AUTOMATISCHE HANDSCHRIFTENERKENNUNG IN E-KREIDE DOKUMENTEN

Florian Theimer
Institut für Informatik, Freie Universität Berlin
Takustr. 9, 14195 Berlin
florian_theimer@hotmail.com

Zusammenfassung

Dieser Bericht ist die Zusammenfassung der gleichnamigen Bachelorarbeit [Theimer04]. Beschrieben wird ein System, das Handschrift aus Vorlesungen, die mit dem elektronischen Tafelsystem E-Kreide aufgezeichnet wurden, zum Zwecke der Indizierung für Internet Suchmaschinen erkennt. Die Umsetzung erfolgt unter Nutzung des Microsoft Tablet PC SDKs und der Google Web API.

Die Eingabedaten der Vorlesung werden eingelesen, analysiert und derart aufbereitet, dass nach Möglichkeit nur einzelne Worte und vor allem keine Zeichnungen an den Erkenner weitergegeben werden. Nach dem Erkennungsprozess werden unterschiedliche Methoden zur Selektion falsch erkannter Wörter angewandt. Im Zuge der Umsetzung eines Generators für Webseiten, die den automatisch generierten Index einer E-Kreide Vorlesung beinhalten, wurden zudem Methoden zur Integration der .NET Module mit Java untersucht.

Inhaltsverzeichnis

1. Einführung 3

2. Vergleichbare Ansätze 4

2.1 Automatische Indizierung von Multimedia-Dokumenten 4

2.2 Ausgewählte Ansätze zur Handschriftenerkennung 4

3. Verwendete Software 5

3.1 Anforderungen an die Handschriftenerkennung 5

3.2 Das Microsoft Tablet PC SDK 6

3.3 Die Google Web API 7

3.4 Anbindung von Java 7

4. Die Anwendung 8

4.1 Die Erkennerkomponente 8

4.2 Der Erkenner als Kommandozeilenwerkzeug 10

4.3 Die Erkenner-Benutzerschnittstelle 10

4.4 Die Java Proxy Klasse 11

4.5 Der HTML Generator 11

5. Ergebnisse.....12

5.1 Bewertung der Selektionsverfahren.....13

6. Zusammenfassung und Ausblick..16

Literaturverzeichnis 17

Internetreferenzen 18

1. Einführung

Im Projekt E-Kreide der Freien Universität Berlin wird ein Multimediasystem für den Präsenz- und Fernunterricht entwickelt [EKREIDEWWW]. Es besteht aus einer elektronischen Tafel und in Java geschriebener Software, die das Gesamtsystem steuert [ECHALK02], [ECHALK03].

Die E-Kreide Anwendung kann wie eine gewöhnliche Tafel benutzt werden. Zusätzlich können multimediale und interaktive Elemente in das Tafelbild integriert und Audiodaten aus der Vorlesung aufgenommen und verwaltet werden. Die Daten werden so gespeichert, dass sie jederzeit aus dem Internet mit Hilfe eines normalen Webbrowsers abspielbar sind (siehe Abbildung 1).

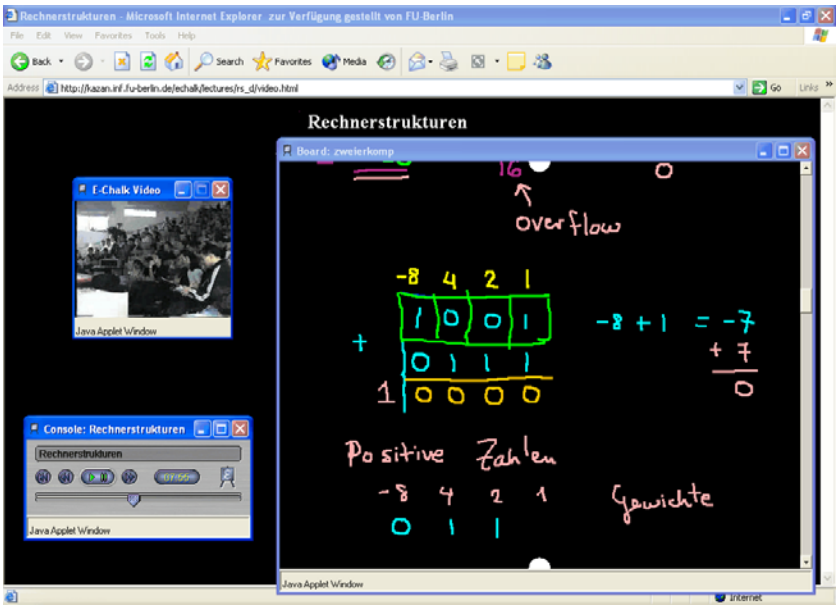

Abbildung 1: Eine E-Kreide Vorlesung im Webbrowser.

Die für die Handschriftenerkennung relevanten Eingabedaten einer E-Kreide Vorlesung werden in einem eigenen Format gespeichert [ECHALK00].

Im Rahmen dieser Arbeit wird die automatische Erkennung der Handschrift aus einer E-Kreide Vorlesung umgesetzt. Die Ergebnisse des Erkennungsprozesses werden dabei derart aufbereitet, dass daraus ein Stichwortverzeichnis generiert werden kann. Zur Handschriftenerkennung wird eine bestehende Softwarebibliothek verwendet. Daneben wird die Möglichkeit evaluiert, Microsoft .NET-Anwendungen in Java zu integrieren.

2. Vergleichbare Ansätze

Spätestens seit Einführung des Apple Newton 1993 ist Handschriftenerkennung ein bedeutender Problemkomplex in der Informatik. Der Handheld Computer setzte als erstes Gerät für Endverbraucher ein System zur Handschriftenerkennung ein. Dabei ist der eingesetzte, wörterbasierte Erkenner von Paragraph International Inc. vor allem durch seine hohe Fehlerrate bekannt geworden.

