upper right corner of the whiteboard. This results in a wider triangulation angle, which is better for positioning [W10]. Like Virtual Ink, they only provide proprietary software for Windows based systems and for the Macintosh. However, there is a Java Applet for viewing recorded lectures.

This system, as well as the Mimio system, stresses on collaborative work. Since a physical

whiteboard is used as a writing surface, there is no way to place images, to use interactive software, or to recognize handwriting. Another problem when using a whiteboard is that you do not have a vertically unlimited screen.

3.2.3 Smartboard

The Smartboard is a hardware device by Smart Technologies, Inc [W12] that exists in three versions. One version is a rear projection system where the image of the computer screen is projected from behind, while the surface on which it is projected is touch sensitive for a special magnetic stylus. The stylus position emulates a mouse pointer position.



The salient disadvantage of this system is its physical volume and its weight.

The second version is a front projection system, the surface is still touch sensitive, but the image is projected via an LCD projector, which stands in front of the system. The main problem with this approach is, that the writer does not see what he is writing, because his hand shades the position on the board where the stylus is currently writing (see section 2.4).



The third version is a transparent touch sensitive overlay for plasma displays. The overlay is fastened on the top of the display. However, as discussed in section 2.4, plasma displays are still too small to be used in great lecture rooms.



3.3 Conclusion of the comparison

Most of the above mentioned projects and products stress on collaborative work. Especially commercial systems seem to focus on mobile collaborative Internet meetings, rather than classroom teaching. The resulting software systems hence stress on one-to-one or many-to-many communication schemes (rather than one-to-many) which requires more bandwidth and mostly equivalent computer systems on each end of the communication channel and overall a higher bandwidth. Nevertheless, as described in section 2.3, we successfully combined some of the hardware products with the E-Chalk software system.

4 Architecture and Implementation of the E-Chalk System

4.1 Overview

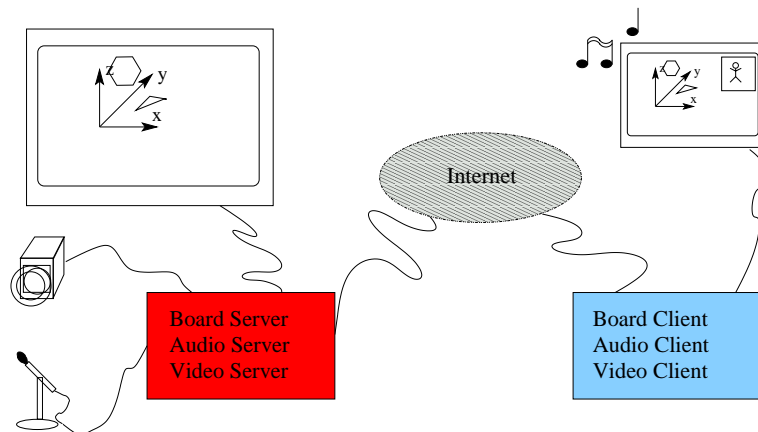
4.1.1 The server system

The server system itself currently consists of three main components and several minor components. The three main components are the audio, the video, and the board server. Minor components are: The database inserter and the PDF converter. All server components are controlled by the E-Chalk Startup Wizard. The wizard is the first program executed when a user starts the E-Chalk server system: The user is asked to configure the system and, when finished, able to start being productive. The E-Chalk wizard then executes the main components and controls them in several ways, depending on the type of component (the idea is similar to an operating system kernel). Minor components can only be started and terminated, while major components can be controlled similarly to the functions of a VCR remote control (see section 4.8.2 for details). This enables the user to pause during a live lecture or to correct mistakes that were recently made. All server components have the possibility to report errors during a live lecture. After the live transmission of a lecture the E-Chalk wizard makes sure, that all components needed to view the lecture as an archived lecture are present in the output directory. This output directory can directly be used to watch the lecture. The E-Chalk wizard then executes the minor components, such as the PDF converter (see 4.3) to make the lecture printable or the database inserter to insert the lecture into a central database (see 4.2).

4.1.2 The client system

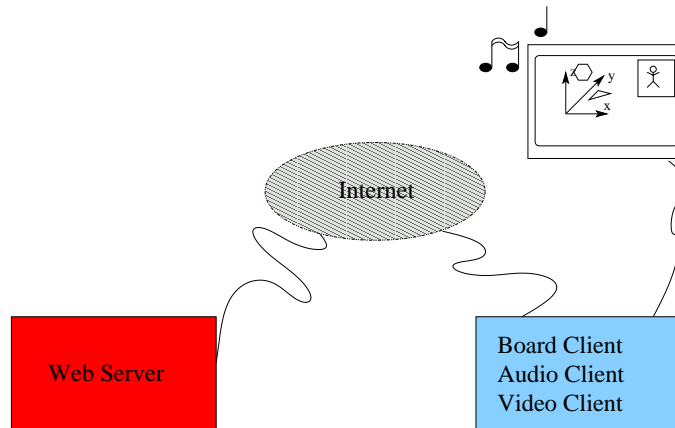
The E-Chalk client system consists of a set independent receiver Applets that synchronize themselves by communicating through our Media Applet Synchronization Interface (MASI). See section 4.8.1 for further details on this interface. Additionally there is a console Applet, which allows operations like fast-forwarding and rewinding, and a slideshow Applet, which allows to display arbitrary webpages synchronized to the audiostream or videostream.

