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. Our philosophy is to provide a tool which enhances classroom teaching and provides distance education as a side effect, i. e. at no extra cost. E-Chalk is based on the blackboard metaphor, which has proven to be an ideal teaching tool since the Ancient Greeks. The software has been entirely written in Java, so that Internet listeners are not forced to install a plug-in to be able to access the lectures. This article provides an introduction to the ideas behind our approach, a basic overview of the features of the software system, and a technical description of E-Chalk.

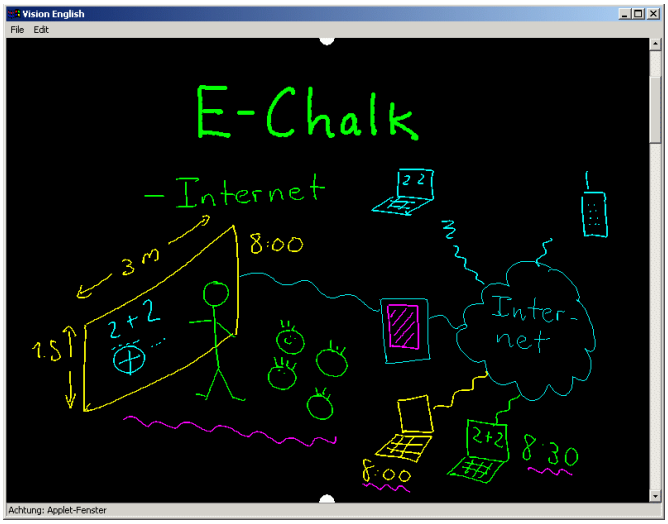
Contents

1 Introduction

The chalkboard is one of the earliest teaching tools in history and still the preferred exposition medium in many scientific disciplines. Mathematics teachers in particular appreciate the chalkboard for its naturalness and simplicity. Chalkboards are easy to handle and have good "imaging" features. Even when used in huge lecture halls, the chalkboard has enough contrast so 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. In our opinion, 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 being 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 are problematic for lectures that deal with rather complicated content and formulas. However, slide presentations allow a better visualization 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 recalling later the lecture. Many students complain about not being able of copying the board image to their notebooks while they also try to understand the lecture. Naturally, this also leads to mistakes.

Besides these problems, distance learning has been gaining 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 (one exception is the MS Multimedia Presentation Manager). 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 audio track. However, this approach does not improve classroom teaching and since it requires a high bandwidth Internet connection the videos usually cannot easily reach every student's home.

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



2 The E-Chalk System

2.1 From Desktop to Blackboard

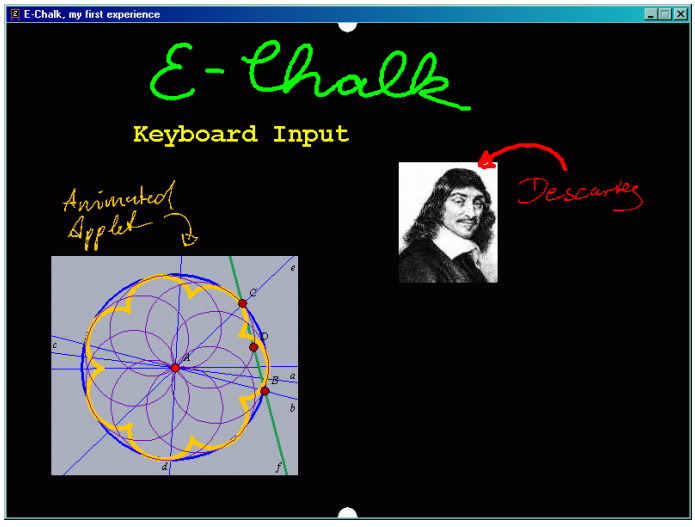
The main idea of E-Chalk is to simulate a chalkboard with a touch sensitive computer screen. After starting E-Chalk a wizard allows the user to configure some properties of the lecture and the system's user interface metaphor changes from a computer desktop to a blackboard.

First of all, the computer screen becomes a visual output tool for more than one person. Everything that is shown on the display can be watched by many persons and must thus be understood by them. Secondly, the keyboard loses its importance and needs to be used only occasionally. Our goal is to avoid using the keyboard completely. Furthermore, the mouse is not an adequate 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 was not easy. We tried several devices discussed in section 2.3. Ideally, we would like to have a really large screen (e.g. 2.5×3.0 meters) that is touch sensitive. This screen must have a high resolution concerning both the amount of displayable pixels and the amount of contact sensitive dots per inch. The display must offer 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

