

Markus Kohm: *Verkleinerte und vergrößerte Ausgabe mit L^AT_EX*, Die T_EXnische Komödie 1/1999, S. 7–26.

Reproduktion oder Nutzung dieses Beitrags durch konventionelle, elektronische oder beliebige andere Verfahren ist nur im nicht-kommerziellen Rahmen gestattet. Verwendungen in größerem Umfang bitte zur Information bei DANTE e.V. melden. Für kommerzielle Nutzung ist die Zustimmung der Autoren einzuholen.

Die T_EXnische Komödie ist die Mitgliedszeitschrift von DANTE, Deutschsprachige Anwendervereinigung T_EX e.V. Einzelne Hefte können von Mitgliedern bei der Geschäftsstelle von DANTE, Deutschsprachige Anwendervereinigung T_EX e.V. erworben werden. Mitglieder erhalten Die T_EXnische Komödie im Rahmen ihrer Mitgliedschaft.

Verkleinerte und vergrößerte Ausgabe mit L^AT_EX

Markus Kohm

Eine der immer wiederkehrenden Fragen zu L^AT_EX betrifft die korrekte Vorgehensweise zur Vergrößerung oder Verkleinerung von Dokumenten. Da diese Fragen unterschiedlich motiviert sein können und der Fragesteller sich oftmals über seine Motive nicht vollständig klar ist, fällt es den Experten häufig nicht leicht, die Frage in der Hinsicht korrekt zu beantworten, daß der tatsächlich gewünschte Effekt erzielt wird.

Im nachfolgenden Artikel werden deshalb ausgehend von einem häufig anzutreffenden Fallbeispiel verschiedene Möglichkeiten vorgestellt und gegeneinander abgewogen. Aufmerksame Leser der deutschsprachigen DE-T_EX-/DANTE-FAQ [10] wird es nicht verwundern, zumindest eine der hier empfohlenen Lösungen auch dort in Kurzform wiederzufinden.

Vorbemerkung

Im Gegensatz zum sonst üblichen Vorgehen findet sich die Quintessenz nicht im allerletzten Abschnitt. Um den Anfänger nicht mit Grundlageninformationen vom Hauptthema abzulenken, wird im Hauptteil vieles nur leicht angeschnitten. Für den interessierten Leser finden sich genauere Informationen dann ab dem Abschnitt „Genau gesagt“.

Ebenso sind besondere Problemfälle nicht im Hauptartikel aufgeführt. Diese werden stattdessen im Abschnitt „Problemkinder“ vorgestellt. Da diese Kenntnisse jedoch für die vorgestellten Lösungswege nicht unbedingt erforderlich sind, finden sich die eigentlichen Lösungen bereits vor diesen Abschnitten.

Das Problem der Vergrößerung

Als häufigste Problemstellung ergibt sich, daß eine Druckerei die Druckvorlage für ein Werk mit dem Zielformat DIN-A5 statt im Zielformat auf A4 vergrößert geliefert bekommen will. Dies ist häufig für die Herstellung des Druckmediums – beispielsweise eines Films – notwendig, um die gewünschte Güte oder Auflösung zu erreichen.

Das DIN-A-Format und das übereinstimmend definierte ISO-A-Format entsteht im Halbierungsverfahren aus der Grundgröße A0. Von dieser Definition abgeleitet kann man auch festlegen, daß das nächst kleinere Format jeweils durch Division der Seitenlängen mit $\sqrt{2}$ gewonnen wird (siehe „Genau gesagt“, „Papierformate aus dem ABC“).

Umgekehrt entspricht die Vergrößerung von DIN-A5 auf DIN-A4 also zunächst einer Verdopplung der Papiergröße beziehungsweise einer Vergrößerung aller Längen um Faktor $\sqrt{2}$. Die anschließende optische Verkleinerung in der Druckerei führt dann wieder zu einer Verkleinerung aller Längen um denselben Faktor. Da diese Verkleinerung optisch erfolgt, ergibt jeder Druckpunkt der Vorlage auch einen Punkt im Druckbild. Damit wird sowohl die horizontale als auch die vertikale Auflösung ebenfalls um den Faktor $\sqrt{2}$ erhöht. Dies ist letztlich der Zweck der ganzen Prozedur. Die Frage bleibt, wie man die Vergrößerung bei Vorlagen, die mit \TeX oder \LaTeX erstellt werden, am besten erreicht.

Die grundsätzliche Idee

Als erstes könnte man natürlich die Vorlagen direkt in der von der Druckerei verlangten Größe DIN-A4 erstellen. Dabei ergibt sich aber beispielsweise das Problem, daß mit entsprechend großer Grundschrift gearbeitet werden müßte. Davon abgesehen, daß es zu den Standardklassen keine Schriftgrößenoptionen 14pt oder größer gibt, wie man sie für Zielgrößen von 10 pt und mehr benötigte, wäre es keineswegs gesagt, daß solche Dateien nach der Verkleinerung in der Druckerei zum gewünschten Ergebnis führten.

Unter typographischen Gesichtspunkten spielt neben den Aspekten, die den Satzspiegel betreffen (siehe „Genau gesagt“, „Vergrößerung oder Verkleinerung



Abbildung 1: Der Buchstabe „W“ in der Größe 24,88pt.

eines Satzspiegels“), die Wahl des Zeichensatzes eine entscheidene Rolle. Dazu muß man wissen, daß eine entsprechend vergrößerte 10pt-Schrift noch lange keine 14pt-Schrift und umgekehrt eine verkleinerte 14pt-Schrift keine 10pt-Schrift ist. Verdeutlicht wird dies in Abbildung 1, die in der Mitte ein auf 24,88 pt vergrößertes „W“ aus einer 5pt-Schrift und links davon ein unvergrößertes „W“ direkt aus einer 24,88pt-Schrift zeigt. Wollte man also für die Vergrößerung statt einer 10pt-Grundschrift einfach eine 14pt-Grundschrift verwenden, so wäre das Ergebnis nach der später wieder erfolgenden Verkleinerung nicht das tatsächlich gewünschte.

Um sowohl im Endergebnis den korrekten Zeichensatz als auch einen korrekten Satzspiegel zu erhalten, ist zu empfehlen, die Vorlage zunächst im Zielformat A5 zu erstellen und lediglich für den Ausdruck zu vergrößern. Das hat darüber hinaus den Vorteil, daß der Autor sich nicht mit der Frage beschäftigen muß, wie das ganze wohl aussieht, wenn es schließlich in der Druckerei verkleinert ist. Stattdessen kann er die Wirkung direkt an seinen unvergrößerten Probeausdrucken begutachten.