The diagram shows an overview of E-Chalk's architecture for live transmissions:



As explained above, E-Chalk has basically two ways of operating: Live transmissions and archived transmissions. During live transmissions the audiostream, videostream, and board events are recorded from their devices, compressed, sent, received, uncompressed, and replayed in real time. While this is done, everything that is sent out is also recorded to files. Resultingly, the E-Chalk client system has two modes, as well. In live mode, each client connects to its corresponding server, through a socket connection. In on demand mode clients use a http connection to receive the files therefore no E-Chalk server is needed.

The next scheme visualizes how archived lectures are transmitted:



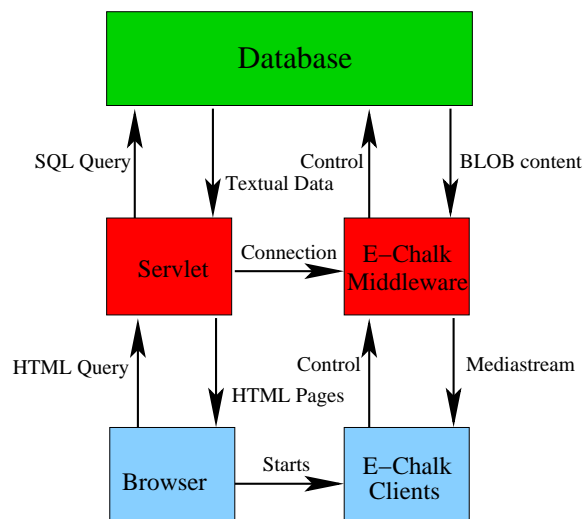
4.2 Database Connectivity

For archiving lectures, the above mentioned architecture is simple and useful. The downside of this architecture is, that a lecture is stored on the haddisk of the computer where the lecture was given. If it is wished to store them in a more centralized manner, e.g. to published them on one single webservice, this has to be done manually. As explained above, every lecture is stored in a directory, this means every lecture contains all the ressources needed for remote listening to the lecture. The disadvantage is that if, for example, there is a new version of the client, each directory has to be updated. The third problem is a technical efficiency problem that arises with fast-forwarding and fast-backwarding of lectures: To skip parts of the lecture the stream has to be reloaded entirely until the point where playback should continue².

That is why we experimented with the storage of E-Chalk lectures in a database system. The actual lecture content is stored as binary large objects (BLOBs) in the database. Additionally

²Even though it has been specified since HTML 1.1 [18], no http server daemon known to us implements the GET command with the ability to specify a start offset.

the database contains tables for user administration, revision controlling of lecture resources, and for editing lecture metadata. The idea is to integrate an E-Chalk server middleware into the standard three tier architecture that is being suggested by the Java Servlet API. An overview of the architecture is shown in the next figure.



On the left half of the scheme shown you see a normal webbrowser communicating with the database via a Servlet. The user can login and identify himself to the system (this is optional, but useful to make certain lectures private). Then he is presented a search engine, where he can search for lectures according to different criterias, e.g professor, topic, or some keyword that has been defined for the lecture. This, together with some other administrative tasks happens through communication between webbrowser, Servlet, and database. When a user actually selects to listen to a lecture, the right half of the above scheme comes to play. The Servlet communicates with the E-Chalk server middleware and sends it a message, that in a few seconds a client with a certain ID will connect. After that the middleware begins to retrieve the lecture from the database. In the same time, the Servlet expands and sends the needed HTML pages and other resources to the client. The webbrowser of the client shows the webpage and download the proper E-Chalk client Applet that itself connects to the middleware. In the next moment the middleware begins to send audio, video, and board event stream to the client. As a result, the user watches the lecture.

When the remote viewer presses a button on the console to fast-forward or rewind the lecture, the E-Chalk client Applet tells the middleware, which forwardes the request to the database. This approach is very efficient, since data seeking is done inside the database system (which is rather optimized for such tasks) and only the information that is actually used is transmitted over the Internet.

Administration of content that is already inserted into the database is done through a webinterface, while insertion of content is controlled through the E-Chalk startup wizard. E-Chalk lectures can be inserted from any host in the Internet. Resultingly this tool is really helpful for centralizing lectures. When there is a new client version, lectures that have version stamps that declare them compatible with new client are automatically updated, so that users always use the newest client possible.

4.3 Printing lectures

Adobe's Portable Document Format (PDF) is more than capable of representing all of the structures resulting from board events in E-Chalk lectures, including lines, text, and graphics. Nevertheless, one complication appears when trying to print Applets, which of course are not supported by PDF, and which are rather difficult to print because their content can be dynamic or even interactive. On the one hand it is possible to represent the applet on the page with a screen

capture from it and put this where the applet would appear on the page, on the other hand the appearance of the Applet on the page is rather unpredictable. The other complications that arise from the temporal nature of the lecture, including erasing marks and events that place marks over those from previous events marks, are easily represented in PDF, because it supports linear page descriptions allowing for the same sort of erasing and superimposing.

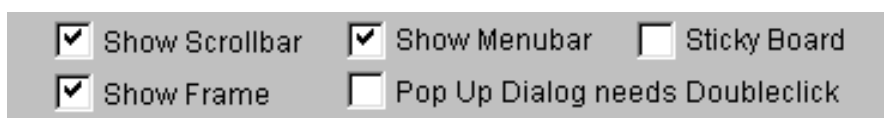
Another problem is breaking up the lecture (which is one, continuous piece) into pages. The approach taken to this is to determine which horizontal lines in the lecture have the least markings, and, weighting this against the distance to the natural page break, a line is chosen which, in the best case, is not perceptibly removed from the natural break, and which has no markings on it.

