

Interaktiver Formeleditor mit GeoGebra-Anbindung für digitale Whiteboards

CHRISTIAN GROSSAUER

DIPLOMARBEIT

eingereicht am
Fachhochschul-Masterstudiengang

INTERACTIVE MEDIA

in Hagenberg

im Juni 2011

© Copyright 2011 Christian Grossauer

Diese Arbeit wird unter den Bedingungen der *Creative Commons Lizenz Namensnennung–NichtKommerziell–KeineBearbeitung Österreich* (CC BY-NC-ND) veröffentlicht – siehe <http://creativecommons.org/licenses/by-nc-nd/3.0/at/>.

Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die aus anderen Quellen entnommenen Stellen als solche gekennzeichnet habe.

Hagenberg, am 27. Juni 2011

Christian Grossauer

Inhaltsverzeichnis

Erklärung	iii
Kurzfassung	vii
Abstract	viii
1 Motivation	1
1.1 Mathematiksoftware	1
1.1.1 Computer Algebra Systeme	2
1.1.2 Dynamische Geometrie Software	2
1.1.3 Weitere Mathematiksoftware	2
1.2 Probleme bei der Wissensvermittlung	2
1.3 Vortragsarten	3
1.3.1 Analoge Vorträge	4
1.3.2 Inaktive Vorträge	5
1.3.3 Interaktive Vorträge	5
1.4 Interaktive Whiteboards	7
1.4.1 Allgemeines	7
1.4.2 Voraussetzungen für Vortragende	7
1.4.3 Entwicklungsstufen der Vortragenden	9
1.5 Motivation für NiCE Formula Editor	9
1.6 NiCE Formula Editor	10
1.6.1 GeoGebra	11
1.6.2 Anoto	12
2 Verwandte Arbeiten	15
2.1 Sketch-Understanding Systems	15
2.2 MathPad ²	16
2.3 Hands-On Math	17
2.4 MathJournal TM	19
2.5 MathBrush	19
3 Implementierung	21
3.1 Kurzbeschreibung der Funktionsweise	21

3.2	Layout	22
3.3	Ablauf	23
3.3.1	Eingabe erstellen	23
3.3.2	Einteilung der Strokes in Gruppen	24
3.3.3	Mathematikinterpretation	27
3.3.4	Textinterpretation	29
3.3.5	Verarbeitung der Ergebnisse	29
3.4	Speichern der erfassten Daten	29
3.4.1	Inhalt der .info Datei	31
3.4.2	Rekonstruktion der Strokegroups	31
3.4.3	Dateistruktur	31
3.5	Verwendete Komponenten	32
3.5.1	Klassenarchitektur	32
3.5.2	MathML Parser	34
3.5.3	Präsentations-Konverter	35
3.5.4	Interpreter	36
3.5.5	Webbrowser	41
3.5.6	MathJax	42
3.5.7	Mathematiksoftware	43
3.6	Probleme	44
3.6.1	MathInputControl Probleme	44
3.6.2	Notizzuordnung	47
3.6.3	Navigation	48
3.6.4	Reihenfolge bei der Erstellung	49
4	Ergebnisse	52
4.1	Einsatzgebiete	52
4.1.1	Vektorrechnung	52
4.1.2	Drehmatrizen	53
4.1.3	Kurvendiskussion	54
4.1.4	Vor- und Nachteile	55
4.2	Funktionalität	55
4.2.1	Hauptfunktionen	55
4.2.2	Weitere Funktionen	57
4.2.3	Optionen	58
4.2.4	Werkzeuge	59
4.2.5	Ansichten	60
5	Ausblick	62
5.1	Umstieg auf HTML5	62
5.2	Interaktion mit Publikum	63
5.3	Weitere Verbesserungen	63
5.3.1	Wechsel des Werkzeugs	64
5.3.2	Export-Möglichkeiten	65

Inhaltsverzeichnis	vi
5.3.3 Mehrsprachigkeit	65
A Anhang	66
B Inhalt der CD-ROM/DVD	69
B.1 Diplomarbeit	69
B.2 Abbildungen	69
B.3 Literatur	69
Literaturverzeichnis	71

Kurzfassung

Da heutzutage traditionelle Tafelvorträge im Bereich der Mathematik noch sehr verbreitet sind, in der Praxis aber vor allem auf den Einsatz von *Computer Algebra Systemen* (CAS) gesetzt wird, stellt sich die Frage, wieso die Theorie immer noch auf analogen Vortragsmedien übermittelt wird bzw. ob sich die Anwendung der Praxis nicht intuitiver gestalten lassen könnte. Die Umsetzung der Theorie in den Labors scheitert oftmals nicht an dem Verständnis, sondern an der technischen Hürde. Die Formulierung der mathematischen Ausdrücke muss für jedes System eigens angepasst werden und hat schlussendlich nichts mehr mit der handgeschriebenen Form gemeinsam. Könnte man den Systemen die Problemstellungen in derselben Form übermitteln, in der man diese in der Theorie vorgetragen bekommen hat, würde dies eine große Erleichterung darstellen.

Würde man analoge Vortragsmedien durch digitale interaktive Vortragsmedien ersetzen, könnte man dadurch die Qualität der Unterrichtsmaterialien steigern, die Aufmerksamkeit der Hörer vergrößern sowie die Inhalte einer größeren Masse zur Verfügung stellen. Interaktive Whiteboards (IWBs) wären etwa solch ein digitales interaktives Vortragsmedium. Diese Arbeit soll einen Überblick über die Vorteile der Verwendung von interaktiven Whiteboards im Mathematikunterricht geben und einen Teil der speziell dafür entwickelten Software näher erläutern.

Der für diese Arbeit erstellte *NiCE Formula Editor* kombiniert den Vorteil traditioneller Vortragsmedien, Gegebenheiten einfach und unkompliziert niederschreiben zu können, mit dem digitaler Vortragsmedien, jederzeit die Unterstützung von Mathematiksoftware in Anspruch nehmen zu können.

Abstract

Today, many math lectures are mainly using traditional blackboards for the demonstration of certain problems. Though during lab hours, the main focus lies in the use of *Computer Algebra Systems* (CAS). The transmission of the theoretical knowledge into the practical usage with an CAS represents the biggest problem. Often, the implementation of the theory fails because of the technical hurdle. There is no natural way for entering mathematical expressions into CAS. Each system uses a specific syntax and most of the time, this syntax has no more similarity with the handwritten form. Ideally a system would understand a handwritten mathematical expression in the form, a mathematician writes it down.

Enhanced quality of teaching materials, increased attention during lectures and the possibility to publish the presented content for a large audience, would be the benefits of replacing the traditional analog with interactive digital media. Interactive Whiteboards (IWBs) would be an example of such a digital interactive media. This work will provide an overview of the benefits of using IWBs in mathematics lessons and explain some of the already existing programs in this field.

The underlying project of this thesis, the *NiCE Formula Editor*, combines the advantage of being able to create simple and straightforward content with the possibility of using a mathematics software, to gain certain advantages.

Kapitel 1

Motivation

Mathematikunterricht und mathematische Vorträge laufen heutzutage oftmals immer noch unter der Verwendung von Tafel und Kreide ab. Wie in Abbildung 1.1¹ übertrieben dargestellt, trifft man im universitären Sektor einerseits häufig auf traditionelle Tafelvorträge ohne jegliche Unterstützung digitaler Medien, andererseits wird bei der praktischen Anwendung im Labor oft nur auf den Einsatz von *Computer Algebra Systemen* (CAS) oder *dynamischer Geometrie Software* (DGS) gesetzt.

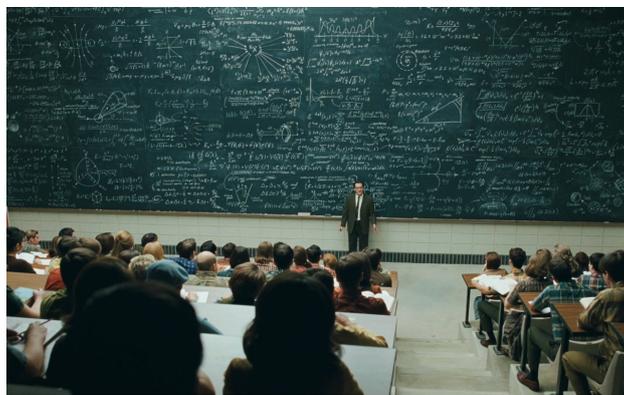


Abbildung 1.1: Traditioneller Tafelvortrag (übertriebene Darstellung).

1.1 Mathematiksoftware

Bei Mathematiksoftware wird grundsätzlich zwischen *Computer Algebra Systemen*, *dynamischer Geometrie Software* und Systemen zur Tabellenkalkulation unterschieden. Gemeinsam mit diverser Statistik- und Systemmodell-

¹<http://geekartgallery.blogspot.com/2011/03/quick-pic-math-class.html>

lierungs-Software werden alle Systeme zusammenfassend als Mathematiksoftware bezeichnet.

1.1.1 Computer Algebra Systeme

Computer Algebra Systeme beschäftigen sich hauptsächlich mit der Lösung von symbolischen Problemstellungen. Sie werden zum Vereinfachen von mathematischen Ausdrücken und Lösen von Gleichungen verwendet. Bekannte CAS sind *Mathematica*², *MuPAD*³ (jetzt Bestandteil von *MATLAB*⁴), *Maple*⁵ und *Derive* (nur noch als Teil der Firmware für *Texas-Instruments*⁶ Taschenrechner weiterentwickelt).

1.1.2 Dynamische Geometrie Software

Dynamische Geometrie Software beschäftigt sich mit der Erstellung und Manipulation geometrischer Gegebenheiten am Computer. Durch die Möglichkeit die Position erstellter Objekte zu ändern und sich diese Veränderung auf zusammenhängende Objekte auswirkt, entsteht eine gewisse Dynamik. Dadurch können beispielsweise Konstruktionsvorgänge veranschaulicht und auf Korrektheit überprüft werden. Verbreitete DGS Programme sind *Geometer's Sketchpad*⁷, *EUKLID DynaGeo*⁸ und Vertreter der *Cabri*-Serie⁹.

1.1.3 Weitere Mathematiksoftware

Weitere Systeme, wie beispielsweise *MATLAB*, beschäftigen sich grundsätzlich nur mit der Lösung von numerischen Problemen. Der Einsatzbereich dieser Systeme besteht aus Datenerfassung, Analyse, Auswertung und Erstellung von numerischen Simulationen. Des Weiteren existieren noch spezielle Systeme, wie *Dynasys*¹⁰, die zur Modellierung und Simulation dynamischer Systeme verwendet werden.

1.2 Probleme bei der Wissensvermittlung

Die Verwendung von Mathematiksoftware im täglichen Mathematikunterricht hat in einigen Schulstufen leider noch nicht Fuß gefasst. Oftmals bieten die verwendeten Programme zahlreiche nicht benötigte Funktionen an, die

²<http://www.wolfram.com/mathematica/>

³<http://www.mupad.com/>

⁴<http://www.mathworks.de/>

⁵<http://www.maplesoft.com/>

⁶<http://www.ti.com/>

⁷<http://www.dynamicgeometry.com/>

⁸<http://www.dynageo.de/>

⁹<http://www.cabri.com/>

¹⁰<http://www.hupfeld-software.de/>



Abbildung 1.2: Traditioneller Tafelvortrag (a) und klassisches Laborszenario (b).

den Umgang lediglich verkomplizieren und somit das Verständnis für gewisse Sachverhältnisse nicht fördern und oftmals sogar verschlechtern.

Wird Mathematiksoftware verwendet, kommt es häufig zu Problemen das erlernte Wissen richtig einzusetzen. Dies liegt zum Teil an der in Abbildung 1.2 dargestellten Trennung von der im Vortrag präsentierten Theorie (a)¹¹ und der im Labor angewandten Praxis (b)¹², da zwischen den beiden Welten nur schwer eine Verknüpfung übermitteln werden kann. Die größte Hürde stellt dabei meistens die Übersetzung eines in der Theorie verstandenen Sachverhaltes in die von der Mathematiksoftware gesprochene Sprache dar. Jedes System verwendet eine eigene Syntax und bedarf einer gewissen Einarbeitungsphase, bis ein akzeptabler Arbeitsfluss entsteht.

1.3 Vortragsarten

Im folgenden Abschnitt wird auf die Kategorisierung der verschiedenen Vortragsarten eingegangen, die während der Recherche angetroffen wurden.

Die verschiedenen Vortragsarten können grob in drei Kategorien eingeteilt werden: Analoge Vorträge ohne digitale Hilfsmittel, „inaktive“ digitale Vorträge und interaktive digitale Vorträge. Weiters existieren noch Mischformen, die Merkmale mehrerer Arten besitzen.

¹¹http://psychology.wikia.com/wiki/Lecture_method

¹²http://www.rci.rutgers.edu/~zylstra/compclass_core.html

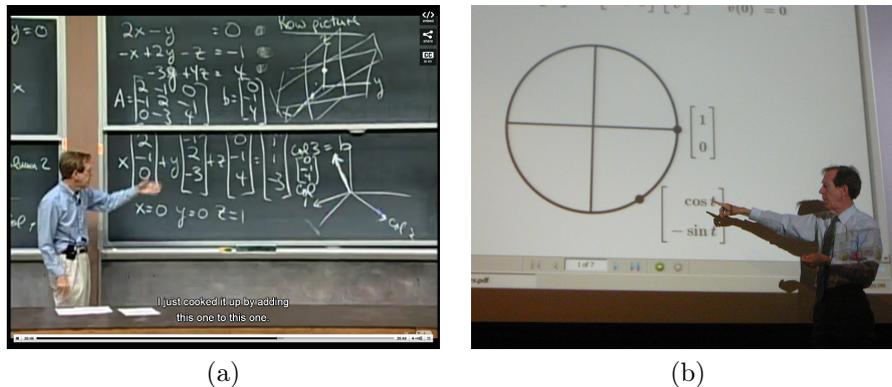


Abbildung 1.3: Gilbert Strang bei einem traditionellen Tafelvortrag (a) und bei der Zuhilfenahme digitaler Medien (b).

1.3.1 Analoge Vorträge

Abbildung 1.3 (a)¹³ zeigt einen Ausschnitt aus dem *MIT OpenCourseWare*¹⁴ Video *The geometry of linear equations* von Gilbert Strang aus dem Jahr 1999. *MIT OpenCourseWare* ist eine webbasierte Publikationsplattform des *Massachusetts Institute of Technology*¹⁵, auf der nahezu alle Kursmaterialien der angebotenen Lehrveranstaltungen kostenlos zur Verfügung stehen.

Die gesamte Reihe *18.06 Linear Algebra* besteht beispielsweise aus 34 Einheiten, die allesamt als Video vorhanden sind. In allen Einheiten verwendet Strang lediglich traditionelle Tafeln zur Verdeutlichung der Inhalte. Abbildung 1.3 (b)¹⁶ zeigt Strang in einer anderen Lehrveranstaltung, bei der Verwendung eines Videoprojektors.

Bei analogen Vorträgen werden ausschließlich Medien wie Tafeln, Flipcharts oder klassische Whiteboards zur Darstellung der Inhalte verwendet. Vor allem traditionelle Tafelvorträge, wie die von Strang, sind heutzutage im Bereich der Mathematik immer noch häufig vertreten.

Der große Nachteil dieser Vorträge liegt im Aufmerksamkeitsverlust der Hörer, da diese neben der Erstellung eigener Notizen mit dem Abschreiben des Tafelinhaltes beschäftigt sind. Dadurch wird oftmals die Möglichkeit zur Interaktion vorweggenommen, da entweder die Zeit neben dem Abschreiben nicht vorhanden ist oder der Inhalt nur oberflächlich aufgenommen und nicht hinterfragt wird [4]. Weiters können durch den Abschreibvorgang Fehler entstehen, die beim späteren Durchsehen der Notizen zu Unklarheiten führen können. Einerseits kann die sukzessive Konstruktion eines Funktionsgraphen

¹³<http://ocw.mit.edu/courses/mathematics/18-06-linear-algebra-spring-2010/video-lectures/lecture-1-the-geometry-of-linear-equations/>

¹⁴<http://ocw.mit.edu>

¹⁵<http://web.mit.edu/>

¹⁶<http://www.flickr.com/photos/24347385@N00/474604078/in/photostream>

per Hand das Verständnis für die Konstruktion verbessern, andererseits kann es aber auch unnötig Zeit in Anspruch nehmen. Dies trifft häufig ein, wenn die Konstruktionen lediglich zur Veranschaulichung benötigt werden oder es sich um mehrere gleichartige Graphen handelt. Der begrenzte Platz auf dem physischen Medium stellt neben dem geringen, aber im Endeffekt doch beträchtlichen Zeitverlust durch das stetige Löschen der Arbeitsfläche, ebenfalls einen Nachteil dar.

Vorteile der in den analogen Vorträgen verwendeten Medien sind die geringen anfallenden Kosten, die hohe physische Widerstandsfähigkeit und die große Zuverlässigkeit.

1.3.2 Inaktive Vorträge

„Inaktive“ digitale Vorträge verwenden zwar ebenfalls Tafel oder Whiteboard zur Verdeutlichung der Inhalte, werden aber, wie beispielsweise in Abbildung 1.3 (b) dargestellt, durch ein digitales Hilfsmittel unterstützt. Dabei handelt es sich meistens um eine Rechner-Projektor-Kombination auf der Unterrichtsmaterialien präsentiert werden. Diese Vortragsart wird deshalb als „inaktiv“ bezeichnet, da der im Vorhinein erstellte Inhalt während des Vortrags lediglich präsentiert wird und keine Änderungen vorgenommen werden. Die vom Vortragenden zusätzlich zu den Unterlagen erstellten Notizen müssen, wie bei analogen Vorträgen, von den Hörern selbst festgehalten werden. Die meist im Vorhinein vorhandenen Unterlagen verringern zwar in den meisten Fällen den Abschreibaufwand und reduzieren dadurch den Aufmerksamkeitsverlust, jedoch kommt es sofort wieder zu Aufmerksamkeitseinbußen sobald zusätzliche Inhalte auf dem analogen Medium erstellt werden. Weiters kann es dabei wieder zu Fehlern beim Übertragen der Inhalte kommen.

1.3.3 Interaktive Vorträge

Der größte Unterschied zu den anderen Vortragsarten besteht aus dem dynamischen Inhalt der Unterrichtsmaterialien, der durch die Interaktion mit den Hörern entsteht. Die grundlegende Struktur kann dabei ebenfalls wie bei „inaktiven“ Vorträgen im Vorhinein erstellt und beispielsweise in Form einer klassischen Präsentation dargeboten werden. Der Inhalt kann jedoch an die aktuelle Situation angepasst und dadurch gezielt auf Unklarheiten eingegangen werden. Natürlich kann auch bei den anderen Vortragsarten direkt und gezielt auf Probleme eingegangen werden, bei interaktiven Vorträgen fließt jedoch der Inhalt anschließend direkt in die Unterlagen ein, wodurch die Unterrichtsmaterialien individuell an die Anforderungen der Hörer angepasst werden. Dies kann beispielsweise durch die Verwendung von interaktiven Whiteboards (IWB) realisiert werden. Da keine analogen Medien mehr verwendet werden stehen Hörern und anderen Vortragenden exakte Kopien aller, während des Vortrags erstellten Inhalte, zur Verfügung. Dies



Abbildung 1.4: *Smart Classrooms* der *Polytechnic Institute of New York University*.

kann sogar so weit gehen, dass neben den Notizen zusätzlich noch die Sprache des Vortragenden aufgenommen wird. Dadurch existiert mehr Zeit um den Inhalt genauer aufzunehmen und bei Unklarheiten gegebenenfalls sofort nachzufragen.

Die in digitaler Form zur Verfügung stehenden Unterlagen können weiters als Basis für zukünftige Unterrichtsmaterialien dienen. Durch die Bereitstellung der Materialien auf öffentlichen Plattformen kann dadurch sogar institutsübergreifender Austausch der Unterlagen und Erfahrungen stattfinden, was sich wiederum positiv auf die Qualität der Materialien auswirkt.

Lediglich der etwas größere Zeitaufwand bei der Erstellung und Planung der Vorträge sowie die speziellen Voraussetzungen an Räumlichkeiten und Equipment stellen Nachteile dar.

Das *Polytechnic Institute of New York University* hat für ihr *Center for Faculty Innovations in Teaching and Learning* (FITL) 21 Klassenzimmer in so genannte *Smart Classrooms*¹⁷ umgerüstet. FITL beschäftigt sich mit der Ausbildung von Lehrenden und der Erforschung und Verbesserung von Ausbildungspraktiken. In den *Smart Classrooms* sollen dabei die neuen innovativen Lernstrategien und Lerntechniken, die auf der Verwendung von modernem Equipment basieren, getestet werden. In Abbildung 1.4 sind die vier verschiedenen Typen der *Smart Classrooms* abgebildet, die sich in Punkto

¹⁷<http://www.poly.edu/academics/support/FITL/smart-classrooms>

Personenkapazität und Ausstattung mit Equipment unterscheiden. Drei der vier Typen sind mit interaktiven Whiteboards ausgestattet, die zusätzlich zu klassischen Whiteboards während des Unterrichts zum Einsatz kommen.

Welche Vortragsart nun die Beste ist, lässt sich nicht allgemein sagen. Nicht jeder Vortragende kann auf allen Arten gleich gut vortragen und manche Thematiken lassen sich nur auf eine gewisse Art wiedergeben. Die Aufwertung der Unterrichtsmaterialien durch die Einbindung individueller Inhalte sowie die Möglichkeit zur Bereitstellung aller im Vortrag erstellten Inhalte, stellen allerdings einen klaren Vorteil interaktiver digitaler Vorträge gegenüber den anderen Arten dar.

1.4 Interaktive Whiteboards

1.4.1 Allgemeines

Die klassische Tafel im Klassenraum stellt laut Breuer et al. [1] eine Leithilfe durch verschiedene Phasen dar, soll die Blicke und Aufmerksamkeit der Hörer auf eine zentrale Stelle fokussieren, ermöglicht den aktuellen Inhalt zur späteren Rekonstruktion zu dokumentieren und verbessert das allgemeine Lernen und Verstehen durch die Schaffung einer klaren Strukturierung. Levy wirft in ihrem Report¹⁸ den Whiteboards vor, die Aufmerksamkeit zu sehr auf den Vortragenden zu ziehen und dadurch die Hörer in eine zu passive Rolle zu versetzen. Neue Ansätze versuchen ein offenes und problemorientiertes Klima zu schaffen, in dem der Lehrende/Vortragende die Rolle eines Moderators innehat und dadurch ein konstruktiver Dialog entstehen kann. Diese Ansätze sollen zur Zusammenarbeit und zum selbstständigen Arbeiten animieren. Statt vorgefertigte Lernunterlagen lediglich präsentiert zu bekommen, sollen die Hörer in den Erstellungsprozess dieser Unterlagen einbezogen werden. Laut Breuer et al. [1] unterstützt die gängige Software für interaktive Whiteboards solche interaktive Funktionen nicht.