Die unterschiedlichen Möglichkeiten der Vergrößerung

Die Vergrößerung selbst kann auf unterschiedlichste Weise stattfinden. Als erste Idee könnte man am Drucker vielleicht eine andere Auflösung einstellen als die, in der tatsächlich gedruckt wird. Stellt man bei einem Drucker beispielsweise als Auflösung 212 dpi ein und füttert ihn anschließend mit Pixeldaten, die eigentlich für 300 dpi bestimmt waren, so wird das Druckergebnis um den gewünschten Faktor $300/212 \simeq \sqrt{2}$ größert. Allerdings verringert man dabei die tatsächliche Auflösung im Gegenzug um eben diesen Faktor auf die Druckerauflösung von 212 dpi. Ergebnis solchen Handelns ist, daß das Ziel der Auflösungserhöhung in der Druckerei zunichte gemacht wird. Das rechte „W“ in Abbildung 1 demonstriert dies, wobei der Unterschied in der Auflösung zwischen dem von METAFONT und vom Drucker vergrößerten Buchstaben von 1270 dpi zu 254 dpi aufgrund der hohen Auflösung nicht unbedingt sofort ins Auge springt.

Die Methode der Manipulation der Druckerauflösung ist meist nicht ganz einfach zu beschreiten, da man irgendwie den Datenstrom vom DVI-Treiber zum Drucker manipulieren muß. Werkzeuge hierfür sind mir nur für PostScript bekannt. Außerdem bieten Drucker im allgemeinen nicht beliebige Auflösungen.

DVI-Treiber verfügen allerdings über eine Möglichkeit, auf die Größe oder Vergrößerung Einfluß zu nehmen. Zur Erklärung dieser Option muß ein wenig ausgeholt werden.

In einer DVI-Datei sind Größeninformationen nicht in *dots per inch* (dpi) sondern in *units* abgelegt. Die Größe einer *unit* selbst ist dabei im Kopf der DVI-Datei als Festkommazahl und einem zusätzlichen Faktor abgelegt. Dieser zusätzliche Faktor in Promille ist abgekürzt mit *mag* für das englische *magnification*, zu Deutsch *Vergrößerung*, bezeichnet. Normalerweise ist dieser Wert auf 1000, was keiner Vergrößerung oder Verkleinerung entspricht, voreingestellt (siehe auch „Grundlage“, „Größenangaben in DVI-Dateien“).

Es bietet sich fast von selbst an, für Größenmanipulationen nicht den erwähnten Festkommawert, sondern eben die Vergrößerungsvariable *mag* zu verwenden. Damit sind dann Vergrößerungen und Verkleinerungen auch eindeutig als solche erkennbar, wohingegen eine Veränderung des anderen Wertes ihre Ursache auch darin haben können, daß mit einer anderen Basiseinheit gearbeitet wurde. Dies ist beispielsweise bei der DVI-Ausgabe von groff [1] der Fall. Bei groff handelt es sich um ein Dokumentformatierungssystem, das verschiedene Ausgabeformate beherrscht und hauptsächlich für Programmanleitungen im Unix-Bereich verwendet wurde.

Zu betrachten wäre dann noch, ob bei dieser Art der Vergrößerung auch der richtige Font geladen wird. Eine Fontdefinition in der DVI-Datei besteht unter anderem aus dem Namen des Fonts und seiner Designgröße [2]. Die Designgröße ist ebenfalls in der Einheit *unit* angegeben. Vom DVI-Treiber wird immer nach dem Zeichensatz in der angegebenen Designgröße gesucht (siehe „Genau gesagt“, „Wie ein DVI-Treiber einen Zeichensatz findet“).

Eine Veränderung von *mag* bewirkt also über die Veränderung der *unit* auch, daß andere Zeichensatzdateien geladen werden. Wichtig ist festzuhalten, daß vorhandene Zeichensatzdateien in Pixeldarstellung nicht auf eine andere Größe umgerechnet werden. Stattdessen werden Zeichensätze, die in der entsprechenden Vergrößerung oder Verkleinerung – im allgemeinen von METAFONT – berechnet wurden, angefordert.

Deshalb bieten die DVI-Treiber auch die Möglichkeit, den *mag*-Wert in der DVI-Datei durch einen anderen Wert zu ersetzen oder mit einem zusätzlichen

```
\documentclass{minimal}
\begin{document}
  \fontsize{10}{10}\selectfont W
\end{document}
```

Abbildung 2: Die Referenzdatei zur Ermittlung der verwendeten Fonts.

Wert zu multiplizieren. Die Nutzung dieser Möglichkeit führt dann dazu, daß sämtliche Ausgaben entsprechend größer oder kleiner werden.

Aber nicht nur die DVI-Treiber kennen die Variable *mag*. \TeX selbst bietet diese Variable ebenfalls an. Mit der folgenden Zuweisung in der Präambel eines Dokuments würde beispielsweise eine Vergrößerung um den gewünschten Faktor $\sqrt{2}$ erreicht.

```
\mag=1414
```

Dies ist auch in \LaTeX zulässig. Da der Wert, der $\backslash\text{mag}$ zugewiesen wird, direkt in den Kopf der DVI-Datei geschrieben wird und sich somit auf das gesamte Dokument auswirkt, sollte die Zuweisung ebenfalls im Kopf des Dokuments also in jedem Fall vor $\backslash\text{begin}\{\text{document}\}$ erfolgen. Es ist sogar zulässig diese noch vor der Auswahl der Dokumentklasse vorzunehmen.

Der einzige relevante Unterschied zwischen der Veränderung von *mag* über eine Option beim Aufruf des DVI-Treibers und der Zuweisung an $\backslash\text{mag}$ im Dokumentkopf ist im Abschnitt „Problemkinder“, „Die true-Einheiten“ zu finden.

Vergrößerung in der Anwendung

In diesem Abschnitt werden nun einige Beispiele für die Vergrößerung demonstriert. Dazu wird die Referenzdatei aus Abbildung 2 auf unterschiedliche Art und Weise vergrößert. Die Referenzdatei erzeugt nur einen einzelnen Buchstaben auf dem Papier. Es wird kontrolliert, welcher Font dafür verwendet wird. Als Referenz-DVI-Treiber wird *dvips* verwendet. Diesen kann man mit der Option `-d 4` dazu veranlassen auszugeben, welchen Font er gerade lädt. Abbildung 3 zeigt die zur Referenzdatei gehörende Ausgabe von *dvips*.