4.4 Board

Following the analogy that the wizard can be compared with an operating system kernel, the board server is the graphical user interface. Although the board server is the heart of the E-Chalk system, it can theoretically work without it, using only audio and/or video transmission. Of course, the result is that the system lacks its greatest feature. The board server mainly consists of five parts: The GUI, the network part, the algebraic server interface, the Applet viewer, and the handwriting recognition part. The handwriting recognition part is described in section 4.7 of this document, while the other parts are described in the following paragraphs.

4.4.1 The GUI

As explained earlier, the metaphor that underlies our GUI is the metaphor of a chalkboard rather than a desktop. As a matter of fact we tried not to stick to old desktop metaphor GUI elements, such as menubars, scrollpanels, or textinputfields. There are two reasons for us still using those elements at certain places: Either we have not implemented an adequate solution yet, or we kept the old elements to make it possible for the user to work with those elements as long as he has not yet accustomed to the new ones. In the latter case, however, we make the old element optional.



It is very important that no GUI element disturbs the flow of the lecture. Molestation of the current flow of the lecture would lead to inattentive students. Another goal therefore is to try to avoid anything that surprises people, like arbitrarily popping up dialog boxes. Due to the facts described in 2.1 we want to make it possible to reach any feature of the board with fewest possible clicks. If an element needs to be more complex, it must be designed in such way, that the audience can understand what the lecturer does, otherwise we get the same effect as with contrainuitive or surprising GUI components: Again the attention of the audience is lost.

Our solution to approach the goal stated above, was to keep the GUI simple (KISS). The basic working environment is a simple black screen, without any windowframe, menubar, or scrollpanel (these three components belong to those which can be switched on or off). The lecturer can simply draw on the screen. To change the chalk color and thickness, he can ideally use predefined function keys on his hardware device (many hardware solutions we presented in chapter 3 have certain functional buttons that the user can configure as shortcuts), so that he does not need any GUI element for that.



Changing to erazor works the same way, since nothing is more annoying than a long complicated way to erase a mistake. However, we experienced that the virtual erazor is not as effective than a mechanical one, i. e. it takes longer to erase something with a virtual erazor than it would

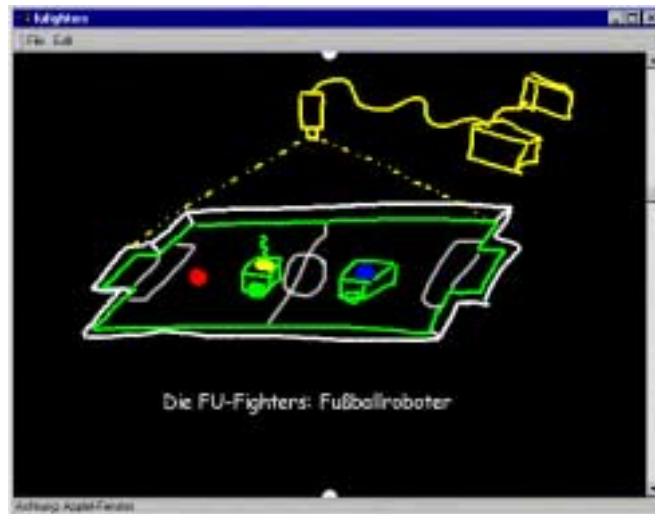
take with a mechanical one, so we implemented an undo/redo functionality, which erases the last linestroke. Mapping one function button to this operation results in a very efficient error correction tool. If you do not have such function buttons or for using operations that are more complex we chose to design a toolbox. The toolbox can be opened by clicking the button on the stylus, that for the operating system maps to the right mouse button (every hardware system known to us has a way to emulate the right mouse button).



The following figure shows the toolbox. The icons have the following semantics (from upper left, to bottom right): exit, undo, redo, switch between erazor and chalk, clear all, help, insert image, insert image from bookmark, insert Applet, insert Applet from bookmark, insert command for algebraic server, insert keyboard writing.



In order to avoid accidental opening of the toolbox, the toolbox can be configured to open only on double clicking the stylus button. Using the toolbox, you can easily change the chalk color and thickness, use the undo/redo function, change to the erazor by only one additional click. Additionally you can access further operations like inserting images, Applets, or requests to the algebraic server. These, however, require further interaction with the system. In the case of the insertion functions for Applets and Images the user can predefine bookmarks to avoid keyboard typing during the lecture. To send out commands for the algebraic server we developed a panel that enables the user to access the most important mathematical functions and operators by one click. The panel for the algebraic server, however, is just a temporal solution. We are working towards a fully featured handwriting recognition for E-Chalk. Later on it will be possible to insert pictures and applets by writing the URL or a shortcut to a URL on the board. Even complex mathematical expression will be evaluated after they were handwritten. If for some reason the use of the keyboard is still wanted, the user also has the option to use it, using different font colors and sizes.



As we found that scrollbars are not the optimal way to scroll a blackboard screen, we designed two scrollpoints. The scrollpoints are located at the top and at the bottom of the screen. To scroll, the lecturer clicks on these scrollpoints and drags the stylus in the direction he wants to scroll. The movement is similar to the one that is done when a person grabs a strap to pull a canvas up or down.

4.4.2 The network server part

The GUI event handler records all events that have taken place on the board and passes them to the network server which sends them out into the Internet and also records them to a file. Events are sent out immediately. If a client connects after the lecture has already started, the server sends it all events that have been recorded up to that moment, so that the client always has the entire board image. Even though the format we are currently using contains no compression, the bandwidth needed is only 3 kbit/s. The newer format uses a simple difference coding compression and needs less than 160 bit/s (images and Applet data excluded).

