Universität Rostock

Traditio et Innovatio

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

Bachelorarbeit am Lehrstuhl für Informations- und Kommunikationsdienste

Fakultät für Elektrotechnik und Informatik

Abstrakt

placeholder

Inhaltsverzeichnis

1 Einleitung

Die schnellstmögliche und einfache Erstellung von Dokumenten beliebiger Natur (formlos) wird heutzutage oftmals über Produkte bekannter Anbieter abgewickelt (bspw. Microsoft's Word, PowerPoint, etc., die vergleichbaren „LibreOffice“, sowie Apple-Produkte). Unterliegen Dokumente allerdings strengeren stilistischen Vorgaben (bspw. bei wissenschaftlichen Abhandlungen) entsteht der Vorteil, dass sich diese Vorgaben wie ein Regelsatz behandeln lassen, aus welchem bestimmte, feste Dokumenten-Strukturen hervorgehen. Hierzu existiert bereits ein geläufiges System namens LaTeX (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 andere Richtung wirft allerdings eine Mehrzahl an Problemen auf, wenn ohne Vorkenntnisse von TeX (bzw. dessen syntaktische Elemente) übersetzt wird. In genanntem Beispiel würde dann das Wort `chapter` aufgegriffen werden und die Zeichenkette `\Kapitel{Erstes Kapitel: Einleitung}` entstehen (ohne weitere, jedoch 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).

Shannon (1948) beschäftigte sich bereits 1948 mit wesentlichen Grundlagen der heutigen Darstellung und Übertragung von Informationen, insbesondere der menschlichen 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. Das zugrundeliegende Konzept kann jedoch sehr schnell auf den Punkt gebracht werden: Eine künstliche Intelligenz (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 (in der Satzstruktur)) dieser „Fehler“ von nun an seltener passiert. Allerdings wird klar, dass eine KI unabdingbar Fehler machen muss, denn nur so kann diese „lernen“. Diese theoretische Grundlage führt dazu, dass immer alle möglichen Permutationen einer Übersetzung in Betracht gezogen werden müssen, so unwahrscheinlich sie auch seien. Angemerkt sei zu dem Vorherigen, dass bekannte und bereits rein in den menschlichen Sprachen auftretende Problem (bzw. mögliche Missverständnisse bereits im Verstehen einer Sprache, ausgelöst durch Mehrdeutigkeiten von Wörtern) in dieser Arbeit nicht näher verfolgt werden.

Bekannte Technologien, wie z.B. Google Translate, DeepL und co. scheinen rein wort-interne Probleme zu lösen. Sprachliche Missverständnisse könnten aber immer dann entstehen, wenn sich mehrere Sprachen miteinander vermischen, welche sich ein Lexikon teilen (bspw. hier: TeX,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 wird *ein Übersetzer* in der folgenden

Schilderung einzelner Problemfälle Fehler machen können *müssen*.

2 Problemfälle

2.1 Herangehensweise

TEX-Quellcodes „einfach“ zu übersetzen, birgt Gefahr Fehler zu produzieren. Bereits innerhalb eines TEX-Quellcodes können (theoretisch) unendlich viele Fehler entstehen, sollten (wohlmöglich) vorfindbare Zeichenketten unbedacht übersetzt werden („unbedacht“: ohne vorher zu prüfen, ob das Wort eine semantische Bedeutung für TEX trägt). Abhängig von der Größe der innerhalb TEX-Quelltexten auftretenden Strukturen (insb. den nach Knuth (1986) vorgesehenen) treten verschiedene Fehlerquellen auf, aber bereits Fehler in den Kleinsten können immense Auswirkungen auf die Inhalte des entgültigen Dokumentes haben und zur Folge tragen, dass der übersetzte Quelltext kein sprachlich äquivalentes Dokument in einer anderen Sprache, als der Ursprünglichen, erzeugt. Hierbei wären die kleinsten denkbaren Strukturen, wenn man sich auf den eigentlichen Verwendungszweck von TEX konzentriet, einzelne Zeichen oder Buchstaben. Diese allein bilden allerdings noch keine interessante (als auch logische) Struktur in TEX, sondern erst konkrete Befehle/Funktionen, sowie diesen übergebbare Parameter. Hierbei zeigen sich diverse Wege, wie sowohl die Übergabe dieser Parameter aussehen *könnte*,als auch die Parameter selbst.Abgesehen der Sinnhaftigkeit vereinzelter Formen solcher Parameter/-übergaben, könnten sie aber mittels TEX produziert werden und müssen daher betrachtet werden. Aufbauend auf diesen kleinsten Strukturen bahnt sich ein Weg in Richtung größerer Strukturen, welche sich nicht nur in einem einzelnen Quelltext befinden, sondern mitunter auf mehrere Quelltextdateien zugreifen (müssen) und nicht zwingend denselben, einheitlichen syntaktischen Vorgaben unterliegen.

Vereinzelte Inhalte von Quellcodes enthalten einzelne String-Literale, deren Übersetzung nicht zwangsweise erforderlich ist oder konsequent und garantierbar richtig ist. Neben solchen Problemen, welche bereits für die Sprachübersetzung allgemein spezifisch sind, bleiben noch Andere, für welche sich nicht immer eindeutige Lösungen finden lassen.

Direkte Quelltext-Beispiele sind dem Anhang **??** zu entnehmen. Dieses Kapitel schildert zunächst das gröbere Aussehen der verschiedenen Fehlerarten, statt konkrete Beispiele aufzuführen.

