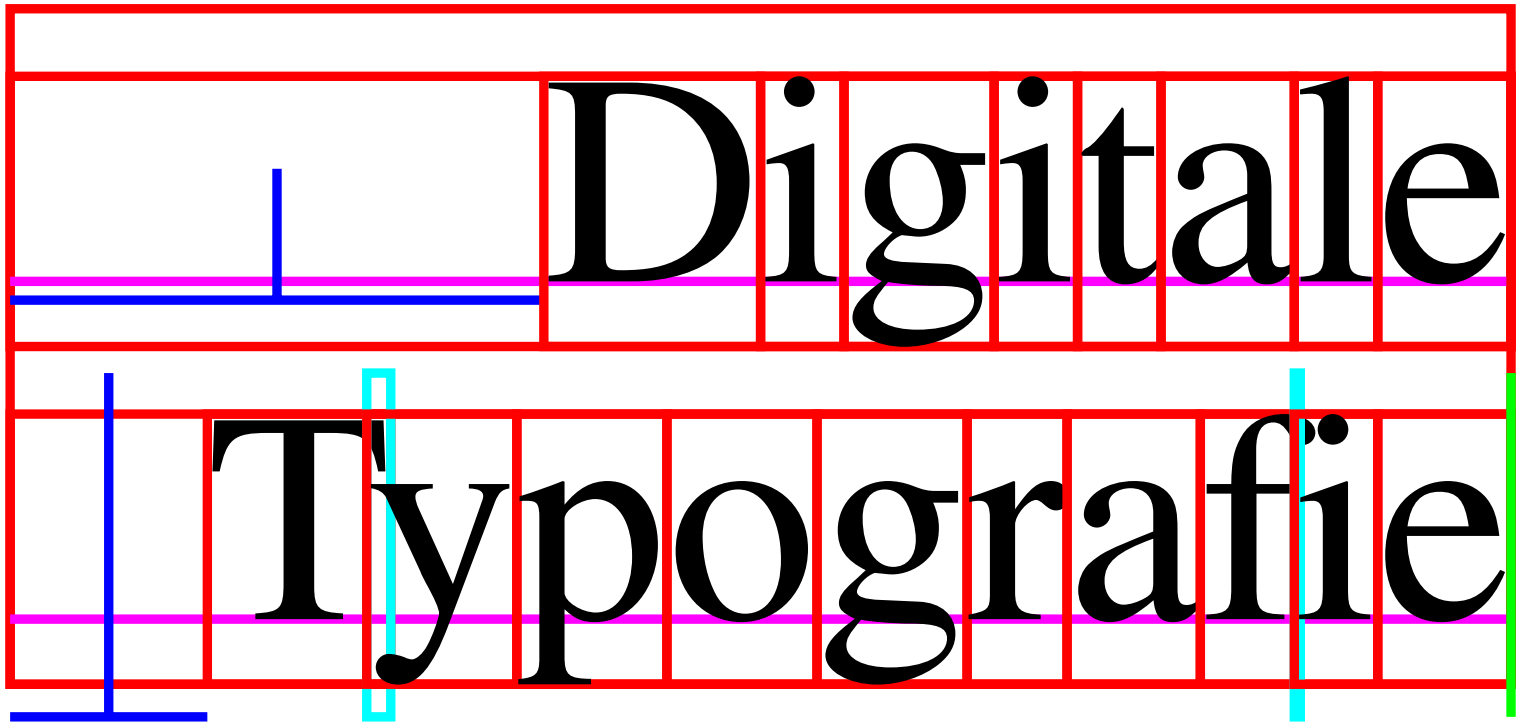


Andreas F. Borchert
Universität Ulm
16. Juli 2018

Inhalte:

- Einführung und historischer Überblick
- Von der geometrisch definierten Fläche zum Pixelraster mit einer Einführung in PostScript und MetaPost
- Digitale Repräsentierungen von Schriften
- Einführung in die Typografie
- Ausgewählte Algorithmen und Verfahrenstechniken

Was ist Typografie? 3

- »Typography exists to honor content.« (Robert Bringhurst)
- »Typografie ist keine Kunst. Typografie ist keine Wissenschaft. Typografie ist Handwerk.« (Hans Peter Willberg)
- »Typografie, das ist die Inszenierung einer Mitteilung in der Fläche, so die kürzeste Definition, die ich kenne.« (Erik Spiekermann)
- »Good typography therefore is a silent art; not its presence but rather its absence is noticeable.«
  (Mittelbach and Rowley: *The pursuit of quality – How can automated typesetting achieve the highest standards of craft typography?*)

Wozu dient Typografie?

Robert Bringhurst fasst es folgendermaßen zusammen:

»[...] typography should perform these services for the reader:

- invite the reader into the text;
- reveal the tenor and the meaning of the text;
- clarify the structure and the order of the text;
- link the text with other existing elements;
- induce a state of energetic repose, which is the ideal condition for reading.«

Was ist digitale Typografie? 5

- »Digital typography is the technology of using computers for the design, preparation, and presentation of documents, in which the graphical elements are organized, positioned, and themselves created under digital control.« (Richard Rubinstein)
- »[...] the problem of printing beautiful books had changed from a problem of metallurgy to a problem of optics and then to a problem of computer science. [...] The future of typography depends on the people who know the most about creating patterns of 0s and 1s; it depends on mathematicians and computer scientists.« (Donald E. Knuth)

Wie erschließt sich die digitale Typografie? 6

- »Digital typography is a field that overlaps two others: that of classical or letterpress typography and that of computer science. Our basic postulate is that digital typography should not be taught without teaching classical typography at the same time.« (Jaques André)

- Das Hauptziel ist das Erlernen der Fähigkeit, Methoden der Informatik anwenden zu können, um Problemstellungen der Typografie zu lösen.
- Um dieses zu erreichen, ist es sinnvoll, zunächst typografische Programmiersprachen und heute gebräuchliche Repräsentierungen für Schriften kennenzulernen.
- Der Erwerb von Grundkenntnissen in der Typografie selbst ist die Voraussetzung dafür, typografische Problemstellungen zu verstehen. Hierbei wird auch darauf einführend eingegangen, wie aus einer Zielsetzung eines Textes eine dazu passende Typografie entwickelt werden kann.

Syllabus: Was ist es nicht? 8

- Auch wenn wir einige der Algorithmen von TEX kennenlernen werden, ist diese Vorlesung kein TEX- oder LATEX-Kurs im engeren Sinne.
- Es steht nicht die Erstellung von Texten im Vordergrund, sondern die Methoden der Informatik, um typografische Probleme bei der Erstellung von Texten lösen zu können.
- Ein Nebeneffekt der Vorlesung kann es natürlich sein, dass das Verständnis für Typografie soweit erweitert wird, dass dies profitabel für die spätere eigene Erstellung von Texten ist.

Syllabus: Warnung

- Ich bin kein TEX-Guru und auch kein Meister der Typografie.
- Entsprechend liefert die Vorlesung dazu nur Einführungen und Einstiegspunkte.
- Deswegen wird die Vorlesung auch mehr aus der Sicht der Informatik betrieben werden. Schwerpunkte liegen daher eher bei Algorithmen, Programmiersprachen und dem Zusammenspiel aller beteiligten Komponenten.

- Grundkenntnisse in Informatik. Insbesondere sollte keine Scheu davor bestehen, etwas zu programmieren.
- Da wir in einigen Übungsblättern Java und C bzw. C++ einsetzen werden, sind Kenntnisse hierin von Vorteil.
- Freude daran, etwas auch an einem Rechner auszuprobieren und genügend Ausdauer, dass nicht beim ersten Fehlversuch aufgegeben wird.

- Jede Woche gibt es zwei Vorlesungsstunden an jedem Montag von 16–18 Uhr in der Helmholtzstraße 22, Raum E.03.
- Die Übungen sind am Dienstag von 10–12 Uhr in der Helmholtzstraße 18 im Pool-Raum E.44 direkt neben der Bibliothek.
- Der Vorlesungstermin trifft auf einen Feiertag am 21. Mai (Pfingstmontag). In dieser Woche wird zum folgenden Übungstermin am 22. Mai eine Vorlesung in der Helmholtzstraße 22, Raum E.03 stattfinden.
- Webseite: `https://www.uni-ulm.de/mawi/mawi-numerik/lehre/sommersemester-2018/vorlesung-digitale-typografie/`

Syllabus: Übungen

- Wir haben keinen Übungsleiter, keine Tutoren und auch keine Korrekteure.
- Die Übungen finden teilweise mit traditionellen Übungsblättern, teilweise auch mit schrittweisen Sitzungen statt, mit denen der Stoff unmittelbar am Rechner erarbeitet werden kann.
- Die Übungstermine finden in einem Pool-Raum statt, so dass bei den rechner-basierten Übungen sofort mit der Lösung begonnen werden kann.
- Ich stehe während der gesamten Übungszeit für Fragen und Hilfestellungen zur Verfügung.
- Lösungen zu Übungsaufgaben werden auf unseren Rechnern mit einem speziellen Werkzeug eingereicht. Details zu dem Verfahren werden zusammen mit der ersten Übungsaufgabe vorgestellt.
- Melden Sie sich bitte bei SLC an, damit Sie an den Übungen teilnehmen können.

- Es gibt keine Vorleistung und die Prüfungen finden mündlich statt.
- Abgesehen von meinen Urlaubszeiten bin ich jederzeit bereit zu prüfen. Nennen Sie eine Woche, ich gebe Ihnen dann gerne einen Termin innerhalb dieser Woche.
- Ich werde in der vorlesungsfreien Zeit aber für vier Wochen nicht da sein (Urlaub).

- Die Vorlesungsfolien und einige zusätzliche Materialien werden auf der Webseite der Vorlesung zur Verfügung gestellt werden.
- Verschiedene Dokumente wie beispielsweise über METAPOST oder PostScript stehen frei zum Herunterladen zur Verfügung.

- Donald E. Knuth, *Digital Typography*, ISBN 1-57586-010-4
- Yannis Haralambous, *Fonts & Encodings*, ISBN 0-596-10242-9

Einige etwas ältere Bücher:

- Richard Rubinstein, *Digital Typography*, ISBN 0-201-17633-5
- Peter Karow, *Digitale Schriften*, ISBN 3-540-54917-X
- Peter Karow, *Schrifttechnologie*, ISBN 3-540-54918-8

- Donald E. Knuth, *The TeXbook*, ISBN 0-201-13448-9
- Donald E. Knuth, *The METAFONTbook*, ISBN 0-201-13445-4
- John D. Hobby, *A User's Manual for MetaPost*, `http://www.tug.org/tutorials/mp/mpman.pdf`
- Adobe Systems Inc., *PostScript Language Reference Manual*, ISBN 0-201-18127-4
- Stephen F. Roth, *Real World PostScript*, ISBN 0-201-06663-7

- Adrian Frutiger, *Type Sign Symbol*, ISBN 3-937715-63-0
- Robert Bringhurst, *The Elements of Typographic Style*, ISBN 0-88179-132-6
- Hans Peter Willberg und Friedrich Forssman, *Lesetypografie*, ISBN 3-87439-652-5
- Friedrich Forssman und Ralf de Jong, *Detailtypografie*, ISBN 3-87439-642-8

- Sie sind eingeladen, mich jederzeit per E-Mail zu kontaktieren:
  E-Mail: andreas.borchert@uni-ulm.de
- Meine reguläre Sprechstunde ist am Mittwoch 10:00–11:30 Uhr. Zu finden bin ich in der Helmholtzstraße 20, Zimmer 1.23.
- Zu anderen Zeiten können Sie auch gerne vorbeischauen, aber es ist dann nicht immer garantiert, daß ich Zeit habe. Gegebenenfalls lohnt sich vorher ein Telefonanruf: 23572.

Syllabus: Nicht verzweifeln!

- Bevor Sie bei der Lösung einer Übungsaufgabe völlig verzweifeln, sollten Sie mir Ihren aktuellen Stand per E-Mail zukommen lassen. Dann werde ich versuchen, Ihnen zu helfen.
- Das kann auch am Wochenende funktionieren.

Syllabus: Feedback

- Feedback ist ausdrücklich erwünscht.
- Auch wenn die Vorlesung früher bereits gehalten worden ist, werden immer noch Anpassungen und Erweiterungen vorgenommen. Das bedeutet auch, dass ich auf Ihre Anregungen eingehen kann und auch Punkte mehr berücksichtigen kann, die Ihnen wichtig sind.
- Lassen Sie bitte Gnade walten, wenn Sie all meine typografischen Ratschläge in Bezug setzen zu meinen eigenen Machwerken. Beherzigen Sie deswegen bitte meine Warnung.

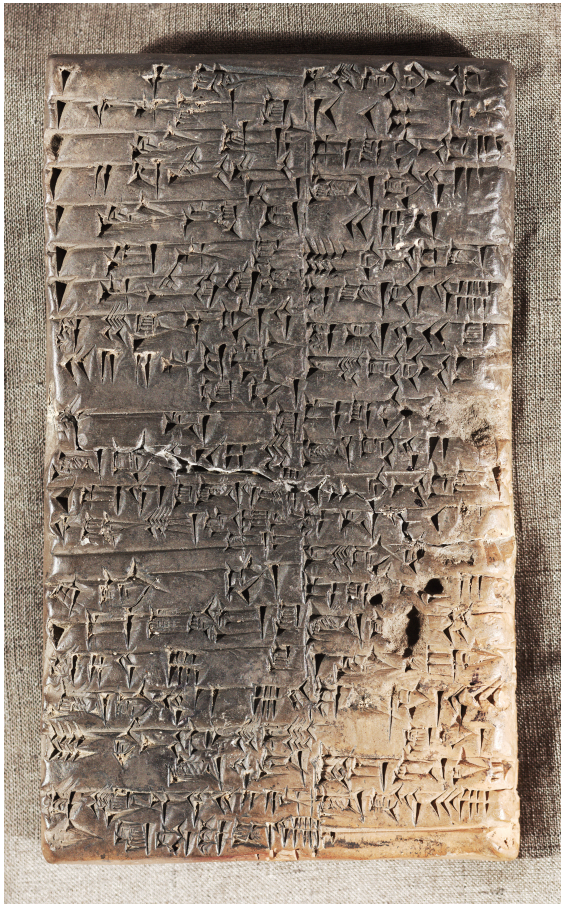
Ludwig Wittgenstein (1889–1951):

> *Eine Schrift kann man auffassen als eine Sprache zur Beschreibung von Lautbildern.*

Adrian Frutiger (*1928):

> *In jedem Zeitalter hat das bearbeitete Material (Stein, Tonerde, Pergament, Papier) auch der Schrift Rhythmus und Form verliehen. Jede Schrift trägt auch das Wesentliche ihrer Zeit in sich.*

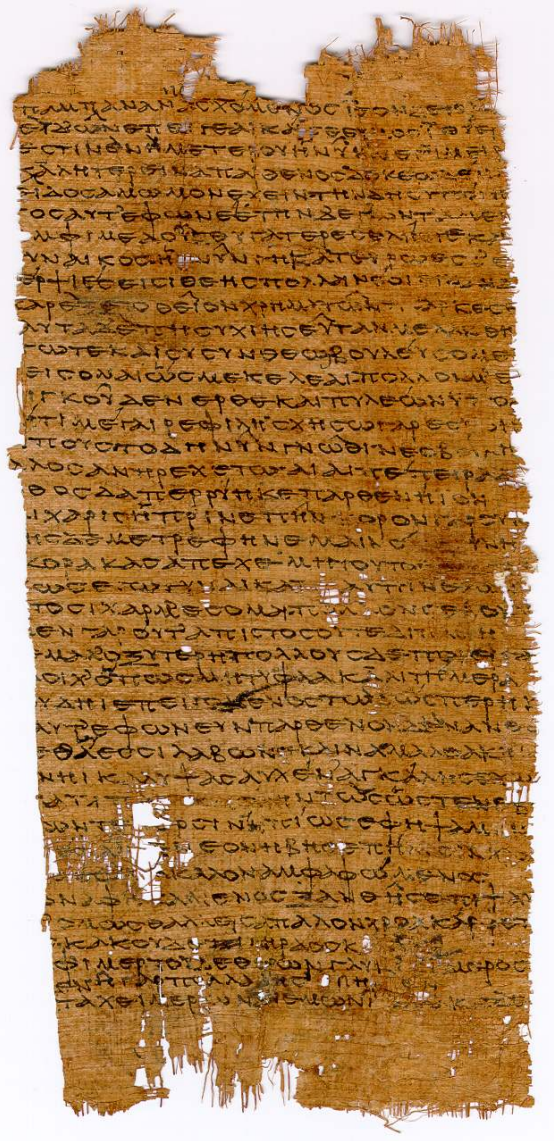
Sumerische Keilschrift



Übernommen von hdl.loc.gov

Die sumerische Keilschrift (wie hier von 2.039 v.Chr.) ist geprägt durch den noch beim Schreiben weichen Ton und der Verwendung eines Schreibgriffels. Der Ton wurde danach getrocknet oder gebrannt. Die Technik führte zur zunehmenden Vereinfachung von Schriftzeichen und so zur Bildung der lautbezogenen Alphabetschriften.

Griechische Schrift

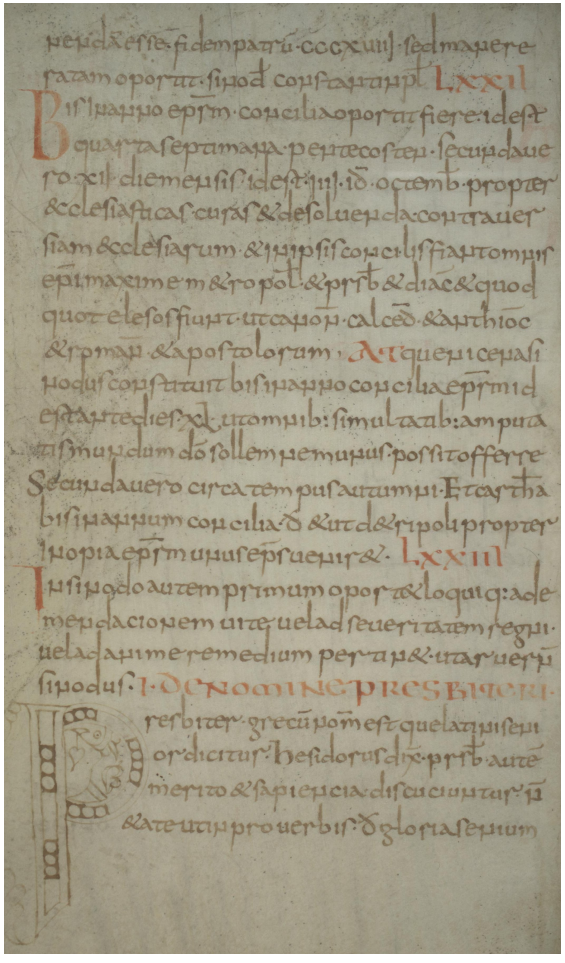


Von Ägypten ausgehend verbreitete sich die Verwendung von Papyrus nach Griechenland und in das römische Reich. Geschrieben wurde mit Tusche unter Verwendung eines Pinsels oder einer gespaltenen Rohrfeder.
Die dargestellte Handschrift ist aus der Sammlung der Universität Köln und zeigt den Text eines Gedichts des griechischen Dichters Archilochos.

Fotografie von Giovanni Dall'Orto

Das Einmeißeln der Schrift in Stein regte die Entwicklung durchgestalteter Schriften mit Serifen an. Diese Entwicklung begann in Griechenland und wurde von Rom indirekt durch die Vermittlung der Etrusker übernommen.
Das dargestellte Fragment ist ein antiker römischer Grabstein an einem Gebäude an der Piazza San Lorenzo in Lucina in Rom.

Im 3. Jahrhundert entstand die jüngere römische Kursive, die als Vorstufe unserer handgeschriebenen Kleinbuchstaben gesehen werden kann. Im 4. und 5. Jahrhundert entstand daraus die Unziale, die diese Formen übernimmt, sie jedoch kalligraphisch ausführt. In Irland wurde dies insbesondere im 7. Jahrhundert weiter entwickelt zu einer insularen Minuskel, die mit großen Zierbuchstaben kombiniert wurde.
Die abgebildete Handschrift entstammt einer Kopie aus dem 7. Jahrhundert des Psalter des heiligen Columbanus in lateinischer Sprache zusammen mit Anmerkungen in Alt-Irisch.



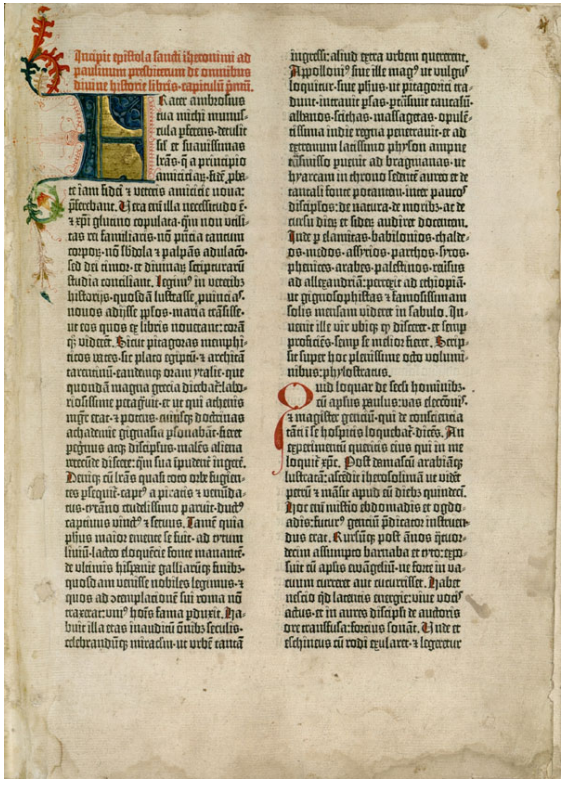
Die Schreibweise aus Irland verbreitete sich zum europäischen Kontinent und regte die Entwicklung der karolingischen Minuskel an. Sie zeichnet sich durch eine hohe Lesbarkeit aus.

Abgebildet ist eine Seite aus der Collectio Canonum Hibernensis aus der zweiten Hälfte 8. Jahrhundert, einer kirchenrechtlichen Schrift aus Irland, die in einem irisch geprägten Kloster in Nordfrankreich geschrieben wurde und dann von dort nach Köln gelangte, wo sie seitdem zur Dombibliothek gehört. Das Bild wurde der Codices Electronici Ecclesiae Coloniensis entnommen.

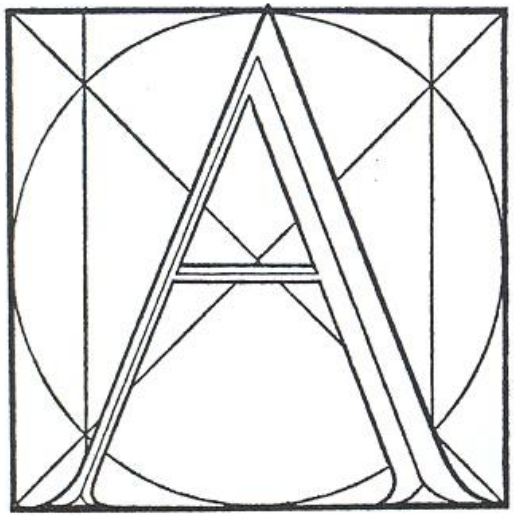
Textur (Gotisch)

Aus der karolingischen Minuskel entwickelte sich die Textur, bei der sämtliche Bögen gebrochen werden.
Abgebildet ist eine Pergamentseite aus einem Evangelium der Abtei Saint-Amand, die etwa 1180-1200 entstanden ist.

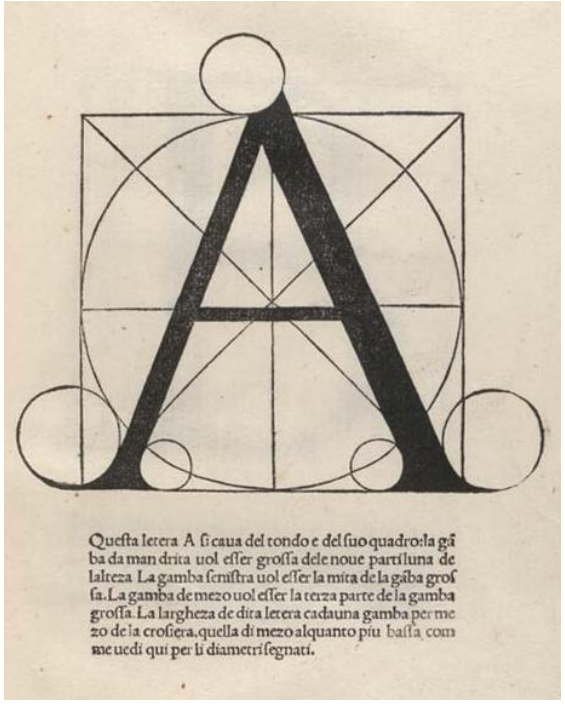
Gutenberg-Bibel



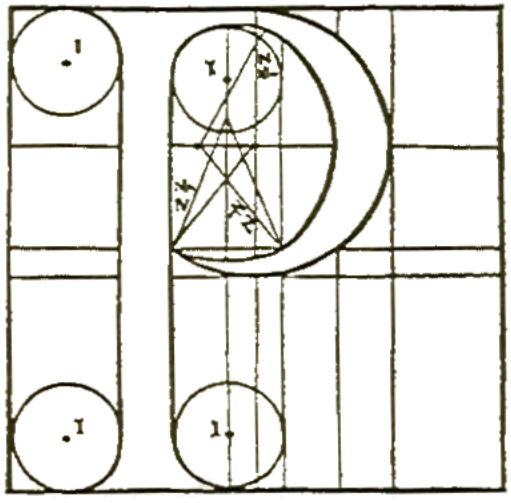
Johannes Gutenberg war ein Perfektionist. Ein so makelloser Blocksatz wie hier bei seiner Bibelausgabe von 1455 mit relativ wenigen Zeichen pro Zeile wurde nie wieder erreicht. Gutenberg gelang dies durch den Einsatz von 290 verschiedenen Buchstaben und Zeichen im Setzkasten.



In der Renaissance suchte Felice Feliciano systematisch nach alten römischen in Stein gemeißelten Inschriften und stellte sein so gewonnenes Alphabet 1460 in einer Handschrift zusammen, die bis heute zur Vatikanischen Bibliothek gehört.
Die Mittellinien in seinem Alphabet deuten das vom Steinmetz geschaffene Profil an.

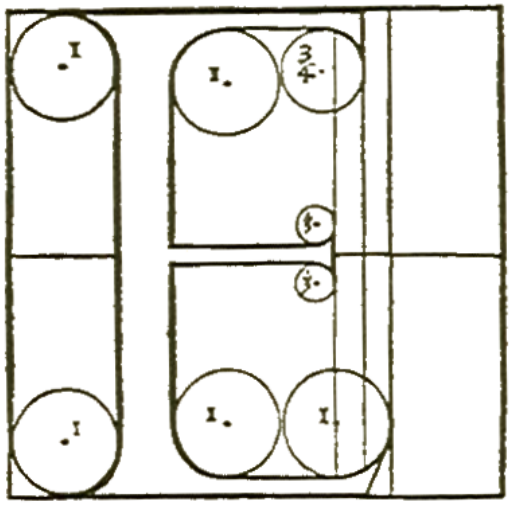


Luca Pacioli (1445–1514) war ein Mathematiker und Wirtschaftswissenschaftler, der in seinem 1509 veröffentlichtem Werk *Divina Proportione* einen ersten Versuch unternahm, das gesamte Alphabet nur mit Hilfe von Lineal und Zirkel darzustellen.



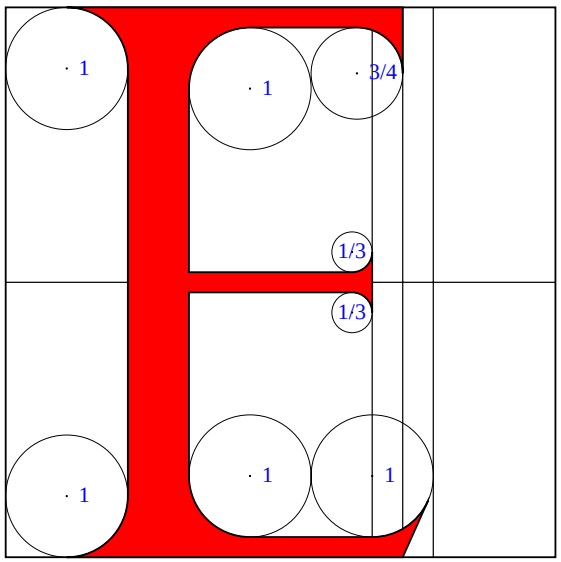
Francesco Torniello (1490–1589) ging einen Schritt weiter und lieferte vollständige geometrische Spezifikationen für alle Versalien. Er kann als Begründer der mathematischen Typografie angesehen werden.

- Torniello definierte als erstes ein Koordinatensystem für die geometrische Spezifikation seiner Versalien.
- Das von ihm verwendete 18×18-Raster diente auch gleichzeitig zur Einführung von zwei Maßeinheiten: Der Einheit (eine Seitenlänge eines Rasterfeldes) und dem Punkt (zwei Einheiten).



Der Buchstabe E wird aus dem Quadrat geformt. Sein Schaft ist innerhalb des Quadrats zwei Punkte von der linken Vertikalen entfernt und erstreckt sich über die gesamte Höhe des Quadrats unter Berücksichtigung der Kreise, die Du links siehst. Der obere und auch der untere Querbalken sollten ein Drittel Punkt dick sein und dreieinhalb Punkte lang sein mit den Kreisen, die Du siehst. Der mittlere Querbalken soll von der gleichen Dicke sein, jedoch nur drei Punkte lang sein mit Kreisen von einem Radius von einem Drittel Punkt, wie es hier gezeigt ist.

Torniellos E

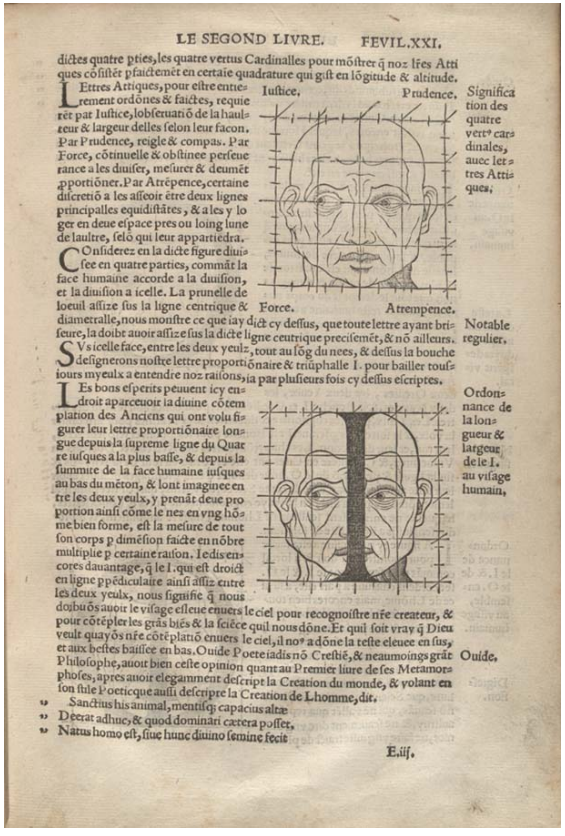


Dieses E wurde entsprechend der Spezifikation von Francesco Torniello mit PostScript reproduziert.

E.eps

```
/E {
   newpath
   100 0 moveto
   650 0 lineto
   691.2630473 92.12144648 lineto
   600 133 100 335.8714332 270 arcn
   400 33 lineto
   400 133 100 270 180 arcn
   300 433.5 lineto
   567 400.5 33 90 0 arcn
   567 499.5 33 0 270 arcn
   300 466.5 lineto
   400 767 100 180 90 arcn
   575 867 lineto
   575 792 75 90 0 arcn
   650 900 lineto
   100 900 lineto
   100 800 100 90 0 arcn
   200 100 lineto
   100 100 100 0 270 arcn
   closepath
} def
```

Albrecht Dürer (1471–1528) kannte sehr gut die italienischen Renaissance-Künstler einschließlich Pacioli und Torniello und veröffentlichte 1525 seine Entwürfe im Rahmen seines Werks *Unterweysung der Messung*.



LE SEGOND LIVRE. FEVIL.XXI.

dictes quatre pties, les quatre vertus Cardinalles pour mõstrer q̃ noz lres Attiques cõsistẽt pfaictemẽt en certaine quadrature qui gist en lõgitude & altitude.

Iustice. Prudence.

Significa tion des quatre vertꝰ cardinales, auec lettres Attiques.

LEttres Attiques, pour estre entierement ordõnes & faictes, requierẽt par Iustice, lobseruatiõ de la hauleur & largeur delles selon leur facon. Par Prudence, reigle & compas. Par Force, cõtinuelle & obstinee perseuerance a les diuiser, mesurer & deumẽt ꝓportiõner. Par Attrẽpence, certaine discretiõ a les asseoir estre deux lignes principalles equidistãtes, & a les y loger en deue espace pres ou loing lune de laultre, selõ qui leur appartiedra.

COnsiderez en la dicte figure diuisee en quatre parties, commãt la face humaine accorde a la diuision, et la diuision a icelle. La prunelle de loeuil assize sus la ligne centrique & diametralle, nous monstre ce que iay dict cy dessus, que toute lettre ayant briseure, la doibt auoir assize sus la dicte ligne centrique precisemẽt, & nõ ailleurs.

Force. Atrempence.

Notable regulier.

SVs icelle face, entre les deux yeulz, tout au lõg du nees, & dessus la bouche desingnerons nostre lettre proportiõnaire & triuphalle I. pour bailler touſiours myeulx a entendre noz raisons, ia par plusieurs fois cy dessus escriptes.

Ordonnance de la longueur & largeur de le I. au visage humain.

LEs bons esperits peuuent icy endroit aparceuoir la diuine cõtemplation des Anciens qui ont volu figurer leur lettre proportiõnaire longue depuis la supreme ligne du Quatre iusques a la plus basse, & depuis la summite de la face humaine iusques au bas du mẽton, & lont imaginee entre les deux yeulx, y prenãt deue proportion ainsi cõme le nez en vng hõme bien forme, est la mesure de tout son corps p dimẽsion faicte en nõbre multiplie p certaine raison. Ie dis encores dauantage, q̃ le I. qui est droict en ligne ꝑpẽdiculaire ainsi assiz entre les deux yeulx, nous signifie q̃ nous doibuõs auoir le visage esleue enuers le ciel pour recognoistre nre createur, & pour cõtẽpler les grãs biẽs & la science quil nous dõne. Et quil soit vray q̃ Dieu veult quayõs nre cõtẽplatiõ enuers le ciel, il nꝰ a dõne la teste eleuee en sus, et aux bestes baissee en bas. Ouide Poeteiadis nõ Crestiẽ, & neaumoings grãt Philosophe, auoit bien ceste opinion quant au Premier liure de ses Metamorphoses, apres auoir elegamment descript la Creation du monde, & volant en son stile Poeticque aussi descripre la Creation de Lhomme, dit.

Ouide.

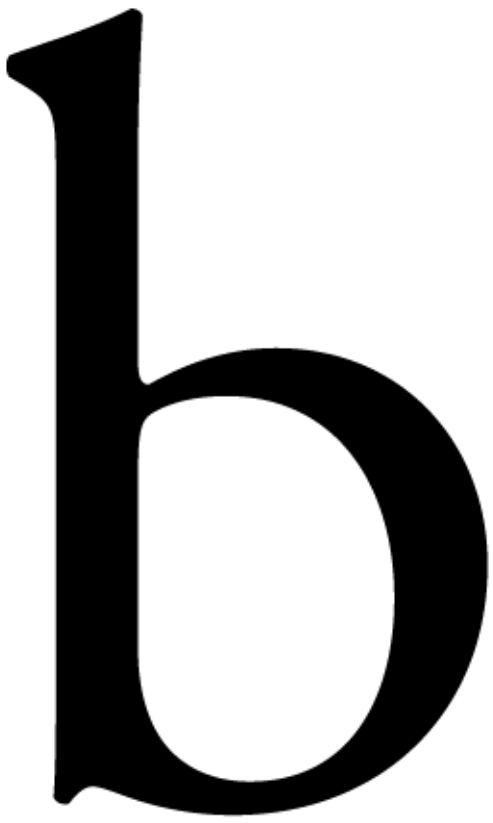
» Sanctius his animal, mentisq; capacius altæ
» Deerat adhuc, & quod dominari cætera posset,
» Natus homo est, siue hunc diuino semine fecit

E.iij.

Geoffroy Tory kannte aus seinen Studien in Italien die Werke von Pacioli und Torniello und übernahm dabei auch Torniellos Raster, das auf 10×10 dimensioniert wurde.
Mit seinem 1529 in Paris veröffentlichtem Werk *Champ Fleury* legte er den Grundstein für die weitere typografische Entwicklung in Frankreich.

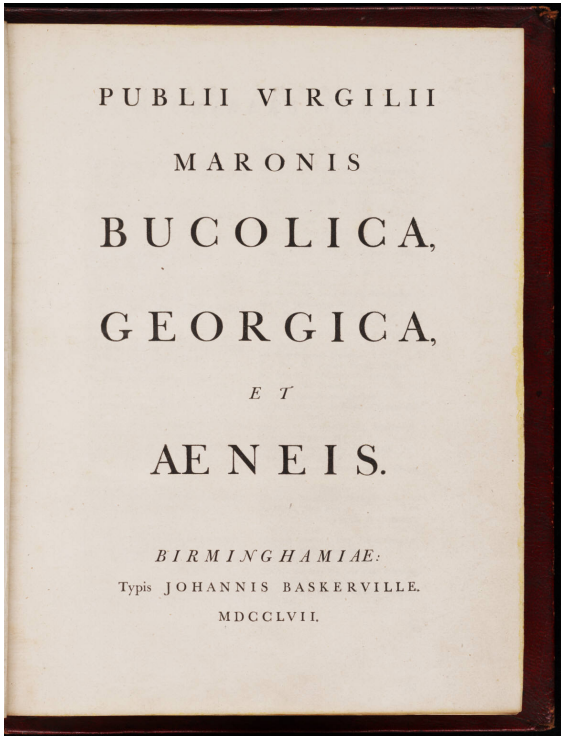
Francesco Cresci studierte die römischen Inschriften erneut. Beispielsweise basiert das nebenstehende B auf der entsprechenden Form in der Inschrift am Piedestal der Trajanssäule in Rom. Cresci schreibt 1560:

> *Ich bin zu dem Schluss gekommen, dass Euklid, der Prinz der Geometrie, wenn er zu unserer Welt zurückkehren würde, er niemals darauf kommen würde, dass die Kurven der Buchstaben mit Hilfe von Zirkeln gezogenen Kreisen so konstruiert werden könnten, dass sie den Proportionen und dem Stil der antiken Buchstaben entsprechen.*

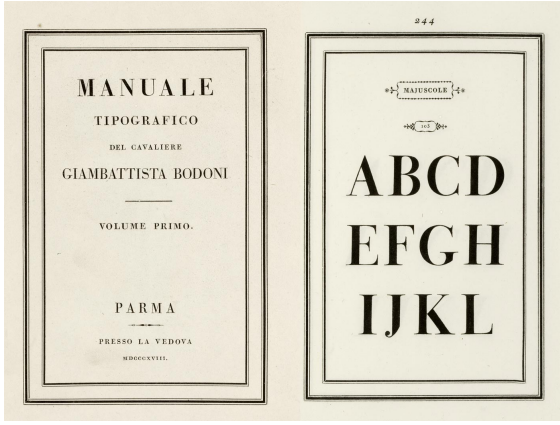


Claude Garamond (1480–1561) war ein Pariser Schriftgießer, der zuvor bei Geoffroy Tory in die Schule gegangen war. Auch er verabschiedete sich von dem Zirkel und schuf 1531 eine nach ihm benannte Schrift, die bis heute populär geblieben ist.

John Baskerville


PUBLII VIRGILII
MARONIS
BUCOLICA,
GEORGICA,
ET
AENEIS.
BIRMINGHAMIAE:
Typis JOHANNIS BASKERVILLE.
MDCCLVII.

John Baskerville (1706–1775) war ein englischer Unternehmer, der insbesondere durch seinen Schriftschnitt bekannt wurde, der nach ihm benannt wurde. Seine Schrift gehört zur Familie der Barock-Antiqua, die einer Übergangsperiode zwischen den Schriften der Renaissance und der klassizistisch geprägten Schriften zuzurechnen ist, die sich durch größere Unterschiede in den Strichstärken auszeichnet und bei denen ein Einfluss der Kupferstecher-Schriften zu erkennen ist.



Giambattista Bodoni (1740–1813) war ein italienischer Buchdrucker und Herausgeber, dessen nach ihm benannten Schriftschnitte die Familie der klassizistischen Antiqua mit ihrer strengen Eleganz begründete. Den hohen Ansprüchen der Maschinenschriften des 20. Jahrhunderts waren sie nicht ohne weiteres gewachsen, so dass sie erst seit einiger Zeit vermehrt zurückkehren. Ein modernes Beispiel einer klassizistischen Antiqua ist die von Donald E. Knuth entwickelte Schrift *Computer Modern Roman*.

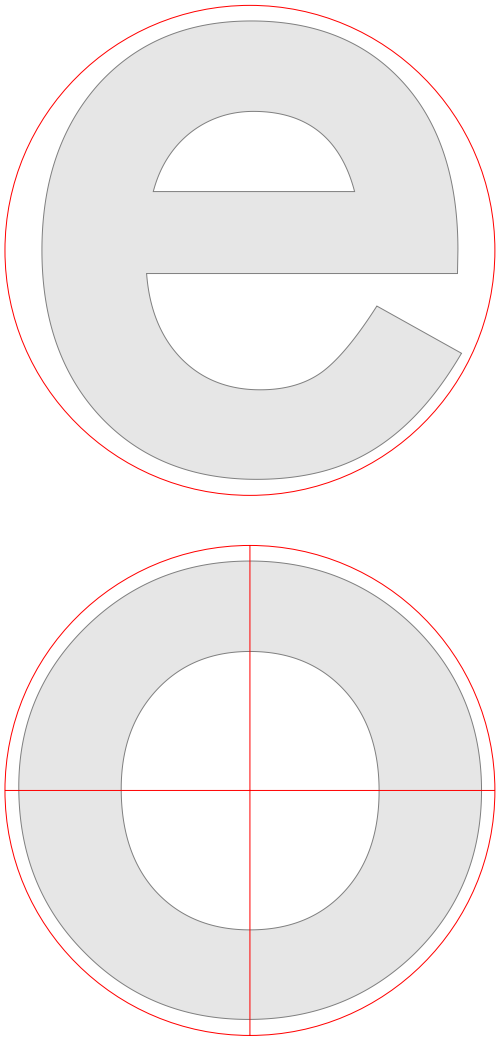
Die Akzidenz-Grotesk war die erste populäre serifenlose Schrift, die eine größere Verbreitung erfuhr, nachdem sie von der Berthold AG seit 1898 angeboten wurde. Sie hat mehrere Vorgänger — die genauen Ursprünge sind aber bis heute noch nicht umfassend aufgeklärt worden. Ursprünglich diente sie als Schrift für Anzeigen und Drucksachen und ist die Grundlage vieler Nachfolger wie etwa die Univers von Frutiger oder die Helvetica von Max Miedinger. Abgebildet ist der 2007 erschienene CD-Cover für *Melancholie* von Bernhard Fleischmann, der von Jan Kruse / Human Empire gestaltet wurde.