4.4.3 The algebraic server interface

The algebraic server interface is currently restricted for the use with Wolfram's Mathematica [W2]. To communicate with Mathematica we use the JLink interface [W13]. Images, that result from graphical commands, like function plotting, are saved and inserted as GIF images, while evaluated expressions are inserted as text objects. Wolfram's Mathematica, of course, is not the only product that provides a Java Interface. Other algebraic servers like Matlab [W14] or Maple [W15] also provide this feature. We are currently extending the E-Chalk to support other algebraic tools, as well as functional programming languages, such as Haskell or Lisp.

4.4.4 The applet viewer

In order to provide the ability to directly insert Java Applets into the board, the E-Chalk system has to contain its own Applet viewer. The Applet viewer takes a URL, loads the corresponding HTML code and parses it. If it finds an Applet tag, it interpretes it and loads all the classes that are needed to execute the Applet from the Internet. The Applet is executed and displayed on the board. Besides that, all loaded classes are also mirrored, so that when the client later replays the lecture, the classes can be found without problems. When the client later replays the lecture, the Applet is not only displayed but entirely executed again. This means the remote viewer is not only able to watch everything exactly as it was done on the board, he is also able to interact with the Applet himself. Unfortunately, this approach has several downsides. When an Applet is executed

on the board, it gets a thread of control since the Java VM has no resource protection or thread killing mechanisms: there is no way to end a misbehaving or resource consuming Applet. The replaying of Applets on the client side only works if the Applet always behaves equally. If the Applet contains some randomized behaviour, relies on date or time, or depends on the machine type where it is executed, the Applet cannot properly be replayed on the client machine: The Applet may show other than the content described by the lecturer or mouse clicks might just end in nirvana. This can also happen if the remote listener plays around with the Applet while or before the lecturer has worked with it. As explained in section 4.3 Applets also cannot be printed out deterministically. Our approach to solve these problems is to define an Interface called EPM (for E-Chalk pluggable module), which allows the programmer to write his Applet E-Chalk friendly. EPM is described in section 4.8.3.

4.5 Audio

The audio subsystem used for E-Chalk is the pure Java successor of the World Wide Radio system by Gerald Friedland and Tobias Lasser [12], called World Wide Radio 2. World Wide Radio 2 (WWR2) [W16] is a TCP/IP based audio streaming system, that uses lossy compression to achieve interruption free transmission over small bandwidth Internet connections. Besides the two transmission modes that were explained in section 4.1, WWR2 can also operate in near demand mode. The near demand mode is very similar to the live mode. They only differ in the fact that the audiodata is not recorded directly from the sound card but read from sound files stored on the harddisk. The programming of sequences and loops can either be done offline via a configuration file or online via a telnet command line interface. For this purpose the WWR2 server has built-in macro language.

In order to guarantee continuous playing the sound data has to be buffered. The drawback of buffering is that it creates a delay between recording and playback, which is especially unacceptable for bidirectional transmissions (which are not yet implemented in E-Chalk). Our experience showed a buffer of about seven seconds gives the best trade off between transmission latency and interruption resistance.

4.5.1 CODECs

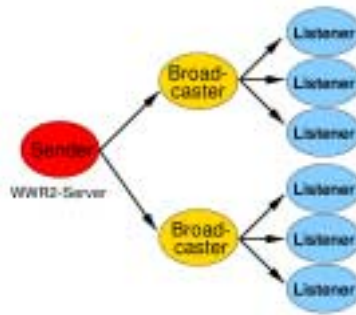
When using pure Java, programs get small and portable. The main benefit is that Java client Applets run in almost any web browser without requiring the user to download any plug-in or to manually install a client program. Updates of the client software can be made transparently, i. e. the user does not have to take any notice of that. In Java, however, the prize for binary portability has to be paid with efficiency and flexibility forfeits. One of this forfeits is that Java Applets running in state-of-the-art web browsers cannot playback sound with a sampling frequency higher than 8999 Hz [W17]. Fortunately, this is perfectly sufficient for the recording of speech. The basic input format for WWR2 is hence 8kHz, 8bits *μlaw* which is defined in [9]. This is the format used for digital telephone lines and hence it requires a connection speed of 64 kbit/s. The first idea to compress these data was to use the low bandwidth perceptual CODEC developed in [8]. However, the adaptive compression that was used there could not be implemented in Java for efficiency reasons. We found that it was not possible to implement even a subset of MPEG audio encoding, since any fast DCT based approach was too slow to run in realtime under Java. As a conclusion we implemented several CODECs for different bandwidth, CPU speeds, and audio quality levels into WWR2 by modifying older compression standards. WWR2 contains a simple and fast 50 kbit/s CODEC, which uses no compression but the Java built-in GZIP³ algorithm. To achieve better compression WWR2 also contains the 4bit version of the *μlaw* CODEC, which is adapted from [9]. This CODEC, together with GZIP, compresses down to 20kbit/s. However, the sound quality is decent. To achieve a good trade-off between sound quality, compression, and execution speed, we modified the ITU ADPCM standard [10]. The result were 4bit, 3bit and 2bit

³Currently, the Java API uses Jean-Loup Gailly's implementation of the GZIP compression [19].

modified-ADPCM CODECs that, combined with GZIP, give an effective average compression of 30 kbit/s, 22 kbit/s, and 15 kbit/s.