Erst in den letzten Jahren erscheinen zunehmend Geräte auf dem Markt, die es dem Benutzer ermöglichen, Text und Grafiken wie auf einem Blatt Papier zu erstellen. Seit der Einführung von Microsoft Windows CE und Microsoft XP Tablet PC Edition sind komplexe Systeme zur Erkennung von Handschriften allgemein verfügbar.

2.1 Automatische Indizierung von Multimedia-Dokumenten

Authoring on the Fly

Die Multimedia-Forschungsgruppe des Lehrstuhls für Algorithmen und Datenstrukturen des Instituts der Informatik an der Albert-Ludwigs-Universität Freiburg arbeitet seit einigen Jahren an einem System zur Indizierung von *Authoring on the Fly (AOF)* Daten an Hand von Audioströmen [AOFSEWWW].

Authoring on the Fly ist ein Forschungsprojekt zur Verknüpfung der drei Aktivitäten Präsentationsaufzeichnung, Teleteaching und Produktion multimedialer Lehrdokumente in einem System. Dabei sind Softwarewerkzeuge entstanden, die in der Praxis eingesetzt werden.

Das Tool *aofSEaudio* dient dabei zur Indizierung von AOF-Vorlesungen anhand von Audiodaten mit Hilfe von Algorithmen zur Spracherkennung [HÜRST03]. Öffentlich zugänglich ist die Indizierung von 31 Vorlesungen zum Thema *Geometrische Algorithmen* aus dem Sommersemester 1999 und von 27 Vorlesungen zum Thema *Algorithmentheorie* aus dem Wintersemester 1999/2000 [AOFVLWWW].

Die Ergebnisse der Erkennung sind nach Angaben der Autoren bei dem eingesetzten Prototypen noch verbesserungswürdig. In einigen Vorlesungen werden weniger als 20% der Wörter richtig erkannt [HÜRST02]. Diese Erkennungsraten reichen jedoch für eine sinnvolle Indizierung der Vorlesungen aus.

2.2 Ausgewählte Ansätze zur Handschriftenerkennung

NICI (Nijmegen Institute for Cognition and Information)

Die *Handwriting Recognition Group* aus dem *Institute for Cognition and Information* der University of Nijmegen setzt sich seit 1990 mit der Thematik der Handschriftenerkennung auseinander [NIKIWWW]. Im Zuge der Forschungsarbeiten wurde unter anderem eine agentenbasierte Software zur Erkennung von Handschrift

entwickelt: *The NICI stroke-based recognizer of online handwriting* [SCHOMAKER90], [NIKIRECWWW].

IBM Thomas J. Watson Research Center

Die *Pen Technologies-Arbeitsgruppe* von IBM arbeitete mehrere Jahre an einer Technologie zur mehrsprachigen online Handschriftenerkennung [IBMPENWWW]. Die eingesetzte Technologie basiert auf Hidden Markov Modellen [KWOK01]. Die Ergebnisse flossen in die *Pen-Computing Platform*[1] von IBM ein.

Communication Intelligence Corporation

Die Firma Communication Intelligence Corporation, welche in Zusammenarbeit mit dem Stanford Research Institute gegründet wurde, bietet Komponenten zur Handschriftenerkennung an, die unter anderem von Palm Inc. verwendet werden [CICWWW]. Die eingesetzten Technologien zur Handschriftenerkennung basieren auf Forschungsergebnissen zur Mustererkennung.

Microsoft

Die in dieser Arbeit verwendete Komponente zur Handschriftenerkennung kommt von der Firma Microsoft. Sie wird mit den Betriebssystemen *Microsoft Windows CE* und *Microsoft Windows XP Tablet PC Edition* angeboten. Weitere Informationen zur eingesetzten Komponente des Tablet PC SDKs finden sich im folgenden Abschnitt.

3. Verwendete Software

3.1 Anforderungen an die Handschriftenerkennung

Um die Handschriftenerkennung aus E-Kreide Daten zu ermöglichen, muss die eingesetzte Komponente die folgenden Bedingungen erfüllen:

- *Die Komponente soll ohne Training einsetzbar sein:*
  Es ist unpraktikabel, vor Beginn einer Vorlesung den Dozenten die Komponente trainieren zu lassen. Auch könnten bereits vorliegende Vorlesungen nur dann bearbeitet werden, wenn der Dozent zu einem nachträglichen Training bereit wäre.
- *Die Komponente soll zusammenhängende Wörter erkennen:*
  Vor allem in PDAs und Pocket PCs werden Erkenner eingesetzt, die einzelne Buchstaben identifizieren. Die Methode ist für diese Anwendung unpraktikabel. Die Einschränkung für den Dozenten wäre zu groß und bisherige Vorlesungen könnten

[1] Im Rahmen der *Pen Computing Platform* wurde zum Bespiel 1998 ein digitales Notizbuch (*The Crosspad™*) entwickelt [IBMCPWWW].

nicht bearbeitet werden. Voraussetzung ist deswegen ein *cursive recognizer* (siehe 1).

- *Die Komponente soll in das E-Kreide System integriert werden:*
  Die Komponente muss eine Integration in das bestehende E-Kreide System ermöglichen. Damit scheiden unter anderem Komponenten aus, die nur auf Pocket PC oder PALM Systemen laufen.

3.2 Das Microsoft Tablet PC SDK

Die Entscheidung für den Einsatz des *Microsoft Tablet PC SDKs* als Komponente zur Handschriftenerkennung basiert vor allem auf der sehr gute Erkennungsleistung und der Verfügbarkeit in der Arbeitsgruppe. Das Tablet PC SDK unterstützt momentan Visual C++, Visual Basic und alle Sprachen des .NET-Frameworks. Im April 2004 waren dies 26 [DOTLANGWWW, DOTNETWWW].