Futura

Muster der Futura Medium
(dailytypesspecimen.com)

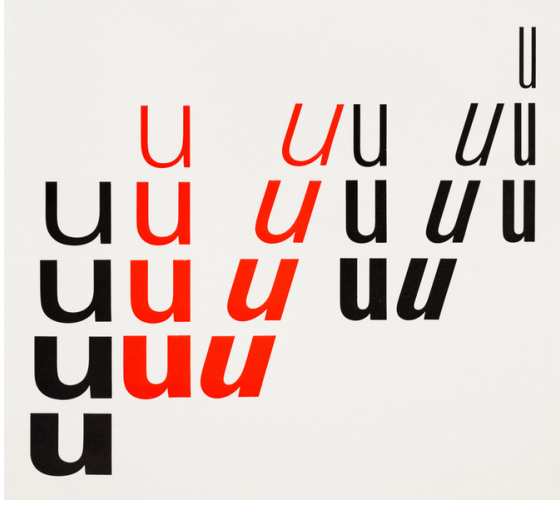
In den 1920er-Jahren wurde insbesondere durch das Bauhaus der Funktionalismus populär. Die Forderung nach Klarheit und Einfachheit wurde Rechnung getragen durch die Rückkehr zu einfachen geometrischen Grundformen. Nach weniger erfolgreichen Vorgängerschriften von Herbert Bayer und Ferdinand Kramer entwickelte Paul Renner 1927 die Futura mit sehr gleichmäßigen Strichstärken und fast kreisförmigen Rundungen, die zu einer der populärsten Schriftschnitten des 20. Jahrhunderts wurde.

Die Futura wurde zum erfolgreichsten Aushängeschild der von Jan Tschichold ausgerufenen „Neuen Typografie“.



Die Futura wird als geometrische Grotesk charakterisiert. Im *Wegweiser Schrift* beschreibt sie Hans Peter Willberg wie folgt:

> *Die kreisrunden Buchstaben der konstruierten Groteskschriften treffen auf ihre Nachbarn im Wort wie Billardkugeln, sie stoßen einander ab. Manche Buchstaben sind sehr ähnlich, sie müssen dem Programm folgen – mehr Roboter als Individualisten.*



Schematische Anordnung der ursprünglich 21 Schriften der *Univers* von Bruno Pfäffli

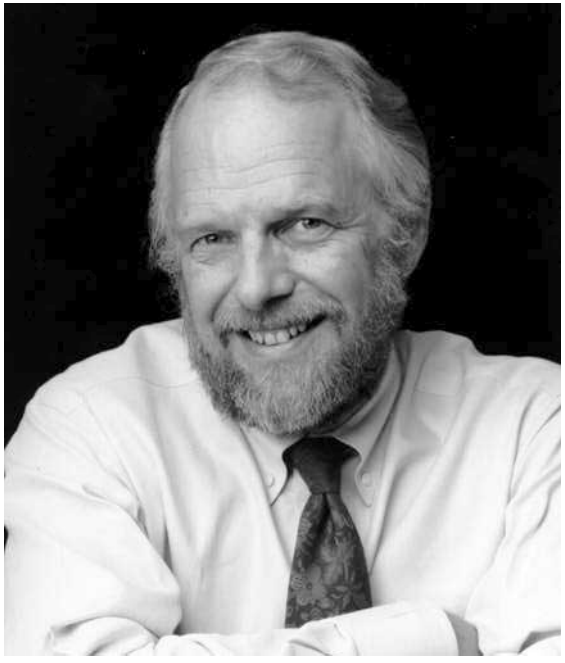
Die von Adrian Frutiger 1953 bis 1957 entwickelte *Univers* ist der erste Versuch, eine Großfamilie zu entwerfen. Sie wendet sich deutlich von den streng geometrischen Formen der Futura ab. Adrian Frutiger begründet dies 1961 wie folgt:

> *Eine rein geometrische Schriftform ist auf die Dauer nicht haltbar. Das Auge sieht horizontale Striche dicker als vertikale, der perfekte Zirkelkreis als O scheint unförmig und sticht im gesetzten Wort heraus. Der moderne Betonbau ist aber nicht unbedingt geometrisch; die Formen sind gespannt, lebendig. Die Schrift muss es auch sein.*

Rotis: 17 einzelne Schriften, vier Familien, eine Schriftsippe (Quelle)

Die von Otl Aicher 1989 herausgegebene Rotis ist eine der ersten sogenannten Schriftsippen (ein Begriff von Hans Peter Willberg), die eine Serif, Semi-Serif, Semi-Sans-Serif und eine Sans-Serif-Schriftfamilie miteinander in abgestimmter Weise vereinen. Vorläufer dieses Ansatzes waren die Schriften Demos (eine Antiqua) und Praxis (eine Grotesk) von Gerard Unger in den Jahren 1976/77.

Der Ulmer Otl Aicher (1922–1991) war ein bedeutender Gestalter des 20. Jahrhunderts, der zusammen mit Max Bill die Ulmer Hochschule für Gestaltung gründete. Bekannt sind seine Piktogramme der Olympischen Spiele von München.



John Warnock 2008, Fotografie von Marvalous, CC-BY-SA-3.0

- 1982 wurde die kalifornische Firma Adobe Systems von John Warnock und Charles Geschke gegründet.
- Beide waren zuvor bei XEROX PARC beschäftigt und dort involviert bei der Entwicklung einer Seitenbeschreibungssprache, die (wie so viele weitere wegweisende Projekte bei PARC) nicht den Weg zur erfolgreichen Vermarktung fand.
- 1985 publizierte Adobe die Programmiersprache PostScript und lizenzierte eine erste Implementierung an Apple, die diese für ihren Apple LaserWriter einsetzte.

- METAFONT ist älter als PostScript (erste Version von 1979, die wesentlich überarbeitete und bis heute gültige Fassung ist von 1984).
- Dennoch sind wohl METAFONT und PostScript weitgehend unabhängig voneinander entwickelt worden, da die Anfänge von PostScript bis in die 70er-Jahre zurückreichen.
- Im Vergleich ist PostScript allgemeiner und mächtiger, aber METAFONT ist in einigen Bereichen (Definition von Kurven, Verwendung von Gleichungssystemen) sehr viel eleganter.

- Anfang der 80er-Jahre kam das Konzept einer Programmiersprache für die Gestaltung einer Seite für viele überraschend.
- Warum soll ausgerechnet der Drucker eine Programmiersprache interpretieren können?
- Die Motivation lag in den vielen Nachteilen der traditionellen Lösungen.

- Schriftssätze in vorgegebenen Größen waren typischerweise in den Drucker integriert oder konnten separat heruntergeladen werden.
- Befehle an den Drucker sahen dann nur die Positionierung vor (eventuell eingeschränkt), die Auswahl eines Schriftssatzes, die Ausgabe von Text und möglicherweise einen Grafik-Modus, bei dem Bitmaps heruntergeladen werden.
- Vorteile: Einfache und schnelle Lösung, kostengünstig.

- Die Portabilität war nicht gegeben, da die Schriftsätze, der Grafik-Modus und die Farbpalette vom Druckermodell abhingen.
- Das Laden von Grafiken in Form von Bitmaps über serielle Leitungen benötigte viel Zeit bei höheren Auflösungen.
- Eine Voransicht auf dem Bildschirm wurde typischerweise nicht unterstützt.

- PostScript wurde zum ersten weit akzeptierten Standard einer Seitenbeschreibungssprache.
- Ein PostScript-Dokument kann auf dem Bildschirm, auf einem Billig-Drucker und einer teuren Offsetdruck-Anlage ausgedruckt werden und im Rahmen der Möglichkeiten sieht sie überall gleich aus.
- PostScript vereinfacht die Herstellung von Grafiken.
- Der Umfang eines Dokuments ist deutlich geringer, wenn Grafiken und besondere Effekte integriert sind.

- Die Drucker mit PostScript waren deutlich teurer, da sie eine leistungsfähigere CPU und mehr Hauptspeicher benötigen.
- Der Zeitaufwand zur Berechnung einer Seite ist nicht nach oben beschränkt.
- Was passiert, wenn bei der Ausführung Fehler auftreten?
- Die Programmiersprache wurde optimiert für eine möglichst einfache Implementierung und nicht in Bezug auf die Freundlichkeit für Programmierer.

- Die Programmiersprache PostScript leitet sich primär von FORTH und Lisp ab.
- Alle Operatoren bzw. Funktionen finden ihre Operanden auf einem Stack und liefern dort ihre Ergebnisse wieder ab.
- Arrays und assoziierte Arrays (*dictionary* genannt) werden unterstützt.
- Programme sind Daten.
- Hinzu kommen Datentypen und Operationen speziell für die Seitengestaltung.

- Zum Experimentieren mit PostScript empfiehlt sich die interaktive Verwendung eines PostScript-Interpreters.
- GhostScript ist eine freie Implementierung von PostScript. Der Aufruf erfolgt hier mit dem Kommando *gs*.
- `https://www.ghostscript.com/`
- Wenn Sie nicht den E.44 nutzen, können Sie sich per „ssh -X“ auf die Theon oder Thales anmelden, um dort *gs* aufzurufen. Die Theon bietet die neueste Version. (Die Optionen „-X“ ist relevant, damit das X-Protokoll von der *ssh* getunnelt wird.

```
theon$ gs
GPL Ghostscript 9.23 (2018-03-21)
Copyright (C) 2018 Artifex Software, Inc.  All rights reserved.
This software comes with NO WARRANTY: see the file PUBLIC for details.
GS>3 2 sub
GS<1>==
1
GS>quit
theon$
```

- Mit *gs* wurde der Interpreter gestartet, der sich mit dem Prompt „GS>“ meldet.
- Gleich zu Beginn wird auch ein Fenster für die grafische Ausgabe eröffnet.

- Der gesamte Programmtext wird in eine Sequenz lexikalischer Symbole (Tokens) konvertiert, die sofort nach der Erkennung zur Ausführung gebracht werden.
- Tokens fallen in eine von zwei Klassen: Operanden und Operatoren.
- Operanden werden bei der Ausführung auf den Operandenstack geladen.
- Operatoren finden ihre Operanden auf dem Stack und liefert dort auch ihr Ergebnis ab.

- Neben der Ausgabe der eigentlichen Grafik-Seite gibt es auch ganz normale datei-orientierte Ein- und Ausgabeverbindungen in PostScript.
- Dazu gehört insbesondere die Standard-Ausgabe und die Standard-Fehlerausgabe, wobei beide in vielen Fällen identisch sind.
- Der Operator == nimmt das oberste Element vom Stack und gibt dieses in PostScript-Syntax auf der Standard-Ausgabe aus.

```
theon$ gs
GPL Ghostscript 9.23 (2018-03-21)
Copyright (C) 2018 Artifex Software, Inc.  All rights reserved.
This software comes with NO WARRANTY: see the file PUBLIC for details.
GS>17
GS<1>13
GS<2>12
GS<3>412
GS<4>==
412
GS<3>==
12
GS<2>==
13
GS<1>==
17
GS>quit
theon$
```

- Beim GhostScript-Interpreter verrät der Prompt, wieviele Elemente noch auf dem Operanden-Stack verbleiben.

```
theon$ gs
GPL Ghostscript 9.23 (2018-03-21)
Copyright (C) 2018 Artifex Software, Inc.  All rights reserved.
This software comes with NO WARRANTY: see the file PUBLIC for details.
GS>2 3 mult
Error: /undefined in mult
Operand stack:
   2   3
Execution stack:
   %interp_exit   .runexec2   --nostringval--   --nostringval--   --nostringval--   2   %stopped_push
Dictionary stack:
   --dict:978/1684(ro)(G)--   --dict:0/20(G)--   --dict:78/200(L)--
Current allocation mode is local
Last OS error: Resource temporarily unavailable
Current file position is 9
GS<2>quit
theon$
```

- In diesem Beispiel wurde versehentlich `mult` anstelle von `mul` für den Multiplikations-Operator angegeben.

- Als Zeichensatz wird ASCII verwendet.
- Neben den druckbaren Zeichen sind nur das Null-Byte, der Tabulator, LF, FF, CR und das Leerzeichen zulässig.
- Alle nicht druckbaren Zeichen dienen als Leerzeichen.
- Aufeinanderfolgende Leerzeichen sind äquivalent zu einem Leerzeichen (abgesehen von Leerzeichen innerhalb von Zeichenketten).
- CR, LF oder CR LF zählt als Zeilentrenner.
- Zu den Sonderzeichen gehören `(`, `)`, `<`, `>`, `[`, `]`, `{`, `}`, `/` und `%`.

- Kommentare beginnen mit % und enden mit dem nächsten Zeilentrenner. Sie sind äquivalent zu einem Leerzeichen.
- Zahlen können mit Vorzeichen (+ oder −) angegeben werden. Beispiele:
  `0 123 -14 +234123`
- Es kann auch die Basis der Zahlendarstellung angegeben werden. Beispiele:
  `8#644 16#AFB00 2#11011`
- Gleitkommazahlen und die Exponentialdarstellung sind zulässig. Beispiele:
  `1.2 .123 -3. 1E10 +1.2e-17`

- Zeichenketten können in `(...)` eingeschlossen werden. Beispiel:
  `(Das ist eine Zeichenkette in PostScript)`
- Klammern sind innerhalb der Zeichenkette zulässig, wenn die Klammernpaare balanciert sind. Beispiel:
  `(Dies ist (ein (Klammergebirge)))`
- Eine leere Zeichenkette ist zulässig: `()`
- Zeichenketten dürfen über mehrere Zeilen gehen. Beispiel:
  `(Hier geht die Zeichenkette los,`
  `die einen Zeilentrenner enthaelt)`
- Sonderzeichen sind zulässig. Beispiel: `(%<}>{!)`

Sonderzeichen können innerhalb einer Zeichenkette mit Sequenzen, die mit einem Rückwärtsschrägstrich beginnen, eingebunden werden:

Sequenz	Bedeutung
\n	Zeilentrenner (LF)
\r	CR
\t	Tabulator
\b	Backspace
\f	FF
\\	Rückwärtsschrägstrich
\(	öffnende Klammer
\)	schließende Klammer
ddd	Zeichen in Oktaldarstellung (*d*: [0-7])

- Namen sind alle Sequenzen von Zeichen, die keine Leerzeichen und keine Sonderzeichen enthalten.
- Namen können mit Ziffern beginnen und gelten als Name, solange sie nicht als Zahlenkonstante interpretiert werden können. Entsprechend ist `1E` ein Name, während `1E1` eine Zahl ist.
- Namen können somit ungewöhnlich sein. Beispiele:
  `1+2 $$ @$@xX`
- Klein- und Großschreibung ist signifikant.

- Im Normalfall gilt ein Name als auszuführender Operator.
- Steht jedoch unmittelbar vor dem Namen ein Schrägstrich (ohne trennende Leerzeichen), dann wird der Name **nicht** ausgeführt und er gilt als Symbol (analog zu Lisp). Der Schrägstrich selbst gehört jedoch nicht zu dem Namen. Beispiel:
  `/Hallo`
- Diese Unterscheidung ermöglicht es, Operatoren als ganz normale PostScript-Objekte zu behandeln.

- Ein Array kann mit den Sonderzeichen `[` und `]` konstruiert werden.
  Beispiel:
  `[ 1 2 3 ]`
- Die Elemente eines Arrays dürfen unterschiedliche Typen haben.
  Beispiel:
  `[ 3.14 /Hallo (Hallo!) ]`
- Arrays können verschachtelt werden. Beispiel:
  `[ (Charles) [ (Philip) [ (Andrew) (Alice) ] (Elizabeth II) ] ]`

- Wenn ein Operator innerhalb einer Array-Konstruktion vorkommt, wird er sofort ausgeführt.
- Aus diesem Grunde ist
  `[ 1 2 1 2 add ]`
  äquivalent zu `[ 1 2 3 ]`

- Prozeduren können mit den Sonderzeichen `{` und `}` konstruiert werden. Beispiel:
  `{ dup mul }`
- Prozeduren ähneln den Arrays. Der entscheidende Unterschied ist, dass sie beim Lesen nicht umgehend ausgeführt werden, anschließend aber als ausführbar gelten.
- Grundsätzlich verwaltet der Interpreter für jedes Objekt die Information, ob es ausführbar ist oder nicht.

- Ein assoziatives Array kann mit den Symbolen << und >> konstruiert werden. Dazwischen stehen jeweils Paare aus einem Schlüssel und dem zugehörigen Wert. Beispiel:
  `<< (Vorname) (Hans) (Nachname) (Maier) (Ort) (Ulm) >>`
- Bei den Schlüsseln sind auch Zahlen zulässig. Zeichenketten und Namen sind bei den Schlüsseln zueinander äquivalent. Obiges Array hätte somit auch so angelegt werden können:
  `<< /Vorname (Hans) /Nachname (Maier) /Ort (Ulm) >>`

PostScript (und auch METAFONT) unterstützen beim Spezifizieren von Grafiken das Stencil-Paint-Modell:

- Zunächst wird eine Kurve beschrieben, bestehend aus Linien, Kreisbögen, Bézier-Kurven und Zeichen zur Verfügung stehender Schriften.
- Kurven können offen oder geschlossen sein, sie dürfen sich auch selbst kreuzen.
- Kurven müssen nicht zusammenhängend sein.
- Eine Kurve dient dann als Grundlage für Zeichenoperationen.

Wenn eine Kurve fertig definiert ist, stehen folgende Optionen zur Verfügung:

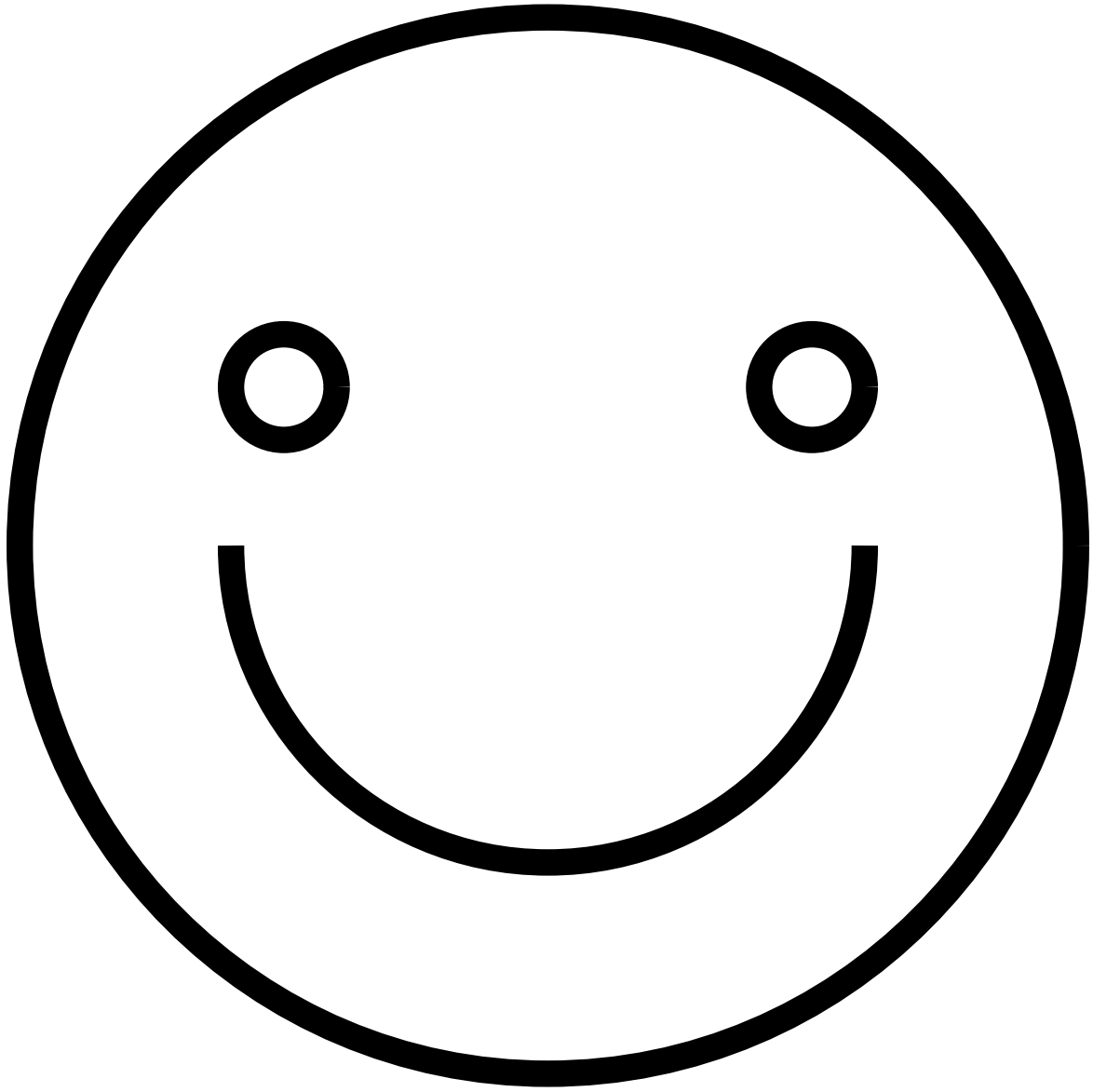
- Die Kurve kann mit einem (ziemlich flexiblen) Stift gezeichnet werden.
- Der Inhalt der Kurve kann mit einer Farbe ausgefüllt werden.
- Die Kurve kann als Schablone (*clipping path*) verwendet werden.

- Für die grafische Ausgabe steht ein Koordinatensystem zur Verfügung, das in Punkten rechnet, wobei ein Punkt typischerweise für $\frac{1}{72}$ Inch steht.
- $(0, 0)$ ist erwartungsgemäß links unten.
- PostScript unterscheidet zwischen dem Koordinatensystem im Programm und dem Koordinatensystem des Ausgabegeräts. Beliebige affine Transformationen (Rotieren, Skalieren, Verschieben) sind möglich.

Eine erste Kurve

smiley.eps

```
%!PS-Adobe-3.0 EPSF-3.0
%%BoundingBox: -5 -5 205 205
newpath                 % eine neue Kurve wird angelegt
100 100 100 0 360 arc % Kreis um (100,100) zeichnen
40 100 moveto           % linke Oberseite des Mundes
100 100 60 180 0 arc  % Mund zeichnen
60 130 moveto           % zum linken Auge
50 130 10 0 360 arc   % linkes Auge
160 130 moveto          % zum rechten Auge
150 130 10 0 360 arc  % rechtes Auge

5 setlinewidth          % Liniendicke definieren
stroke                  % Zeichnen
```



smiley.eps

```
%!PS-Adobe-3.0 EPSF-3.0
%%BoundingBox: -5 -5 205 205
```

- Diese Kommentare sind nicht notwendig, um das Beispiel mit `gs smiley.eps` auszuführen.
- Ohne diese Kommentare klappt es jedoch nicht mit *gv* oder dem Einbetten als Grafik in ein Textsystem (wie beispielsweise LaTeX).

smiley.eps

```
newpath                 % eine neue Kurve wird angelegt
100 100 100 0 360 arc % Kreis um (100,100) zeichnen
```

- Mit `newpath` beginnt eine neue Kurve (und wirft zuvor die alte Kurve weg). Da zuvor noch keine existierte, hätte das hier auch wegfallen können.
- `arc` erwartet 5 Parameter: Koordinaten des Mittelpunkts, Radius, Winkel, an dem der Kreisbogen beginnt, und der Winkel, an dem der Kreisbogen endet.
- Winkel werden in Grad angegeben. 0 Grad bedeutet genau rechts vom Mittelpunkt, dann geht es im mathematisch positiven Sinne weiter.

smiley.eps

```
40 100 moveto          % linke Oberseite des Mundes
100 100 60 180 0 arc   % Mund zeichnen
```

- Ein Kreisbogen beginnt implizit mit einer Linie ausgehend vom aktuellen Punkt.
- Um eine unerwünschte Linie zwischen dem Gesicht und dem Mund zu vermeiden, ist eine Positionierung mit `moveto` sinnvoll.
- Alternativ hätten wir auch mit mehreren getrennten Pfaden operieren können.

smiley.eps

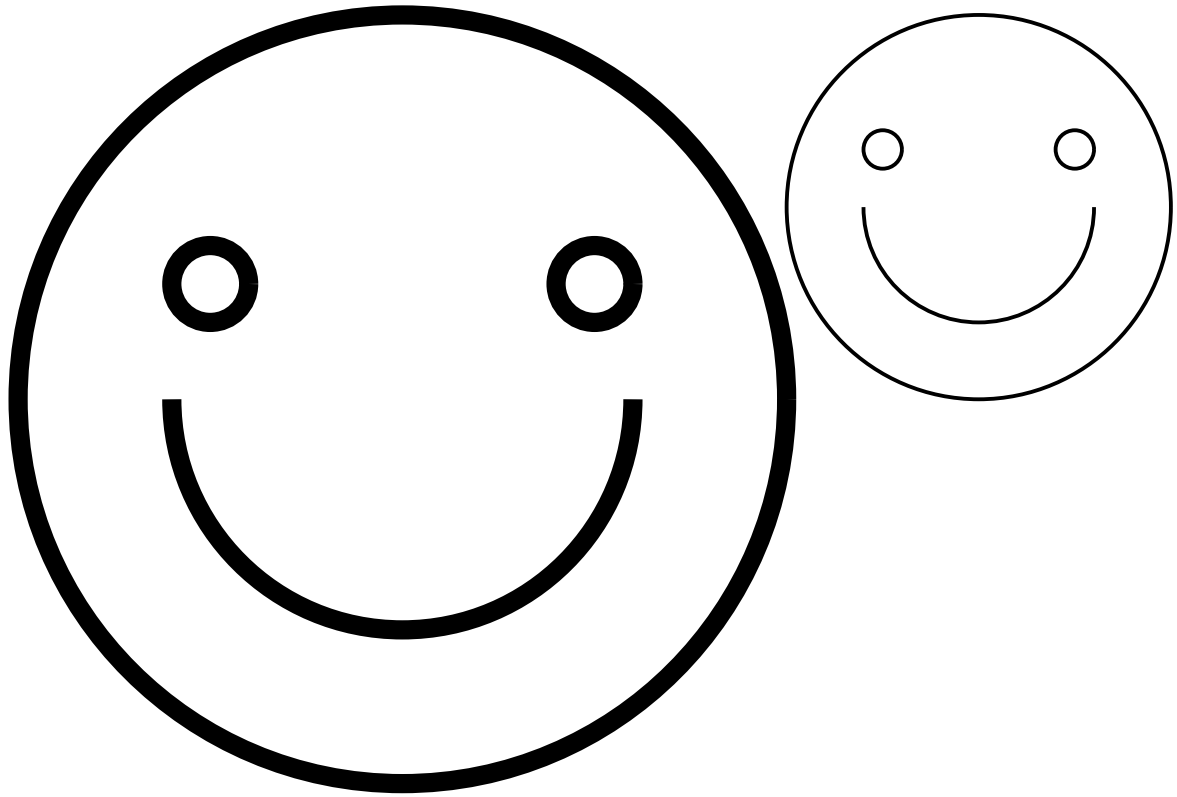
```
5 setlinewidth          % Liniendicke definieren
stroke                  % Zeichnen
```

- Zu den vielen Parametern, die den Zeichenstift definieren, gehört auch die Dicke des Stifts, die mit `setlinewidth` verändert werden kann.
- Mit `stroke` geht der Zeichenstift entlang der aktuellen Kurve in Aktion.

Zwei Smileys

smiley2.eps

```
%!PS-Adobe-3.0 EPSF-3.0
%%BoundingBox: -5 -5 305 205
/Smiley {
   newpath                % eine neue Kurve wird angelegt
   100 100 100 0 360 arc % Kreis um (100,100) zeichnen
   40 100 moveto          % linke Oberseite des Mundes
   100 100 60 180 0 arc   % Mund zeichnen
   60 130 moveto          % zum linken Auge
   50 130 10 0 360 arc    % linkes Auge
   160 130 moveto         % zum rechten Auge
   150 130 10 0 360 arc   % rechtes Auge
} def

gsave Smiley 5 setlinewidth stroke grestore
gsave 200 100 translate 0.5 0.5 scale
Smiley 2 setlinewidth stroke grestore
```



smiley2.eps

```
/Smiley {
   newpath                % eine neue Kurve wird angelegt
   100 100 100 0 360 arc  % Kreis um (100,100) zeichnen
   40 100 moveto          % linke Oberseite des Mundes
   100 100 60 180 0 arc   % Mund zeichnen
   60 130 moveto          % zum linken Auge
   50 130 10 0 360 arc    % linkes Auge
   160 130 moveto         % zum rechten Auge
   150 130 10 0 360 arc   % rechtes Auge
} def
```

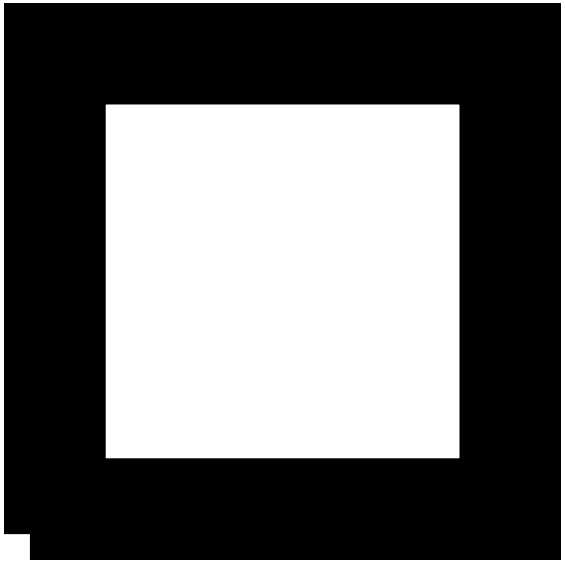
- Nach `/Smiley { ... }` liegen ein Name und ein ausführbares Array auf dem Stack.
- `def` holt sich einen Namen und ein Objekt und trägt das Paar in dem assoziativen Array (*dictionary*) ein, das ganz oben beim Stack der assoziativen Arrays liegt.

- Namensräume werden in PostScript durch assoziative Arrays geschaffen.
- Auf dem Stack der assoziativen Arrays (*dictionary stack*) liegen mehrere solcher Namensräume.
- Wenn für einen Namen das zugehörige Objekt gesucht wird, geht der Interpreter alle Namensräume in dem entsprechenden Stack durch, bis er fündig wird.
- Standardmäßig sind auf diesem Stack `systemdict`, `globaldict` und `userdict`.
- In `systemdict` sind Operatoren wie `add` enthalten, `globaldict` dient als globaler Namensraum und `userdict` wird von `save` und `restore` beeinflusst.

smiley2.eps

```
gsave Smiley 5 setlinewidth stroke grestore
```

- Zum grafischen Zustand gehören u.a. die aktuelle Position, der aktuelle Pfad, die Konfiguration des Zeichenstifts und die aktuelle Abbildung des Koordinatensystems.
- Mit `gsave` kann der aktuelle grafische Zustand auf einen eigenen Stack gesichert werden.
- Mit `grestore` wird der vorherige grafische Zustand wieder restauriert.

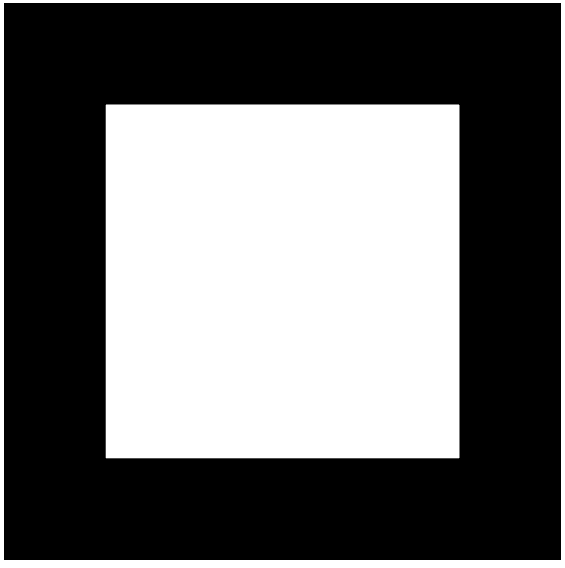


badbox.eps

```
/Box {
   newpath
   100 100 moveto
   200 100 lineto
   200 200 lineto
   100 200 lineto
   100 100 lineto
} def

Box 30 setlinewidth stroke
```

- Der Anfang und das Ende eines mit *stroke* gezogenen Striches unterliegt der mit *setlinecap* veränderbaren Konfiguration.
- In jedem Falle unterscheidet sich das von einem durchgezogenen Pfad.

Schließen eines Pfads

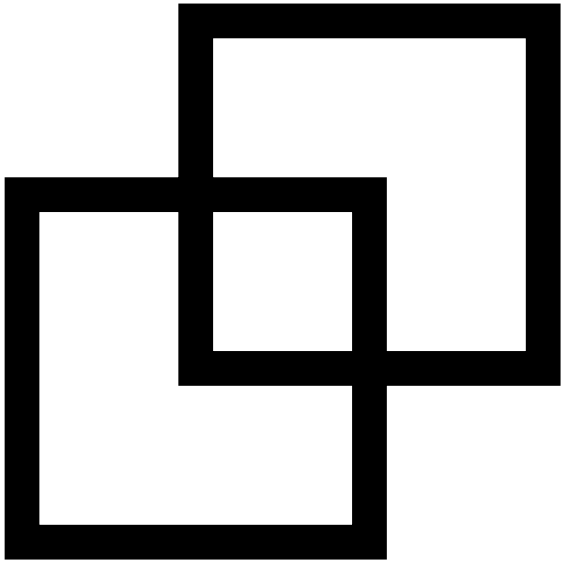


goodbox.eps

```
/Box {
   newpath
   100 100 moveto
   200 100 lineto
   200 200 lineto
   100 200 lineto
   closepath
} def

Box 30 setlinewidth stroke
```

- Wenn ein Pfad explizit ringförmig geschlossen werden soll, dann geht dies mit *closepath*.
- Das Ziehen einer geraden Linie von dem letzten Punkt zu dem Anfang der Kurve erfolgt dabei implizit.

Mehrfaches Schließen bei einem Pfad



twoboxes.eps

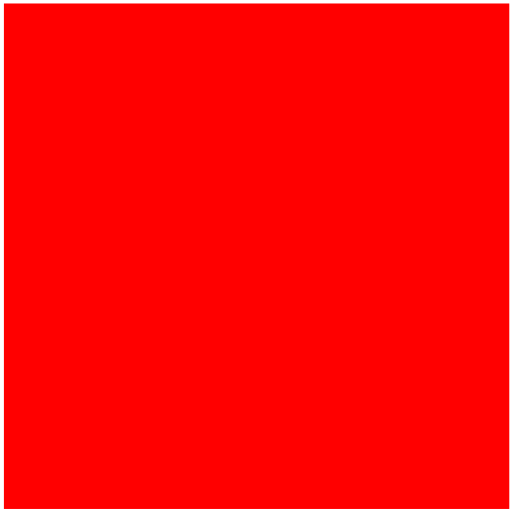
```
/TwoBoxes {
   newpath
   100 100 moveto 200 100 lineto
   200 200 lineto 100 200 lineto
   closepath
   150 150 moveto 250 150 lineto
   250 250 lineto 150 250 lineto
   closepath
} def

TwoBoxes 10 setlinewidth stroke
```

- Ein Pfad kann aus beliebig vielen zusammenhängenden Kurven bestehen.
- *closepath* schließt dabei nur die letzte Kurve und keinesfalls den gesamten Pfad.

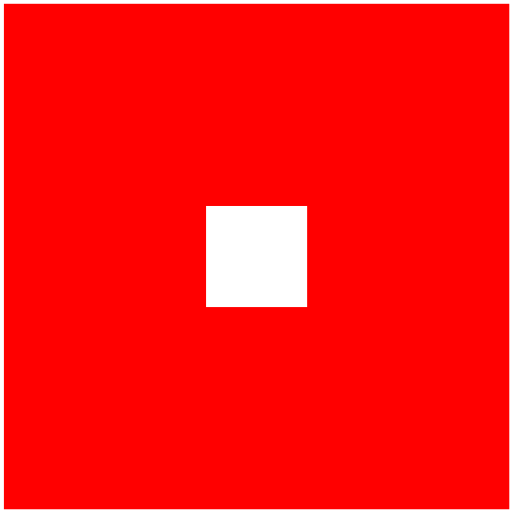
Füllen eines Pfads

filledboxes.eps

```
/TwoBoxes {
   newpath
   100 100 moveto 200 100 lineto
   200 200 lineto 100 200 lineto
   closepath
   100 300 moveto 200 300 lineto
   200 400 lineto 100 400 lineto
   closepath
} def

TwoBoxes 1 0 0 setrgbcolor fill
```



- Der *fill*-Operator schließt alle möglicherweise noch offenen Kurven eines Pfads und füllt diese dann mit der aktuellen Farbe.
- Wie bei *stroke* wird danach implizit der *newpath*-Operator aufgerufen.

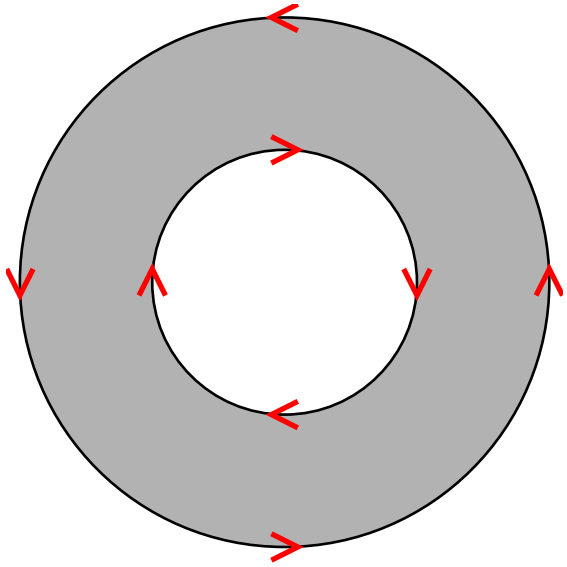


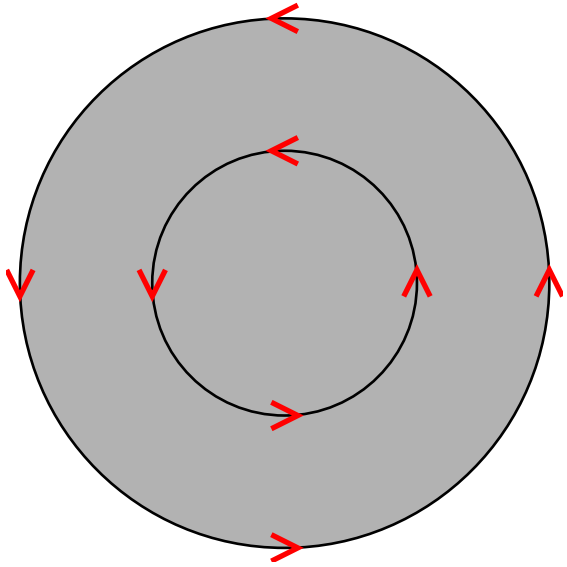
nestedboxes.eps

```
/NestedBoxes {
   newpath
   100 100 moveto 200 100 lineto
   200 200 lineto 100 200 lineto
   closepath
   140 140 moveto 140 160 lineto
   160 160 lineto 160 140 lineto
   closepath
} def

NestedBoxes 1 0 0 setrgbcolor fill
```

- In einfachen Fällen erscheint es trivial, was innen ist und somit gefüllt werden soll.
- Bei mit sich selbst überschneidenden Kurven und Flächen ist es nicht mehr so trivial.

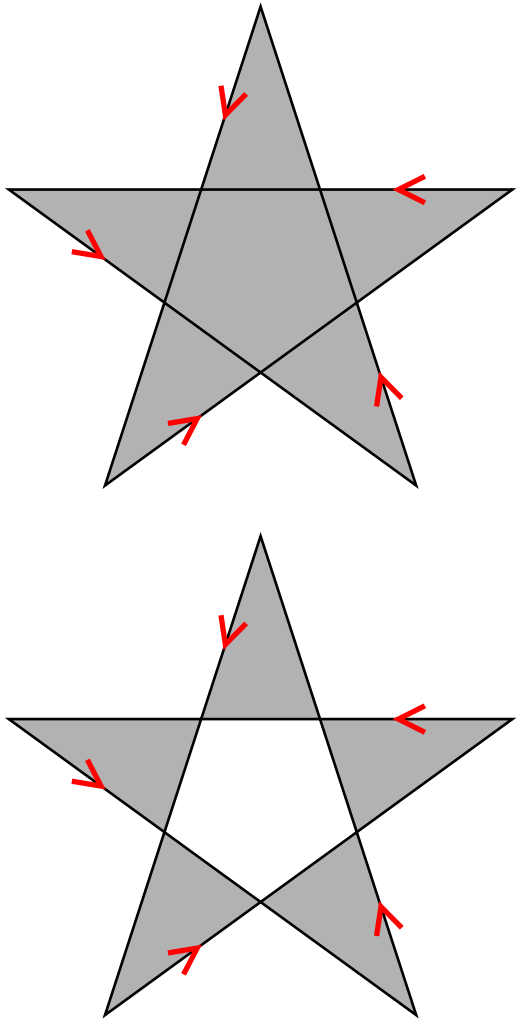
Non-zero winding number rule





- Ist ein Punkt p innerhalb oder außerhalb des Pfades?
- Bei der *Non-zero winding number rule* wird ausgehend von dem Punkt p ein beliebiger Strahl gewählt. Die *winding number* wird zu Beginn auf 0 gesetzt. Wenn dieser Strahl eine Kurve schneidet, die von links nach rechts verläuft, wird die *winding number* um 1 erhöht, bei Kurven von rechts nach links um 1 gesenkt. Wenn all die Schnittpunkte berücksichtigt worden sind und die *winding number* gleich 0 ist, dann ist der Punkt außerhalb, ansonsten innen.
- Siehe Abschnitt 4.5.2 im *PostScript Language Reference Manual*

Even-odd rule



- Bei der etwas einfacheren und sich sehr viel effizienter umsetzbaren *even-odd rule* werden ausgehend von dem Strahl in p in eine beliebige Richtung nur die Schnittpunkte gezählt. Ist die Zahl ungerade, wird der Punkt als innen betrachtet.
- Der *eofill*-Operator in PostScript arbeitet nach dieser Regel.
- Das obere Pentagram wurde mit *fill* gefüllt, das untere mit *eofill*.