4.5.2 Multicasting

Internet multimedia streaming systems have a common problem: They do not scale concerning the amount of users they can serve simultaneously. Several streaming systems use the MBONE [W8] in order to try to get around this hitch. Unfortunately, MBONE needs special router configuration and is not very popular outside the academic domain. Having the structure of MBONE in mind, WWR2 has its own multicast infrastructure, the so-called broadcaster servers. Broadcaster servers get the compressed audio stream from a regular WWR2 server or from another broadcaster and send them to clients or other broadcasters. The result is a tree structure similar to the one that exists in the MBONE.



The benefit of this solution is that bandwidth needs can be distributed. They can be saved where connections are slow and concentrated where connections are fast. The multicasting system of WWR2 is transparent, i. e. the user does not know that he actually listens to a stream which comes from a broadcaster server even though he originally connected to another server. A server automatically forwards clients if the number of simultaneous connections exceeds a threshold. There are two types of broadcaster servers: active servers and passive ones. Active broadcaster servers are always connected to its parent server and get the audio stream, even if no client is currently connected. Passive broadcaster servers only connect to its parent server if and only if at least one client requests audio data. The second method results in a short delay for the first user and leads to problems if the parent connection cannot be established (if this happens, the broadcaster server redirects the client again). In order to become a broadcaster server, only a small Java based daemon program has to be started on a computer that is connected to the Internet. The software can be started as user level program and does not require the intervention of the system administrator. This technique is successfully being used for the web presentation of a company in the entertainment business [W18] and is currently being extended for the entire E-Chalk system.

4.6 Video

The development of the video subsystem, which is called World Wide Video (WWV), was directed by the same idea that underlies World Wide Radio 2: Build a fully featured Internet streaming system that runs on any hardware or platform. Small computers, such as handhelds or mobile phones, are explicitly included. WWV did not only inherit the idea of WWR2, it also inherited the problems. The processing of video data is even more expensive since more information per second accumulates. For that reason we decided to create an asymmetric system in the one meaning, that we assume that the server side has rather unlimited resources, while the client has almost no computational performance.

In contrast to the audio system there is no way to capture video data in pure Java, therefore we use a native interface to the video hardware device. Under Linux we use a native interface to

Video4Linux [W19] and under Windows we use an interface to Microsoft's DirectX 8.0a [W20]. Other operating systems are not yet supported. Of course, WWV also needs a buffering strategy to guarantee continuous operation. As on most client systems the available memory would not last for caching seven seconds of video data, we decided to implement a dynamic cache that increases and decreases with the amount of memory available. Fortunately, uncontinuous image streams are not as disturbing as uncontinuous audio streams.

4.6.1 The CODEC

There are several approaches to build Java based MPEG players [W21, W22]. These players have a high demand on CPU and memory resources and are not suitable for the use in combination with other media, especially when used on small computers. DCT based encoders are not yet implementable in pure Java, due to the reasons explained in section 4.4. For this reason we chose to implement the coder as a native program, too. Fortunately, the JPEG image format⁴ decoding algorithms are part of the Java SDK, which makes it possible to decode JPEG images in an efficient way on the client side. Our train of thoughts thus was to use JPEG as still image compression algorithm and build some lightweight motion compensation around it. Currently we are developing the second version of the video CODEC. The first version, which is the one currently working in the E-Chalk system, is a very simple differential frame coder, which works as follows:

The first frame recorded or sent when a client connects, is an I-Frame (i.e. a full JPEG image). All subsequent frames are, as we call them, transparency frames (T-Frames). T-Frames consist only of those blocks in the picture that have changed significantly. All other blocks are copied from the old picture. A block is a 8x8 pixel matrix. The difference is calculated in the YUV color space. We use the sum over the euclidean distances of the pixels with the Y component weighted more importantly. If the sum exceeds a certain factor above the average of all sums over all blocks, the block is considered as having changed significantly.

This algorithm is interesting because it is very easy and the images are rather self repairing and thus very few I-Frames are needed. In the teaching situation we can find in E-Chalk, where only a small area of the picture (mainly the gestures and mimics of the lecturer) have to be updated, we use no I-Frames. The compression ratio obtained is roughly 40:1. In the E-Chalk system we use a quarter picture of NTSC, that is 192x144, using 4 frames per second to obtain a bandwidth of 64kbit/s.

In the second version of the video coder, we use vector quantization to reduce the variety of blocks. This new type of differential frame does not contain blocks any more, but references to an index in a blockbuffer. By doing this we can save the transmission of blocks that moved from one place to another. The blockbuffer is organised like virtual memory: The least recently used blocks are discarded. However, blocks survive over more than one frame, which improves compression over the simple differential frame. By counting the number of references to a certain block over the time, we can distinguish, which blocks belong mainly to the background and which belong to the foreground. The background ones are the blocks that are referenced more often, while the foreground blocks are referenced less often. Knowing what areas belong to the background helps improving the quality for the background can be displayed with higher detail, but lower update rate and the foreground blocks with lower detail, but higher update rate.

4.7 The Handwriting Recognition

The handwriting recognition (HCR) is a fundamental part of E-Chalk's idea. As said above, it is simply impractical to use a keyboard while giving a lecture. E-Chalk's handwriting recognition is no optical character recognition (OCR) because E-Chalk's HCR does not scan or compare pixel patterns. E-Chalk's HCR uses the dynamic information it gets while a character is written. In other words E-Chalk's character recognition bases on the information how a character has been

⁴Strictly speaking the term "JPEG format" is incorrect. The right term should be JFIF (JPEG file interchange format), whereas the specification can be found in ISO/IEC 10918-1. However, the term "JPEG format" is used more commonly.