Die Umsetzung der Erkennerkomponente dieser Arbeit erfolgte im Rahmen der Microsoft .NET Umgebung mit der Programmiersprache *Microsoft Visual C#* und basiert auf dem *Tablet PC SDK, Version 1.5*, welches von Microsoft kostenfrei bezogen werden kann [TPSDKWWW].

Das SDK kann mit Windows 2000, Windows XP und Windows XP Tablet PC Edition eingesetzt werden. Es stehen unter Windows 2000 und XP nur das Framework und keine Erkennermodule zur Verfügung. Aus diesem Grund ist ein sinnvoller Einsatz zurzeit nur mit der Tablet PC Edition von Windows XP möglich.

Die Struktur des Microsoft Tablet PC SDKs

Das Tablet PC SDK kann grob in vier Bereiche unterteilt werden [JARRET02]:

Sammlung von Eingabedaten (Ink Collection)

Dieser Bereich unterstützt den Entwickler beim Sammeln von handschriftlichen Eingabedaten. Da diese Daten schon durch das E-Kreide System erhoben wurden, wird auf diesen Bereich nicht weiter eingegangen.

Verwaltung der Eingabedaten (Ink Management)

Dieses Modul des SDKs umfasst Objekte zur Datenverwaltung. Im Folgenden sind die Zentralen davon vorgestellt:

- *Das Packet Objekt*: Stellt die kleinste Eingabeeinheit dar. Dies umfasst Daten, die das Eingabemedium zu einem bestimmten Zeitpunkt zur Verfügung stellt, zum Beispiel ein Zeitstempel, eine Position, der Druck und die Größe des Stifts.
- *Das Stroke Objekt*: Repräsentiert eine Sammlung von Packet Daten die in einer `Stift-Aufsetzen-Bewegen-Absetzen`-Bewegung entstehen.
- *Das Ink Objekt*: Ein grundlegender Datentyp der die Eingabedaten verwaltet, manipuliert und speichert. Es enthält einen oder mehrere Stroke Objekte sowie Methoden und Eigenschaften, um diese zu verwalten und zu manipulieren.

Erkennen von Eingabedaten (Ink Recognition)

Dieser Bereich des SDKs dient dazu, `Ink` Daten in Text umzuwandeln. Das `Recognizer` Objekt erzeugt unter Berücksichtigung bestimmter Beschränkungen, wie zum Beispiel Sprache, einem Benutzerwörterbuch oder Informationen über die Art der Eingabe (zum Beispiel Postleitzahl), ein `RecognitionResult` Objekt. Dieses enthält immer ein Erkennungsergebnis und unter Umständen mehrere Alternativen. Für die Sprache US-Englisch steht auch ein Indikatorwert, für die Erkennungsgüte, in drei Abstufungen zur Verfügung.

Analyse von Eingabedaten (Ink Analysis)

Zur Analyse der Eingabedaten dient das `Divider` Objekt. Es unterteilt die Daten eines `Ink` Objekts, in Texteingabe und Zeichnungen. Bei Textdaten unterscheidet das `Divider` Objekt zusätzlich zwischen Segmenten (in der Regel ein Wort), Zeilen und Abschnitten.

3.3 Die Google Web API

Google Inc. bietet einen kostenlosen Webservice zur Suche in ihrer Datenbank an [GOOGLEWWW]. Die Kommunikation mit dem Service läuft mit der *Web Service Definition Language (WSDL)* über das S*imple Object Access Protocol (SOAP)* [SOAPWWW, WSDLWWW]. Um den Webservice nutzen zu können, ist eine Anmeldung mit einem kostenfreien, persönlichen Zugangscode erforderlich. Dieser berechtigt in der derzeitigen Betaversion bis zu 1000 Anfragen pro Tag. Diese Einschränkung ist in der Regel nicht problematisch, unter der Voraussetzung, dass jeder Nutzer einen eigenen Zugangscode besitzt und nicht mehr als zwei Vorlesungen am Tag indiziert.

Dieser Dienst ermöglicht die Suche in den indizierten Webseiten[2] von Google und liefert strukturierte Informationen über die Suchergebnisse. Von besonderem Interesse für diese Anwendung sind dabei die Anzahl der Hits sowie die Anzahl der Webverzeichnisse, die zu einem Suchbegriff gefunden werden. Google ermöglicht es, die Schreibweise eines Wortes zu überprüfen und liefert gegebenenfalls einen Verbesserungsvorschlag.

3.4 Anbindung von Java

Zur Erleichterung der Anwendung der Erkennerkomponente in einer Java-Umgebung, wurde eine Java Proxy-Klasse entwickelt, welche die eigentliche C# Anwendung anspricht. Um die Funktionalität der Klasse zu verdeutlichen, entstand eine Testanwendung zur Erzeugung von HTML-Dokumenten aus E-Kreide Vorlesungen.

Caffeine ist ein Open-Source Projekt zur Entwicklung einer Bibliothek für die Herstellung von Interoperabilität zwischen Java und dem .NET Framework [CAFFWWW]. Diese vielversprechende Software befindet sich noch in einem Alpha-

[2] Am 19. April 2004 waren es über 4 Milliarden indizierte Webseiten.

Stadium, was gegen den Einsatz in diesem Projekt sprach. Zu einem späteren Zeitpunkt wäre es sinnvoll, die jetzige XML basierte Implementierung zum Austausch von Daten zwischen Java und dem .NET Erkenner, durch diese Bibliothek zu ersetzen.

4. Die Anwendung

4.1 Die Erkennerkomponente

Die Erkennerkomponente ist das Kernstück der Anwendung. Sie implementiert das Erkennen von Wörtern aus E-Kreide Vorlesungen. Die Einbindung der Komponente in eigene Anwendungen ist sowohl als *Dynamic Linked Library* als auch als *COM Server* möglich.