1.4.2 Voraussetzungen für Vortragende

Durch die Verwendung von interaktiven Whiteboards müssen Struktur und Denkweise der Vortragenden verändert werden. Ein guter Vortragender ist nicht automatisch auch ein guter „interaktiver Whiteboard Vortragender“. Durch die von Glover et al. [3] durchgeführten Interviews und Fragebögen, konnten drei verschiedene Haltungen von Lehrenden gegenüber der neuen Technologie erkannt werden.

¹⁸<http://dis.shef.ac.uk/eirg/projects/wboards.htm>

Missioners

Diese Lehrenden sind auf die neue Technologie durch eigenes Interesse oder durch Übungskurse aufmerksam geworden und wenden diese in jeder Situation, in der es möglich ist, an. Durch die ständige Anwendung verbessern sie sich unentwegt im Umgang mit den interaktiven Whiteboards und versuchen weitere Lehrende für diese Technologie zu begeistern. Der gesamte Lernprozess wurde hingehend Interaktivität und Flexibilität angepasst um das volle Potential dieser Technologie nützen zu können.

Missioners bewerteten grundsätzlich die Verwendung von interaktiven Whiteboards sehr positiv, lediglich der zeitliche Mehraufwand bei der Vorbereitung von Lehreinheiten, sowie die Angst auf Grund mangelnder Begabung an der technischen Hürde scheitern zu können, wurde von einigen als Problem angeführt.

Tentatives

Tentatives bekamen meist eine Einführung in die Nutzung von interaktiven Whiteboards und hatten auch Zugang zu Räumlichkeiten, die mit der Technologie ausgestattet waren. Das Interesse wurde aber durch den Gedanken an zusätzliche Weiterbildung und der Angst vor der Zeit die benötigt würde, um die neuen Situationen im täglichen Unterricht bewältigen zu können, wesentlich gemindert.

Sie verwendeten nur einen kleinen, grundlegenden Teil des Spektrums der verfügbaren Anwendungsmöglichkeiten dieser Technologie. Sie nutzten interaktive Whiteboards maximal für eine Einheit pro Tag, aber mindestens einmal pro Woche. Sie erhofften sich durch die Verwendung eine höhere Aufmerksamkeit der Schüler sowie die Erstellung von besonders gut in Szene gesetzter Unterrichtsmaterialien für gewisse Themengebiete.

Luddites

Luddites hatten ebenfalls die Möglichkeit die Technologie zu nutzen, besaßen aber auf Grund der Angst vor Unbekanntem und der eventuell benötigten Umstrukturierung der Unterrichtseinheiten, von Grund auf eine gewisse abgeneigte Haltung. Ein Teilnehmer gab beispielsweise folgenden Kommentar ab: „*Ich habe wirklich auch schon ohne all die zusätzliche Einarbeitung in die neue Technologie genug zu tun. Ich weiß, dass andere Lehrer in der Schule Stunden in Vorbereitungen investieren und es dann trotzdem Probleme gibt, da etwas schief geht. Ich habe gehört, dass die Kinder mit den Stiften gerne herumalbern und dass einige mehr als der Lehrer über Computer wissen und man dadurch dann seine Glaubwürdigkeit verliert. Traditioneller Unterricht hält die Klasse unter Kontrolle.*“

1.4.3 Entwicklungsstufen der Vortragenden

In *Enhancing mathematics teaching through new technology: the use of the IAW* [9], einem *Nuffield Foundation research project*¹⁹ werden drei Entwicklungsstufen beschrieben, die Vortragende durchmachen, wenn sie anfangen interaktive Whiteboards zu verwenden. Diese Entwicklung wird in *Supported Didactic*, *Interactive* und *Enhanced Interactive* eingeteilt.

Supported Didactic

Das interaktive Whiteboard wird lediglich zur visuellen Unterstützung des Vortrags verwendet. Es existiert wenig Interaktivität, die Hörer werden kaum in den Vortrag involviert und es kommt selten zu konstruktiven Diskussionen.

Interactive

Die potentiellen Vorteile des interaktiven Whiteboards werden von Zeit zu Zeit in Anspruch genommen um die Aufmerksamkeit der Hörer zu erhalten. Durch die Verwendung von verbalen, visuellen und ästhetischen Mitteln sollen die Hörer mehr gefordert werden.

Enhanced Interactive

Der Übergang zu dieser Stufe findet zum größten Teil durch ein Umdenken des Vortragenden bei der Verwendung von neuen Technologien im täglichen Arbeitsalltag statt. In dieser Stufe kommen interaktive Whiteboards in nahezu allen Vorträgen zum Einsatz. Leider nicht häufig genug, werden auch die Interaktionsmöglichkeiten zwischen Hörern und Vortragenden verwendet. Vortragende sind sich der Vorteile der Technik bewusst und können diese problemlos in den Vortrag integrieren. Die Struktur der Vorträge wird darauf ausgelegt, dass Hörer auf die durch das interaktive Whiteboard erzeugten Reize als Individuen, als Paare oder als Gruppen reagieren können und so aktives Lernen ermöglicht wird.

Durch die richtige Verwendung eines interaktiven Whiteboards werden die Hörer zu Diskussionen animiert, können Prozesse besser veranschaulicht oder Hypothesen entwickelt werden, die anschließend durch den Einsatz von geeigneten Anwendungen getestet und verifiziert werden können. *NiCE Formula Editor* soll eben für mathematische Vorträge so eine Anwendung darstellen.

1.5 Motivation für NiCE Formula Editor

Die Vorteile von digitalen und analogen Vortragsmöglichkeiten sollen kombiniert und so weit wie möglich genutzt werden. Dies soll durch die Kombi-

¹⁹<http://www.nuffieldfoundation.org/>

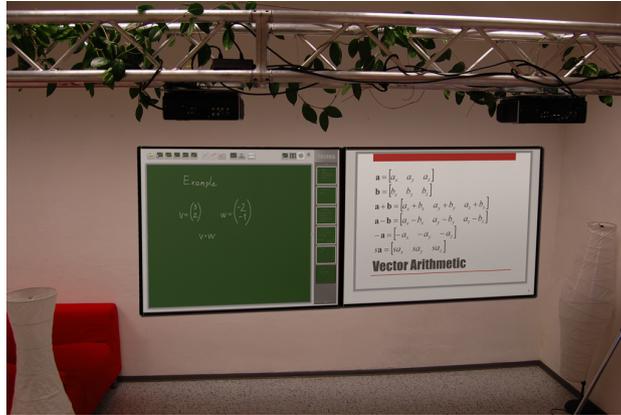


Abbildung 1.5: *NiCE Formula Editor* in Aktion.

nation von analogen Tafel- mit digitalen Computervorträgen passieren. Die problemlose und schnelle Eingabe wie bei Tafel und Kreide soll dabei mit den zahlreichen Unterstützungsmöglichkeiten durch digitale Medien zusammengefügt werden. Die Eingabe soll wie gewohnt erfolgen und kann gleich an ein *Computer Algebra System* zur Darstellung übergeben oder einfach für einen späteren Zeitpunkt gespeichert werden. Der als Grundlage dieser Arbeit dienende *NiCE Formula Editor* setzt für diese Kombination auf die Verwendung von interaktiven Whiteboards.

NiCE Formula Editor kombiniert Vorteile analoger Vorträge, wie die gewohnte Eingabe per Hand, mit Vorteilen digitaler Vorträge, wie grenzenlos zur Verfügung stehender Platz, die Unterstützung durch Mathematiksoftware und die Möglichkeit Inhalte für eine spätere Verwendung zu speichern.

1.6 NiCE Formula Editor

Die Ausstattung traditioneller Hörsäle besteht heutzutage immer noch zumindest aus einer oder mehreren Tafeln, in einigen Fällen aus Whiteboards. Zusätzlich existiert meistens eine Projektionsfläche für Videoprojektoren bzw. vereinzelt noch für Overhead-Projektoren. Diese Kombination aus analogen und digitalen Vortragsmedien ist sehr weit verbreitet, könnte aber durch den geschickten Einsatz von interaktiven Whiteboards ersetzt werden. Durch die Verwendung von zwei interaktiven Whiteboards kann man jedes aus beliebig vielen Tafeln und einer Projektionsfläche bestehende Szenario nachstellen und hat trotzdem die gleichen Möglichkeiten zur Verfügung. Jedes interaktive Whiteboard kann dabei als Projektions- und gleichzeitig als Notizfläche verwendet werden, wodurch sich diese Konstellation für den *NiCE Formula Editor* als am besten geeignet herausstellte.

In Abbildung 1.5 ist das Demo Szenario des *NiCE Formula Editor* darge-

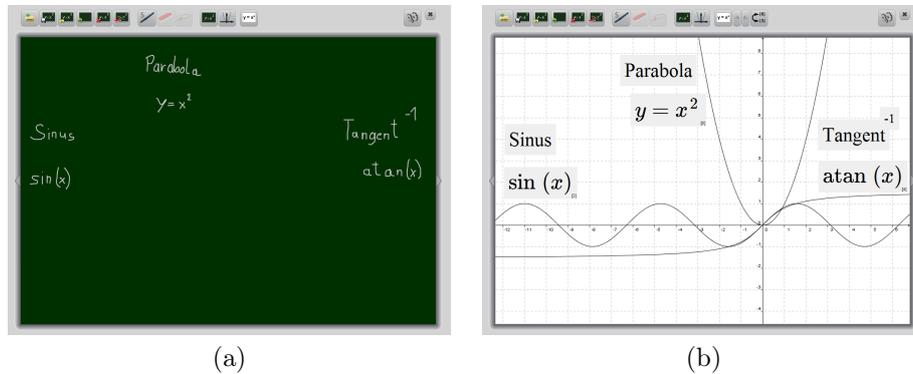


Abbildung 1.6: Handgeschriebene Notiz (a) und Darstellung in *GeoGebra* (b).

stellt. Dieses besteht eben aus zwei interaktiven Whiteboards, zwei Projektoren und einem Rechner. Die zwei *eno classic 2610* Whiteboards der Marke *Polyvision*²⁰ können sowohl als traditionelles als auch als digitales Whiteboard verwendet werden. Die aus einem Keramik-Stahl-Verbund bestehende Oberfläche der Whiteboards ist magnetisch, mit einem *Anoto*²¹-Muster bedruckt und kann mit handelsüblichen Markerstiften oder digitalen Stiften beschrieben werden.

Das Szenario spielt während eines mathematischen Vortrages, in dessen Zentrum eine Präsentation steht, die mit handgeschriebenen Kommentaren/Notizen ergänzt wird. Dabei wird die Präsentation auf einem der beiden Whiteboards angezeigt, während das Zweite zur Erstellung dazugehöriger Notizen verwendet wird. Die Inhalte der Notizen können, wie in Abbildung 1.6 zu sehen, entweder in der ursprünglichen Form als handgeschriebene Notizen oder als Text- bzw. Mathematik-Interpretation angezeigt werden. Mathematische Ausdrücke können zusätzlich mit der im System integrierten Mathematiksoftware *GeoGebra*²² visuell dargestellt werden. Es können ebenso Notizen erstellt werden ohne dass dabei eine Präsentation geöffnet ist. Die zur Verfügung stehenden Funktionen bleiben dieselben, lediglich die Verknüpfung der Notiz mit einer Folie fällt dabei weg.

1.6.1 GeoGebra

Zur Visualisierung der mathematischen Gegebenheiten wird die von Hohenwarter entwickelte Mathematiksoftware *GeoGebra* verwendet [5,6]. *GeoGebra* ist eine auf Java basierte betriebssystemunabhängige kostenlose dynamische Mathematiksoftware und benötigt lediglich eine *Java Runtime Environment*

²⁰<http://www.polyvision.com/>

²¹<http://www.anoto.com/>

²²<http://www.geogebra.org/>

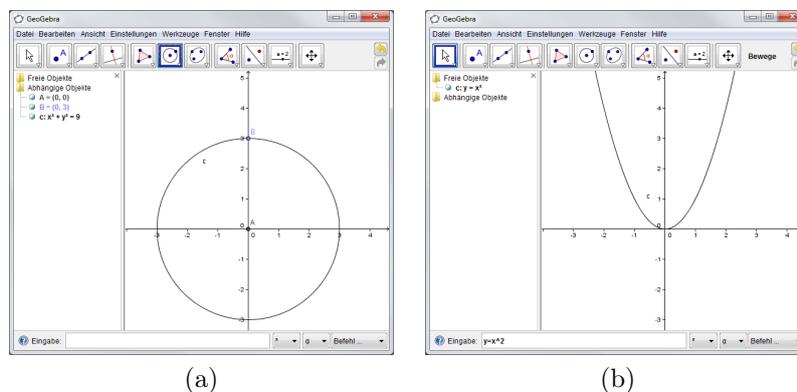


Abbildung 1.7: Objekterzeugung in Version 3.2 von *GeoGebra* durch Verwendung eines Werkzeuges (a) und durch Eingabe einer Funktion (b).

um ausgeführt werden zu können. Das Kunstwort *GeoGebra* setzt sich aus Geometrie und Algebra zusammen, was gleichzeitig die Anwendungsgebiete widerspiegeln soll. Wie in Abbildung 1.7 (a) zu sehen ist, können mit *GeoGebra* Objekte schnell und einfach mit der Maus erstellt und bearbeitet werden. Die in (b) abgebildete Parabel wurde durch Eingabe des mathematischen Ausdrucks $y = x^2$ erzeugt. Veränderungen der Geometrie wirken sich ebenfalls auf die algebraische Beschreibung aus, wodurch man *GeoGebra* auch als eine Art *dynamische Geometrie-Software* betrachten kann. Verschiebt man beispielsweise die Parabel aus Abbildung 1.7 (b) um eine Einheit nach oben, so ändert sich die der mathematische Ausdruck auf $y = x^2 + 1$.

1.6.2 Anoto

Die Technologie die von *Anoto* verwendet wird, ermöglicht eine schnelle und zuverlässige Digitalisierung von handgeschriebenen Texten und Zeichnungen. Dafür werden lediglich ein digitaler Stift und eine mit dem *Anoto*-Muster versehene Oberfläche benötigt. Das Verfahren kommt nicht nur auf Papier, sondern ebenfalls auf Whiteboards, Glas und LCD-Bildschirmen zur Anwendung.

Digitaler Stift

Ein digitaler Stift, wie er in Abbildung 1.8 dargestellt ist, funktioniert wie ein normaler Kugelschreiber. Er enthält jedoch zusätzlich eine integrierte Digitalkamera, einen Mikroprozessor, eine Batterie und ein mobiles Kommunikationsgerät für die drahtlose Verbindung zum Rechner.

Wird mit einem digitalen Stift auf einer mit dem *Anoto*-Muster versehenen Oberfläche eine Eingabe getätigt, kann die Handschrift erfasst, gespeichert und versendet werden.

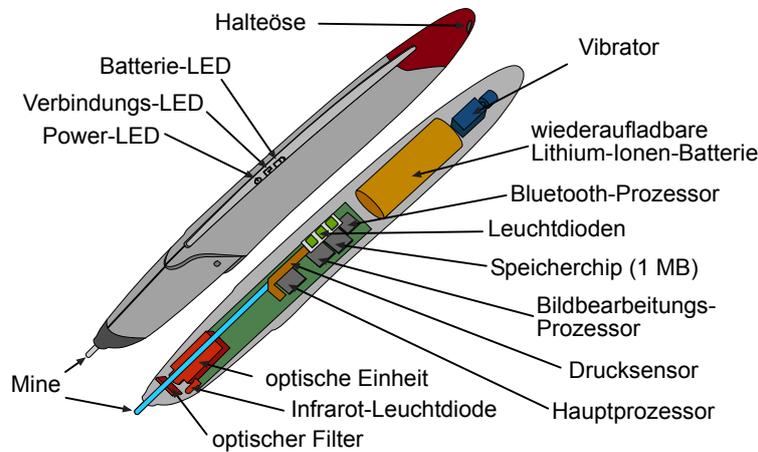


Abbildung 1.8: Digitaler Stift (Grafik: Jakob Leitner).

Anoto-Muster

Das *Anoto*-Muster besteht aus 0,1 mm großen schwarzen Punkten, die von dem digitalen Stift gelesen werden können. Die Punkte sind dabei an einem 0,3 mm großen Raster ausgerichtet. Jedoch sind sie, wie in Abbildung 1.9 zu sehen, immer etwas nach oben, unten, links oder rechts versetzt. Diese Anordnung passiert nach einem speziellen System, wodurch jeder von der eingebauten Kamera erfasste Teilausschnitt des Musters (normalerweise eine Gruppe von sechs mal sechs Punkten) eine einzigartige Kombination von Punkten enthält und so die exakte Position bestimmt werden kann.

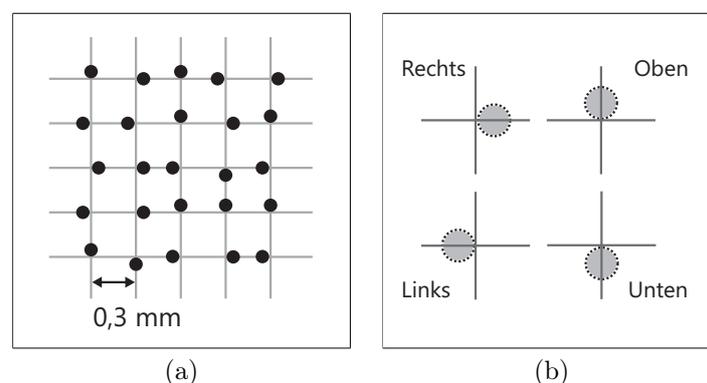


Abbildung 1.9: Eine Ausschnitt des *Anoto*-Musters (a) und die vier verschiedenen Anordnungsmöglichkeiten der Punkte (b).

Der Schreibvorgang

Während des Schreibvorganges werden von der Kamera automatisch zwischen 50 und 100 Aufnahmen pro Sekunde vom Muster auf der Oberfläche erstellt. Jede dieser Aufnahmen enthält genug Informationen um die exakte Position des digitalen Stifts und dadurch die gerade getätigte Eingabe zu bestimmen.

Jeder Stift besitzt eine eindeutige ID, wodurch die gleichzeitige Verwendung mehrerer Stifte ermöglicht wird. Zusätzlich stehen Druckstärke, Neigungswinkel des Stiftes und Zeitstempel für jede Aufnahme zur Verfügung. Diese Informationen können für spätere Vorgänge, wie etwa zur Schrifterkennung, herangezogen werden. Die erfassten Daten können anschließend via *Bluetooth*²³ an ein beliebiges Gerät übermittelt und dort weiterverarbeitet werden.

²³<http://www.bluetooth.com/>

Kapitel 2

Verwandte Arbeiten

In diesem Kapitel werden Systeme beschrieben, die einen ähnlichen Aufgabenbereich wie der entwickelte *NiCE Formula Editor* abdecken. Die vorgestellten Systeme sind *Sketch-Understanding Systems* [2], *Mathpad²* [8], *Hands-On Math* [13], *MathJournalTM* [12] und *MathBrush* [7]. Einige dieser Systeme verwenden im Hintergrund eines der in Abschnitt 1.1.1 genannten CAS, sind aber oftmals sehr kostspielig und zu komplex für den täglichen Mathematikunterricht. Des Weiteren sind sie oftmals nur für Tablet PCs und nicht für großflächige Oberflächen konzipiert.

2.1 Sketch-Understanding Systems

Die von der *Design Rationale Group*¹ entwickelten *Sketch-Understanding Systems* sind Systeme, die sich mit der Interpretation von handgeschriebenen Zeichnungen beschäftigen. Die verschiedenen Systeme sind jeweils speziell für ein Einsatzgebiet entwickelt worden. Neben dem in Abbildung 2.1 dargestellten System aus dem Bereich der Physik, existieren weitere Systeme für UML-Diagramme, analoge Schaltkreise und chemische Strukturen. Das System soll den Inhalt automatisch erkennen und für die weitere Verwendung am Computer umwandeln, wodurch man auf einfachste Weise mit dem Computer interagieren kann.

Das gezeigte System namens *Assist (A Shrewd Sketch Interpretation and Simulation Tool)*, kann einfache Zeichnungen die physikalische Gegebenheiten enthalten interpretieren und anschließend die Situation in einer Simulation nachstellen. Das Beispiel aus der Abbildung zeigt den Querschnitt eines simplen Fahrzeugs, das auf einem schrägen Objekt steht. Durch die Erstellung des nach unten zeigenden Pfeiles wird dem System vermittelt, dass eine Kraft in diese Richtung wirkt. Durch das Kreuz innerhalb des schrägen Objektes, wird dieses als unbewegbar interpretiert. Die zwei Kreise, die innerhalb der beiden Räder eingezeichnet wurden, werden vom System als Achsen

¹<http://rationale.csail.mit.edu/index.shtml>

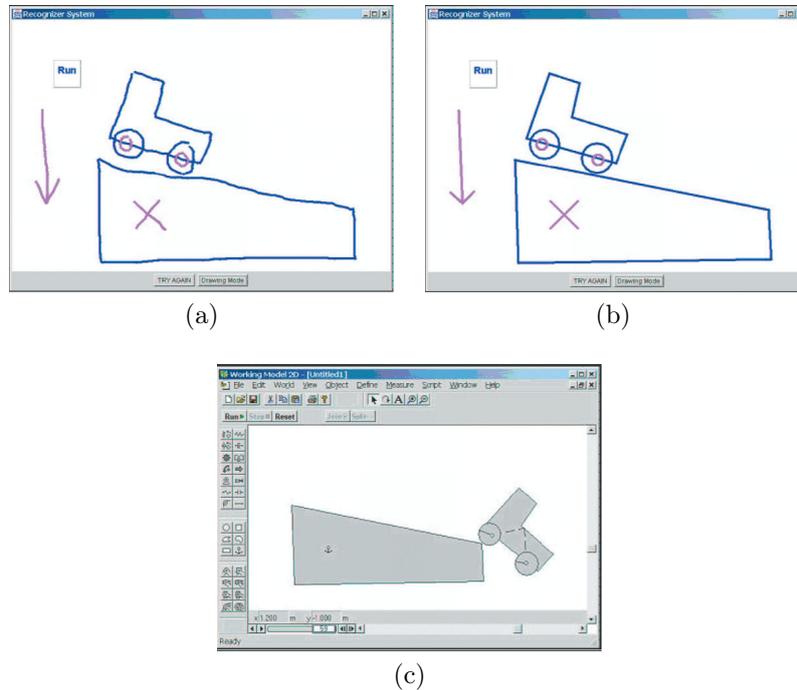


Abbildung 2.1: Handgeschriebene Originalzeichnung (a), Interpretation durch *Assist* (b) und durchgeführte Simulation (c). Aus [2].

interpretiert. Abbildung 2.1 (b) zeigt die Situation nach dem Glättungs- und Abstraktions-Vorgang. Wird anschließend die Animation gestartet, rollt das Fahrzeug, wie in (c) dargestellt, auf der schrägen Fläche herab, bis es schließlich vom Objekt völlig herunterfällt.