Table 1: Classification rates

Classifier	Parameters	Support Vectors	Training error	Class. error	chars/sec
k-NN	k=1	-	-	1.20%	135
k-NN	k=3	-	-	1.83%	130
k-NN	k=5	-	-	3.03%	125
NeuroNet	h=100	-	0.04 ⁵	11.25%	4000
SVM Polynomial	d=1, $\gamma = 10^{-6}$	11.92%	0.89%	2.68%	2900
SVM Polynomial	d=6, $\gamma = 10^{-6}$	11.49%	0.00%	1.09%	1300
SVM RBF	$\gamma = 10^{-5}$	17.61%	0.00%	1.14%	700
SVM RBF	$\gamma = 10^{-6}$	12.59%	0.00%	1.20%	2100

written on the board, not how it looks like (this approach is also known as online recognition). Right now, the second version of the handwriting recognition is being developed. This version includes digits, math operators, and letters. The first version, which is the one currently built into the system, only recognizes digits and math operators. This version is primarily described here.

The HCR system consists of three stages: The preprocessor, the classification system, and the postprocessor. The recognition starts when the lecturer draws in the color, that was configured to be the handwriting recognition color.

4.7.1 Preprocessing

The input for the preprocessor is a sequence of arbitrary length containing x and y coordinates. One sequence corresponds to one linestroke, that is all stylus positions that have been recorded in between the impression on the board and the relocation in temporal order. Given a sequence of linestrokes, the first step the preprocessor has to do, is to decide whether several linestrokes form part of one symbol or not. This is done by calculating the bounding box of each linestroke and then checking for intersections of the bounding boxes. Our experience showed a 50 %-stretching of the bounding box in the both y-directions gives better results, since some people tend to miss crossing points when writing. If several linestrokes are detected to be belonging to the same symbol, the stroke sequences are concatenated. However, the amount of strokes is remembered, as it is a selection criterion for the classification system. The decimal point is already detected at this stage: A point is a sequence with a small cardinality. Then we use the preprocessing algorithm proposed by [12], which uses spatial resampling to transform the input sequences to constant length feature vectors. The output vectors are regularly spaced in arc length and have a constant dimension. Prior to the resampling, the vectors are centered and normalized to make the classification of the symbols invariant of their size and position on the board. Bounding boxes and sizes are memorized as a selection criterion for the postprocessor.

However, this method leads to some problems. If, for example, one writes the digit sequence "2/", people are accustomed to write a very long slash such that it crosses the bounding box of the symbol "2". The system then decides that the sequence is one symbol with two strokes and classifies it as a "4". In the second version, we use the distance between strokes instead of the bounding box. Two strokes belong to the same symbol, if their distance is lower than a certain threshold, which depends on the thickness of the line stroke. To construct the output vector, we use a so called curve evolution algorithm by [13].

4.7.2 Classification System

Table 1 shows a list of the classification methods we tried out for classifying the symbols of the E-Chalk system together with their classification rates.

In this experiment, the training and test sets contained the characters "1234567890+*,/^()" and the termination symbol (see 2.2). The training set contained 7004 vectors and the test set

⁵quadratic error

contained 1752 vectors. The classification speeds were measured on a state-of-the-art PC system, they should only be used as qualitative speed indicators. The first three rows show the results of using the k-nearest-neighbours algorithm using euclidean metric and three different ks. The next row shows the results for a 3 layer neural network. The hidden layer consists of h elements. The last four rows contain the results for our experiments with a support vector machine (SVM). The system is a directed acyclic graph SVM (DAGSVM) as described in [14]. A modification of the Sequential Minimal Optimization (SMO) algorithm [15,16] was used. The kernel function and the regularization parameter are the same for all classification nodes. We tried a polynomial kernel and a kernel bases on radial basis functions (RBF). The polynomial kernel is defined as

$$K(x_i, x) = (\gamma((x_i \cdot x) + 1))^d$$

whereas the RBF kernel is defined as:

$$K(x_i, x) = e^{-\gamma \|x_i - x\|}$$

A more detailed description of the classification system can be found in [17].

Due to the ease of implementation, the actual version of the E-Chalk systems uses a k-NN classifier. The second version of the HCR system uses a support vector machine.

4.7.3 Postprocessing

Postprocessing is used to reclassify symbols that could not be classified by the classification system. This may happen, because the classification system does not know anything about the context in which a symbol appears. For example, it is impossible to distinguish the letter “x” from the multiplication operator, if you do not know the position of the symbol relative to the surrounding symbols. One also knows, that it does not make sense, that a multiplication operator follows a division operator without any symbol inbetween, therefore it makes more sense to classify this last symbol as the letter “x”.

4.8 APIs

In order to be extendable the E-Chalk system provides several interfaces where third party programs can plug in, to cooperate with the E-Chalk system.

4.8.1 Media Applet Synchronization Interface (MASI)

MASI can be used to synchronize several media Applets located in one document to make them cooperate and deliver the same multimedia content. The E-Chalk system uses this to synchronize the audio, video, and board client. The underlying concept of this Interface is the notion of a frame. All methods that address offsets use frames as their basic unit. A frame is a certain amount of time. Frames are atomic in the sense that there are no fractions of frames. Every Applet has to provide methods to convert the abstract unit frame into concrete amounts of time. The amounts of time should be chosen as small as possible to allow fine granular control of the streams. MASI provides around 26 methods for synchronization and stream control, these include methods for pausing, fast-forwarding, and rewinding as well as communication methods. For a detailed description of MASI, see our website [W23].