Der Ablauf eines Erkennungsprozesses

Es wird zwischen vier Phasen der Erkennung unterschieden: die Datenaufbereitung, die Textanalyse, die Texterkennung und die Behandlung der Ergebnisse.

Die Datenaufbereitung

Diese Phase beinhaltet das Auslesen der Daten einer E-Kreide Vorlesung und die Umwandlung in das interne Datenformat. Zur Verbesserung der Erkennerleistung werden die Daten in zeitlich zusammenhängende Blöcke von Linienpunkten (siehe 4.1), deren zeitliche Abfolge einen bestimmten Schwellwert nicht überschreitet, gruppiert.

Diese Heuristik, die davon ausgeht, dass ein Wort zeitlich zusammenhängend geschrieben wird, unterstützt das Microsoft SDK bei der Aufgabe Wörter abzugrenzen.

Die Textanalyse

Sind die `eWordBlock` Daten generiert, werden sie mit dem `Divider` Objekt aus dem Microsoft SDK klassifiziert, um Wörter und Zeichnungen zu unterscheiden. Ausschließlich jene Daten, die als *Wort* gekennzeichnet wurden, werden an das `Recognizer` Objekt weitergegeben.

Die Texterkennung

Das `Recognizer` Objekt des Microsoft SDKs liefert zu jedem Datensatz ein Ergebnis, unabhängig von seiner Wahrscheinlichkeit. So wird auch aus einer Zeichnung ein in diesem Fall sinnloses Wort ermittelt. Ein Ergebnis beinhaltet meist mehrere Alternativen, wobei das beste Ergebnis gekennzeichnet wird [MICROSOFT03].

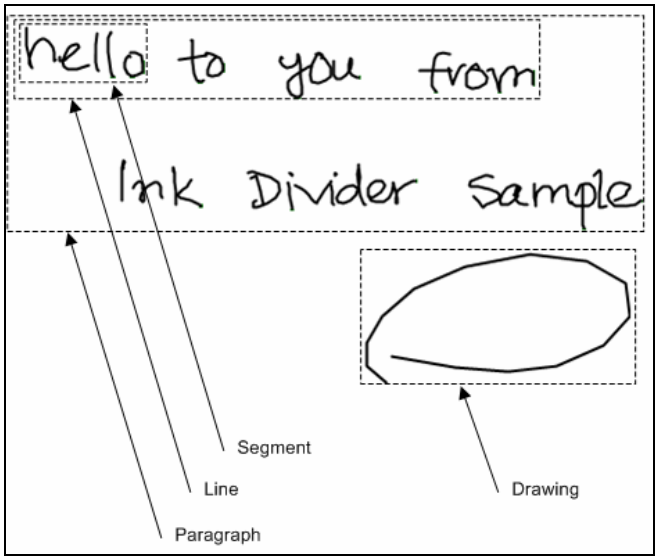

Abbildung 2: Die Klassifikationskategorien des Divider Objekts [MICROSOFT03].

Die Selektion der Ergebnisse

Aus den genannten Gründen, werden viele Wörter falsch erkannt. Vor allem in Aufzeichnungen, in denen zahlreich Zeichnungen oder Formeln vorkommen, werden viele fehlerhafte Resultate geliefert. Um die Fehlerrate zu reduzieren, bieten sich zwei Lösungsansätze an. Entweder die Unterscheidung von Text und Nicht-Text Bestandteilen vor dem Erkennungsprozess wird verbessert, oder falsch erkannte Wörter werden nach dem Erkennungsprozess aussortiert. Da die Verbesserung des Microsoft `Divider` Objekts den Umfang dieser Arbeit sprengen würde, wurden ausschließlich Methoden zur Selektion der Wörter nach dem Erkennungsprozess umgesetzt.

Insgesamt wurden letztendlich drei Methoden eingesetzt, welche wechselseitig miteinander kombiniert werden können. Weitere Methoden zur Selektion von falschen Wörtern, sind durch die Bewertung der Grammatik oder der Semantik von Sätzen denkbar. Da hierbei ganze Sätze betrachtet werden müssen und in den Vorlesungen oft nur einzelne Schlagwörter vorkommen, wurde auf eine nähere Betrachtung dieser Methoden verzichtet.

Selektion mittels Heuristik

Eine einfache Methode Wörter zu selektieren, kann über die Implementierung einfacher Heuristiken, die sich auf die gängige Struktur eines Wortes beziehen, realisiert werden. So lässt sich zum Beispiel über jedes deutsche Wort sagen, dass es aus mindestens zwei Buchstaben besteht, das erste Zeichen ein Buchstabe ist und es nur wenige Zeichen enthält, welche keine Buchstaben sind (wie etwa der Bindestrich).

Selektion mittels einer Wortliste

Eine weitere Maßnahme Wörter zu selektieren, ist der Abgleich mit einer Wortliste. Nur Worte die auf der Wortliste zu finden sind werden als Ergebnis akzeptiert. Das Gesamtergebnis ist damit von der Güte der benutzten Wortliste abhängig. Dies ist vor allem dann kritisch, wenn der Text viele domänenspezifische Wörter oder Eigennamen enthält, die in allgemeinen Wortlisten nicht vorkommen.

Selektion mittels Google

Zur Verbesserung der Selektion von Eigennamen und domänenspezifischen Wörtern, wurde eine Internet Suchmaschine benutzt, um die Gebräuchlichkeit eines Wortes zu bestimmen. So werden die meisten Wortlisten den Begriff „Babbage“ ablehnen, ebenso wie die Word Rechtschreibkorrektur. Die Suchmaschine Google liefert genügend Treffer[3], so dass es als gebräuchliches Wort angenommen werden kann[4].