2.2 MathPad²

Der von LaViola et al. entwickelte Prototyp *Mathpad²* beschäftigt sich mit der Lösung von mathematischen Problemen. In der für einen Tablet PC entwickelten Anwendung werden jedoch die Eingaben nicht mit Tastatur und Maus sondern, wie vom Umgang mit Papier gewohnt, mit einem Stift erstellt. Die Eingaben können, wie in Abbildung 2.2 zu sehen, in traditioneller Form als mathematische Ausdrücke erstellt werden. Zusätzlich können Zeichnungen erstellt werden, die durch Verknüpfung mit den mathematischen Hintergrundinformationen anschließend sogar als Animation abgespielt werden können.

Das Beispiel in Abbildung 2.2 zeigt die Möglichkeit auf, wie eine Problemstellung durch die Verwendung von *Mathpad²* gelöst werden kann. Die Frage in diesem Beispiel ist, ob ein Ball über einen 125 Meter entfernten,

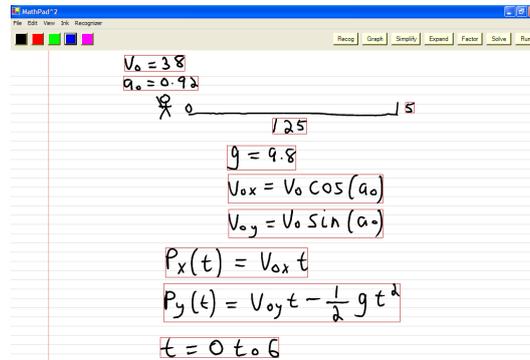


Abbildung 2.2: Beispiel eines horizontalen Wurfes, dargestellt in *Mathpad*². Aus [8].

5 Meter hohen Zaun fliegt oder davor liegen bleibt. Die bekannten Größen sind die Entfernung und Höhe des Zaunes, die Abwurfgeschwindigkeit $v_0 = 38$, der Abwurfwinkel $\alpha_0 = 0,92$ und die Gravitationsbeschleunigung $g = 9,8$. Die beiden Ausdrücke $v_{0x} = v_0 \cos(\alpha_0)$ und $v_{0y} = v_0 \sin(\alpha_0)$ beschreiben die horizontale bzw. vertikale Komponente der Anfangsgeschwindigkeit, $P_x(t) = v_{0x}t$ und $P_y(t) = v_{0y}t - \frac{1}{2}gt^2$ stellen die Bahngleichung in x- und y-Richtung dar. Zuletzt wird noch das Intervall $t = 0$ to 6 angegeben, wodurch die Animation abgespielt werden kann. *Mathpad*² verknüpft dazu die zur Verfügung stehenden Informationen der mathematischen Gegebenheiten mit der erstellten Skizze. In diesem konkreten Beispiel fliegt der Ball über den Zaun. Wird ein Parameter verändert wirkt sich dies direkt auf die Animation aus. So können verschiedene Situationen dargestellt und dadurch die Problemstellung veranschaulicht werden.

2.3 Hands-On Math

Hands-On Math ist ein von Zeleznik et al. entwickeltes System, welches die Vorteile von CAS mit der Eingabefreiheit von Papier und Bleistift kombinieren soll. Eingaben können dabei einfach mit einem digitalen Stift erstellt, vom System interpretiert und gegebenenfalls durch das CAS gelöst oder dargestellt werden. Weiters bietet *Hands-On Math* ein leicht verständliches, auf Bezel-Gesten [11] basiertes Seitenmanagement an, welches die Organisation von Dokumenten und Notizen mit mathematischem Inhalt erleichtert. In Abbildung 2.3 ist die Erstellung einer neuen Seite durch die Anwendung einer Bezel-Geste dargestellt. Dazu muss lediglich mit zwei Fingern von außerhalb der Anwendung über den rechten oder linken Rand gestrichen werden.

Der unbegrenzte Platz stellt, neben der Unterstützung durch ein CAS, einen weiteren Vorteil von der Verwendung des *Hands-On Math* Systems im Vergleich zu Papier und Bleistift dar. Ein Blatt Papier besitzt physische

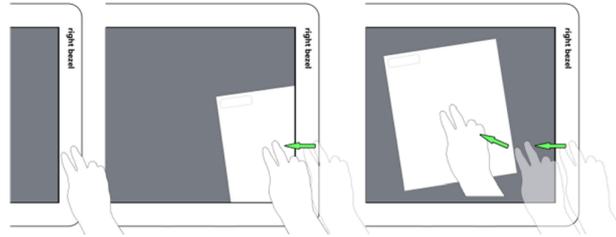
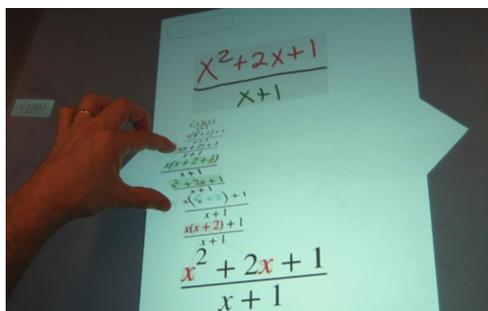


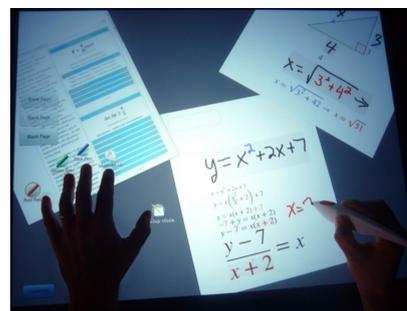
Abbildung 2.3: Bezel-Geste zur Erstellung einer neuen Notiz. Aus [13].

Grenzen und benötigt Platz, die Notizen in *Hands-On Math* können eine beliebige Größe besitzen und nehmen keinen „realen“ Raum in Anspruch. Es können beliebig viele Notizen erstellt und mit beliebig viel Inhalt versehen werden. Um nun einen Überblick über den Inhalt einer Notiz zu erhalten, wurde die Möglichkeit, Inhalte zusammenzufügen, zur Verfügung gestellt. Wendet man, wie in Abbildung 2.4 (a) dargestellt, eine Falt-Geste auf eine Inhaltspassage an, wird diese „versteckt“, als würde man ein Blatt Papier auf die dargestellte Weise falten.

Durch die bereitgestellte Multi-Touch Funktionalität, können in das System importierte Dokumente und erstellte Eingaben direkt per Hand manipuliert werden. Abbildung 2.4 (b) zeigt ein Anwendungsbeispiel, in dem ein Dokument und zwei Notizen mit mathematischen Inhalten dargestellt werden. Auf der unteren der beiden Notizen, wurde der mathematische Ausdruck $y = x^2 + 2x + 7$ erstellt und vom System auf den Ausdruck $\frac{y-7}{x+2} = x$ umgeformt. Mit der rechten Hand wird gerade eine neue Eingabe erstellt, in der x einen Wert zugewiesen bekommt, wodurch der Ausdruck nun lösbar wäre. Die auf der Oberfläche liegende linke Hand, ruft das sogenannte *PalmPrint* Menü auf, welches ausgewählte Funktionen von *Hands-On Math* beinhaltet.



(a)



(b)

Abbildung 2.4: Zusammenfassen von Inhalten in *Hands-On Math* (a) und *Hands-On Math* im Einsatz (b). Aus [13].

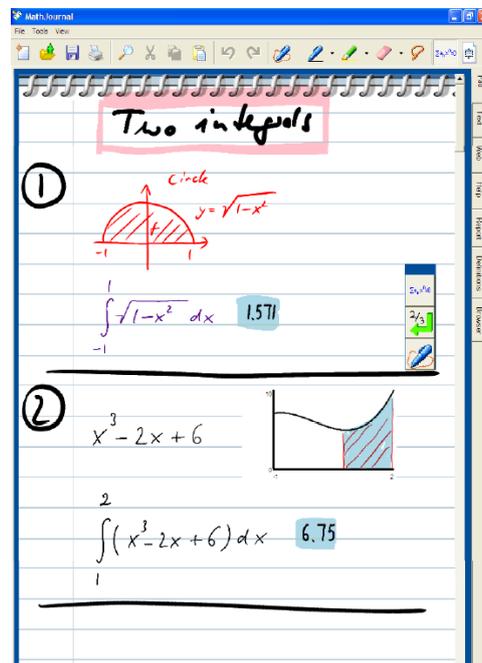


Abbildung 2.5: Integralrechnung in *MathJournal*TM. Aus [12].

2.4 MathJournalTM

*MathJournal*TM ist ein kommerzielles Programm für Tablet PCs, welches für das Lösen von schematischen und grafischen Mathematikproblemen entwickelt wurde. Per Stift erstellte handgeschriebene Eingaben können entweder im *free-form mode* oder dem *math mode* erstellt werden. Im *math mode* erstellte Eingaben können vom Programm interpretiert, visuell dargestellt und numerisch gelöst werden. *Free-form mode* Eingaben werden nicht interpretiert sondern lediglich dargestellt.

Das erste Beispiel in Abbildung 2.5 zeigt den im *math mode* erstellten mathematischen Ausdruck $\int_{-1}^1 \sqrt{1-x^2} dx$ und die im *free-form mode* erstellte dazugehörige Skizze. Der blau hinterlegte numerische Wert 1.571 stellt das von *MathJournal*TM ermittelte Ergebnis des Integrals dar. Das untere Beispiel zeigt die vom Programm erstellte visuelle Darstellung des mathematischen Ausdrucks $x^3 - 2x + 6$ sowie die numerische Lösung von 6.75 für die Fläche unter der Kurve innerhalb der angegebenen Grenzen von eins bis zwei.

2.5 MathBrush

Das von Labahn et al. entwickelte System, interpretiert auf einem Tablet PC getätigte, handgeschriebene Eingaben als *MathML* (vgl. Abschnitt 3.5.4) und

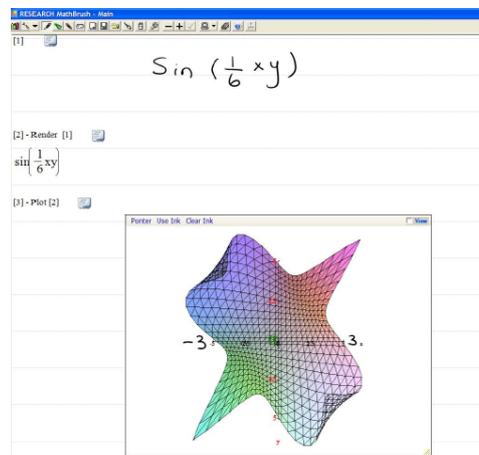


Abbildung 2.6: Modifikation der Darstellungsgrenzen durch Überschreiben der vorhandenen Darstellung in *MathBrush*. Aus [7].

übergibt diese Interpretationsergebnisse anschließend an das für die Lösung und Darstellung verantwortliche CAS. In Abbildung 2.6 ist die Interpretation und visuelle Darstellung des mathematischen Ausdrucks $\sin\left(\frac{1}{6}xy\right)$ zu sehen, bei dem die aktuellen Grenzen (-5 bis 5) durch -3 bis 3 ersetzt werden sollen.

Kapitel 3

Implementierung

In diesem Kapitel werden das Layout, die Abläufe während der Anwendung, die verwendeten Komponenten und die technische Umsetzung des *NiCE Formula Editors* beschrieben.

3.1 Kurzbeschreibung der Funktionsweise

Abbildung 3.1 zeigt einen Überblick über die Funktionsweise des *NiCE Formula Editor*. Die mit einem digitalen Stift auf einem interaktiven Whiteboard erstellten Eingaben werden von dem System erfasst und je nach Typ als Text oder mathematischer Ausdruck interpretiert. Die Interpretationen werden in eine HTML-Datei eingebettet, die anschließend in dem eingebundenen Webbrowser angezeigt werden kann. Handelt es sich dabei um mathematische Inhalte, können diese mittels der ebenfalls im Webbrowser angezeigten *Java-Applet* Version von *GeoGebra* visualisiert werden.

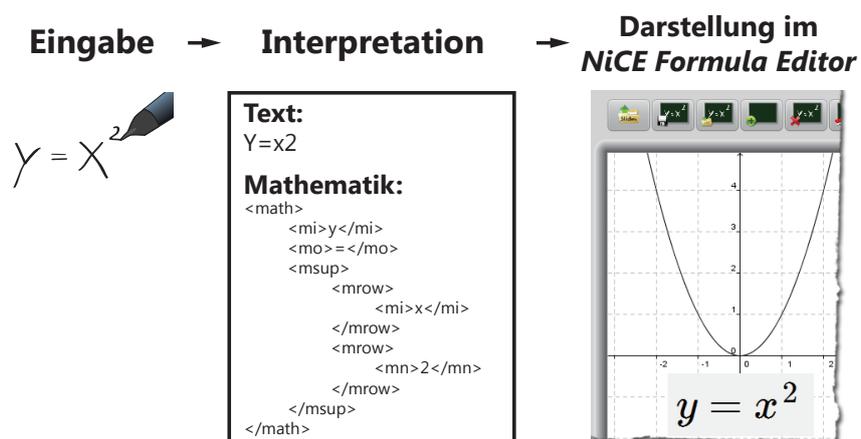


Abbildung 3.1: Ablauf im Überblick.

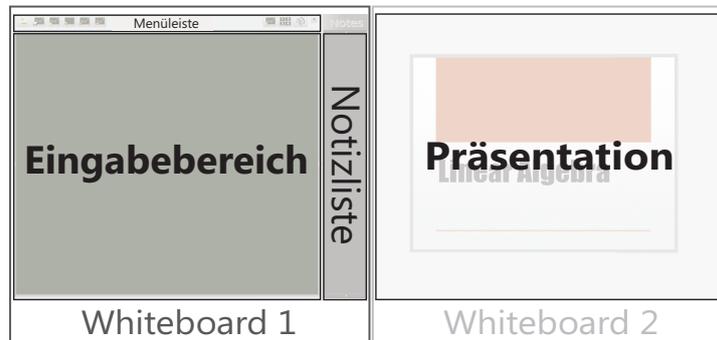


Abbildung 3.2: Layout des *NiCE Formula Editors*.

3.2 Layout

In Abbildung 3.2 ist das Layout der Oberfläche des *NiCE Formula Editors* zu sehen. Es besteht aus einem Bereich in dem die Präsentation angezeigt wird, einem Eingabebereich zur Notizerstellung, einer Menüleiste und bei geöffneter Präsentation, einer Liste der bereits erstellten Notizen.

Wie in Abbildung 3.3 dargestellt wird, besteht der Eingabebereich aus einer `System.Windows.Controls.InkCanvas`¹. Um die Text- bzw. Mathematik-Interpretation sowie die visuelle Repräsentation der mathematischen Ausdrücke in *GeoGebra* darstellen zu können, wird der Eingabebereich ausgeblendet und ein `System.Windows.Controls.Webbrowser`² an dessen Stelle eingeblendet. Der Webbrowser wird benötigt, um die verwendete Mathematiksoftware *GeoGebra*, die als *Java-Applet* zum Einsatz kommt, in den *NiCE Formula Editor* einbetten zu können. Zudem liefert die Interpretation eines mathematischen Ausdrucks eine für Webbrowser geeignete *MathML*-Repräsentation (vgl. Abschnitt 3.5.4) der Eingabe zurück.

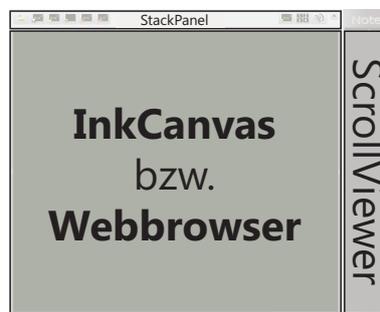


Abbildung 3.3: Hauptkomponenten der *NiCE Formula Editor*-Oberfläche.

¹<http://msdn.microsoft.com/en-us/library/system.windows.controls.inkcanvas.aspx>

²<http://msdn.microsoft.com/de-de/library/system.windows.controls.webbrowser.aspx>

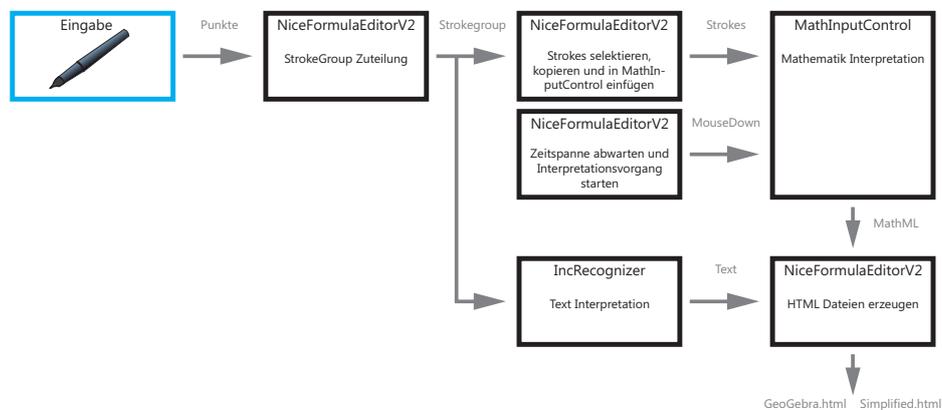


Abbildung 3.4: Zusammenspiel der Komponenten.

Weiters besteht die Menüleiste aus mehreren ineinander verschachtelten `System.Windows.Controls.StackPanel`³ und die Notizliste aus einem `System.Windows.Controls.ScrollViewer`⁴.

3.3 Ablauf

Im folgenden Abschnitt wird der Ablauf von der Erstellung einer Eingabe bis hin zur Darstellung in der Mathematiksoftware *GeoGebra* beschrieben. Als Beispiel dient dafür der mathematische Ausdruck $y = x^2$.

3.3.1 Eingabe erstellen

In Abbildung 3.4 ist ein Überblick über das Zusammenspiel der einzelnen Komponenten zu sehen. Wird mit dem digitalen Stift eine Eingabe getätigt, werden die als eine Menge von Punkten erfassten Daten an den *NiCE Formula Editor* übermittelt und zu einem `System.Windows.Ink.Stroke`⁵ zusammengefügt.

Der mathematische Ausdruck $y = x^2$ besteht, wie in Abbildung 3.5 zu sehen ist, aus sechs **Strokes** (*A* bis *F*). Um eine entsprechende Leserlichkeit zu erhalten werden nur Teile der in Wirklichkeit erfassten Punkte der einzelnen **Strokes** dargestellt.

Da zwei verschiedenen Systeme zur Interpretation der handgeschriebenen Eingaben zum Einsatz kommen (Text- bzw. Mathematik-Interpretation), benötigt das System zusätzlich den Typ der gerade erstellten Eingabe. Aus diesem Grund gibt es zwei verschiedene Modi für das *Pen Tool*, in denen man

³<http://msdn.microsoft.com/de-de/library/system.windows.controls.stackpanel.aspx>

⁴<http://msdn.microsoft.com/de-de/library/system.windows.controls.scrollviewer.aspx>

⁵[http://msdn.microsoft.com/en-us/library/system.windows.ink.stroke\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/system.windows.ink.stroke(v=vs.85).aspx)

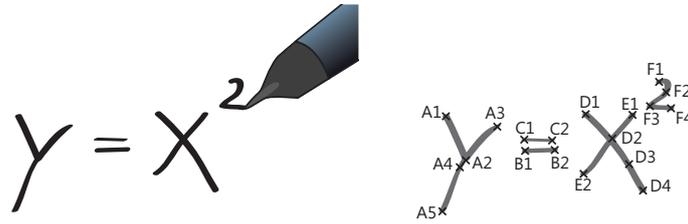


Abbildung 3.5: Erstellen der Eingabe.

Notizen erstellen kann. Den *Mathematik-Modus* zum Erstellen von mathematischen Ausdrücken und den *Text-Modus* zum Erstellen von Text. Wurde eine Notiz im falschen Modus erstellt, kann dieser in einer der beiden Interpretations-Ansichten umgestellt werden. In Abbildung 3.6 ist zu sehen wie das System, je nach Typ, einen Ausdruck unterschiedlich interpretiert. Der eingegebene Ausdruck *Test* wird, wenn er als Text interpretiert wird, korrekterweise als das Wort *Test* erkannt. Die Mathematik Interpretation würde in diesem Fall den Ausdruck $Test^{\Sigma}$ als Resultat liefern. Wird der mathematische Ausdruck $y = x^2$ fälschlicherweise als Text interpretiert, erhält man $4=+2$. Die Mathematik Interpretation hingegen liefert das korrekte Resultat $y = x^2$.

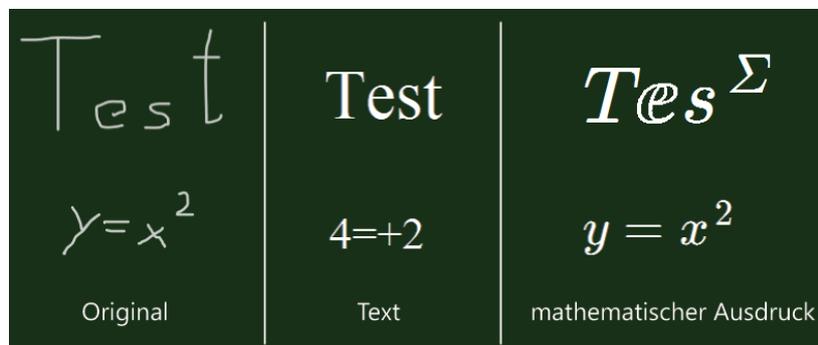


Abbildung 3.6: Unterschiedliche Interpretationen.

3.3.2 Einteilung der Strokes in Gruppen

Um die erfasste Eingabe korrekt zu interpretieren dürfen nur zusammengehörige **Strokes** an die beiden Interpreter weitergeleitet werden. Aus diesem Grund wird jeder **Stroke** einer **Strokegroup** (siehe Abschnitt 3.5.1) zugeteilt. Dies geschieht anhand zweier Kriterien: der örtlichen Nähe und der Zeitspanne zwischen den Erstellungszeitpunkten.

Jeder **Stroke** besitzt eine *Bounding Box* die seinen Grenzen entspricht.