4.8.2 Media Application Group Interface (MAGIC)

MAGIC is a package that currently contains two interfaces: Launchable and Controllable. Implementing one of these Interfaces allows a Java programmer to Plug In to the E-Chalk wizard. If a class implements Launchable, the wizard is only able to start and terminate execution of this module. The class then behaves as a minor component, as we called them in section 4.1.1. Controllable is the Interface for major components, that means, Controllables can be paused, fast-forwarded, rewinded, and so on. This allows a user to rerecord certain areas to fix errors. These

interfaces are currently under development, but further information can also be obtained at our website [W24, W25].

4.8.3 E-Chalk Pluggable Modules (EPM)

E-Chalk pluggable modules are regular Applets that behave “E-Chalk friendly”. The EPM Interface enables Applets to write on a transparent screen, to be in a deterministic printable state, to know whether a lecturer or a remote viewer uses the Applet. EPMS can also be killed or paused to save CPU resources. More information about our EPM Interface can be obtained at our website [W26].

5 Summary and Perspective

The E-Chalk system is a software system that tries to enhance classroom teaching by combining all the advantages of the traditional chalkboard with the functionality of modern multimedial presentation tools. The lecturer can insert images, applets, function plottings, as well as evaluate formulas. E-Chalk assumes a wide touch sensitive display with such a great luminosity and contrast that it can be used in big lecture rooms. Such devices are not yet affordable, though they will be in the near future. E-Chalk lectures are transmitted live into the web. The user needs no plug-in or manual installation of any kind, since the E-chalk client system is Applet based. E-Chalk transmits the dynamic creation of the board image, together with audio. A video stream is optional. The E-Chalk system also archives lectures into a directory, which can easily be exported to the web or inserted into a database system. The E-Chalk system can easily be extended through several APIs, which we are currently expanding.

In the future, E-Chalk will be able to evaluate complex mathematical formulas. The keyboard will be entirely substituted by pervasive handwriting recognition, which will also make it possible to index lectures, insert Applets and images, and will enable the lecturer to handwrite small programs that are interpreted and executed directly on the board. The handwriting will automatically be smoothed, so that lectures become more pleasing to the viewer. Graphical primitives will speed up the production of graphics and sketches. We will add an editing system to the E-Chalk project, so that lecture can be fully post-edited.

Credits

The following people mainly contributed to the E-Chalk project:

Prof. Dr. Raúl Rojas had the idea for the E-Chalk system and is head of the project. He is professor for artificial intelligence, acting head of the Center for Digital Media, and former director of the department of computer science at the Freie Universität Berlin (FU Berlin).

Dipl.-Inform., Dipl.-Math. Lars Knipping is researcher at the Center for Digital Media of the department of computer science at FU Berlin. He developed the Applet viewer and the algebraic server interface of the board server. He did the main work on the GUI and redesigned the architecture of the board server entirely.

Gerald Friedland is a student of computer science at the FU Berlin. He contributed the audio system to E-Chalk, which he had developed together with Bernhard Frötschl. He also contributed the video system and the E-Chalk wizard and implemented the first version of the handwriting recognition. Together with several other students, he also developed E-Chalk’s database connectivity.

Dipl.-Inform. Ulrich Raffel is researcher at the FU Berlin. He developed a first version of the board server as diploma thesis [W27].

Ernesto Tapia, M. Sc. is a doctorate student at the FU Berlin and is currently developing the second version of the handwriting recognition. His experiments with support vector machines led to the experimental data in table 1.

The E-Chalk project also benefited from the experience and the ideas of several people from the entire department. Especially we would like to thank the following people for their contributions:

Dipl.-Inform. Sven Behnke provided helpful tips and tricks about image processing that helped during the development of WWV.

Mary Ann Brennan is a student at Stanford University, California. She contributed the PDF converter while she had a 3 month internship with the FU Berlin.

Dipl.-Inform. Margareta Esponda, M. Sc. build the Mathematica panel.

Karsten Flügge wrote a first version of the console.

Dipl.-Inform. Bernhard Frötschl implemented the CODECs for the WWR2 and helped to correct several articles and documentation on the E-Chalk project.

Tobias Lenz programmed the native video interface for Windows.

Dipl.-Met. Christian Zick provided us with helpful practical hints on video streaming and image editing.

We also would like to thank the 200 students that supported E-Chalk during the first test lecture in the winter term 2000.