4.2 Der Erkenner als Kommandozeilenwerkzeug

Die Konsolenanwendung ist ein einfaches Werkzeug, das die Daten für den Erkennungsprozess aus einer XML Datei liest und die Ergebnisse in eine XML Datei schreibt. Die Anwendung ist in C# entwickelt. Zum Lesen und Schreiben der XML Dateien wird die .NET XML Bibliothek verwendet.

4.3 Die Erkenner-Benutzerschnittstelle

Um einen einfachen Zugang zu der Funktionalität der Erkennerkomponente zur Verfügung zu stellen, wurde eine graphische Benutzerschnittstelle implementiert. Diese ermöglicht es, E-Kreide Vorlesungen zu laden und die Parameter für den Erkennungsprozess zu setzen. Nach dem Erkennungsprozess werden die Ergebnisse sowohl als Wortliste als auch graphisch dargestellt.

---

[3] Am 12. April 2004 ergab die Suche bei Google eine geschätzte Anzahl von 208.000 Treffern für den Suchbegriff *Babbage*.

[4] Natürlich können auch richtige Wörter falsch erkannt worden sein, wenn zum Beispiel *Haus* als *Maus* erkannt wird. Gegen diese Art der Fehler kann nachträglich nicht mehr vorgegangen werden. Dagegen würde eine Verbesserung des Worterkenners Abhilfe schaffen.

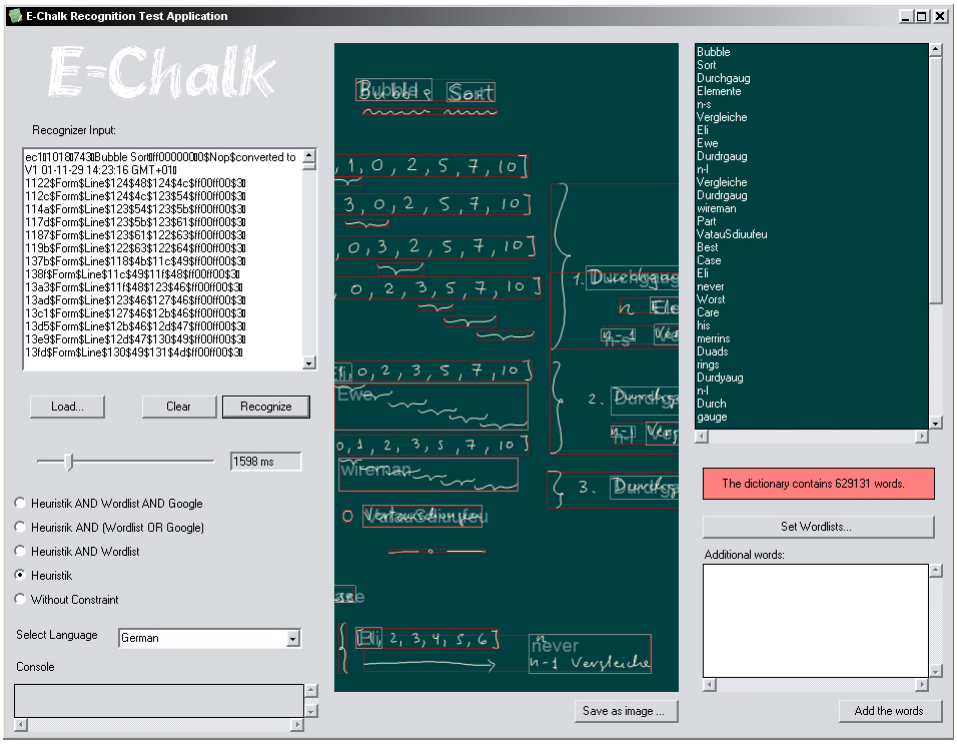

Abbildung 3: Die graphische Benutzerschnittstelle des Erkenners.

Als besonders hilfreich hat sich die Komponente zur graphischen Ergebnisrepräsentation erwiesen. Diese zeigt den Inhalt der E-Kreide Vorlesung an und kennzeichnet darin die `eWordBlock` Bereiche sowie die darin erkannten Wörter mit den Ergebnissen des Erkenners.

4.4 Die Java Proxy Klasse

Zur einfachen Nutzung der Erkennerkomponente in Java, dient eine Klasse, gemäß dem Proxy Pattern [GOF94]. Sie bietet Java Entwicklern einen einfachen Zugang zu der Erkennerkomponente und liefert eine für den Benutzer transparente Kommunikation mit der .NET Anwendung.

4.5 Der HTML Generator

Diese Java Anwendung lädt eine E-Kreide Vorlesung, gibt sie über die Java Proxy Klasse an den Erkenner weiter und erstellt aus den Ergebnissen eine HTML Seite. Die Ergebnisse werden sowohl im HTML Körper (mit einigen zusätzlichen Informationen) dargestellt, als auch in META Tags geschrieben, welche von gängigen Suchmaschinen zur Indizierung verwendet werden [MICHAEL03].

Der Zweck des HTML Generators ist es, die Anwendung der Erkennerkomponente von Java aus zu demonstrieren.