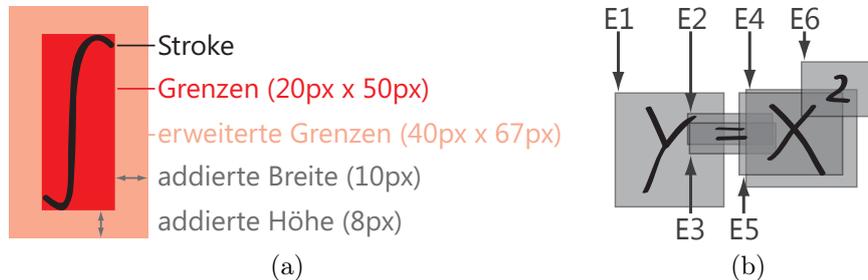


Abbildung 3.7: Grenzen eines einzelnen **Strokes** (a) und erweiterte Grenzen des mathematischen Ausdrucks $y = x^2$ (b).

Da sich diese Grenzen aber in den meisten Fällen nie überschneiden würden, müssen diese erweitert werden, damit nebeneinanderliegende **Strokes** als zusammengehörig erkannt werden. Dazu werden die Grenzen um einen gewissen Wert erweitert („aufgeblasen“), weshalb diese erweiterte Grenzen genannt werden. Überschneiden sich die in Abbildung 3.7 (a) dargestellten erweiterten Grenzen zweier **Strokes**, werden diese der gleichen **Strokegroup** zugeteilt. Wie man in Abbildung 3.7 (b) deutlich erkennen kann, existieren bei dem mathematischen Ausdruck $y = x^2$ zahlreiche Überschneidungen der sechs erweiterten Grenzen ($E1$ bis $E6$), wodurch alle **Strokes** vom System der gleichen **Strokegroup** zugeteilt werden.

	Berechnung	Min	Max
Addierte Breite (px)	$\frac{\text{Breite der Originalgrenzen}}{4}$	10	25
Addierte Höhe (px)	$\frac{\text{Höhe der Originalgrenzen}}{6}$	5	30

Tabelle 3.1: Bestimmung der Dimensionen der erweiterten Grenzen.

Die Ausmaße der erweiterten Grenzen sind, wie in Tabelle 3.1 gezeigt, von der Größe des **Strokes** abhängig. Als konkretes Beispiel dient der **Stroke** aus Abbildung 3.7 (a). Der **Stroke** besitzt eine Breite von 20px und eine Höhe von 40px. Die Ausmaße der erweiterten Grenze erhält man durch beidseitiges vergrößern der Breite um $\frac{20\text{px}}{4}$, sprich um 5px. Per Definition werden aber mindestens 10px dazu addiert, wodurch die Breite der erweiterten Grenze 40px entspricht. Das gleiche Verfahren wird für die erweiterte Höhe durchgeführt. Die ursprüngliche Höhe von 50px wird unten und oben jeweils um $\frac{50\text{px}}{6}$ erweitert, sprich um etwa 8px (8,33px). Die Höhe der erweiterten Grenze entspricht dann also 67px (66,66px). Diese Werte spielen eine wichtige Rolle für den reibungslosen Ablauf des Interpretationsvorganges, da sie dafür verantwortlich sind, dass die getätigten Eingaben korrekt als zusammengehörig bzw. im Gegenzug fälschlicherweise als zusammengehörig interpretiert wer-

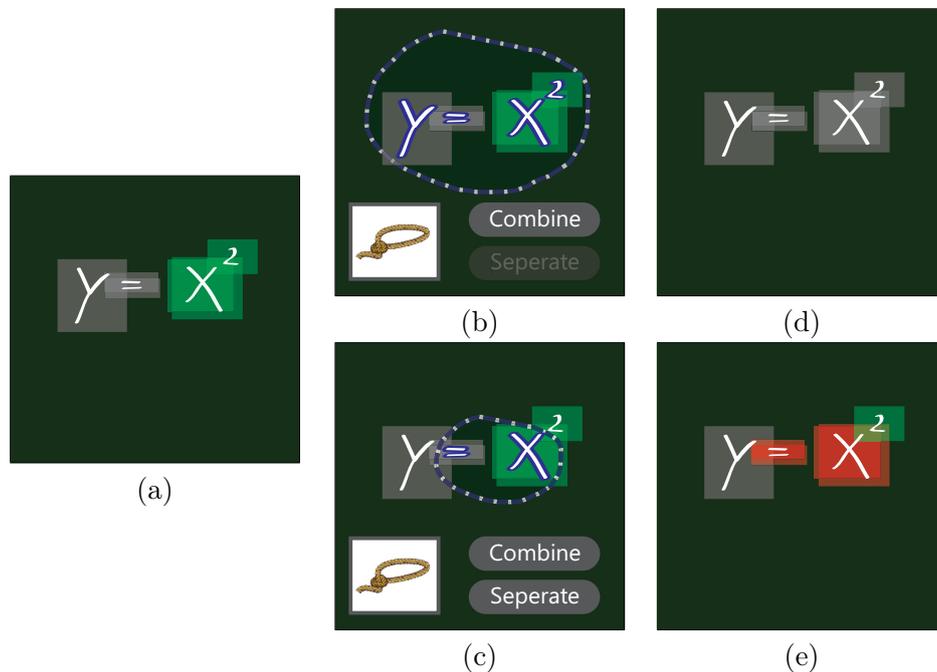


Abbildung 3.8: Bearbeitungsmöglichkeiten einer **Strokegroup**.

den. Aus diesem Grund war die Bereitstellung einer Möglichkeit zur Bearbeitung der Zusammengehörigkeit unumgänglich. Abbildung 3.8 zeigt Beispiele, wie sich durch das *Lasso Selection Tool* Teile einer **Strokegroup** in mehrere **Strokegroups** separieren bzw. mehrere **Strokegroups** zu einer zusammenfügen lassen. Die farbigen Rechtecke stellen dabei die erweiterten Grenzen der **Strokes** dar.

Die Ausgangssituation in (a) zeigt den auf Grund des zu großen Abstandes zwischen den einzelnen **Strokes** ungewollterweise in zwei **Strokegroups** aufgeteilten mathematischen Ausdruck $y = x^2$. In (b) und (c) wird mit dem *Lasso Selection Tool* eine Selektion getroffen. Die Selektion in (b) und (c) beinhalten jeweils zwei **Strokegroups** bzw. Teile davon. In (b) sind jedoch alle **Strokes** der beiden **Strokegroups** selektiert, wodurch nur die Möglichkeit die Selektion zu einer **Strokegroup** zusammenzufügen zur Verfügung steht. (d) zeigt den Ausdruck nach dem Zusammenfügen.

Wird eine Selektion wie in (c) getroffen, in der Teile einer **Strokegroups** selektiert sind, existiert die Möglichkeit diese Teile vom Rest zu separieren. Dazu erscheint, wie auch beim Zusammenfügen, neben der Selektion eine Schaltfläche um den Vorgang durchzuführen. Da in diesem Fall Teile von zwei verschiedenen **Strokegroups** selektiert worden sind, existiert zusätzlich wieder, wie bei (b), die Möglichkeit die beiden **Strokegroups** zu kombinieren. (e) zeigt schließlich den Ausdruck nach dem Separieren.



Abbildung 3.9: Bearbeitungsmöglichkeiten einer Strokegroup.

Abbildung 3.9 zeigt die Situation von Abbildung 3.8 (c) mit der eingblendeten *seperate* und *combine* Schaltfläche im *NiCE Formula Editor*.

Überschneiden sich die erweiterten Grenzen des erfassten **Strokes** mit keiner anderen und wurde er auch nicht innerhalb einer festgelegten Zeitspanne nach der letzten Eingabe erstellt, wird eine neue **Strokegroup** angelegt, zu der der **Stroke** hinzugefügt wird.

Das Resultat der Einteilung besteht aus einer Liste der **Strokegroups** für die aktuelle Notiz. In diesem Fall besteht diese für den mathematischen Ausdruck $y = x^2$ aus lediglich einer einzigen **Strokegroup**.

3.3.3 Mathematikinterpretation

Da der Interpretationsvorgang für mathematische Ausdrücke viel Rechenleistung in Anspruch nimmt und es dadurch zu Verzögerungen bei der Eingabe kommen kann, wird diese erst gestartet, wenn eine mathematische Interpretation angezeigt werden soll bzw. die Option *Interpretieren* aktiviert ist. Diese Option, genauso wie *Strokegroups anzeigen*, dient zur Unterstützung bei der Erstellung von Notizen. Wie in Abbildung 3.10 zu sehen ist, sorgt *Strokegroups anzeigen* dafür dass die Grenzen jedes **Strokes** als farbiges Rechteck angezeigt werden. Dabei werden alle Grenzen die der gleichen **Strokegroup** angehören mit der gleichen Farbe versehen.

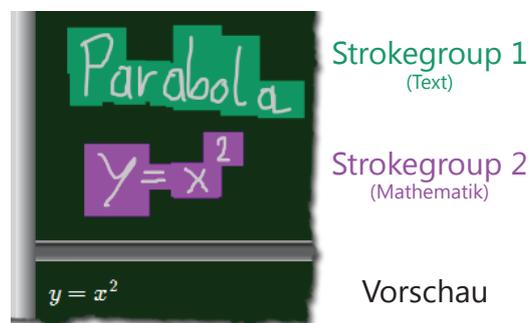


Abbildung 3.10: Angezeigte Strokegroups und eingblendete Interpretation.

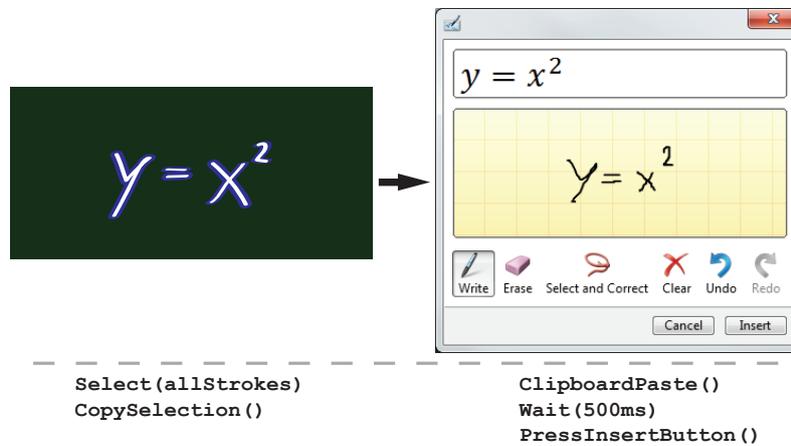


Abbildung 3.11: Übertragungsvorgang der Strokes in die MathInputControl.

Ist die Option *Interpretieren* aktiviert, wird im unteren Teil des Eingabebereichs eine Vorschau eingeblendet, welche die Interpretation der gerade getätigten Eingabe anzeigt. Diese Option sollte jedoch wegen des bereits erwähnten hohen Rechenaufwandes nur in Sonder- bzw. Spezialfällen wie beispielsweise bei komplexen mathematischen Ausdrücken verwendet werden. Des Weiteren sollte die Eingabegeschwindigkeit verringert werden damit es bei der Eingabe zu keinen Problemen kommt.

Zum Interpretieren werden, wie in Abbildung 3.11 dargestellt, die **Strokes** einer **Strokegroup** selektiert, kopiert und in die **MathInputControl** Komponente eingefügt. Um sicherzustellen dass alle **Strokes** eingefügt wurden, wird der Interpretationsvorgang erst nach $500ms$ gestartet.

Das Resultat der Interpretation ist die in Programm 3.1 dargestellte

```

1 <math>
2   <mi>y</mi>
3   <mo>=</mo>
4   <msup>
5     <mrow>
6       <mi>x</mi>
7     </mrow>
8     <mrow>
9       <mn>2</mn>
10    </mrow>
11  </msup>
12 </math>

```

Programm 3.1: $y = x^2$ in MathML.

*MathML*⁶ Repräsentation des Ausdrucks. Anschließend wird noch die vom Parser aus der *MathML* Repräsentation des Ausdrucks erstellte *GeoGebra*-Syntax der **Strokegroup** zugewiesen, womit alle Interpretationsvorgänge abgeschlossen sind. Das Resultat des Parsers für den mathematischen Ausdruck wäre $y=x^2$.

3.3.4 Textinterpretation

Nach der Zuteilung zu einer **Strokegroup**, wird die gesamte **Strokegroup** (neu) interpretiert. Die Textinterpretation wird dabei jedes Mal durchgeführt, unabhängig welchen Typ die **Strokegroup** besitzt. Der interpretierte Text wird anschließend der **Strokegroup** zugewiesen. Eingaben die den Typ Text besitzen, werden allerdings nicht an *GeoGebra* übergeben und somit nur als Textinterpretation dargestellt. Die Textinterpretation von $y = x^2$ wäre demnach $Y = x2$.

3.3.5 Verarbeitung der Ergebnisse

Sind die Interpretationsvorgänge aller **Strokegroups** abgeschlossen, werden mit den Ergebnissen zwei HTML-Dateien erstellt. Eine für die Text- und Mathematik-Interpretation und eine für die Darstellung in *GeoGebra*. Die Programmteile A.1 bzw. A.2 im Anhang A zeigen den gesamten HTML-Inhalt für den mathematischen Ausdruck $y = x^2$.

Da die gesamte Funktionalität von *GeoGebra* innerhalb des *NiCE Formula Editors* genutzt werden kann, muss dafür Sorge getragen werden, dass Inhalte die via *GeoGebra* erzeugt wurden, nicht verloren gehen. Aus diesem Grund werden zuletzt noch zusätzlich alle via *GeoGebra* erzeugten Elemente exportiert und mit der dazugehörigen Notiz verknüpft. Ein Beispiel dafür sind die in Abbildung 3.12 dargestellten Tangenten der Parabel, die mittels *GeoGebra* erzeugt wurden.

3.4 Speichern der erfassten Daten

Die handgeschriebenen Eingaben werden als **.isf** (*Ink Serialized Format*) Dateien hinterlegt. Dieses Format wird dazu verwendet um **ink**-Inhalte zu speichern. Ein **ink**-Objekt besteht aus mehreren **Stroke**-Objekten, die wiederum aus Listen von Punkten bestehen. Eine **.isf** Datei beinhaltet nun die Koordinaten aller mit dem digitalen Stift erzeugten Eingaben einer Notiz, sowie zusätzliche Informationen über Aufpressdruck, Neigungswinkel und Erstellungszeitpunkt für jeden Punkt.

Um jedoch eine Notiz komplett rekonstruieren zu können werden weitere Informationen, wie die **Strokegroup**-Zugehörigkeit jedes **Strokes**, sowie ob

⁶<http://www.w3.org/Math/>

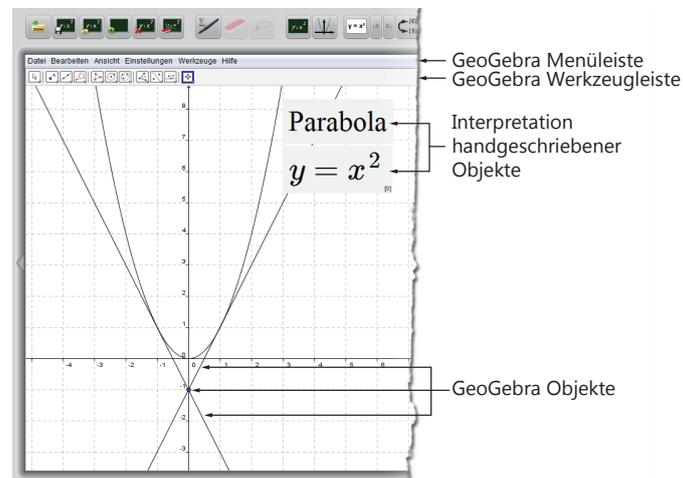


Abbildung 3.12: Handgeschriebene und mittels *GeoGebra* zusätzlich erstellte Inhalte.

es sich bei einer **Strokegroup** um eine Formel handelt und welche Schriftgröße die Interpretation besitzt, benötigt.

Gehört die Notiz zu einer Folie einer Präsentation, werden diese Informationen in der `data.xml` Datei gespeichert. Der Programmteil A.3 im Anhang A zeigt einen Auszug aus der `data.xml` Datei. Handelt es sich um eine einzelne Notiz, wird eine `.info` Datei erstellt, welche diese Informationen beinhaltet. Im Folgenden sieht man einen Auszug aus einer `.info` Datei. Die Zahlen wurden zur Verbesserung der Leserlichkeit durch Variablen ersetzt:

```

1 <?xml version="1.0"?>
2 <additionalInformation>
3   <transformation>
4     xPos;yPos;width;height
5   </transformation>
6   <strokeGroups>
7     <strokeGroup id="0" isFormula="True" zoomLevel="20">
8       <stroke>s1_x1;s1_y1;s1_x2;s1_y2;</stroke>
9       <stroke>s2_x1;s2_y1;s2_x2;s2_y2;</stroke>
10      <stroke>s3_x1;s3_y1;s3_x2;s3_y2;</stroke>
11    </strokeGroup>
12    ...
13  </strokeGroups>
14 </additionalInformation>

```

Programm 3.2: Auszug aus einer `.info` Datei.

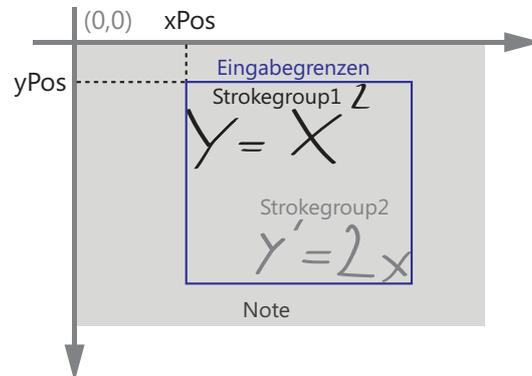


Abbildung 3.13: Translationswerte einer Notiz.

3.4.1 Inhalt der .info Datei

`xPos`, `yPos`, `width` und `height` des `transformation`-Elements aus dem angeführten Dateiauszug entsprechen dabei den wie in Abbildung 3.13 dargestellten Eigenschaften der Eingabegrenzen. Diese Information muss explizit gespeichert werden, da die `Strokes` beim Speichern als `.isf` Datei immer an der linken oberen Ecke ausgerichtet werden und dadurch sonst die Information über die Position auf der Notiz verloren gehen würde. Beim Laden der handgeschriebenen Notiz wird eine Translation um `xPos` bzw. `yPos` auf die gespeicherten `Strokes` angewandt, um die Originalposition der `Strokes` auf einer Notiz zu erhalten.

Nach dem `transformation`-Element sind die einzelnen `Strokegroups` mit der jeweiligen `id` (`id` Attribut), der Eigenschaft ob es sich um eine Formel handelt (`isFormula` Attribut) und der Schriftgröße (`zoomlevel` Attribut) aufgelistet. Jedes `Strokegroup` Element beinhaltet wiederum die Koordinaten der Grenzen jedes einzelnen `Strokes`, welche zur Rekonstruktion der `Strokegroups` aus der `.isf` Datei benötigt werden.

3.4.2 Rekonstruktion der Strokegroups

Um die `Strokegroups` zu rekonstruieren, werden die Koordinaten der einzelnen `Strokes` aus der `.isf` Datei mit den Positionswerten aus der `data.xml` bzw. `.info` Datei verglichen. Liegt der Positionsunterschied unter dem gewählten Schwellwert von 4px, gilt das Vergleichskriterium als erfüllt. Dadurch können dem `Stroke` aus der `.isf` Datei die hinterlegten Eigenschaften zugewiesen werden.

3.4.3 Dateistruktur

Die in Abschnitt 3.4 Anfangs erwähnten Informationen, die benötigt werden um eine Präsentation mit den dazugehörigen Notizen zu beschreiben, werden

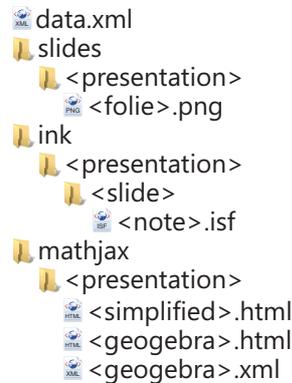


Abbildung 3.14: Dateistruktur.

wie in Abbildung 3.14 gezeigt hinterlegt.

Wurde eine Präsentation wie in Abschnitt 3.5.3 beschrieben in eine Reihe von Bildern konvertiert, ist diese Bildserie unter `slides/<präsentation>` abgelegt. Unter `ink/<präsentation>/<slidenummer>` werden die dazugehörigen Notizen als `.isf` Dateien hinterlegt. Die in Abschnitt 3.3.5 erwähnten HTML- und XML-Dateien werden unter `mathjax/<präsentation>` gespeichert.

3.5 Verwendete Komponenten

Im folgenden Abschnitt werden die Klassenarchitektur und die verwendeten Komponenten beschrieben. Zu den verwendeten Komponenten zählen der Präsentations-Konverter, die zwei Interpreter, der eingebundene Webbrowser, die Mathematiksoftware *GeoGebra* und die Open Source *JavaScript* Bibliothek *Math.Jax*

3.5.1 Klassenarchitektur

In Abbildung 3.15 sind Teile der `Note`-, `Strokegroup`- und `Presentation`-Klasse dargestellt.

Note-Klasse

Diese Klasse basiert auf einer `System.Windows.Controls.InkCanvas`⁷ und dient zur Beschreibung einer Notiz zu einer Folie einer Präsentation. Darüber hinaus besitzt jedes Objekt dieser Klasse die Id der verknüpften Folie (`belongsToPage`), die Repräsentation des *GeoGebra* Inhalts (`geogebraXML`),

⁷<http://msdn.microsoft.com/en-us/library/system.windows.controls.inkcanvas.aspx>

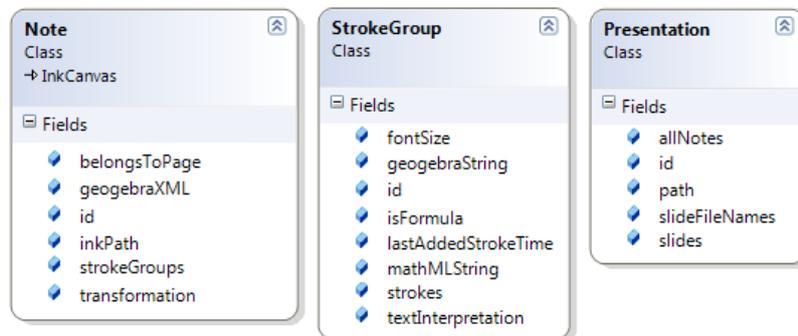


Abbildung 3.15: Grundlegende Inhalte der Klassen.

eine innerhalb der Präsentation eindeutige Id (`id`), den Pfad zum Speicherort der `.isf` Dateien (`inkPath`), eine Liste der enthaltenen `Strokegroups` (`strokeGroups`) und die Translationswerte der Notiz (`transformation`).

