

Automatische Sprachübersetzung von \LaTeX -Dokumenten

Name: Hendrik Theede

Matrikelnummer: 221201256

Abgabedatum: 02.12.2025

Betreuer und Gutachter: Prof. Dr. rer. nat. habil. Clemens H. Cap
Universität Rostock
Fakultät für Elektrotechnik und Informatik

Abstrakt


placeholder

Inhaltsverzeichnis

1	Einleitung	1
1.1	Hintergrund	1
1.2	Thematische Abgrenzung	1
2	Problemfälle	3
2.1	Hinweise für Leser	3
2.2	Elemente eines Dokumentes	6
2.3	Strukturen beliebiger Dokumenten	6
2.4	T _E X's Category Codes	6
2.5	Abweichende Probleme	6
3	Stand der Technik	7
3.1	Anforderungen	7
3.2	Denkbare Ansätze	7
3.3	Existierende Ansätze	7
3.3.1	Testverfahren	7
3.3.2	Durchführung	7
3.3.3	Auswertung	7
3.4	Grenzen der Lösungen	7
3.5	Takeaways	7
4	Eigenständigkeitserklärung	8
	Literatur	9
A	Anhänge	10
A.1	Fontskalierung auf Webseiten	10



1 Einleitung

1.1 Hintergrund

 Die schnellstmögliche und einfache Erstellung von Dokumenten beliebiger Natur (formlos) wird heutzutage oftmals über Produkte bekannter Anbieter abgewickelt (bspw. Microsoft's Word, PowerPoint, ... und die vergleichbaren „LibreOffice“-Software, sowie die korrespondierenden Apple-Produkte). Unterliegen Dokumente allerdings strengeren stilistischen Vorgaben (bspw. bei wissenschaftlichen Veröffentlichungen) entsteht der Vorteil, dass sich diese Vorgaben wie ein Regelsatz behandeln lässt, aus welchem sich bestimmte, vorprogrammierte Dokumentenstrukturen definieren lassen. Hierzu existiert bereits ein geläufiges System welches den Namen \LaTeX trägt (mit dem *La* nach einem der ursprünglichen Entwickler Lamport (1994)), welches selbst auf dem von Knuth (1986) entwickelten Zeichensetzungs-System und der verbundenen Programmiersprache \TeX basiert.

Die \TeX -Syntax selbst basiert auf englischen Begriffen, allerdings ist nicht davon auszugehen, dass nur englischsprachige Menschen \LaTeX und \TeX nutzen werden. Quelltexte und Dokumentenbeschreibungen werden also nicht immer in einer rein englischsprachigen Form vorliegen (z.B. `\chapter{Erstes Kapitel: Einleitung}` oder der Quelltext dieses Werkes). Ein Zurückführen solcher Dokumente in die englische Sprache ist einfach, insofern ein Verständnis der deutschen Sprache besteht (`\chapter{First Chapter: Introduction}`). Die Rückrichtung zeigt sich allerdings genau dann problematisch, sollte ohne Vorkenntnisse von \TeX (bzw. dessen syntaktische Elemente) bestehen. In genanntem Beispiel würde dann das Wort `chapter` aufgegriffen werden und die Zeichenkette `\Kapitel{Erstes Kapitel: Einleitung}` entstehen (ohne weitere, mögliche Anpassungen entsteht hier keine Kapitelüberschrift mehr. Das erstere „Kapitel“ würde ignoriert werden und die innerhalb der Klammern stehende Zeichenkette als einfacher Fließtext gedruckt werden).