Aus abstraktester Ebene kann man jedoch danach unterscheiden, ob einzelne dieser Fehler dazu führen, dass überschüssige Wörter im Dokument entstehen, einzelne Wörter (oder gar Dateien) missachtet werden (was provoziert, dass evtl. erforderliche Kontexte verloren gehen) oder fälschliche Übersetzungen entstehen (als Resultat dieses beschriebenen „Verlust eines Kontexts“).

2.2 Elemente in einem Quelltext

2.2.1 Kommandos

2.2.1.1 Unterscheidung Kommandos sind in der reinen TEX-Syntax durch ein Backslash `\` gekennzeichnet und tragen immer eine Bedeutung für die TEX-Engine, wodurch von einem Übersetzen dieser abgesehen sein sollte. Diese Entscheidung ist allerdings für Wörter, welche direkt auf ein Kommando folgen, nicht trivial und es zeigen sich dort bereits innerhalb TEX erste Konflikte und Unterscheidungen, je nach Art des Kommandos. Es existieren zwei Wege einen String zu interpretieren (aus logischer Perspektive). Entweder soll er als Wort (Literal) im Dokument als dieser String angezeigt werden, oder dient als anderer z.B. Parameter für eine Funktion.Beispielsweise müsste innerhalb von Auflistungen der nach einem `\item` folgende String übersetzt werden und

steht vorteilhafterweise vom Kommando getrennt, wodurch er leicht erkannt werden kann. Andererseits wäre in einer Definition mit `\def` vorerst davon abzusehen den folgenden String zu übersetzen, da dieser ein neues Makro/Kommando/etc. definiert (welche Wörter sind, welche dem Übersetzer zu „leicht“ auffallen und von diesem als TEX-Semantik tragende syntaktische Elemente interpretiert werden). Konkreter dürfte `\def\hello` nicht zu: `\def\hallo` werden (es sei denn alle folgenden `\hello` werden auch übersetzt), aber `\item pessimism` sollte zu `\item Pessimismus` werden.[1]

2.2.1.2 Erkennung Zu übersetzende String (Literale) lassen sich nicht, wie es `\item` deuten lässt, anhand fester Zeichen erkennen. Daher muss stets betrachtet werden, ob einzelne Befehle Strings als Parameter aufgreifen sind (unsichtbar im Dokument) oder ob diese Strings, so wie sie stehen, Teil des Dokumentes werden sollen (diese könnten, technisch gesehen, auch als *Parameter* des Befehles interpretiert werden, wovon in diesem Kontext allerdings abgesehen wird, da es spätere Komplikationen hervorrufen würde). Erkannt wird die Zugehörigkeit von Strings ggb. einem Kommando zunächst dadurch, dass diese Parameter unmittelbar und innerhalb eckiger oder geschwungener Klammern auf einen Befehl (bzw. ein Kommando) folgen. Neben diesen üblichen Klammern, welche Parameter umfassen können (`{}` oder `[]`), existieren aber auch Wege, um andere Zeichen an deren Stelle zu nutzen. Einfachstes Beispiel wäre hierfür der *wortwörtliche* `\verb`-Befehl, welcher alle ASCII-Zeichen (in UTF8, inkl. LATIN-Extensions) als eine Umklammerung von (wie gezeigt, mittels: `\verb$\verb$` reproduzierbar) einer Zeichenkette (oftmals: einem Befehl) zulässt, um somit diesen für den TEX's Parser nicht als den Befehl, sondern als *genau* diese Zeichenkette, kenntlich zu machen. Ein Übersetzer muss demnach jegliche Permutation für diesen Befehl abdecken. Hierbei ist jegliches UTF8 (ASCII + LATIN1) Zeichen denkbar, wobei bei den Wort-Charakteren ein Leerzeichen vor den eigentlichen „wortwörtlichen“ Inhalten zu notieren ist.

Beispielsweise kann die Zeichenfolge `\verb athinga` auftreten, welche in `\verb aDinga` übersetzt werden müsste, jedoch nicht in `\verb Ein Dinga` resultieren darf, da dies einen Fehler provoziert, da alle Zeichen bis zum nächsten `E` als Teil der *Verbatim*-Zitierung interpretiert werden würden. Zu übersetzen sind also nur Wörter, welche mit Gewissheit Teil einer menschlichen Sprache sind, ohne dabei Teile dieses beispielhaften Befehles zu beinhalten.

2.2.1.3 Ausnahme Jedes Zeichen in TEX-Quellcodes trägt eine eigene Bedeutung für den Parser der Sprache. Problematisch wird aber, dass sich diese Bedeutung (sogenannter *category code*, nach (Knuth 1986, Appendix D, Seite 371)) mit Hilfe des Befehles `\catcode'` ändern lässt. Auf diese Art und Weise ist es spezifischen Umgebungen (bspw. der von pgf/Ti*k*Z) erlaubt, zusätzlichen Whitespace zu ignorieren oder beispielweise ein LATEX-Kommando anhand eines Ausrufezeichens, statt einem gespiegelten Schrägstrich (Backslash) für den Parser als Kommando erkenntlich zu machen.

2.2.2 Zusätzliche Parameter

Unterscheidung Parameter sind, neben Kommandos, eine weitere Möglichkeit Informationen an TEX zu übermitteln und äußern sich entweder als String-Literale oder Teil einer bestimmten Menge an, für ein Kommando, verfügbaren Optionen (bspw. `\color{red}`). Genauso, wie aus einzelnen Kommandos bereits bekannt ist, sollten

[1] Letzteres ist allerdings nicht immer als gewährleistet zu betrachten.

nur solche Strings übersetzt werden, die auch tatsächlich im Dokument als ebendiese erscheinen. Das wesentliche Problem liegt hier wieder in der Erkennung und Unterscheidung dieser. Die übliche TEX-Notation sieht hierbei geschwungene Klammern für zwingend erforderliche Parameter vor, und eckige Klammern für optionale Parameter. Einzelne Befehle/Kommandos unterscheiden allerdings für optionale und erforderliche Parameter, ob diese als eigentliche Strings ins Dokument eingehen, oder ob sie als alphanumerischer Parameter aufgefasst werden müssen. Auch die logische Reihenfolge solcher erwarteter Parameter, ob optional oder zwingend erforderlich, können einzelne Befehle selbst bestimmen.