Uns interessiert an dieser Stelle aus der Ausgabe von *dvips* nur die hervorgehobene Zeile. Diese bedeutet, daß der Zeichensatz *cmr10* in der Auflösung 300 dpi für die Schriftgröße 10.0pt geladen wird. 300 dpi ist hierbei die voreingestellte Auflösung des Druckers. Die 10 im Fontnamen *cmr10* steht ebenfalls für 10 pt.

```

This is dvips 5.58 Copyright 1986, 1994 Radical Eye Software
' TeX output 1997.11.21:1050' -> refdatei.ps
Defining font () cmr10 at 10.0pt
Loading pk font cmr10.300pk at 10.0pt
<texc.pro>. [1]

```

Abbildung 3: dvips-Ausgabe zu der Referenzdatei aus Abbildung 2.

Es wird also die 10pt-Version von cmr in der Größe 10 pt und der Druckerauflösung 300 dpi geladen, was keiner Vergrößerung entspricht. Verwendet man statt der OT1- die T1-Codierung und die EC-Fonts, so wird stattdessen ecrm1000.300pk at 10.0pt geladen. Hier steht 1000 im Namen nicht etwa für 1000 pt, sondern für 10,00 pt, da bei den EC-Fonts ein anderes Namensschema verwendet wird als bei CM-Fonts.

Ergänzen wir nun die Präambel obiger Grunddatei um eine Zeile für die geforderte Vergrößerung um den Faktor $\sqrt{2}$.

```
\mag=1414 % Vergrößerung um ca. Wurzel 2
```

Daraufhin gibt dvips an, daß nun cmr10.424pk at 10.0pt geladen wird. Dies bedeutet also, daß die 10pt-Version von cmr in der Auflösung 424 dpi und der Größe 10 pt verwendet wird. Da die Ausgabe auf dem Drucker jedoch nicht mit 424 dpi sondern mit 300 dpi erfolgt, entspricht dies einer Vergrößerung um den Faktor $424/300 \simeq \sqrt{2}$.

Verdeutlicht wird die unterschiedliche Behandlung von CM- und EC-Schriften noch, wenn man explizit eine Schrift in 14,40pt-Größe anfordert. Dies ist möglich, indem man in der Referenzdatei die Werte 10 im Größenauswahlbefehl, `\fontsize{10}{10}`, jeweils durch 14.40 ersetzt. Man erhält dann je nach Codierung als Ergebnis beim Lauf von dvips eine der folgenden Zeilen.

```

Loading pk font cmr12.360pk at 14.4pt
Loading pk font ecrm1440.300pk at 14.4pt

```

Bei CM wird also eine vergrößerte Schrift benutzt, wenn auch nicht cmr10 sondern cmr12, also ausgehend von Entwurfsgröße 12 pt, während bei EC die entsprechende Entwurfsgröße angefordert wird. Ganz nebenbei sind wir so auf einen interessanten Vorteil der EC-Schriften gegenüber den CM-Schriften gestoßen: es existieren wesentlich mehr Entwurfsgrößen. Übrigens wären die oben angegebenen Zeichensätze genau diejenigen, die angefordert würden, wenn wir gemäß Problemstellung nicht in A5, sondern in A4 arbeiten würden. Bei der Verkleinerung in der Druckerei würde so ein verkleinerter 12pt- beziehungsweise

| | | | | |
|--------------|----------|----------|----------|----------|
| Sollgröße: | 10pt | 14pt | 14pt | 14pt |
| \mag: | 1000 | 1414 | 1440 | 1000 |
| Zeichensatz: | cmr10 | cmr10 | cmr10 | cmr12 |
| Auflösung: | 254dpi | 359dpi | 366dpi | 305dpi |
| Beispiel: | W | W | W | W |
| Zeichensatz: | ecrm1000 | ecrm1000 | ecrm1000 | ecrm1440 |
| Auflösung: | 254dpi | 359dpi | 366dpi | 254dpi |
| Beispiel: | W | W | W | W |

Tabelle 1: Zusammenstellung der verwendeten Vergrößerungen und Zeichensätze bei einer Druckerausgabe mit 254dpi.

14pt-Zeichensatz entstehen und nicht der gewünschte 10pt-Zeichensatz. Wir sehen also, daß die Entscheidung in der Zielgröße zu arbeiten vorteilhaft war.

Es stellt sich nun die Frage, welches Ergebnis wir erhalten, wenn wir dvips per Option sagen, daß die Ausgabe der Grunddatei um 1,414 vergrößert werden soll. Es ist nicht sonderlich verwunderlich, daß das Ergebnis exakt das gleiche wie bei Angabe eines entsprechenden \mag-Wertes in der Präambel ist, da beide Vorgehensweisen zu einer entsprechenden Änderung des mag-Wertes führen.

Tabelle 1 zeigt eine Zusammenfassung der Beispiele aus diesem Kapitel. In den Spalten steht jeweils untereinander als oberstes die Sollgröße und darunter der mag-Wert, dann jeweils für OT1- und T1-Codierung der verwendete Zeichensatz, darunter die daraus und aus dem mag-Wert resultierende Auflösung ausgehend von einer Druckerauflösung von 254 dpi und darunter jeweils ein Buchstabenbeispiel, wie es mit Hilfe der Referenzdatei erstellt worden ist. Um das Raster der Buchstaben sichtbar zu machen, wurden sie in einer geringeren Auflösung erzeugt und zusätzlich vergrößert. Es entstanden so doppelt große Buchstaben in einem Zehntel der Auflösung der vorliegenden Zeitschrift „Die T_EXnische Komödie“ von 1270 dpi. Die neben der um $\sqrt{2}$ angegebenen Vergrößerung um den Faktor 1,440 spielt im Abschnitt „Genau gesagt“, „Gute Ergebnisse mit weniger Zeichensatzberechnungen“ eine Rolle.

Ergebnismanipulation

Nachdem `dvips` schon diverse PostScript-Dateien erstellt hat, liegt die Idee nahe, statt der DVI-Datei doch die PostScript-Ausgabe der Grunddatei direkt mit PostScript-Mitteln zu vergrößern. Diese Idee ist weder abwegig noch schwer umzusetzen, gibt es doch in den PS-Utilities [3] genau die richtigen Werkzeuge für allerlei Manipulation an PostScript-Dateien.