References

- [1] R. Rojas et al, "Ende der Kreidezeit - Die Zukunft des Mathematikunterrichts", DMV Mitteilungen, Berlin, 01/2001.
- [2] R. Rojas, L. Knipping, G. Friedland, U. Raffel, "Elektronische Kreide: Eine Java-Multimedia-Tafel für den Präsenz und Fernunterricht", Proceedings of LearnTEC Conference, Karlsruhe (Germany), 2001.
- [3] M. Weiser, R. Gold, J.S. Brown, "The origins of ubiquitous computing research at PARC in the late 80s", IBM Systems Journal, Vol. 38, No. 4, 1999.
- [4] Eric Saud, "Image Mosaicing and a Diagrammatic User Interface for an Office Whiteboard Scanner", Technical Report, Xerox Palo Alto research Center, 1999.
- [5] Hiroshi Ishii and Brygg Ullmer, "Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms", Proceedings CHI-97, 1997.
- [6] Armando Fox, Brad Johanson, Pat Hanrahan, and Terry Winograd, "Integrating Information Appliances into an Interactive Workspace", IEEE CG&A, May/June 2000.
- [7] Ch. Bacher, R. Müller, Th. Ottmann, M. Will: "Authoring on the Fly. A new way of integrating telepresentation and courseware production", Proceedings ICCE '97, Kuching, Sarawak, Malaysia, December 1997, pages 89 - 96.
- [8] G. Friedland and T. Lasser "World Wide Radio - Audio Live Übertragung durch das Internet", Bundeswettbewerb Jugend forscht, München, Germany, 1998.
- [9] Recommendation G.711, International Telecommunication Union (ITU, former CCITT)
- [10] Marc Nelson, "Datenkomprimierung - Effiziente Algorithmen in C", page 309, Heise Verlag, Hannover, 1993.

- [11] Recommendation G.726, ITU
- [12] F. Alimoglu, E. Alpaydin, "Methods of Combining Multiple Classifiers Based on Different Representations for Pen-based Handwriting Recognition," Proceedings of the Fifth Turkish Artificial Intelligence and Artificial Neural Networks Symposium (TAINN 96), June 1996, Istanbul, Turkey.
- [13] L. J. Latecki, R.-R. Ghadially, R. Lakämper, and U. Eckhardt: Continuity of the discrete curve evolution. *Journal of Electronic Imaging* 9 (3), pp. 317-326, July 2000.
- [14] J. C. Platt, N. Christiani, and J. Shawe-Taylor. Large margin DAGs for multiclass classification. *Advances in Neural Information Processing Systems*, 12:547-553, 2000.
- [15] J. C. Platt. Fast training of support vectore machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 185 - 208, Cambridge, MA, 1999. MIT Press
- [16] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. technical Report, Dept of CSA, IISc, Bangalore, India, 1999.
- [17] E. Tapia, R. Rojas, "Recognition of Handwritten Digits in the E-Chalk System using Support Vector Machines", Freie Universität Berlin, June 2000, to appear.
- [18] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee, "Hypertext Transfer Protocol HTTP 1.1", Network Working Group, RFC 2068, January 1997.
- [19] P. Deutsch, "GZIP file format specification version 4.3", Network Working Group, RFC 1952, May 1996.
- [W1] E-Chalk, The Electronic Chalkboard:
<http://www.echalk.de>
- [W2] Mathematica by Wolfram Research, Inc:
<http://www.wolfram.com>
- [W3] NEC presentation products:
<http://www.nectech.com/presentationproducts/>
- [W4] Wacom, Inc:
<http://www.wacom.com>
- [W5] Wacom's LCD tablets:
<http://www.wacom.com/lcdtablets/>
- [W6] Numonics, Inc:
<http://www.numonics.com>
- [W7] Course 19520, Neural Networks by Raúl Rojas, FU Berlin Summer 2001:
<http://www.inf.fu-berlin.de/lehre/SS01/NeuronaleNetze>
- [W8] MBone (or IP Multicast) Information Web:
<http://www.dante.net/mbone/>
- [W9] Virtual Ink Mimio - turning your whiteboard into a virtual whiteboard in seconds:
<http://www.mimio.com>
- [W10] The DigiWB project - a GNU/Linux driver for Mimio, E-Beam, and IntelliBoard:
<http://mimio.spline.inf.fu-berlin.de>

- [W11] eBeam Electronics for Imaging:
<http://www.e-beam.com>
- [W12] Smart Technologies, Inc:
<http://www.smarttech.com>
- [W13] Mathematica JLink Interface:
<http://www.wolfram.com/solutions/mathlink/jlink/>
- [W14] Matlab by The Mathworks, Inc:
<http://www.mathworks.com>
- [W15] Maple by Waterloo maple, Inc:
<http://www.maplesoft.com>
- [W16] WWR2 a totally Java based Internet audio live and ondemand streaming system:
<http://www.javaradio.de>
- [W17] Sun Java 1.1 API Documentation:
<http://java.sun.com/products/jdk/1.1/docs/api/packages.html>
- [W18] Berliner Gruselkabinett Entertainment GmbH:
<http://www.gruselkabinett-berlin.de>
- [W19] Video 4 Linux (2):
<http://www.thedirks.org/v4l2/>
- [W20] Microsoft's DirectX 8.0a:
<http://www.microsoft.com/directx/>
- [W21] Inline MPEG - 1 - player in JAVA (with MPEG layer I decoder):
http://rnvs.informatik.tu-chemnitz.de/~jan/MPEG/MPEG_Play.html
- [W22] Sureplayer an open source Java based MPEG 1 player:
<http://www.sureplayer.org>
- [W23] Media Applet synchronization Interface:
<http://kazan.inf.fu-berlin.de/echalk/docs/MASI/masi/MASI.html>
- [W24] MAGIC Launchable:
<http://kazan.inf.fu-berlin.de/echalk/docs/MAGIC/MAGIC/Launchable.html>
- [W25] MAGIC Controllable:
<http://kazan.inf.fu-berlin.de/echalk/docs/MAGIC/MAGIC/Controllable.html>
- [W26] E-Chalk pluggable modules Interface:
<http://kazan.inf.fu-berlin.de/echalk/docs/EPM/EPM/epm.html>
- [W27] Ulrich Raffel's Diploma thesis:
<http://www.inf.fu-berlin.de/~raffel>