1.2 Thematische Abgrenzung

 Bereits Shannon (1948) beschäftigte sich mit theoretischen Grundlagen der Darstellung und Übertragung von Informationen, insbesondere der menschlicher Sprache und Kommunikation. Heutige maschinelle Systeme zu diesem Zweck (bspw. ChatGPT, DeepL, Gemini und co.) wirken zunächst wie „magische“ Blackboxen, arbeiten jedoch auf Grundlage von statistischen Modellen. Spricht man hier von Magie, dann ist jeder Mitarbeiter eines Wetterdienstes oder der Klimaforschung „bezaubernd“. Das zugrundeliegende Konzept kann jedoch sehr schnell auf den Punkt gebracht werden: Eine KI erhält einen Input, für welchen ein bestimmter Output erwartet wird (Beispiel: „Einfügen“ als nächstes Wort eines zu übersetzenden Satzes und „insertion“ im Kontext innerhalb des Satzes als Erwartung (substantiviertes Verb)), und produziert einen Output, welcher mit dem Erwarteten abgeglichen wird. Sollte das Resultat von der Erwartung abweichen (z.B. „insert“ entstehen), so kann dieser Fehler erkannt werden und im Modell dazu beitragen, dass (gegeben einer bestimmten, sequentiellen Folge von Wörtern (Satzstruktur)) dieser „Fehler“ von nun an seltener passiert. Allerdings wird klar, dass eine KI unabdingbar Fehler machen muss, denn nur so kann diese lernen. Dies gilt es stets zu berücksichtigen, wodurch theoretische gesehen immer mehrere Permutationen betrachtet werden müssen, sollten Probleme entstehen, in welchen ein Input mehrere Ausgaben erzeugen könnte. Angemerkt sei zu dem Vorherigen, dass bekannte und bereits rein in den menschlichen Sprachen auftretende Problem (bzw. mögliche Missverständnisse bereits im Verstehen  ner Sprache, ausgelöst durch Mehrdeutigkeiten von Wörtern) in dieser Arbeit nicht näher verfolgt werden.

Aufgrund bekannter Technologien, wie z.B. Google Translate, DeepL und co. sollten solche Fehler innerhalb einzelner Wörter als ein „gelöstes Problem“ betrachtbar sein. Sprachliche Missverständnisse könnten aber immer dann entstehen, wenn sich mehrere Sprachen miteinander vermischen, welche sich ein Lexikon teilen (bspw. hier: $\text{T}_{\text{E}}\text{X}$, \LaTeX , ... und die, zunächst, englische Sprache). Die Frage ob ein Wort übersetzt werden darf oder nicht, unterscheidet einzelne Problemarten (Fälle). Die Hoffnung besteht, dass diese Probleme bereits gelöst sind, allerdings soll *ein Übersetzer* in der folgenden Schilderung alle Fehler machen *dürfen*.

2 Problemfälle

2.1 Hinweise für Leser

Herangehensweise

Die nachfolgende Auflistung an verschiedenen Fällen, welche Probleme gegenüber der T_EX-Syntax, bzw. innerhalb von L^AT_EX-Dokumenten hervorrufen könnten, benötigt per se keine Reihenfolge, da sie möglichst alle behoben sein sollen. Ein zufälliges sequenzielles Nennen dieser würde jedoch die Lesbarkeit und Nachvollziehbarkeit dieser Arbeit mindern und eventuell auftretende, aber übersehene Fälle für Andere nicht schnellig ersichtlich machen. Deshalb wird eine Reihenfolge gewählt, welche nicht auf L^AT_EX-Ebene beginnt, sondern sich so weit wie möglich dem Ursprung dieses Systemes nähert. Von den bereits in T_EX auftretenden Problemen muss ein Weg in Richtung der auf dieser Software aufbauenden Technologien gebahnt werden. Als besonders relevante Systeme kämen hier zunächst L^AT_EX, BibT_EX, TikZ und Weiterführende in Frage, allerdings bringen die Technologien selbst neue Probleme mit sich, da sie auf T_EX aufsetzen und demnach bereits dort entstehen. Eine Fehlerunterscheidung findet nach Funktionalität in einem realen Dokument (herkömmlich, das nach dem Kompilieren entstehende) und einem virtuellen Dokument. Beispiele in realen Dokumenten sind nach auftreten sortiert, wie man sie bei der Erstellung eines beliebigen T_EX-Dokumentes auffinden würde. Die möglicherweise entstehenden Fehler in einem „virtuellen“ Dokument werden nach den Teilen des Dokumentes klassifiziert, welche sie verändern (sollen). Hierbei können entweder einzelne Paragraphen angepasst werden (reiner Fließtext ohne vorgegebene Struktur), Literaturverzeichnisse oder Glossare entstehen (reiner Fließtext mit vorgegebener Struktur), Bilder und Graphiken eingebunden werden oder erstellt werden (Fließtexte innerhalb einer vorgegebenen Struktur), als auch Graphiken innerhalb eines Dokumentes an vorgesehenen Stellen beschrieben sein (Fließtexte in einer losen Struktur) und insbesondere letzteres zu diversen Verschachtelungen führen. Abschließend muss dann allerdings ein Unterpunkt, welcher nach der beschriebenen Reihenfolge in einem der ersten Abschnitte zu erwarten wäre an das Ende gestellt werden, da aus diesem zu viele neue, eigene Probleme entstehen könnten. Zudem sind ein paar zusätzliche, sprachliche und teils unlösbare Probleme gelistet, welche nicht unbedingt als Anforderungen der gegebenen Problemstellung zu verstehen sind und daher als „abweichend“ zu verstehen sind, aus welchen sich aber spätere Erweiterungspotentiale zeigen könnten.