Abbildung des Koordinatensystems

- Die Koordinate (x, y) wird abgebildet zu

$$(x, y) \begin{pmatrix} a & b \\ c & d \end{pmatrix} + (t_x, t_y)$$

- In PostScript wird dies zu einem sechs-elementigen Array zusammengefasst: $[a\ b\ c\ d\ t_x\ t_y]$.
- Die Abbildung von (x, y) zu (x', y') lässt sich auch durch eine 3×3-Matrix darstellen:

$$(x', y', 1) = (x, y, 1) \begin{pmatrix} a & b & 0 \\ c & d & 0 \\ t_x & t_y & 1 \end{pmatrix}$$

Verschieben

- Operation: t_x t_y `translate`
- Verschiebung der Koordinaten um (t_x, t_y).
- Die entsprechende 3×3 Transformationsmatrix:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{pmatrix}$$

Skalieren

- Operation: s_x s_y `scale`
- Skalieren der Koordinaten um (s_x, s_y).
- Die entsprechende 3×3 Transformationsmatrix:

$$\begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- Operation: θ `rotate`
- Drehen der Koordinaten um θ.
- Die entsprechende 3×3 Transformationsmatrix:

$$\begin{pmatrix} cos\ \theta & sin\ \theta & 0 \\ -sin\ \theta & cos\ \theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- Zum grafischen Zustand gehört die aktuelle Transformationsmatrix (*CTM*).
- Die Operatoren `translate`, `scale` und `rotate` multiplizieren jeweils die angegebenen 3×3 Transformationsmatrizen mit der *CTM*, um daraus die neue *CTM* zu bestimmen.
- Mit `concat` kann die Matrix auch explizit angegeben werden:

  matrix `concat`

  Dies entspricht $CTM = matrix \times CTM$

smiley3.eps

```
% x y radius linewidth DrawSmiley -
/DrawSmiley {
   4 dict begin
   /linewidth exch def
   /radius exch 100 div def
   /y exch 100 radius mul sub def
   /x exch 100 radius mul sub def
   gsave
   x y translate
   radius radius scale
   Smiley
   linewidth setlinewidth
   stroke
   grestore
   end
} def

100 100 100 5 DrawSmiley
250 150 50 2 DrawSmiley
```

smiley3.eps

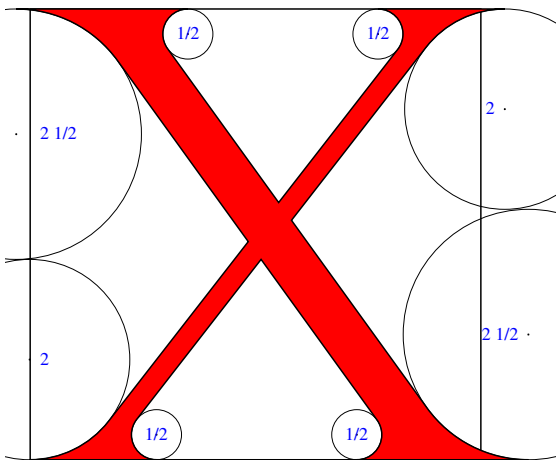
```
4 dict begin
/linewidth exch def
/radius exch 100 div def
/y exch 100 radius mul sub def
/x exch 100 radius mul sub def
% ...
end
```

- Mit `4 dict` wird ein leeres assoziatives Array mit einer Mindestkapazität von 4 angelegt.
- Mit `begin` wird ein assoziatives Array auf den *dictionary stack* befördert.
- Mit `end` wird das oberste Element des *dictionary stack* abgeräumt.

smiley3.eps

```
4 dict begin
/linewidth exch def
/radius exch 100 div def
/y exch 100 radius mul sub def
/x exch 100 radius mul sub def
% ...
end
```

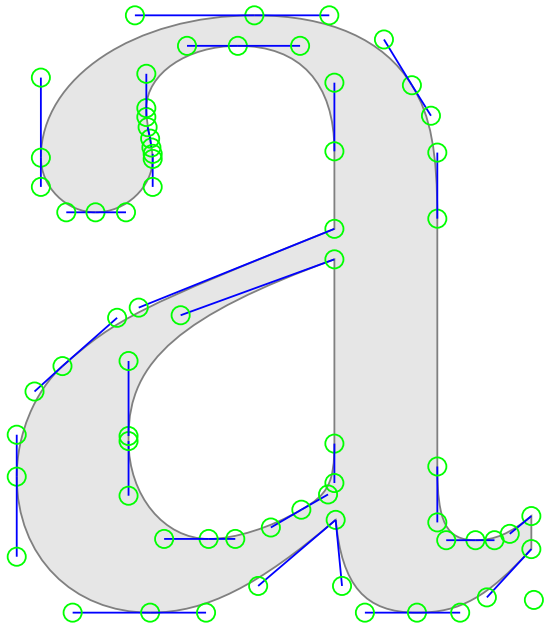
- Die Parameter müssen in umgekehrter Reihenfolge (LIFO!) abgeholt werden.
- `exch` vertauscht die beiden obersten Elemente auf dem Stack.
- Da `Smiley` den Mittelpunkt nicht auf $(0, 0)$, sondern auf $(100, 100)$ setzt mit einem Radius von 100, müssen die Parameter entsprechend angepasst werden.



Wie können oder sollten Kurven repräsentiert werden?

- Torniello kam mit Geraden und Kreisen aus.
- Einige versuchten es nur mit Punkten und Geraden (etwa Hershey im Jahr 1972).
- 1976 wurden Splines bei Xerox eingesetzt.
- Ronald McIntosh und Peter Purdy schlugen in einer 1978 veröffentlichten Arbeit Spiralen vor.

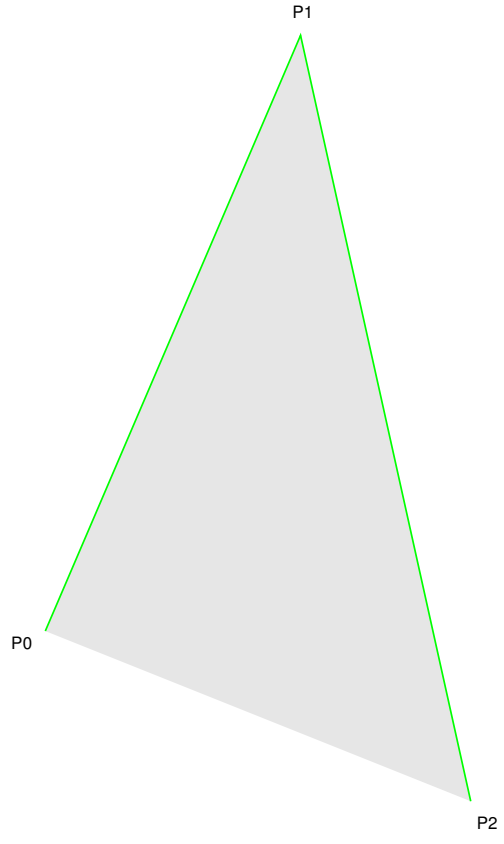
Ziele einer Repräsentierung für Kurven



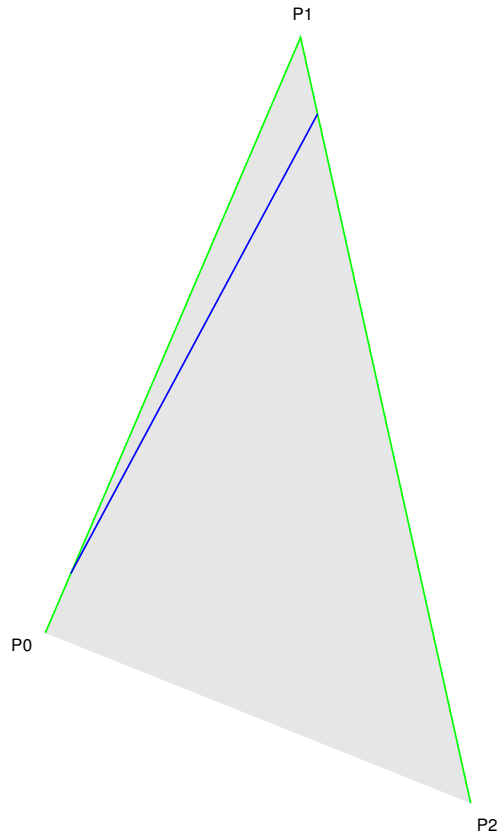
Die Darstellung einer Kurve sollte

- das Original bis zu einer gegebenen Auflösung möglichst gut approximieren, so dass dem menschlichen Auge der Unterschied nicht auffällt,
- ausreichend komprimierend sein,
- Veränderungen einer Kurve erleichtern,
- effizient und numerisch stabil berechenbar sein und
- affine Abbildungen unterstützen.

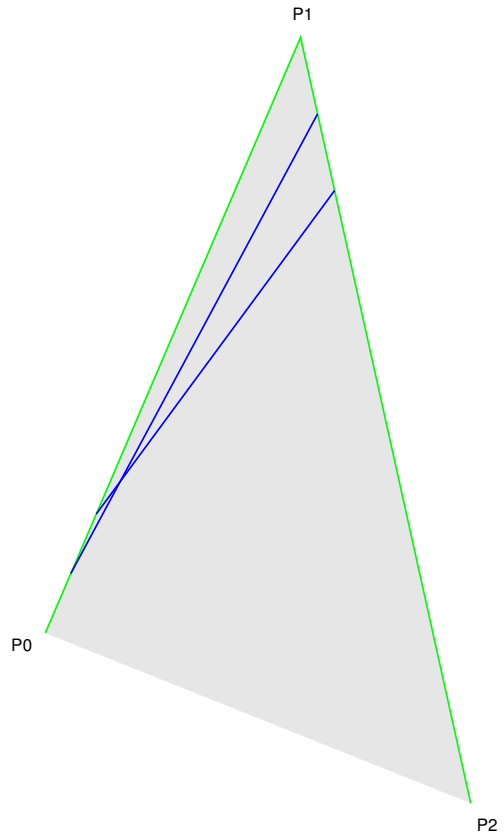
- Pierre Etienne Bézier (1910–1999) war bei Renault Leiter der Abteilung für die Entwicklung von Werkzeugmaschinen für die Automobilproduktion.
- 1960 gab es erste Computer-kontrollierte Maschinen (*Computer Aided Manufacturing*), die die Herstellung von 3-dimensionalen Oberflächen aus Holz oder Stahl erlaubten. Das Problem war die Programmierung einer passenden 3-dimensionalen Fläche.
- Bézier löste das Problem mit den nach ihm benannten 2-dimensionalen Kurven, die mit zwei Endpunkten und zwei Kontrollpunkten in der Mitte vollständig spezifiziert waren. Die 2-dimensional definierten Kurven lassen sich kreuzweise verflechten, um eine 3-dimensionale Fläche zu definieren.
- Unabhängig davon stießen Paul de Casteljau (bei Citroen) und der Flug-Ingenieur James Ferguson (bei Boeing) zuvor auf die gleichen Kurven. Das stellte sich jedoch erst später heraus, weil damals diese Techniken weitgehend als Firmengeheimnisse unter Verschluss blieben.



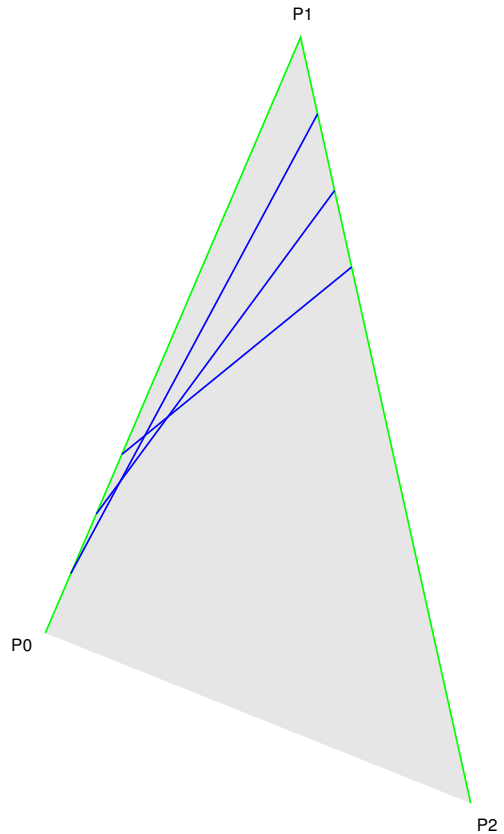
- Gegeben seien drei Punkte P_0, P_1 und P_2 in $\mathbb{R}^2$.



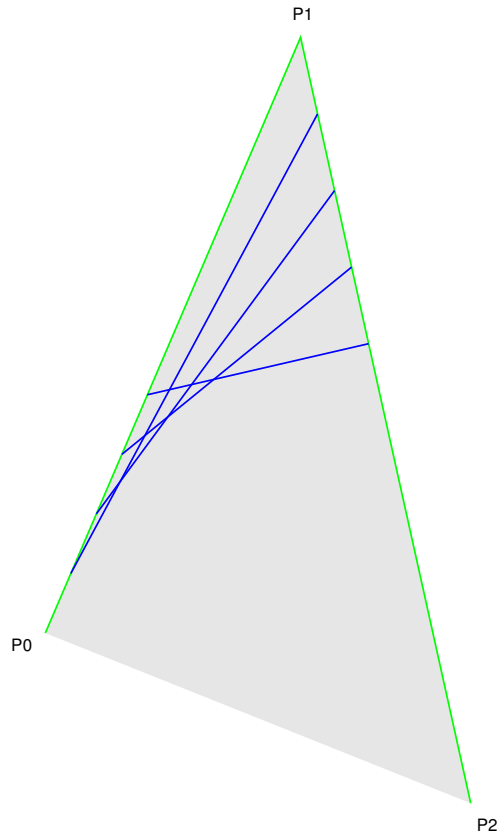
- Gegeben seien drei Punkte P_0, P_1 und P_2 in $\mathbb{R}^2$.
- Notation für die Spezifikation eines Punktes auf einer Linie zwischen zwei Punkten für $t \in [0, 1]$: $t[(x_1, y_1), (x_2, y_2)] = (x_1, y_1) + t(x_2 - x_1, y_2 - y_1)$
- Wir verbinden $0.1[P_0, P_1]$ mit $0.1[P_1, P_2]$.



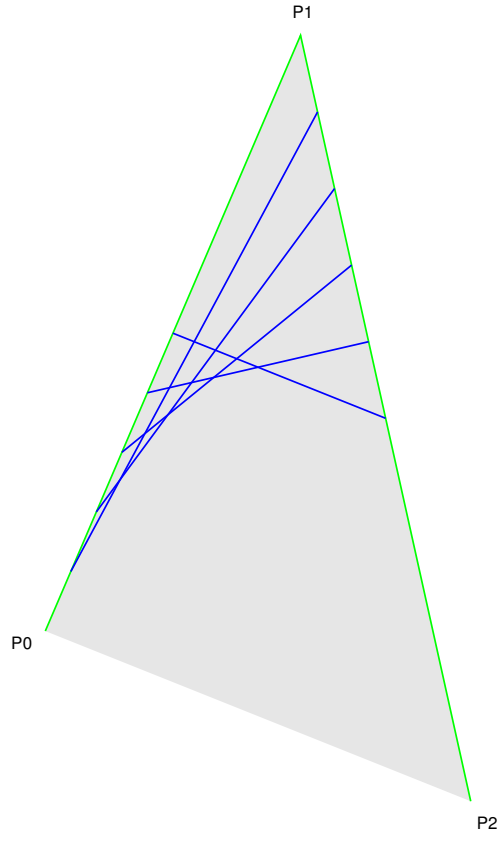
- Gegeben seien drei Punkte P_0, P_1 und P_2 in $\mathbb{R}^2$.
- Notation für die Spezifikation eines Punktes auf einer Linie zwischen zwei Punkten für $t \in [0, 1]$: $t[(x_1, y_1), (x_2, y_2)] = (x_1, y_1) + t(x_2 - x_1, y_2 - y_1)$
- Wir verbinden $0.2[P_0, P_1]$ mit $0.2[P_1, P_2]$.



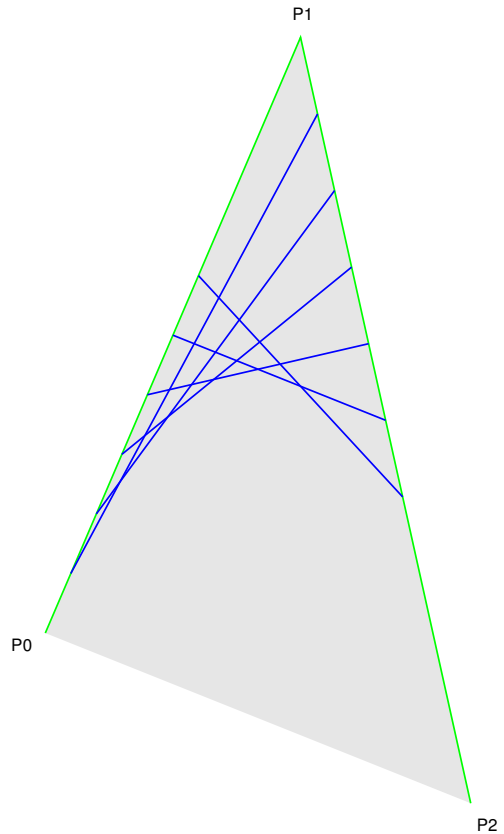
- Gegeben seien drei Punkte P_0, P_1 und P_2 in $\mathbb{R}^2$.
- Notation für die Spezifikation eines Punktes auf einer Linie zwischen zwei Punkten für $t \in [0, 1]$: $t[(x_1, y_1), (x_2, y_2)] = (x_1, y_1) + t(x_2 - x_1, y_2 - y_1)$
- Und so weiter...



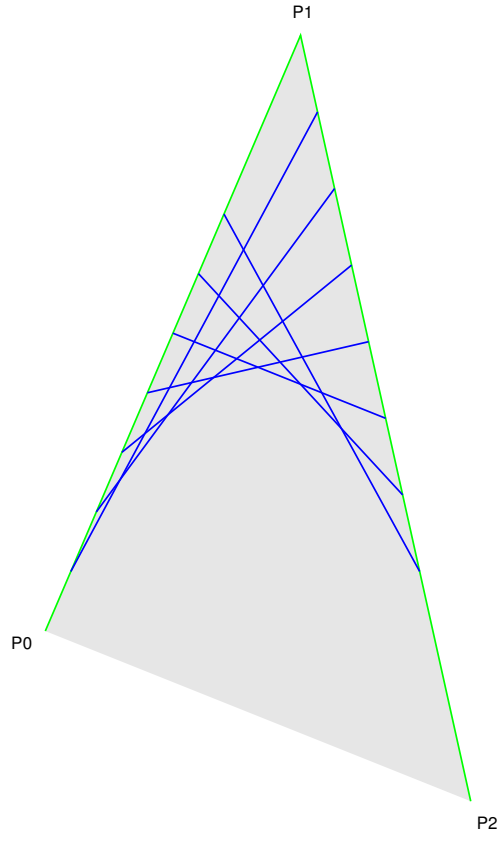
- Gegeben seien drei Punkte P_0, P_1 und P_2 in $\mathbb{R}^2$.
- Notation für die Spezifikation eines Punktes auf einer Linie zwischen zwei Punkten für $t \in [0, 1]$: $t[(x_1, y_1), (x_2, y_2)] = (x_1, y_1) + t(x_2 - x_1, y_2 - y_1)$
- Und so weiter...



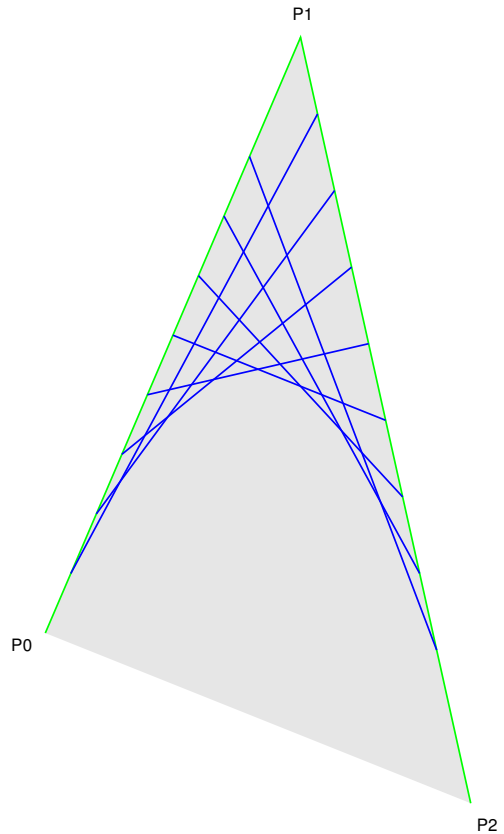
- Gegeben seien drei Punkte P_0, P_1 und P_2 in $\mathbb{R}^2$.
- Notation für die Spezifikation eines Punktes auf einer Linie zwischen zwei Punkten für $t \in [0, 1]$: $t[(x_1, y_1), (x_2, y_2)] = (x_1, y_1) + t(x_2 - x_1, y_2 - y_1)$
- Und so weiter...



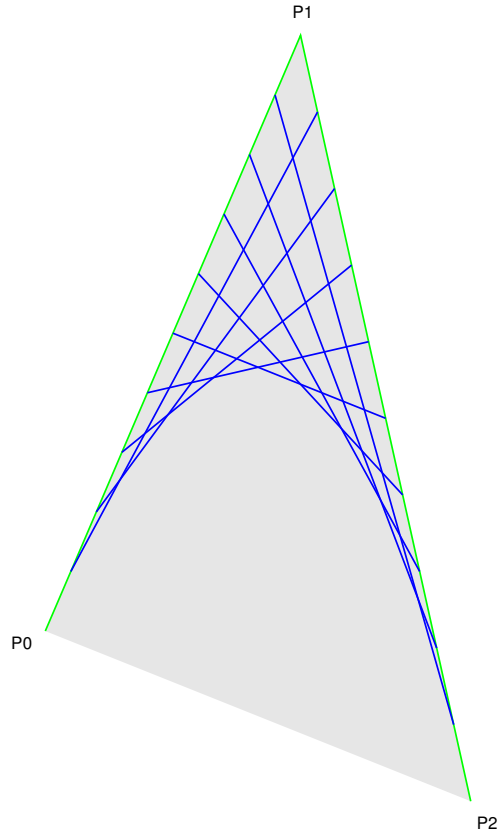
- Gegeben seien drei Punkte P_0, P_1 und P_2 in $\mathbb{R}^2$.
- Notation für die Spezifikation eines Punktes auf einer Linie zwischen zwei Punkten für $t \in [0, 1]$: $t[(x_1, y_1), (x_2, y_2)] = (x_1, y_1) + t(x_2 - x_1, y_2 - y_1)$
- Und so weiter...



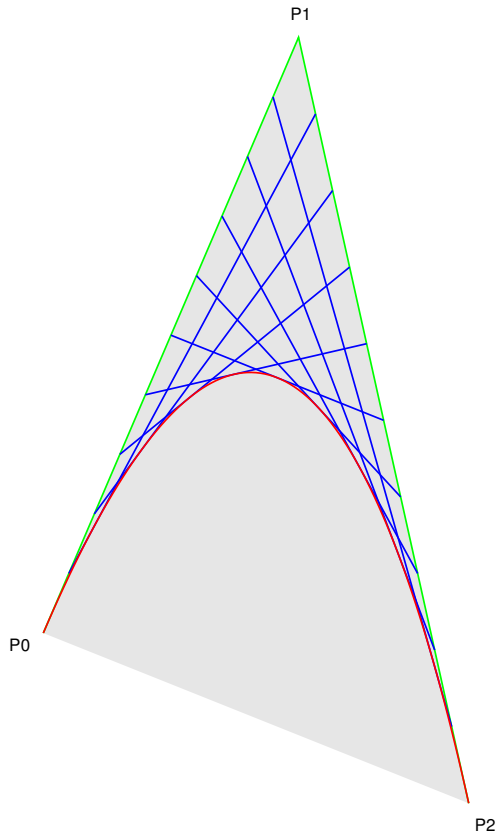
- Gegeben seien drei Punkte P_0, P_1 und P_2 in $\mathbb{R}^2$.
- Notation für die Spezifikation eines Punktes auf einer Linie zwischen zwei Punkten für $t \in [0, 1]$: $t[(x_1, y_1), (x_2, y_2)] = (x_1, y_1) + t(x_2 - x_1, y_2 - y_1)$
- Und so weiter...



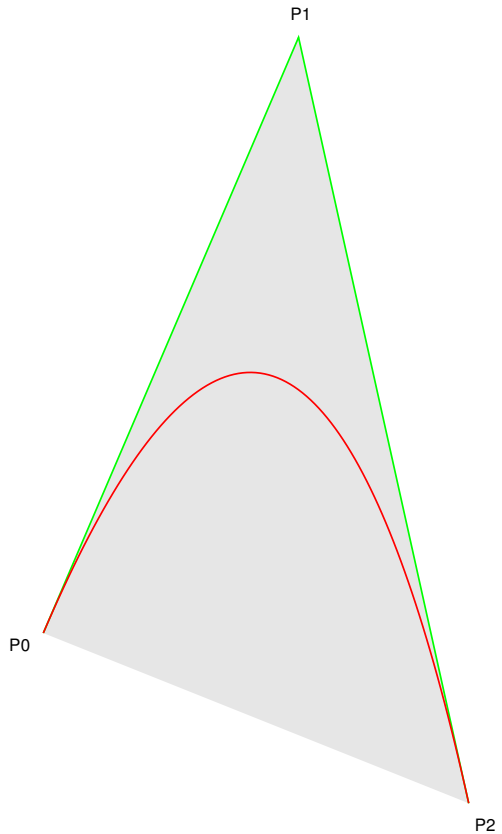
- Gegeben seien drei Punkte P_0, P_1 und P_2 in $\mathbb{R}^2$.
- Notation für die Spezifikation eines Punktes auf einer Linie zwischen zwei Punkten für $t \in [0, 1]$: $t[(x_1, y_1), (x_2, y_2)] = (x_1, y_1) + t(x_2 - x_1, y_2 - y_1)$
- Und so weiter...



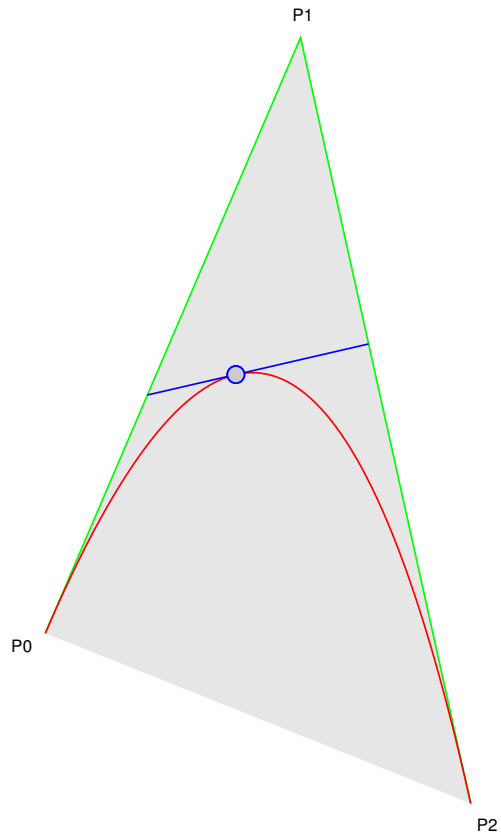
- Gegeben seien drei Punkte P_0, P_1 und P_2 in $\mathbb{R}^2$.
- Notation für die Spezifikation eines Punktes auf einer Linie zwischen zwei Punkten für $t \in [0, 1]$: $t[(x_1, y_1), (x_2, y_2)] = (x_1, y_1) + t(x_2 - x_1, y_2 - y_1)$
- Und so weiter...



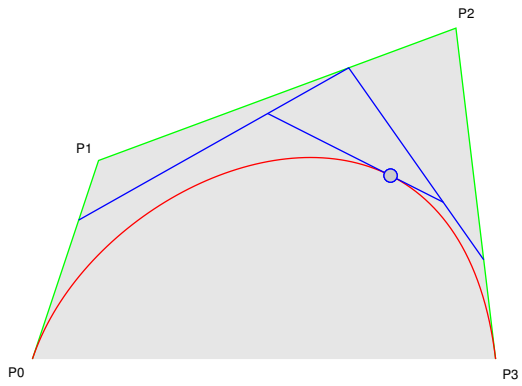
- Wenn das Spiel weiter getrieben wird, nähern wir uns der roten Kurve.



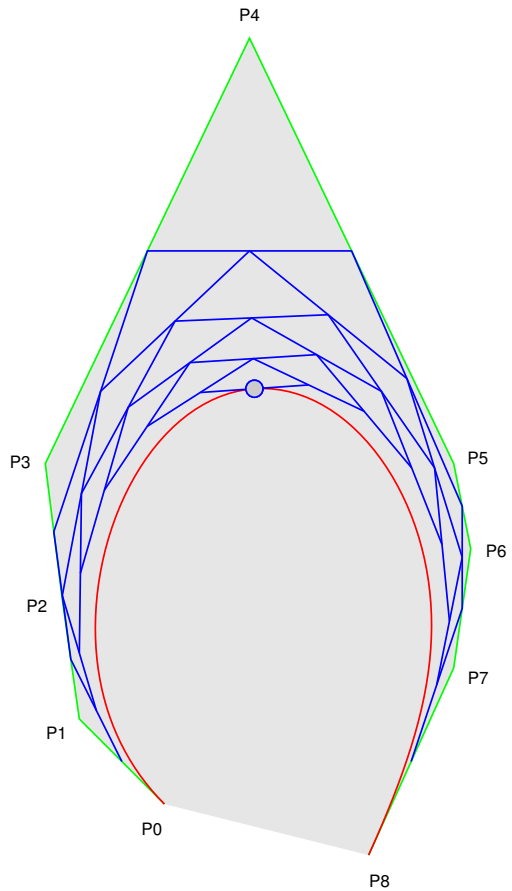
- Wenn das Spiel weiter getrieben wird, nähern wir uns der roten Kurve.



- Die rote Kurve kann auch definiert werden über $C(t) = t[t[P_0, P_1], t[P_1, P_2]]$ für $t \in [0, 1]$.
- Das lässt sich umrechnen zu: $C(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2$ für $t \in [0, 1]$.
- Das Diagramm demonstriert dies für $t = 0,4$.



- Wenn ein Punkt hinzukommt, dann vergrößert sich das System der Zwischenpunktberechnungen um eine Ebene.
- Entsprechend erhalten wir folgende Kurve: $t[t[t[P_0, P_1], t[P_1, P_2]], [t[P_1, P_2], t[P_2, P_3]]]$ für $t \in [0, 1]$.
- Das lässt sich umrechnen zu: $C(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3$
- Das Diagramm demonstriert dies für $t = 0,7$.

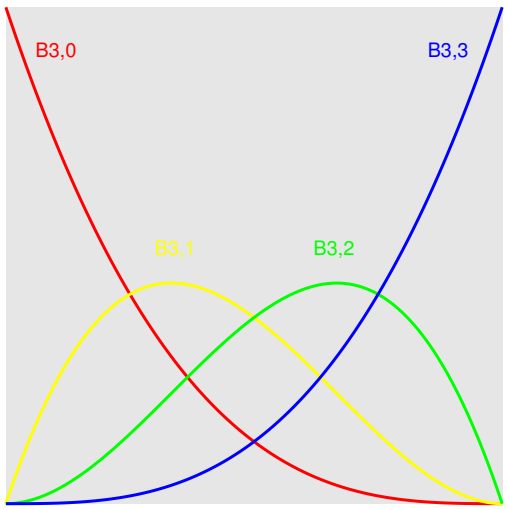


- Eine Bézier-Kurve n-ten Grades wird definiert über $n+1$ Punkte $P_0 \ldots P_n$:

$$C(t) = \sum_{i=0}^{n} B_{n,i}(t) P_i$$

- Die Bernstein-Polynome $B_{n,i}(t)$ werden wie folgt definiert:

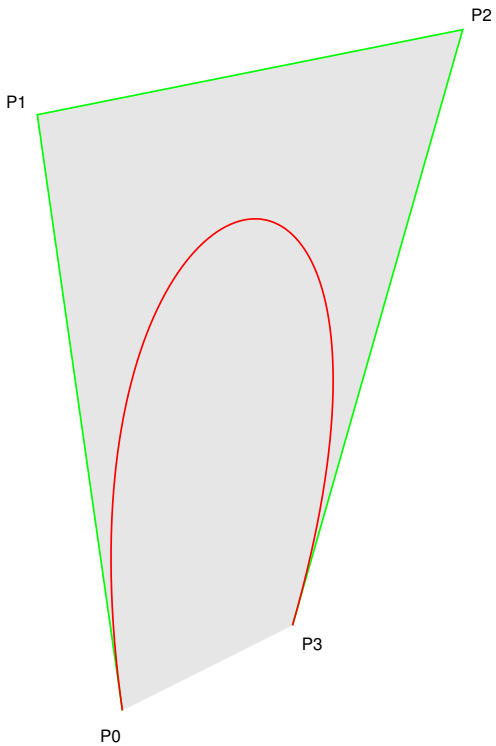
$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$



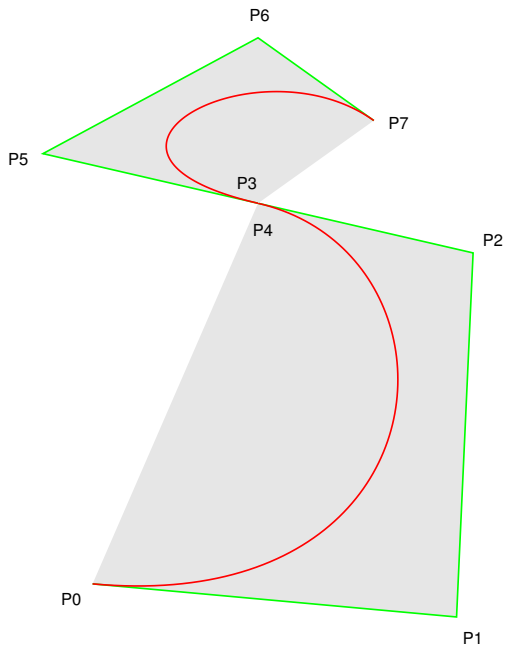
- Die Bernstein-Polynome sind benannt nach Sergei Natanowitsch Bernstein, der sie 1911 zuerst in einem Beweis verwendete. Sie haben folgende Eigenschaften auf dem Intervall $[0, 1]$:
- Positivität: $B_{n,i}(t) > 0$ für $t \in (0, 1)$
- Zerlegung der Eins:

$$\sum_{i=0}^{n} B_{n,i}(t) = 1 \quad \forall t \in [0, 1]$$

Anfang und Ende der Bézier-Kurve

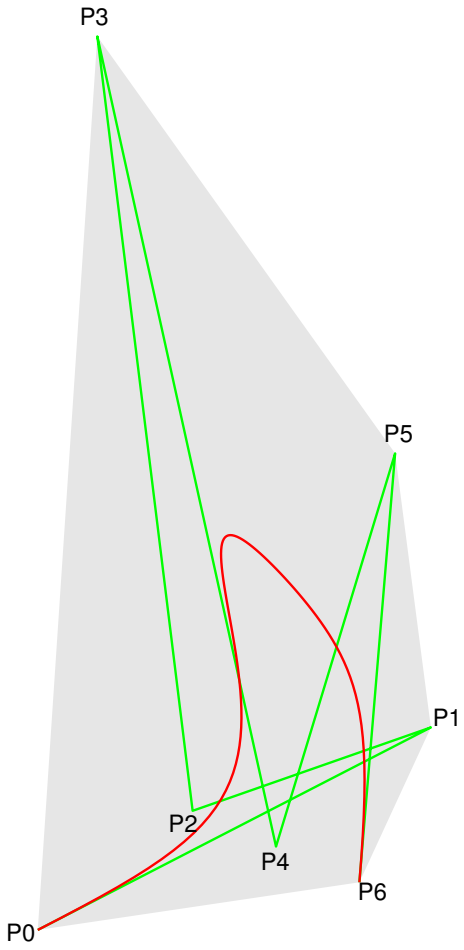


- $\overline{P_0P_1}$ und $\overline{P_{n-1}P_n}$ sind Tangenten der Bézier-Kurve, die diese am Anfangs- bzw. Endpunkt berühren.
- Somit kontrollieren die Kontrollpunkte P_1 und P_{n-1} direkt die tangentiale Ausrichtung der Kurvenenden.
- Das erleichtert das knick-freie Zusammenfügen mehrerer Bézier-Kurven.

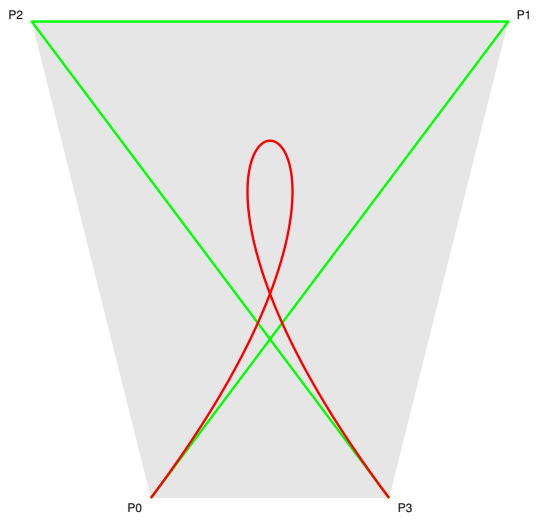


- In diesem Diagramm sind zwei Bézier-Kurven knickfrei zusammengelegt worden (G^1-Stetigkeit).
- Allerdings ist damit noch keine C^1-Stetigkeit gewonnen, da die Tangenten zwar in der Richtung übereinstimmen, jedoch noch nicht notwendigerweise in ihrem Betrag.
- Dies wird gelegentlich als G^1-Stetigkeit bezeichnet.

Konvexe Hülle der Kontrollpunkte

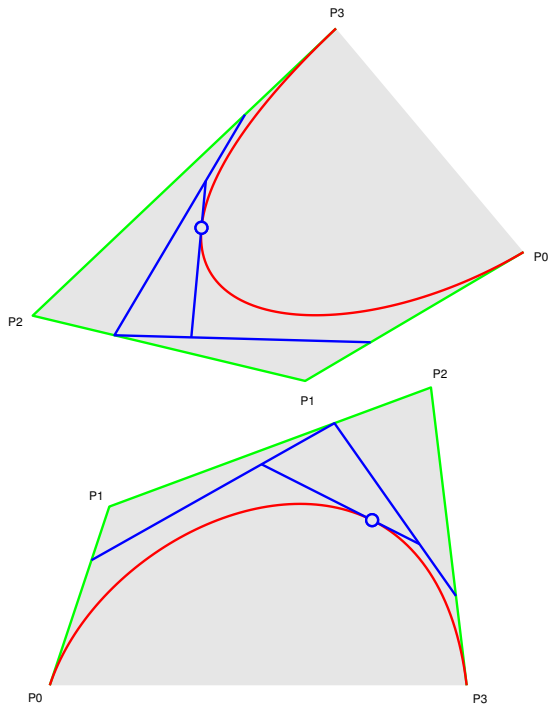


- Bézier-Kurven verlaufen immer innerhalb der konvexen Hülle ihrer Kontrollpunkte.
- Im Diagramm ist die konvexe Hülle grau hinterlegt.

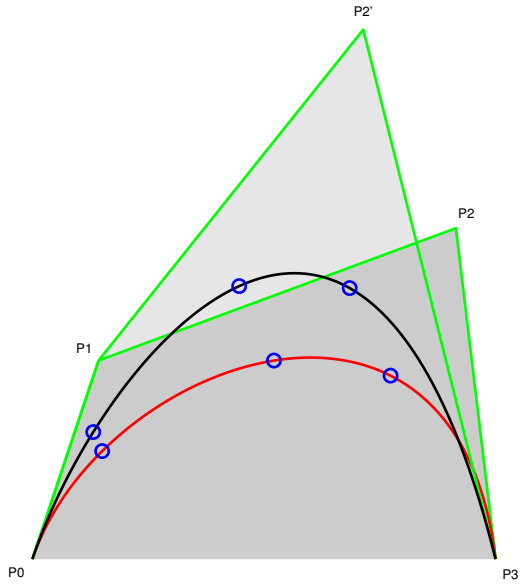


- Grundsätzlich können sich Bézier-Kurven sich selbst überschneiden.
- Allerdings ist die Zahl der Überschneidungen einer Bézier-Kurve nach oben beschränkt durch die Zahl der Überschneidungen des Polygon-Zuges, der durch die Kontrollpunkte geht.

Affine Invarianz



- Bei affinen Transformationen von Bézier-Kurven genügt es, die Kontrollpunkte $P_0, \ldots, P_n$ entsprechend abzubilden und dann ausgehend von den abgebildeten Kontrollpunkten die Kurve neu zu zeichnen.
- Das vereinfacht dramatisch die Implementierung der Operatoren `translate`, `scale` und `rotate` in PostScript.
- Diese Vereinfachung betrifft sämtliche Kurven in PostScript, da jede Kurve in PostScript nur aus einer Folge von Bézier-Kurven besteht.

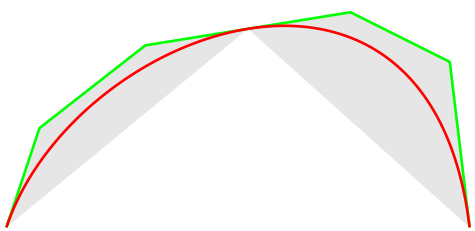


- Wenn genau ein Kontrollpunkt P_i durch P_i' ersetzt wird, dann bewegen sich alle Punkte der Bézier-Kurve in der Richtung von $P_i\vec{P}_i'$.
- Diese Bewegung wird in genau dem Maße gedämpft, wie es dem Gewicht von P_i entsprechend des betreffenden Gliedes des Bernstein-Polynoms an der jeweiligen Kurvenstelle entspricht:

$$C'(t) = C(t) + B_{n,i}(t)(P_i' - P_i)$$

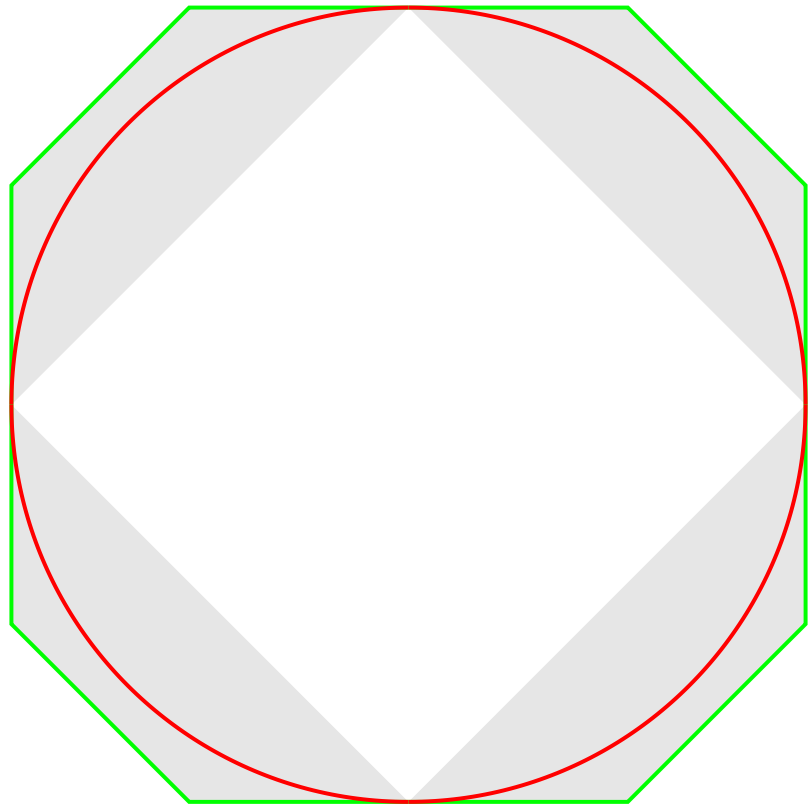
- Jede Änderung eines Kontrollpunktes macht sich **global** bemerkbar.

- Zu berechnen ist $C(t)$ für $t \in [0,1]$.
- Rekursiv werden Punkte $P_{i,j}$ definiert:
  - $P_{0,i} = P_i$ für $i = 0 \ldots n$
  - $P_{k,i} = t[P_{k-1,i}, P_{k-1,i+1}]$ für $k = 1 \ldots n$ und $i = 0 \ldots n-k$
- Dann ist $C(t) = P_{n,0}$.
- Um doppelte Berechnungen zu vermeiden, empfiehlt sich ein Pyramidenschema.
- Der Berechnungsaufwand ist $O(n^2)$.
- Bei kubischen Bézier-Kurven sind 6 Linear-Kombinationen zu berechnen pro Zwischenpunkt.
- Dieses Berechnungsverfahren ist effizient und numerisch stabil.