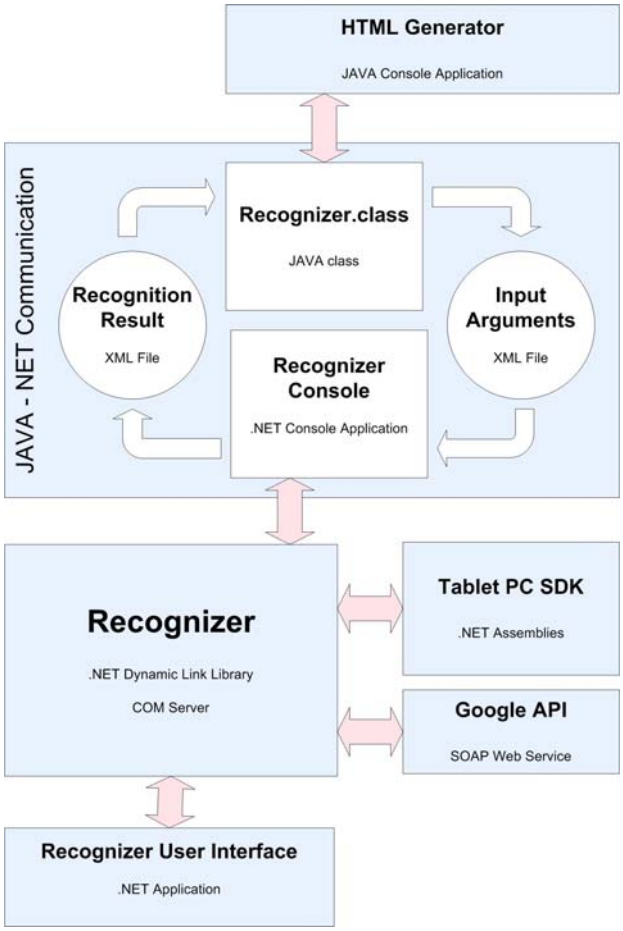

Abbildung 4: Die Erkennerkomponenten und ihr Zusammenwirken.

5. Ergebnisse

Zum Test der Erkennerkomponente wurden sechs Vorlesungen, mit folgenden Themen aus der Informatik, verwendet:

- Bubblesort (VL Algorithmen und Programmierung I, deutsch)
- Quicksort (VL Algorithmen und Programmierung I, deutsch)
- Farben (VL Bildverarbeitung, deutsch)
- Soccer playing robots (Kurzvortrag, englisch)
- Introduction to Wavelets (VL Bildverarbeitung, englisch)
- Zweierkomplementdarstellung (VL Rechnerstrukturen, deutsch)

Alle genannten E-Kreide Vorlesungen sind über die Webseite des Projekts erhältlich [EKREIDEWWW].

5.1 Bewertung der Selektionsverfahren

Zur Bewertung der Selektionsverfahren sind drei Vorlesungen auf ihr Verhältnis von richtig und falsch indizierten Wörtern, abhängig von den Selektionsmethoden, untersucht worden.

Die Vorlesungen wurden dazu mit den jeweiligen Selektionsmethoden indiziert. In dem Erkennungsergebnis wurden die richtig erkannten Wörter, die fälschlicherweise als Wort interpretierten Nicht-Wörter und die falsch aussortierten Wörter gezählt. Zu der Gruppe der Nicht-Wörter wurden auch falsch erkannte Wörter gerechnet, die nicht aussortiert wurden.

- *Wörter*: Wörter aus der Vorlesung die richtig erkannt und nicht aussortiert wurden.
- *Selektierte Wörter*: Wörter aus der Vorlesung, die richtig erkannt und aussortiert wurden.
- *Nicht-Wörter*: Zeichnungen, Formeln, etc. die als Wort interpretiert wurden; Wörter aus der Vorlesung die falsch erkannt und nicht aussortiert wurden.

VL Robotic Soccer	Ohne	Heuristik	Heuristik, Wörterliste	Heuristik, Wörterliste, Google
Wörter	6	5	4	4
Selektierte Wörter	0	1	2	2
Nicht-Wörter	44	4	4	4

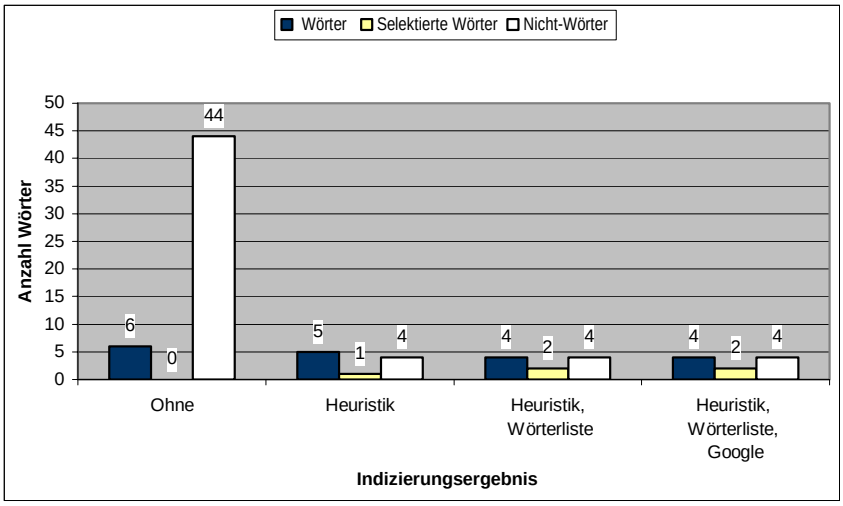

Abbildung 5: Ergebnis der Robotic Soccer Vorlesung.

In der kurzen *Robotic Soccer* Vorlesung (siehe Abbildung 5) werden ohne Selektion 44 falsche Wörter mit in den Index übernommen. Sobald die heuristische Selektion zuschaltet wird, werden nur noch fünf Nicht-Wörter indiziert. Weitere Selektionsverfahren verbessern hier die Indizierung nicht.

Das Ergebnis der längeren *Bubblesort* Vorlesung (siehe Abbildung 6) fällt ähnlich aus. Der Einsatz der heuristischen Selektion verringert die Anzahl falscher Wörter von 96 auf 34. Durch Einsatz von Wortlisten und Google verbessert sich dieses Ergebnis geringfügig.