Aus theoretischer Sicht ist die Menge an möglichen Parametern (quasi-) unendlich, aber TEX selbst begrenzt diese auf 9 und setzt damit ein technisches Limit, zumindest für `def`.

Dieses ist auf verschiedene Arten umgehbar (einzelne Parameter/Strings können mehr Informationen kodieren, Definitionen mit Relays erweitert werden oder *key-value*-Strukturen genutzt werden (bspw. in `hyperref`'s `\hypersetup` wiederzufinden)), zeigt aber nur Probleme auf, wenn wortsprachliche Strings und keine alphanumerischen Werte innerhalb diesen genutzt werden, solche Strings also als String-Literale genutzt werden und daher übersetzt werden sollen. Konkreter muss abgewägt werden, ob es sich bei einzelnen Parametern um Werte handelt, welche für Operationen ihren exakten Wert tragen müssen (bspw. `\hypersetup{urlcolor=red}` darf weder zu `\hypersetup{URL-Farbe=rot}`, noch `\hypersetup{URL-Farbe=red}` oder `\hypersetup{urlcolor=rot}` werden) oder um Strings, welche im eigentlichen Dokument erscheinen sollen (bspw. `\paragraph{Trick or Treat}` muss zu `\paragraph{Süßes oder Saueres}` werden, darf aber, bekanntlich, nicht `\Paragraph{Süßes oder Saures}` produzieren). Aus den reinen Funktions-/Kommandoaufrufen geht allerdings nicht immer die Information hervor, ob die Übersetzung stattfinden darf. Betrachtet man z.B. die Kommandos des `theorem` Paketes (*package(s)* werden später noch eingeführt), so erfordern diese String-Literale als anzuzeigender String für eine neue nummerierte Umgebung.
Beispielsweise

```
\newtheorem{lemma}{problem}
```

würde bei jedem Aufruf von `\begin{lemma}` (bzw. inkl. `\end{lemma}`) den String `problem` gefolgt mit einer Nummer korrespondierend zu `kapitel.unterkapitel.problemNummer` produzieren. Wichtig ist hierbei, dass

```
\newtheorem{lemma}{problem}
```

nicht zu

```
\newtheorem{Lemma}{Problem}
```

im Deutschen werden darf, sondern idealerweise in

```
\newtheorem{lemma}{Problem}
```

übersetzt werden sollte.[2] Ähnliches ist bei optionalen Parametern denkbar. So dürfte ein `\usepackage[english]{babel}` nicht zu `\usepackage[Englisch]{babel}` werden. Gegenüber zuvor erwähnten *key-value*-Paaren ist in solchen Fällen keine Zuweisung eines Schlüssels zu einem Wert anhand eines = erkennbar.

[2] „Lemma“ statt „Problem“ in diesem Kontext zu produzieren, könnte man als eine Art *Bonus* interpretieren, in welchem der Kontext erkannt wurde und ein „üblicheres“ Wort im Deutschen gewählt wurde.

2.2.2.1 Erkennung Beschränkt man die Anzahl an Parametern (optional und erforderlich) zunächst auf 3, gelangt man bei $2 * 2^3 = 64$ möglichen Permutationen an (ein Parameter kann entweder optional oder erforderlich sein, dessen Inhalte dürfen entweder Übersetzt werden oder nicht und es existieren, bekanntlich, drei Parameter. Die Zahlen 1–4 in beliebiger Reihenfolge in einem 3-elementigen Array anordnen.). Verschiedene zuvor beschriebene Permutationen sind beispielhaft in unterem Beispiel, in welchem zur einfacheren Lesbarkeit die Kürzel *t* für übersetzbare Felder (translatable string) und *p* für nicht zu übersetzende Strings (parameter) stehen.

2.2.2.1.1 Problem bei diesem Ansatz zeigt sich darin, wenn man versuchen würde alle nach einem Kommando stehenden Klammer-Paare ggb. diesem Kommando zurückzuverfolgen. Dieser Ansatz vergisst, dass geschwungene Klammern auch dazu dienen könnten, dass Umgebungen geöffnet und geschlossen werden.

2.2.3 Makros und Umgebungen

2.2.3.1 Umgebungen TEX trägt ein Konzept von „Umgebungen“, für deren Inhalte (ob textlich oder mittels parameterisierter Kommandos produzierte) dieselben Befehle innerhalb dieser prozedural angewendet werden. So könnte man die Farbe der ausgegebenen Zeichenkette mittels `{\color{red} text}` (text) zu rot wechseln. Solche einfachen Umgebungen selbst produzieren noch keine neuen Fehler, allerdings exisiteren spezifische, teils vordefinierte Umgebungen, innerhalb welcher von Übersetzungen abzusehen ist (bzw. zu übersetzende Inhalte nur anhand bestimmter Elemente erkannt werden können).