Strokegroup-Klasse

Diese Klasse dient zur Beschreibung eines zusammengehörigen Ausdrucks der (meist) aus mehreren `Strokes` besteht. Jedes `Strokegroup`-Objekt beinhaltet die Schriftgröße (`fontSize`), welche die Interpretation der `Strokegroup` im Webbrowser besitzt, die Interpretation des Ausdrucks in *GeoGebra*-Syntax (`geogebraString`), eine innerhalb der Notiz eindeutige Id (`id`), den Typ des Ausdrucks (`isFormula`), den Zeitpunkt zu dem der letzte `Stroke` hinzugefügt wurde (`lastAddedStrokeTime`), die *MathML* Interpretation des Ausdrucks (`mathMLString`), die enthaltenen `Strokes` (`strokes`) und die Interpretation des Ausdrucks als Text (`textInterpretation`).

Presentation-Klasse

Diese Klasse dient zur Beschreibung einer einzelnen Präsentation. Sie beinhaltet eine Liste der Ids der enthaltenen Notizen (`allNotes`), eine eindeutige Id (`id`), den Pfad zur ursprünglichen Präsentation (`path`), einer Liste der Pfade zu den Bildrepräsentationen der Folien (`slideFileNames`) und eine Liste aus `Slide`-Objekten.

Die `allNotes`-Liste beinhaltet die Reihenfolge der Notizen, in der sie erstellt wurden. Diese Reihenfolge spielt eine wichtige Rolle um die Präsentation eins zu eins wiederzugeben, da man nicht automatisch davon ausgehen kann, dass Notizen zu späteren Folien erst später im Vortrag vorgekommen sind. Auf diese Problematik wird in Abschnitt 3.6.2 genauer eingegangen.

<i>Element</i>	<i>Beschreibung</i>	<i>Beispiel</i>
<code>mrow</code>	Horizontal ausgerichtete Gruppe	x^{a+b}
<code>munderover</code>	Element mit Hoch und Tiefstellung	\int_0^1
<code>mfenced</code>	Eingeklammertes Element	$\sin(x)$
<code>mo</code>	Operator	$+$
<code>mn</code>	Numerisches Element	1
<code>mi</code>	Symbol, Kennzeichner	π
<code>mfrac</code>	Bruch	$\frac{1}{2}$
<code>msup</code>	Hochgestelltes Element	x^2
<code>msqrt</code>	Quadratwurzel	$\sqrt{2}$
<code>mroot</code>	Wurzel	$\sqrt[3]{2}$
<code>mtable</code>	Tabelle, Matrix	$\begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{pmatrix}$
<code>mtr</code>	Tabellenreihe	
<code>mtd</code>	Tabellenzelle	

Tabelle 3.2: Unterstützte Elemente.

3.5.2 MathML Parser

Derzeit gibt es noch keine offizielle Unterstützung für die Umwandlung von *MathML* in *GeoGebra*-Syntax. Durch das große Spektrum an verschiedenen mathematischen Ausdrücken kann es passieren, dass Ausdrücke zwar richtig interpretiert aber von dem erstellten Parser falsch umgewandelt werden. Anders ausgedrückt, die Eingabe wird korrekt in *MathML* umgewandelt, aber bei der Übersetzung in *GeoGebra*-Syntax kommt es zu Problemen.

Des Weiteren sind die derzeit implementierten Funktionen des Parsers noch einigermaßen limitiert, können aber jederzeit ohne größeren Aufwand erweitert werden.

Funktionsweise des Parsers

Der vom Mathematikinterpreter ermittelte *MathML*-Ausdruck, wird an einen `System.Xml.XmlReader`⁸ übergeben und anschließend Node für Node abgearbeitet. Dabei wird zwischen den `System.Xml.XmlNodeType`⁹ Typen `Element`, `Text` und `EndElement` unterschieden. In Tabelle 3.2 sind die vom Parser unterstützten `Element` Typen aufgelistet. Eine vollständige Liste der *MathML*-Elemente (Version 3.0) findet man auf der offiziellen Seite des *W3C*¹⁰.

Während der Abarbeitung der einzelnen Nodes, wird jedem `Element` ein Eltern-Element und der beinhaltende Text zugeteilt. Abbildung 3.16 zeigt das Resultat dieser Zuteilung für den mathematischen Ausdruck $y = x^2$.

⁸<http://msdn.microsoft.com/en-us/library/system.xml.xmlreader.aspx>

⁹<http://msdn.microsoft.com/en-us/library/system.xml.xmlreader.nodetype.aspx>

¹⁰<http://www.w3.org/TR/MathML3/>

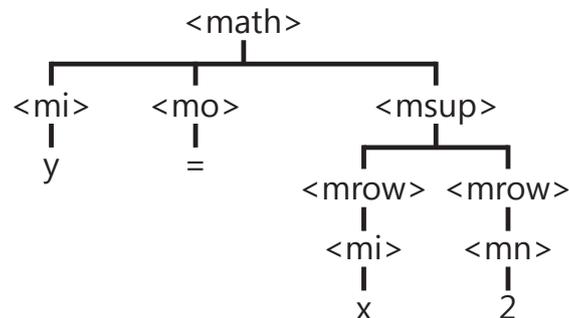


Abbildung 3.16: Objekthierarchie der *MathML* Elemente.

Dabei ist anzumerken, dass lediglich die **Element** Typen der untersten Ebene (`mi`, `mo` und `mn`) einen Text beinhalten, jedes andere **Element** beinhalten höchstens eine Liste mit Kind-Elementen. Bei der Textzuteilung ist das Fehlerpotential derzeit noch am größten, da viele Ausdrücke noch nicht unterstützt werden. Dies trifft verstärkt auf Symbole zu, da derzeit lediglich α , π und d erkannt werden. Neben den Operatoren der Grundrechnungsarten werden \int und \sum unterstützt.

3.5.3 Präsentations-Konverter

Bevor Notizen zu einer Präsentation erstellt werden können, muss die Präsentation zuerst in eine Reihe von Bildern konvertiert werden. Durch die Verwendung der `Microsoft.Office.Interop.PowerPoint`¹¹ COM-Komponente kann eine Microsoft PowerPoint Präsentation automatisch in eine Reihe von Portable Network Graphics konvertiert werden. Dafür muss allerdings Microsoft PowerPoint auf dem Rechner installiert sein. Ist dies nicht der Fall, muss der Benutzer aus den einzelnen Folien der Präsentation manuell eine Bildserie erstellen. Getestete Dateiformate, die unterstützt werden sind in Tabelle 3.3 aufgelistet.

Präsentationsdateien	.ppt, .pptx, .odp
Bilddateien (Bildserie)	.png, .jpeg, .jpg, .tif, .bmp, .gif

Tabelle 3.3: Unterstützte Dateiformate.

¹¹<http://msdn.microsoft.com/en-us/library/microsoft.office.interop.powerpoint.aspx>

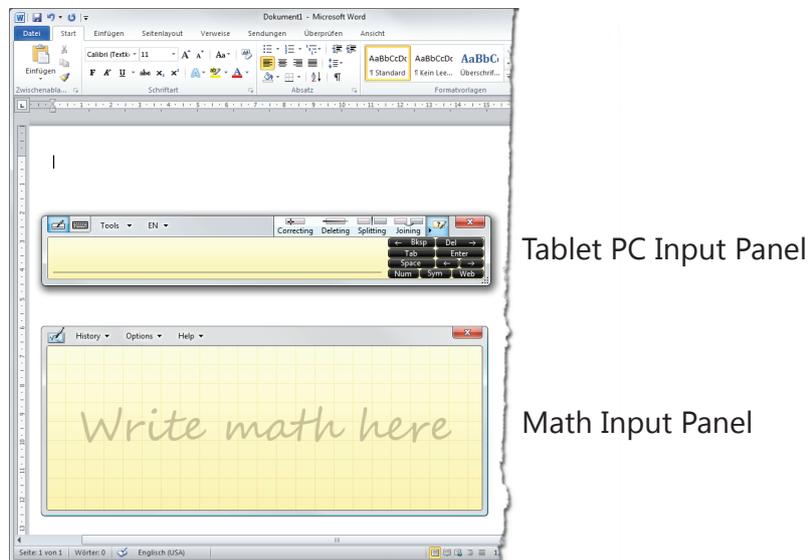


Abbildung 3.17: Microsoft Word mit geöffnetem *Tablet PC-* und *Math Input Panel*.

3.5.4 Interpreter

Für die Interpretation der handgeschriebenen Eingabe werden zwei verschiedene Interpreter verwendet. Ein Recognizer des *Microsoft.Ink-Assembly*¹² zur Texterkennung und der *Math Recognizer* von Windows 7.

Diese beiden Recognizer werden in Windows 7 von dem in Abbildung 3.17 gezeigten *Tablet PC Input Panel*¹³ und dem *Math Input Panel*¹⁴ zur Interpretation von Eingaben verwendet.

Das *Tablet PC Input Panel* wandelt Handschrift in maschinengeschriebene Schrift um. Es besitzt ein Eingabefeld, auf dem man, wie in Abbildung 3.18 zu sehen, wie auf einem normalen Stück Papier schreiben kann.



Abbildung 3.18: Handschriftliche Eingabe des Wortes *Parabola*.

¹²<http://msdn.microsoft.com/en-us/library/ms826516.aspx>

¹³<http://windows.microsoft.com/en-US/windows7/Using-the-writing-pad-and-touch-keyboard-in-Tablet-PC-Input-Panel>

¹⁴<http://windows.microsoft.com/en-US/windows7/Use-Math-Input-Panel-to-write-and-correct-math-equations>

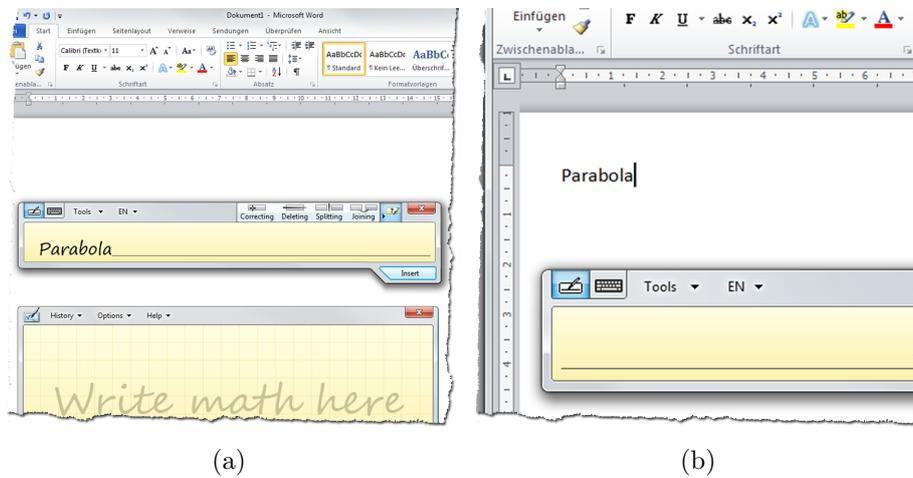


Abbildung 3.19: Interpretationsvorschlag im *Tablet PC Input Panel* (a) und als eingefügter Text (b).

Das Panel bietet neben Gesten zum Trennen bzw. Zusammenfügen von Wörtern weiters die Möglichkeit durch Durchstreichen von getätigten Eingaben, diese wieder zu löschen. Des Weiteren verfügt es, wie aus modernen Webbrowsern bekannt, über Autovervollständigung und liefert alternative Textvorschläge, wenn es zu Korrekturen kommen sollte. Zusätzlich kann das *Tablet PC Input Panel* auf die eigene Handschrift trainiert werden. Dabei lernt das System die verwendeten Worte und deren Schreibweise, wodurch die Resultate der Handschrifterkennung deutlich verbessert werden.

Abbildung 3.19 (a) zeigt die vorgeschlagene Interpretation der vorher getätigten Eingabe. Durch betätigen der **Insert**-Schaltfläche wird, wie in (b) zu sehen, der interpretierte Text in das Word Dokument eingefügt. Auf diese Weise kann Text für jede Anwendung in Windows 7 eingegeben werden.

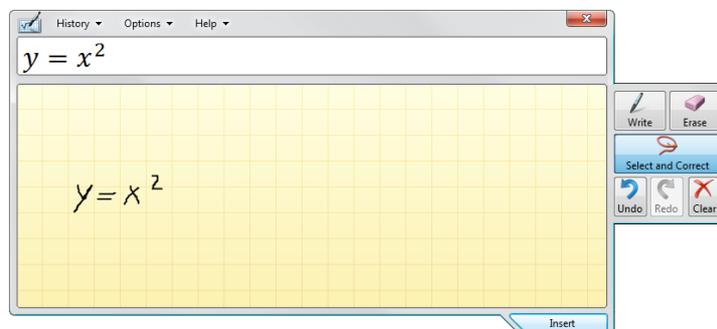


Abbildung 3.20: Handgeschriebener Ausdruck und Interpretationsvorschlag im *Math Input Panel*.

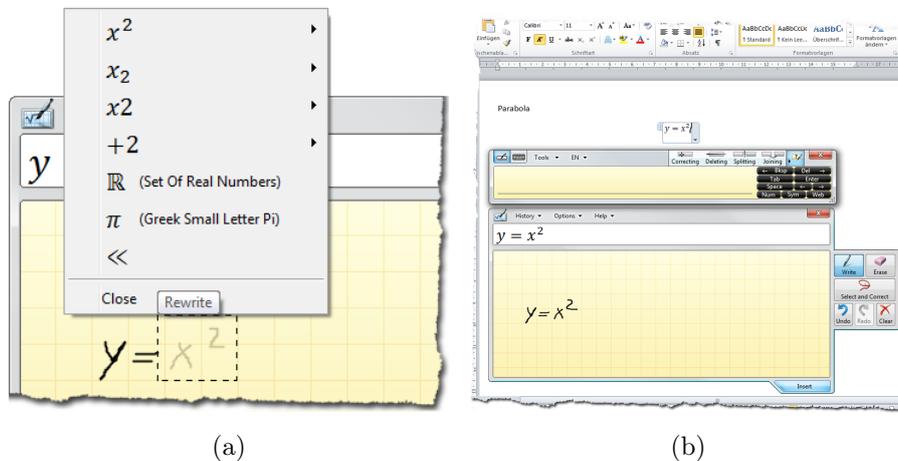


Abbildung 3.21: Alternativvorschläge für x^2 (a) und der in ein Word Dokument eingefügte Ausdruck.

Das *Math Input Panel* ist das Werkzeug zur Erkennung von handgeschriebenen mathematischen Ausdrücken in Windows 7. Wie in Abbildung 3.20 zu sehen, besitzt das *Math Input Panel*, wie das *Tablet PC Input Panel*, ebenfalls einen Eingabebereich auf dem handgeschriebene Ausdrücke erstellt werden können. Nach der Eingabe erscheint der Interpretationsvorschlag direkt über dem Eingabebereich.

Entspricht die Interpretation nicht dem gewünschten Resultat, kann die Eingabe durch die Verwendung des *Select and Correct* Werkzeuges bearbeitet werden. Markiert man damit eine erstellte Eingabe, wie in Abbildung 3.21 (a) zu sehen, erhält man ein Liste alternativer Interpretationsmöglichkeiten angeboten. Befindet sich das erwartete Resultat immer noch nicht darunter, kann man den Ausdruck einfach überschreiben, wodurch der Interpretationsvorgang neu durchgeführt wird. Das Resultat der Interpretation ist eine Repräsentation des Ausdrucks in *MathML*. Durch betätigen der *Insert*-Schaltfläche wird, wie in Abbildung 3.21 (b) zu sehen, der interpretierte mathematische Ausdruck in das Word Dokument eingefügt. Dies ist allerdings nur möglich, da Microsoft Word die Anzeige von *MathML* unterstützt. Unter anderen Programmen, die das Format nicht unterstützen, hat das Betätigen der Schaltfläche keine Auswirkungen.

Verwendung des Text Interpreter im NiCE Formula Editor

Der Text Interpreter teilt die erfasste Eingabe in einzelne Segmente und wandelt diese in Text um. Programmausschnitt 3.3 zeigt die grundsätzlichen Einstellungen die beim Anlegen des Interpreters getroffen werden.

Beim Anlegen des Interpreters muss ein *Language Code Identifier*, kurz

```

1 int LCID = 1033;
2 Recognizers inkRecognizers = new Recognizers();
3 Recognizer recognizer = inkRecognizers.GetDefaultRecognizer(LCID);

```

Programm 3.3: InkRecognizer-Einbindung.

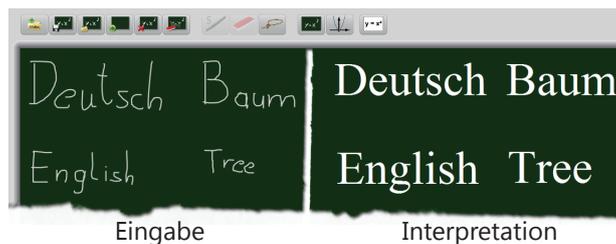


Abbildung 3.22: Text-Interpretationen.

LCID, angegeben werden, der festlegt welche Sprache er verwenden soll. Derzeit verwendet der *NiCE Formula Editor* standardmäßig Englisch mit einem *LCID* von 1033. In Abbildung 3.22 erkennt man allerdings, dass sogar deutsche Ausdrücke korrekt interpretiert werden.

Verwendung des Mathematik Interpreter im NiCE Formula Editor

Durch die Einbindung der *micautLib* COM-Bibliothek, kann der in Windows 7 eingebaute *Math Recognizer* als eigenständige Komponente im *NiCE Formula Editor* verwendet werden.

Programmausschnitt 3.4 zeigt die Einstellungen, die beim Anlegen der Komponente getroffen werden. Durch das Setzen von `EnableAutoGrow(true)` passt sich die Größe der `MathInputControl` immer an den zu interpretierenden Inhalt an. Weiters wird ein `EventListener` auf das `InsertEvent` gesetzt. Dieser reagiert auf das betätigen der `Insert`-Schaltfläche und bekommt die Interpretation des Ausdrucks in *MathML* übergeben.

In Abbildung 3.23 ist die `MathInputControl` Komponente abgebildet, die eine große Ähnlichkeit mit dem in Abbildung 3.20 dargestellten *Math Input Panel* aufweist.

```

1 MathInputControl ctrl = new MathInputControl();
2 ctrl.EnableAutoGrow(true);
3 ctrl.Insert += new _IMathInputControlEvents_InsertEventHandler(Insert);
4 ctrl.Show();

```

Programm 3.4: MathInputControl-Einbindung.

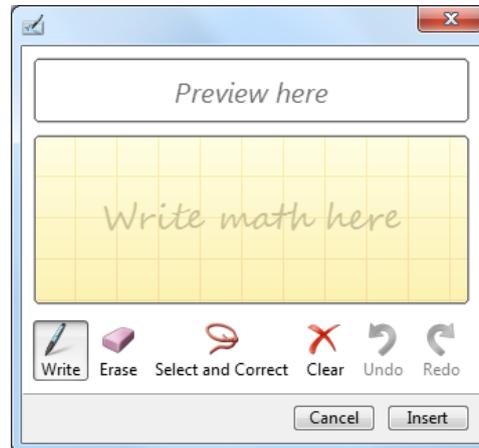


Abbildung 3.23: Die MathInputControl Komponente.

In Abbildung 3.24 sind die Ergebnisse der Text- und Mathematik-Interpretationen während der Erstellung des Ausdrucks $y = x^2$ dargestellt. Nach der Erstellung des ersten **Strokes** liefern beide Interprete noch y als Ergebnis. Nachdem der erste Strich des Gleichheitszeichens hinzugefügt wurde, liefert die Interpretation als Text $Y-$, als mathematischer Ausdruck y_- . Mit fertiggestellten Gleichheitszeichen, erkennt keiner der beiden Interpreten die Eingabe korrekterweise als $y =$ sondern interpretieren sie als 45 bzw. \leftarrow . Nach dem hinzufügen der ersten Hälfte von x liefert die Text-Interpretation $Y = 1$, nach der zweiten Hälfte EX und schließlich nach der Fertigstellung des Ausdrucks $Y = x2$. Die Mathematik-Interpretation interpretiert zwar für die erste Hälfte des x noch als 2, nach der Fertigstellung von x und dem hinzufügen des Exponenten liefert sie aber korrekte Ergebnisse.

Eingabe	Interpretation	
	Text	Mathematik
Y	y	y
$Y-$	$Y-$	y_-
$Y=$	45	\leftarrow
$Y = \backslash$	$Y=1$	$y=2$
$Y = X$	EX	$y=x$
$Y = X^2$	$Y=x2$	$y=x^2$

Abbildung 3.24: Interpretationsschritte für $y = x^2$.

3.5.5 Webbrowser

Da die verwendete Mathematiksoftware *GeoGebra* als *Java-Applet* zur Anwendung kommt, musste zur Darstellung ein Webbrowser eingebunden werden. Dies war insofern ein glücklicher Zufall, da der Mathematik Interpreter als Resultat eine *MathML* Repräsentation der Ausdrücke liefert und dieses Format zum Teil genau für die Darstellung in Webseiten entwickelt wurde.

Wie in Tabelle 3.4 zu sehen, gibt es allerdings noch Probleme bzw. Einschränkungen bezüglich der Wahl des Browsers, um *MathML* korrekt anzeigen zu können.

Windows	IE 5.0 <i>Techexplorer</i> Plugin IE 5.5 mit <i>MathPlayer</i> bzw. <i>Techexplorer</i> Plugin IE 6.0+, optional mit <i>MathPlayer</i> or <i>Techexplorer</i> Plugin Netscape 6.1 mit <i>Techexplorer</i> Plugin Netscape 7.0+ Amaya, alle Versionen (nur <i>Presentation MathML</i>) Mozilla 0.9.9+
Macintosh	IE 5.0+ mit <i>Techexplorer</i> Plugin Mozilla 0.9.9+
Linux/Unix	Netscape 6.1 mit <i>Techexplorer</i> Plugin Netscape 7.0+ Mozilla 0.9.9+ Amaya, alle Versionen (nur <i>Presentation MathML</i>)

Tabelle 3.4: *MathML*-Browserkompatibilität¹⁵.

MathML Darstellungsproblem

MathML genießt zwar den Status einer World Wide Web Consortium (W3C) Empfehlung, wird jedoch noch immer nicht von allen Webbrowsern unterstützt. Der verwendete `System.Windows.Controls.Webbrowser` basiert auf dem Internet Explorer von Microsoft und kann, wie in Abbildung 3.25 zu sehen, *MathML* nur unter der Verwendung eines speziellen Plugins, wie beispielsweise *MathPlayer*¹⁶ oder *Techexplorer*¹⁷, anzeigen. Aus diesem Grund wurde anfangs sogar die Verwendung eines alternativen Webbrowsers in Erwägung gezogen bzw. kam dieser später sogar kurzzeitig zum Einsatz.