VL Bubble Sort	Ohne	Heuristik	Heuristik, Wörterliste	Heuristik, Wörterliste, Google
Wörter	7	11	4	7
Selektierte Wörter	0	0	7	7
Nicht-Wörter	96	34	30	27

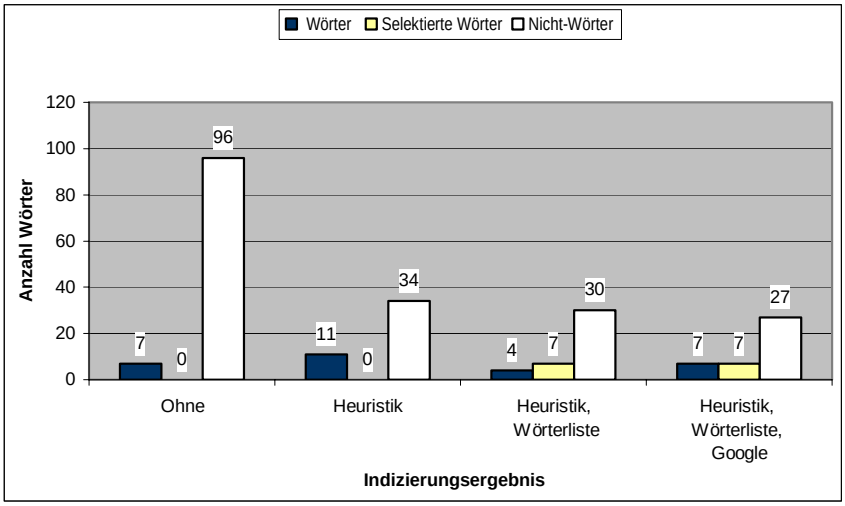

Abbildung 6: Die Bubblesort Vorlesung.

Deutlich wird auch, dass besonders der Einsatz einer Wortliste, die Gefahr mit sich bringt, richtig erkannte Wörter zu selektieren. In der Vorlesung werden sieben solcher Wörter durch die eingesetzte Wortliste aussortiert.

Beim Einsatz von Heuristik, Google und der Wortliste werden trotz des restriktiveren Selektionsverfahrens mehr Wörter akzeptiert als beim Einsatz von nur Heuristik und Wortliste. Der Grund ist, dass sich in der Wortliste unübliche Wörter befinden. Durch Google werden diese aussortiert und von einem Alternativwort ersetzt. In der Vorlesung führt dies zu einer Verbesserung der Indizierungsleistung.

Die Vorlesung zum Thema Zweierkomplement enthält lediglich 15 Wörter und zahlreiche Zeichnungen und Formeln. Entsprechend schlechter fällt auch das Indizierungsergebnis aus (siehe Abbildung 7).

VL Zweierkomplement	Ohne	Heuristik	Heuristik, Wörterliste	Heuristik, Wörterliste, Google
Wörter	4	6	8	9
Selektierte Wörter	0	0	2	3
Nicht-Wörter	121	41	25	23

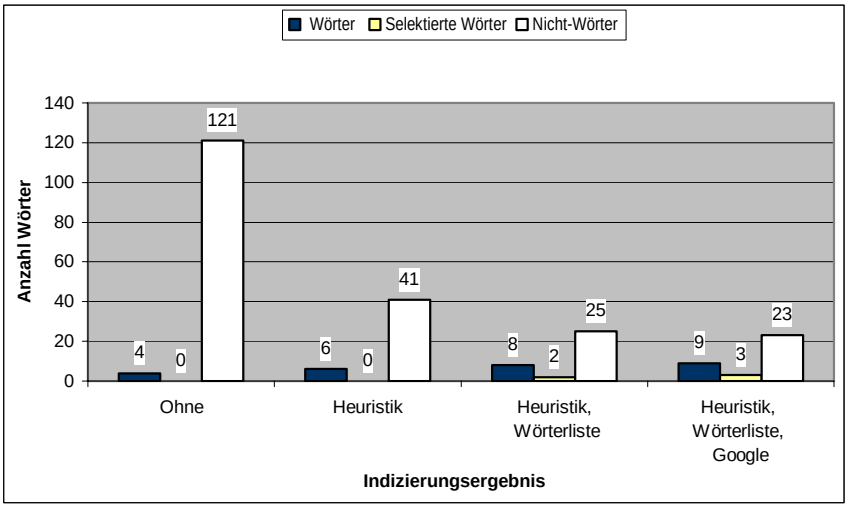

Abbildung 7: Die Vorlesung Zweierkomplementdarstellung.

Zusammenfassend lässt sich sagen, dass der Einsatz einer Heuristik ein effektives Mittel zur Selektion von Nicht-Wörtern ist. Der zusätzliche Einsatz von Wortlisten führt in der Regel zu einer leichten Verbesserung des Selektionsergebnisses. Es besteht aber die Gefahr, richtig erkannte Wörter zu eliminieren. Dies betrifft unglücklicherweise vor allem domänenspezifische Wörter, die oft nicht in Wortlisten enthalten sind. Dieser Gefahr kann man durch Einsatz von spezialisierten Wortlisten begegnen.

Die zusätzliche Selektion durch Google verbessert das Ergebnis nur geringfügig. Wie bei der Selektion durch Wortlisten besteht die Gefahr, domänenspezifische Wörter zu entfernen. Allerdings kann ein zusätzlich durch Google akzeptiertes Wort das Indizierungsergebnis qualitativ stark verbessern, wie zum Beispiel im Falle von Eigennamen. Um diesen Effekt zu erreichen, darf die Selektion durch Google nicht zusätzlich restriktiv eingesetzt werden. Wörter, die von Google akzeptiert der Wortliste aber abgelehnt werden, sollten in die Indizierung übernommen werden.