Mathematische Einfachstes Beispiel hierfür sind mathematische Umgebungen, welche z.B. Formeln, griechische Buchstaben und sonstige mathematische Sonderzeichen darstellen können, jedoch auch an vorher nicht unbedingt bekannten Stellen textliche Inhalte tragen können (siehe: unten). Ein Wechsel zwischen solchen Umgebungen ist theoretisch unendlich oft möglich, solange auf jedes Betreten ein Verlassen (der Umgebung) erfolgt. Hierbei wäre es ideal, wenn alle textuellen Inhalte (des Quellcodes) übersetzt werden und alle Mathematischen nicht (im Beispiel: *reminder:* zu *Erinnerung:* und *frequency* zu *Frequenz*).

```
\begin{align*}\label{problems:tables:eq}
    \sin(\omega t - k\vec{r})                    &= 0 \\
    \arcsin(\sin(\omega t - k\vec{r}))           &= \arcsin(0) \\
    (\omega t - k\vec{r})                        &= 0 \\
    \omega t                                     &= k\vec{r} \text{ reminder: $\omega = 2\pi \text
\end{align*}
```

Insbesondere müssen Fälle betrachtet werden, in welchen Sätze (oder teils Paragraphen) durch mathematische Inhalte unterbrochen/segmentiert werden. So sollte bspw. `The below equation $$x=4+x$$ is false.` zu `Die untenstehende Gleichung $$x=4+x$$ ist falsch.` werden und nicht als zwei einzelne Sätze aufgegriffen werden (`Die untere Gleichung. $$x=4+x$$ Ist falsch.`).Andererseits darf nicht nur, weil eine mathematische Umgebung betreten wurde, auf einmal Texte innerhalb dieser vergessen werden oder erwartet sein, dass die mathematischen Inhalte nicht selbst menschensprachliche Worte tragen. Beispielsweise sollte `$a != b \land a \text{ a isn't b and a}$` in `$a != b \land a \text{ a ist nicht b und a}$`

resultieren, aber `$cars = 3, value = 20 ct$` die Variablen unverändert lassen, dementsprechend nicht `$Autos = 3, Wert = 20 ct$` produzieren. Hierbei sind höhere Verschachtelungen (bzw. Vernestungen) zu prüfen.

Captions und Table of Contents In ähnlicher Art und Weise wie mathematische Umgebungen können auch Tabellen syntaktische Elemente, Umgebungen und textliche Inhalte miteinander vermischen.Neben recht einfach erkennbaren Unterschieden ist es ebenso denkbar, dass vereinzelt Umgebungen innerhalb einzelner Tabellen-Zellen genutzt werden, welche ihrerseits entweder Wörter enthalten, welche nicht übersetzt werden dürfen oder auf String-Literale zugreifen, die an anderer Stelle stehen. Bei der Übersetzung in eine andere Sprache als Englisch muss hierbei natürlich bedacht werden, dass vordefinierte Untertitel bzw. Beschreibungen angepasst werden, damit z.B. eine Tabelle oder Abbildung nicht als *table* oder *figure*, sondern als *Tabelle* oder *Abbildung* berzeichnet wird. Gleiches muss für einzelne Abschnitte, wie zum Beispiel die Überschriften des Inhalts-, Abbildungs- und Tabellenverzeichnisses, als auch einer Nomenklatur oder eines Glossares gewährleistet sein und beinhaltet Änderungen, welche idealerweise in die Präambel des Dokumentes aufgenommen werden.

Tabellen Tabellen sind eine einfache Art und Weise verschiedene, unterschiedliche Inhalte geordnet (gegenüber einander) darzustellen. Technisch sind hier allerdings mehrere Probleme möglich. Zum Einen können höhere Grade an Verschachtelungen entstehen, da es auch (theoretisch) möglich ist, dass man Tabellen innerhalb von Tabellen vorfinden könnte oder Diagramme/Graphen innerhalb einer Tabelle darstellen möchte und die korrespondierenden Beschreibungen daneben.Hierbei muss insbesondere darauf geachtet sein, dass jegliche syntaktische relevanten Strukturen erhalten bleiben und auch, dass jegliches anderes Element von TEX einen Weg in eine Tabelle finden könnte.

Darüber hinaus zeigen sich hier potentielle Schwierigkeiten auf, sollte man es erlauben, dass übersetzte Wörter und Sätze, sollten sie zu groß (im Sinn: Platzbedarf, 2-dimensional) für ihre Zelle werden und dadurch in eine Andere gedruckt werden. Dies birgt Gefahr, dass einzelne Texte andere überlappen könnten, wodurch diese in einem reellen Dokument unlesbar wären. [3]

Verbatim Ähnlich wie bei der Unterbrechung von einzelnen Sätzen durch mathematische Formeln können in TEX auch verschiedene Wege genutzt werden, wortwörtliche Zitate zu verwenden um Sätze oder Paragraphen zu unterbrechen. Neben dem bekannten Befehl `\verb` existiert auch eine großflächigere `verbatim`-Umgebung. Denkbar ist ihre Nutzung um kleinere TEX-Quellcodes[4] darzustellen. Innerhalb dieser Bereiche sind selbstverständlich alle bisher bekannten Fehlerquellen ihrerseits zu umgehen. Wie verhält sich jedoch ein Übersetzer, sollten Teile dieser *verbatim*-Umgebungen zum Wortlaut des formulierten Satzes beitragen? Soll aus `I detest using the \verb|\verb command| unnecessaringly.` eher `Ich lehne ein ünnötiges Nutzen des \verb|\verb` oder `Ich lehne ein unnötiges Nutzen des \verb|\verb Kommando| ab.` werden? Denkbar sind, in diesem Kontext, beide Alternativen, allerdings müssen größere Beispiele die Inhalte der wortwörtlichen Zitierung näher auswerten und als TEX-Quellcode behandeln.

[3] Innerhalb eines PDF-Readers wäre man aber dazu in der Lage solche Zeichenketten herauszukopieren und in einem anderen Medium darzustellen.