Als Alternative wurde die Open Source Komponente *GeckoFX*¹⁸ verwendet. Durch die während der Entwicklung gefundene Open Source *JavaScript*

¹⁵<http://www.w3.org/Math/XSL/>

¹⁶<http://www.dessci.com/en/products/mathplayer/>

¹⁷<http://www.integretechpub.com/techexplorer/>

¹⁸<http://www.geckofx.org/>

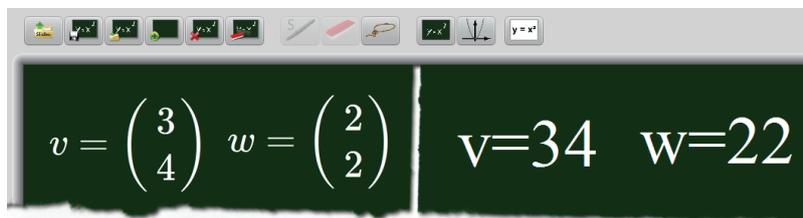


Abbildung 3.25: Darstellung mit (links) und ohne *MathJax* (rechts).

Bibliothek *MathJax*, konnte jedoch wieder der Standard Webbrowser verwendet werden, da durch *MathJax* die Darstellung von *MathML* ermöglicht wurde. Der alternative Webbrowser würde zwar ohne *MathJax* auskommen, benötigt allerdings einen größeren Konfigurationsaufwand.

3.5.6 MathJax

*MathJax*¹⁹ ist eine Open Source *JavaScript* Bibliothek mit der mathematische Ausdrücke in allen modernen Browsern dargestellt werden können. Durch die Verwendung von *MathJax* wird das *MathML* Darstellungsproblem in dem verwendeten Webbrowser behoben.

MathJax wurde entwickelt um mathematische Ausdrücke einfach und schnell auf Webseiten einfügen zu können. Der Ausdruck kann dazu in gängigen Formaten wie *MathML* oder *LaTeX*²⁰ erstellt werden. Selbst wenn der Webbrowser standardmäßig das Format nicht unterstützt, liefert *MathJax* trotzdem qualitativ hochwertige Ergebnisse. Existieren für den Webbrowser zusätzlich professionelle Mathematik Schriftarten, beispielsweise das Schriftartenpaket *STIX*²¹, wird das Ergebnis nochmals verbessert.

Aufbau und Funktion von MathJax

MathJax ist in mehrere Komponenten aufgeteilt, einem Präprozessor, dem Ein- und Ausgabe Prozessor, den sogenannten *jax* und dem *MathJax Hub*, das die einzelnen Komponenten verwaltet und die Verbindung zwischen ihnen herstellt.

MathJax durchsucht die Webseite nach speziellen Elementen, die mathematische Ausdrücke beinhalten. Sind die mathematischen Inhalte nicht innerhalb solcher speziellen Elemente, da beispielsweise die standardisierte *TeX* Schreibweise innerhalb zwei \$ bzw. \$\$-Symbole verwendet wurde, können durch den Einsatz des passenden Präprozessors diese Inhalte erkannt und innerhalb spezieller Elemente platziert werden. Präprozessoren existie-

¹⁹<http://www.mathjax.org/>

²⁰<http://www.latex-project.org/>

²¹<http://www.stixfonts.org/>

ren für die *TeX*, *MathML* und *jsMath* Notation. Deren Einsatz verlangsamt allerdings den Umwandlungsprozess.

Nachdem die Elemente gefunden wurden, wird für jedes ein Eingabe *jax* allokiert, der den Ausdruck in ein internes Format, dem *element jax*, konvertiert. Dieses Format wird anschließend von einem Ausgabe *jax* das gewünschte Ausgabeformat umgewandelt.

Anwendung im NiCE Formula Editor

Da der eingebundene Webbrowser *MathML* nativ nicht unterstützt und als Eingabeformat ausschließlich Ausdrücke in *MathML* verwendet wird, kommt die `MML_HTMLorMML-full` Konfiguration zum Einsatz. Dabei wird dem Eingabe *jax* der *MathML* Ausdruck übergeben, der anschließend vom Ausgabe *jax* in HTML-CSS Form umgewandelt wird.

3.5.7 Mathematiksoftware

NiCE Formula Editor verwendet *GeoGebra* als ein in eine HTML-Datei eingebundenes *Java-Applet*, mit dem via *JavaScript* kommuniziert werden kann. Durch die Verwendung eines *Java-Applets* kam es jedoch zu Problemen bei der Darstellung der interpretierten Inhalte.

Applet Einbindung

Bei der Darstellung des *Java-Applets* von *GeoGebra* im eingebundenen Webbrowser erschien auf Grund der nicht modifizierbaren Sicherheitseinstellungen des Browsers immer der Warnhinweis:

„*To help protect your security, Internet Explorer has restricted this webpage from running scripts or ActiveX controls that could access your computer. Click here for options...*“

Weiters wurde sogar die Ausführung des *Java-Applets* verhindert. Erst durch die Angabe des für die Sicherheitszonen relevanten *Mark of the Web*²² Kommentars `<!-- saved from url=(0014)about:internet -->`, konnte diese Sicherheitseinstellung umgangen und dadurch das *Java-Applet* korrekt dargestellt werden.

z-Index Problematik

Da *Java-Applets* standardmäßig immer im Vordergrund dargestellt werden, stellte sich die Positionierung der interpretierten Ausdrücke als relativ kompliziert heraus. Die interpretierten Inhalte werden in `div` Elementen angezeigt und würden immer von dem *Java-Applet* überdeckt werden. Durch die Verwendung von `iFrames` ohne Inhalt, können jedoch „Löcher“ in das *Java-Applet* „gestanzt“ werden.

²²[http://msdn.microsoft.com/en-us/library/ms537628\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/ms537628(v=vs.85).aspx)

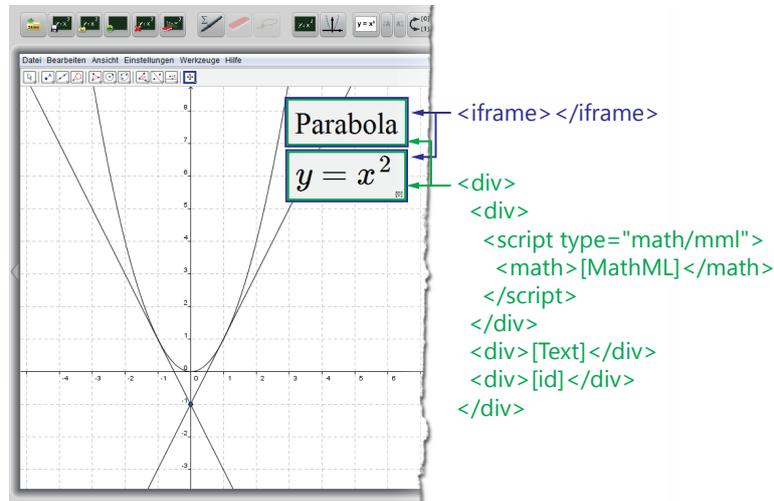


Abbildung 3.26: Überlagerung des *Java-Applet* durch *iFrames*.

Wie in Abbildung 3.26 zu sehen, besitzen jeweils ein `iFrame` und ein `div` Element die gleiche Position und Dimension. Dadurch wirkt es so, als würde sich das `div` Element, mit dem interpretierten Inhalt, vor dem *Java-Applet* befinden.

3.6 Probleme

Schon in der Konzeptionsphase, vor allem aber während der Entwicklung, traten einige Probleme auf. Eine Beschreibung dieser Probleme sowie die gefundenen Lösungen werden im folgenden Abschnitt beschrieben.

3.6.1 MathInputControl Probleme

Leider existiert für die `MathInputControl` Komponente keine Dokumentation. Es finden sich auch nahezu keine Aufzeichnungen von Anwendungsbeispielen und Erfahrungsberichten anderer Entwickler.

Probleme auf Komponentenebene

Das größte Problem stellt die fehlende Möglichkeit den Interpretationsvorgang mittels der Komponente zu starten dar. Das `InsertEvent` kann also nur durch betätigen der `Insert`-Schaltfläche abgesetzt werden (vgl. Abbildung 3.23). Die einzige Möglichkeit die Schaltfläche zu bestätigen, besteht aus dem Senden eines `Mousedown`-Befehls auf Betriebssystemebene. Dafür wird einerseits der `Window-Handle` der `MathInputControl` Komponente und die Koordinaten der Schaltfläche benötigt. Diese Position ist lei-

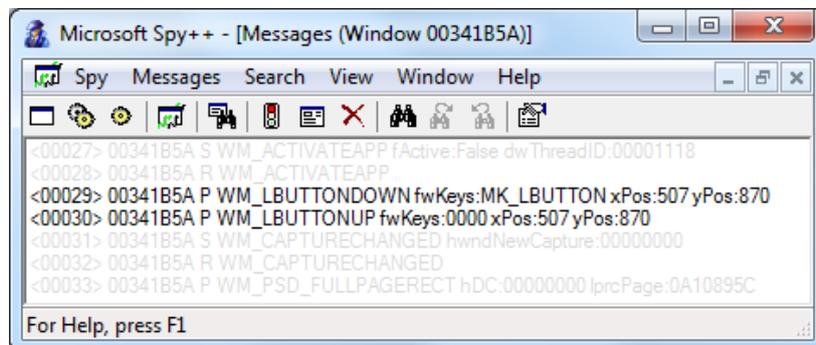


Abbildung 3.27: Auszug aus *Spy++*.

der Auflösungsabhängig wodurch wiederum Probleme auftreten könnten, da der Mousedown-Befehl möglicherweise „ins Leere“ geschickt wird. Abbildung 3.27 zeigt den via *Spy++*²³ aufgezeichneten Mousedown-Befehl, der an das *MathInput_Window*, mit dem Window-Handle 00341B5A gesendet und auf der Position $x = 507$, $y = 870$ ausgeführt wird.

Weiters existiert zwar ein *PaintEvent* auf das ein *EventListener* gesetzt werden kann, jedoch wird dieses Event nie abgesetzt. Aus diesem Grund musste auf die Verwendung dieser Funktion leider verzichtet werden.

Nicht-interpretierbare Ausdrücke

Ein weiteres Problem stellen nicht-interpretierbare Ausdrücke dar. Zum größten Teil wird jede Eingabe interpretiert, korrekt oder nicht spielt dabei keine Rolle. Ein Problem tritt auf, wenn ein Ausdruck nicht interpretiert werden kann. Dies trifft beispielsweise auf fälschlicherweise als zusammengehörig angenommene *Strokegroups* oder Zeichnungen wie in Abbildung 3.28 zu. In diesem Fall erscheint in der Interpretationsvorschau der *MathInputControl* die Meldung: „*Please write well formed math expressions.*“

Ist ein Ausdruck nicht-interpretierbar, löst das Betätigen der *Insert*-Schaltfläche kein Event aus, wodurch die Abarbeitung weiterer *Strokegroups* nicht fortgesetzt werden kann. Aus diesem Grund wird nach einer gewissen Zeit der Inhalt der *MathInputControl* Komponente geleert und der Ausdruck als nicht interpretierbar gehandhabt. Als mathematische Interpretation wird in diesem Fall der *Strokegroup* der Ausdruck `[unknown math]` zugewiesen.

Allgemeine Interpretationsprobleme

Während der Entwicklung des *NiCE Formula Editors* stellte sich heraus, dass die Interpretation von Texten und mathematischen Ausdrücken bereits

²³[http://msdn.microsoft.com/en-us/library/aa264396\(v=vs.60\).aspx](http://msdn.microsoft.com/en-us/library/aa264396(v=vs.60).aspx)

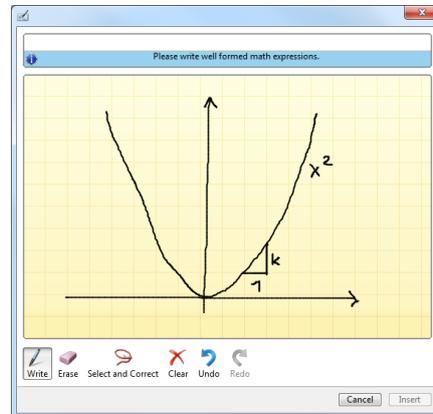


Abbildung 3.28: Nicht-interpretierbarer Ausdruck.

sehr gut funktioniert und meistens korrekte Ergebnisse liefert. Lediglich bei der Interpretation einzelner Zeichen, beispielsweise Variablen, stößt man regelmäßig auf Probleme.

Abbildung 3.29 zeigt ein konkretes Beispiel bei der Interpretation der mathematischen Ausdrücke $a = 3$ und $y = x^2 + a$. Durch die etwas unterschiedliche Schreibweise der Variablen a im ersten bzw. zweiten Ausdruck, liefert der Mathematik-Interpreter im ersten Fall m und im Zweiten a zurück. Zusätzlich zu der Verwendung einer einheitlichen Schreibweise, kann das Interpretationsergebnis durch die Verwendung der englischen Schreibweise verbessert werden. Beispiele hierfür wären die handgeschriebene Eins, die in den englischsprachigen Ländern lediglich aus einem Strich besteht und die Sieben, bei der der mittlere Querbalken weggelassen wird.

Wie bei allen Interpretationswerkzeugen kann dieses Problem durch häu-

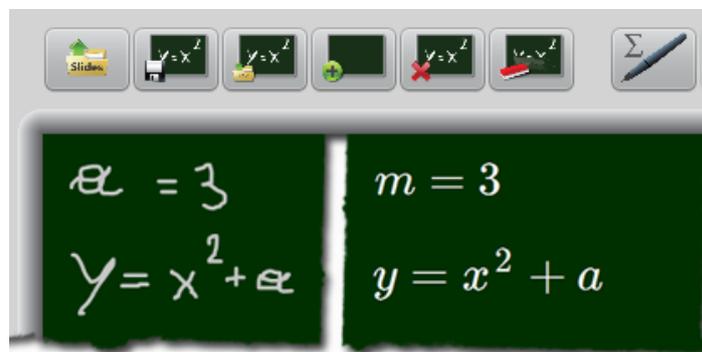


Abbildung 3.29: Handgeschriebene Eingabe (links) und Missinterpretation der Variablen a (rechts).

Abbildung 3.30 zeigt zwei Darstellungen der diskreten Fouriertransformation. Teil (a) zeigt eine handgeschriebene Formel auf einem grünen Hintergrund: $f_m = \sum_{k=0}^{2n-1} \left(x_k \cdot e^{-\frac{2\pi i}{2n} \cdot m \cdot k} \right)$. Teil (b) zeigt die mathematische Interpretation derselben Formel: $f_m = \sum_{k=0}^{2n-1} \left(x_k \cdot e^{-\frac{2\pi i}{2n} \cdot m \cdot k} \right)$.

Abbildung 3.30: Diskrete Fouriertransformation als handgeschriebener Ausdruck (a) und mathematische Interpretation (b).

figen Umgang mit dem System minimiert werden, indem man beispielsweise den Schreibstil an die Interpretationsweise anpasst. Die Interpretation von komplexeren Ausdrücken, wie beispielsweise die Definition der diskreten Fouriertransformation der Größe $2n$,

$$f_m = \sum_{k=0}^{2n-1} \left(x_k \cdot e^{-\frac{2\pi i}{2n} \cdot m \cdot k} \right), \quad (3.1)$$

stellt unter Beachtung der Interpretationsweise kein Problem dar. In Abbildung 3.30 sind der handgeschriebene Ausdruck und das Ergebnis der Interpretation dargestellt.

3.6.2 Notizzuordnung

Um ein möglichst plausibles Verhalten bei der Zuordnung der Notizen zu den jeweiligen Folien zu erhalten, wurden mögliche Szenarien erstellt und durchgespielt.

Eine Folie kann beliebig viele verschiedene Notizen besitzen. Beobachtungen von verschiedenen Mathematik-Vorlesungen und Vorträgen haben allerdings gezeigt, dass zu einer einzelnen Folie die Anzahl der Notizen den einstelligen Bereich sehr selten überschreitet.

Ein weiterer Fall wäre die Zugehörigkeit einer Notiz zu mehreren Folien. Prinzipiell ist jede erstellte Notiz immer genau einer Folie zugeordnet. Diese Zuordnung passiert sobald die erste Eingabe getätigt wird oder wenn einer Notiz Inhalt hinzugefügt wird, während eine andere Folie angezeigt wird. Die Verknüpfung von ein und derselben Notiz zu mehreren (direkt aufeinanderfolgender) Folien erwies sich dabei als überflüssig. Erst wenn Inhalt zur Notiz hinzugefügt wird und man sich gerade auf einer anderen als die ursprünglich zugeordnete Folie befindet, wird eine neue Zuordnung erstellt.

Abbildung 3.31 zeigt die Erstellung von drei Notizen. Im ersten Schritt wird der Ausdruck *Vector & Matrices* auf einer leeren Eingabefläche geschrieben. Die dadurch erstellte Notiz wird mit der ersten Folie verknüpft. Im nächsten Schritt wird über das Menü eine neue Notiz angelegt und mit dem Inhalt *Note 1b* versehen. Da immer noch die erste Folie angezeigt wird, wird diese Notiz ebenfalls mit der ersten Folie verknüpft. In Schritt drei und vier wird innerhalb der Präsentation jeweils eine Folien nach vorne navigiert.

	Anzeige	Zuordnung	Aktion
1			Schreiben
2			Neue Notizseite, Weiterschreiben
3			Nächste Folie
4			Nächste Folie
5			Notiz bearbeiten
6			

Abbildung 3.31: Verhalten beim Erstellen von Notizen.

Die angezeigte Notiz bleibt dabei immer die Gleiche, nur der linke Teil auf dem die Präsentation abgebildet ist, ändert sich. Im fünften Schritt wird nun die Notiz bearbeitet, indem der Ausdruck $1b$ gelöscht und dafür an dessen Stelle $3a$ niedergeschrieben wird. Da jetzt eine andere Folie angezeigt wird, wird eine neue Notiz erstellt, die mit der dritten Folie der Präsentation verknüpft wird.

Zusammenfassend ist zu sagen, dass der Vortragende sich bewusst sein muss wie der Vortrag ablaufen wird um eine nachvollziehbare Struktur der Folien-Notiz Zuordnung zu erhalten. Nicht lineare Abläufe sind vor allem für die spätere Ausgabe auf Papier schwer nachzustellen.

3.6.3 Navigation

Die als manchmal nebensächlich wirkende Möglichkeit der Navigation fällt in dem *NiCE Formula Editor* Szenario etwas komplexer aus. Wie in Abschnitt 1.6 beschrieben, besteht das Szenario aus zwei Whiteboards. Um nun nicht immer zwischen den beiden hin und her wechseln zu müssen, wurde die Interaktionsmöglichkeiten auf ein Whiteboard beschränkt.

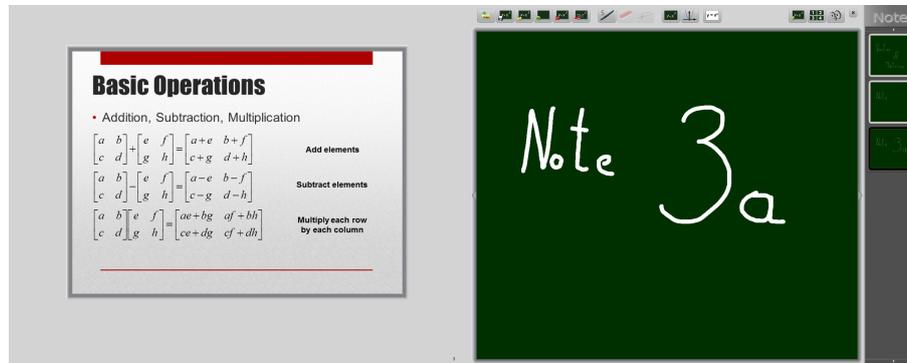


Abbildung 3.32: Navigations-Problematik.

Zweifach-Navigation

Einerseits muss innerhalb der Präsentation navigiert werden, andererseits zwischen den Notizen. Problematisch ist die Navigation zwischen den Notizen, da es nicht eindeutig nachvollziehbar ist, ob man durch Navigieren zwischen den Notizen auch innerhalb der Präsentation navigieren soll. Anders ausgedrückt, es ist nicht klar ob man durch Navigieren zu einer Notiz gleichzeitig zu der verknüpften Folie navigieren soll.

Abbildung 3.32 zeigt nochmals die Endsituation aus Abschnitt 3.6.2. Es existieren drei Notizen, die erste und zweite Notiz sind mit der ersten Folie, die dritte Notiz mit der dritten Folie verknüpft. Die Problematik liegt nun bei folgenden zwei Punkten: Welche Notiz soll angezeigt werden, wenn zurück zur ersten Folie navigiert wird bzw. wie soll sich die Präsentation verhalten wenn zwischen zwei Notizen gewechselt wird, die mit einer unterschiedlichen Folie verknüpft sind.

Durch Testen verschiedener Navigationsarten, wurde eine nachvollziehbare Methode ermittelt, welche nun zur Anwendung kommt. Der Benutzer hat dabei die Möglichkeit über die Notizliste zwischen den Notizen zu navigieren, ohne die aktuelle Folie zu wechseln. Wählt der Benutzer jedoch die bereits angezeigte Notiz noch einmal aus, wird zur dazugehörigen Folie navigiert. Navigieren innerhalb der Präsentation hat dabei keinen Einfluss auf die angezeigte Notiz. Zur Steuerung der Präsentation wird der linke und rechte Teil des um den Eingabebereich platzierten Rahmens verwendet. Zusätzlich kann man zu jeder Folie bzw. Notiz über die Overview gelangen.

3.6.4 Reihenfolge bei der Erstellung

Innerhalb einer **Strokegroup** spielt die Schreibrichtung keine Rolle für die Interpretation. Damit aber *GeoGebra* die Anweisungen korrekt ausführt, muss bei der Erstellung der Inhalte auf die Reihenfolge geachtet werden. Abbil-

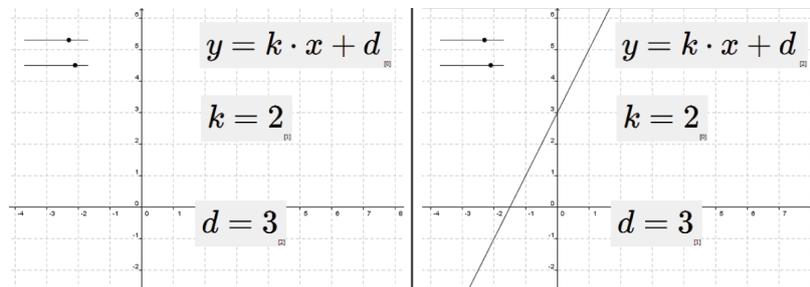


Abbildung 3.33: Falsche und richtige Reihenfolge.