Wissen muß man nun allerdings, daß `dvips` die Daten aus den oben bezeichneten `pk`-Dateien als Fonts in die PostScript-Ausgabe einbindet. Diese `pk`-Dateien enthalten keine Vektor- sondern nur Bitmap-, also Pixelinformationen der Fonts. Die von `dvips` eingebundenen Fonts sind also auch Bitmap-Fonts. Vergrößerung oder Verkleinerung von Pixelbildern führt aber immer zu erheblichen Qualitätsverlusten. Dieser Weg, der für das rechte „W“ in Abbildung 1 verwendet wurde, ist also zunächst abzulehnen. Anders sieht es aus, wenn statt der Bitmap-Fonts echte PostScript-Fonts eingebunden werden. Von den CM-Fonts selbst gibt es beispielsweise auch mehrere PostScript-Versionen [8], die jedoch gewissen Copyright-Bedingungen unterliegen und auch nicht immer vollständig sind. Dem PostScript-Treiber `dvips` kann man per Option bzw. Fontmap-Datei auch beibringen, solche Fonts zu verwenden. Allerdings vergrößern sich dabei die erzeugten PostScript-Dateien erheblich.

Mit Hilfe von `dvips` wäre es auch möglich, andere PostScript-Fonts zu verwenden. Leider haben PostScript-Schriftfamilien wie Times den Nachteil, daß sie in der Regel nur in einer Entwurfsgröße vorliegen. Werden andere Größen benötigt, so entstehen diese dann wiederum durch einfache Skalierung der vorhandenen Grundgröße. Daß dieses Vorgehen nachteilig und typographisch bedenklich ist, wurde bereits im Abschnitt „Die grundsätzliche Idee“ erläutert und durch Abbildung 1 verdeutlicht. Gerade die Verfügbarkeit diverser Entwurfsgrößen insbesondere der CM- und EC-Standardzeichensätze und deren automatische Auswahl stellt einen entscheidenden Vorteil von \LaTeX dar. Man sollte ihn deshalb nicht ohne guten Grund aufgeben.

Verwendet man `dvips` mit Bitmap-Fonts ist außerdem zu beachten, daß diese in der Auflösung des Druckers erstellt wurden, mit dem der Ausdruck dann auch tatsächlich stattfinden soll. Ist dies nicht der Fall, müßten die Zeichensätze wiederum als Bitmaps verkleinert oder vergrößert werden, was ja nicht ohne, teilweise erhebliche, Qualitätsverluste möglich ist. Neben der Auflösung gibt es weitere Geräteeigenschaften zu beachten. All diesen Faktoren wird bei der Fontberechnung durch `METAFONT` mit Hilfe des Geräteparameters `\mode` Rechnung getragen. Nähere Erklärungen hierzu sind [5] und der Geräteparameterdatei `modes.mf` zu entnehmen.

Nebenbei haben wir also gelernt, daß `dvips` die Bitmap-Fonts im *mode* des Zieldruckers benötigt. Dies ist auch dann der Fall, wenn eine Druckerei direkt mit PostScript-Dateien beliefert werden kann. Gegebenenfalls muß man also in der Druckerei nachfragen, mit welchem Gerät und in welcher Auflösung die Ausgabe erfolgt.

Zwischenfazit

Geht es darum, Druckvorlagen in einer anderen als der Zielgröße zu erstellen, damit bei der optischen Verkleinerung auf die Zielgröße eine höhere Auflösung erreicht wird, so stellt die Manipulation des Vergrößerungsfaktors *mag* der DVI-Datei einen Weg dar, der auf jedenfall zum gewünschten Ergebnis führt. Ob die Manipulation direkt in der \TeX -Datei über eine Zuweisung an `\mag` oder durch eine Option beim Aufruf des DVI-Treibers erfolgt, ist dabei hauptsächlich eine Frage der Handhabung. Mir persönlich erscheint die Verwendung einer DVI-Treiber-Option praktischer, da ich so alle Probeausdrucke ohne zusätzliche Angabe in der Zielgröße erhalte und ohne Änderung der Dokumentdatei jederzeit auch Druckvorlagen erstellen kann.

Demgegenüber führt die nachträgliche Manipulation der PostScript-Datei nicht immer zu einem optimalen, sondern teilweise auch zu einem dem eigentlichen Ziel entgegenstehenden, nämlich in der Auflösung nicht verbesserten, Ergebnis.

Die direkte Dokumenterstellung in einem anderen als dem Zielformat, also insbesondere in der von der Druckerei gewünschten Größe, scheidet aus typographischen Gründen von vornherein aus.

Problemkinder

Eigentlich sollte man denken, wir wären nun am Ziel angelangt und könnten als Ergebnis festhalten, daß eine Vergrößerung oder Verkleinerung sehr einfach entweder per `\mag`-Zuweisung in der Präambel oder per DVI-Treiber-Option realisiert werden kann. Leider gibt es ein paar Problemfälle, die in den nachfolgenden Abschnitten kurz angesprochen werden sollen. Teilweise ist dafür auch das tiefere Verständnis aus dem Abschnitt „Genau gesagt“ von Vorteil.

Bilder in Fremdformaten

Werden Bilder nicht mit \TeX - oder \LaTeX -Mitteln erstellt, sondern in Fremdformaten realisiert, so findet bei deren Ausgabe nicht zwangsläufig eine Umrechnung der Größe statt. Ursache dafür ist, daß diese Bilder mit Hilfe der

`\special`-Anweisung eingebunden werden. `\special`-Anweisungen werden als unveränderte Textinformation in die DVI-Datei geschrieben. Angenommen, die Anweisung `\special{hline 10cm}` würde von einem DVI-Treiber als Linie der Länge 10 cm interpretiert. Im Falle einer Vergrößerung würde in der DVI-Datei nach wie vor „`hline 10cm`“ stehen, da die textuelle Information „10cm“ von \TeX unberührt bleibt. Der DVI-Treiber müßte also wissen, daß eine Vergrößerung oder Verkleinerung vorliegt, um die Linie gegebenenfalls in der Größe zu ändern. Analoges gilt, wenn andere Zahlenwerte, beispielsweise zur Beschreibung der Ausmaße eines PostScript-Bildes, angegeben werden.

Wie wir jedoch festgestellt haben, stellt eine vom Standardwert 1000 abweichende *mag*-Angabe im Kopf der DVI-Datei ein eindeutiges Indiz für eine Vergrößerung bzw. Verkleinerung dar. Die Frage ist, ob dieses von DVI-Treibern auch tatsächlich entsprechend interpretiert wird.

Tatsächlich haben wir Glück und sowohl bei einer Vergrößerung per `\mag=1414` als auch bei einer Vergrößerung per `dvips`-Option wird eine nachgeladene PostScript-Datei mit vergrößert, wie ein Experiment mit einer Testdatei aus Abbildung 4 zeigt. Diese Datei erzeugt zunächst mit Hilfe der `filecontents`-Umgebung die PostScript-Datei `a.ps` und bindet diese dann mit Hilfe des `graphics`-Pakets in das \LaTeX -Dokument ein. Vergrößert man dieses Dokument entweder durch eine zusätzliche Zeile `\mag=1414` oder durch Angabe der Option `-x 1414` beim Aufruf von `dvips`, so wird die Graphik ebenfalls vergrößert. Abbildung 5 zeigt das unvergrößerte Ergebnis. Die PostScript-Datei `a.ps` ist übrigens der Anleitung zum `graphics`-Paket entnommen.