[4] Quelltexte anderer Programmiersprachen werden später betrachtet.

Mit eigener Syntax Umgebungen können mitunter ihre eigen syntaktische Forderungen stellen, für welche sie beispielsweise englishe Begriffe nutzen, wie z.B. Ti*k*Z. In solchen Umgebungen muss wieder ausgewertet werden, wann einzelne Zeichenketten übersetzt werden dürfen, denn z.B. nodes erlauben es, dass Texte an diesen platziert werden, aber die Erstellung dieser erfordert den englischen Begriff *at* um ihrer Position innerhalb einer Graphik eine Koordinate zu geben. `\node at (1,1) {hello}` muss in `\node at (1,1) {Hallo}` resultieren und darf nicht zu `\node bei (1,1) {Hallo}` werden.

Kommentare Wohingegen einzeilige Kommentare in TEX mit einem `\%` gekennzeichnet werden, sind auch Mehrzeilige erlaubt, oder solche, die innerhalb ein- und derselben Zeile starten und enden. Hierbei müssen diese Kommentare missachtet werden. So muss ein `This statement is \begin{comment}not\end{comment} true.` nicht zwingend in `Diese Aussage ist \begin{comment}nicht\end{comment} wahr.` enden, darf aber unter keinen Umständen `Diese Aussage ist falsch.` oder Ähnliches produzieren, innerhalb welcher der Kommentar interpretiert wurde und die Ausgabe kommentarlos angepasst wurde.

Makro Erneuerungen Ähnlich wie Kommandos sind Makros eine Art und Weise kürzere Befehle zu nutzen um Ausgaben zu produzieren.Wie bereits einzelne Kommandos dazu in der Lage sind, rein menschensprachliche Strings zu beinhalten (welche ihrerseits übersetzt werden sollten), so sind es auch Makros. Sie vermischen hierüber hinaus die Nutzung von Kommandos und textlichen Inhalten, stellen jedoch in der Syntaktik und der Erkennung solcher Strings zunächst keine weitere Hürde dar. Problematischer wird der Befehl `\renewcommand`, da dieser dazu in der Lage ist auch bekannte Kommandos und Makros zu verändern. Geht man davon aus, dass zum Beispiel Kommandos, wie z.B. `\TeX` oder `\LaTeX` immer nur TEX oder LATEX produzieren, liegt man falsch. Genauso ist es denkbar, dass solche Befehle eine neue Definition erhalten (`\renewcommand{\TeX{}}{This is a TeX example!}`), von denen einzelne Inhalte übersetzt werden müssen, jedoch nur (!) nachdem das Makro erneuert wurde.

2.3 Mehrere Quelltexte

2.3.1 TEX und Andere

Nicht alle Informationen, welche benötigt sind, um ein Dokument zu kompilieren, liegen zwingend in einem Quelltext vor. Insbesondere größere Dokumente, wie bspw. Bücher können davon profitieren ihre Texte auf mehrere `.tex`-Dateien zu verbreiten und ausgehend von einem ursprünglichen Dokument diese per `\include` oder `\input` mit einzubinden. Die Inhalte dieser Dateien sollten hierbei als weitere Teile des vorliegenden Quelltextes (für eine Übersetzung) zu sehen sein und demnach nicht übersehen werden.

Neben der Verwendung mehrerer TEX-Quelltextdateien kann es ebenso von Interesse sein, dass verschiedene Quelltexte (verschiedener Programmiersprachen) in einem Dokument darstellen möchte. Pakete wie `lstlisting` oder `minted` erlauben dies auf verschiedene Wege. Quelltexte (quasi-) beliebiger Sprachen können entweder innerhalb entsprechender Umgebungen eingefügt werden oder es kann ein URL zur Datei-Adresse an passender Stelle angegeben sein. Hierbei sind nicht nur syntaktisch relevante Wörter von bekannten Sprachen zu erhalten, sondern auch z.B. in diesen Quelltexten vorhandene Kommentare, welche die Funktionsweise des Codes erklären

oder auszugebende Strings (falls es sich um Codefetzen mit String-Ausgaben handelt) für einen Übersetzer in Betracht zu ziehen.[5]

2.3.2 Pakete und Klassen

Bereits einige vorherige Beispiele zogen sog. „Pakete“ zur Erklärung der potentiellen Fehlerquellen heran. Mit Hilfe dieser können einem TEX-User (die nicht zu verwechseln mit späteren „Endnutzern“ sind, bei welchen es sich um die Leser des entgültigen, kompilierten Dokumentes handelt) zusätzliche Umgebungen, Kommandos oder Makros zu Verfügung gestellt werden. Diese Funktionen der Pakete sind bekannterweise (genauso, wie vorangegangene Kommandos, Umgebungen, etc.) dazu in der Lage Strings zu tragen, welche möglicherweise im Dokument landen könnten. Zwar technisch eigentlich unterschiedlich, sind Klassen trotzdem gleich behandelbar, da auch in diesen nur Makro-Erstellungen, -Erneuerungen und Umgebungen nachvollzogen werden müssen und deren textliche Inhalte, welche im Dokument landen werden, übersetzt werden sollen.