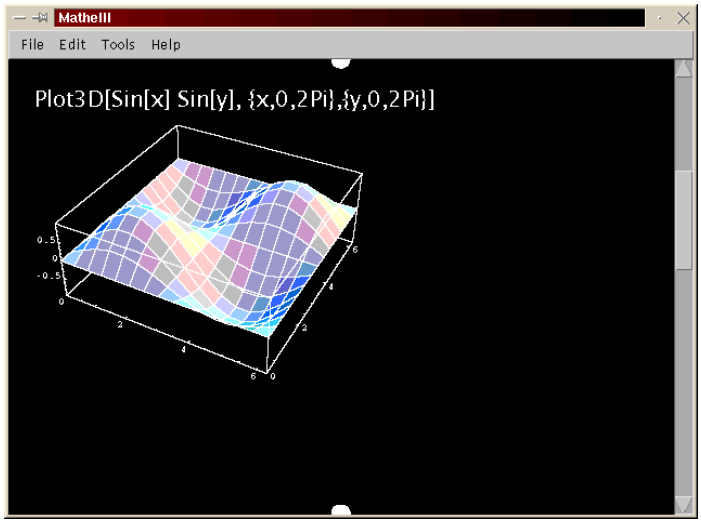
The software transforms the screen on a black surface where you can paint using different colors and pen thicknesses. The board can be scrolled up and down vertically, providing the lecturer an unlimited surface to write on. The user can also use an eraser to delete part of or the whole board content.

The system allows the user to paste images from the local hard drive or from the Internet. These images can be loaded directly from the web just entering their URL during the lecture. Even better, they can be loaded using bookmarks prepared prior to the lecture. Images pasted onto the board can be annotated and edited just as if they were paper pictures stuck to a traditional blackboard. This also creates the possibility of inserting and annotating legacy slides, so that old lectures can be reused and updated. If animations or interactive programs are desired, E-Chalk allows the user to paste Java Applets to 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 selecting from a previously defined list of bookmarks.

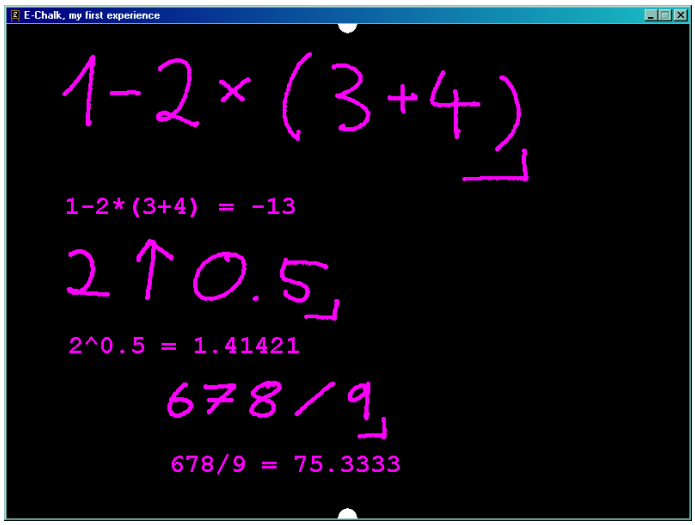


E-Chalk also allows the integration of an algebraic server system, such as Mathematica [W2]. The user can evaluate expressions and plot functions directly on the board. Function plots are

treated like images: They can be placed on the board anywhere and in any color. After a function plot has been pasted it can be annotated and edited. Mathematical expressions are evaluated as text objects.



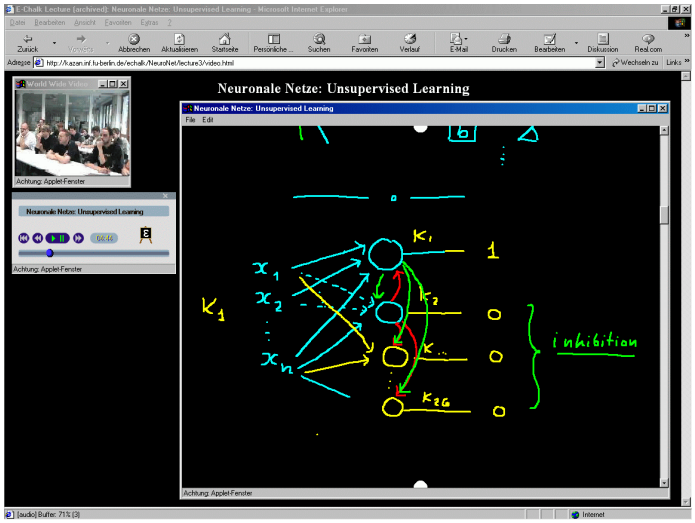
In the preliminary versions of E-Chalk, the lecturer was forced to use the keyboard to enter all commands for the algebraic server. However, this was not ergonomic and did not reflect the idea of E-Chalk. To provide a better way of interaction we are implementing handwriting recognition. So far, the system recognizes digits, the basic mathematical operators (plus, minus, product, divide, and power), the decimal point, and parenthesis. We are still working on a full alphabet handwriting recognizer.