6. Zusammenfassung und Ausblick

Diese Arbeit beschreibt die Umsetzung eines Systems zur Erkennung von Handschrift aus E-Kreide Vorlesungen zum Zwecke der Indizierung für Suchmaschinen. Die Umsetzung erfolgte unter Nutzung des Microsoft Tablet PC SDKs und der Google Web API. Dabei werden die Eingabedaten der Vorlesung, nachdem sie eingelesen, analysiert und aufbereitet wurden, mit dem `Recognizer` Objekt des Microsoft Tablet PC SDKs einem Erkennungsprozess unterworfen. Danach werden mit unterschiedlichen heuristischen Methoden, als fehlerhaft erkannte Wörter aussortiert. Unter anderem wird dazu die Google Web API eingesetzt. Probleme, die bei der Integration von .NET und Java auftreten, wurden aufgezeigt und teilweise gelöst. Zur Demonstration der Einsatzmöglichkeiten der entwickelten Erkennerkomponente wurde eine Java Anwendung zur Generierung einer HTML Webseite mit Schlagwörtern aus E-Kreide Vorlesungen entwickelt.

Der augenscheinlichste Weg zur Verbesserung der Erkennungsleistung, ist die Entwicklung einer verbesserten Komponente zur Unterscheidung von Text, Formeln und Grafiken. Würde der Erkenner nur die Teile der Vorlesung erhalten, die als Text klassifiziert wurden, könnte das Resultat drastisch verbessert werden. Dazu könnte unter anderem der Formelerkenner eingesetzt werden, der momentan in der Arbeitsgruppe entwickelt wird [ROJAS03].

Eine weitere Idee zur Verbesserung der Extraktion von Schlagwörtern ist, dem Autor der Vorlesung eine Möglichkeit an die Hand zu geben, Schlagwörter direkt zu markieren, zum Beispiel indem er sie in einer bestimmten Farbe schreibt oder einen Kreis um sie zeichnet. Diese so markierten Wörter hätten eine höhere Erkennungswahrscheinlichkeit und würden nicht fälschlicherweise aussortiert werden.

Literaturverzeichnis

[ECHALK02] G. Friedland, L. Knipping, R. Rojas; E-Chalk: Technical Description; Technical Report B-02-11, FU Berlin, Institut für Informatik, May 2002.

[ECHALK03] G. Friedland, L. Knipping, R. Rojas, E. Tapia; Web Based Education as a Result of AI Supported Classroom Teaching; Lecture Notes in Artificial Intelligence (LNAI) Vol. 2774, pp 290, Springer Verlag, Berlin Heidelberg, Oktober 2003

[GOF94] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides; Design Patterns*;* Addison-Wesley; October 1994

[HÜRST02] W. Hürst, T. Kreuzer, M. Wiesenhütter:

A Qualitative Study Towards Using Large Vocabulary Automatic Speech Recognition to Index Recorded Presentations for Search and Access over the Web.

Proceedings of WWW/Internet 2002, IADIS International Conference, Lisbon, Portugal, November 2002.

[HÜRST03] Wolfgang Hürst: Indexing, searching, and skimming of multimedia documents containing recorded lectures and live presentations. ACM Multimedia 2003: 450-451

[KWOK01] Thomas Kwok, Michael P. Perrone: Adaptive N-Best List Handwritten Word Recognition. ICDAR 2001: 168ff

[MICHAEL03] Alex Michael, Ben Salter; Marketing Through Search Optimization: How to be found on the web; Butterworth-Heinemann; August 2003

[MICROSOFT03] Documentation; Tablet PC Platform SDK, Microsoft, 2003

[PLAMONDO00] Réjean Plamondo, Sargur N. Srihari, IEEE Transactions on Pattern Analysis and machine intelligence, Vol. 22, No. 1, January 2000, p. 63ff

[ROJAS03] R. Rojas, E. Tapia; Recognition of handwritten Digits in the E-Chalk System, 7th International Conference on Document Analysis and Recognition (IDCAR); Edinburgh, August 2003

[SCHOMAKER90] Schomaker, L.R.B., & Teulings, H.-L. (1990). A Handwriting Recognition System based on the Properties and Architectures of the Human Motor System. Proceedings of the International Workshop on Frontiers in Handwriting Recognition (IWFHR). (pp. 195-211). Montreal: CENPARMI Concordia.

[THEIMER04] Theimer, F; Automatische Handschrifterkennung in E-Kreide Dokumenten; Bachelorarbeit; Institut für Informatik, Freie Universität Berlin, April 2004.

Internetreferenzen

[AOFSEWWW]	http://ad.informatik.uni-freiburg.de/mmgroup/aofSEaudio/
[AOFVLWWW]	http://ad.informatik.uni-freiburg.de/mmgroup/aofSEaudio/index01.html
[AOFWWW]	http://ad.informatik.uni-freiburg.de/mmgroup.aof/index.html.de
[CAFFWWW]	http://caffeine.berlios.de/site/
[CICWWW]	http://www.cic.com/
[DOTLANGWWW]	http://www.jasonbock.net/dotnetlanguages.html
[DOTNETWWW]	http://www.microsoft.com/windowsxp/tabletpc/developers/default.asp
[EKREIDEWWW]	http://www.ekreide.de/
[GOOGLEWWW]	http://www.google.com/apis/
[IBMCPWWW]	http://www.research.ibm.com/electricInk/
[IBMGLSWWW]	http://www.research.ibm.com/electricInk/glossary.html
[IBMPENWWW]	http://www.research.ibm.com/electricInk/
[LECTUWWW]	http://www.lecturnity.de/
[NIKIRECWWW]	http://hwr.nici.kun.nl/recog/nici-stroke-based-recognizer.html
[NIKIWWW]	http://hwr.nici.kun.nl/
[PARAWWW]	http://www.paragraph.com
[SCOWLWWW]	http://wordlist.sourceforge.net/
[SOAPWWW]	http://www.w3.org/TR/soap/
[TPSDKWWW]	http://www.microsoft.com/windowsxp/tabletpc/developers/default.asp
[WSDLWWW]	http://www.w3.org/TR/wsdl/

Stand: 13. April 2004