2.3.3 Hilfsdateien

Einige typische Elemente von Dokumenten erfordern es aus logischer Sicht, dass einzelne Informationen in eine Hilfsdatei geschrieben werden müssen und erst beim einem erneuten Kompilieren eine richtige, vorgesehene Ausgabe produzieren können. Hierbei sind vielerlei Möglichkeiten denkbar, daher wird zunächst mit einer, den meisten TEX-Usern vertrautesten (Quelltext-), Datei gestartet und zwar einer `.bib`, welche für die Literaturverwaltung genutzt werden kann. Innerhalb dieser existieren verschiedene Felder, welche für eine Übersetzung interessant werden könnten, beispielsweise Titel oder Abstrakt einzelner zitierter Werke, allerdings auch Notizen oder sonstige Anpassungen, wie zum Beispiel ein Vermerk „verfügbar unter: <URL>“ im entsprechenden Feld eines BibTeX assozierten Paketes.[6] Bei der Nutzung dieser Technologie (BibTeX) bemerkt ein TEX-User direkt, dass aus logischer Perspektive mehrfach kompiliert werden muss, da zunächst ein TEX-Compiler alle Zitationen im Quellcode sehen muss, diese Information dann über die `.aux` übermittelt, auf welche BibTeX zugreift. Nachdem BibTeX dann diese Datei angepasst hat, kann der TEX-Compiler im nächsten Lauf die Zitationen an richtiger Stelle einfügen und am Ende entsprechend im Literaturverzeichnis vermerken. Die gleiche Denkweise lässt sich auf andere Hilfsdateien anwenden. Bei einem Übersetzen von z.B. Kapiteltiteln reicht es nicht aus nur `\chapter{Incredible}` zu `\chapter{Unglaublich}` zu übersetzen, sondern genauso muss neu kompiliert werden (einfach, wenn die `.toc` angepasst wurde, ansonsten zweifach). Genauso verhält es sich mit Verzeichnissen anderer Art (bspw. Abbildungen, Tabellen, etc.) oder den Inhalten von Glossaren oder Nomenklaturen, sollten sich diese in anderen Dateien befinden.[7]

[5] Wobei hier die Übersetzung eingebundener TEX-Codes insbesondere von Interesse ist, da diese überwiegend textliche Inhalte tragen könnten.

[6] BibLaTeX oder NatBib

[7] Obwohl dies theoretisch gesehen schon unter „TEX“ und andere behandelt sein sollte, wäre eine Auswirkung auf insb. Glossare interessant und inwiefern Abkürzungen angepasst werden. Eine deutsche WDF (Wahrscheinlichkeitsdichtefunktion) wäre im Englischen als PDF (probability density function) zu erwarten.

2.3.4 Zitationen

Hierbei handelt es sich nicht um ein direktes, technisches Problem von übersetzenden Programmen, aber um ein Implizites. Viele Wörter benötigen in bestimmten Kontexten eine spezifische Übersetzung. Diese Kontexte müssen in LaTeX nicht nur aus den direkten Sätzen hervorgehen, sondern können auch hinter Verweisen/Referenzen oder Zitationen an anderer Stelle im Quelltext verborgen stehen. Zitiert man beispielsweise ein Originalwerk vom französischen Mathematiker Évariste Galois und verwendet danach das Wort *fields* („ **?**| defines fields as . . . “) so sollte dieses nicht einfach zum Deutschen *Felder* werden, sondern man würde das Wort *Körper* erwarten (von „endlichen Körpern“, englisch: *finite fields* oder *galois fields*). Fraglich ist hierbei, wie weit ein solcher Kontext nachverfolgt wird. Liegt er direkt in einem vorigen Teil des Quelltextes vor und mittels `~\ref` ein mit passendem *key* ein entsprechendes `\label` verlinkt, so gibt es wenig Gründe, die für einen Verlust des Kontextes sprächen. Ähnlich zu betrachten ist eine (Hyper-) Referenz auf eine andere, lokale Quelltextdatei. Interessant könnten auch BibTeX Zitationen in diesem Kontext werden, da bei diesen verschiedene, theoretisch ausreichende Verweise auf Informationsquellen angegeben werden können, denn ein solcher Literatureintrag muss nicht zwingend vollständig ausgefüllt werden. Es könnte der Fall eintreten, dass nur eine URL oder DOI angegeben wird, eine ISBN (zu mitunter nicht allgemein zugängigen Werken) oder ISSN. Zu erwarten wäre, dass aus solchen Angaben genau so viel Kontext entnommen wird, wie es auch ein Mensch könnte. Dieses Konzept wäre per se auch auf jegliche URL (meist) hinter Sätzen/Paragraphen anwendbar, da dies eine recht schnelle und einfache Art und Weise ist, mit welcher sich nötigste Quellangaben liefern ließen, wird allerdings aufgrunde mangelnder Professionalität solcher Zitationsweisen nicht weiter untersucht.

2.4 Spezifisch

2.4.1 Sprachliche Hürden

Neben technischen Problemen, bzw. Anforderungen an einen Übersetzer können auch einige Tücken oder Unlösbarkeiten näher betrachtet werden und deren „bestmögliche“ Lösung in Betracht gezogen werden. Hierbei existieren zwei Kernprobleme, welche bereits bei der alleinigen Sprachübersetzung (an sich) unabhängig von TeX berücksichtigt werden sollten/könnten. Als Ersteres sind sprachliche Mehrdeutigkeiten von Interesse.

2.4.2 Fehlinterpretierbare Verweise

2.4.3 Layouting-Probleme

Das, in diesem Kontext, interessantere Problem sind der räumliche Platzanspruch, welcher sich für verschiedene Sprachen unterscheidet. Hierbei ist nicht nur gemeint, dass einzelne Sätze, abhängig von der Sprache, länger oder kürzer enden könnten, was ggf. im Layout von Graphiken zu Verschiebungen oder Überlappungen führen könnten (wobei Zweiteres unlesbare Inhalte produzieren könnte), sondern auch einzelne Symbole (bspw. aus Kanji, Hanzi, Devanagari,. . .), die in größeren Zeichenketten/Sätzen in anderen Sprachen resultieren. Insbesondere Graphiken/Graphen/Diagramme, oder sonstige Situationen, in welchen einzelne Zeichenketten an festen, absoluten Positionen stehen müssen, können in die Situation geraten, dass bspw. eine Überschrift, welche nur nach unten hin vergrößert werden kann, ohne außerhalb des Anzeigebereiches der Graphik zu gelangen, könnte die anderen Elemente überdecken.