In order to signal the system that the symbols written on the board should be recognized, one color is defined as the handwriting recognition color. The lecturer writes as he would do on a traditional blackboard. If he makes a mistake, he can erase the digits with the eraser. To close the expression to be recognized, the lecturer writes a special ending symbol: A bottom left corner symbol (a vertical stroke that goes down, stops, and goes horizontally to the left).

Everything that appears on the board is automatically transmitted by E-Chalk over the Internet. Remote users can connect to the computer where the E-Chalk system is running and can view everything as seen in the classroom. Remote listeners receive the audio signal and hear the lecturer and optionally a small video of the teacher. The small video is not transmitted as

content related information, but to provide an impression of the classroom in order to achieve psychological closeness to the lecturer.



Since video streams consume CPU and connection resources, the client can close the video stream at any time in order to save bandwidth and processor time. The connection speed needed for a complete E-Chalk lecture with blackboard image, audio, and video is roughly 128kbit/s (that is the speed of two B-channels ISDN). Without the video stream the required connection bandwidth does not exceed 64kbit/s. Users do not require 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 provides all resources needed to follow lectures remotely. That includes the mirroring of Applets (a forthcoming feature) 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 held with the E-Chalk system are not only transmitted live, they are also automatically stored for posterior on-demand access. As with live lectures, all resources are installed automatically and the remote user only has to access the created webpages using a web browser. When viewing archived lectures the remote user sees a control console that enables him to regulate the content flow as done with a VCR, i. e. pausing, fast-forwarding, and rewinding. One advantage of this simple directory-based storage of lectures is that they can be easily packed into one file for download - a feature which many students use to save connection costs when they want to watch a lecture at home. However, the simple directory based storage of lectures also led to some problems 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 also discussed in section 4.2.

Last but not least, archived lectures are automatically converted to Adobe's well known PDF format. This enables remote users to print a transcript 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 all 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 have enough resolution. Unfortunately, plasma screens also have several disadvantages. For 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 diagonal length for big lecture rooms. Several manufacturers, such as NEC Inc., provide systems that arrange four plasma displays into one large screen [W3]. However, they do not provide a touch screen overlay yet, since such large systems are mainly used for trade fair

presentations. Finally, plasma displays are quite susceptible to screen burn-in, which restricts their use to animated presentations. Our first experiment with a 43" plasma display from Fujitsu was not very satisfactory because the touch screen overlay's resolution was insufficient for drawing. The resolution was only usable for standard desktop operations, such as clicking on icons or dragging objects from one window to another.

Our next idea was to use the Mimio ultrasound pen tracking 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 Mimio 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 the display of the line on the screen was so large, that the final combination was unusable.

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 produces a voltage at the point were it is in contact with the tablet. The position is tracked and reported to the computer. The user can therefore write as he would do on a sheet of paper and the LCD projector shows the image.

The disadvantage of this approach is its "look and feel": It differs remarkably from the handling of a traditional blackboard. This is especially noticeable since there is no visual feedback from the tablet. The lecturer has to adapt to looking on the screen and write on the tablet. This can be done in a few minutes though. Wacom, Inc. also provides digitizer tablets with built-in LCD display [W5] to get around this problem. However, in this systems there is a short visual distance between the display and the stylus (parallax error). Therefore such tablets also require some training and adaption from the user. In general, this approach is an acceptable solution because digitizing tablets are supported in most operating systems.

The next alternative are electronic whiteboards. Numonics, Inc [W6] and Hitachi provide

front projection electronic whiteboard systems called “Interactive Presentation Manager” and ”Starboard”, respectively. They are very similar to the one described in section 3.2.3. The main virtue of such whiteboards is that they are very similar to the traditional blackboards. The lecturer stands in front of the audience and writes directly on the board while the audience watches the lecture looking at it. The whiteboard has a resolution of more than 300dpi and uses a magnetic stylus with a small rechargeable battery. We are currently using several of such front projection whiteboards as this solution comes closest to our basic idea. Yet, this approach has disadvantages, too. Although a 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 (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 lecture projects a shadow on the whiteboard which can be annoying.

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 at our university. The lecture hall was filled up with 200 students and Professor Rojas used an LCD projector in connection with a digitizer tablet. The lecture was archived and can be watched at our home page [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 neural networks course, whose lectures can be found at the website of our institute [W1]. Roughly 20 students attended the course and we used a whiteboard-projection combination as 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 favorized downloading the packed versions of the archived lectures in order 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, a joint course between the Freie Universität and Stanford University. In this 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 reorient from the traditional chalkboard to the E-Chalk system.