Damit ist also auch im Falle der Vergrößerung oder Verkleinerung die Einbindung von Bildern und Graphiken zumindest als PostScript-Datei unproblematisch.

Der Nullpunkt einer Seite

Bei DVI-Dateien liegt der Nullpunkt, von dem aus alle Koordinaten berechnet werden, in der linken oberen Ecke jeder Seite. Allerdings ist er aus historischen Gründen nicht beim Zusammentreffen der beiden Papierkanten, sondern jeweils ein Inch von diesen Kanten in das Innere des Papiers verschoben. Letzlich bedeutet dies, daß ein DVI-Treiber zu den ermittelten Koordinaten jeweils ein Inch hinzuaddieren muß. Hat man also beispielsweise einen oberen und linken Rand von jeweils einem Inch, so sind in \LaTeX tatsächliche Randeinstellungen von Null für `\topmargin` und `\oddsidemargin` zu finden. Würde ein DVI-Treiber bei Vergrößerung und Verkleinerung also nur diesen Null-Wert ändern

```

\begin{filecontents*}{a.ps}
%!
%%BoundingBox:100 100 172 172
100 100 moveto
72 72 rlineto
72 neg 0 rlineto
72 72 neg rlineto
stroke
/Times-Roman findfont
72 scalefont
setfont
(A) show
showpage
\end{filecontents*}
\documentclass{article}
\usepackage[dvips]{graphics}
\pagestyle{empty}
\begin{document}
  \fbox{\includegraphics{a.ps}}
\end{document}

```

Abbildung 4: Beispieldatei zur Vergrößerung von PostScript-Bilder.

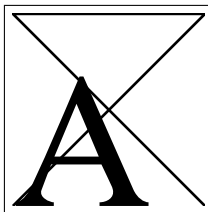


Abbildung 5: Das unvergrößerte Ergebnis der Beispieldatei aus Abbildung 4.

und danach wieder ein Inch hinzuzählen, würden sich die Ränder nicht korrekt mitändern.

Auch hier dient wieder der *mag*-Wert als eindeutiges Indiz, daß eine Vergrößerung oder Verkleinerung vorliegt. In diesem Fall wird der Standardoffset-Wert von einem Inch einfach ebenfalls mit vergrößert bzw. verkleinert.

Wir halten fest, daß sich Vergrößerung oder Verkleinerung in korrekter Weise auf den Nullpunkt auswirkt und daher auch in dieser Hinsicht unproblematisch ist.

Die *true*-Einheiten

In \TeX und \LaTeX ist es möglich, Größenangaben dadurch von Vergrößerung und Verkleinerung auszunehmen, daß man der Einheit ein `true` voranstellt. Man schreibt dann beispielsweise `12trueem` an Stelle von `12mm`. Allerdings betrifft dies nur \TeX -interne Vergrößerung und Verkleinerung durch Zuweisungen an `\mag`. Die Frage ist, in welcher Weise diese Größen vor Beeinflussung durch den geänderten *mag*-Wert ausgenommen werden.

\TeX selbst stellt sicher, daß alle *true*-Einheiten von der Vergrößerung oder Verkleinerung unberührt bleiben, indem diese entsprechend dem veränderten *mag*-Wert umgerechnet werden. Die DVI-Dateien kennen keine *trueunit*, also keine Größenangaben, auf die der *mag*-Wert im Kopf der DVI-Datei nicht anzuwenden ist. Anders gesagt, der *mag*-Wert muß auf alle Größenangaben in den DVI-Dateien angewendet werden.

Zur Verdeutlichung werden im Beispiel aus Abbildung 6 zwei Längenvariablen mit den Namen `\NormaleLaenge` und `\TrueLaenge` definiert. Beiden wird als Länge 1414pt zugewiesen, der `\NormaleLaenge`-Variable in gewohnter Weise, der Variable `\TrueLaenge` jedoch als 1414truept. Die Gesamtausgabe wird außerdem per `\mag=1414` um den Faktor 1,414 vergrößert. Desweiteren wird \LaTeX in diesem Beispiel dazu veranlaßt, die aktuellen Werte der beiden Variablen nach der Zuweisung auszugeben. Dies geschieht mit Hilfe des `\the`-Primitivs, das im \TeX book [4] beschrieben ist. Eine DVI-Datei wird nicht erzeugt, da für uns nur die Bildschirmausgabe des `\typeout`-Makros interessant ist. Weil \TeX für die Vergrößerung lediglich den *mag*-Wert im Kopf der DVI-Datei verändert, erwarten wir, daß der Wert von `\NormaleLaenge` unverändert mit 1414pt angegeben wird. Da die Werte der Längenvariablen mittels `\the` immer in pt und nicht etwa in truept ausgegeben werden, erwarten wir jedoch, daß für `\TrueLaenge` nicht 1414pt ausgegeben werden, wie dies der

```

\documentclass{minimal}
\mag=1414 % Vergrößerung
% Längenvariablen definieren
\newlength{\NormaleLaenge}
\newlength{\TrueLaenge}
% Werte zuweisen
\setlength{\NormaleLaenge}{1414pt}
\setlength{\TrueLaenge}{1414truept}
% Tatsächlich gespeicherte Werte ausgeben
\typeout{\NormaleLaenge=\the\NormaleLaenge}
\typeout{\TrueLaenge=\the\TrueLaenge}
% Keine DVI-Datei erzeugen
\begin{document}\end{document}

```

Abbildung 6: Beispiel zur Verdeutlichung von *true*-Größen.

Fall wäre, wenn wir auf die Vergrößerung mittels `\mag=1414` verzichtet hätten. Tatsächlich werden unter anderem die folgenden beiden Ausgabezeilen erzeugt.

```

\NormaleLaenge=1414.Opt
\TrueLaenge=1000.Opt

```

Demnach wird die `\TrueLaenge` also um den `\mag` zugewiesenen Wert verkleinert, damit bei der anschließenden Vergrößerung – durch den `mag`-Wert 1414 in der DVI-Präambel statt des normalen Wertes 1000 – wieder 1414pt entstehen. Demgegenüber werden aus den 1414,0pt von `\NormaleLaenge` infolge der Multiplikation mit `mag/1000` dann effektive 1999,396pt.