Abbildung 3.33 zeigt ein Beispiel, bei dem die Reihenfolge eine Auswirkung auf das von *GeoGebra* gelieferte Ergebnis hat. Als erstes wurde die Geradengleichung $y = k \cdot x + d$ niedergeschrieben, anschließend wurden die beiden mathematischen Ausdrücke $k = 2$ und $d = 3$ erstellt.

Das System interpretiert die **Strokegroups** hintereinander und gibt sie anschließend an *GeoGebra* weiter. Da die Erstellung der Ausdrücke $k = 2$ und $d = 3$ im ersten Fall erst nachdem sie bereits in der Geradengleichung $y = k \cdot x + d$ verwendet wurden passierte, ist diese Anweisung zu diesem Zeitpunkt nicht definiert und kann nicht ausgeführt werden. Dies kann dadurch vermieden werden, indem man die Variablen immer im Vorhinein definiert. Da es aber manchmal vorkommen kann, dass im Nachhinein Änderungen getroffen werden, wurde eine Möglichkeit die Reihenfolge nachträglich zu ändern zur Verfügung gestellt. Dazu müssen die Elemente lediglich nacheinander in der korrekten Reihenfolge angeklickt werden.

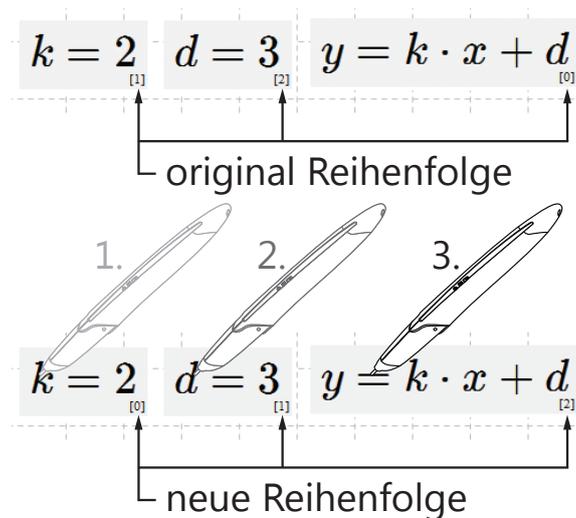


Abbildung 3.34: Reihenfolge der Eingaben.

Damit der Benutzer weiß, in welcher Abfolge die Eingaben an *GeoGebra* übergeben werden, besitzt jeder mathematische Ausdruck eine der Reihenfolge entsprechenden Nummer. Diese wird, wie in Abbildung 3.34 zu sehen ist, in der rechten unteren Ecke angezeigt.

Kapitel 4

Ergebnisse

In diesem Kapitel werden die Vorteile und Einschränkungen anhand einiger Beispiele sowie die entwickelten Funktionen des *NiCE Formula Editors* dargestellt.

4.1 Einsatzgebiete

Im folgenden Abschnitt werden einige der möglichen Einsatzgebiete aufgezählt, in denen sich die Verwendung des *NiCE Formula Editors* sehr eignet.

4.1.1 Vektorrechnung

Durch die Verwendung des *NiCE Formula Editors* kann eine klassische Präsentation über Vektorrechnung sehr gut aufgewertet werden. Die Präsentation, die auf einem der beiden Whiteboards angezeigt wird, kann wie gewohnt Schritt für Schritt durchgegangen werden. Gleichzeitig können am zweiten

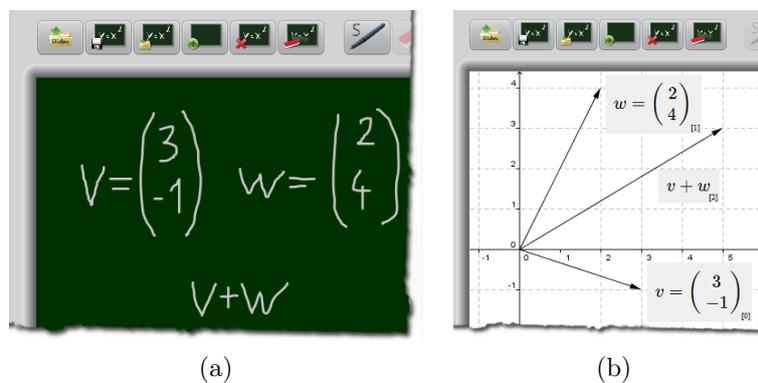


Abbildung 4.1: Handgeschriebene Eingabe (a) und Darstellung in *GeoGebra* (b).

Whiteboard Notizen zu den Folien erstellt werden. Da sich die Vorgangsweisen bei den verschiedenen arithmetischen Operationen zwischen Vektoren ähneln und diese sehr gut visuell dargestellt werden können, ist die im *NiCE Formula Editor* zur Verfügung stehende Darstellung in *GeoGebra* sehr dafür geeignet. Addition und Subtraktion zweier Vektoren im zweidimensionalen Raum sowie Multiplikation mit einem Skalar stellen ideale Anwendungsgebiete dar.

Um beispielsweise den Additionsvorgang der Vektoren $v = (3, -1)$ und $w = (2, 4)$ besser erklären zu können, kann die in Abbildung 4.1 (a) gezeigte Notiz erstellt und wie in (b) zu sehen, mittels *GeoGebra* dargestellt werden. Um ein weiteres Beispiel zu bringen, kann im Eingabe-Modus mit dem *Stroke Eraser Tool* die x-Komponente des v -Vektors gelöscht werden um diese anschließend durch eine Variablen a zu ersetzen. Der Variablen a wird nun der Wert 3 zugewiesen. Wird die neue Situation jetzt wieder mittels *GeoGebra* dargestellt, kann der Vektor $v = (a, -1)$ nicht angezeigt werden, da dem System die Variable erst bekannt ist, nachdem sie bereits im Vektor verwendet wurde. Aus diesem Grund muss die Reihenfolge der Ausdrücke geändert werden, wodurch nun wieder die Ausgangssituation angezeigt wird. Die x-Komponente des v -Vektors ist jetzt jedoch von der Variablen a abhängig. Werden numerische Variablen in *GeoGebra* dargestellt, erzeugt *GeoGebra* automatisch einen Schieberegler, mit dem der Wert geändert werden kann. Verändert man die Werte von a mittels des Reglers, wirken sich die Änderungen sofort auf den v -Vektor und somit auf das Ergebnis der Addition mit dem w -Vektor aus.

4.1.2 Drehmatrizen

Um ein konkretes Beispiel für die Anwendung von Matrix-Multiplikationen zu veranschaulichen, werden die in Abbildung 4.2 (a) gezeigten Eingaben erstellt. Dabei handelt es sich um einen Punkt A , der numerischen Variablen α , den beiden Variablen a und b , einer Matrix R sowie eine Multiplikation der Matrix mit dem Punkt. Die Matrix stellt in diesem Fall eine Rotationsmatrix im zweidimensionalen Raum dar. Dadurch dass α der Wert $\pi/2$ zugewiesen wurde, müsste die Multiplikation von R und A einen um 90° um den Ursprung gedrehten Punkt ergeben. Um diese Vermutung zu bestätigen, soll das Ergebnis der Eingaben in *GeoGebra* dargestellt werden. In Abbildung (b) sieht man den Punkt B , der dem um 90° gedrehten Punkt A entspricht. Da es sich bei α wieder um eine numerische Variable handelt, stellt *GeoGebra* einen Schieberegler zur Verfügung, dessen Veränderung sich direkt auf das Ergebnis, sprich auf den Rotationswinkel, auswirkt.

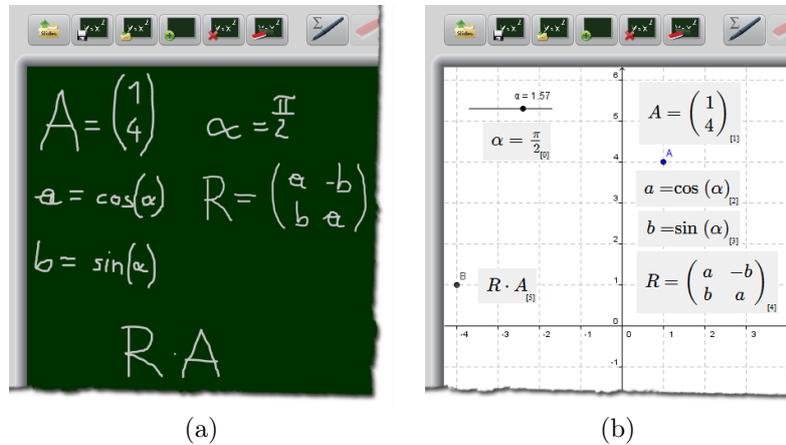


Abbildung 4.2: Handgeschriebene Eingabe (a) und Darstellung in *GeoGebra* (b).

4.1.3 Kurvendiskussion

Im Bereich der Kurvendiskussion stellt sich die Verwendung des *NiCE Formula Editors* als besonders nützlich heraus. Funktionen können, wie vom Umgang mit traditionellen Tafeln gewöhnt, durch Niederschreiben der Funktionsgleichung definiert und sofort mittels *GeoGebra* dargestellt werden. Flächen unter Kurven und Ableitungen sowie die Auswirkungen von zusätzlich eingefügten Faktoren stehen ohne großen Arbeitsaufwand zur Verfügung.

Abbildung 4.3 (a) zeigt eine Diskussion der Funktion $y = x^2$. Dazu soll neben der Funktion an sich, die Ableitung, die Fläche unter der Kurve innerhalb bestimmter Grenzen, sowie die Auswirkung einer Addition eines konstanten Wertes auf die ganze Funktion visuell dargestellt werden. Nach dem

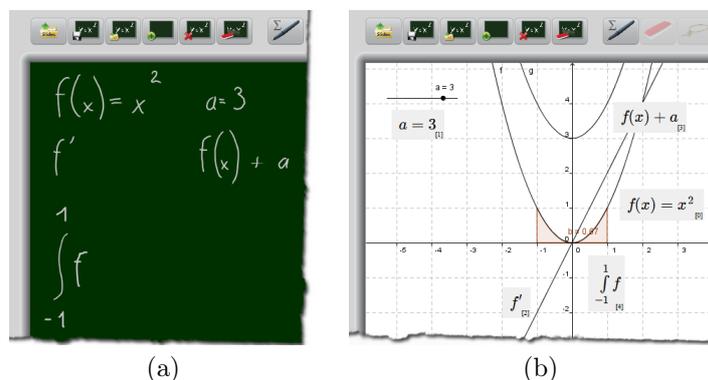


Abbildung 4.3: Handgeschriebene Eingabe (a) und Darstellung in *GeoGebra* (b).

Erstellen der Eingaben muss lediglich auf die Darstellung mittels *GeoGebra* gewechselt werden um alle gewünschten Funktionen angezeigt zu bekommen.

4.1.4 Vor- und Nachteile

Durch die Verwendung des *NiCE Formula Editors* können schnell und mühelos mehrere Konstruktionen von Funktionen erstellt werden, da die Neuzeichnung per Hand wegfällt. Durch die Verwendung von Variablen und des von *GeoGebra* bereitgestellten Schiebereglers, können Vorgänge dynamisch als eine Art Animation dargestellt werden. Die Bereitstellung sofortiger Lösungen für gewisse Problemstellungen, wie beispielsweise der Ermittlung des bestimmten Integrals, stellt einen weiteren Vorteil dar.

Den Verlust der Fähigkeit Konstruktionen per Hand zu erstellen, könnte als Nachteil bezeichnet werden. Werden alle Konstruktionsschritte vom System übernommen, könnte dadurch das grundsätzliche Wissen über die Vorgehensweise beim Konstruieren verloren gehen. Dies obliegt schlussendlich jedoch dem Vortragenden, da die Möglichkeit zur händischen Konstruktion immer noch zur Verfügung steht und genauso wie bei analogen Vorträgen genutzt werden kann.

4.2 Funktionalität

Die grundlegende Funktion des *NiCE Formula Editor* besteht aus der Erstellung von handgeschriebenen Notizen auf einem interaktiven Whiteboard. Zur Unterstützung können mathematische Gegebenheiten in der dynamischen Mathematiksoftware *GeoGebra* visuell dargestellt werden.

Zusätzlich soll der Benutzer die Möglichkeit besitzen eine Präsentation zu öffnen, Notizen zu den Folien zu erstellen und diese zusammen mit der Präsentation zu exportieren. Weiters soll die Möglichkeit gegeben sein, eine Präsentation mit bereits erstellten Notizen zu öffnen und so den Ablauf des Vortrags rekonstruieren zu können. Die für diese Aufgaben benötigte Funktionalität wird im Folgenden beschrieben.

4.2.1 Hauptfunktionen

Die Hauptfunktionen des *NiCE Formula Editor* bestehen aus

- Präsentation öffnen,
- Aktuelle Notiz speichern,
- Einzelne Notiz öffnen,
- Neue Notiz erstellen,
- Notiz löschen und
- Notiz leeren.

Präsentation öffnen

Diese Funktion erzeugt wie in Kapitel 3 Abschnitt 3.5.3 beschrieben, aus einer vorhandenen Präsentation eine Reihe von Bildern bzw. erstellt eine Verknüpfung zu einer bereits existierenden Bildserie. Diese können anschließend im *NiCE Formula Editor* angezeigt und mit Notizen versehen werden.

Aktuelle Notiz speichern

Diese Funktion speichert die Inhalte der aktuellen Notiz. Ob die Notiz einer Präsentation zugehörig ist oder nicht, spielt dabei keine Rolle. Wie in Abschnitt 3.3.2 beschrieben, werden die **Strokes** dabei als `.isf`, die zusätzlich benötigten Informationen in einer `.info` und die via *GeoGebra* erstellten Inhalte als `.xml` Datei hinterlegt.

Einzelne Notiz öffnen

Diese Funktion ermöglicht das Öffnen einer einzelnen Notiz. Führt man diese Funktion während eine Präsentation geöffnet ist aus, besitzt der Benutzer die Möglichkeit die aktuelle Notiz entweder zu überschreiben oder die zu öffnende Notiz nach der aktuellen Notiz einzufügen. Die **Strokegroups**, der Typ der jeweiligen **Stropkegroup**, gesetzte Schriftgrößen und gegebenenfalls getätigte *GeoGebra* Inhalte werden dabei geladen.

Neue Notiz erstellen

Diese Funktion löscht den Inhalt des Eingabebereichs und erstellt im Hintergrund eine neue Notiz. Ist gerade eine Präsentation geöffnet, wird die Notiz erst der Präsentation und somit der Notizliste hinzugefügt, wenn die erste Eingabe getätigt wird. So bleibt dem Benutzer noch die Möglichkeit innerhalb der Präsentation zu navigieren und so die Verknüpfung der Notiz zu einer bestimmten Folie erstellen.

Notiz löschen

Ist gerade keine Präsentation geöffnet, löscht diese Funktion lediglich den Inhalt des Eingabebereichs. Ist gerade eine Präsentation geöffnet, wird die Notiz zusätzlich von der Präsentation und somit auch aus der Notizliste entfernt. Anschließend wird die nachfolgende Notiz (bzw. vorherige wenn es sich um die Letzte handelt) geladen und zu der damit verknüpften Folie navigiert.

Notiz leeren

In Kombination mit dem im Abschnitt 4.2.4 beschriebenen *Lasso Selection Tool*, ermöglicht diese Funktion das selektive Löschen von **Strokes**. Sind

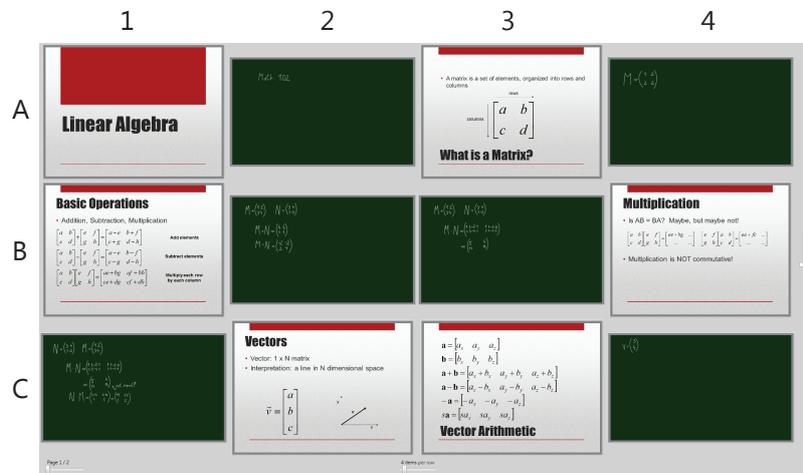


Abbildung 4.4: Übersicht über eine Präsentation.

gerade keine **Strokes** selektiert, löscht diese Funktion den gesamten Inhalt des Eingabebereichs, entfernt jedoch nicht die Notiz und die Verknüpfung zur Folie.

4.2.2 Weitere Funktionen

Bei geöffneter Präsentation stehen des Weiteren noch

- Präsentation exportieren,
- Overview anzeigen und
- Steuerung der Präsentation

zur Verfügung.

Präsentation exportieren

Diese Funktion exportiert die Folien der aktuell geöffneten Präsentation und die dazugehörigen Notizen als HTML-Dateien. Aktuell werden nur die Text- und Mathematik-Interpretationen der Notizen exportiert. Der Export der ursprünglichen handgeschriebenen Notizen wird derzeit noch nicht unterstützt, könnte aber durch das seit HTML5 eingeführte **canvas** Element realisiert werden. Näher wird darauf im Kapitel 5, Abschnitt 5.1 eingegangen.

Overview anzeigen

Wie in Abbildung 4.4 abgebildet, kann man sich eine Übersicht über die Elemente der Präsentation, sprich über die Folien und den dazugehörigen Notizen, anzeigen lassen.

In der Abbildung sind 12 Elemente dargestellt, wobei es sich bei Element $A1$, $A3$, $B1$, $B4$, $C2$ und $C3$ um Folien, bei den restlichen um die dazugehörigen Notizen handelt. Dabei erkennt man, dass die Folie $B1$ zwei Notizen ($B2$ und $B3$) besitzt, die Folie $C2$ hingegen keine einzige. Wählt man nun eine Folie aus, springt die Präsentation zu der ausgewählten Folie, der Notizbereich bleibt jedoch unverändert. Wählt man eine Notiz aus, bleibt die aktuell angezeigte Folie unverändert, jedoch wird die ausgewählte Notiz angezeigt. Die Anzahl der angezeigten Elemente pro Reihe bzw. Spalte kann durch zwei Schieberegler eingestellt werden.

Steuerung der Präsentation

Wie in Abschnitt 3.6.3 bereits erwähnt, kann man innerhalb der Präsentation auf drei verschiedene Möglichkeiten navigieren: Über die Overview, die Notizliste und durch betätigen der „Eingaberahmen-Schaltflächen“. Klickt man in der Overview auf eine Folie, wechselt die auf dem Whiteboard angezeigte Präsentation dorthin. Wählt man die gerade angezeigte Notiz in der Notizliste aus, springt man zu der Folie, die mit der Notiz verknüpft ist. Betätigt man die zu Schaltflächen umfunktionierten Teile des Rahmens und die Eingabefläche, navigiert man innerhalb der Präsentation zurück (linker Teil des Rahmens) bzw. vor (rechter Teil des Rahmens).

4.2.3 Optionen

Standardmäßig wird während der Darstellung in *GeoGebra-Ansicht* keine Menüleiste angezeigt. Bei Bedarf kann man durch aktivieren der Optionen *GeoGebra Toolbar anzeigen* bzw. *GeoGebra Menubar anzeigen* die gewünschte Menüleiste einblenden. Anschließend stehen die gleichen Funktionen wie in der Standalone Variante von *GeoGebra* zur Verfügung.

Wie in Abschnitt 3.3 beschrieben wird, spielt die Zeitspanne zwischen den einzelnen Eingaben eine Rolle bei der Zuteilung zu einer **Strokegroup**. Der Standardwert der Zeit die zwischen zwei Eingaben liegen darf um noch zur gleichen **Strokegroup** zugeteilt zu werden, liegt bei einer Sekunde. Dieser Wert kann hier in den Optionen eingestellt und so an die Schreibgeschwindigkeit des Benutzers angepasst werden.



Abbildung 4.5: Standard Werkzeug- und Menüleiste von *GeoGebra*.

4.2.4 Werkzeuge

In der Menüleiste stehen die Werkzeuge

- *Pen Tool*,
- *Stroke Eraser Tool* und
- *Lasso Selection Tool*

zum Erstellen und Bearbeiten von Notizen zur Verfügung.

Pen Tool

In Kombination mit dem digitalen Stift entspricht das *Pen Tool* der Kreide bzw. dem Whiteboardmarker bei der Verwendung von Tafeln bzw. Whiteboards. Dieses Werkzeug ermöglicht dem Benutzer handgeschriebene Notizen zu erstellen, indem er (wie vom Umgang mit Tafel und Kreide gewohnt) mit dem Stift auf dem Whiteboard agiert. Da das System nicht selbstständig erkennen kann ob es sich bei der getätigten Eingabe um einen mathematischen Ausdruck handelt, besitzt das *Pen Tool* zwei verschiedene Modi, den *Mathematik-Modus* und den *Text-Modus*. Tätigt man die Eingabe im *Mathematik-Modus*, werden die **Strokes** anschließend als mathematischer Ausdruck gehandhabt und an den Mathematik Interpreter geschickt. Zur Unterscheidung in welchem Modus man sich gerade befindet, wird entweder ein \sum für den *Mathematik-Modus* bzw. ein S für den *Text-Modus* auf der *Pen Tool* Schaltfläche angezeigt.

Stroke Eraser Tool

Dieses Werkzeug dient zum Entfernen von erstellten **Strokes**. Im Gegenzug zu einem *Point Eraser*, der wirklich nur die Punkte entfernt, mit denen er in Berührung kommt, entfernt der *Stroke Eraser* gleich den gesamten **Stroke**, sobald man einen Punkt davon berührt. Diese Vorgangsweise empfiehlt sich auf Grund der Formel- bzw. Schrifterkennung, da das System Eingaben die in einem Fluss erstellt werden, korrekter interpretiert.

Dieses Werkzeug könnte auch durch den Einsatz einer Durchstreich-Geste oder durch Wischen auf einem drucksensitiven Whiteboard ersetzt werden. Genauer wird darauf im Kapitel 5, Abschnitt 5.3 eingegangen.