3 Related Projects

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

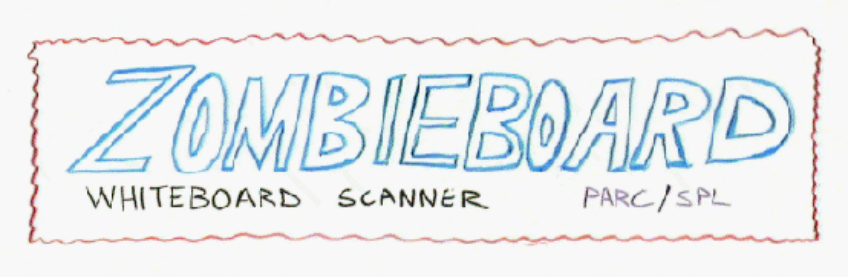
3.1 Academic Projects

3.1.1 “Ubiquitious Computing” at Xerox Parc

At the end of the 1980s, computer scientists at Xerox Parc already thought of the possibility of using wide LCD monitors as a kind of whiteboard. Their so called “live board” was targeted to the Internet in order 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 use of this project was to enhance computer supported collaborative work via small notepads in which the participants could write their comments on - it was not targeted to classroom teaching. The idea survived in today’s collaborative whiteboards.

3.1.2 The “Zombieboard”

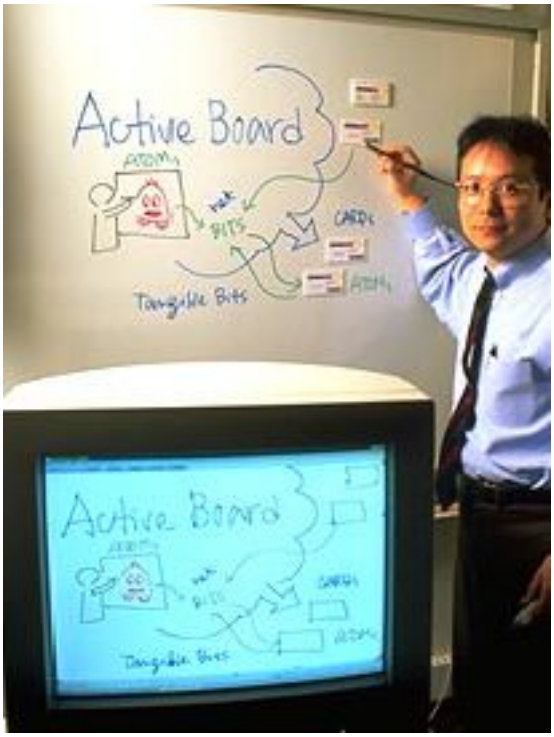
Another project at Xerox Parc was the so called “Zombieboard” [4]. It consists of a plastic whiteboard that is scanned by a camera system. The images are analyzed by the software to cut out the lecturer from the image. Only the contents of the board is transmitted through the Internet.



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

3.1.3 Transboard

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



MIT's approach is similar to the E-Chalk system, but it is mainly targeted to collaborative work rather than teaching. The main idea of the Transboard is to store objects (sets of linestrokes) in physical maps which can be read and interpreted by the computer. The main use of the system is for capturing brain storming sessions in front of the whiteboard. Remote users are able to participate in the discussions. Handwriting recognition or the use of interactive software do 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.

Winograd's 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 software.

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 into 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 two German 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.

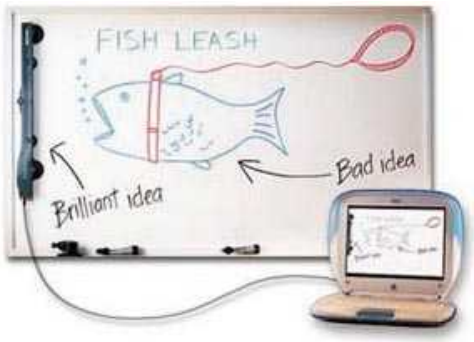
Though the initial idea to enhance classroom teaching through an electronic board and store the lecture for later use in distance learning applications is the same, both systems are rather different. First, E-Chalk does not require the lecturer to prepare postscript slides and it also does not require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

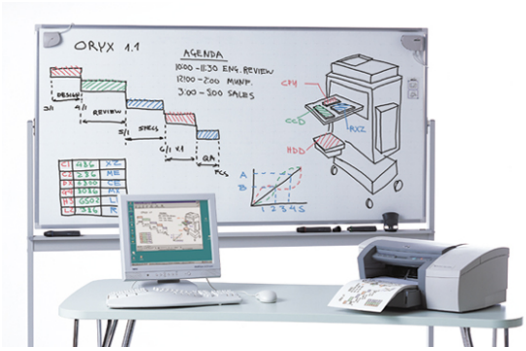
Virtual Ink Cooperation [W9] provides a hardware software combination called Mimio. It captures whiteboard images and transmits them over the net.