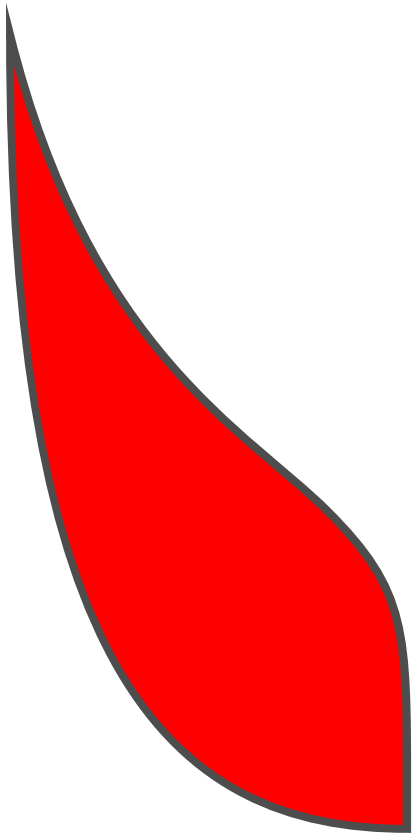
- Jede Bézier-Kurve kann für jedes $t \in (0, 1)$ in zwei Bézier-Kurven des gleichen Grades zerlegt werden.
- Die Kontrollpunkte der beiden neuen Kurven ergeben sich aus den Zwischenpunkten, die nach dem Algorithmus von de Casteljau berechnet worden sind:
  - 1. Kurve: $P_{0,0}, P_{1,0}, \ldots, P_{n,0}$
  - 2. Kurve: $P_{0,n}, P_{1,n-1}, \ldots, P_{n,0}$
- Zerlegungen können Rasterisierungs-Algorithmen vereinfachen, wenn t geeignet gewählt wird.

- Ziel ist ein Kreis um den Punkt $(0,0)$ mit Radius 1.
- Der Kreis wird in 4 Segmente entsprechend den Quadranten aufgeteilt.
- Für das Kreissegment von $(1,0)$ nach $(0,1)$ wird eine kubische Bézier-Kurve gesucht, die folgende Forderungen erfüllt:
  - Die Bézier-Kurve hat die gleiche Tangente wie der Kreis in $(1,0)$ und $(0,1)$.
  - Die Bézier-Kurve verläuft durch $M = (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})$.
- Wegen der ersten Forderung müssen die Kontrollpunkte von der Form $(1,a)$ bzw. $(b,1)$ sein. Wegen der Symmetrie ergibt sich $a = b$. Somit ist nur noch a so zu wählen, dass die Bézier-Kurve durch M läuft.
- Da $C(0,5) = \frac{1}{8}(4+3a, 4+3a)$, ergibt sich $a = 4\frac{\sqrt{2}-1}{3}$.
- In PostScript generiert der Operator `arc` eine Folge von Bézier-Kurven nach diesem Schema.



- In PostScript lassen sich kubische Bézier-Kurven mit dem Operator `curveto` zeichnen.
- Dabei gilt die aktuelle Position als P_0. Die Punkte P_1, P_2 und P_3 sind explizit zu spezifizieren.
- Beispiel: `newpath 100 100 moveto 150 250 420 350 450 100 curveto stroke`
- Die Kontrollpunkte müssen also explizit angegeben werden.
- John D. Hobby veröffentlichte 1986 Formeln zur Bestimmung der Kontrollpunkte, so dass nach von ihm angegebenen formalen Kriterien die Kurven möglichst „gut“ aussehen und passen.
- Dieses Verfahren ist Bestandteil von METAFONT/METAPOST, die es erlauben, Bézier-Kurven auf alternative Weise zu spezifizieren, ohne die Kontrollpunkte explizit angeben zu müssen.

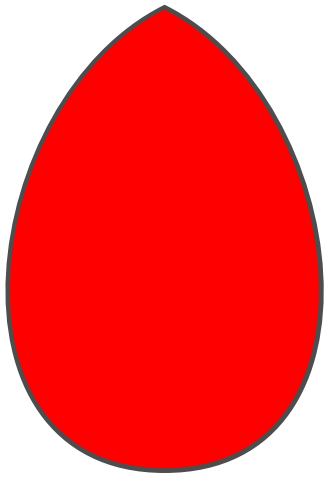
Tulpen mit Bézier-Kurven: Linkes Blatt



tulips.eps

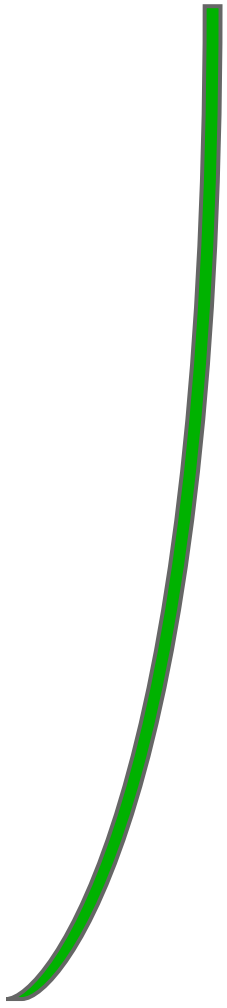
```
/LeftTulipLeaf {
   newpath
   0 0 moveto
   -30 0 -50 20 -50 100 curveto
   -40 60 -20 50 -10 40 curveto
   0 30 0 25 0 5 curveto
   closepath
} def
```

Tulpen mit Bézier-Kurven: Mittleres Blatt



tulips.eps

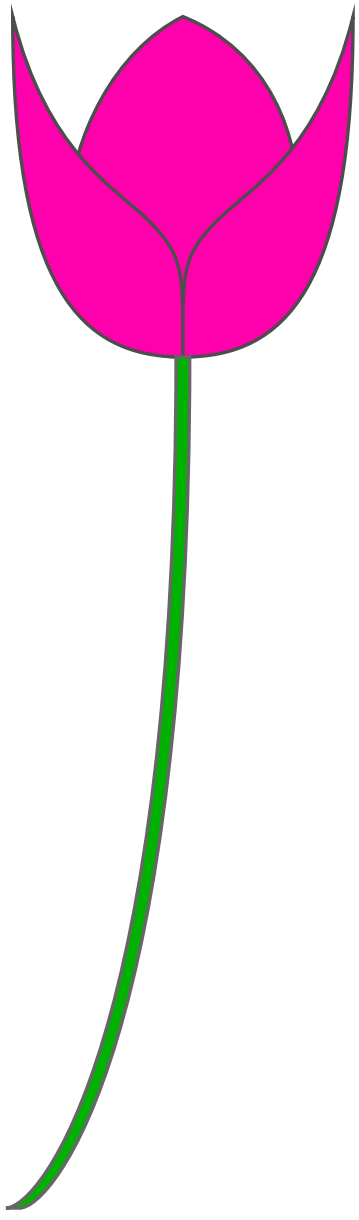
```
/MiddleTulipLeaf {
   newpath
   0 0 moveto -50 0 -40 80 0 100 curveto
   40 80 50 0 0 0 curveto
   closepath
} def

% color FillTulip -
/FillTulip {
   1 dict begin
   /color exch def
   gsave
      color aload pop sethsbcolor
      fill
   grestore
   0.3 setgray stroke
   end
} def
```



tulips.eps

```
/Stem {
   gsave
      -2 0 moveto
      -2 -200  -42 -250  -52 -250 curveto
      -48 -250 lineto
      -38 -250  2 -200  2 0 curveto
      closepath
      gsave
         0 0.7 0 setrgbcolor
         fill
      grestore
      0.4 0.4 0.4 setrgbcolor stroke
      stroke
   grestore
} def
```

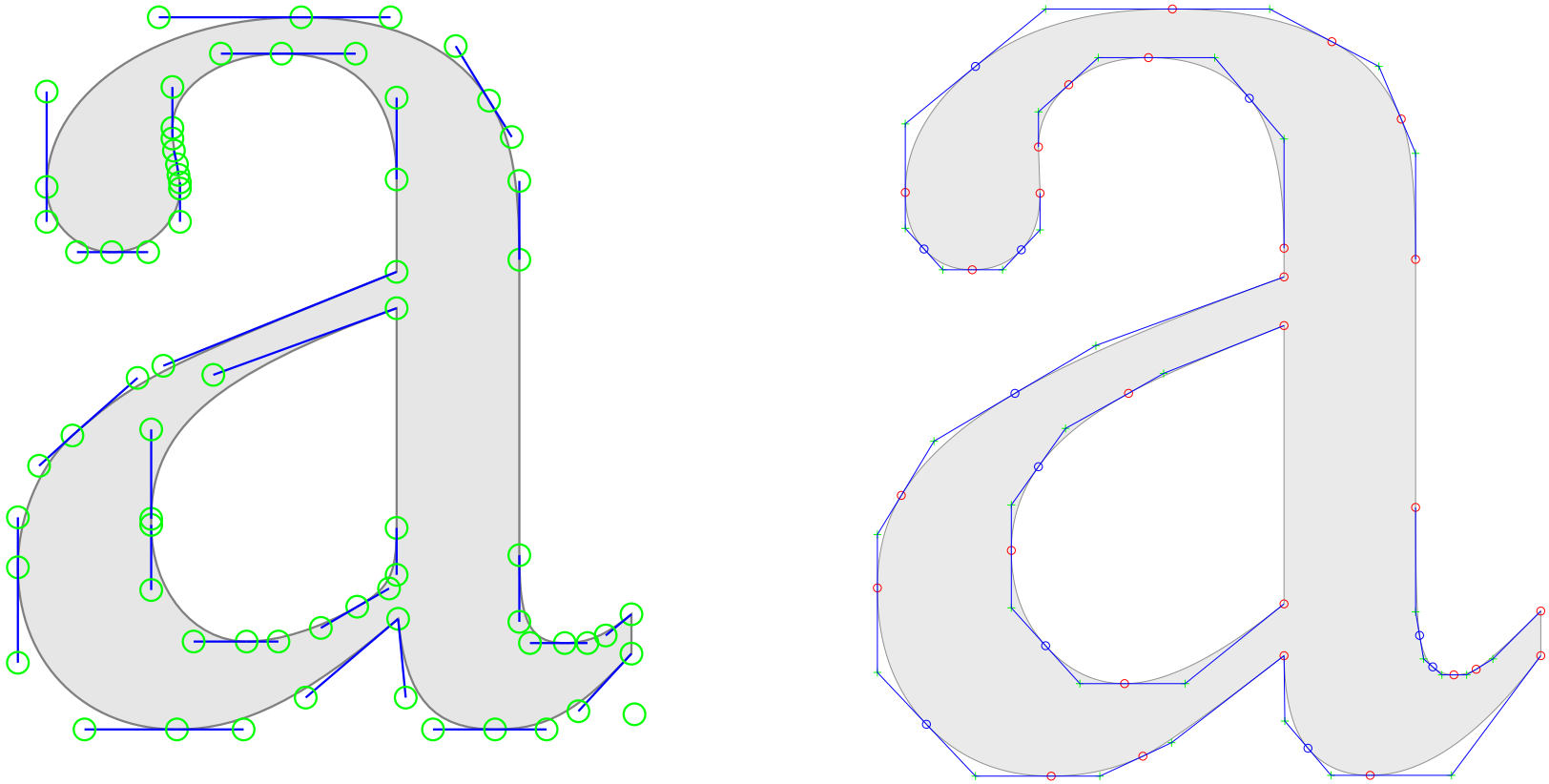
Tulpen mit Bézier-Kurven: Zusammenbau



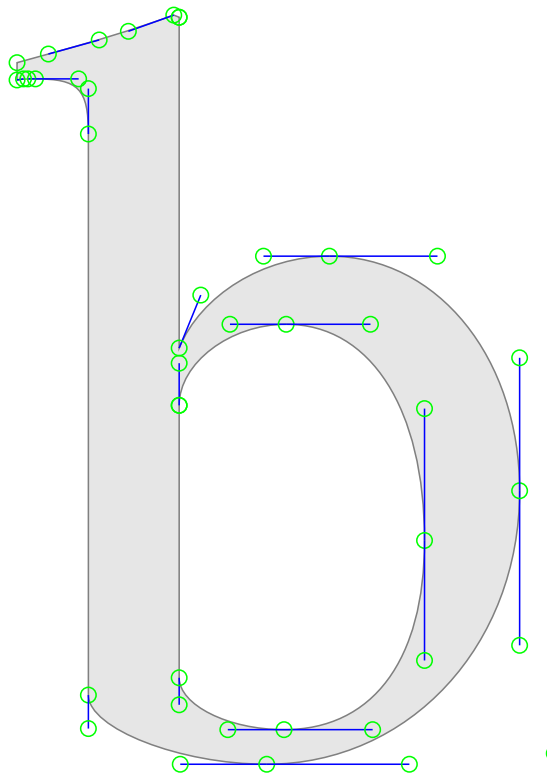
tulips.eps

```
/RandomColor {
   [
      rand 32 mod 220 add 256 div
      1 1
   ]
} def

/PlotTulip {
   1 dict begin
   /color RandomColor def
   gsave
      MiddleTulipLeaf color FillTulip
      LeftTulipLeaf color FillTulip
      RightTulipLeaf color FillTulip
      Stem
   grestore
   end
} def

PlotTulip
```

Quadratische vs. Kubische Bézier-Kurven



- Links ist der Type-1-Schriftschnitt für „a“ in *Times Roman* von URW++ Design and Development in Hamburg mit kubischen Bézier-Kurven.
- Rechts ist „a“ im TrueType-Schriftschnitt für *Times New Roman* mit quadratischen Bézier-Kurven.



- Grundsätzlich werden Schnitte einzelner Schriftzeichen auf Basis von Kurven konstruiert – genauso wie die bisherigen geometrischen Figuren auch.
- Eine Reihe von Figuren für Schriftzeichen bilden einen Schriftschnitt, der in speziellen assoziativen Arrays (*dictionaries*) zusammengefasst wird, der einigen speziellen Konventionen genügt.
- PostScript bietet eine Vielzahl von speziellen Operatoren an, die mit Schriftschnitten und Figuren für Schriftzeichen umgeht.
- Dies erlaubt es, Schriftschnitte besonders effizient zu implementieren (z.B. durch die Verwendung von Caches).

hello.ps

```
/Times-Roman findfont 12 scalefont setfont
50 700 moveto
(Hello World!) show
showpage
```

- *findfont* sucht nach dem genannten Schriftschnitt und lädt das zugehörige assoziative Array auf den Stack.
- Mit `12 scalefont` wird der Schriftschnitt entsprechend skaliert. Das wäre prinzipiell auch mit dem *scale*-Operator machbar. *scalefont* bezieht sich aber nur auf den einen Schriftschnitt und nicht auf die übrigen Pfade.
- Mit *setfont* wird der oben auf dem Stack liegende Schriftschnitt zum aktuellen Schriftschnitt. Es gibt hierfür keine Voreinstellung!
- *show* erwartet eine Zeichenkette und stellt diese mit dem aktuellen Schriftschnitt an der aktuellen Position (die wohldefiniert sein muss) dar.

Standardmäßig gehören die drei Schriftfamilien Times-Roman, Helvetica und Courier zu den unter PostScript verfügbaren Schriftschnitten:

Times-Roman	Helvetica	Courier
Times-Italic	*Helvetica-Oblique*	*Courier-Oblique*
Times-Bold	**Helvetica-Bold**	**Courier-Bold**
Times-BoldItalic	***Helvetica-BoldOblique***	***Courier-BoldOblique***

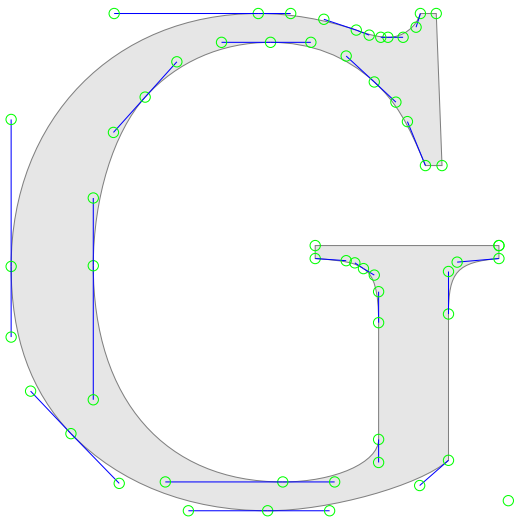
Hinzu kommt noch ein Schriftschnitt für diverse (insbesondere mathematische) Symbole. Optional stehen typischerweise zahlreiche weitere Schriftschnitte zur Verfügung.

Ivan Tschichold, 1928:

> *Der Mensch des 15. Jahrhunderts stand aufrecht vor dem Lesepult und las sich oder anderen den Inhalt laut vor. Daher die großen Lettern, die für die Mehrzahl der gotischen Bücher charakteristisch sind. Erst die zunehmende Beschleunigung des Lesetempos machte in der Folgezeit die Verwendung kleinerer Typen möglich und notwendig. Das laute und langsame Lesen, das „Abtasten“ des Einzelbuchstabens, des Einzelworts, ist in unserer Zeit dem Überfliegen des Textes gewichen. Die Lesetechnik des heutigen Menschen erzeugte die spezifische Form des Zeitungssatzes [...] Die optische Erscheinung der Zeitung gibt ein Sinnbild des heutigen Lebenstempos.*

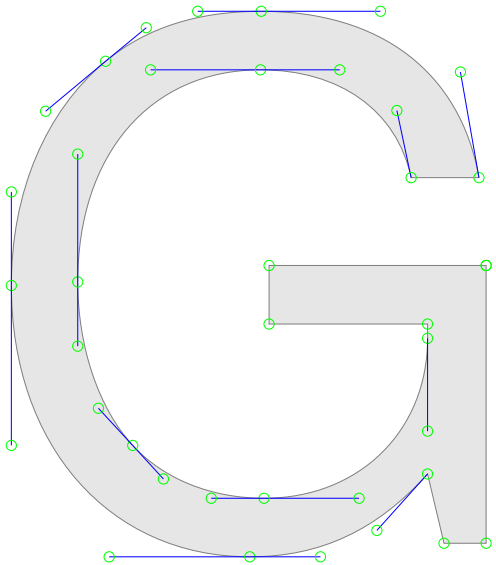
(zitiert nach *Schrift und Typografie* von Stefan Waidmann)

Times Roman



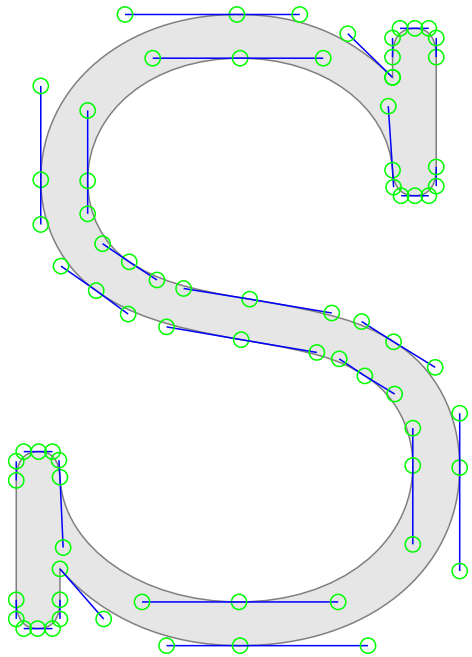
- In einer Beilage der *Times* über den Buchdruck erschien am 29. Oktober 1929 ein Artikel von Stanley Morrison unter dem Titel *Newspaper Types: A Study of The Times*, in dem die Zeitungstypografie heftig kritisiert wurde.
- Das nahm die Times zum Anlass, Morrison mit dem Entwurf eines neuen Schriftschnitts zu beauftragen. 1931 war Morrison mit seiner Arbeit fertig. Der Schriftschnitt wurde von Monotype (zuerst in 9 Punkt) implementiert.
- Seit dem 3. Oktober 1932 bis heute (abgesehen von einer kurzen Unterbrechung) verwendet die Times diese Schriftfamilie.
- Sie wurde aber zwischenzeitlich mehrfach überarbeitet.

Helvetica



- Bereits zu Beginn des 19. Jahrhunderts gab es serifenlose Schriften. Prominentes Beispiel ist die Akzidenz Grotesk, die 1898 bei der Berthold AG in Berlin erschien.
- Im 20. Jahrhundert stießen bei der Suche nach elementaren Formen und einer neuen Sachlichkeit diese Schriftschnitte auf Interesse. Das galt insbesondere für das Bauhaus (hier entwarf Paul Renner 1925 die Futura) und später in den 40-er und 50-er Jahren für Schweizer Typografen.
- Die Helvetica entstand als Überarbeitung der Akzidenz Grotesk in der Schriftgießerei Haas in Basel. Die Schrift wurde von Max Miedinger nach Vorgaben von Eduard Hoffmann entwickelt und erschien 1957.

Courier



- Courier wurde für Schreibmaschinen 1952 von Howard Kettler im Auftrag von IBM entworfen.
- Courier gewann recht rasch auch Popularität bei anderen Schreibmaschinen-Herstellern.
- Analog zu Schreibmaschinen sind alle Zeichen gleich weit.

Digitale Schriften erfüllen mehrere Aufgaben:

- Sie enthalten alle Informationen, die für einen Textformatierer (wie beispielsweise TEX oder LibreOffice) notwendig sind, um eine Textsequenz in einer konkreten Schrift korrekt zu setzen.
- Zum Zeitpunkt der Rasterung werden die Schriftschnitte benötigt.

Gelegentlich sind die Spezifikationen aufgeteilt worden – etwa bei Adobe in *AFM* (*Adobe font metrics*) und die eigentlichen Type-1-Schriftschnitte. Bei TrueType sind alle Informationen in einer Datei integriert.

Jede digitale Schrift ist eine Datenstruktur, die mindestens die folgenden Informationen liefert:

- Zeichensatz: Welche Zeichen sind in dem Schriftschnitt enthalten? Dieser kann prinzipiell beliebig umfangreich sein.
- Metriken zum gesamten Schriftschnitt und für jedes einzelne Zeichen. Wie sieht das Koordinatensystem aus? Wie werden aufeinanderfolgende Zeichen relativ zueinander positioniert?
- Schriftschnitt für jedes Zeichen.
- Abbildungen von Zeichensatzkodierungen auf Zeichen.

Typischerweise können noch sehr viel mehr Informationen enthalten sein, beispielsweise zum Kerning oder zur Verbesserung der Rasterung bei geringen Auflösungen.

- PostScript unterstützt diverse digitale Schriften wie beispielsweise Type-1 oder TrueType. Diese werden aber nur in PostScript als Objekte eingebettet und können nicht innerhalb von PostScript spezifiziert werden.
- Ein Sonderfall sind Type-3-Schriftschnitte. Diese können in PostScript selbst spezifiziert werden, müssen aber ohne die Vorteile von Type-1 oder TrueType auskommen.
- Type-3 ist somit nur eine Notlösung, die es aber erlaubt, einen ersten Blick auf digitale Repräsentierung von Schriften zu werfen.

.AGLprocessed~GS	true
.Alias	Helvetica
CharStrings	dictionary with 893 elements
Decoding	dictionary with 150 elements
Encoding	array with 256 elements
FAPI	FreeType
FID	--nostringval--
FontBBox	1075.0
FontInfo	dictionary with 10 elements
FontMatrix	[0.001 0.0 0.0 0.001 0.0 0.0]
FontName	Helvetica
FontType	1
OrigFont	dictionary with 16 elements
PaintType	0
PathLoad	%rom%Resource/Font/NimbusSans-Regular
Private	private dictionary

- Ein Schriftschnitt ist in PostScript ein assoziatives Array mit Einträgen, die vorgegebenen Konventionen entsprechen.

showfont-sorted.eps

```
% dict DisplayDict
/DisplayDict {
   6 dict begin
      /d exch def
      /keys d GetSortedKeys def
      /Helvetica findfont 12 scalefont setfont
      /x 10 def
      /htab 120 def
      /y d length 16 mul def
      keys {
         /key exch def
         /value d key get def
         x y moveto
         key ToString show
         x htab add y moveto value ToString show
         /y y 16 sub def
      } forall
   end
} def

/Helvetica findfont DisplayDict
```

showfont-sorted.eps

```
keys {
   /key exch def
   /value d key get def
   x y moveto
   key ToString show
   x htab add y moveto value ToString show
   /y y 16 sub def
} forall
```

- *DisplayDict* durchläuft mit *forall* alle Schlüssel aus dem Array *keys*, die zuvor von *GetSortedKeys* sortiert worden sind.
- Mit dem *get*-Operator wird der jeweils zugehörige Wert ermittelt.
- *ToString* konvertiert ein Objekt typabhängig in eine Zeichenkette.

Schlüssel	Typ	Bedeutung
FontType	integer	Art der Spezifikation des Schriftschnitts
FontMatrix	array	Transformations-Matrix für die Kurven innerhalb der Definitionen für die Zeichen; typisch ist ein 1000×1000-Koordinatensystem
FontInfo	dict	Array mit weiteren Feldern, die den Schriftschnitt beschreiben
Encoding	dict	bildet Werte aus dem Bereich $[0, 255]$ in Namen für die einzelnen Zeichen ab
FontBBox	array	Bounding-Box aller übereinander gezeichneter Zeichen
CharStrings	dict	spezielle Repräsentierungen der einzelnen Zeichen bei Type-1 Schriftschnitten

Der *FontType* wählt eine der drei folgenden Spezifikationsarten aus:

Type 0	Zusammengesetzter Schriftschnitt, der auf anderen Schriftschnitten basiert
Type 2	Besteht aus speziell kodierten Prozeduren für die einzelnen Zeichen, die dem *Adobe Type 1 Font Format* entsprechen
Type 3	Alle Prozeduren für die Zeichen sind reguläre PostScript-Prozeduren

Type 1 Schriftschnitte sind im Vergleich zu Type 3 kompakter, effizienter und haben optional zusätzliche Hinweise zur optimalen Darstellung in Abhängigkeit der Rasterung und der gewählten Schriftgröße
Weitere Typen sind bei Level 3 hinzugekommen wie etwa Type 42 für TrueType-Schriftschnitte.

torniello.eps

```
% Generierung eines Type-3-Fonts mit den Buchstaben
% E, T und X von Francesco Torniello:
% name Torniello font
/Torniello {
   2 dict begin
      /name exch def % unter diesem Namen wird der Font registriert
      /newfont 8 dict def
      newfont begin
         % Definition der einzelnen Eintraege ...
      end
      name newfont definefont
   end
} def

% Demo
/TorniellosFont Torniello 100 scalefont 0 0 moveto setfont (TEX) show
```

- Type-3-Schriftschnitte lassen sich mit regulären PostScript-Anweisungen erstellen.
- Im wesentlichen ist ein assoziatives Array entsprechend den Konventionen richtig zu füllen und mit dem Operator *definefont* in einen Schriftschnitt zu konvertieren.

Definition eines Type-3-Schriftschnitts

torniello.eps

```
newfont begin
   /FontType 3 def  % Font, der in PostScript definiert ist
   /FontMatrix [.001 0 0 .001 0 0] def
   /FontBBox [-30 0 950 950] def
   /Encoding 256 array def
   0 1 255 {
      Encoding exch /.notdef put
   } for
   Encoding 69 /E put
   Encoding 84 /T put
   Encoding 88 /X put
   % ... weitere Eintraege ...
end
```

- Assoziative Arrays lassen sich relativ elegant füllen in einer entsprechenden Klammerung mit *begin* und *end*.
- Die *FontMatrix* entspricht derjenigen, die mit
  `1 1000 div dup matrix scale`
  erzeugt werden würde.
- Die Operatoren *get* und *put* erlauben einen indizierten Zugriff auf Arrays, assoziative Arrays und Strings.

torniello.eps

```
% Tabelle der Weiten der einzelnen Buchstaben
/CharWidth 3 dict def
CharWidth begin
   /E 700 def
   /T 850 def
   /X 950 def
end
```

- Für jedes einzelne Zeichen ist die Weite in einer Tabelle festzulegen.
- Diese Weite gibt an, wieviel Platz für ein Zeichen reserviert wird. Das Zeichen kann weniger Platz benötigen oder auch darüber hinausragen – mit der Gefahr, dass sich dann die Darstellung mit derer anderer Zeichen überschneidet.

torniello.eps

```
% Bounding-Boxes der einzelnen Buchstaben
/CharBB 3 dict def
CharBB begin
   /E [ 100 0 700 900 ] def
   /T [ 0 0 900 950 ] def
   /X [ -30 0 950 900 ] def
end
```

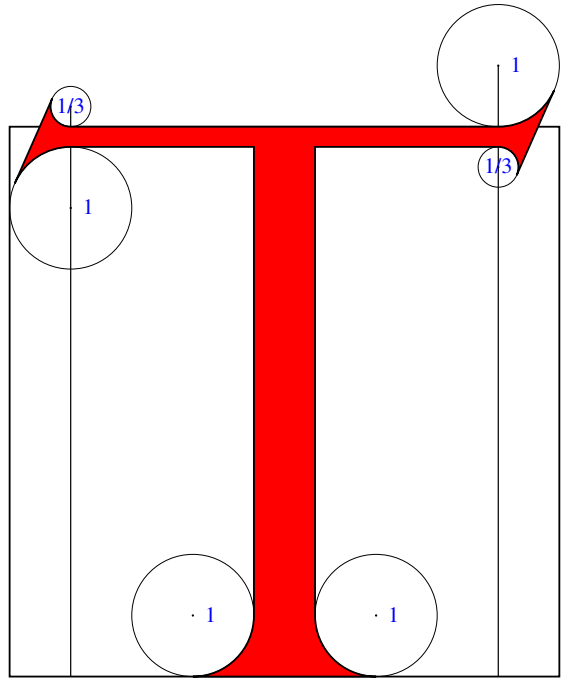
- Ferner ist es ratsam, für jedes einzelne Zeichen die Bounding-Box einzugrenzen.
- Diese Bounding-Box wird für das Caching verwendet, d.h. Teile des Zeichens, die auf diese Weise abgeschnitten werden, können mal gezeichnet werden oder auch mal entfallen.
- Je enger die Bounding-Box ist, umso effizienter kann PostScript damit arbeiten und benötigt dann auch dafür weniger Speicherplatz im Cache.

- Die Weite wird verwendet, wenn es um das Aneinanderreihen von Schriftzeichen geht (etwa mit *show*).
- Die Bounding-Box legt nur eine obere Schranke fest, in dem das Schriftzeichen liegt. Dies ist nur für das Caching relevant.
- Die roten Schachteln machen hier die Weite deutlich. Zu erkennen ist, dass das »T« und das »X« etwas über die angegebenen Weiten herausragen.

torniello.eps

```
% Zeichenprozeduren fuer die einzelnen Buchstaben
/CharProcs 4 dict def
CharProcs begin
   /.notdef {} def
   % ... die einzelnen Zeichen ...
end
```

- Das assoziative Array *CharProcs* enthält für jedes Zeichen eine Prozedur, die dieses zeichnet.



Spezifikation nach Torniello:

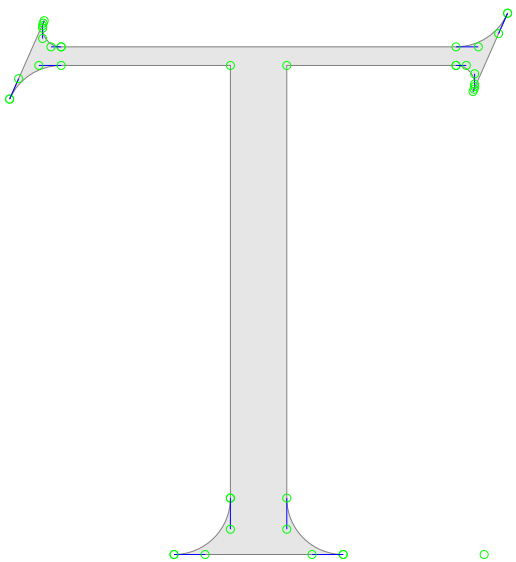
Der Buchstabe T wird aus dem Quadrat geformt. Zunächst wird der Schaft, der ein Punkt breit ist, in die Mitte des Quadrats platziert mit den Kreisen an der Grundlinie, so wie Du sie siehst. Danach beginne einen Punkt innerhalb der Ecke oben links und zeichne entlang der oberen Horizontalen, einen Punkt vor der rechten Ecke endend, und füge die Kreise wie gezeigt hinzu. Dann zeichne eine weitere horizontale Linie innerhalb des Quadrats ein Drittel eines Punktes von der zuvor gezeichneten Linie entfernt und von der gleichen Länge, und binde sie in die Kreise ein, die Du eingezeichnet siehst.

Definition eines Type-3-Schriftschnitts

torniello.eps

```
/T {
   newpath
   300 0 moveto
   600 0 lineto
   600 100 100 270 180 arcn
   500 867 lineto
   800 867 lineto
   800 834 33 90 336.1956690 arcn
   800 150 2563 sqrt mul 83 div add 79650 83 div lineto
   800 1000 100 336.1956690 270 arcn
   100 900 lineto
   100 933 33 270 156.1956690 arcn
   100 150 2563 sqrt mul 83 div sub 67011 83 div lineto
   100 767 100 156.1956690 90 arcn
   400 867 lineto
   400 100 lineto
   300 100 100 0 270 arcn
   closepath
   fill
} def
```

- Dies ist die Zeichenprozedur für das »T«.



- An dieser aus der internen Repräsentierung der Figur abgeleiteten Darstellung des »T« lässt sich erkennen, wie die Bögen in Bézier-Kurven konvertiert worden sind.
- Der isolierte Punkt ganz rechts unten ergibt sich aus einem impliziten *moveto* ganz am Ende, das sich aus der Weite ergibt.

Definition eines Type-3-Schriftschnitts

torniello.eps

```
% font charname BuildGlyph
/BuildGlyph {
   3 dict begin
      /charname exch def
      /font exch def
      /cw font /CharWidth get def
      /cbb font /CharBB get def
      cw charname get 0 % Weite
      cbb charname get aload pop % Bounding-Box
      setcachedevice
      /cp font /CharProcs get def
      cp charname known not {
         /charname /.notdef def
      } if
      cp charname get exec
   end
} bind def
```

- *BuildGlyph* wird für jedes zu zeichnende Zeichen, das noch nicht im Cache zur Verfügung steht, aufgerufen.
- *setcachedevice* sorgt dafür, dass das Zeichen simultan im Cache und in der tatsächlichen Ausgabe landet.

torniello.eps

```
/BuildChar {
   1 index /Encoding get exch get
   1 index /BuildGlyph get exec
} bind def
```

- *BuildChar* wird nur von älteren Level-1-Interpretern verwendet und kann auf Basis von *BuildGlyph* formuliert werden.

TEX

torniello.eps

```
/TorniellosFont Torniello 100 scalefont 0 0 moveto setfont (TEX) show
```

- Nach der Definition kann der Schriftschnitt sofort verwendet werden.
- Schriftschnitte des Typs 3 haben jedoch einige Schwächen (z.B. in Bezug auf die Rasterung) und erfreuen sich keiner großen Unterstützung, so dass sie immer seltener eingesetzt werden.

uphill.eps

```
/Helvetica-ISO findfont 20 scalefont setfont

20 20 moveto
60 rotate
(Es geht aufwärts!) show
```

- Die Transformations-Operatoren *rotate*, *scale* und *translate* lassen sich auch auf Zeichen aus Schriftschnitten anwenden.

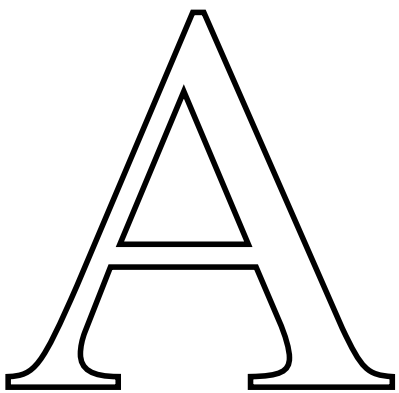
uphill.eps

```
/Helvetica-ISO /Helvetica findfont MakeISOFont pop
```

- Per Voreinstellung enthält der Encoding-Vektor der vorgegebenen Type 1 Schriftschnitte nur ASCII-Zeichen, selbst wenn der Schriftschnitt auch Zeichen aus ISO-8859-1 unterstützt.
- Der Standard-Vektor *Encoding* lässt sich aber durch den Vektor `ISOLatin1Encoding` austauschen.
- *MakeISOFont* macht genau dieses auf einer Kopie. (Letzteres ist zwingend notwendig.)
- Nicht mit einem Byte kodierbare Zeichen können über ihren Namen angesprochen werden.
- Es gibt auch Kodierungstechniken, die mit Byte-Folgen operieren können.

uphill.eps

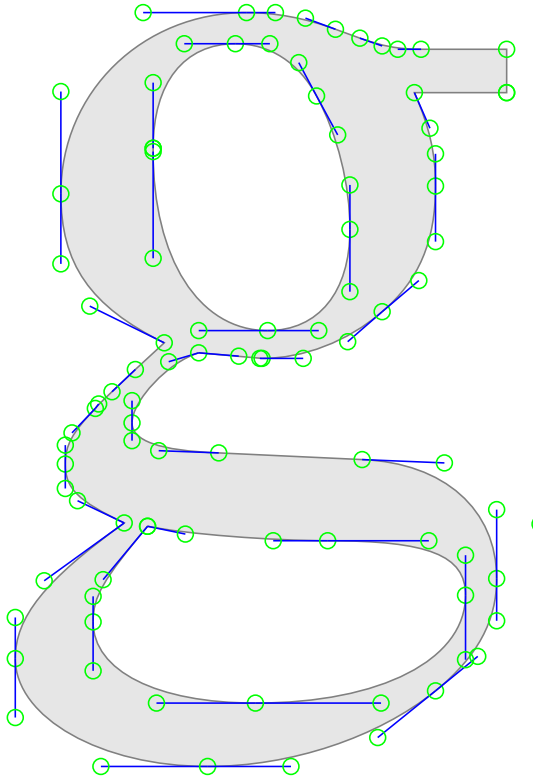
```
% name font MakeISOFont font
/MakeISOFont {
   3 dict begin
      /font exch def
      /name exch def
      /newfont font length dict def
      newfont begin
         font {
            % copy all entries with the exception of /FID
            1 index /FID ne {
               def % copy
            } {
               pop pop % discard
            } ifelse
         } forall
         /Encoding ISOLatin1Encoding def
      end
      name newfont definefont
   end
} def
```



strokedA.eps

```
/Times-Roman findfont 100 scalefont setfont
10 10 moveto
(A) false charpath stroke
```

- Der Operator *charpath* zeichnet den gegebenen String nicht, sondern ergänzt den aktuellen Pfad um die Pfade der einzelnen Zeichen in dem aktuellen Schriftschnitt.
- Der Boolean-Operator sollte gesetzt werden in Abhängigkeit von der Folge-Operation für den Pfad:
  - *false* für *stroke*
  - *true* für *fill* und *clip*



glyph-outline.eps

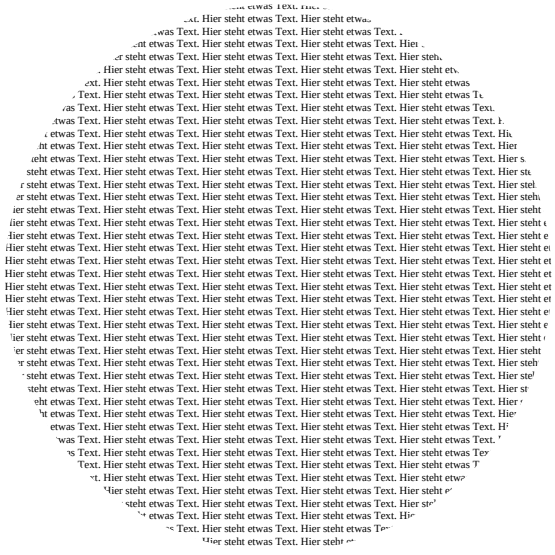
```
newpath 0 0 moveto (g) true charpath
% x y moveto -
{
    % ...
}
% x y lineto -
{
    % ...
}
% x1 y1 x2 y2 x3 y3 curveto -
{
    % ...
}
% - closepath -
{
    % ...
}
pathforall
```

- Mit *pathforall* ist es möglich, durch alle Komponenten eines Pfads durchzuiterieren.

Iteration durch einen Pfad

glyph-outline.eps

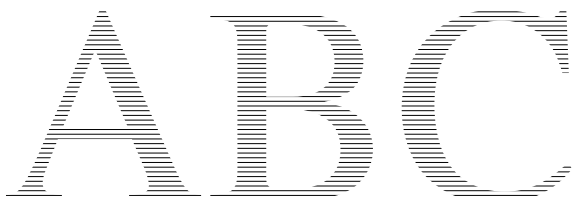
```
newpath 0 0 moveto ch true charpath
% x y moveto -
{
   /y exch def /x exch def
   x y MarkPoint x y moveto
}
% x y lineto -
{
   /y exch def /x exch def
   x y MarkPoint x y lineto
}
% x1 y1 x2 y2 x3 y3 curveto -
{
   /y3 exch def /x3 exch def
   /y2 exch def /x2 exch def
   /y1 exch def /x1 exch def
   currentpoint /y exch def /x exch def
   gsave
      newpath
      x y moveto x1 y1 lineto
      x2 y2 moveto x3 y3 lineto
      0 0 1 setrgbcolor 1 setlinewidth stroke
   grestore
   x1 y1 MarkPoint x2 y2 MarkPoint x3 y3 MarkPoint
   x1 y1 x2 y2 x3 y3 curveto
}
% - closepath -
{ closepath }
pathforall
```



circletext.eps

```
300 300 300 0 360 arc clip

/Times-Roman findfont 12 scalefont setfont
595 -14 0 {
   /y exch def
   0 y moveto
   6 { (Hier steht etwas Text. ) show } repeat
} for
```

- Mit *clip* wird der sogenannte Clipping-Pfad definiert.
- Außerhalb der Füllfläche des Clipping-Pfades wird nicht gezeichnet.
- Auf diese Weise können unerwünschte Teile einer Zeichnung weggeblendet werden.
- Der Clipping-Pfad gehört zum graphischen Zustand, der von *gsave* und *grestore* erfasst wird.



linedabc.eps

```
/Times-Roman findfont 60 scalefont setfont
0 0 moveto
(ABC) true charpath clip
clippath pathbbox
/ury exch cvi def
/urx exch cvi def
/lly exch cvi def
/llx exch cvi def

newpath
lly 1 ury {
   dup llx exch moveto
   urx exch lineto
} for
0.2 setlinewidth stroke
```

- Auch die Umrisse von Zeichen können als Clipping-Pfad verwendet werden.

linedabc.eps

```
clippath pathbbox
/ury exch cvi def
/urx exch cvi def
/lly exch cvi def
/llx exch cvi def
```

- *clippath* macht den Clipping-Pfad zum aktuellen Pfad.
- *pathbbox* liefert die Bounding-Box des aktuellen Pfades in Form der vier Zahlen, die hier allerdings Gleitkommazahlen sein können.
- Durch das Ausmessen des Schriftzuges wissen wir, in welchem Bereich es sich überhaupt lohnt, Linien zu zeichnen.

- Die erste Fassung von METAFONT wurde von Donald E. Knuth 1979 entwickelt, 1984 erschien eine revidierte, sehr viel elegantere Fassung.
- METAFONT erzeugt ein Raster mit einer einstellbaren Auflösung, das nur schwarze und weiße Pixel kennt.
- Das bedeutet, dass Farben nicht unterstützt werden und die Ausgabe von METAFONT nicht mehr skalierbar ist. Da Drucker METAFONT-Programme nicht ausführen können, muss im Vorfeld über die Auflösung entschieden werden.
- METAPOST wurde 1989-1994 von John D. Hobby entwickelt und bietet einen ähnlichen Sprachumfang wie METAFONT an, generiert aber PostScript.
- Das bedeutet, dass die Ausgabe skalierbar bleibt und direkt an einen PostScript-Drucker weitergereicht werden kann. Farben und Clipping werden unterstützt, rasterbezogene Operatoren fallen jedoch weg.

- Donald E. Knuth im METAFONTbook:

  The 'META-' part is more interesting: It indicates that we are interested in making high-level descriptions that transcend any of the individual fonts being described.
  [...]
  Meta-design is much more difficult than design; it's easier to draw something than to explain how to draw it. One of the problems is that different sets of potential specifications can't be easily be envisioned all at once. Another is that a computer has to be told absolutely everything. However, once we have successfully explained how to draw something in a sufficiently general manner, the same explanation will work for related shapes, in different circumstances; so the time spent in formulating a precise explanation turns out to be worth it.