Lasso Selection Tool

Neben dem separieren und kombinieren von **Strokegroups** dient das *Lasso Selection Tool* zur Bearbeitung von getätigten Eingaben. Wie aus diversen Grafikprogrammen bekannt, besitzt der Benutzer die Möglichkeit mit dem *Lasso Selection Tool* **Strokes** zu markieren und anschließend zu verschieben oder mittels *Notiz leeren* selektiv zu löschen.

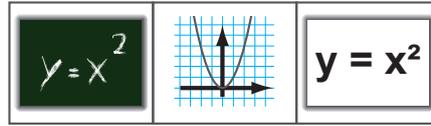


Abbildung 4.6: Symbole der *Sketching*, *GeoGebra* und Interpretations-Ansicht in der Menüleiste.

4.2.5 Ansichten

Die erstellten Notizen können auf drei verschiedene Arten dargestellt werden: Als ursprüngliche handgeschriebene Eingaben (*Sketching*-Ansicht), als Text- und Mathematik-Interpretationen oder als visuelle Repräsentation in *GeoGebra* (*GeoGebra*-Ansicht).

Sketching-Ansicht

In dieser Ansicht wird die Eingabe des Benutzers erfasst und dient somit zur Erstellung der Notizen. *Abbildung 4.7 (a)* zeigt zwei handgeschriebene Ausdrücke. Dabei handelt es sich einmal um das Wort *Parabola* und einmal um den mathematischen Ausdruck $f(x) = x^2$.

GeoGebra-Ansicht

In dieser Ansicht werden einerseits die Interpretationen der vom Benutzer erstellten Eingaben dargestellt. Handelt es sich dabei um mathematische Ausdrücke werden diese zusätzlich an *GeoGebra* übergeben und visualisiert. *Abbildung 4.7 (b)* zeigt die Interpretation der handgeschriebenen Eingaben und die Visualisierung des mathematischen Ausdrucks in *GeoGebra*.

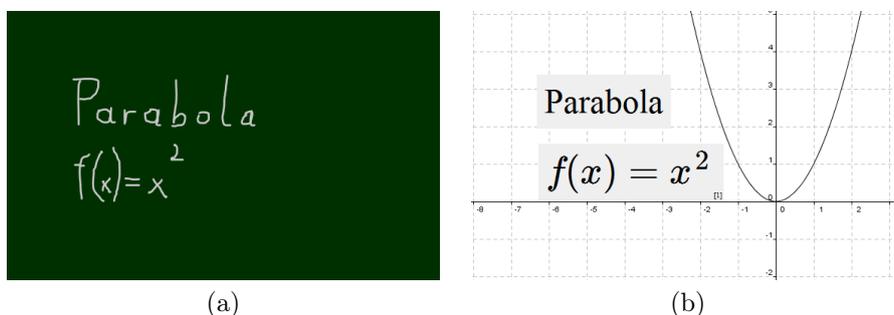


Abbildung 4.7: Handgeschriebene Eingabe (a) und Darstellung in *GeoGebra* (b).

Text- und Mathematik-Interpretation

In dieser Ansicht werden, wie in der *GeoGebra-Ansicht*, die Interpretationen der Eingaben dargestellt. Diese Ansicht ist vor allem für die spätere Aufbereitung der Unterrichtsmaterialien gedacht. Näher wird darauf in Kapitel 5 eingegangen.

Kapitel 5

Ausblick

Im folgenden Kapitel werden einige Verbesserungsmöglichkeiten hinsichtlich neuer Features und technischer Veränderungen beschrieben. Verbesserungen, wie beispielsweise der Umstieg auf HTML5, beziehen sich lediglich auf technische Aspekte, durch die der Umgang mit dem *NiCE Formula Editor* verbessert werden kann. Durch die Interaktion mit dem Publikum würde sich zwar das ursprüngliche Szenario des *NiCE Formula Editor* etwas verändern, jedoch würden sich aus den erweiterten Einsatzmöglichkeiten neue Anwendungsgebiete ergeben.

5.1 Umstieg auf HTML5

Um die handgeschriebenen Eingaben ebenfalls in den exportierten HTML Dateien anzeigen zu können, wurde zuerst der Ansatz verfolgt, Screenshots der Inhalte zu erstellen und anzuzeigen. Die kann aber durch die Verwendung des seit HTML5 eingeführten `canvas` Elements umgangen werden. In Abbildung 5.1 ist die Darstellung der diskreten Fouriertransformation in einer HTML Datei, innerhalb des `canvas` Elements zu sehen.

Programmausschnitt 5.1 zeigt die Repräsentation des ersten `Strokes` der Diskrete Fouriertransformation als Linienzug innerhalb des `canvas` Elements. Durch die Möglichkeit, die handgeschriebenen Eingaben auf diese Weise innerhalb einer HTML Datei anzuzeigen, können diese leichter exportiert und dadurch besser wiederverwendet werden. Weiters könnte die

$$f_m = \sum_{k=0}^{2n-1} \left(x_k \cdot e^{-\frac{2\pi i}{2n} \cdot m \cdot k} \right)$$

Abbildung 5.1: Diskrete Fouriertransformation innerhalb des `canvas` Elements.

```
1 var ctx = canvas.getContext("2d");
2 ctx.beginPath();
3   ctx.moveTo(287.023605511811,161.001095898745);
4   ctx.lineTo(287.023605511811,161.001095898745);
5   ctx.lineTo(288.006282677165,161.001095898745);
6   ctx.lineTo(288.006282677165,161.001095898745);
7   ctx.lineTo(290.009432283464,161.001095898745);
8   ctx.lineTo(290.992109448819,161.001095898745);
9   ctx.lineTo(292.012581889764,161.001095898745);
10  ctx.lineTo(294.015731496063,161.001095898745);
11  ctx.lineTo(294.998408661417,161.001095898745);
12  ctx.lineTo(296.018881102362,161.983773064099);
13  ctx.lineTo(298.022030708661,161.983773064099);
14  ctx.lineTo(299.004707874016,161.983773064099);
15  ctx.stroke();
```

Programm 5.1: Auszug aus einer `.info` Datei.

HTML-Repräsentation innerhalb der Darstellung in *GeoGebra* verwendet werden um beispielsweise die ursprünglichen Eingaben als Beschriftungen zu verwenden oder handgezeichnete Skizzen mit Inhalten aus *GeoGebra* zu kombinieren.

5.2 Interaktion mit Publikum

Durch die Verwendung von speziellem Papier (z.B. mit *Anoto*-Pattern versehen) und digitalen Stiften könnten Ergebnisse der Teilnehmer direkt auf das Whiteboard geholt werden. Dadurch könnten die verschiedenen Lösungen bzw. aufgetretene Probleme besprochen oder korrigiert werden. Diese Ergebnisse könnten anschließend direkt in die Unterrichtsmaterialien miteinbezogen werden. Der direkte Einfluss auf den Ablauf und die Inhalte durch die Teilnehmer könnte sich positiv auf die Aufmerksamkeit auswirken. Die selbst eingebrachten Inhalte erzeugen eine Art von Verbundenheit mit der Thematik durch die eine höhere Motivation und Akzeptanz erzeugt werden könnte.

5.3 Weitere Verbesserungen

Durch den regelmäßigen Umgang mit dem *NiCE Formula Editor* zeigten sich einige Punkte die Verbesserungswürdig wären. Diese Verbesserungsmöglichkeiten werden hier kurz beschrieben.

5.3.1 Wechsel des Werkzeugs

Da die Interpretation häufig noch falsche Ergebnisse liefert, muss das Öfteren unter Verwendung des *Stroke Eraser Tools* eine Eingabe gelöscht und neu erstellt werden. Dadurch, dass das Werkzeug nur über die oben im *NiCE Formula Editor* platzierte Menüleiste verfügbar ist und man nachdem der Löschvorgang abgeschlossen ist, wieder auf das *Pen Tool* wechseln muss, bewegt man sich häufig zwischen der Menüleiste und dem Eingabebereich hin und her. Dies unterbricht den Arbeitsfluss wesentlich mehr als der bei Tafeln oder Whiteboards durch Wischen durchgeführte Löschvorgang. Des Weiteren stellt sich der häufige Wechsel als körperlich anstrengend heraus, was nicht unterschätzt werden sollte. Diese Problematik könnte durch den Einsatz von Gesten, durch die Verwendung eines drucksensitiven Whiteboards oder durch eine Umgestaltung des Menüs behoben werden.

Gesten

Wie beispielsweise von Zeleznik et al. in [14] beschrieben, könnte man das *Stroke Eraser Tool* durch die Einführung einer Durchstreich-Geste ersetzen. Dadurch müsste nicht mehr zwischen den beiden Werkzeugen gewechselt werden, da man mit dem *Pen Tool* die Geste durchführen könnte.

Zusätzlich könnte man auch Gesten für die Navigation innerhalb der Präsentation oder zwischen den einzelnen Notizen einführen. Für die restlichen Funktionen würde sich die Erstellung intuitiver Gesten jedoch als sehr kompliziert erweisen.

Drucksensitives Whiteboard

Ein anderer Ansatz um den derzeit mühevollen Wechsel zwischen dem *Stroke Eraser* und *Pen* Werkzeug zu vermeiden, wäre die Verwendung eines drucksensitiven Whiteboards oder des von Rosenberg et al. entwickelten *UnMousePads* [10]. Dadurch könnte der Löschvorgang wie von Tafeln und Whiteboards gewohnt, durch Wischen über die zu löschenden **Strokes** passieren. Ob dafür ein Objekt (zum Beispiel ein Schwamm) oder die bloße Hand verwendet wird, spielt prinzipiell keine Rolle.

Neugestaltung des Menüs

Der letzte Lösungsansatz um das ständige Wechseln zwischen Menüleiste und Eingabebereich zu vermeiden bzw. um es müheloser zu gestalten, könnte durch das Auslösen des Menüs aus der Menüleiste realisiert werden. Das Menü sollte dann immer neben der aktuellen Eingabe angezeigt werden, wodurch sich die Wege, die beim Wechseln zurückgelegt werden müssen, verkürzt und dadurch erträglicher gestaltet.

5.3.2 Export-Möglichkeiten

Derzeit existiert bereits die Möglichkeit aus den Unterlagen und den dazugehörigen Notizen, eine HTML-Repräsentation zu generieren. Die Struktur der exportierten Seiten ist dabei mit dem Aufbau der mit *Javadoc*¹ erstellten Dokumentationen vergleichbar. Zusätzlich könnte die Stimme des Vortragenden aufgenommen werden, wodurch der gesamte Vortrag Schritt für Schritt rekonstruiert werden könnte. Dazu müsste lediglich passend zum gesprochenen Text die Erstellung der Notizen sequenziell wiedergegeben werden.

5.3.3 Mehrsprachigkeit

Da der derzeitige Interpreter Englisch als Standardsprache verwendet, wurde die Oberfläche des *NiCE Formula Editors* ebenfalls in Englisch gehalten. Die Möglichkeit die Sprache zu ändern, wenn dafür ein Interpreter zur Verfügung steht, würde die entstehenden Materialien nochmal aufwerten, da man sie dann in der jeweiligen Muttersprache erstellen könnte.

¹<http://www.oracle.com/technetwork/java/javase/documentation/javadoc-137458.html>

Anhang A

Anhang

```
1 <body>
2   <div name="0" id="mathDiv0" class="mathDiv">
3     <div class="innerMathML" id="mathML0">
4       <math>
5         <mi>y</mi>
6         <mo>=</mo>
7         <msup>
8           <mrow>
9             <mi>x</mi>
10            </mrow>
11           <mrow>
12             <mn>2</mn>
13           </mrow>
14         </msup>
15       </math>
16     </div>
17     <div class="innertext" id="text0">4=+2</div>
18     <div class="idTag" id="idTag0">[0]</div>
19   </div>
20 </body>
```

Programm A.1: Ausschnitt aus der HTML-Datei der Text- bzw. Mathematik-Interpretation.

```
1 <body>
2   <div id="GeoGebraApplet">
3     <applet id="ggbApplet" name="ggbApplet" code="geogebra.
4       GeoGebraApplet" archive="../geogebra.jar" MAYSCRIPT>
5       <param name="filename" value="../empty.ggb" />
6       <param name="framePossible" value="false" />
7       <param name="language" value="de" />
8       <param name="showAlgebraInput" value="false" />
9       <param name="showMenuBar" value="false" />
10      <param name="showToolBar" value="false" />
11      <param name="maxIconSize" value="16" />
12      Please <a href="http://www.java.com">install Java 1.4.2</a> (or
13      later) to use this page.
14    </applet>
15    <iframe frameborder="no" id="mathDiv0iframe"></iframe>
16    <div name="0" id="mathDiv0" class="mathDiv">
17      <div class="innerMathML" id="mathML0">
18        <math>
19          <mi>y</mi>
20          <mo>=</mo>
21          <msup>
22            <mrow>
23              <mi>x</mi>
24            </mrow>
25            <mrow>
26              <mn>2</mn>
27            </mrow>
28          </msup>
29        </math>
30      </div>
31      <div class="innertext" id="text0">4=+2</div>
32      <div class="idTag" id="idTag0">[0]</div>
33    </div>
34  </div>
35 </body>
```

Programm A.2: Ausschnitt aus der HTML-Datei der Darstellung in *GeoGebra*.

```

1 <?xml version="1.0"?>
2 <presentations>
3   <presentation>
4     <path>C:\Test1.pptx</path>
5     <order>
6       <orderElement>0</orderElement>
7       <orderElement>1</orderElement>
8       <orderElement>2</orderElement>
9     </order>
10    <slideImgs>
11      <slideFileName>C:\Folie1.PNG</slideFileName>
12      <slideFileName>C:\Folie2.PNG</slideFileName>
13    </slideImgs>
14    <slides>
15      <slide>
16        <pageNumber>1</pageNumber>
17        <note>
18          <noteId>0</noteId>
19          <xPos>318,968391300282</xPos>
20          <yPos>232,025125920773</yPos>
21          <baseNotes />
22          <strokeGroups>
23            <strokeGroup id="0" isFormula="False" zoomLevel="20">
24              <stroke>318;232;32;39;</stroke>
25              <stroke>364;242;23;30;</stroke>
26            </strokeGroup>
27          </strokeGroups>
28        </note>
29        <note>
30          <noteId>1</noteId>
31          <xPos>246,0625</xPos>
32          <yPos>134,076375246048</yPos>
33          <baseNotes />
34          <strokeGroups>
35            <strokeGroup id="0" isFormula="True" zoomLevel="20">
36              ...
37            </strokeGroup>
38          </strokeGroups>
39        </note>
40      </slide>
41      <slide>
42        <pageNumber>2</pageNumber>
43        <note>
44          <noteId>2</noteId>
45          ...
46        </note>
47      </slide>
48    </slides>
49  </presentation>
50  <presentation>
51    <path>C:\Test2.ppt</path>
52    ...
53  </presentation>
54 </presentations>

```

Programm A.3: Auszug aus *data.xml*.

Anhang B

Inhalt der CD-ROM/DVD

B.1 Diplomarbeit

Pfad: /

_DaBa.pdf Diplomarbeit (Gesamtdokument)

B.2 Abbildungen

Pfad: /figures/

1_Motivation/ Bilder und Grafiken des Kapitels *Motivation*

2_VerwandteArbeiten/ Bilder und Grafiken des Kapitels *Verwandte Arbeiten*

3_Implementierung/ . . Bilder und Grafiken des Kapitels *Implementierung*

4_Ergebnisse/ Bilder und Grafiken des Kapitels *Ergebnisse*

5_Ausblick/ Bilder und Grafiken des Kapitels *Ausblick*

B.3 Literatur

Pfad: /references/

iwb.htm *Interactive Whiteboards in learning and teaching in two Sheffield schools*, von Levy

Breuer - 2007.pdf . . . *A lightweight open space for the classroom - collaborative learning with whiteboards and pen-tablets*, von Breuer et al.

Davis - 2007.pdf . . . *Magic Paper: Sketch-Understanding Research*, von Davis

Glover - 2001.pdf . . . *Missioners, tentatives and luddites:*

- leadership challenges for school and classroom posed by the introduction of interactive whiteboards into schools in the united kingdom*, von Glover et al.
- He - 2003.pdf *Why take notes? Use the whiteboard capture system*, von He et al.
- Hohenwarter - 2002.pdf *GeoGebra - ein Softwaresystem für dynamische Geometrie und Algebra der Ebene*, von Hohenwarter
- Hohenwarter - 2006.pdf *GeoGebra - didaktische Materialien und Anwendungen für den Mathematikunterricht*, von Hohenwarter
- Labahn - 2008.pdf . . . *Math-brush: A system for doing math on pen-based devices*, von Labahn et al.
- LaViola - 2004.pdf . . . *Mathpad2: a system for the creation and exploration of mathematical sketches*, von LaViola et al.
- Miller - 2004.pdf *Enhancing mathematics teaching through new technology: The use of the interactive whiteboard*, von Miller et al.
- Rosenberg - 2009.pdf . *The unmousepad: an interpolating multi-touch force-sensing input pad*, von Rosenberg et al.
- Roth - 2009.pdf *Bezel swipe: conflict-free scrolling and multiple selection on mobile touch screen devices*, von Roth et al.
- Wenzel - 2003.pdf . . . *Mathjournal™ - an interactive tool for the tablet pc, 2003*, von Wenzel et al.
- Zeleznik - 2010.pdf . . *Hands-on math: a page-based multi-touch and pen desktop for technical work and problem solving*, von Zeleznik et al.
- Zeleznik - 2008.pdf . . *Mathpaper: Mathematical sketching with fluid support for interactive computation*, von Zeleznik et al.

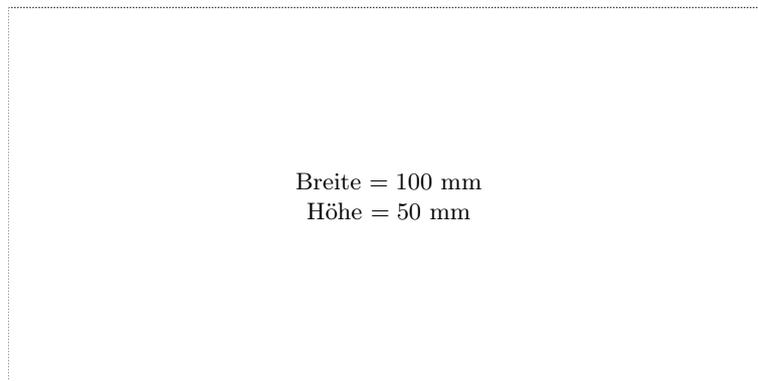
Literaturverzeichnis

- [1] Breuer, H., C. Sousa, N. Baloian und M. Matsumoto: *A lightweight open space for the classroom - collaborative learning with whiteboards and pen-tablets*. In: *Seventh IEEE International Conference on Advanced Learning Technologies (ICALT 2007)*, S. 66–70, Niigata, Japan, Juli 2007. IEEE Computer Society Press.
- [2] Davis, R.: *Magic paper: Sketch-understanding research*. *Computer*, 40:34–41, 2007.
- [3] Glover, D. und D. Miller: *Missioners, tentatives and luddites: leadership challenges for school and classroom posed by the introduction of interactive whiteboards into schools in the united kingdom*. In: *Paper for the BEMAS Conference*, Newport Pagnell, United Kingdom, Okt. 2001. <http://www.keele.ac.uk/media/keeleuniversity/fachumsocsci/sclpppp/education/interactivewhiteboard/Missioners.pdf>.
- [4] He, L. W., Z. Liu und Z. Zhang: *Why take notes? use the whiteboard capture system*. In: *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '03)*, Bd. 5, S. 776–779, Hong Kong, China, Apr. 2003. IEEE Computer Society Press.
- [5] Hohenwarter, M.: *GeoGebra - ein Softwaresystem für dynamische Geometrie und Algebra der Ebene*. Diplomarbeit, Naturwissenschaftliche Fakultät der Paris-Lodron-Universität Salzburg, Salzburg, Austria, Feb. 2002.
- [6] Hohenwarter, M.: *GeoGebra - didaktische Materialien und Anwendungen für den Mathematikunterricht*. Dissertation, Naturwissenschaftliche Fakultät der Paris-Lodron-Universität Salzburg, Salzburg, Austria, Jan. 2006.
- [7] Labahn, G., E. Lank, S. MacLean, M. Marzouk und D. Tausky: *Math-brush: A system for doing math on pen-based devices*. In: *The Eighth IAPR International Workshop on Document Analysis Systems (DAS '08)*, S. 599–606, Nara, Japan, Sep. 2008. IEEE Computer Society Press.

- [8] LaViola, J. und R. Zeleznik: *Mathpad²: a system for the creation and exploration of mathematical sketches*. In: *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, S. 432–440, New York, NY, USA, Aug. 2004. ACM.
- [9] Miller, D. und D. Glover: *Enhancing mathematics teaching through new technology: The use of the interactive whiteboard*. A Report to the Nuffield Foundation, Juli 2004. [http://www.nuffieldfoundation.org/sites/default/files/NuffieldFinalReport\(Miller-Keele\)\(1\).pdf](http://www.nuffieldfoundation.org/sites/default/files/NuffieldFinalReport(Miller-Keele)(1).pdf).
- [10] Rosenberg, I. und K. Perlin: *The unmousepad: an interpolating multi-touch force-sensing input pad*. In: *ACM SIGGRAPH 2009 papers*, SIGGRAPH '09, S. 1–9, New York, NY, USA, Aug. 2009. ACM.
- [11] Roth, V. und T. Turner: *Bezel swipe: conflict-free scrolling and multiple selection on mobile touch screen devices*. In: *Proceedings of the 27th International Conference on Human Factors in Computing Systems*, CHI '09, S. 1523–1526, New York, NY, USA, Apr. 2009. ACM.
- [12] Wenzel, L. und H. Dillner: *MathjournalTM - an interactive tool for the tablet pc*, 2003. http://www.xthink.com/downloads/MathJournal_2003.pdf.
- [13] Zeleznik, R., A. Bragdon, F. Adeptura und H. S. Ko: *Hands-on math: a page-based multi-touch and pen desktop for technical work and problem solving*. In: *Proceedings of the 23rd annual ACM Symposium on User Interface Software and Technology*, UIST '10, S. 17–26, New York, NY, USA, Okt. 2010. ACM.
- [14] Zeleznik, R., T. Miller, C. Li und J. J. Laviola, Jr.: *Mathpaper: Mathematical sketching with fluid support for interactive computation*. In: *Proceedings of the 9th International Symposium on Smart Graphics*, SG '08, S. 20–32, Berlin, Heidelberg, Aug. 2008. Springer-Verlag.

Messbox zur Druckkontrolle

— Druckgröße kontrollieren! —



— Diese Seite nach dem Druck entfernen! —