3 Stand der Technik

3.1 Übersicht

Erste Ansätze zur automatischen Be- und Verarbeitung von LaTeX-Dokumenten lassen sich bis in die 1990er Jahre verfolgen **?** (abgesehen von einem Compiler natürlich). Die Übersetzung von den rein wort-sprachlichen, textuellen Inhalten eines Dokumentes[8] zeigt sich wiederum als ein Problem, welches eine Niche bedienen zu scheint. Von besonderem Interesse für die Sprachübersetzung heutzutage sind große Sprachmodelle, bzw. „künstliche Intelligenzen", wie zum Beispiel DeepL, Google Gemini (Cloud Translate) oder andere GPT-Modelle, welche in diesem Kontext trainiert wurden (GPT=generative pre-trained text). Deshalb seien auch im Kontext der Sprachübersetzung von LaTeX-Dokumenten Ansätze der Art in den Vordergrund gerückt und jegliche betrachtete Technologie kurz aufgeführt.

3.1.1 Nicht auf TeX spezialisierte Werkzeuge

ChatGPT Als „All-Rounder" unter den heutigen künstlichen Intelligenzen ist es denkbar, dass auch ein *general-purpose* Sprachmodell, gegeben eines passenden *prompting*, dazu in der Lage sein **könnte** zuvor beschriebene Fehlerquellen zu meiden. Ein Ansatz dieser Art könnte allerdings genau dann scheitern, sobald einige benötigte Informationen (bspw. die Definition eigener Makros, Umgebungen, . . .) nicht mehr in einem einzigen Quellcode vorliegt.

GitHub Copilot Ein auf die Arbeit mit Quellcode spezialisiertes Tool, wie benanntes „GitHub Copilot", könnte unter Umständen noch effektiver/effizienter/besser mit Quelltexten umgehen und ist ähnlich wie ChatGPT zu behandeln.

DeepL Konkurrierend zu dem jahrelangen Marktführer (in Hinsicht auf durch Computer realisierte Sprachübersetzung) Google, gewann DeepL zunehmend an Bekanntheit und ist neben, Google Translate, ein viel genutztes Werkzeug, insbesondere im Kontext von Anwendungsgebieten, in welchen das Übersetzen durch einen Menschen zu viel (Echt-) Zeit beanspruchen würde. Die API lässt hierbei die Übergabe von einem *Kontext* zu, welcher sich allerdings in der gewählten Wortwahl innerhalb der Ausgabe zeigen wird und keinen Einfluss auf die eingelesenen Inhalte haben sollte. (In dem Sinn, dass durch einen Kontext „LaTeX" nicht davon auszugehen ist, dass Quellcode als Input erwartet ist, sondern Texte über die benannte(n) Technologie(n)). Von besonderem Interesse ist bei einem Ansatz dieser Art also, inwiefern diese Technologie selbstständig dazu fähig ist Quellcode zu erkennen und weniger, ob sie *perfekte* Ausgaben (fehlerfrei) liefert.

Google Cloud Translate Natürlich ist besagter Marktführer für die Übersetzung menschlicher Sprache im gleichen Kontext, wie auch DeepL, zur Bewältigung derselben Aufgabe heranziehbar. Nur muss zunächst klargestellt sein, dass Google Translate nicht gleich Google Cloud Translate ist. Google Translate bezeichnet üblicherweise das bekannte Browser-Tool, welches höchstwahrscheinlich (hoffentlich) mittlerweile auch auf einem

[8] `pgfplots` wäre bspw. dazu in der Lage Audiosignale darzustellen, welche sich maschinell interpretieren ließen. Solche Ansätze werden hier allerdings nicht weiter verfolgt

GPT aufbaut. Da ein solches Tool auf eine möglichst einfache Bedienbarkeit zugeschneidert ist, verliert es an Transparenz hinsichtlich der eigentlichen zugrundeliegenden Technik. Wirklich sicher, dass ein solcher, moderner Ansatz (auf Grundlage großer Sprachmodelle, statt reinen statistischen Modellen) [9]. genutzt wird, kann man sich allerdings nur sein, sollte man vollständige Einsicht in die Zugrundeliegenden Quelltexte erhalten. Dies ist aber, aus verständlichen, marktwirtschaftlichen Gründen nicht immer möglich. Daher wird darauf vertraut, dass auf Angabe des Unternehmens die „Google Cloud Translate"-API so arbeitet, wie es versprochen wird.

Microsoft Translate und Weitere Google, Microsoft und DeepL sind nicht die Ersten und werden auch nicht die letzten Anbieter von maschineller Sprachübersetzung sein. Statt hier noch weitere unkonkrete Ansätze zu listen, wird nun spezifischere Technologien in den Vordergrund gerückt.

3.1.2 Existierende Technologien und Ansätze

GPT-LaTeX-Translator Der von Suñé und Arcuschin (2023) entwickelte Ansatz verfolgt die zuvor beschriebene Idee, ein *general purpose* Sprachenmodell heranzuziehen. Ihre genaue herangehensweise geht aus Zeile 57 des Quellcodes (Python) „GPTTranslator.py" hervor, in welchem sie lediglich zugrundeliegende `.tex`-Dateien so weit zerlegen, dass sie der Anwendungsschnittstelle (damalig) gerecht werden und dann der künstlichen Intelligenz die Information mitteilen, dass LaTeX-Quellcode vorliegt.