- Der erste Namensteil META von METAFONT und METAPOST steht also dafür, dass eine Quelle eine ganze Familie von Schriftschnitten (oder Diagrammen) generieren kann.
- Im Unterschied zu PostScript sind METAFONT und METAPOST primär Quellformate für die Definition von Schriftschnitten und Zeichnungen.
- Viele PostScript-Schriftschnitte werden mit speziellen Werkzeugen entwickelt, die dann Type-1-Schriftschnitte generieren.
- Die Motivation von Donald E. Knuth für einen anderen Ansatz ergibt sich daraus, dass Schriftschnitte nicht mit akzeptablen Ergebnis auf triviale Weise skalierbar sind. Zahlreiche Parameter sind zu verändern, wenn besonders kleine oder große Schriftschnitte zu generieren sind.
- Bei PostScript gibt es eine ähnliche Möglichkeit nur über die sogenannten *hints* in den Type-1-Schriftschnitten.

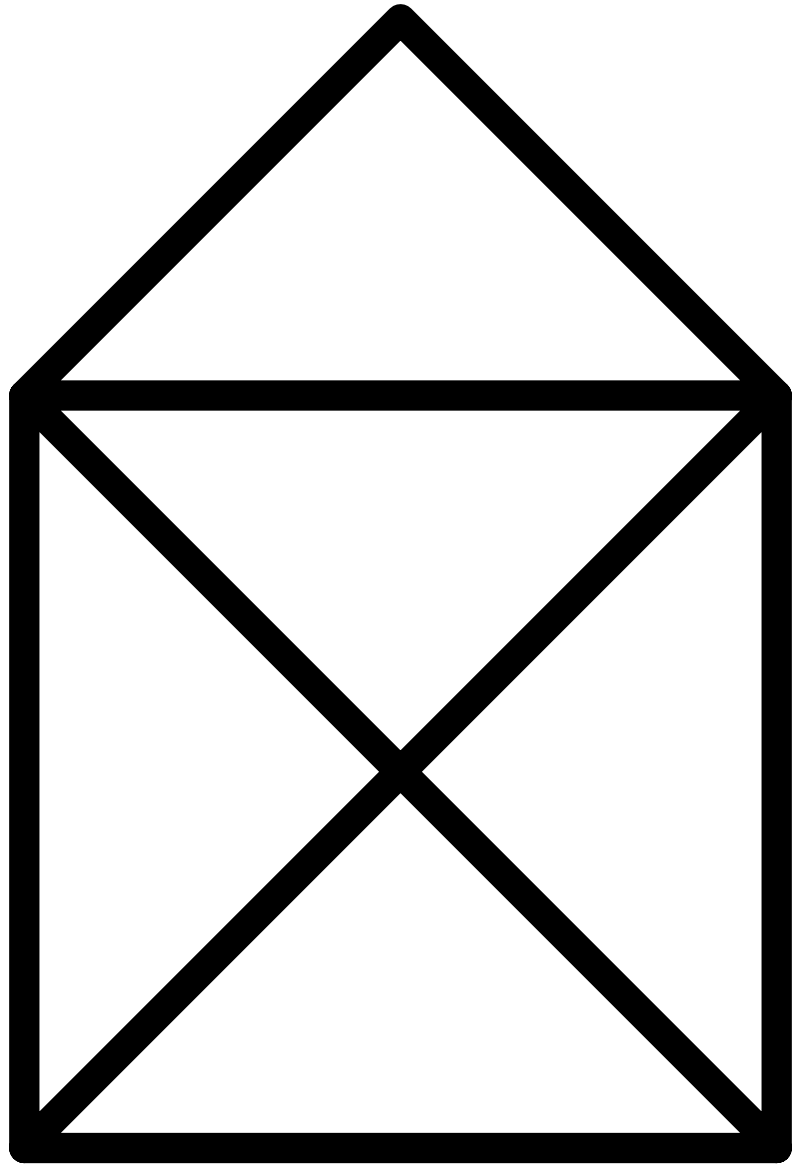
- Donald E. Knuth, *The METAFONTbook*, Addison-Wesley, 1986, ISBN 0-201-13445-4
- John D. Hobby, *A User's Manual for METAPOST*, `http://www.tug.org/docs/metapost/mpman.pdf`
- Alan Hoenig, *TEX Unbound*, Kapitel 13, *Using METAFONT and METAPOST*, 1998, ISBN 0-19-509686-X
- Michael Goossens et al, *The LATEX Graphics Companion*, Second Edition, Kapitel 3 und 4: *METAFONT and METAPOST: TEX's Mates*, *METAPOST Applications*, 2007, ISBN 0-321-50892-0

- METAFONT dient in erster Linie dazu, Schriftschnitte zu definieren.
- METAPOST unterstützt auch diesen Einsatzzweck (und erlaubte somit auch die skalierbare Einbettung von in METAFONT formulierten Schriftschnitten in PostScript).
- Im Gegensatz zu METAFONT unterstützt aber METAPOST die Verwendung von Schriften und ermöglicht damit die elegante Spezifikation von Diagrammen mit eingebetteten Texten.
- Als Vorbild diente hier auch das 1982 von Brian W. Kernighan entwickelte *pic*, das als Filter die Generierung von Diagrammen für *troff* unterstützte.
- Da METAPOST EPS-Dateien erzeugt, lässt es sich von jedem Text-System aus verwenden, das die Integration von EPS-Dateien ermöglicht.

house.mp

```
% Ein Haeusle mit 8 Strichen
prologues := 1;
beginfig(1);
   l := 1in; % Masseinheit fuer das Haeusle
   pickup pencircle scaled 3pt;
   draw (0,0) -- (0,1l) -- (0.5l,1.5l) -- (1l,1l) --
        (0,1l) -- (1l,0) -- (0,0) -- (1l,1l) -- (1l,0);
endfig;
end.
```

- Die Syntax erinnert an Pascal. Kommentare beginnen mit „%“.
- Die Zuweisung prologues := 1; sorgt dafür, dass die speziellen PostScript-Kommentare erzeugt werden, die im Einklang mit den PostScript-Strukturierungskonventionen stehen.
- Mit einem METAPOST-Programm lässt sich eine Vielzahl von EPS-Dateien erzeugen. *beginfig(1)* .. *endfig* umschließt das erste zu generierende Diagramm.
- Mit *end* wird das METAPOST-Programm beendet.



house.mp

```
l := 1in; % Masseinheit fuer das Haeussle
pickup pencircle scaled 3pt;
```

- Die Maßeinheiten entsprechen denen von PostScript. Es können aber auch diverse vordefinierte Maßeinheiten verwendet werden wie *pt* ($\frac{1}{72.27}$ Inch), *in* (Inch) oder *cm* (Zentimeter). Zusätzlich können (wie hier mit *l*) eigene Maßeinheiten definiert werden.
- Mit *pickup* wird ein neuer Zeichenstift definiert. Anders als in PostScript kann METAPOST beliebige geschlossene konvexe Pfade dafür nehmen. Hier liefert *pencircle scaled 3pt* einen kreisförmigen Zeichenstift mit einem Durchmesser von 3 Punkt. (Allerdings können nicht direkt Pfade an *pickup* übergeben werden, sie müssen noch zuvor mit der *makepen*-Funktion in einen Stift konvertiert werden. Bei *pencircle* ist dies bereits geschehen.)

house.mp

```
draw (0,0) -- (0,1l) -- (0.5l,1.5l) -- (1l,1l) --
     (0,1l) -- (1l,0) -- (0,0) -- (1l,1l) -- (1l,0);
```

- *draw* akzeptiert einen Pfad als Operanden und zeichnet diesen mit dem aktuellen Zeichenstift.
- Pfade können u.a. auch das Verbinden von Punkten mit geraden Strichen (unter Verwendung des Operators --) konstruiert werden.
- Die Idee eines aktuellen Pfades (wie bei PostScript) gibt es nicht. Stattdessen sind Pfade ganz normale Objekte, die in Variablen abgelegt werden können und auf die sich eine Reihe von Operatoren beziehen.

```
theon$ ls
house.mp
theon$ mpost house.mp
This is MetaPost, version 2.00 (TeX Live 2018) (kpathsea version 6.3.0)
(/opt/ulm/ballinrobe/texlive/texmf-dist/metapost/base/mpost.mp
(/opt/ulm/ballinrobe/texlive/texmf-dist/metapost/base/plain.mp
Preloading the plain mem file, version 1.005) ) (./house.mp [1{psfonts.map}] )
1 output file written: house.1
Transcript written on house.log.
theon$ ls
house.1    house.log  house.mp
theon$
```

- Der METAPOST-Interpreter wird mit *mpost* aufgerufen.
- Dieser wird interaktiv, falls kein *end* in der Quelle vorkommt und nicht beim Aufruf von *mpost* mit der Option „-interaction batchmode“ auf Interaktionen verzichtet wurde.
- Für jede in *beginfig*..*endfig* eingeschlossene Zeichnung wird eine Ausgabedatei erzeugt, deren Namen mit dem Basisnamen der Quelle beginnt, gefolgt von einem Punkt und der Nummer der Zeichnung. Die Datei-Endung „.eps“ wird nicht angefügt.

```
theon$ mpost error.mp
This is MetaPost, version 2.00 (TeX Live 2018) (kpathsea version 6.3.0)
(/opt/ulm/ballinrobe/texlive/texmf-dist/metapost/base/mpost.mp
(/opt/ulm/ballinrobe/texlive/texmf-dist/metapost/base/plain.mp
Preloading the plain mem file, version 1.005) ) (./error.mp
>> 0
! Improper ‘addto’.
<to be read again>
                   withpen
draw->...:also(EXPR0)else:doublepath(EXPR0)withpen
                                                  .currentpen.fi._op_
<to be read again>
                   {
--->{
     curl1}..{curl1}
l.6    draw (0) --
                   (0,1l) -- (0.5l,1.5l) -- (1l,1l) --
? X
Transcript written on error.log.
theon$
```

- Wenn *(0,0)* innerhalb der Pfadkonstruktion durch *(0)* ersetzt wird, kommt es zu obigem Fehler.

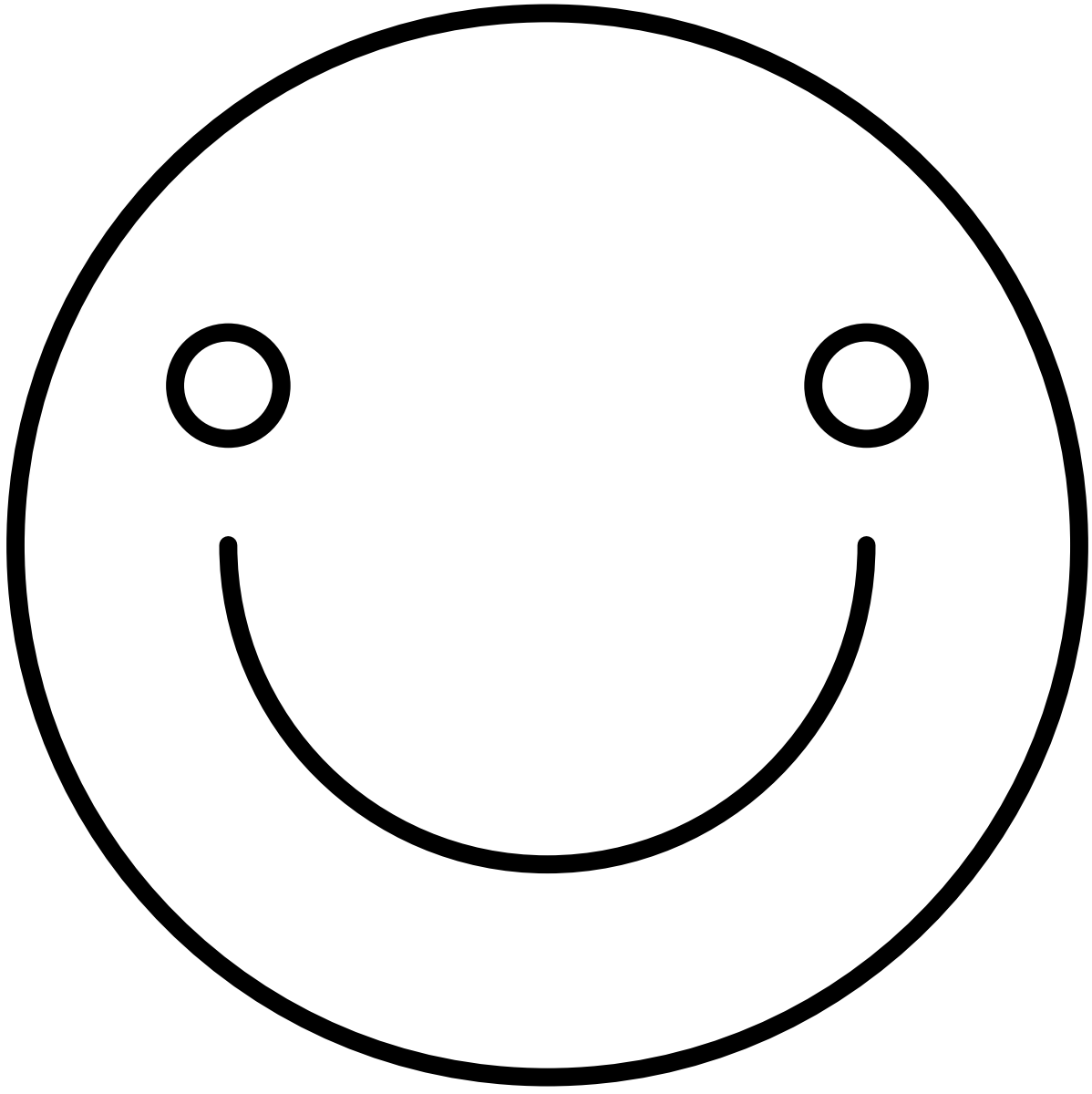
```
l.6    draw (0) --
                    (0,1l) -- (0.5l,1.5l) -- (1l,1l) --
? X
Transcript written on error.log.
theon$
```

- Die Fehlerstelle lässt sich ganz unten erkennen. „l.6“ steht für die Zeile 6 und die genaue Position lässt sich an dem Zeilenumbruch erkennen.
- METAPOST wird bei Fehlern interaktiv, es sei denn, dass dies von der Kommandozeile ausdrücklich ausgeschlossen wurde.
- Mit einer Eingabe von „X“ kann die METAPOST-Sitzung beendet werden. Ansonsten besteht die Möglichkeit, u.U. den Fehler interaktiv vorläufig zu korrigieren.

smiley.mp

```
% Ein Smiley
prologues := 1;
beginfig(1);
   l := 2in; % Radius des Gesichts
   pickup pencircle scaled 5pt;
   path circle;
   circle := (-1,0) .. (0,1) .. (1,0) .. (0,-1) .. cycle;
   draw circle scaled l; % Gesicht
   draw (-0.6l,0) .. (0,-0.6l) .. (0.6l,0); % Mund
   % linkes Auge
   draw circle scaled 0.1l shifted (-0.6l,0.3l);
   % rechtes Auge
   draw circle scaled 0.1l shifted (+0.6l,0.3l);
endfig;
end.
```

- Man beachte, dass sich *scaled 0.1l* nur auf den Pfad bezieht, nicht auf den Stift. (Bei PostScript ließ sich das nicht ohne weiteres trennen.)



smiley.mp

```
path circle;
```

- In METAFONT gibt es die 8 Datentypen boolean, string, path pen, picture, transform, pair und numeric.
- In METAPOST kommt noch der Datentyp color hinzu.
- Variablen, die nicht vom Typ numeric sind, müssen explizit mit ihrem Typnamen definiert werden.
- Variablennamen bestehen typischerweise aus Klein- und Großbuchstaben einschließlich dem Unterstrich _.
- Ziffern können nicht Bestandteil eines Variablennamens sein.
- Vordefinierte Namen dürfen nicht überdefiniert werden.
- Variablen-Deklarationen sind normalerweise global.

smiley.mp

```
path circle;
circle := (-1,0) .. (0,1) .. (1,0) .. (0,-1) .. cycle;
```

- In METAFONT/METAPOST können Pfade Variablen zugewiesen werden.
- Für Pfade gibt es zahlreiche Operatoren. In diesem Beispiel legt der Pfad-Operator .. eine Bézier-Kurve zwischen einem Pfad und einen Punkt, so dass der Pfad ohne Ecken durch die beiden Punkte verläuft und vom Aussehen her optimiert wird.
- *cycle* ist ein spezieller Operator in einem Pfad-Ausdruck, der dem *closepath*-Operator in PostScript entspricht.

smiley.mp

```
draw circle scaled 1; % Gesicht
```

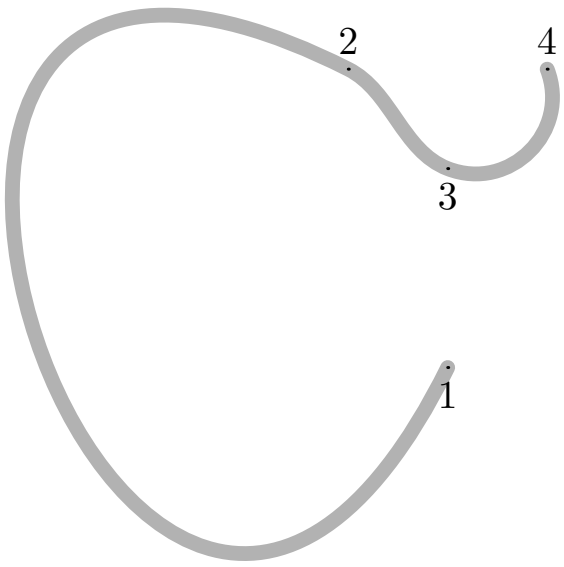
- Auf Pfade können diverse Transformations-Operatoren angewendet werden:

$$
\begin{aligned}
(x,y)\ \textit{shifted}\ (a,b) &= (x+a, y+b)\\
(x,y)\ \textit{scaled}\ s &= (sx, sy)\\
(x,y)\ \textit{xscaled}\ s &= (sx, y)\\
(x,y)\ \textit{yscaled}\ s &= (x, sy)\\
(x,y)\ \textit{slanted}\ s &= (x+sy, y)\\
(x,y)\ \textit{rotated}\ \theta &= (x\cos\theta - y\sin\theta, x\sin\theta + y\cos\theta)\\
(x,y)\ \textit{zscaled}\ (u,v) &= (xu-yv, xv+yu)
\end{aligned}
$$

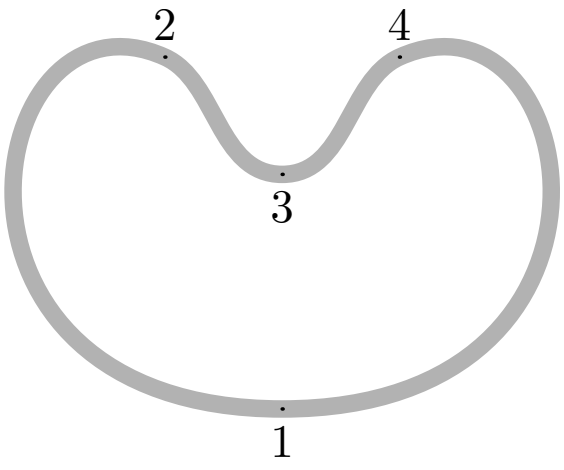
- Es gibt auch den generellen Operator *transformed*, der als rechten Operanden eine Variable oder einen Ausdruck vom Typ transform erwartet.

curves.mp

```
z1 = (0,-100); z2 = (-100,200);
z3 = (0,100); z4 = (100,200);
draw z1 .. z2 .. z3 .. z4;
```



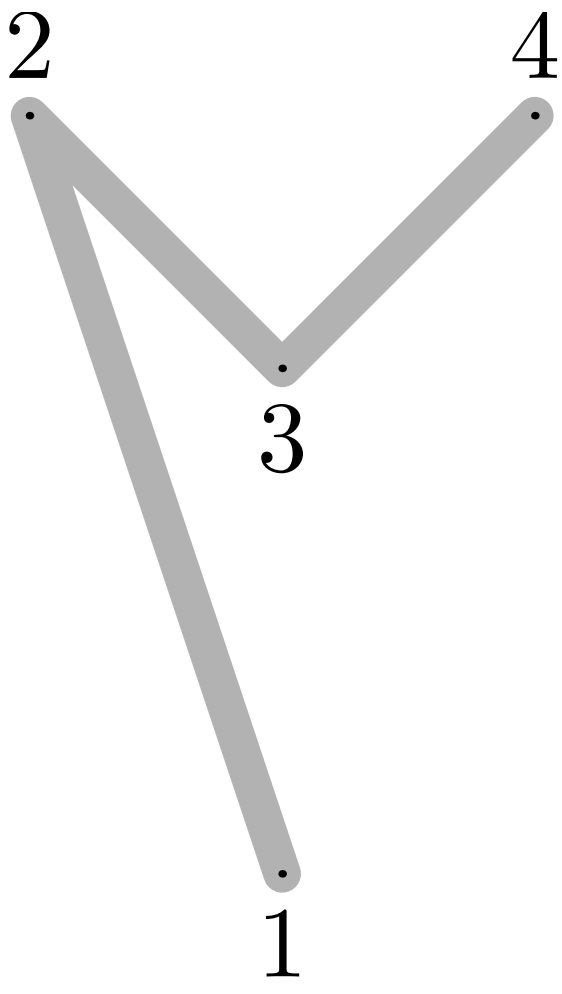
- *z* ist ein vordefiniertes Array von Punkten (Typ pair).
- *z1* ist äquivalent zu *z[1]*.
- Es gilt *z1* = *(x1,y1)*, d.h. *x* und *y* sind ebenfalls Arrays (vom Typ numeric), die mit dem Array *z* entsprechend als Aliase verknüpft sind.
- Der Operator .. konstruiert eine Bézier-Kurve, wobei darauf geachtet wird, dass aufeinanderfolgende Bézier-Kurven ohne Kanten ineinander übergehen.



curves.mp

```
z1 = (0,-100); z2 = (-100,200);
z3 = (0,100); z4 = (100,200);
draw z1 .. z2 .. z3 .. z4 .. cycle;
```

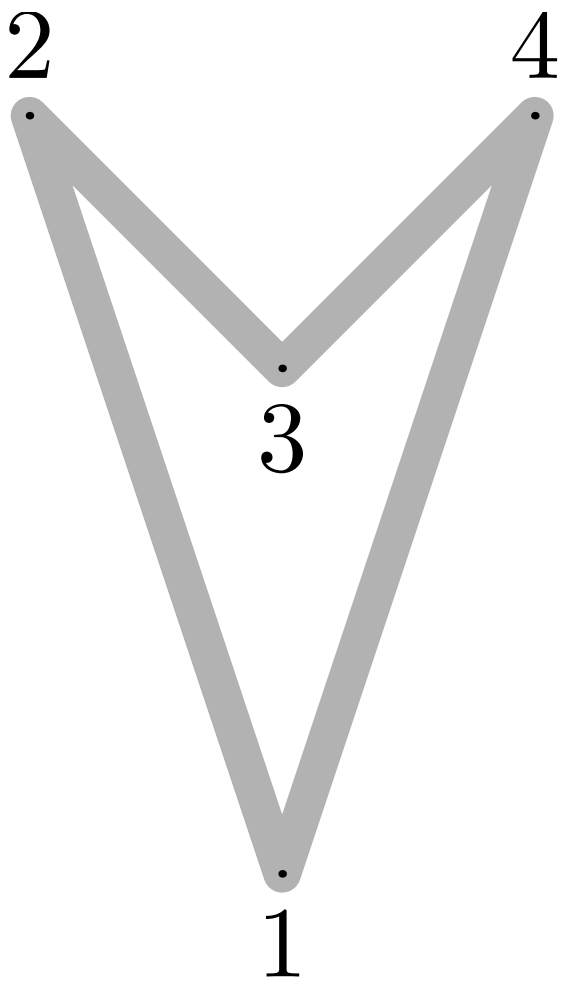
- Mit *cycle* wird der Pfad geschlossen.
- Das bedeutet in Kombination mit dem ..-Operator, dass die gesamte Kurve ohne Kanten gestaltet wird.



curves.mp

```
z1 = (0,-100); z2 = (-100,200);
z3 = (0,100); z4 = (100,200);
draw z1 -- z2 -- z3 -- z4;
```

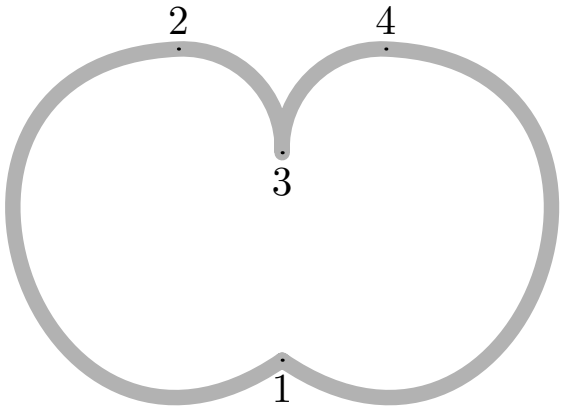
- Mit -- werden gerade Linien gezogen.



curves.mp

```
z1 = (0,-100); z2 = (-100,200);
z3 = (0,100); z4 = (100,200);
draw z1 -- z2 -- z3 -- z4 -- cycle;
```

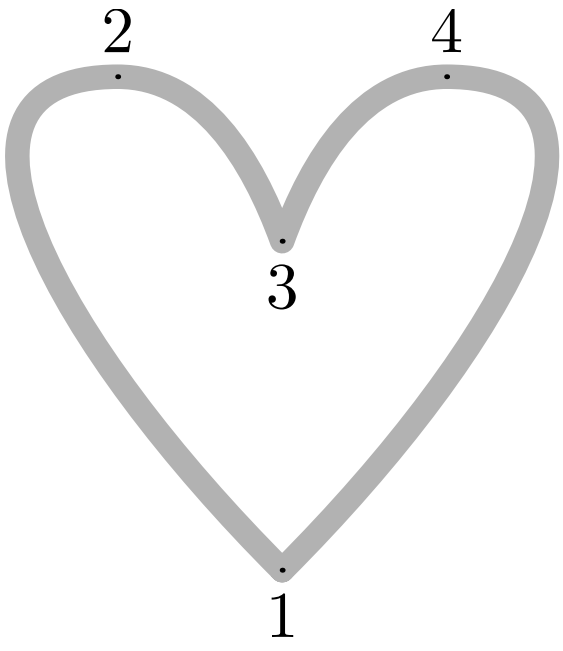
- Auch hier führt ein *cycle* dazu, dass die Kurve geschlossen wird.



curves.mp

```
z1 = (0,-100); z2 = (-100,200);
z3 = (0,100); z4 = (100,200);
draw z1 .. z2 .. z3 & z3 .. z4 .. z1;
```

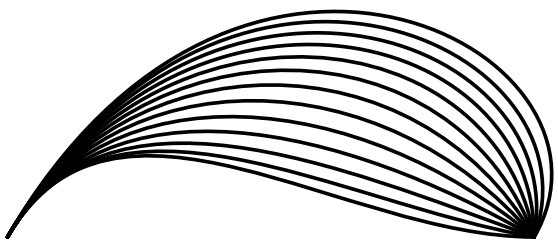
- Der Operator & verknüpft zwei Pfade miteinander, wobei keine Glättung der Kurve erzwungen wird.



curves.mp

```
z1 = (0,-100); z2 = (-100,200);
z3 = (0,100); z4 = (100,200);
draw z1{dir 135} .. z2{right} .. {dir -70}z3 &
     z3{dir 70} .. z4{right} .. z1{dir 225};
```

- Unmittelbar vor oder hinter einem Punkt kann eine Richtung angegeben werden, die dann jeweils diese Richtung in Richtung des Punktes bzw. ausgehend von dem Punkt erzwingt.
- Richtungen können als *left*, *right*, *up* and *down* spezifiziert werden. Alternativ kann die Richtung auch in Grad mit dem *dir*-Operator angegeben werden. Ferner ist es auch möglich, die Differenz zweier Punkte anzugeben.

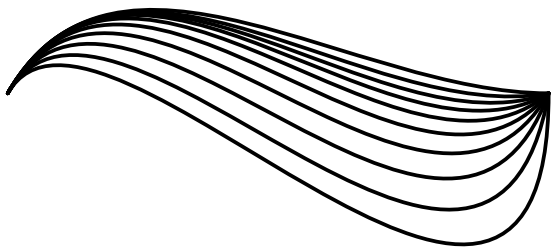


dcurves.mp

```
for d = 240 step 10 until 360:
   draw (0,0){dir 60} .. {dir d}(1in,0);
endfor
```

- Zu beachten ist hier, dass die unteren Kurven einen Wendepunkt haben, d.h. dass sie auf dem Wege von links nach rechts sich nicht immer nur nach rechts drehen, sondern ab einem Punkt in der Mitte nach links.
- Dieses und die folgenden zwei Diagramme wurden (in etwas modifizierter Form) dem METAFONTbook von Donald E. Knuth übernommen aus den Seiten 18 und 19.

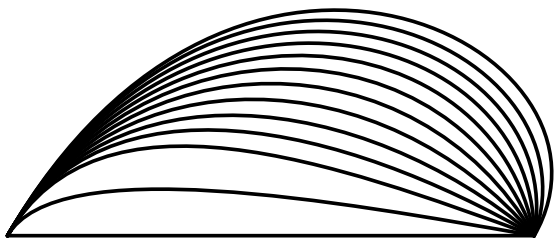
Angabe von Richtungen bei Pfaden



dcurves.mp

```
for d = 0 step 10 until 90:
   draw (0,0){dir 60} .. {dir d}(1in,0);
endfor
```

- Hier sind die Wendepunkte noch deutlicher zu sehen.



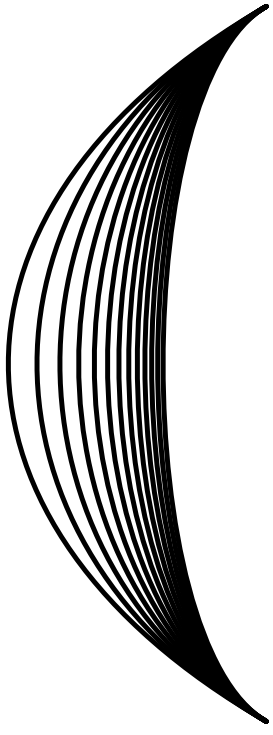
dcurves.mp

```
for d = 240 step 10 until 360:
   draw (0,0){dir 60} ... {dir d}(1in,0);
endfor
```

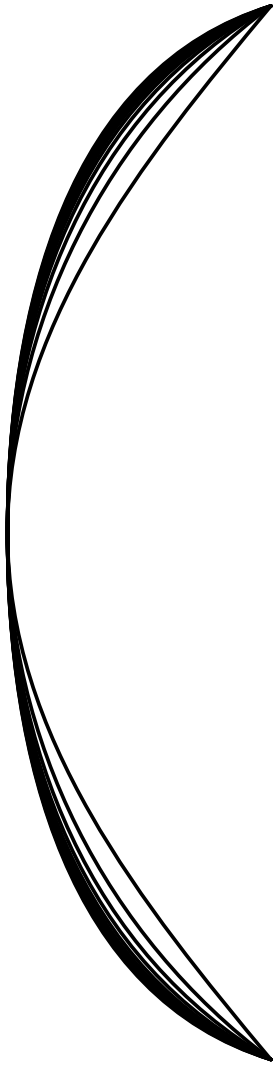
- Sei z_0 der linke Ausgangspunkt und z_1 der rechte Endpunkt.
- Dann gibt es möglicherweise einen Punkt z als Schnittpunkt der von z_0 und z_1 ausgehenden Strahlen entsprechend der angegebenen Richtungen.
- Falls es z gibt, dann liegt beim Operator „...“ die Kurve innerhalb des von z_0, z_1 und z gebildeten Dreiecks, d.h. ein Wendepunkt wird vermieden.
- Falls es kein z gibt, entspricht dieser Operator dem „..“.

tension.mp

```
for t = -2 step 1 until 10:
   draw (0,0){dir 150} .. tension (1+t/10) .. {dir 30}(0,1in);
endfor
```



- Es ist möglich, die Spannung einer Kurve anzugeben. Per Voreinstellung wird eine *tension* von 1 gewählt.
- Je höher die Spannung wird, umso weiter nach rechts wird die Kurve in diesem Beispiel gedrückt.
- Die Spannung kann auch kleiner als 1 sein, muss aber mindestens den Wert $\frac{3}{4}$ haben. In diesen Fällen darf der Bogen sich noch weiter nach links ausstrecken.
- Das bedeutet, dass die Kontrollpunkte einer Bézier-Kurve umso näher zu den Ausgangs- und Endpunkten rücken, je höher die Spannung gewählt wird.

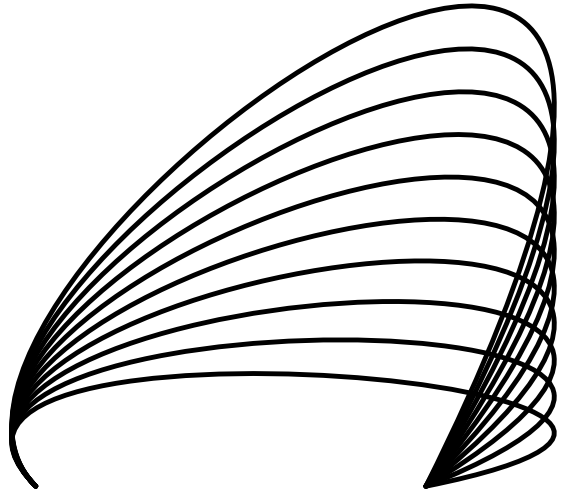


curl.mp

```
for c = 0 step 1 until 10:
   draw (0,0){curl c} .. (-0.5in,1in) .. {curl c}(0,2in);
endfor
```

- Wenn nichts anderes angegeben wird, entstehen annäherungsweise Kreisbögen.
- Mit *curl* kann an den Endpunkten ein Biegungsfaktor ausgewählt werden. Je höher dieser ist, umso steiler wird der Winkel, mit dem der Ausgangspunkt verlassen bzw. der Endpunkt erreicht wird.
- Ein Biegungsfaktor von 0 ist zulässig. In diesem Falle geht die Kurve fast direkt auf den Endpunkt zu. Per Voreinstellung liegt der Wert an den Endpunkten bei 1.
- Wenn *curl* inmitten einer Kurve angegeben wird, entsteht eine Knickstelle.

Explizite Angabe der Kontrollpunkte



controls.mp

```
for y = 10 step 10 until 100:
   draw (0,0) .. controls (-20, 20) and (90, y) .. (40, 0);
endfor
```

- Die beiden inneren Kontrollpunkte einer Bézier-Kurve können auch mit *controls* explizit angegeben werden.

`ABCDEFGHIJKLMNOPQRSTUVWXYZ_abcdefghijklmnopqrstuvwxyz`	
`<=>:|`	
`‘’`	
`+-`	
`/*\`	
`!?`	
`#&@$`	
`^~`	
`[`	
`]`	
`{}`	
`.`	Bestandteil von Gleitkommazahlen; wird ansonsten ignoriert
`, ; ( )`	stehen jeweils für sich alleine
`"`	Beginn einer Zeichenkette
`0123456789`	Numerische Konstante
`%`	Kommentare (bis zum Ende der Zeile)

- METAFONT und METAPOST konvertieren den Programmtext in eine Sequenz von Symbolen.
- Leerzeichen, Zeilentrenner, Kommentare und Punkte trennen Symbole.
- Solange aufeinanderfolgende Zeichen der gleichen Zeichenklasse angehören, werden sie zu einem Symbol zusammengefasst.
- Eine besondere Behandlung erfahren numerische Konstanten, die mit einem Punkt (gefolgt von mindestens einer Ziffer) oder mit einer Ziffer beginnen. Es werden soviele Zeichen verschlungen, wie maximal zu einer numerischen Konstante gehören können.
- Zeichenketten gehen von " bis zum darauffolgenden ".
- Beispiele:

Eingabe	Symbole
circle.radius	„circle“, „radius“
x2p	„x“, „2“, „p“
grmbl$!#@	„grmbl“, „$“, „!“, „#@“

- In METAFONT und METAPOST werden alle eingelesenen Symbole, die keine Zeichenketten oder numerischen Konstanten sind, als Namen bezeichnet.
- Alle Namen fallen in eine von zwei Kategorien: Funken (*sparks*) und Etiketten (*tags*).
- Zu den Funken gehören alle vordefinierten Operatoren und die zusätzlich definierten Makros. Vordefinierte Variablen gehören **nicht** dazu.
- Beispiele für Funken:
  beginfig draw path := + ( ) ; [
- Beispiele für Etiketten:
  c circle d l prologues x

Variablennamen

- Variablennamen bestehen aus einer Sequenz von Etiketten, bei denen auch numerische Konstanten enthalten sein können (jedoch nicht am Anfang).
- Mit einer Sequenz von Etiketten sind hierarchische Variablennamen möglich. Beispiele:
  a *a'* *a'b* *a'b::c*
- Wenn aufeinanderfolgende Etiketten der gleichen Zeichenklasse angehören, empfiehlt sich die Verwendung eines Punktes als Trenner. Beispiel: *a.b*
- Numerische Konstanten werden als Index eines Arrays interpretiert. Beispiel: *x2p* ist äquivalent zu *x[2].p* und
  a1.5b ist äquivalent zu *a[3/2]b*
- Die eckigen Klammern *[...]* sind nur notwendig, wenn für den Index ein Ausdruck verwendet wird.

- Obwohl eine Notation wie *a.b* Verbundsstrukturen nahelegt, handelt es sich dabei um Hierarchien. Das bedeutet, dass *a* nicht für das Gesamtobjekt steht. Stattdessen kann *a* vom Datentyp *numeric* sein, während *a.s* vom Datentyp *string* ist und *a.p* den Datentyp *path* hat, wenn dies zuvor so deklariert wurde: *string a.s; path a.p*
- Es ist aber möglich, einen Teil eines hierarchischen Namens als Parameter bei einem Makro oder als zu durchlaufenden Wert bei einer *forsuffixes*-Schleife anzugeben. Entsprechend können hierarchische Namen genutzt werden, um verbundartige Strukturen zu repräsentieren.

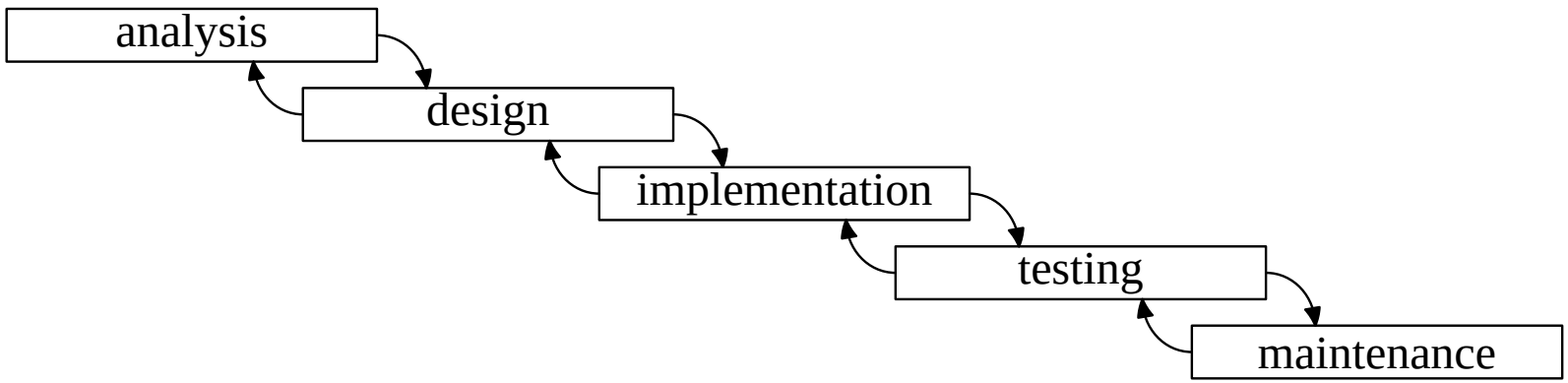
- Neben dem Zuweisungsoperator *:=* gibt es auch den Operator *=*, der nicht nur dem Vergleich dient, sondern auch die Spezifikation linearer Gleichungssysteme erlaubt.
- Beispiel:
  a = 2b+c; 7b = c/2 - a; 4a = 5c+b - 3;
  ist äquivalent zu:
  a=1.92; b=-0.12; c=2.16;
- Somit gibt es Variablen mit einem unbekannten Wert und ein System partiell aufgelöster Gleichungssysteme.

- Folgende Operationen mit unbekannten Werten sind zulässig:

  $-\ \langle \text{unknown} \rangle$
  $\langle \text{unknown} \rangle + \langle \text{unknown} \rangle$
  $\langle \text{unknown} \rangle - \langle \text{unknown} \rangle$
  $\langle \text{unknown} \rangle * \langle \text{known} \rangle$
  $\langle \text{known} \rangle * \langle \text{unknown} \rangle$
  $\langle \text{unknown} \rangle / \langle \text{known} \rangle$
  $\langle \text{known} \rangle\ [\langle \text{unknown} \rangle, \langle \text{unknown} \rangle]$
  $\langle \text{unknown} \rangle\ [\langle \text{known} \rangle, \langle \text{known} \rangle]$

- Diese Restriktionen stellen sicher, dass das Gleichungssystem linear bleibt.