Ganz anders sieht es aus, wenn die Vergrößerung mit Hilfe einer Option des DVI-Treibers geschieht. Da \TeX in diesem Fall keine Informationen besitzt, daß eine Vergrößerung erfolgen soll, werden `truept` und `pt` gleich behandelt. Sie bleiben von \TeX unverändert. Somit werden identische *unit*-Werte in die DVI-Datei geschrieben. Vom DVI-Treiber selbst sind die Werte nicht mehr unterscheidbar und werden damit beide gleichermaßen vergrößert.

Diese unterschiedliche Behandlung von *true*-Größen ist im Normalfall der einzige signifikante Unterschied zwischen einer `\mag`-Zuweisung in \TeX bzw. \LaTeX und der Angabe einer `mag`-Option beim Aufruf des DVI-Treibers.

Spickzettel per `\mag`?

Im Gegensatz zur bisherigen Problemstellung könnte man sich natürlich auch vorstellen, daß jemand ein Minidokument in A6 und 8pt-Schrift erstellen will. Da die Standardklassen weder eine Option `a6paper` noch `8pt` bieten, liegt die Idee nahe, stattdessen in A5 mit 11pt zu arbeiten und die oben genannten Möglichkeiten anschließend zur Verkleinerung zu verwenden. Für eine Verkleinerung gilt nahezu alles bisher festgestellte, handelt es sich dabei doch lediglich um eine Vergrößerung um einen Faktor kleiner als 1, also mit einem `mag`-Wert kleiner 1000. Wer den Artikel bisher gründlich gelesen hat, weiß daher auch, daß dieser Ausweg nur ein Holzweg wäre. Zum einen ist, wie bereits eingangs ausgeführt wurde, ein A5-Satzspiegel nach der Verkleinerung eben noch immer ein verkleinerter A5- und kein A6-Satzspiegel. Zum anderen unterscheidet sich die Entwurfsgröße 8pt eben von einer auf 8pt verkleinerten Entwurfsgröße mit 11pt, wie Abbildung 1 überdeutlich demonstriert.

Die Lösung für dieses Problem liegt in der Definition der entsprechenden Papiergröße und Grundschrift, die ich in diesem Artikel jedoch schuldig bleibe. Hinweise hierzu finden sich beispielsweise in der Anleitung für Paketautoren [9], die jeder L^AT_EX-Distribution beiliegt, in der Implementierungsdokumentation der Standardklassen [7] sowie in einem früheren Artikel der Zeitschrift „Die T_EXnische Komödie“ [6].

Genau gesagt

In den folgenden Abschnitten werden Grundlagen behandelt, die für das eigentliche Verständnis des Artikels nicht zwingend erforderlich sind. Sie erschienen mir aber interessant genug, um sie wenigstens kurz anzuschneiden und weiterführende Literatur dazu anzugeben.

Papierformate aus dem ABC

DIN-A-, -B-, -C- und -D-Formate sowie die gleichbezeichneten Papierformate nach ISO sind jeweils ausgehend von den Grundgrößen A0, B0, C0 und D0 definiert. Dabei gilt beispielsweise für A0 folgendes Seitenverhältnis von längerer Seite zu kürzerer Seite.

$$\frac{H_{A0}}{B_{A0}} = \frac{1189 \text{ mm}}{849 \text{ mm}} = \sqrt{2}.$$

Bei diesem Verfahren wird zur Gewinnung des nächst kleineren Papierformats das Papier in der Mitte der Längsseite halbiert. Somit gilt allgemein für das

| ISO-Format | Breite/mm | Höhe/mm |
|------------|-----------|---------|
| A0 | 841 | 1189 |
| B0 | 1000 | 1414 |
| C0 | 917 | 1297 |
| D0 | 771 | 1090 |

Tabelle 2: Die Grundgrößen des ISO-A-, -B-, -C- und -D-Formates.

A-Format:

$$H_{A(n+1)} = B_{A(n)} = \sqrt{2} \cdot H_{A(n)} \quad (1)$$

$$B_{A(n+1)} = \frac{1}{2} H_{A(n)} = \sqrt{2} \cdot B_{A(n)}. \quad (2)$$

Gerechnet wird dabei mit abgerundeten Millimeterwerten.

Analoges gilt auch für B-, C- und D-Format. Die Grundgrößen sind in Tabelle 2 angegeben.

Vergrößerung oder Verkleinerung eines Satzspiegels

Will man bei der Veränderung des Papierformates die Schriftgröße dazu passend ändern, so ist es aus typographischer Sicht nicht ausreichend, alle Werte in `size10.clo` mit dem entsprechenden Faktor zu multiplizieren, um so beispielsweise zu `size12.clo` zu kommen. Bei der Erstellung der Dateien für die Grundschriftgrößen ist weit mehr als nur einfache Mathematik zu beachten. Der *Graueindruck* einer Seite spielt hierbei eine zentrale Rolle. Dieser entsteht, wenn man eine Seite auf gewisse Entfernung unscharf betrachtet. Der Graueindruck soll möglichst gleichmäßig sein. Dadurch kann es je nach Schriftbild und Schriftgröße beispielsweise notwendig werden, den Zeilenabstand größer oder kleiner zu wählen.

Aber auch die Verteilung von Text und Rändern ist wichtig. Zwar kann bei einer insgesamt proportionalen Größenänderung die optimale Zeilenlänge beibehalten werden. Dabei blieben aber Grenzwerte, die ebenfalls wichtig sind, gänzlich unberücksichtigt. So wird ein zu kleiner Rand rasch als gar nicht mehr existent wahrgenommen. Aber auch rein praktische Gründe spielen eine Rolle. Beispielsweise sollte in einem Buch geblättert werden können, ohne dabei Fingerabdrücke zu hinterlassen. Aufgrund der häufig nicht vollständigen Abriebfestigkeit und Fettbeständigkeit von Druckerschwärze entstehen Fingerabdrücke meist dann, wenn man beim Umblättern oder Halten eines Buches gezwungen ist, in den Textbereich zu fassen. Ränder werden darüber hinaus

unter Umständen vom Leser auch für handschriftliche Anmerkungen verwendet. Ein nur wenige Millimeter breiter Rand ist hierfür nicht zu gebrauchen. So spielt also auch die beabsichtigte Verwendung eines Druckwerks eine Rolle bei der Festlegung des Satzspiegels.

Da ein Rand bei der Verkleinerung aber ebenfalls kleiner wird, erfüllt er die einmal getroffenen Mindestanforderungen anschließend unter Umständen nicht mehr. Hinzu kommt, daß auch diese Mindestanforderungen vom Seitenformat und der Schriftgröße abhängen können. Eine Erhöhung der Seitenlängen des Papiers um einen konstanten Wert muß also nicht zwangsläufig eine Vergrößerung der Ränder um denselben Wert rechtfertigen.