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 a 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, eBeam provides 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 more quickly. This is achieved through sensors which are not fixed to a vertical capture bar. There are two separate sensors, one of them is attached to the upper left corner and the other 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 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 main disadvantage of this system is its physical volume and its weight.

The second version is a front projection system whose surface is also 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 projects a shade on the board over the stylus (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 large lecture rooms.

3.3 Conclusion of the comparison

Most of the projects and products mentioned above put the accent on collaborative work. Especially commercial systems seem to focus on mobile collaborative Internet meetings, rather than on classroom teaching. The resulting software systems hence stress one-to-one or many-to-many communication schemes (rather than one-to-many) which require more bandwidth and peer computers at each end of the communication channel. Nevertheless, as described in section 2.3, we could successfully combine some of those hardware products with the E-Chalk software system.

4 Architecture and Implementation of the E-Chalk System

4.1 Overview

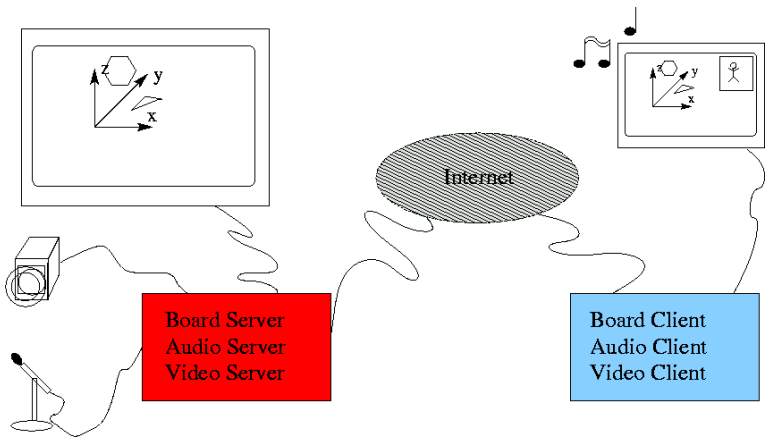
4.1.1 The server

The server software currently consists of three main and several minor components. The three main components are the audio, the video, and the board server. Minor components are the database manager and the PDF converter. All server components are managed by the E-Chalk Startup Wizard. The wizard is the first program executed when a user starts the E-Chalk server: The user is first prompted to configure the system. The E-Chalk wizard then starts the main components and controls them in several ways, depending on its type (the idea is similar to an operating system kernel handling processes). Minor components can only be started and terminated, while major components can be controlled in a way similar to a VCR (see section 4.8.2 for details). This enables the user to pause during a live lecture or to correct mistakes recently made. All server components report errors during a live lecture (on a hidden control window). After the live transmission of a lecture is finished, the E-Chalk wizard makes sure that all components needed to view the lecture are archived in the output directory. This output directory can be accessed 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 that puts the lecture in a central database (see 4.2).

4.1.2 The client

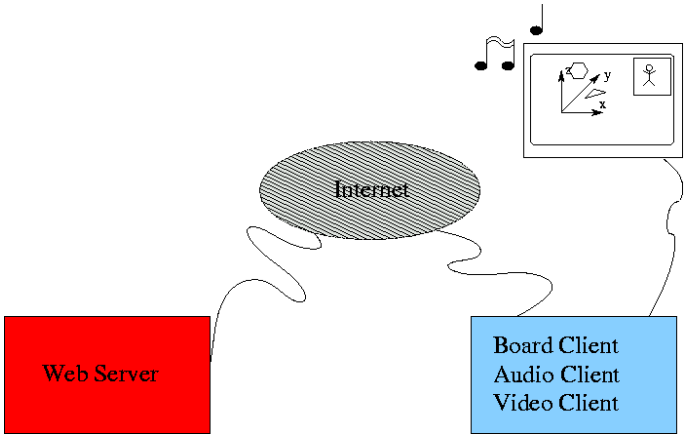
The E-Chalk client system consists of a set of independent receiver Applets that synchronize themselves by communicating through our Media Applet Synchronization Interface (MASI). See section 4.8.1 for further details on this interface. Additionally, there is a console Applet, which allows operations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