Mit den vorgestellten Techniken lassen sich leicht Diagramme wie dieses Wasserfall-Modell erstellen:



- Die folgende Implementierung in METAPOST verfolgt folgende Ziele:
  - Es sollte leicht möglich sein, Phasen umzutaufen, hinzuzufügen oder wegzunehmen.
  - Alle Kästchen sollten genauso groß sein und den verwendeten Texten genügend Platz bieten.
  - Die Konfiguration, wie die Kästchen relativ zueinander platziert werden, sollte an einer zentralen Stelle erfolgen. Das gleiche gilt für die Anordnung der Pfeile.
- Wenn dies umgesetzt wird, kann nicht nur der Inhalt leicht variiert werden, sondern es können auch leicht verschiedene Parameter durchprobiert werden, um die gefälligste Variante zu ermitteln.

waterfall.mp

```
prologues := 1;
defaultfont := "ptmr8r"; % Times Roman
beginfig(1);
   % pictures containing the labels of the individual boxes
   picture stage.p[];
   % center, north/south/west/east point of the corresponding box
   pair stage.c[], stage.n[], stage.s[], stage.w[], stage.e[];
   string lbl; picture p;
   mw := 0; mh := 0; % maximal text width & height, seen so far
   stages := 0;
   % corners of our box
   x0 = x3 = -x1 = -x2 = maxwidth/2;
   y0 = y1 = -y2 = -y3 = maxheight/2;
   % configure all boxes
   % ...
   % construct our box
   maxwidth = mw * 5/4; maxheight = mh * 2;
   path box;
   box = z0 -- z1 -- z2 -- z3 -- cycle;
   % draw everything
   % ...
endfig;
end.
```

waterfall.mp

```
% corners of our box
x0 = x3 = -x1 = -x2 = maxwidth/2;
y0 = y1 = -y2 = -y3 = maxheight/2;
```

- Die Punkte *z0*, *z1*, *z2* und *z3* werden so definiert, dass sie ein Rechteck bilden.
- *maxwidth* und *maxheight* spezifizieren die Weite und die Höhe der Kästchen, sind aber zum Zeitpunkt dieser Gleichungen noch unbekannt. Sie werden erst dann bestimmt, wenn alle Beschriftungen ausgemessen worden sind.

waterfall.mp

```
% configure all boxes
for lbl = "analysis", "design", "implementation",
        "testing", "maintenance":
   stages := stages + 1;
   stage.p[stages] := p := thelabel(lbl, origin);
   % measure this label
   width := xpart(lrcorner p - llcorner p);
   if width > mw: mw := width; fi;
   height := ypart(urcorner p - lrcorner p);
   if height > mh: mh := height; fi;
   % position the corresponding box
   if stages = 1:
      stage.c[stages] = origin;
   else:
      stage.c[stages] - stage.c[stages-1] =
         (maxwidth*4/5, -maxheight*3/2);
   fi;
   % compute the connecting points of our box
   stage.n[stages] = 2/3[z0,z1] + stage.c[stages];
   stage.w[stages] = 1/2[z1,z2] + stage.c[stages];
   stage.e[stages] = 1/2[z0,z3] + stage.c[stages];
   stage.s[stages] = 2/3[z2,z3] + stage.c[stages];
endfor;
```

waterfall.mp

```
string lbl;
% ...
for lbl = "analysis", "design", "implementation",
          "testing", "maintenance":
   % ...
endfor;
```

- Die *for*-Schleife kann auch dazu verwendet werden, eine Reihe vorgegebener Werte zu durchlaufen.
- Nur an dieser Stelle wird entschieden, wieviele Kästchen gezeichnet werden, in welcher Reihenfolge sie erscheinen und welche Beschriftungen sie tragen.

waterfall.mp

```
    stage.p[stages] := p := thelabel(lbl, origin);
    % measure this label
    width := xpart(lrcorner p - llcorner p);
    if width > mw: mw := width; fi;
    height := ypart(urcorner p - lrcorner p);
    if height > mh: mh := height; fi;
```

- *thelabel* erzeugt ein Objekt des Datentyps *picture*, das eine Beschriftung enthält, die aus der angegebenen Zeichenkette in dem aktuellen Schriftschnitt (Variable *defaultfont*) erzeugt wurde.
- Der zweite Parameter spezifiziert die Position. Per Voreinstellung wird es zentriert. Aber es lässt sich auch relativ dazu positionieren, so würde *thelabel.bot* die Beschriftung unterhalb der angegebenen Position platzieren.
- Die Operatoren *lrcorner*, *llcorner*, *urcorner* und *lrcorner* liefern die Eckpunkte der Bounding-Box eines Pfades oder einer Zeichnung (Datentyp *picture*).
- Die Operatoren *xpart* und *ypart* selektieren die erste bzw. zweite Komponente eines Objekts mit dem Datentyp *pair*.

waterfall.mp

```
% position the corresponding box
if stages = 1:
   stage.c[stages] = origin;
else:
   stage.c[stages] - stage.c[stages-1] =
      (maxwidth*4/5, -maxheight*3/2);
fi;
```

- Das erste Kästchen wird um den Ursprung zentriert.
- Alle weiteren Kästchen werden jeweils relativ zum vorangegangenen Kästchen positioniert.
- Das vergrößert jeweils das Gleichungssystem, da zu diesem Zeitpunkt *maxwidth* und *maxheight* noch unbekannt sind, genauso wie alle Positionen mit Ausnahme der des ersten Kästchens.

waterfall.mp

```
% compute the connecting points of our box
stage.n[stages] = 2/3[z0,z1] + stage.c[stages];
stage.w[stages] = 1/2[z1,z2] + stage.c[stages];
stage.e[stages] = 1/2[z0,z3] + stage.c[stages];
stage.s[stages] = 2/3[z2,z3] + stage.c[stages];
```

- Der Operator $t\,[a, b]$ ist eine praktische Kurzform für $a + (b - a) * t$, die sowohl für numerische Werte als auch Objekte des Datentyps *pair* zulässig ist.
- Dieser Operator ermöglicht die elegante Spezifikation von Zwischenpunkten.
- Hier werden die Punkte festgelegt, von denen die Pfeile ausgehen und hinzeigen.
- Auch dies bereichert das Gleichungssystem, da sowohl die Punkte *z0* bis *z3* noch unbekannt sind wie auch *stage.c[stages]*.

waterfall.mp

```
maxwidth = mw * 5/4; maxheight = mh * 2;
```

- Erst durch diese beiden Gleichungen wird das gesamte Gleichungssystem lösbar.
- Erst wenn die Punkte alle bekannt sind, dürfen sie in einer Pfadkonstruktion verwendet werden:

waterfall.mp

```
path box;
box = z0 -- z1 -- z2 -- z3 -- cycle;
```

waterfall.mp

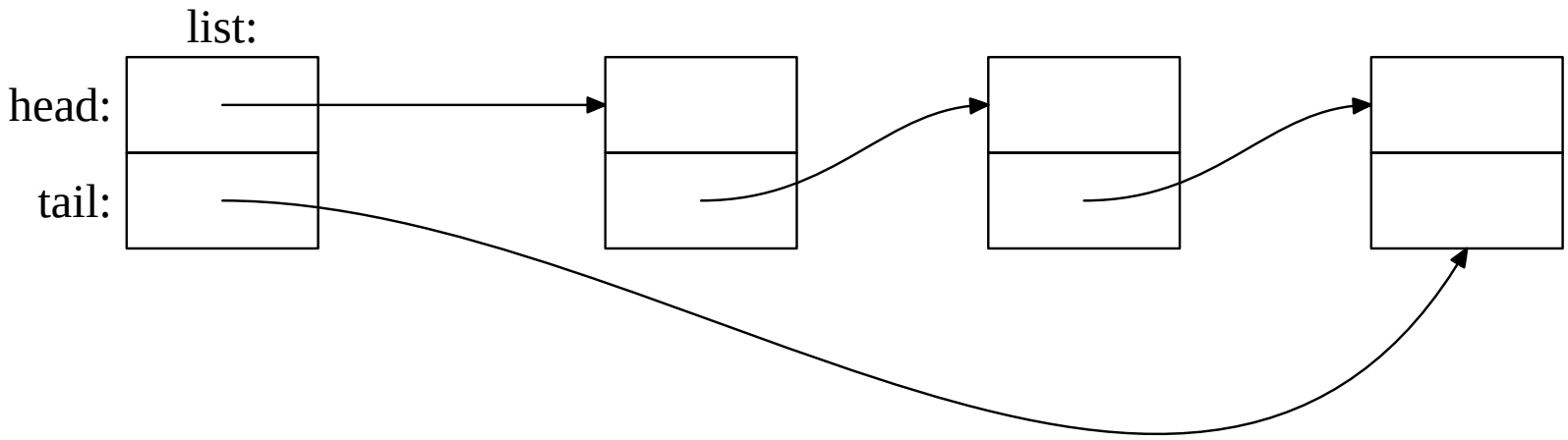
```
% draw everything
for i = 1 upto stages:
   draw box shifted stage.c[i];
   label(stage.p[i], stage.c[i]);
   if i > 1:
      drawarrow stage.e[i-1]{right} .. stage.n[i]{down};
      drawarrow stage.w[i]{left} .. stage.s[i-1]{up};
   fi;
endfor;
```

- *label* akzeptiert eine Zeichenkette oder ein bereits fertiges Objekt des Datentyps *picture* und zeichnet es an der angegebenen Position.
- *drawarrow* operiert wie *draw*, fügt aber am Ende noch einen Pfeil hinzu.

- Standardmäßig gehört zu METAPOST das *boxes*-Paket, das den Zusammenbau von Diagrammen mit Kästchen und Pfeilen entsprechend dem Vorbild von *pic* erlaubt.
- Die Idee liegt darin,
  - einzelne Kästchen (ggf. mit Beschriftungen) zu definieren,
  - die relativen Positionierungen der Kästchen zueinander mit Gleichungen zu beschreiben und
  - Pfeile, Linien, Beschriftungen und andere Figuren in Beziehung dazu zu setzen.

Beispielzeichnung mit *boxes*

Zeichnungen wie diese lassen sich leicht mit dem *boxes*-Paket spezifizieren:



list.mp

```
def node suffix $ =
   boxjoin(a.sw = b.nw; a.se = b.ne);
   forsuffixes $$ = $1, $2:
      boxit$$();
      ($$dx,$$dy) = (20pt,10pt);
   endfor;
enddef;
node list;
```

- METAFONT und METAPOST unterstützen Makrodefinitionen.
- Die Definition eines Makros besteht aus einem Namen (hier *node*), einer Parameterliste (hier *suffix $*) und einer Sequenz von Symbolen, die als Ersatztext dienen.
- Bei einer Makrodefinition wird der Ersatztext nur aufgesammelt. Er muss noch keiner Syntax genügen.
- Bei einem Makroaufruf wird eine Kopie des Ersatztextes mit ersetzten Parametern erzeugt. Diese Symbolsequenz wird dann an der Aufruf-Stelle eingefügt und erst dann parsiert.

- Bei Makros gibt es drei Parametertypen:

expr	beliebiger Ausdruck, der vor der Ersetzung ausgewertet wird
suffix	beliebiger Variablenname
text	beliebiger Text

- Der letzte Parameter kann (wie hier in diesem Beispiel) ohne Klammern spezifiziert werden. Alle vorangehenden Parameter benötigen Klammern und werden auch so spezifiziert.

list.mp

```
forsuffixes $$ = $1, $2:
   boxit$$();
   ($$dx,$$dy) = (20pt,10pt);
endfor;
```

- *forsuffixes* setzt die Schleifenvariable auf die aufgezählten Namensbestandteile, die dann entsprechend innerhalb der Schleife jeweils ersetzt werden.
- In diesem Beispiel ist *$$* die Schleifenvariable und *$* der Makroparameter.
- Entsprechend stehen *$1* und *$2* für die Variable *$* indiziert mit 1 und 2.

list.mp

```
    boxjoin(a.sw = b.nw; a.se = b.ne);
```

- *boxjoin* ist ein Makro mit einem Text-Parameter:

```
def boxjoin(text equations) =
% ...
enddef
```

- Das Makro wird implizit beim Deklarieren einer Box aufgerufen mit jeweils *a* und *b* als Namen für die letzte und die gerade aktuelle Box.
- Der Aufruf findet aber nur statt, wenn eine letzte Box existiert und diese nach dem letzten *boxjoin* deklariert wurde.

list.mp

```
boxit$$();
```

- Boxen werden mit *boxit* deklariert. Eine Beschriftung kann über den Parameter spezifiziert werden (entweder als *string* oder als *picture*).
- Dabei handelt es sich um ein Makro mit einem Suffix-Parameter:

```
vardef boxit@# (text t) =
% ...
enddef
```

- *vardef* ist ähnlich wie *def*. Das Makro wird aber nur dort erkannt, wo Variablennamen zulässig sind.
- *@#* ist ein Spezialparameter bei *vardef*, einen beliebigen folgenden Namensbestandteil schluckt und über den Namen *@#* innerhalb des Makros zugänglich macht.
- Aufeinanderfolgende Boxen werden dann entsprechend den mit *boxjoin* deklarierten Gleichungen zusammengefügt.

list.mp

```
def draw_node (text $) =
   forsuffixes $$ = $:
      drawboxed($$1, $$2);
   endfor;
enddef;
```

- Ein *text*-Parameter kann insbesondere auch eingesetzt werden, um variable lange Listen von Variablennamen zu akzeptieren, die durch Kommata getrennt sind.
- Mit einer *forsuffixes*-Schleife ist es danach möglich, durch die Liste durchzuiterieren.
- *drawboxed* ist aus dem *boxes*-Paket und funktioniert nach dem gleichen Muster. Es zeichnet die angegebenen Boxen und positioniert sie, soweit dies nicht bereits durch vorangegangene Gleichungen festgelegt ist.

list.mp

```
node list;
members := 3;
for i = 1 upto members:
   node m[i];
   if i > 1:
      xpart m[i][1].c = xpart m[i-1][1].c + 80pt;
   else:
      xpart m[i][1].c = xpart list1.c + 100pt;
   fi;
   ypart m[i][1].c = ypart list1.c;
endfor;
draw_node(list
   for i = 1 upto members:
      , m[i]
   endfor);
```

- *for*-Schleifen generieren Textersatz, der an syntaktisch passender Stelle eingebettet werden kann.

list.mp

```
label.top("list:", list1.n);
label.lft("head:", list1.w);
label.lft("tail:", list2.w);
drawarrow list1.c -- m[1][1].w;
drawarrow list2.c{right} .. {dir 60}m[members][2].s;
for i = 2 upto members:
   drawarrow m[i-1][2].c{right}
      .. tension 3/4 and 1 .. {right}m[i][1].w;
endfor;
```

- Beschriftungen können mit dem Makro *label* angebracht werden.
- Wenn ein Etikett hinter *label* angegeben wird, legt es fest, in welcher Richtung relativ zum angegebenen Punkt die Beschriftung auszurichten ist. Fehlt sie, so wird die Beschriftung um den genannten Punkt zentriert.
- Folgende Ausrichtungen werden unterstützt:

top	überhalb des Punktes
bot	unterhalb des Punktes
lft	links neben dem Punkt
rt	rechts neben dem Punkt

- Bei strukturierten Diagrammen ist es sinnvoll, Makros für einzelne Bauelemente zu definieren, so dass nur noch Makroaufrufe notwendig sind, um den Bildinhalt zu definieren.
- Die Makros sollten dabei so funktionieren, dass sie eine geeignete Positionierung automatisch berechnen können — ggf. mit der Möglichkeit, dies überzudefinieren.

Die Makros sollten so strukturiert werden, dass

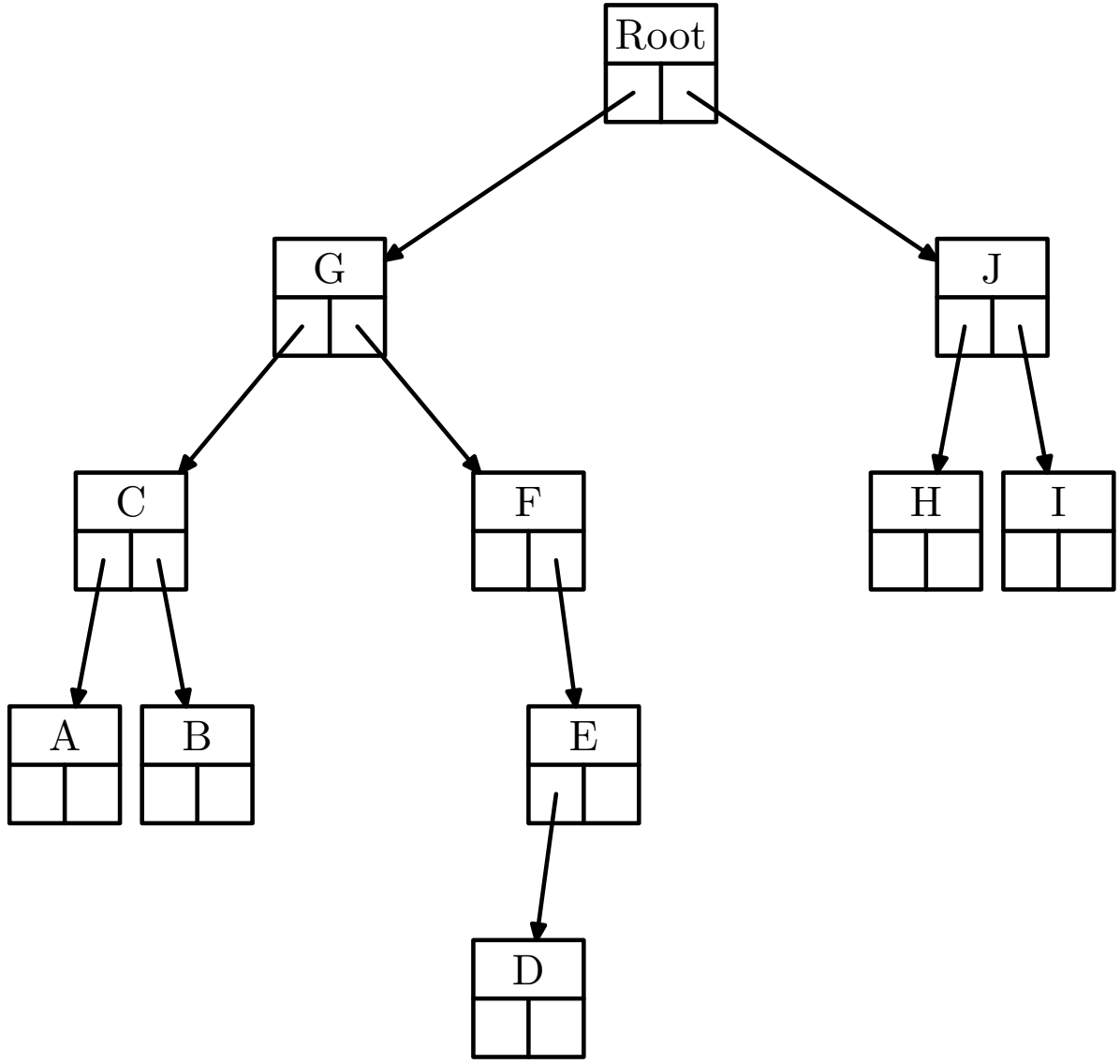
- sie beim Aufruf ein Bauelement mitsamt seinen Parametern aufnehmen, in die Datenstruktur aufnehmen und soweit möglich Gleichungen definieren und
- sie mit Hilfe von **vardef** zumindest eine Methode für das Zeichnen definieren, die nicht nur das eigene Bauelement zeichnet, sondern auch rekursiv alle benachbarten Bauelemente zeichnen lässt.

bintree.mp

```
beginfig(1);
   pickup pencircle scaled 1pt;
   Node(A, btex A etex, Nil, Nil);
   Node(B, btex B etex, Nil, Nil);
   Node(C, btex C etex, A, B);
   Node(D, btex D etex, Nil, Nil);
   Node(E, btex E etex, D, Nil);
   Node(F, btex F etex, Nil, E);
   Node(G, btex G etex, C, F);
   Node(H, btex H etex, Nil, Nil);
   Node(I, btex I etex, Nil, Nil);
   Node(J, btex J etex, H, I);
   Node(Root, btex Root etex, G, J);
   Root.Draw;
endfig;
```

- Die Idee ist, dass aus der Beschreibung eines binären Baums automatisiert eine geeignete grafische Repräsentierung gefunden wird, so dass sich keine Knoten versehentlich überlappen.

Zeichnen von binären Bäumen



- So eine Lösung wäre akzeptabel.

- Jedes Makro, das als einen der Parameter einen **suffix** erhält, kann in diesem Namensraum mit **vardef** untergeordnete Makros definieren. Diese können wie objekt-orientierte Methoden später verwendet werden.
- Alle Parameter des übergeordneten Makros stehen auch dem untergeordneten Makro zur Verfügung analog zur Sprachtechnik der *closure* in Lisp, Scheme oder Perl. (Es handelt sich implementierungstechnisch natürlich nur um Textersatz, d.h. das untergeordnete Makro wird bei jedem Aufruf des übergeordneten Makros neu erzeugt und benötigt entsprechend weiteren Speicherplatz. Da die Makroparameter selbst nicht änderbar sind, führt dies zu keinem sichtbaren Unterschied.)
- In diesem Beispiel haben Objekte die Methode *Draw* zum Zeichnen, *Width* zum Ausmessen der benötigten Weite (damit es zu keinen Überlappungen kommt) und das **boolean**-Feld *node*, das bei Knoten wahr ist und bei Nil-Objekten unwahr.

bintree.mp

```
def Node(suffix $)(expr nodelabel)(suffix leftnode, rightnode) =
   boolean $.node; $.node := true;
   % ...
   vardef $.Width =
      % ...
   enddef;
   vardef $.Draw =
      % ...
   enddef;
enddef;

boolean Nil.node;
Nil.node := false;
vardef Nil.Width = 0 enddef;
vardef Nil.Draw = enddef;
```

- Zu beachten ist hier, dass *Nil* ein singuläres Objekt ist, während es beliebige viele Inkarnationen von *Node* geben kann, alle mit unterschiedlichen Namensraumpräfixen.

Es gibt zwei prinzipielle Ansätze, alles in Zusammenhang zu bringen und zu zeichnen:

- Alle Bauelemente sind hierarchisch organisiert. Entsprechend werden beim Makroaufruf für ein übergeordnetes Bauelement explizit alle untergeordneten Bauelemente genannt. Die *Draw*-Methode muss dann rekursiv auch die untergeordneten Bauelemente zeichnen lassen.
- Die Bauelemente sind unabhängig voneinander. Zusätzlich kommen Verbindungselemente (z.B. Pfeile) dazu, die jeweils die zu verbindenden Bauelemente gegeneinander positionieren und zeichnen lassen.

Im Falle der binären Bäume ist der erste Ansatz sinnvoll. Entsprechend gibt es die Parameter *leftnode* und *rightnode* für die beiden untergeordneten Knoten, wobei jeweils auch die Angabe von *Nil* zulässig ist.

Wenn (wie in diesem Beispiel) alle Knoten gleich groß und gleichzeitig für alle Beschriftungen groß genug gestaltet werden sollen, dann ist es notwendig,

- dafür Variablen zu verwenden, die in den Gleichungssystemen aufgenommen werden, und
- gleichzeitig Variablen zu verwalten, die die jeweilige Minima, Maxima oder sonstigen Berechnungszustände nach Deklaration aller bisherigen Bauelemente repräsentieren.

Die entsprechenden Variablen aus den Gleichungssystemen können dann beim ersten Aufruf einer *Draw*-Methode oder durch ein speziell dafür geschaffenes Makro gesetzt werden.

bintree.mp

```
% maintain maximal height and width of node labels
nodemaxheight := 0;
nodemaxwidth := 0;
```

- Die Variablen *nodemaxheight* und *nodemaxwidth* verwalten die bisherigen Maxima für die Höhe und die Weite der Beschriftungen der Knoten.
- In die Gleichungssysteme gehen die Variablen *nodeheight* und *nodewidth* ein.

Wie wird alles ausgemessen?

bintree.mp

```
pair $.c, $.n, $.s, $.e, $.w, $.nw, $.ne, $.se, $.sw;
$.height = nodeheight; $.width = nodewidth;
xpart $.n = xpart $.c = xpart $.s;
ypart $.w = ypart $.c = ypart $.e;
$.c = 1/2[$.n,$.s] = 1/2[$.w,$.e];
ypart $.n - ypart $.s = $.height;
xpart $.e - xpart $.w = $.width;
ypart $.nw = ypart $.n = ypart $.ne;
ypart $.sw = ypart $.s = ypart $.se;
xpart $.nw = xpart $.w = xpart $.sw;
xpart $.ne = xpart $.e = xpart $.se;
```

- Innerhalb des *Node*-Makros verwenden die Gleichungen die zu Beginn noch unbekannten Gleichungsvariablen *nodeheight* und *nodewidth*.

Wie wird alles ausgemessen?

bintree.mp

```
% measure caption and update nodemaxheight and nodemaxwidth, if necessary
picture $.caption;
$.caption = thelabel(nodelabel, origin);
cw := xpart(lrcorner $.caption - llcorner $.caption);
ch := ypart(ulcorner $.caption - llcorner $.caption);
if cw > nodemaxwidth:
   nodemaxwidth := cw;
fi;
if ch > nodemaxheight:
   nodemaxheight := ch;
fi;
```

- Innerhalb des Makros *Node* wird die übergebene Beschriftung ausgemessen und den bisherigen Maxima verglichen.

bintree.mp

```
if not known nodeheight:
   nodeheight = 4 nodemaxheight;
fi;
if not known nodewidth:
   nodewidth = 1.2 nodemaxwidth;
fi;
if not known $.c:
   $.c = origin;
fi;
```

- Dann wird innerhalb der *Draw*-Methode festgestellt, ob die Variablen *nodeheight* und *nodewidth* aus den Gleichungssystemen bereits bekannt sind. Falls nein, werden sie in Abhängigkeit von den zuvor ausgerechneten Maxima bestimmt.

bintree.mp

```
vardef $.Width =
   1.2 $.width + leftnode.Width + rightnode.Width
enddef;
```

- Die *Width*-Methode liefert in einem rekursiven Textersatz den Ausdruck für die gesamte Weite eines Unterbaums.
- Die Rekursion endet bei *Nil*:

bintree.mp

```
vardef Nil.Width = 0 enddef;
```

bintree.mp

```
boolean $.drawn;
$.drawn := false;
vardef $.Draw =
   if not $.drawn:
      $.drawn := true;
      % ...
   fi;
enddef;
```

- Bei rekursiven Zeichenprozeduren kann es sinnvoll sein, sich gegen mehrfache Aufrufe zu schützen. In streng hierarchischen Fällen (wie diesem) könnte darauf auch verzichtet werden.

bintree.mp

```
path $.p;
$.p := $.nw -- $.ne -- $.se -- $.sw -- cycle;
draw $.p;
draw $.w -- $.e;
draw $.c -- $.s;
draw $.caption shifted 0.5[$.n,$.c];
```

- Es ist sinnvoll, den Pfad des zu zeichnenden Bauelements explizit in einer Variablen abzuspeichern. Das erlaubt danach die elegante Verwendung des Pfads in **intersectionpoint**, wenn es darum geht, Pfeile oder andere verbindende Elemente passend einzuzeichnen.

bintree.mp

```
forsuffixes $$ = leftnode, rightnode:
   if $$.node:
      ypart $$.c = ypart $.c - 2 nodeheight;
   fi;
endfor;
if leftnode.node and rightnode.node:
   xpart leftnode.c + 1/2 * (leftnode.Width + rightnode.Width) =
      xpart rightnode.c;
   xpart $.c = xpart 0.5[leftnode.c,rightnode.c];
elseif leftnode.node:
   xpart leftnode.c = xpart $.c - 1/2 nodewidth;
elseif rightnode.node:
   xpart rightnode.c = xpart $.c + 1/2 nodewidth;
fi;
leftnode.Draw; Arrow(0.5[$.c,$.sw], leftnode);
rightnode.Draw; Arrow(0.5[$.c,$.se], rightnode);
```

- Zunächst werden hier die untergeordneten Knoten passend vertikal platziert. Danach erfolgt über die *Width*-Methode eine geeignete horizontale Positionierung.

bintree.mp

```
def Arrow(expr pfrom)(suffix target) =
   if target.node:
      path dline;
      dline := pfrom -- target.c;
      pair pto;
      pto := dline intersectionpoint target.p;
      drawarrow pfrom -- pto;
   fi;
enddef;
```

- **intersectionpoint** bestimmt den Schnittpunkt zweier Pfade.
- Wenn zwei Pfade sich mehrfach kreuzen, wird der »früheste« Schnittpunkt bestimmt nach einer etwas speziellen Definition (siehe Seite 137 im METAFONT-Buch).

- »Typography exists to honor content.« (Robert Bringhurst)
- »Typografie ist keine Kunst. Typografie ist keine Wissenschaft. Typografie ist Handwerk.« (Hans Peter Willberg)
- »Typografie, das ist die Inszenierung einer Mitteilung in der Fläche, so die kürzeste Definition, die ich kenne.« (Erik Spiekermann)
- Zusammenfassung von Robert Bringhurst:
  »[...] typography should perform these services for the reader:
  - invite the reader into the text;
  - reveal the tenor and the meaning of the text;
  - clarify the structure and the order of the text;
  - link the text with other existing elements;
  - induce a state of energetic repose, which is the ideal condition for reading.«

- Die Typografie wird häufig als ein Regelwerk missverstanden, das einheitlich für alle Fälle Regeln vorgibt.
- Die Lesetypografie stellt den Leser und bestimmte Lesarten in den Vordergrund, um daraus geeignete typografische Richtlinien abzuleiten.
- Auch für wissenschaftliche Werke gibt es eine Vielzahl unterschiedlicher Lesarten und Leser.
- »Read the text before designing it.« (Robert Bringhurst)
- Als Grundlage dienen insbesondere folgende Werke:
  - Hans Peter Willberg und Friedrich Forssman, *Lesetypografie*, ISBN 3-87439-652-5
  - Robert Bringhurst, *The Elements of Typographic Style*, ISBN 0-88179-132-6

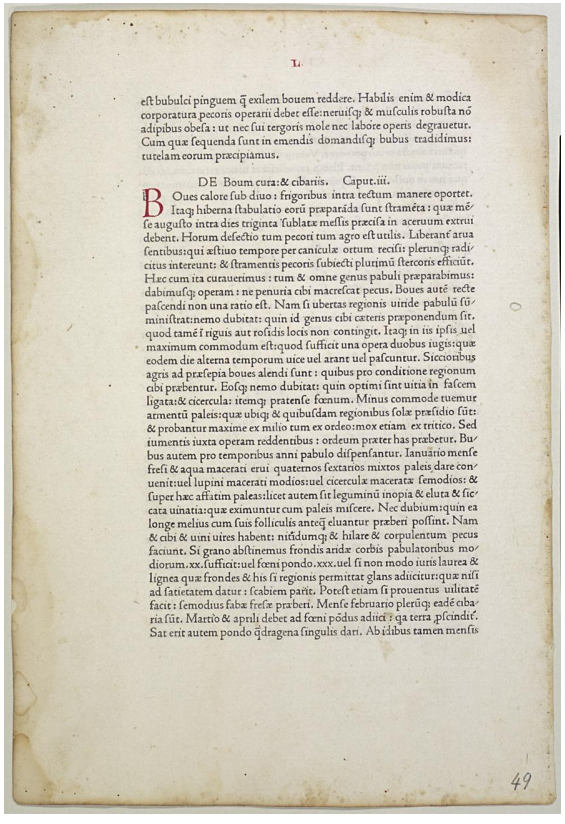
- Die Detailtypografie liefert das Regelwerk, das aus den jahrhunderte-alten Erfahrungen des Schriftsatzes entstanden ist.
- Jedoch greift in vielen Fällen das Regelwerk erst, wenn zuvor eine Konzeption aus der Lesetypografie heraus entwickelt worden ist.
- Auch typografische Regelwerke haben keinen Absolutheitsanspruch. Es kann durchaus im Ausnahmefall sinnvoll sein, gängige typografische Regeln zu brechen. Allerdings sollte das dann eine bewusst gefällte und abgewogene Entscheidung sein.
- Robert Bringhurst: »By all means break the rules, and break them beautifully, deliberately and well.«
- Als Grundlage dient u.a. folgendes Werk: Friedrich Forssman und Ralf de Jong, *Detailtypografie*, ISBN 3-87439-642-8

- Die folgende Einführung wählt einen Ansatz, bei dem ausgehend von ausgewählten Lese-Arten die zugehörige Typografie entwickelt wird.
- Dabei werden wir uns ausgehend von den Zielsetzungen Schritt für Schritt den Details nähern.

- Das *lineare Lesen* ist die klassische Art des Lesens, bei der ein Buch von vorne bis hinten sequentiell durchgelesen wird.
- Die Teile bauen Schritt für Schritt aufeinander auf. Es ist nicht vorgesehen, dass der Text überflogen oder selektiv gelesen wird.
- Der Leser soll sich voll und ganz auf den Text konzentrieren können. Dabei sollte er nicht abgelenkt werden.
- Ziel ist maximaler Lesekomfort.

- Die Augen gleiten nicht gleichmäßig über den Text hinweg.
- Stattdessen gibt es abwechselnd **Augenbewegungen** und **Fixationspausen**.
- Während einer Fixationspause kann das Auge etwa 8 bis 12 Buchstaben scharf sehen.
- Entsprechend werden nicht einzelne Buchstaben entziffert, sondern ganze Wortgebilde auf einmal erkannt.
- Nach einer Fixationspause bewegen sich die Augen weiter. Da in der Bewegungsphase die Augen nicht scharf sehen, benötigen sie genügend Orientierung (insbesondere durch Zwischenräume), so dass sie sich leicht neu positionieren können.
- Die Fixationspausen machen 93 bis 95 Prozent der Lesezeit aus.
- (Die Angaben wurden einem Artikel von Norbert Küpper entnommen: *Lesen heißt arbeiten – Das Leserverhalten wissenschaftlich betrachtet.*)

Lineares Lesen: Nicolaus Jenson, 1472, Venedig 240



li

eſt bubulci pinguem q̄ exilem bouem reddere. Habilis enim & modica corporatura pecoris operarii debet eſſe: neruiſq; & muſculis robuſta nō adipibus obeſa: ut nec ſui tergoris mole nec labore operis degrauetur. Cum quæ ſequenda ſunt in emendis domandiſq; bubus tradidimus: tutelam eorum præcipiamus.

DE Boum cura: & cibariis. Caput. iii.

BOues calore ſub diuo: frigoribus intra tectum manere oportet. Itaq; hiberna ſtabulatio eorū præparāda ſunt ſtramēta: quæ mēſe auguſto intra dies triginta ſublatæ meſſis præciſa in acerum extrui debent. Horum deſectio tum pecori tum agro eſt utilis. Liberant æſtuo ſentibus: qui æſtiuo tempore per caniculæ ortum recīſi plerunq; radicitus intereunt: & ſtramentis pecoris ſubiecti plurimū ſtercoris efficiūt. Hæc cum ita curauerimus: tum & omne genus pabuli præparabimus: dabimuſq; operam: ne penuria cibi macreſcat pecus. Boues autē recte paſcendi non una ratio eſt. Nam ſi ubertas regionis uiride pabulū ſūminiſtrat: nemo dubitat: quin id genus cibi cæteris præponendum ſit. quod tamē ſi riguis aut roſidis locis non contingit. Itaq; in iis ipſis uel maximum commodum eſt: quod ſufficit una opera duobus iugis: quæ eodem die alterna temporum uice uel arant uel paſcuntur. Siccioribus agris ad præſepia boues alendi ſunt: quibus pro conditione regionum cibi præbentur. Eoſq; nemo dubitat: quin optimi ſint uitia in faſcem ligata: & cicercula: itemq; pratenſe fœnum. Minus commode tuemur armentū paleis: quæ ubiq; & quibuſdam regionibus ſolæ præſidio ſūt: & probantur maxime ex milio tum ex ordeo: mox etiam ex tritico. Sed tumentis iuxta operam reddentibus: ordeum præter has præbetur. Bubus autem pro temporibus anni pabulo diſpenſantur. Ianuario menſe freſi & aqua macerati erui quaternos ſextarios mixtos paleis dare conuenit: uel lupini macerati modios: uel cicercula maceratæ ſemodios: & ſuper hæc affatim paleas: licet autem ſit leguminū inopia & eluta & ſiccata uinatia: quæ eximuntur cum paleis miſcere. Nec dubium: quin ea longe melius cum ſuis folliculis anteq̄ eluantur præberi poſſint. Nam & cibi & uini uires habent: niſidumq; & hilare & corpulentum pecus faciunt. Si grano abſtinemus frondis aridæ corbis pabulatoribus modiorum. xx. ſufficit: uel fœni pondo. xxx. uel ſi non modo iuris laurea & lignea quæ frondes & his ſi regionis permittat glans adiicitur: quæ niſi ad ſatietatem datur: ſcabiem parit. Poteſt etiam ſi prouentus uilitatē facit: ſemodius fabæ freſæ præberi. Menſe februario plerūq; eadē cibaria ſūt. Martio & aprili debet ad fœni pōdus adiici ? qa terra ,pſcindi. Sat erit autem pondo q̄dragena ſingulis dari. Ab idibus tamen menſis

49

- Nebenstehender Druck entstammt dem Werk *Scriptores rei rusticae*, das 1472 in Venedig von Nicolaus Jenson gedruckt worden ist.
- Nicolaus Jenson (1420–1480) war ein französischer Typograf, Schüler von Johannes Gutenberg, der ab 1470 Bücher in Venedig druckte.
- Verwendet wurde ein von Jenson selbst 1470 entwickelter Antiqua-Schriftschnitt.
- Der Text beschreibt das Landleben und stammt von mehreren römischen Autoren aus dem 2. vorchristlichen bis zum 1. nachchristlichen Jahrhundert.

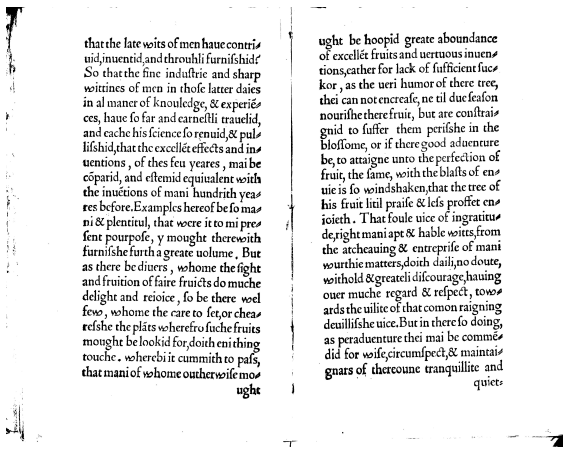
Die von Nicolaus Jenson verwendete Typografie ist zeitlos und befindet sich in Übereinstimmung mit typografischen Regeln, die bis heute für lineares Lesen gelten:

- Unaufdringlicher Schriftschnitt.
- Enger Blocksatz ohne negativ auffallende Hohlräume.
- Etwa 60 bis 70 Zeichen pro Zeile und 30 bis 40 Zeilen pro Seite.
- Absätze und Kapitel sind klar, jedoch nicht übertrieben getrennt.

that the late wits of men haue contri-
uid,inuentid,and throuhli furniſshid:
So that the fine induſtrie and sharp
wittines of men in thoſe latter daies
in al maner of knouledge, & experiē-
ces, haue ſo far and earneſtli trauelid,
and eache his ſcience ſo renuid,& pul-
liſshid,that the excellēt effects and in-
uentions , of thes feu yeares , mai be
cōparid, and eſtemid equiualent with
the inuētions of mani hundrith yea-
res before.Examples hereof be ſo ma-
ni & plentitul, that were it to mi pre-
ſent pourpoſe, y mought therewith
furniſshe furth a greate uolume . But
as there be diuers , whome the ſight
and fruition of faire fruicts do muche
delight and reioice , ſo be there wel
few , whome the care to ſet,or chea-
reſshe the plāts wherefro ſuche fruits
mought be lookid for,doith eni thing
touche . wherebi it cummith to paſs,
that mani of whome outherwiſe mo-
ught

ught be hoopid greate aboundance
of excellēt fruits and uertuous inuen-
tions,eather for lack of ſufficient ſuc-
kor , as the ueri humor of there tree,
thei can not encreaſe, ne til due ſeaſon
nouriſhe there fruit, but are conſtrai-
gnid to ſuffer them periſshe in the
bloſſome, or if there good aduenture
be, to attaigne unto the perfection of
fruit, the ſame, with the blaſts of en-
uie is ſo windshaken,that the tree of
his fruit litil praiſe & leſs proffet en-
icieth . That foule uice of ingratitu-
de,right mani apt & hable witts,from
the atcheauing & entrepriſe of mani
wurthie matters,doith daili,no doute,
withold &greateli diſcourage,hauing
ouer muche regard & reſpect , tow-
ards the uilite of that comon raigning
deuilliſshe uice.But in there ſo doing,
as peraduenture thei mai be commē-
did for wiſe,circumſpect,& maintai-
gnars of thereoune tranquillite and
quiet-

Lineares Lesen: Venedig 1551



that the late wits of men haue contri-
uid, inuentid, and throuhli furnisshid.
So that the fine industrie and sharp
wittines of men in those latter daies
in al maner of knouledge, & experie-
ces, haue so far and earnestli trauelid,
and eache his science so renuid, & pul-
lisshid, that the excellet effects and in-
uentions, of thes feu yeares, mai be
cōparid, and estemid equiualent with
the inuētions of mani hundrith yea-
res before. Examples hereof be so ma-
ni & plentitul, that were it to mi pre-
sent pourpose, y mought therewith
furnisshe furth a greate uolume. But
as there be diuers, whome the sight
and fruition of faire fruicts do muche
delight and reioice, so be there wel
few, whome the care to set, or chea-
refshe the plats wherefro suche fruits
mought be lookid for, doith eni thing
touche. wherebi it cummith to pass,
that mani of whome outherwise mo-
ught

ught be hoopid greate aboundance
of excellet fruits and uertuous inuen-
tions, eather for lack of sufficient suc-
kor, as the ueri humor of there tree,
thei can not encrease, ne til due season
nourishe there fruit, but are constrai-
gnid to suffer them perishe in the
blossome, or if there good aduenture
be, to attaigne unto the perfection of
fruit, the same, with the blasts of en-
uie is so windshaken, that the tree of
his fruit litil praise & less proffet en-
ioieth. That foule uice of ingratitu-
de, right mani apt & hable wits, from
the atcheauing & entreprise of mani
wurthie matters, doith daili, no doute,
withold & greateli discourage, hauing
ouer muche regard & respect, tow-
ards the uilite of that comon raigning
deuillishe uice. But in there so doing,
as peraduenture thei mai be comme-
did for wise, circumspect, & maintai-
gnars of thercoune tranquillite and
quiet-

- Andreas Vesalius: *Epistola, rationem modumque propinandi radicis chynae decocti pertractan*, ins Englische übersetzt von Thomas Raynalde, 1551 in Venedig erschienen
- Andreas Vesalius ist der Begründer der modernen Anatomie und beschreibt in diesem Werk die Heilwirkungen eines Öls.
- Abbild des originalen Exemplars in der Henry E. Huntington Library and Art Gallery.

- Dieses Werk wurde der damaligen typischen englischen Typografie angepasst, allerdings unter Verwendung einer Antiqua-Schrift, während in London um diese Zeit erschienene Werke üblicherweise in Fraktur gesetzt waren.
- Aufgrund des kleineren Buchformats beschränkt sich ein Textblock auf 23 Zeilen von etwa 30 Zeichen.
- Die kürzere Zeilenlänge führt zu einer höheren Zahl von notwendig werdenden Trennungen.
- Um die seitenübergreifenden Trennungen in ihrer unterbrechenden Wirkung zu begrenzen, war es in der damaligen englischen Typografie üblich, die erste Silbe der folgenden Seite am unteren Rand der vorangehenden Seite in abgesetzter Position zu wiederholen.

re, welcher sich die Astronomen im 17ten Jahrhundert bedienten, waren von einer unbequemen, und übertriebenen Länge. Auf Befehl *Ludwig XIVten* wurde von *Campani* in *Bologna* ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große *Cassini* die zwei nächsten Trabanten des Saturn entdeckte. *Auzout* in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich *Herschel* des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen,

matten Wolke einschrumpfen, in der sich keine einzelnen Sterne mehr entdecken ließen. Wenn unser Auge von der Milchstraße nur um einen Durchmesser derselben entfernt wäre, so würde sie uns nur unter einem Winkel von 60° erscheinen, nicht viel größer als das Gestirn des großen Bären; in einer Entfernung von 10 Durchmessern, würde sie nur unter einem Winkel von 2° 25 Min., ungefähr so groß wie das Siebengestirn, und auf 100 Durchmesser unter einem Winkel von 17 Min., kleiner als der berühmte Fleck in der *Andromeda* erscheinen. Sie würde in dieser Entfernung dem bloßen Auge unsichtbar seyn, und durch Fernröhre als ein Wölkchen von schwachem Licht, ähnlich den kleinen Lichtmassen dastehen, denen die Astronomen den Namen der Nebelflecke gegeben haben, und deren, wie früher erwähnt, seit *Herschel* bereits 3000 am Himmel entdeckt sind.