Allein schon aus diesem Grund ist es nahezu zwingend, den Satzspiegel für das Zielformat festzulegen und dann sinnvollerweise auch mit diesem Format zu arbeiten. Eine für einen Zwischenschritt der Produktion erforderliche Größenveränderung sollte sich nicht negativ auf das Endergebnis auswirken also möglichst nur für diesen einen Schritt erfolgen.

Größenangaben in DVI-Dateien

Wie bereits erwähnt, werden alle Größenangaben in DVI-Dateien in *units* angegeben. Die Größe einer *unit* ist dabei nicht in der Definition des DVI-Formats festgelegt, sondern eine variable Größe. Sie hängt unter anderem vom Programm ab, das DVI-Ausgaben tätigt und wird im Kopf der DVI-Datei abgelegt [2]. Um sich auf Ganzzahlberechnungen beschränken zu können, erfolgt die Speicherung in Form eines Bruches zweier natürlicher Zahlen, *num* und *den*. Diese beiden Abkürzungen stehen für die englischen Bezeichnungen *numerator* (Zähler eines Bruchs) und *denominator* (Nenner eines Bruchs). Der so entstehende Wert in 10^{-7} m wird außerdem noch mit dem Tausendstel des Vergrößerungsfaktors *mag* multipliziert. Dieser Variablenname ist die Abkürzung des englischen Wortes *magnification* (Vergrößerung). Damit gilt also:

$$unit = \frac{mag}{1000} \cdot \frac{num}{den} \cdot 10^{-7} \text{ m.}$$

Die Werte für *num* und *den* werden bei \TeX -Ausgaben durch die Grundeinheit *sp* (*scaled point*) bestimmt, in der \TeX rechnet [4].

$$1 \text{ sp} = 1/65536 \text{ pt} = 254/473628672 \text{ cm} = 25400000/473628672 \cdot 10^{-7} \text{ m}$$

Damit ist *num* = 25400000 und *den* = 473628672. Im Normalfall, also ohne Vergrößerung und Verkleinerung ist *mag* = 1000.

Will man die Ausgabe vergrößern oder verkleinern, so muß man lediglich eine der drei Variablen entsprechend verändern. Um Vergrößerungen und Verklei-

nerungen eindeutig identifizieren zu können, ist es sinnvoll nicht *num* oder *den*, sondern ausschließlich *mag* für diesen Zweck zu verwenden.

Das Maximalmaß, das \TeX in eine DVI-Datei schreiben kann, sind 2^{31} sp \simeq 11,5 m. Mit anderen Programmen und anderen Werten für *num* und *den* gelten andere Maximalwerte.

Wie ein DVI-Treiber einen Zeichensatz findet

Generell sind Verzeichnis und Name einer Zeichensatzdatei, wie sie DVI-Treiber verwenden, sowohl vom System als auch vom DVI-Treiber selbst abhängig. Bei Verwendung von pk-Fonts gibt es jedoch ein paar grundsätzliche und ein paar häufige Gemeinsamkeiten.

Zum Laden eines Fonts rechnet der DVI-Treiber die Designgröße, die in der DVI-Datei in *units* angegeben ist, im allgemeinen in dpi um und gewinnt den Dateinamen dann aus dem Fontnamen und dieser Größe in dpi. Der Fontname ist normalerweise identisch mit dem Namen der entsprechenden tfm-Datei ohne deren Endung „tfm“. Die tfm-Datei beinhaltet die sogenannte Fontmetrik. Sie ist alles, was \TeX selbst über den Font an Informationen benötigt. Vom DVI-Treiber wird sie normalerweise ebenfalls verwendet.

Der dpi-Wert wird dabei auf ganze dpi gerundet. Leider verwenden unterschiedliche Treiber hierzu teilweise unterschiedliche Rundungsverfahren.

Je nach System wird dann beispielsweise bei einer Druckerauflösung von 300 dpi und ohne Vergrößerung oder Verkleinerung die Datei `FONT0300\cmr10.pk` oder `cmr10.300pk` geladen. Vereinzelt wird statt dem dpi-Wert auch direkt der *mag*-Wert im Datei- oder Verzeichnisnamen des Fonts verwendet.

Der Suchpfad für diese Datei enthält in der Regel auch noch den Namen des METAFONT-Modes, für den die Fonts erzeugt wurden, oder eine Abkürzung für diesen. Zur Kontrolle ist die Designgröße des Zeichensatzes dann auch noch in der pk-Datei selbst gespeichert. Ebenso ist der METAFONT-Mode normalerweise als Kommentar in der pk-Datei zu finden. Ob tfm- und pk-Datei zusammenpassen, wird über eine Prüfsumme kontrolliert. Diese Prüfsumme ist für alle pk-Dateien, die zu derselben tfm-Datei gehören, also insbesondere unabhängig vom METAFONT-Mode, gleich.

Findet ein DVI-Treiber eine pk-Datei nicht, so gibt es unterschiedliche Strategien, die teilweise auch kombiniert verwendet werden.

- Es wird eine Toleranzabweichung in der Designgröße zugelassen. Diese Abweichung kann je nach DVI-Treiber bei wenigen Promille bis zu mehreren

Prozent liegen und ist teilweise auch vom Anwender einstellbar. Je nach Größe der tatsächlichen Abweichung kann die Ausgabe durch den DVI-Treiber dann erkennbare oder auch für den normalen Leser nicht sichtbare Fehler aufweisen.

- Es wird ein Ersatzzeichensatz in der gewünschten Designgröße verwendet. Da dieser Zeichensatz in der Regel eine andere Fontmetrik und auch ein anderes Aussehen als der eigentlich gewünschte hat, sind die Abweichungen hier mit bloßem Auge zu erkennen und können sich auch in völlig falschen Zeichen auswirken.
- Es wird ein Ersatzzeichensatz in einer zuvor festgelegten Designgröße verwendet. Normalerweise ist dies ein Zeichensatz, der immer vorhanden sein sollte, wie etwa `cmr10` in der Auflösung des Druckers. Diese Methode kann zu nichts dienen als der notdürftigen Wiedergabe des Inhaltes eines Textes.
- Statt der entsprechenden Zeichen wird entsprechender Leerraum ausgegeben. Diese Methode ist häufig als letzte Lösung zu finden. Sie hat gegenüber der Methode der Ersatzzeichensätze den Vorteil, daß sie die Ausgabe weniger verfälscht, der Mißstand eindeutiger zu erkennen und der Rest besser zu beurteilen ist.
- Es wird ein Programm gestartet, das dafür sorgt, daß der fehlende Zeichensatz in der gewünschten Designgröße beispielsweise mit Hilfe von `METAFONT` erzeugt wird. Diese Methode ist inzwischen die Regel und teilweise soweit perfektioniert, daß bei echten PostScript-Fonts nicht `METAFONT`, sondern ein Programm aufgerufen wird, das `pk-Fonts` aus PostScript-Fonts erzeugen kann.
- In eine Datei wird eine Information über den fehlenden Zeichensatz geschrieben. Diese Datei kann dann von einem anderen Programm für die Erzeugung aller fehlenden Zeichensätze einer DVI-Datei ausgewertet werden. Diese Methode wird hauptsächlich auf Systemen mit traditionell wenig Arbeitsspeicher verwendet, bei denen fehlende Zeichensätze aus zu erwartendem Speicherplatzmangel nicht unmittelbar erzeugt werden können.
- Fast immer weist der DVI-Treiber zusätzlich zur sonstigen Vorgehensweise auf den Mißstand hin.