Struktur eines Beispiels

Die Darstellung einzelner Beispiele erfolgt tabellarisch und demonstriert zunächst gewünschte (bzw. zulässige) Verhalten und danach Unerwünschte (bzw. Fehlerhafte). Untige 1 dient hierbei als einfaches Beispiel und zeigt, wie das Übersetzen von Zeichenketten, welche z.B. Tags enthalten, theoretisch unterschiedliche Permutationen erzeugen können. Inwiefern sich die einzelnen zuvor unterschiedenen, möglichen Verhalten unterschiedlich äußern, soll für jedes Beispiel konkretisiert werden. Dieses einleitende Beispiel nutzt daher (zunächst) den rein imaginären Befehl `ink` mit einer auswählbaren Farbe, die auf einen String angewendet wird. Wünschenswert ist, dass nur der in geschweiften Klammern stehende String übersetzt wird, aber ein Auslassen dieses wäre in erster Betrachtung nicht T_EX-Syntax brechend und daher zulässig. Unerwünscht wäre das Übersetzen der zusätzlichen Option in eckigen Klammern, wobei man davon ausgehen würde, dass für diese Option auf einen *default*-Wert zurückgegriffen wird. Dies ist zwar ein syntaktischer Fehler, würde jedoch den Befehl selbst aus-

föhrbar lassen. Als „Fälschlich“ werden demnach Fehler betrachtet, welche die T_EX-Syntax insofern brechen, dass der Kompilierprozess unterbrochen/abgebrochen wird. Abstrahiert man von diesem detaillierterem Beispiel, so können „erwünschte“ Übersetzungen als solche angesehen werden, welche keine Probleme verursachen, „zulässige“ Übersetzungen als solche, die dem Endnutzer/Leser (vollständig) verborgen bleiben, „unerwünschte“ Übersetzungen erst im Dokument sichtbare Fehler produzieren würden, jedoch dessen sprachliche Inhalte nicht verändern und „falsche“ Übersetzungen als solche, die größere Teile des Dokumentes verschwinden lassen oder dafür Sorgen, dass der Quellcode gar nicht erst kompiliert werden kann.

English	Mögliche Übersetzung
Erwünscht	
<code>1\ink[red]{word}</code>	
Zulässig	
<code>1\ink[red]{word}</code>	
Unerwünscht	
<code>1\ink[red]{word}</code>	
Falsch	
<code>1\ink[red]{word}</code>	<code>1\ink[red]{word}</code>

Tabelle 1: Abstrakte Struktur der folgenden Beispiele

- 2.2 Elemente eines Dokumentes
- 2.3 Strukturen beliebiger Dokumente
- 2.4 T_EX's Category Codes
- 2.5 Abweichende Probleme

3 Stand der Technik

3.1 Anforderungen

Abgelitten aus der Problemliste werden hier die Probleme umformuliert als Anforderungen dargestellt und in absteigender Reihenfolge nach Relevanz in Bezug auf die gegebene Aufgabenstellung aufgeführt.