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



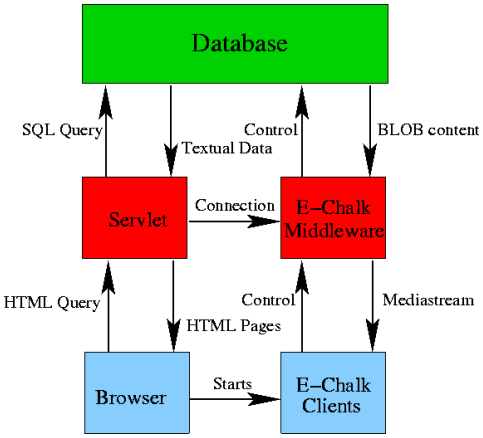
As explained above, E-Chalk has basically two ways of operating: on-line for live transmissions and off-line for archived transmissions. During live transmissions the audio stream, video stream, and board events are recorded from their devices, compressed, and sent, then they are received, uncompressed, and replayed in real time. While this is done, everything that is sent out is also stored in files. As a result, 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 and no E-Chalk server is needed.

The following diagram shows how archived lectures are transmitted.



4.2 Database Connectivity

The architecture mentioned above is simple and useful for archiving lectures. A disadvantage is that a lecture is stored on the harddisk of the computer used to show the lecture. If it is wished to store them in a more centralized manner, e.g. to published them on a common single webserver, this has to be done manually. As explained above, every lecture is stored in a directory, and this means that every lecture contains all the resources needed for remote listening. The disadvantage is that if, for example, there is a new version of the client, each directory has to be updated. The third problem has to do with a technical efficiency that arises with fast-forwarding and fast-backwarding of lectures: in order to skip parts of a lecture the stream has to be reloaded entirely until the point where playback should continue[1].



To solve this problem, we experimented with the storage of E-Chalk lectures in a database

[1]Even though it was already specified for HTML 1.1 [18], no http server daemon known to us implements the GET command with the ability to specify a start offset.

system. The actual lecture content is stored as binary large objects (BLOBs) in the database. Additionally, the database contains tables for user administration, revision control of lecture resources, and for editing lecture metadata. Our idea is to integrate in the future an E-Chalk server middleware into the standard three tier architecture suggested by the Java Servlet API. An overview of the planned architecture is shown in the figure above.

On the left half of the diagram 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 a search engine prompts the user to search for lectures according to different criteria, e.g professor, topic, or some keyword that has been defined for the lecture. The search, together with some other administrative tasks, is done through the communication between webbrowser, Servlet, and database. When a user finally selects to listen to a lecture, the right half of the diagram comes into play. The Servlet communicates with the E-Chalk server middleware and sends it a message, announcing that in a few seconds a client with a certain ID will connect. Thereafter, the middleware begins to retrieve the lecture from the database. At the same time, the Servlet expands and sends the needed HTML pages and other resources to the client. The webbrowser of the client displays the webpage and downloads the proper E-Chalk client Applet, which connects itself to the middleware. The middleware begins then to send audio, video, and board event streams to the client.

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

Administration of content already present in the database is done through a web interface, while inserting content is controlled through the E-Chalk startup wizard. E-Chalk lectures can be inserted from any host in the Internet. As a result, this tool is helpful for centralizing lectures. When there is a new client version, lectures that have version stamps declaring them compatible with a new client are automatically updated, so that users always use the newest client possible. As stated before, there is an experimental version of this database which will become available in a future release of E-Chalk.

4.3 Printing lectures

Adobe's Portable Document Format (PDF) is more than capable of representing all of the image 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 captured from it and paste 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 on older 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. Right now we break the page when it is full, but a better approach would be to determine which horizontal spacing represents a line break in the lecture and use it for a page break.

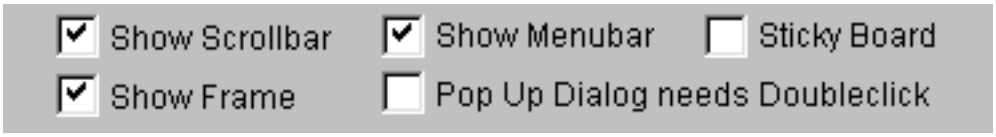
4.4 Board

In line with the analogy that the wizard we use can be compared to an operating system kernel, the board server can be compared to 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 most pervasive 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 module is described in section 4.7 of this document, while the other modules are described in the following paragraphs.

4.4.1 The GUI

As explained above, the metaphor that underlies our GUI is a chalkboard rather than a desktop. As a matter of fact, we tried not to stick to old desktop metaphor GUI elements, such as menu bars, scroll panels, or text input fields. There are two reasons for still using those elements at certain places: Either we have not implemented an adequate solution yet, or we kept the traditional elements to make it possible for the user to work with them as long as he has not yet become used to the new ones. In the latter case, however, we make the traditional element optional.