Gute Ergebnisse mit weniger Zeichensatzberechnungen

Eine Suche nach `pk-Dateien` für um Faktor 1,414 vergrößerten Zeichensätzen ergibt auf Standardsystemen, daß solche Zeichensätze normalerweise nicht exi-

stiert. Sie müssen also von METAFONT erst noch erzeugt werden. Ursache ist, daß das Vergrößerungsschema von L^AT_EX, das mit Befehlen wie `\small`, `\large`, `\Large` etc. verwendet wird, nicht auf $\sqrt{2}$ sondern auch 1,2 basiert. In diesem Schema wäre also $1,2^2 = 1,440$ ein möglicher Vergrößerungswert. Verwendet man diesen Wert, so lädt `dvips` stattdessen `cmr10.432pk at 10.0pt`. Dieser Zeichensatz ist normalerweise direkt verfügbar. Verwendet man jedoch die T1-Codierung und EC-Fonts, so wird entsprechend `ecrm1000.432pk at 10.0pt` angefordert. Eine entsprechende Datei ist normalerweise jedoch nicht verfügbar und muß also von METAFONT erst noch berechnet werden. Das liegt daran, daß in der OT1-Codierung tatsächlich kein `cmr14` existiert und deshalb vom L^AT_EX-Größenumschaltbefehl stattdessen `cmr10` in entsprechender Vergrößerung angefordert wird. Bei T1-Codierung existiert jedoch ein `ecrm1440`, also eine 14,40 pt-Variante von `ecrm`, die normalerweise von L^AT_EX auch verwendet wird. Da `\mag` jedoch kein L^AT_EX-Befehl ist, sondern auf T_EX-Ebene durchgeführt wird, wird hier nicht eine andere Entwurfsgröße, sondern die vorgegebene Entwurfsgröße in anderer Auflösung erzeugt.

Die tolerierbar geringe Abweichung von weniger als zwei Prozent erspart also bei den OT1-codierten CM-Schriften die Berechnung einer erheblichen Menge von Zeichensätzen. Demgegenüber bringt sie bei den T1-codierten EC-Schriften keinerlei Vorteil. Zu dieser Abweichung ist noch zu bemerken, daß bei T_EX und L^AT_EX in der Regel mit der Einheit Point (pt) gerechnet wird, wohingegen im europäischen und insbesondere deutschsprachigen Buchdruck normalerweise eigentlich Didot Punkte (dd) verwendet werden. 1157 dd entsprechen 1238 pt. Sind also von der Druckerei eigentlich 10-Punkte-Schriften gefordert, so arbeitet man bei Verwendung von 10pt-Schriften eigentlich mit etwas zu kleinen Zeichen, nämlich 9,3 dd. Die Vergrößerung um Faktor 1,440 statt 1,414 wirkt dem teilweise entgegen. Die sich dadurch ergebende Abweichung bei den Rändern liegen im Bereich von Bruchteilen von Millimetern und sind damit in der Regel vernachlässigbar.

Fazit

Es gibt verschiedene sehr einfache Möglichkeiten L^AT_EX-Dokumente zu vergrößern, beispielsweise um Druckereianforderungen zu erfüllen. Damit ist es auch kein Problem, der Empfehlung nachzukommen, ein Dokument mit L^AT_EX immer in der Größe zu entwerfen, in der das Endergebnis erscheinen soll, und gegebenenfalls nur die Druckvorlagen in geänderter Größe auszugeben. Die beiden ausführlich aufgezeigten Wege der Manipulation des *mag*-Wertes sind bis auf wenige Ausnahmen äquivalent. Vernachlässigt man Fragen der Hand-

habung, so bleibt als tatsächliches Entscheidungskriterium lediglich die Frage, ob *true*-Größen unberührt bleiben sollen oder nicht.

Literatur

- [1] James Clark: *groff*; Juni 1995; ftp: [prep.ai.mit.edu/pub/gnu/](ftp://prep.ai.mit.edu/pub/gnu/).
- [2] The TUG DVI driver standards committee: *The DVI Driver Standard, Level 0*; Draft version 0.05; Aug. 1991; CTAN: [/tex-archive/dviware/driv-standard/level-0](#).
- [3] Angus J. C. Duggan: *PSUtils*; 1995; CTAN: [/tex-archive/support/psutils/](#).
- [4] Donald E. Knuth: *The T_EXbook*; Bd. A von *Computers and Typesetting*; Addison-Wesley Publishing Company; Reading, Mass.; 19. Aufl.; 1990.
- [5] Donald E. Knuth: *The METAFONTbook*; Bd. C von *Computers and Typesetting*; Addison-Wesley Publishing Company; Reading, Mass.; 6. Aufl.; 1991.
- [6] Markus Kohm: KOMA-SCRIPT – *Eine Alternative zu den Standardklassen?*; *Die T_EXnische Komödie*; 2/96, S. 14–33; 1996.
- [7] Leslie Lamport, Frank Mittelbach und Johannes Braams: *Standard Document Classes for L^AT_EX version 2_ε*; Juni 1997; CTAN: [/tex-archive/macros/latex/base/classes.dtx](#).
- [8] Basil K. Malyshev: *BaKoMa Fonts Collection*; Nov. 1994; CTAN: [/tex-archive/fonts/cm/ps-type1/bakoma](#).
- [9] L^AT_EX3 Project: *L^AT_EX 2_ε for class and package writers*; Juni 1997; CTAN: [/tex-archive/macros/latex/base/clsguide.tex](#).
- [10] Bernd Raichle und Thomas Hafner: *DE-T_EX-/DANTE-FAQ*; Dez. 1997; CTAN: [/tex-archive/usergrps/dante/de-tex-faq](#).