Die Technologien dienen den Anforderungen, sollten sie:

1. kompilierbare Dokumente erzeugen
2. alle Abschnitte in Dokumenten übersetzen
3. kontextuell terminologisch richtige Übersetzungen wählen (die richtigen Lexeme/Wörter treffen)
4. den Kontext selbstständig aus den wörtlichen und erreichbaren (lokalen) Informationen (Dateien) ablesen können
5. den Kontext aus den mathematischen, graphischen, tabellarischen, ... Inhalten einer Datei ablesen können
6. den Kontext aus externen Verweisen (Links) erfassen können (Lokal, als auch Web)
7. ...

3.2 Denkbare Ansätze

Alle Lösungswege und Workflows, die ich mir vorstellen kann und denken konnte. Definiert evtl. Rollen,

3.3 Existierende Ansätze

Alle Technologien, die diese Rolle (n) in den entsprechenden Ansätzen füllen könnten.

3.3.1 Testverfahren

logischerweise: In den denkbaren Ansätzen schon gegenargumentieren, was unsinnig ist und warum. Reduziert die Menge an zu testenden Lösungen.

3.3.2 Durchführung

3.3.3 Auswertung

3.4 Grenzen der Lösungen

3.5 Takeaways

4 Eigenständigkeitserklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig angefertigt und ohne fremde Hilfe verfasst habe. Dazu habe ich keine außer den von mir angegebenen Hilfsmitteln und Quellen verwendet und die den benutzten Werken inhaltlich und wörtlich entnommenen Stellen habe ich als solche kenntlich gemacht. Ich versichere, dass die eingereichte elektronische Fassung mit den gedruckten Exemplaren übereinstimmt.

Rostock, den 02.12.2025

Hendrik Theede

Literatur

Knuth, D. E. (1986), *The TeXbook*, ISBN: 9780201134476, Addison-Wesley Professional.

Lamport, L. (1994), *LaTeX: A Document Preparation System, 2nd Edition*, ISBN: 9780201529838, Addison-Wesley Professional. available at: <https://www.latex-project.org/help/books/tlc3-digital-chapter-samples.pdf> (last Access: 04.10.2025).

Shannon, C. E. (1948), 'The mathematical theory of communication', *The Bell System Technical Journal*, ISSN: 0343-6993 (vol. 27). Harvard Reprint available at <https://people.math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf> (last Access: 16.10.2025).

A Anhänge

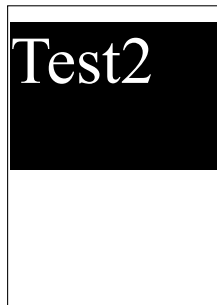
A.1 Fontskalierung auf Webseiten

Beispielsweise produziert die folgende HTML-Notation bei einer Skalierung im Browser von 120 Prozent (Abbildung 2a) und 50 Prozent (Abbildung 2b) jeweilig zwei verschiedene PDF (unter welchen nur Zweitere alle textlichen Inhalte offenbart). Ähnliches kann auch innerhalb T_EX geschehen, sollte

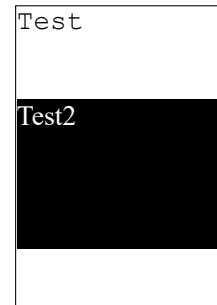
```
<html>
  <head>
    <title>Example</title>
    <style>
      /*formatting options are: none and black*/
      .t{
        font-size:13em;
        height:50%;
      }
      /*formatting option: none = no background, black, courier*/
      .t#none{
        font-family: 'Courier New', Courier, monospace;
      }
      /*formatting option: black = black background, white, serif*/
      .t#black{
        background-color:black;
        color:white;
        margin-top: -2em;
      }
    </style>
  </head>
  <body>
    <div class="t" id="none">Test</div>
    <div class="t" id="black">Test2</div>
  </body>
</html>
```

Abbildung 1: HTML-Beschreibung einer Webseite mit zwei Textflächen

Abbildung 2: Um die Dokumente von der restlichen Papierfläche abzugrenzen wurden schwarze Rahmen mittels TikZ hinzugefügt.



(a) Zuvorige HTML-Beschreibung liefert bei einer Browser-Skalierung von 120% obige Graphik



(b) Zuvorige HTML-Beschreibung liefert bei einer Browser-Skalierung von 50% obige Graphik