Textsynth/trsltx Helluy (2025) arbeitet an einem sehr spezifischen Ansatz, welcher sich der gegebenen Aufgabenstellung sehr gezielt zu nähern scheint. Allerdings zeigt er selbst auf, inwiefern seine Herangehensweise Fehler in LaTeX produzieren kann. Insbesondere verweist er auf Schwächen hinsichtlich eigener Makros und anscheinenden Labels, Referenzen und Zitationen, die die KI eigenständig hinzufügt. Auch bemerkt er, dass einige Umstände zu unvollständigen Übersetzungen führen können, welche es zu erproben gilt.

MathTranslate Sun (2025) präsentiert auf der WebSite von seinem Tool „MathTranslate" einen Ansatz, welcher zunächst äußerst vielversprechend erscheint. Ein näheres Betrachten des vorliegenden Quelltextes zeigt aber schnell potentielle Schwächen auf, welche auf die gewählten Übersetzungs-Softwares zurückführbar sind. Der Quelltext `translate.py` weist hierbei nur auf eine Nutzungsmöglichkeit von entweder (zuvor entgegen argumentiertem) Google Translate oder aber auf Tencent hin. Hierbei wird zweitere Engine näher betrachtet.

TransLaTeX

PolyMath Translator Ohri und Schmah (2020) gingen einen Weg über die Konvertierungs-Software pandoc, um in dieser den abstrakten Syntax-Graphen/Baum zu ermitteln, auf dessen Grundlage sehr klar ist, welche

[9] *Große Sprachmodelle*: Lernen die Sprache, bzw. werden auf Grundlage einer Sprache trainiert, wohingegen *statistische Modelle* sich rein auf die wörtlichen Übersetzungen fokussieren. Wesentlichster Unterschied der Ansätze ist, dass große Sprachmodelle eine Semantik in den Wörtern suchen (in dem Sinn, dass sie die Inhalte „versuchen zu verstehen", wohingegen einfachere statistische Modelle nur die Wörter und deren grammatikalischen Zusammenhänge untersuchen, um daraus die treffenste Übersetzung zu finden)

Inhalte textliche Strings sind, welche in einem kompilierten Dokument angezeigt werden sollen und welche Inhalte lediglich Teile der Dokument-Beschreibung sind. Allerdings zeigt sich ihre veröffentlichte Lösung als heute (Ende 2025) nicht mehr zugänglich, sodass nur noch der Ansatz getestet werden könnte.

3.1.3 Nicht auf Sprachübersetzung konzentrierte Werkzeuge

Pandoc

Translate Package

3.2 Testverfahren

Entlang der zuvor geschilderten Problemfälle ließen sich theoretisch gesehen unendlich viele explizite Fehlerbeispiele produzieren, welche ihrerseits inhaltlich einem *Lorem Ipsum* gleichen könnten („Lorem Ipsum“ bezieht sich auf ein häufig genutztes Beispiel, wenn es darum geht Texte und deren graphische Aufbereitung zu visualisieren. Dieser Platzhalter-Text sieht aus wie Latein, ist inhaltlich aber sinnfrei (**??**)). Da man bei der Nutzung von TEX aber immer die hauptsächliche Nutzung für wissenschaftliche/mathematische Kontexte im Vordergrund belassen sollte, werden insbesondere die auf ArXiV verfügbaren TEX-Quellcodes von Interesse (wobei eine Kompilation von älteren Texten der Art durch **?** zur Verfügung gestellt sind). Dies bedeutet, dass zunächst solche Quelltexte getestet werden, und inwiefern sie „richtige“ Übersetzungen liefern. Dem hierigen Autor ist ein Bestätigen in dieser Hinsicht nur für das Sprachpaar EN-DE möglich, allerdings existieren Technologien (Methoden), wie (Sacre-) BLEU oder TeXBLEU, um algorithmische, maschinelle Bewertungsgrundlagen zu schaffen.

Diese Methodik bedient allerdings bisher nur eine Auswertung der rein sprachlichen Richtigkeit ggb. menschlichen Sprachen und der Maschinensprache TEX. Auf diese Art und Weise werden einige spezifischere Fälle und „Ausnahmen“ nicht näher betrachtet, wodurch gezielt auf diese geprobt werden muss.

Neben den Bulk-Datensätzen von ArXiV entstanden daher auch eigene, kleinere Beispiele, welche dem Anhang **??** zu entnehmen sind.

3.3 Tests

3.3.1 Erzielte Werte

3.3.2 Auswertung

3.3.3 Übrige Probleme

4 Beitrag

4.1 Existierenden Herangehensweisen

4.1.1 Übersetzen des Quellcodes

Diese Herangehensweise verfolgt den Weg den gesamten Quelltext, ohne größere Interpretation, als Eingabe für einen Übersetzer zu nutzen, um daraus einen neuen, übersetzten Quelltext zu erhalten, in welcher sämtliche LaTeX relevanten Inhalte unverändert (ggb. dem Original) vorzufinden sind. Dieser neue Quelltext soll dann, genauso wie der Originale, mit Hilfe eines bekannten TeX-Compilers (bspw. `pdflatex`, `xelatex`, `lualatex`, ...) in eine PDF überführt werden.

4.1.2 Quelltext-filternde Ansätze

4.1.3 Ausweichen in andere Formate

4.1.4

4.2 Andere Lösungswege

4.2.1 Abstrakte Beschreibung

4.2.2 Technische Umsetzungsmöglichkeiten

4.3

5 Fazit

5.1 Zusammenfassung

5.2 Ausblick

5.3 Weiterführend

6 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