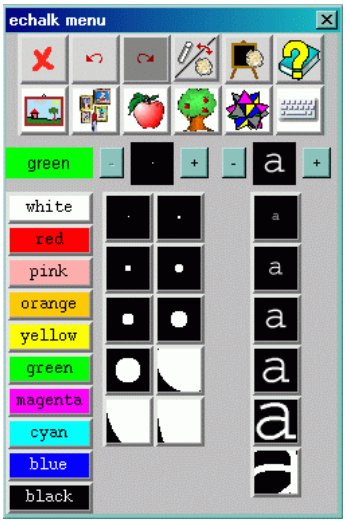
It is very important that no GUI element disturbs the flow of the lecture. Interference with the lecture flow would lead to inattentive students. Another goal, therefore, was to avoid anything that surprises people, like arbitrarily popping up dialog boxes. As described in 2.1 we want to make it possible to reach any feature of the board with the 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 contraintuitive or surprising GUI components: the attention of the audience is lost.

Our solution to approach the goal stated above was to keep the GUI simple. The basic work environment is a featureless 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 to do this.

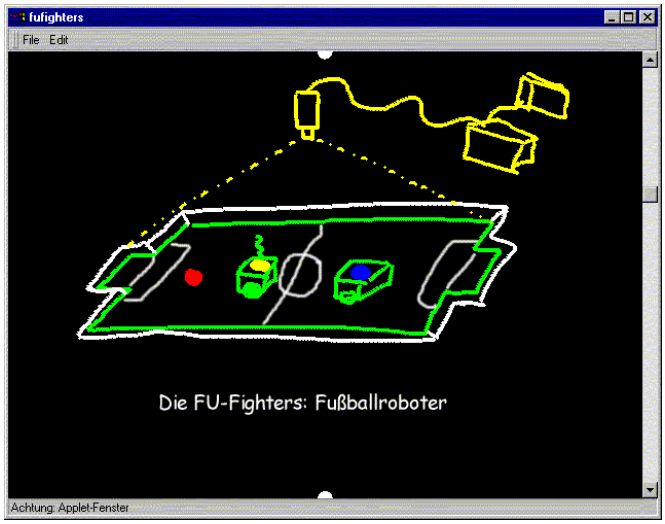
Changing to the eraser works in the same way, since nothing is more annoying than a long and complicated procedure to erase a mistake. However, we noticed that the virtual eraser is not as effective as the traditional one, i. e. it takes longer to erase something with a virtual eraser than it would take with a traditional one. Therefore, we implemented an undo buttons which erases the last linestroke (a redo button restores it). Mapping one function button (from a keyboard or whiteboard) to this operation results in a very efficient error correction tool. For more complex operations we choose to design a toolbox. The toolbox can be opened by clicking the button on the stylus (which maps to 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, swith between eraser and pen, 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 accidentally opening the toolbox, the toolbox can be configured to open only after 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 eraser. 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 in order to avoid having to type an URL 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. In the future 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.



We found that scrollbars are not the optimal way to scroll a blackboard screen and therefore

we provide two special scroll points. The scroll points are located at the top and at the bottom of the screen. To scroll, the lecturer clicks on a scroll point and drags the stylus in the direction he wants to scroll. The movement is similar to the one that is done when a person pulls a sliding blackboard up or down.

4.4.2 The network server

The GUI event handler records all events that occur on the board and passes them to the network server which sends them out to the Internet, recording them also 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 receives the entire board image. Even though the format we are currently using contains no compression, the bandwidth needed is only 3 kbit/s. The latest version of our 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 use with Mathematica from Wolfram Research [W2]. To communicate with Mathematica we use the JLink interface [W13]. Images that result from graphical commands, like function plot, are saved and inserted as GIF images, while mathematical expressions are inserted as text objects. Mathematica, of course, is not the only product that provides a Java Interface. Other algebraic servers such as Matlab [W14] or Maple [W15] provide this feature too. We are currently extending 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 paste Java Applets to the board, the E-Chalk system has to contain its own Applet viewer. The Applet viewer takes an URL, loads the corresponding HTML code and parses it. If it finds an Applet tag, it interprets 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. 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 disadvantages. 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 in the same way in both machines. If the Applet contains some randomized behavior, 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. When EPM becomes available and is used by programmers we will activate the Applet features in commercial versions of E-Chalk. At the moment Applets are only available in experimental versions of E-Chalk.