Die unsere Begriffe fast übersteigende Entfernung dieser unendlich weit entlegenen Weltkörper, sind wir dennoch zu berechnen im Stande, seitdem wir gelernt haben die Geschwindigkeit des Lichtes zu messen. Nicht unser Erdkörper bietet aber den Maasstab dazu dar; am Himmel selbst muß die Messung vorgenommen werden. *Olof Römer*, ein Däne, fand in der Verfinsterung der *Jupiters* Trabanten, das Mittel dieses wichtige Problem zu lösen. Er hatte in den Jahren 1670/75 mit dem älteren *Cassini* auf der Sternwarte viele Verfinsterungen der *Jupiters* Monde beobachtet, und gefunden, daß der erste Mond nicht immer zur berechneten Zeit aus dem Schatten trat, und daß der Austritt desselben sich immer mehr verspätete, je weiter sich die Erde vom *Jupiter* entfernte: wogegen der Eintritt früher erfolgte, jemehr sie sich demselben näherte, so daß der größte Unterschied 14 Min. betrug. *Römer* schloß, daß diese Ungleichheit von dem Abstande der Erde und des *Jupiters* von einander abhange, und eine Folge der verschiedenen Zeit sey, welche das Licht brauche, um bei ungleicher Entfernung die Erde zu erreichen. – Genauere Berechnungen haben später gezeigt, daß das Licht in einer Sekunde 40,000 Meilen zurücklegt; es gelangt daher von der Sonne bis zu uns, in 8 Min. 13 Sek. Dagegen braucht es vom *Syrius* 31 Jahr,

re, welcher sich die Astronomen im 17ten Jahrhundert bedienten, waren von einer unbequemen, und übertriebenen Länge. Auf Befehl *Ludwig XIVten* wurde von *Campani* in *Bologna* ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große *Cassini* die zwei nächsten Trabanten des Saturn entdeckte. *Auzout* in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich *Herschel* des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen,

2

- Nebenstehender Text ist einem Mitschrieb einer Vorlesung *Über das Universum* von Alexander von Humboldt aus den Jahren 1827/1828 an der Berliner Singakademie entnommen. Er ist über das Gutenberg-Projekt frei verfügbar und wurde von mir mit LaTeX frisch gesetzt.
- Als Schriftschnitt wurde Garamond gewählt (aus der Kollektion der frei verfügbaren Type-1-Schriftschnitte von URW++ Design und Development GmbH, Hamburg, die von Walter Schmidt für TeX angepasst worden sind).
- Die Schrift wurde in 11 Punkt gesetzt. Das Textfeld wurde so dimensioniert, dass im Blocksatz 35 Zeilen mit jeweils etwa 65 Buchstaben stehen.

re, welcher sich die Astronomen im 17ten Jahrhundert bedienten, waren von einer unbequemen, und übertriebenen Länge. Auf Befehl *Ludwig XIVten* wurde von *Campani* in *Bologna* ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große *Cassini* die zwei nächsten Trabanten des Saturn entdeckte. *Auzout* in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich *Herschel* des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen,

2

- Typografen bezeichnen mit der *Farbe* den Grauwert des gesetzten Textes.
- Es geht also nicht um die verwendete Druckfarbe, sondern um die Dichte des gesetzten Textes.
- Die Farbe sollte für lineares Lesen möglichst gleichmäßig und ohne Flecken sein. Andernfalls würde das den Leser ablenken.
- Die Farbe eines gesetzten Textes hängt ab von
  - dem ausgewählten Schriftschnitt,
  - dem Abstand zwischen den Buchstaben,
  - dem Abstand zwischen den Worten und
  - dem Abstand zwischen den Zeilen.

Lineares Lesen: Times Roman an Stelle von Garamond 248

bequemen, und übertriebenen Länge. Auf Befehl *Ludwig XIV^ten* wurde von *Campani* in *Bologna* ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große *Cassini* die zwei nächsten Trabanten des Saturn entdeckte. *Auzout* in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich *Herschel* des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen, matten Wolke einschrumpfen, in der sich keine einzelnen Sterne mehr entdecken ließen. Wenn unser Auge von

der Milchstraße nur um einen Durchmesser derselben entfernt wäre, so würde sie uns nur unter einem Winkel von 60° erscheinen, nicht viel größer als das Gestirn des großen Bären; in einer Entfernung von 10 Durchmessern, würde sie nur unter einem Winkel von 2° 25 Min., ungefähr so groß wie das Siebengestirn, und auf 100 Durchmesser unter einem Winkel von 17 Min., kleiner als der berühmte Fleck in der *Andromeda* erscheinen. Sie würde in dieser Entfernung dem bloßen Auge unsichtbar seyn, und durch Fernröhre als ein Wölkchen von schwachem Licht, ähnlich den kleinen Lichtmassen dastehen, denen die Astronomen den Namen der Nebelflecke gegeben haben, und deren, wie früher erwähnt, seit *Herschel* bereits 3000 am Himmel entdeckt sind.

Die unsere Begriffe fast übersteigende Entfernung dieser unendlich weit entlegenen Weltkörper, sind wir dennoch zu berechnen im Stande, seitdem wir gelernt haben die Geschwindigkeit des Lichtes zu messen. Nicht unser Erdkörper bietet aber den Maasstab dazu dar; am Himmel selbst muß die Messung vorgenommen werden. *Olof Römer*, ein Däne, fand in der Verfinsterung der *Jupiters* Trabanten, das Mittel dieses wichtige Problem zu lösen. Er hatte in den Jahren 1670/75 mit dem älteren *Cassini* auf der Sternwarte viele Verfinsterungen der *Jupiters* Monde beobachtet, und gefunden, daß der erste Mond nicht immer zur berechneten Zeit aus dem Schatten trat, und daß der Austritt desselben sich immer mehr verspätete, je weiter sich die Erde vom *Jupiter* entfernte: wogegen der Eintritt früher erfolgte, jemehr sie sich demselben näherte, so daß der größte Unterschied 14 Min. betrug. *Römer* schloß, daß diese Ungleichheit von dem Abstande der Erde und des *Jupiters* von einander abhange, und eine Folge der verschiedenen Zeit sey, welche das Licht brauche, um bei ungleicher Entfernung die Erde zu erreichen. – Genauere Berechnungen haben später gezeigt, daß das Licht in einer Sekunde 40,000 Meilen zurücklegt; es gelangt daher von der Sonne bis zu uns, in 8 Min. 13 Sek. Dagegen braucht es vom *Syrius* 31 Jahr, und vom entferntesten Nebelfleck mindestens 94,000 Jahr. Dies giebt eine Entfernung von 33,000 Billionen Meilen. Es folgt daraus, daß das Weltgebäude ein Alter von wenig-

Lineares Lesen: Times Roman an Stelle von Garamond 249

- Times Roman wurde für die enge mehrspaltige Zeitungstypografie entworfen.
- Im Buchformat mit breiteren Zeilen fallen zwei Dinge auf:
  - Da die Zeichen von Times Roman schmaler sind, erhöht sich die Zahl der Zeichen pro Zeile.
  - Die kleineren Buchstabenbreiten, die höheren Mittellängen und die kurzen kräftigen Serifen von Times Roman, die Halt im engen Spaltensatz geben, stiften im größeren Format etwas mehr Unruhe.
- Die Auswirkung ist noch fataler, wenn bei Abschlussarbeiten und Dissertationen DIN A4 sorglos mit über 80 Zeichen pro Zeile gefüllt wird.

Andere dieser Lichtmassen am Himmel, denen die Astronomen den Namen der Nebelflecke gegeben haben, lösen sich vor den Telescopen in einzelne helle Punkte auf, die aber wahrscheinlich nur näher zusammenstehende Systeme von Sonnen, welches Milchstraßen sind, die wenigstens um 100 ihrer Dunstmassen von uns entfernt sind. – *Herschel* hat diese entfernten Milchstraßen am ganzen Himmel aufgesucht, und es sind davon bereits über 3000 entdeckt worden.

Die aufgeklärtesten alten Philosophen vermutheten schon, daß das nie erlöschende, unbewegliche Licht der Milchstraße von unzählichen Sternen entstehen müsse, die wegen der großen Entfernung einander so nahe scheinen, daß ihr Licht zusammenfließt, und wir sie nicht unterscheiden können. Die Neueren zweifelten nicht an der Richtigkeit dieser Erklärung, obgleich sie selbst durch die stärksten Fernröhre nicht mehr einzelne Sterne entdeckten, als an andern Stellen des Himmels. - Die Fernröhre, welcher sich die Astronomen im 17ten Jahrhundert bedienten, waren von einer unbequemen, und übertriebenen Länge. Auf Befehl *Ludwig XIV*^{ten} wurde von *Campani* in *Bologna* ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große *Cassini* die zwei nächsten Trabanten des Saturn entdeckte. *Auzout* in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich *Herschel* des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen, matten Wolke einschrumpfen, in der sich keine einzelnen Sterne mehr entdecken ließen. Wenn unser Auge von der Milchstraße nur um einen Durchmesser derselben entfernt wäre, so würde sie uns nur unter einem Winkel von 60° erscheinen, nicht viel größer als das Gestirn des großen Bären; in einer Entfernung von 10 Durchmessern, würde sie nur unter einem Winkel von 2° 25 Min., ungefähr so groß wie das Siebengestirn, und auf 100 Durchmesser unter einem Winkel von 17 Min., kleiner als der berühmte Fleck in der *Andromeda* erscheinen. Sie würde in dieser Entfernung dem bloßen Auge unsichtbar seyn, und durch Fernröhre als ein Wölkchen von schwachem Licht, ähnlich den kleinen Lichtmassen dastehen, denen die Astronomen den Namen der Nebelflecke gegeben haben, und deren, wie früher erwähnt, seit *Herschel* bereits 3000 am Himmel entdeckt sind.

Die unsere Begriffe fast übersteigende Entfernung dieser unendlich weit entlegenen Weltkörper, sind wir dennoch zu berechnen im Stande, seitdem wir gelernt haben die Geschwindigkeit des Lichtes zu messen. Nicht unser Erdkörper bietet aber den Maasstab dazu dar; am Himmel selbst muß die Messung vorgenommen werden. *Olof Römer*, ein Däne, fand in der Verfinsterung der *Jupiters* Trabanten, das Mittel dieses wichtige Problem zu lösen. Er hatte in den Jahren 1670/75 mit dem älteren *Cassini* auf der Sternwarte viele Verfinsterungen der *Jupiters* Monde beobachtet, und gefunden, daß der erste Mond nicht immer zur berechneten Zeit aus dem Schatten trat, und daß der Austritt desselben sich immer mehr verspätete, je weiter sich die Erde vom *Jupiter* entfernte: wogegen der Eintritt früher erfolgte, jemehr sie sich demselben näherte, so daß der größte Unterschied 14 Min. betrug. *Römer* schloß, daß diese Ungleichheit von dem Abstande der Erde und des *Jupiters* von einander abhange, und eine Folge der verschiedenen Zeit sey, welche das Licht brauche, um bei ungleicher Entfernung die Erde zu erreichen. – Genauere Berechnungen haben später gezeigt, daß das Licht in einer Sekunde 40,000 Meilen zurücklegt; es gelangt daher von der Sonne bis zu uns, in 8 Min. 13 Sek. Dagegen braucht es vom *Syrius* 31 Jahr, und vom entferntesten Nebelfleck mindestens 94,000 Jahr. Dies giebt eine Entfernung von 33,000 Billionen Meilen. Es folgt daraus, daß das Weltgebäude ein Alter von wenigstens 24,000 Jahr hat, weil das Licht was wir heute sehen, schon vor so langer Zeit von dort ausgegossen ist. Schwindel erregend! gleich der Betrachtung, daß die zerstörendsten Revolutionen jene leuchtenden Gestirne längst vernichtet haben können, welche mit ruhiger Klarheit unsere Nächte erhellen, und daß vielleicht Generationen vergehen, ehe nur die Kunde davon zu uns gelangt.

Eine sehr merkwürdige Erscheinung am Himmel sind die veränderlichen Sterne, deren Licht entweder in beständigen Perioden ab und zunimmt, oder die nachdem sie einmal erschienen sind, auf immer verschwinden. Manche Sterne sind am Himmel verloren gegangen, manche sieht man, wo man sonst keine bemerkte. – Durch die Erscheinung eines neuen Sterns ward *Hipparch*, 125 Jahr vor Chr. Geb. zur Verfertigung eines Verzeichnisses der Fixsterne bewogen. Einer ähnlichen Erscheinung verdanken wir das von *Tycho* gemachte Verzeichnis der Sterne. Der von *Tycho* beobachtete Stern erschien 1572 plötzlich mit einem Glanz der den des *Jupiter* und *Sirius* übertraf; so daß der Stern sogar am Tage sichtbar war. Einen Monat nachher nahm sein Glanz stufenweise ab, bis zum März 1574, da er ganz verschwand.

Neuere Astronomen haben eine Menge Sterne beobachtet, die in beständigen Perioden eine Ab und Zunahme des Lichtes leiden, und sogar ganz verschwinden. Diese Perioden sind sehr verschieden; von einigen Tagen, bis zu mehreren Jahren. – Den Grund dieser Erscheinung hat man wahrscheinlich, theils in physischen Veränderungen, die auf diesen Weltkörpern vor sich gehen, theils in ihrer Umdrehung um die Axe zu suchen. Dies letzte ist, besonders bei solchen Sternen zu vermuthen, deren Lichtwechsel periodisch ist. Wenn nämlich ein Theil der Oberfläche dunkler als der andere, oder so beschaffen ist, daß er weniger Licht verbreitet, so wird der Stern uns mehr oder weniger glänzen, nachdem er uns während seiner Rotationsperiode seine helle oder dunkle Seite zukehrt. Andere Sterne die plötzlich erscheinen, und dann wieder verschwinden, erlitten vielleicht irgend eine große Revolution: es entwickelten sich bisher ruhende Kräfte, und machten seinem veralteten Daseyn ein Ende, um ihn schöner aus der Asche wieder hervorgehen zu lassen.

Doppelsterne nennt man zwei oder mehrere Sterne, die so nahe bei einander stehen, daß sie dem bloßen Auge, und selbst durch kleine Fernröhre, wie ein einzelner Stern erscheinen, durch stärkere Vergrößerungen aber aus einander gerückt werden. *Bessel* hat gezeigt, daß einige derselben sich um einen gemeinsamen Schwerpunkt drehen, sich also wohl noch nicht selbstständig haben konstituiren können. Man findet 3-4 zusammen; ja im *Sigma* des *Orion* laufen 16 Sterne um einen Schwerpunkt. Man hat bis jetzt nahe 700 (675) dieser Doppelsterne entdeckt. Merkwürdig und auffallend ist die Verschiedenheit der Farbe, welche an denselben bemerkbar. Sie erscheinen abwechselnd blau – roth – weiß doch so, daß der mittlere Stern stets ein weißes, die circulirenden Weltkörper dagegen ein buntfarbiges Licht ausstrahlen. Man hat die Vermuthung aufgestellt, daß, da besonders verlöschendes Licht farbig erscheint, diese Körper verlöschende, in einer Abnahme des Lichtprocesses begriffen seyn mögten. – Auf keinen Fall kann man ihnen ein planetarisches Licht zuschreiben; sie müssen selbstleuchtend seyn, da ein reflectirtes Licht in so unermeßlicher Ferne nicht sichtbar seyn könnte. Auch ist zu erwähnen, daß die Bewegung mancher Doppelsterne von Osten nach Westen geht; im Gegensatze unseres Systems, wo alle Bewegung von Westen nach Osten fortrückt.

Auffallend ist die Geschwindigkeit mit welcher diese mehrfachen Sonnen sich bewegen. *Bessel* hat im Schwan einen Doppelstern entdeckt, dessen Fortrücken schon nach 6 Monaten bemerkbar erschien.

Eine merkwürdige Erscheinung am südlichen Himmel sind die sogenannten Magellanischen Wolken, deren lichtgebende Dünste jeden Abend in der Nähe des Südpols sichtbar werden. Diesen entgegengesetzt sind jene räthselhaften, von Sternen entblösten schwarzen Stellen, unpoetisch Kohlensäcke (coalbags) genannt, die ich ebenfalls in der südlichen Hemisphäre mehrfach beobachtet habe. Die eine dieser Stellen erscheint in der Spitze des südlichen Kreuzes, die andere in der Eiche Carl II, nahe am Südpol. Auffallend ist, daß, die durch meteorologische Instrumente bemerkbare Veränderung der Atmosphäre, auf das sichtbar werden dieser Flecken keinen Einfluß zu haben scheint. In jenen Nächten, wenn die übrigen Gestirne im schönsten Glanze leuchten, waren die dunkeln Stellen oft nicht sichtbar, und erschienen dagegen, wenn gleich das Hygrometer andeutete, daß die Luft stark mit Dünsten angefüllt sey. – Man hat diese Erscheinung aus dem Kontraste erklären wollen, den eine minder mit Sternen besäete Stelle am Himmelsraum, gegen den besonders hellfunkelnden Glanz der südlichen Gestirne, hervorbrächte. Ich kann dieser Meinung nicht seyn, die auch die beiden Forster nicht theilen, welche Coock's 2te Erdumseglung begleitend, dieser Erscheinung eine vorzügliche Aufmerksamkeit gewidmet haben. – Im *Scorpion* befindet sich ein Raum von 3°, auf dem

Lineares Lesen: Times Roman zweispaltig

Billionen Meilen. Es folgt daraus, daß das Weltgebäude ein Alter von wenigstens 24,000 Jahr hat, weil das Licht was wir heute sehen, schon vor so langer Zeit von dort ausgegossen ist. Schwindel erregend! gleich der Betrachtung, daß die zerstörendsten Revolutionen jene leuchtenden Gestirne längst vernichtet haben können, welche mit ruhiger Klarheit unsere Nächte erhellen, und daß vielleicht Generationen vergehen, ehe nur die Kunde davon zu uns gelangt.

Eine sehr merkwürdige Erscheinung am Himmel sind die veränderlichen Sterne, deren Licht entweder in beständigen Perioden ab und zunimmt, oder die nachdem sie einmal erschienen sind, auf immer verschwinden. Manche Sterne sind am Himmel verloren gegangen, manche sieht man, wo man sonst keine bemerkte. – Durch die Erscheinung eines neuen Sterns ward *Hipparch*, 125 Jahr vor Chr. Geb. zur Verfertigung eines Verzeichnisses der Fixsterne bewogen. Einer ähnlichen Erscheinung verdanken wir das von *Tycho* gemachte Verzeichnis der Sterne. Der von *Tycho* beobachtete Stern erschien 1572 plötzlich mit einem Glanz der den des *Jupiter* und *Sirius* übertraf; so daß der Stern sogar am Tage sichtbar war. Einen Monat nachher nahm sein Glanz stufenweise ab, bis zum März 1574, da er ganz verschwand.

Neuere Astronomen haben eine Menge Sterne beobachtet, die in beständigen Perioden eine Ab und Zunahme des Lichtes leiden, und sogar ganz verschwinden. Diese Perioden sind sehr verschieden; von einigen Tagen, bis zu mehreren Jahren. – Den Grund dieser Erscheinung hat man wahrscheinlich, theils in physischen Veränderungen, die auf diesen Weltkörpern vor sich gehen, theils in ihrer Umdrehung um die Axe zu suchen. Dies letzte ist, besonders bei solchen Sternen zu vermuthen, deren Lichtwechsel periodisch ist. Wenn nämlich ein Theil der Oberfläche dunkler als der andere, oder so beschaffen ist, daß er weniger Licht verbreitet, so wird der Stern uns mehr oder weniger glänzen, nachdem er uns während seiner Rotationsperiode seine helle oder dunkle Seite zukehrt. Andere Sterne die plötzlich erscheinen, und dann wieder verschwinden, erlitten vielleicht irgend eine große Revolution: es entwickelten sich bisher ruhende Kräfte, und machten seinem veralteten Daseyn ein Ende, um ihn schöner aus der Asche wieder hervorgehen zu lassen.

Doppelsterne nennt man zwei oder mehrere Sterne, die so nahe bei einander stehen, daß sie dem bloßen Auge, und selbst durch kleine Fernröhre, wie ein einzelner Stern erscheinen, durch stärkere Vergrößerungen aber aus einander gerückt werden. *Bessel* hat gezeigt, daß einige derselben sich um einen gemeinsamen Schwerpunkt drehen, sich also wohl noch nicht selbstständig haben konstituiren können. Man findet 3-4 zusammen; ja im *Sigma* des *Orion* laufen 16 Sterne um einen Schwerpunkt. Man hat bis jetzt nahe 700 (675) dieser Doppelsterne entdeckt. Merkwürdig und auffallend ist die Verschiedenheit der Farbe, welche an denselben bemerkbar. Sie erscheinen abwechselnd blau – roth – weiß doch so, daß der mittlere Stern stets ein weißes, die circulirenden Weltkörper dagegen ein buntfarbiges Licht ausstrahlen. Man hat die Vermuthung aufgestellt, daß, da besonders verlöschendes Licht farbig erscheint, diese Körper verlöschende, in einer Abnahme des Lichtprocesses begriffen seyn mögten. – Auf keinen Fall kann man ihnen ein planetarisches Licht zuschreiben; sie müssen selbstleuchtend seyn, da ein reflectirtes Licht in so unermeßlicher Ferne nicht sichtbar seyn könnte. Auch ist zu erwähnen, daß die Bewegung mancher Doppelsterne von Osten nach Westen geht; im Gegensatze unseres Systems, wo alle Bewegung von Westen nach Osten fortrückt.

Auffallend ist die Geschwindigkeit mit welcher diese mehrfachen Sonnen sich bewegen. *Bessel* hat im Schwan einen Doppelstern entdeckt, dessen Fortrücken schon nach 6 Monaten bemerkbar erschien.

Eine merkwürdige Erscheinung am südlichen Himmel sind die sogenannten Magellanischen Wolken, deren lichtgebende Dünste jeden Abend in der Nähe des Südpols sichtbar werden. Diesen entgegengesetzt sind jene räthselhaften, von Sternen entblösten schwarzen Stellen, unpoetisch Kohlensäcke (coalbags) genannt, die ich ebenfalls in der südlichen Hemisphäre mehrfach beobachtet habe. Die eine dieser Stellen erscheint in der Spitze des südlichen Kreuzes, die andere in der Eiche Carl II, nahe am Südpol. Auffallend ist, daß, die durch meteorologische Instrumente bemerkbare Veränderung der Atmosphäre, auf das sichtbar werden dieser Flecken keinen Einfluß zu haben scheint. In jenen Nächten, wenn die übrigen Gestirne im schönsten Glanze leuchten, waren die dunkeln Stellen oft nicht sichtbar, und erschienen daggegen, wenn gleich das Hygrometer andeutete, daß die Luft stark mit Dünsten angefüllt sey. – Man hat diese Erscheinung aus dem Kontraste erklären wollen, den eine minder mit Sternen besäete Stelle am Himmelsraum, gegen den besonders hellfunkelnden Glanz der südlichen Gestirne, hervorbrächte. Ich kann dieser Meinung nicht seyn, die auch die beiden Forster nicht theilen, welche Coock's 2te Erdumseglung begleitend, dieser Erscheinung eine vorzügliche Aufmerksamkeit gewidmet haben. – Im *Scorpion* befindet sich ein Raum von 3°, auf dem

- Die IEEE-Journale verwenden einen zweispaltigen Satz mit Times Roman.
- Dabei besteht jede Kolumne aus etwa 50 Zeilen, die jeweils etwa 45 Zeichen enthalten.
- Das führt zu einer erheblichen Textverdichtung, die aber dank der Vorzüge von Times Roman die Lesbarkeit nicht zu sehr beeinflusst.

- Die Abstände zwischen den Zeichen regeln den harmonischen Zusammenhalt der Wörter. Wenn dies gut geregelt ist, sind die Wörter leichter erkennbar. Die Abstände sind auch wesentlich für die Farbe des Textes.
- In jedem Schriftschnitt wird für jedes Zeichen eine Weite festgelegt. Diese ist normalerweise weiter als die Bounding-Box, da damit der normale Abstand zwischen den Zeichen festgelegt wird. Diese für jedes Zeichen individuell bestimmte Weite wird die **Zurichtung** genannt.
- Zusätzlich können für Zeichenpaare Abweichungen festgelegt werden, die auf die jeweiligen Formen Rücksicht nehmen. Diese paarweise bestimmten Abweichungen werden **Kerning** genannt.
- Unabhängig davon kann generell der Abstand zwischen den Zeichen vergrößert oder verkleinert werden. Dies ist der sogenannte **Laufweitenausgleich**.

type3.eps

```
% Tabelle der Weiten der einzelnen Buchstaben
/CharWidth 3 dict def
CharWidth begin
   /E 700 def
   /T 850 def
   /X 950 def
end
```

- Beim dritten Übungsblatt haben wir bei den Type-3-Schriftschnitten in PostScript bereits die Zurichtung individuell festgelegt gehabt.
- Die aufsummierte Zurichtung einer Zeichenkette kann in PostScript mit dem Operator `stringwidth` ermittelt werden.
- PostScript erlaubt die explizite Spezifikation des Laufweitenausgleichs, wenn `ashow` an Stelle von `show` verwendet wird. Alternativ können auch virtuelle Schriftschnitte definiert werden.
- Kleinere Schriftschnitte benötigen vergleichsweise größere Laufweiten, während die Laufweite bei größeren Schriftschnitten (ab etwa 16 Punkten) eher zu reduzieren ist.

ptmr8a.afm

```
C 65 ; WX 722 ; N A ; B 15 0 706 674 ;
C 66 ; WX 667 ; N B ; B 17 0 593 662 ;
C 67 ; WX 667 ; N C ; B 28 -14 633 676 ;
C 68 ; WX 722 ; N D ; B 16 0 685 662 ;
C 69 ; WX 611 ; N E ; B 12 0 597 662 ;
```

- Adobe Font Metrics (AFM) ist ein Textformat zur Beschreibung der Metrik eines Schriftschnitts.
- Die Spezifikation ist öffentlich unter `http://partners.adobe.com/public/developer/en/font/5004.AFM_Spec.pdf`
- Zu sehen sind hier die Zurichtungen der Versalien »A« bis »E« für *Adobe Times Roman*.
- `C` gibt die Kodierung an, `WX` spezifiziert die Zurichtung entlang der x-Achse, `N` nennt den Namen des Zeichens und mit `B` wird die Bounding-Box definiert.

- Die Hauptaufgabe von Kerning ist die Behandlung von Sonderfällen, bei dem die Zurichtung alleine zu große oder zu kleine Abstände erzeugt.
- Typisch sind Kombinationen wie »VA« oder »Te«, bei denen die beiden Buchstaben etwas näher rücken sollten, um nicht zu große Lücken entstehen zu lassen.
- Die Tradition des Kerning hängt auch von der verwendeten Sprache ab. Im Deutschen wird beispielsweise »ck« gerne näher gerückt. Im Niederländischen wird analog »ij« enger gefasst. In einem englischen Text wäre beides übertrieben.
- Auch wenn Kerning-Tabellen für Zeichenpaare spezifiziert werden, so sollte das Kerning nicht paarweise bestimmt werden. Das Ziel ist nicht die optimale Kombination, sondern die gleichmäßige Farbe der Schrift.
- Viele Schriftschnitte kommen ohne Kerning-Tabellen aus und manche führen Buchstaben übertrieben nahe. Im Zweifelsfall ist es besser, kein Kerning zu haben als ein inkonsistentes oder übertriebenes Kerning.

- Mustertext aus dem Werk *Detailtypografie*:

  Aufhalten (ja auf) Wolf? Torf Tell!; fährt

  Aufhalten (ja auf) Wolf? Torf Tell!; fährt

  Aufhalten (ja auf) Wolf? Torf Tell!; fährt

- Typische Problempunkte (aus dem gleichen Werk zitiert):
  - Berühren sich »fh«, »(j«, »f)«, »f?« und »fä«?
  - Berühren sich »f T« – trotz des Wortzwischenraums – beinahe?
  - Sind »Wo«, »To« und »Te« zu eng?

ptmr8a.afm

```
KPX A y -92
KPX A w -92
KPX A v -74
KPX A u 0
KPX A quoteright -111
KPX A quotedblright 0
KPX A p 0
KPX A Y -105
KPX A W -90
KPX A V -135
KPX A U -55
KPX A T -111
KPX A Q -55
KPX A O -55
KPX A G -40
KPX A C -40
```

- Der Ausschnitt enthält sämtliche Kerning-Spezifikationen für Adobe Times Roman, bei der das »A« links steht.
- `KPX` spezifiziert eine Kerning-Korrektur entlang der x-Achse, danach folgen die Namen der beiden Zeichen.

ptmr8a.afm

```
KPX f quoteright 55
```

- Negative Kerning-Werte führen zum Zusammenrücken, positive Werte sorgen für einen größeren Abstand.
- Hier ist ein Beispiel für eine positive Kerning-Angabe, die verhindert, dass das »f« mit dem »'« kollidiert:

‘Wf’

- Bei einer Ligatur verschmelzen zwei benachbarte Zeichen zu einem einzigen.
- Die Standard-Ligaturen sind: fi fl
- Weitere Ligaturen (seltener): ff ffi ffl
- Damit wird vermieden, dass sich zwei Zeichen so nahe zueinander kommen, dass sie sich treffen oder schwer auseinanderzuhalten sind (mit minimalen Abstand).
- Ähnlich wie beim Kerning wird sichergestellt, dass Zeichen-Kombinationen die Farbe des Textes nicht negativ beeinflussen.
- Nicht überall sind Ligaturen zulässig:

  Schilfinseln Schilfinseln

Ligaturen im AFM-Format

ptmr8a.afm

```
C 102 ; WX 333 ; N f ; B 20 0 383 683 ; L i fi ; L l fl ;
```

- Auf Ligaturen wird beim AFM-Format bei den Zurichtungstabellen verwiesen.
- Die Ligaturen selbst werden durch selbständige Zeichen repräsentiert, für die eigene Einträge existieren:

ptmr8a.afm

```
C 174 ; WX 556 ; N fi ; B 31 0 521 683 ;
C 175 ; WX 556 ; N fl ; B 32 0 521 683 ;
```

- In der Idealform des idealen Lesens sind im Text integrierte Auszeichnungen nur dann zulässig, wenn sie die Farbe des Textes nicht beeinflussen.
- Sie sind dann keine Ansprungpunkte und werden als Auszeichnung nur dann wahrgenommen, wenn das Auge sie an der entsprechenden Lesestelle wahrnimmt.
- Schriftfamilien, die für das lineare Lesen geeignet sind, bieten daher sorgfältig gestaltete kursive Schriftschnitte an, die die gleiche Farbe besitzen.
- Alternativ kommen auch Kapitälchen in Betracht, wenn sie passend entworfen worden sind und nicht etwa nur herunterskaliert wurden.

Auszeichnungen durch Kapitälchen (Times Roman) 262

bequemen, und übertriebenen Länge. Auf Befehl LUDWIG XIV^TEN wurde von CAMPANI in BOLOGNA ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große CASSINI die zwei nächsten Trabanten des Saturn entdeckte. AUZOUT in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

HERSCHEL gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich HERSCHEL des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen, matten Wolke einschrumpfen, in der sich keine einzelnen Sterne mehr entdecken ließen. Wenn unser Auge

von der Milchstraße nur um einen Durchmesser derselben entfernt wäre, so würde sie uns nur unter einem Winkel von 60° erscheinen, nicht viel größer als das Gestirn des großen Bären; in einer Entfernung von 10 Durchmessern, würde sie nur unter einem Winkel von 2° 25 Min., ungefähr so groß wie das Siebengestirn, und auf 100 Durchmesser unter einem Winkel von 17 Min., kleiner als der berühmte Fleck in der ANDROMEDA erscheinen. Sie würde in dieser Entfernung dem bloßen Auge unsichtbar seyn, und durch Fernröhre als ein Wölkchen von schwachem Licht, ähnlich den kleinen Lichtmassen dastehen, denen die Astronomen den Namen der Nebelflecke gegeben haben, und deren, wie früher erwähnt, seit HERSCHEL bereits 3000 am Himmel entdeckt sind.

Die unsere Begriffe fast übersteigende Entfernung dieser unendlich weit entlegenen Weltkörper, sind wir dennoch zu berechnen im Stande, seitdem wir gelernt haben die Geschwindigkeit des Lichtes zu messen. Nicht unser Erdkörper bietet aber den Maasstab dazu dar; am Himmel selbst muß die Messung vorgenommen werden. OLOF RÖMER, ein Däne, fand in der Verfinsterung der JUPITERS Trabanten, das Mittel dieses wichtige Problem zu lösen. Er hatte in den Jahren 1670/75 mit dem älteren CASSINI auf der Sternwarte viele Verfinsterungen der JUPITERS Monde beobachtet, und gefunden, daß der erste Mond nicht immer zur berechneten Zeit aus dem Schatten trat, und daß der Austritt desselben sich immer mehr verspätete, je weiter sich die Erde vom JUPITER entfernte: wogegen der Eintritt früher erfolgte, jemehr sie sich demselben näherte, so daß der größte Unterschied 14 Min. betrug. RÖMER schloß, daß diese Ungleichheit von dem Abstande der Erde und des JUPITERS von einander abhange, und eine Folge der verschiedenen Zeit sey, welche das Licht brauche, um bei ungleicher Entfernung die Erde zu erreichen. – Genauere Berechnungen haben später gezeigt, daß das Licht in einer Sekunde 40,000 Meilen zurücklegt; es gelangt daher von der Sonne bis zu uns, in 8 Min. 13 Sek. Dagegen braucht es vom SYRIUS 31 Jahr, und vom entferntesten Nebelfleck mindestens 94,000 Jahr. Dies giebt eine Entfernung von 33,000 Billionen Meilen. Es folgt daraus, daß das

re, welcher sich die Astronomen im 17ten Jahrhundert bedienten, waren von einer unbequemen, und übertriebenen Länge. Auf Befehl **Ludwig XIVten** wurde von **Campani** in **Bologna** ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große **Cassini** die zwei nächsten Trabanten des Saturn entdeckte. **Auzout** in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich **Herschel** des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen,

matten Wolke einschrumpfen, in der sich keine einzelnen Sterne mehr entdecken ließen. Wenn unser Auge von der Milchstraße nur um einen Durchmesser derselben entfernt wäre, so würde sie uns nur unter einem Winkel von 60° erscheinen, nicht viel größer als das Gestirn des großen Bären; in einer Entfernung von 10 Durchmessern, würde sie nur unter einem Winkel von 2° 25 Min., ungefähr so groß wie das Siebengestirn, und auf 100 Durchmesser unter einem Winkel von 17 Min., kleiner als der berühmte Fleck in der **Andromeda** erscheinen. Sie würde in dieser Entfernung dem bloßen Auge unsichtbar seyn, und durch Fernröhre als ein Wölkchen von schwachem Licht, ähnlich den kleinen Lichtmassen dastehen, denen die Astronomen den Namen der Nebelflecke gegeben haben, und deren, wie früher erwähnt, seit **Herschel** bereits 3000 am Himmel entdeckt sind.

Die unsere Begriffe fast übersteigende Entfernung dieser unendlich weit entlegenen Weltkörper, sind wir dennoch zu berechnen im Stande, seitdem wir gelernt haben die Geschwindigkeit des Lichtes zu messen. Nicht unser Erdkörper bietet aber den Maasstab dazu dar; am Himmel selbst muß die Messung vorgenommen werden. **Olof Römer**, ein Däne, fand in der Verfinsterung der **Jupiters** Trabanten, das Mittel dieses wichtige Problem zu lösen. Er hatte in den Jahren 1670/75 mit dem älteren **Cassini** auf der Sternwarte viele Verfinsterungen der **Jupiters** Monde beobachtet, und gefunden, daß der erste Mond nicht immer zur berechneten Zeit aus dem Schatten trat, und daß der Austritt desselben sich immer mehr verspätete, je weiter sich die Erde vom **Jupiter** entfernte: wogegen der Eintritt früher erfolgte, jemehr sie sich demselben näherte, so daß der größte Unterschied 14 Min. betrug. **Römer** schloß, daß diese Ungleichheit von dem Abstande der Erde und des **Jupiters** von einander abhange, und eine Folge der verschiedenen Zeit sey, welche das Licht brauche, um bei ungleicher Entfernung die Erde zu erreichen. – Genauere Berechnungen haben später gezeigt, daß das Licht in einer Sekunde 40,000 Meilen zurücklegt; es gelangt daher von der Sonne bis zu uns, in 8 Min. 13 Sek. Dagegen braucht es vom **Syrius** 31 Jahr,

- Die einzelnen Worte müssen deutlich voneinander getrennt sein, so dass sich die Augen in ihrer Bewegungsphase leicht orientieren können.
- Allerdings dürfen diese Abstände nicht so groß werden, daß der Zusammenhalt fehlt.
- Insbesondere muss deswegen der Abstand zwischen den Worten kleiner sein als der Zeilenabstand.

scheiden können. Die Neueren zweifelten nicht an der Richtigkeit dieser Erklärung, obgleich sie selbst durch die stärksten Fernröhre nicht mehr einzelne Sterne entdeckten, als an andern Stellen des Himmels. - Die Fernröhre, welcher sich die Astronomen im 17ten Jahrhundert bedienten, waren von einer unbequemen, und übertriebenen Länge. Auf Befehl *Ludwig XIVten* wurde von *Campani* in *Bologna* ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große *Cassini* die zwei nächsten Trabanten des Saturn entdeckte. *Auzout* in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich *Herschel* des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen, matten Wolke einschrumpfen, in der sich keine einzelnen Sterne mehr entdecken ließen. Wenn unser Auge von der Milchstraße nur um einen Durchmesser derselben entfernt wäre, so würde sie uns nur unter einem Winkel von 60° erscheinen, nicht viel größer als das Gestirn des großen Bären; in einer Entfernung von 10 Durchmessern, würde sie nur unter einem Winkel von 2° 25 Min., ungefähr so groß wie das Siebengestirn, und auf 100 Durchmesser unter einem Winkel von 17 Min., kleiner als der berühmte Fleck in der *Andromeda* erscheinen. Sie würde in dieser Entfernung dem bloßen Auge unsichtbar seyn, und durch Fernröhre als ein Wölkchen von schwachem Licht, ähnlich den kleinen Lichtmassen dastehen, denen die Astronomen den Namen der Nebelflecke gegeben haben, und deren, wie früher erwähnt, seit *Herschel* bereits 3000 am Himmel entdeckt sind.

Die unsere Begriffe fast übersteigende Entfernung dieser unendlich weit entlegenen Weltkörper, sind wir dennoch zu berechnen im Stande, seitdem wir gelernt haben die Geschwindigkeit des Lichtes zu messen. Nicht unser Erdkörper bietet aber den Maasstab dazu dar; am Himmel selbst muß die Messung vorgenommen werden. *Olof Römer*, ein Däne, fand in der Verfinsterung der *Jupiters* Trabanten, das Mittel dieses wichtige Problem zu lösen. Er hatte in den Jahren 1670/75 mit dem älteren *Cassini* auf der Sternwarte viele Verfinsterungen der *Jupiters* Monde

die Astronomen im 17ten Jahrhundert bedienten, waren von einer unbequemen, und übertriebenen Länge. Auf Befehl *Ludwig XIVten* wurde von *Campani* in *Bologna* ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große *Cassini* die zwei nächsten Trabanten des Saturn entdeckte. *Auzout* in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich *Herschel* des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen, matten Wolke einschrumpfen, in der

sich keine einzelnen Sterne mehr entdecken ließen. Wenn unser Auge von der Milchstraße nur um einen Durchmesser derselben entfernt wäre, so würde sie uns nur unter einem Winkel von 60° erscheinen, nicht viel größer als das Gestirn des großen Bären; in einer Entfernung von 10 Durchmessern, würde sie nur unter einem Winkel von 2° 25 Min., ungefähr so groß wie das Siebengestirn, und auf 100 Durchmesser unter einem Winkel von 17 Min., kleiner als der berühmte Fleck in der *Andromeda* erscheinen. Sie würde in dieser Entfernung dem bloßen Auge unsichtbar seyn, und durch Fernröhre als ein Wölkchen von schwachem Licht, ähnlich den kleinen Lichtmassen dastehen, denen die Astronomen den Namen der Nebelflecke gegeben haben, und deren, wie früher erwähnt, seit *Herschel* bereits 3000 am Himmel entdeckt sind.

Die unsere Begriffe fast übersteigende Entfernung dieser unendlich weit entlegenen Weltkörper, sind wir dennoch zu berechnen im Stande, seitdem wir gelernt haben die Geschwindigkeit des Lichtes zu messen. Nicht unser Erdkörper bietet aber den Maasstab dazu dar; am Himmel selbst muß die Messung vorgenommen werden. *Olof Römer*, ein Däne, fand in der Verfinsterung der *Jupiters* Trabanten, das Mittel dieses wichtige Problem zu lösen. Er hatte in den Jahren 1670/75 mit dem älteren *Cassini* auf der Sternwarte viele Verfinsterungen der *Jupiters* Monde beobachtet, und gefunden, daß der erste Mond nicht immer zur berechneten Zeit aus dem Schatten trat, und daß der Austritt desselben sich immer mehr verspätete, je weiter sich die Erde vom *Jupiter* entfernte: wogegen der Eintritt früher erfolgte, jemehr sie sich demselben näherte, so daß der größte Unterschied 14 Min. betrug. *Römer* schloß, daß diese Ungleichheit von dem Abstande der Erde und des *Jupiters* von einander abhange, und eine Folge der verschiedenen Zeit sey, welche das Licht brauche, um bei ungleicher Entfernung die Erde zu erreichen. – Genauere Berechnungen haben später gezeigt, daß das Licht in einer Sekunde 40,000 Meilen zurücklegt; es gelangt daher von der Sonne bis zu uns, in 8 Min. 13 Sek. Dagegen braucht es vom *Syrius* 31 Jahr, und vom entferntesten Nebelfleck

- Die richtige Wahl des Zeilenabstands hängt ab von
  - der gewählten Größe der Schrift,
  - der Zeilenlänge und
  - dem Schriftschnitt selbst.
- Es kommt dabei darauf an, dass
  - das Auge mühelos im Bewegungsmodus von einem Zeilenende zum folgenden Zeilenanfang springen kann und dass
  - die Farbe des Textes gleichmäßig bleibt.
- Je länger die Zeilen sind, umso größer sollte auch der Zeilenabstand gewählt werden.

fließt, und wir sie nicht unterscheiden können. Die Neueren zweifelten nicht an der Richtigkeit dieser Erklärung, obgleich sie selbst durch die stärksten Fernröhre nicht mehr einzelne Sterne entdeckten, als an andern Stellen des Himmels. - Die Fernröhre, welcher sich die Astronomen im 17ten Jahrhundert bedienten, waren von einer unbequemen, und übertriebenen Länge. Auf Befehl *Ludwig XIVten* wurde von *Campani* in *Bologna* ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große *Cassini* die zwei nächsten Trabanten des Saturn entdeckte. *Auzout* in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich *Herschel* des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen, matten Wolke einschrumpfen, in der sich keine einzelnen Sterne mehr entdecken ließen. Wenn unser Auge von der Milchstraße nur um einen Durchmesser derselben entfernt wäre, so würde sie uns nur unter einem Winkel von 60° erscheinen, nicht viel größer als das Gestirn des großen Bären; in einer Entfernung von 10 Durchmessern, würde sie nur unter einem Winkel von 2° 25 Min., ungefähr so groß wie das Siebengestirn, und auf 100 Durchmesser unter einem Winkel von 17 Min., kleiner als der berühmte Fleck in der *Andromeda* erscheinen. Sie würde in dieser Entfernung dem bloßen Auge unsichtbar seyn, und durch Fernröhre als ein Wölkchen von schwachem Licht, ähnlich den kleinen Lichtmassen dastehen, denen die Astronomen den Namen der Nebelflecke gegeben haben, und deren, wie früher erwähnt, seit *Herschel* bereits 3000 am Himmel entdeckt sind.