4.5 Audio

The audio subsystem used for E-Chalk is the pure Java successor of the World Wide Radio system written by Gerald Friedland and Tobias Lasser [8], called World Wide Radio 2. World Wide Radio 2 (WWR2) [W16] is a TCP/IP based audio streaming system, that uses lossy compression to

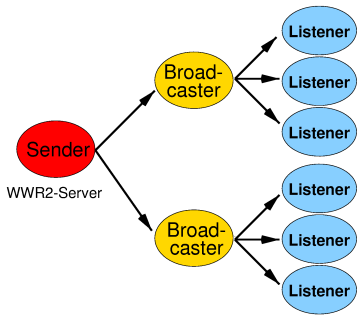
achieve interruption free transmission over small bandwidth Internet connections. Besides the two transmission modes that were explained in section 4.1, WWR2 can also operate in near demand mode. The near demand mode is very similar to the live mode. They only differ in the fact that the audio data is not recorded directly from the sound card but is read from sound files stored on the harddisk. The programming of sequences and loops can either be done off-line via a configuration file or on-line via a telnet command line interface. For this purpose the WWR2 server has a 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 has shown that a buffer of about seven seconds provides the best trade-off between transmission latency and interruption resistance.

4.5.1 CODECs

When using pure Java, programs are 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 know about new releases. In Java, however, the prize for binary portability is paid with efficiency and flexibility penalties. One of this penalties 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 sufficient for recording 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 GZIP[2] 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. The sound quality is adequate. 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 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



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

Internet multimedia streaming systems have a common problem: They do not scale according to the amount of users they can serve simultaneously. Several streaming systems use the MBONE [W8] in order to avoid this difficulty. Unfortunately, MBONE needs special router configurations 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 to the entire E-Chalk system.

4.6 Video

The development of the video subsystem, which is called World Wide Video (WWV), was guided 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 explicitely 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, i.e. we assume that the server side has rather unlimited resources while the client has low 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. Since in 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, discontinuous image streams are not as disturbing as discontinuous audio streams.

4.6.1 The CODEC

There exist several approaches for building Java based MPEG players [W21, W22]. These players require fast CPUs and large memory resources and are not suitable for 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 format[3] 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 a still image compression algorithm and

[3]Strictly 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.

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 described next.

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 euclidian distances of the pixels with the Y component having more weight. If the sum exceeds the 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 of its simplicity; 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 192×144, using 4 frames per second to obtain a bandwidth of 64kbit/s.

For the second version of the video coder, we implemented 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 organized like a 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 blocks are those that are referenced more often, while the foreground blocks are referenced less. Knowing what areas belong to the background helps to improve the quality, because the background can be displayed with higher detail and lower update rate, while the foreground blocks are displayed with lower detail but higher update rate.

4.7 Handwriting Recognition

Handwriting recognition (HCR) is a fundamental part of E-Chalk's vision. 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 is based on the temporal information recorded when writing (this is also known as on-line handwriting recognition). The second version of the handwriting recognizer includes digits, math operators, and certain letters.

The HCR system consists of three stages: The preprocessor, the classification system, and the postprocessor. The recognition starts when the lecturer draws using the color 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 between the first contact with the board and the relocation. Given a sequence of linestrokes, the first step for the preprocessor, 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 between them. Our experience showed that a 50 %-stretching of the bounding box in the y-direction gives better results, since some people tend to miss crossing points when writing. If several linestrokes are detected to belong 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

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[4]	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

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 resampling, the vectors are centered and normalized to make the classification of the symbols invariant to their size and position on the board. Bounding boxes and sizes are memorized as a selection criterion for the postprocessor.

However, this method leeds to some problems. If, for example, one writes the digit sequence "2/", people are used 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 used 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 contained 1752 vectors. The classification speeds were measured on a state-of-the-art PC system; they should only be seen as qualitative speed indicators. The first three rows show the results of using the k-nearest-neighbors algorithm using the euclidian metric and three different k's. 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 first version of the E-Chalk systems used a k-NN classifier. The current version of the HCR system uses a support vector machine.

[4] quadratic error

4.7.3 Postprocessing

Postprocessing is used to reclassify symbols that could not be correctly classified. 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 context symbols.

4.8 APIs

In order to be extendable, the E-Chalk system provides several interfaces for third party programs to plug-in in order 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

E-Chalk is a software system that tries to enhance classroom teaching by combining all the advantages of the traditional chalkboard with the functionality of modern multimedia presentation tools. The lecturer can paste images, Applets, function plots, as well as evaluate formulas on-line. The E-Chalk vision is to use a a large touch sensitive display with good luminosity and contrast, so that it can be viewed in large lecture rooms. Such devices are not yet affordable, though they will be in the near future. E-Chalk lectures are transmitted live over the Internet. The user needs no plug-ins or special programs since the E-chalk client system is Applet based. E-Chalk transmits the dynamics 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 handwritten 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. 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 already have an experimental version of an editor for E-Chalk files, so that lectures can be fully post-edited.

Credits

The following people contributed to our E-Chalk project:

Prof. Dr. Raúl Rojas conceived 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 chairman 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 recognizer. 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 PhD student at the FU Berlin and is currently developing the second version of the handwriting recognizer. 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, Hannovver, 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 Sy stems, 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