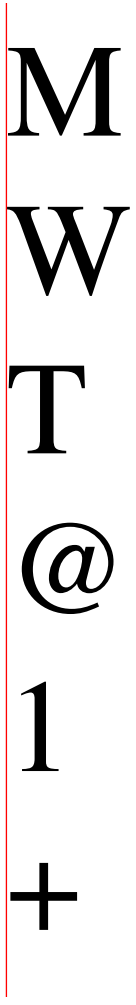
Die unsere Begriffe fast übersteigende Entfernung dieser unendlich weit entlegenen Weltkörper, sind wir dennoch zu berechnen im Stande, seitdem wir gelernt haben die Geschwindigkeit des Lichtes zu messen. Nicht unser Erdkörper bietet aber den Maasstab dazu dar; am Himmel selbst muß die Messung vorgenommen werden. *Olof Römer*, ein Däne, fand in der Verfinsterung der *Jupiters* Trabanten, das Mittel dieses wichtige Problem zu lösen. Er hatte in den Jahren 1670/75 mit dem älteren *Cassini* auf der Sternwarte viele Verfinsterungen der *Jupiters* Monde

von 250 Fuß Brennweite verfertigt, durch welches der große *Cassini* die zwei nächsten Trabanten des Saturn entdeckte. *Auzout* in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich *Herschel* des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen, matten Wolke einschrumpfen, in der sich keine einzelnen Sterne mehr entdecken ließen. Wenn unser Auge von der Milchstraße nur um einen Durchmesser derselben entfernt wäre, so würde sie uns nur unter einem Winkel von 60° erscheinen, nicht viel größer als das Gestirn des großen Bären; in einer Entfernung von 10 Durchmessern, würde sie nur unter einem Winkel von

2° 25 Min., ungefähr so groß wie das Siebengestirn, und auf 100 Durchmesser unter einem Winkel von 17 Min., kleiner als der berühmte Fleck in der *Andromeda* erscheinen. Sie würde in dieser Entfernung dem bloßen Auge unsichtbar seyn, und durch Fernröhre als ein Wölkchen von schwachem Licht, ähnlich den kleinen Lichtmassen dastehen, denen die Astronomen den Namen der Nebelflecke gegeben haben, und deren, wie früher erwähnt, seit *Herschel* bereits 3000 am Himmel entdeckt sind.

Die unsere Begriffe fast übersteigende Entfernung dieser unendlich weit entlegenen Weltkörper, sind wir dennoch zu berechnen im Stande, seitdem wir gelernt haben die Geschwindigkeit des Lichtes zu messen. Nicht unser Erdkörper bietet aber den Maasstab dazu dar; am Himmel selbst muß die Messung vorgenommen werden. *Olof Römer*, ein Däne, fand in der Verfinsterung der *Jupiters* Trabanten, das Mittel dieses wichtige Problem zu lösen. Er hatte in den Jahren 1670/75 mit dem älteren *Cassini* auf der Sternwarte viele Verfinsterungen der *Jupiters* Monde beobachtet, und gefunden, daß der erste Mond nicht immer zur berechneten Zeit aus dem Schatten trat, und daß der Austritt desselben sich immer mehr verspätete, je weiter sich die Erde vom *Jupiter* entfernte: wogegen der Eintritt früher erfolgte, jemehr sie sich demselben näherte, so daß der größte Unterschied 14 Min. betrug. *Römer* schloß, daß diese Ungleichheit von dem Abstande der Erde und des *Jupiters* von einander abhange, und eine Folge der verschiedenen Zeit sey, welche das Licht brauche, um bei ungleicher Entfernung die Erde zu erreichen. – Genauere Berechnungen haben später gezeigt, daß das Licht in einer Sekunde 40,000 Meilen zurücklegt; es gelangt daher von der Sonne bis zu uns, in 8 Min. 13 Sek. Dagegen braucht es vom *Syrius* 31 Jahr, und vom entferntesten Nebelfleck mindestens 94,000 Jahr. Dies giebt eine Entfernung von 33,000 Billionen Meilen. Es folgt daraus, daß das Weltgebäude ein Alter von wenigstens 24,000 Jahr hat, weil das Licht was wir heute sehen, schon vor so langer Zeit von dort ausgegossen ist. Schwindel erregend! gleich der Betrachtung, daß die zerstörendsten Revolutionen jene leuchtenden Gestirne längst vernichtet haben können, welche mit ruhiger Klarheit unsere Nächte erhellen, und daß vielleicht Generationen vergehen, ehe nur die Kunde davon zu uns gelangt.



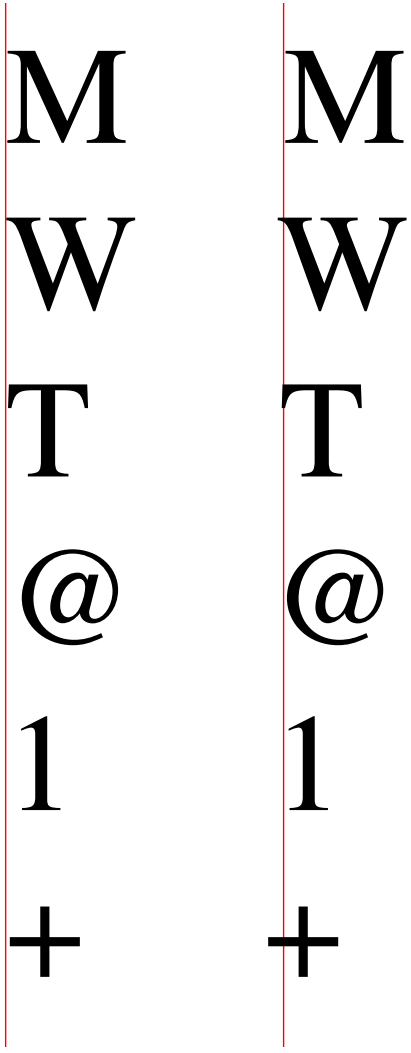
- Die Zeichen am linken Rand werden per Voreinstellung alle ausgerichtet entsprechend dem jeweiligen Ursprung des Koordinatensystems, das bei der Definition verwendet wurde.
- Die *llx*-Komponente (*lower left x*) der Bounding-Box liegt typischerweise nicht bei 0. Bei »M« und »W« in dem Schriftschnitt Adobe Times Roman ist sie beispielsweise 12, bei »T« 17, bei »@« 116, bei »1« 111 und bei »+« 30.
- Diese Werte sind dahingehend optimiert, gute Abstände zwischen den Zeichen innerhalb eines Wortes zu erreichen.
- Am linken Rand sieht das jedoch ungleichmäßig aus. Das Subtrahieren von *llx* wäre keine Abhilfe, da das Auge nicht die Bounding-Box erkennt, sondern sich von der Farbe leiten lässt.

- Das gleiche Problem existiert am rechten Rand.
- Hier werden bei einem Blocksatz normalerweise die Zeichen entsprechend ihrer Weite positioniert.
- Da die Weite die *urx*-Komponente überragt, verbleibt auch rechts variierender weißer Raum, so dass die rechte Seite ebenfalls für das Auge ungleichmäßig wirkt.

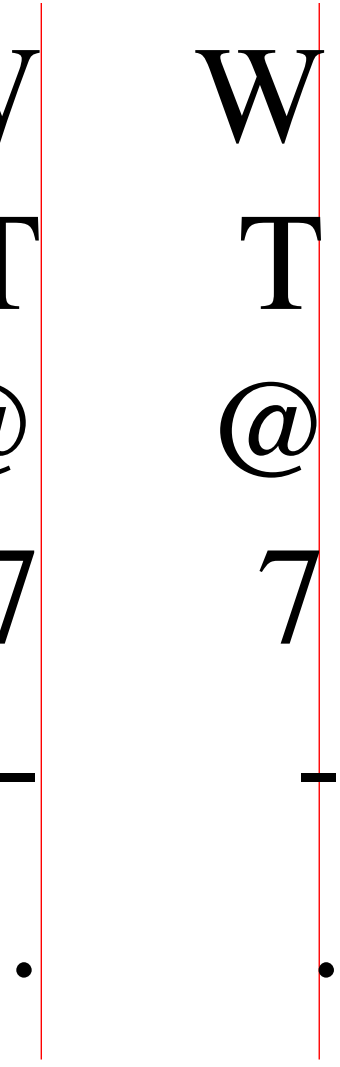
W
T
@
7
-
.

- Einige wenige Satzsysteme unterhalten für ausgewählte Schriftschnitte Tabellen für den Randausgleich. Diese sind leider nicht Bestandteil der Adobe Type-1-Schriftschnitte (oder anderer üblicher Repräsentierungen).
- Das Paket microtype für pdfTEX unterhält solche Tabellen u.a. auch für Adobe Times Roman:

Zeichen	*llx*	*urx*	Weite	Randausgleich in ‰	
				links	rechts
M	12	863	889	0	0
W	12	932	944	50	50
T	17	593	611	50	50
@	116	809	921	100	100
1	111	394	500	150	150
7	20	449	500	50	100
+	30	534	564	250	250
-	39	285	333	500	500
.	70	181	250	0	700



- Hier sind beide Fälle jeweils im Vergleich, jeweils links ohne und rechts mit Randausgleich.

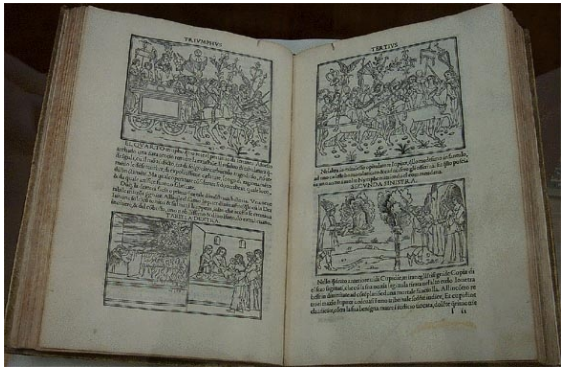


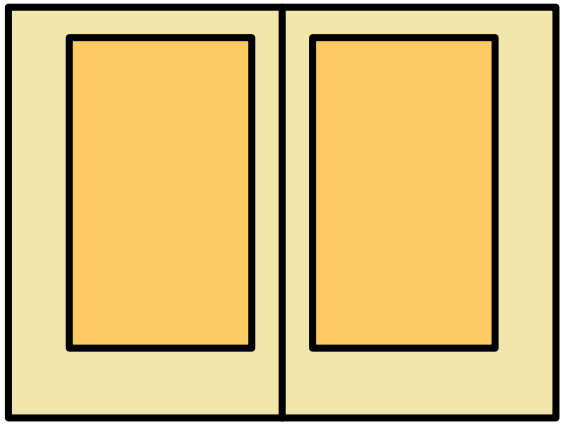
bequemen, und übertriebenen Länge. Auf Befehl *Ludwig XIV*^{ten} wurde von *Campani* in *Bologna* ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große *Cassini* die zwei nächsten Trabanten des Saturn entdeckte. *Auzout* in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich *Herschel* des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen, matten Wolke einschrumpfen, in der sich keine einzelnen Sterne mehr entdecken ließen. Wenn unser Auge von

bequemen, und übertriebenen Länge. Auf Befehl *Ludwig XIV*^{ten} wurde von *Campani* in *Bologna* ein Fernrohr von 250 Fuß Brennweite verfertigt, durch welches der große *Cassini* die zwei nächsten Trabanten des Saturn entdeckte. *Auzout* in Frankreich brachte sogar ein Objectiv von 600 Fuß Brennweite zu Stande, das aber aus Mangel einer schicklichen Vorrichtung nicht gebraucht werden konnte.

Herschel gelang es endlich, durch die Vergrößerung und Lichtstärke seines 20 füßigen Telescops, den Schimmer der Milchstraße vollkommen in kleine Sterne aufzulösen, die sich deutlich von einander unterscheiden lassen; auch bemerkte er in der That, daß jede Stelle der Milchstraße um so sternenreicher ist, je glänzender sie dem bloßen Auge erscheint. – Um sich einen Begriff von der unzähligen Menge der Sterne zu machen, die den Schimmer der Milchstraße hervorbringen, bediente sich *Herschel* des genau bestimmten Feldes seines Telescops als Maaß. Er fand im Durchschnitt, daß ein Raum der Milchstraße von 2° Breite, und 15° Länge nicht weniger als 50 000 Sterne enthält, die noch groß genug waren um deutlich gezählt zu werden, und wenigstens 100,000 die wegen ihres schwachen Lichtes sich nicht mehr zählen ließen. Da nun die Milchstraße im Durchschnitt eine Breite von wenigstens 12° hat, und sich über den ganzen Himmel durch 360° erstreckt, so würde dies wenigstens 20 Millionen Sterne in der Milchstraße geben. – Wären wir aber auch im Stande die Menge der Sterne in der Milchstraße einigermaßen genau zu bestimmen, so würde uns dies bei weitem nicht einen hinlänglichen Begriff von der Unermeßlichkeit auch nur desjenigen Theils des Universums geben, den unser Auge erreichen kann. Wir wissen nicht, wie viele Sternhaufen, der Milchstraße gleich, über den Himmel verbreitet liegen. Es ist offenbar, daß wenn die Milchstraße tausendmal weiter von uns entfernt wäre, die einzelnen Sterne, welche man jetzt noch in ihr entdecken kann, in eben dem Verhältniß an Lichtstärke verlieren, und näher zusammenrücken würden: das Ganze würde endlich zu einer kleinen, matten Wolke einschrumpfen, in der sich keine einzelnen Sterne mehr entdecken ließen. Wenn unser Auge von



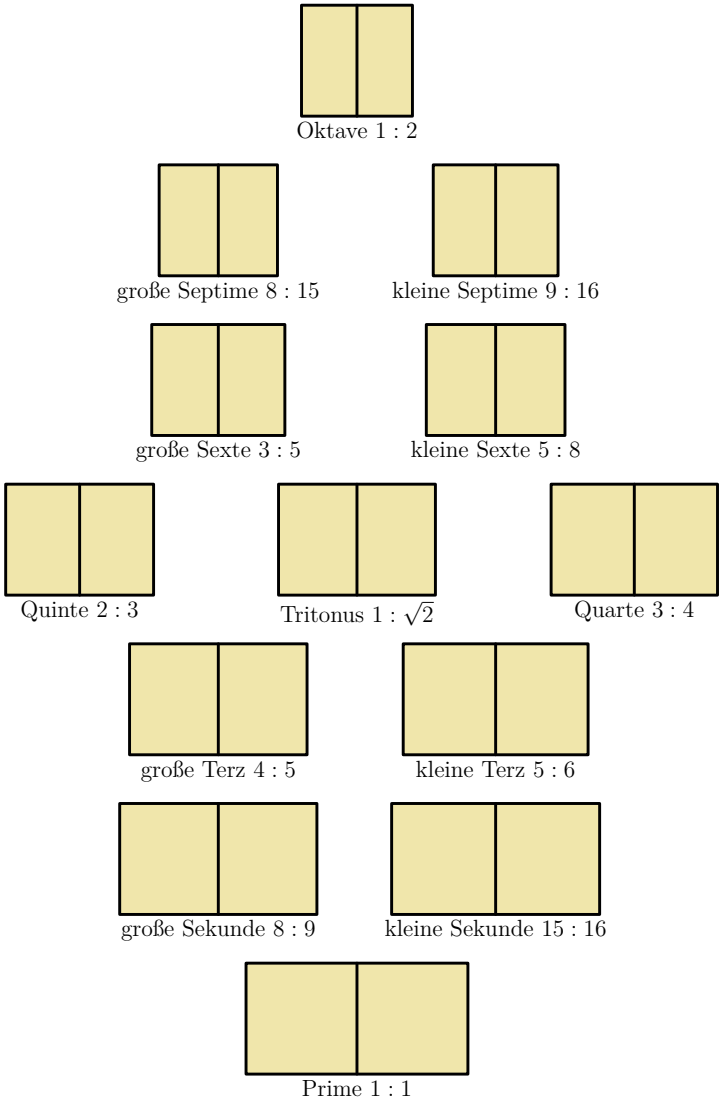


Seitenformat des Werks *Hypnerotomachia Poliphili*, Venedig 1499, gedruckt von Aldus Manutius

- »If the book appears to be only a paper machine, produced at their own convenience by other machines, only machines will want to read it.« (Bringhurst)
- Da die menschlichen Sinne manche Proportionen als harmonisch und andere als unharmonisch empfinden, sollten diese nicht rein zufällig gewählt werden.
- Es gibt historisch eine lange Suche nach idealen Proportionen. So beschäftige sich bereits Pythagoras mit der Harmonielehre.
- Deswegen ist es auch wenig sinnvoll, die Erfahrungen aus dieser Tradition zu ignorieren.

- Bereits die Griechen stellten fest, dass Töne miteinander harmonieren, wenn ihre Frequenzen in einem rationalen Verhältnis zueinander stehen. (Der Zusammenhang wird auch ohne den Begriff der Frequenz klar, wenn das Verhältnis der Saitenlängen untersucht wird.)
- So hat in unserer Tonleiter eine Oktave das Verhältnis $1 : 2$, und der Grundakkord zu C, der sich aus dem 3. und 5. Oberton ergibt, ist im Verhältnis $4 : 5 : 6$.
- Die gesamte chromatische Tonleiter mit ihren 12 Tönen in einer Oktave lässt sich aus der Proportion $2 : 3$ ableiten. (Es geht nicht ganz genau auf, was dazu führt, dass es verschiedene Lösungen dazu gibt, die in der Musiktheorie Stimmungen genannt werden.)
- Interessanterweise lässt sich der Harmoniebegriff aus der Musik auch in andere Felder übertragen.
- Das Verhältnis von Seitenbreite und Seitenhöhe des vorangegangenen Beispiels ist $2 : 3$.

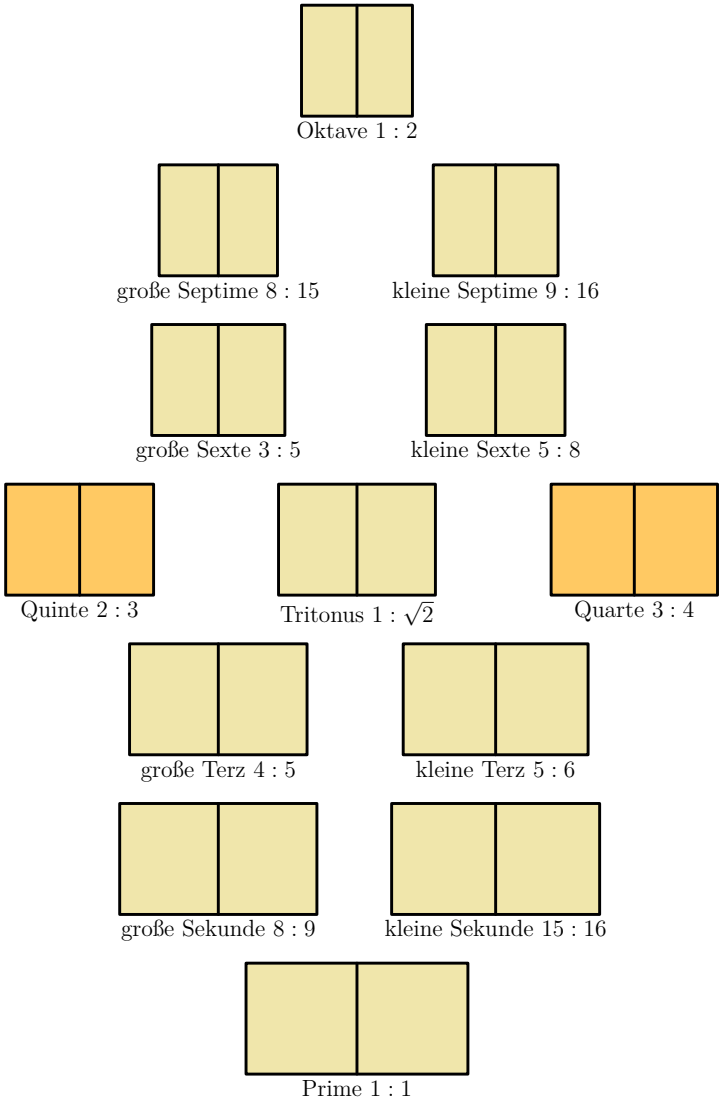
Chromatische Leiter der Seitenproportionen



- Für die Wahl des Seitenformats werden gerne die Proportionen verwendet, die auch bei der chromatischen Tonleiter entsprechend der reinen Stimmung relevant sind.
- Im Mittelalter waren die zum Grundakkord gehörenden Verhältnisse 2 : 3 und 3 : 4 sehr beliebt.
- In der Renaissance wurden schmalere Seiten populär mit den Formaten 8 : 15, 9 : 16, 3 : 5 und 5 : 8.

Diagramm nach dem Werk von Bringhurst

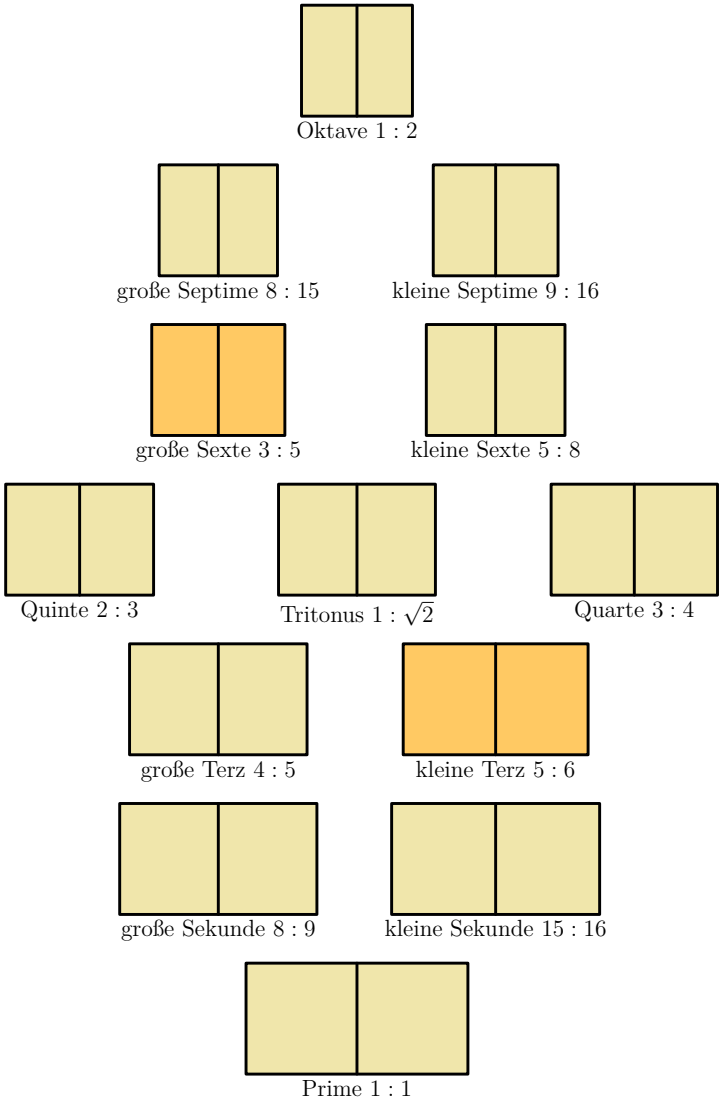
Chromatische Leiter der Seitenproportionen



- Wenn eine Seite im Format 2 : 3 gefaltet wird, so erhalten wir das Format 3 : 4.
- Umgekehrt führt eine Faltung des Formats 3 : 4 zurück zu 2 : 3.
- Dabei ist zu berücksichtigen, dass immer beide Formate gleichzeitig zu sehen sind in Form der einzelnen Seite und der aufgeschlagenen Doppelseite.
- Somit sollten beide in einer harmonischen Beziehung zueinander stehen.
- In der Musiktheorie ist das eine Intervall jeweils das Komplementärintervall des anderen.

Diagramm nach dem Werk von Bringhurst

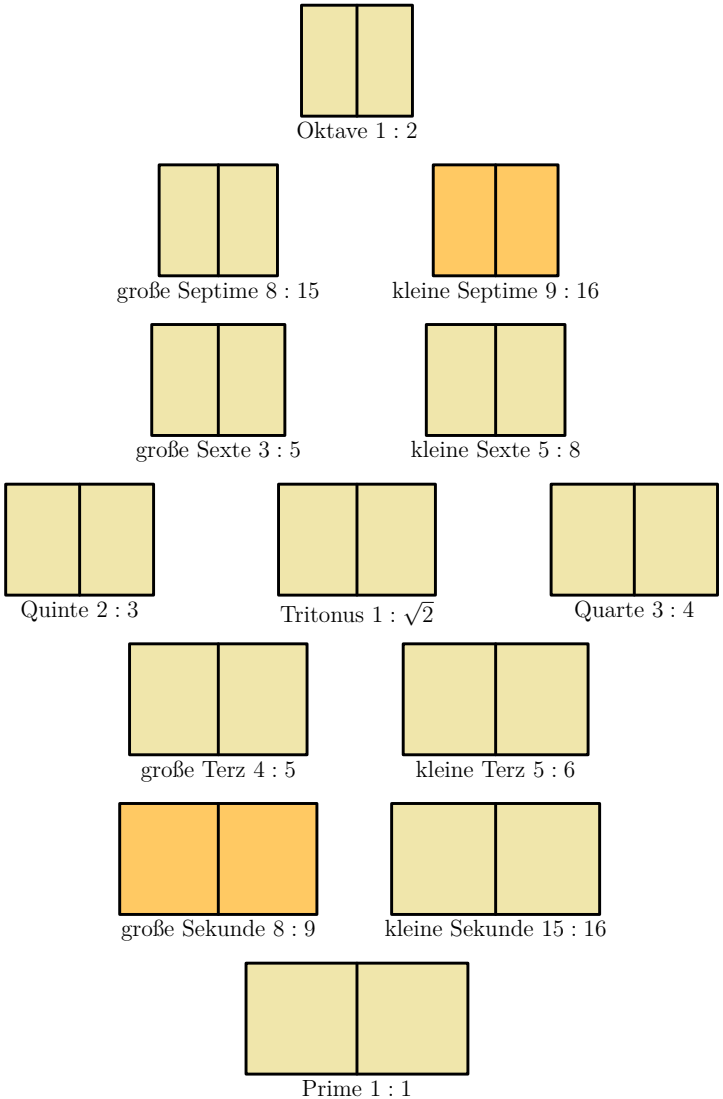
Chromatische Leiter der Seitenproportionen



- Analog steht 3 : 5 in Beziehung mit 5 : 6.

Diagramm nach dem Werk von Bringhurst

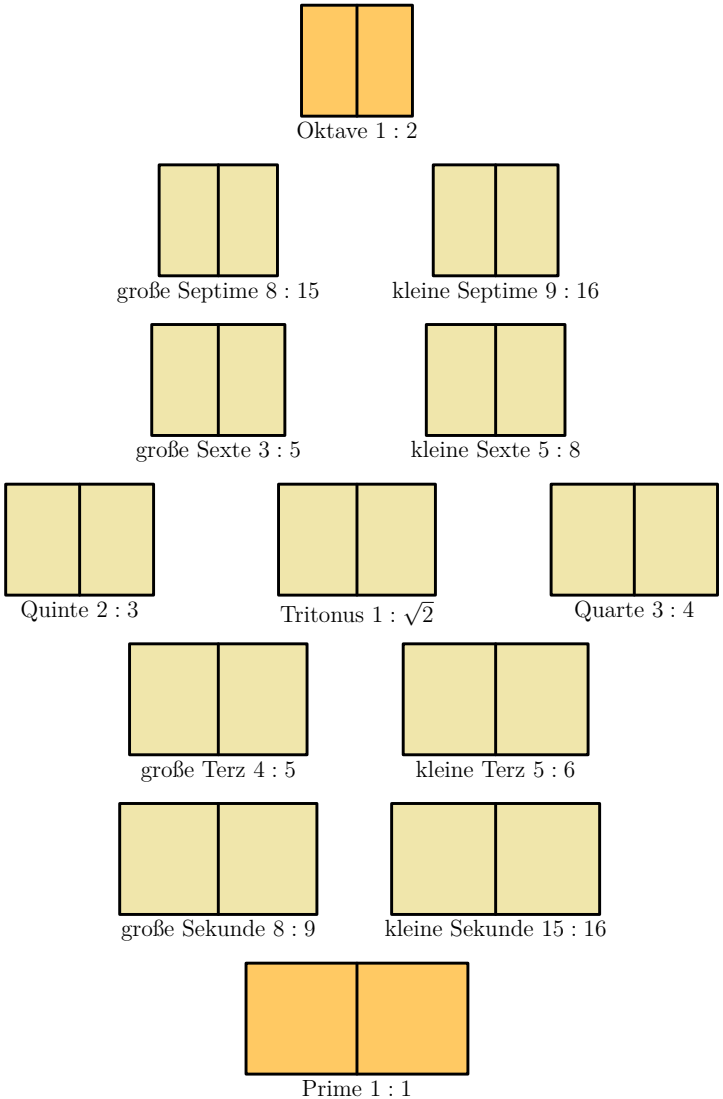
Chromatische Leiter der Seitenproportionen



- Oder 9 : 16 in Beziehung mit 8 : 9.

Diagramm nach dem Werk von Bringhurst

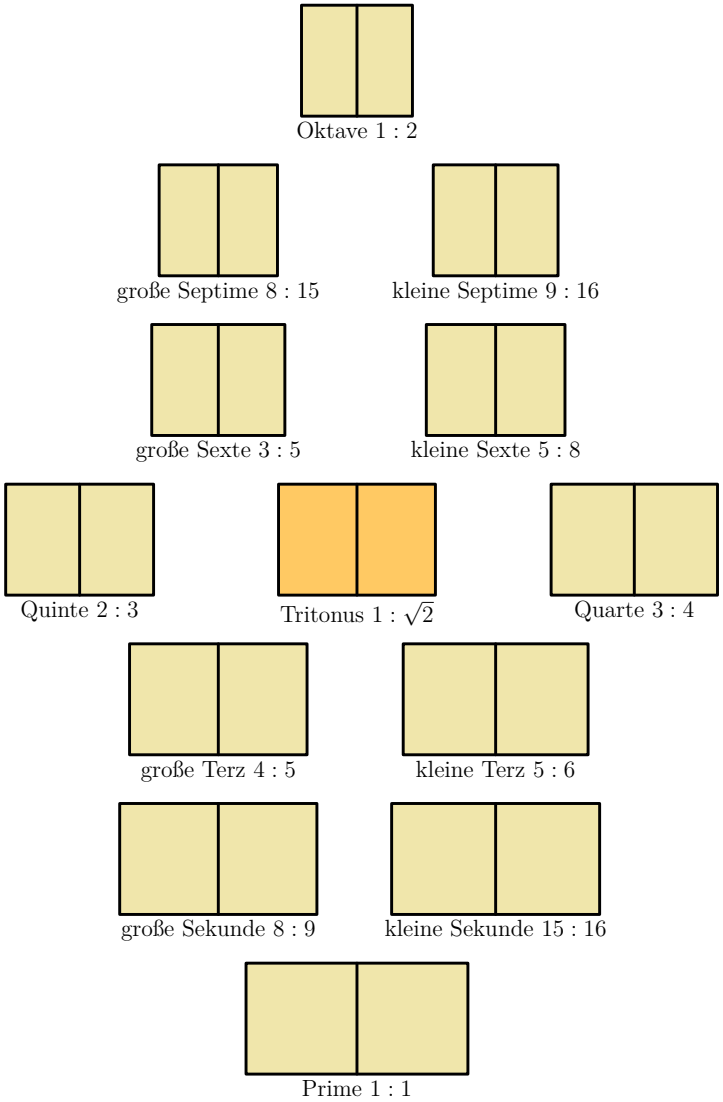
Chromatische Leiter der Seitenproportionen



- Und auch 1 : 2 mit 1 : 1.

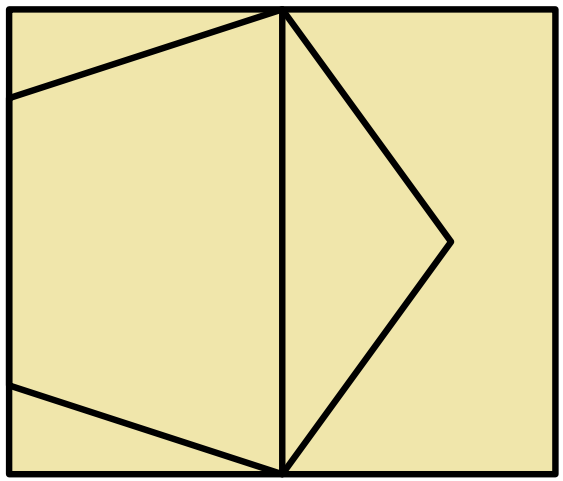
Diagramm nach dem Werk von Bringhurst

Chromatische Leiter der Seitenproportionen

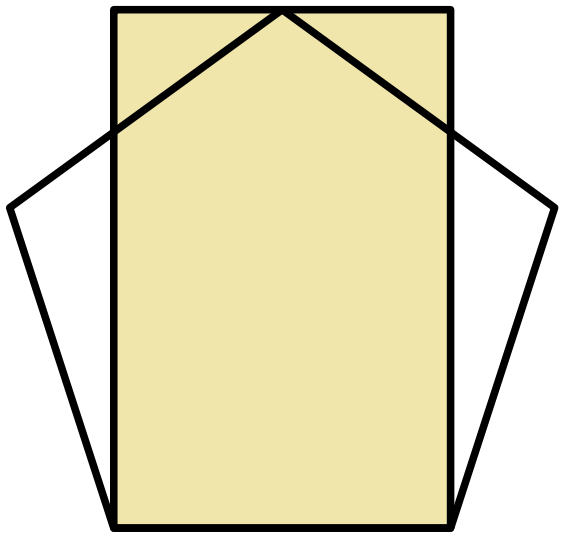


- Nur $1 : \sqrt{2}$, das (in der gleichstufigen Stimmung) die Oktave genau teilt, steht nur zu sich selbst in Beziehung.
- In der Musiktheorie gilt entsprechend der Tritonus als das disharmonischste Intervall.
- Das Format entspricht der ISO-A-Familie, also ISO A4, A3 etc.
- Das Papierformat ist ebenfalls disharmonisch, weil der Kontrast zu dem Komplementärformat fehlt.
- Dies lässt sich nur retten, indem mit Geschick ein Satzspiegel gewählt wird, der den fehlenden Kontrast nachholt.

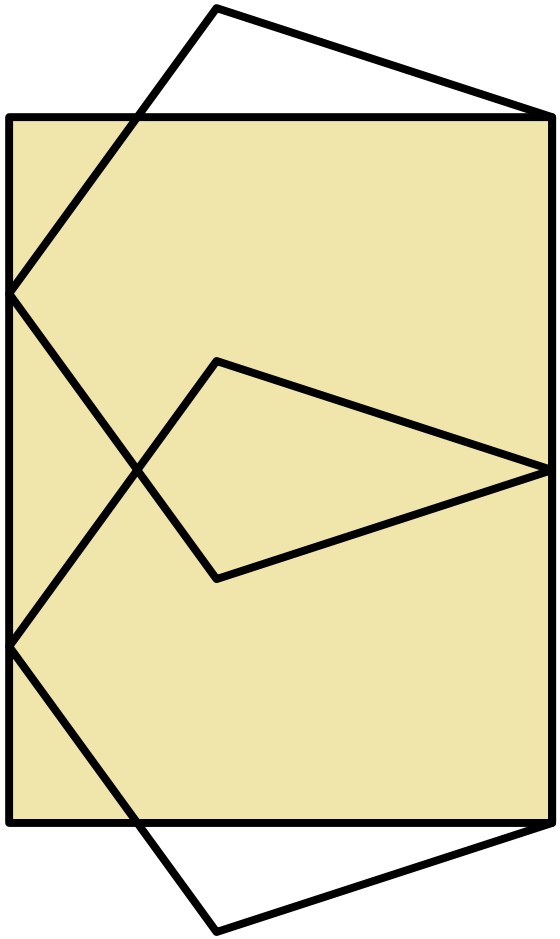
Diagramm nach dem Werk von Bringhurst



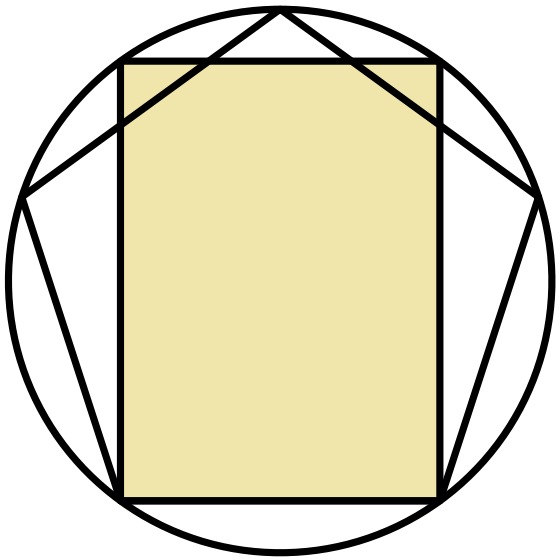
- Neben den Proportionen aus der Musiktheorie fanden in der Renaissance auch vom Pentagon abgeleitete Proportionen Anwendung.
- Nebenstehende Proportion im Verhältnis von etwa $1 : 1,701$ kam für den Textblock des Werks *Hypnerotomachia Poliphili* zum Einsatz.



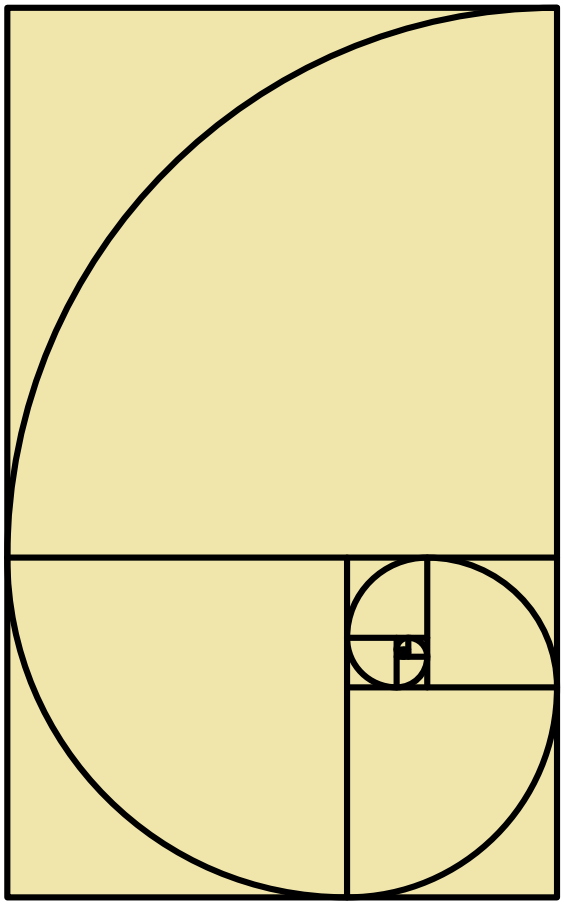
- Nebenstehendes Format mit der Proportion $1 : 1,539$ wird als Pentagonseite bezeichnet.
- Beispielsweise verwendet der Springer-Verlag gerne dieses Format für Fachbücher wie beispielsweise die *Lecture Notes in Computer Science*.



- Wenn das Pentagon-Format verdoppelt wird, sind wir sehr nahe am US-Letter-Format mit $11 \times 8\frac{1}{2}$ Inch.



- Nebenstehendes Format mit der Proportion $1 : 1,376$ wird als kurze Pentagonseite bezeichnet.
- Viele amerikanische Verlage (wie beispielsweise Addison-Wesley) verwenden dieses Format in der Größe $6,5 \times 9$ Inch für Fachbücher.



- Zwei Zahlenwerte a und b mit $a < b$ erfüllen die Proportion des goldenen Schnitts, wenn $\frac{a}{b} = \frac{b}{a+b}$.
- Dies entspricht dem Verhältnis $1 : \phi$ mit $\phi = \frac{1+\sqrt{5}}{2}$.
- Der goldene Schnitt kommt vielfach in der Natur vor, wurde von den Griechen bereits als besonders harmonische Proportion betrachtet und fand in der Renaissance neue Beachtung.

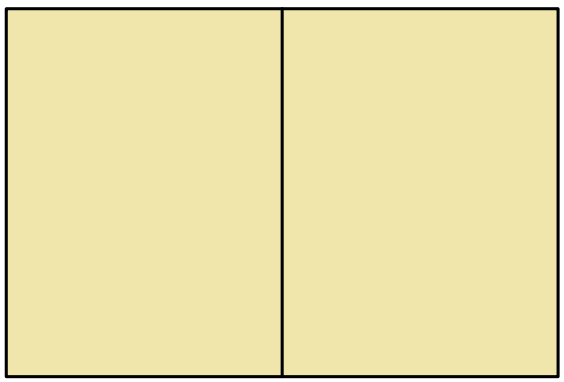
- Die Fibonacci-Folge $\{F_n\}_{n=1}^{\infty}$ wird rekursiv wie folgt definiert:
  $F_1 = F_2 = 1$
  $F_n = F_{n-1} + F_{n-2}$
- Es gilt

$$\lim_{n\to\infty} \frac{F_n}{F_{n-1}} = \phi$$

  wie bereits 1753 der schottische Mathematiker Robert Simson nachwies.
- Aus diesem Grunde werden gerne die Proportionen aufeinander folgender Fibonacci-Zahlen als Näherungswert verwendet.

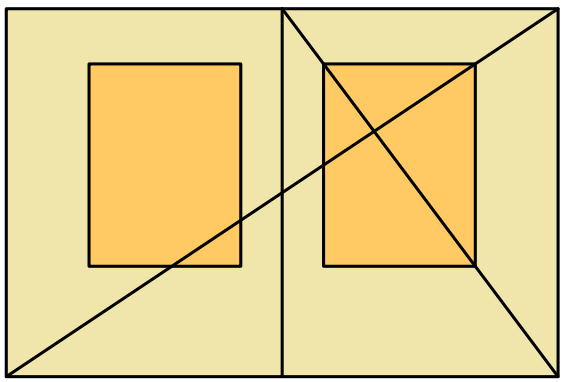
- Fibonacci-Zahlen werden auch gerne verwendet, um eine Reihe von Größen zu erhalten, die zueinander harmonisch wirken.
- So könnten beispielsweise für Schriftgrößen F_5 bis F_{11} verwendet werden: $5, 8, 13, 21, 34, 55, 89$.
- Wenn das nicht genügt oder unpassend erscheint, kann auch eine Folge mit anderen Ausgangswerten verwendet werden. Beispiele:
  $6, 10, 16, 26, 42, 68, 110, \ldots$
  $4, 7, 11, 18, 29, 47, 76, \ldots$
- Bringhurst schlägt vor, solche Folgen bei Bedarf zu mischen:
  $6, 8, 10, 13, 16, 21, 26, 34, 42, \ldots$
- So eine Fibonacci-Folgen-Kombination wurde auch von dem Architekten Le Corbusier verwendet:
  $4, 5, 6\frac{1}{2}, 8, 10\frac{1}{2}, 13, 17, 21, 27\frac{1}{2}, 34, 44\frac{1}{2}, \ldots$

Diagonalkonstruktionen nach Milchsack und Tschichold 285



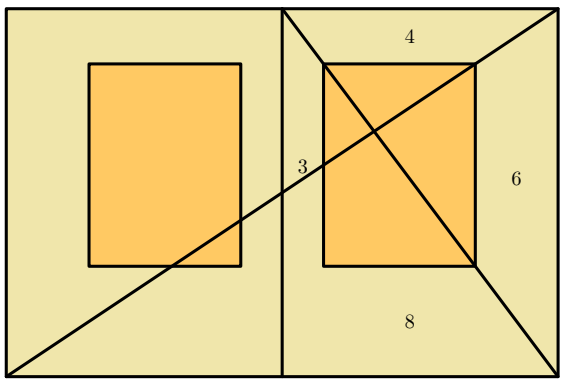
- Gustav Milchsack war ein Bibliothekar (1850–1919), dem auffiel, dass die neueren Bücher nicht mehr die schönen Proportionen der alten Bücher aufwiesen.
- Das Problem war, dass es zu seiner Zeit üblich war, den Textblock genau auf der Seite zu zentrieren. Dies führt nach der Ansicht von Tschichold dazu, dass solche Satzflächen heruntergeglitten erscheinen.
- Gustav Milchsack stellte aufgrund der Proportionen in alten Werken einige Randproportionen auf, die harmonisch wirken.
- Er ging dabei von harmonischen Seitenproportionen (wie beispielsweise 3 : 2) aus, die damals noch selbstverständlich waren.

Diagonalkonstruktionen nach Milchsack und Tschichold 285



- Mehrere Konstruktionen, die Textblöcke innerhalb einer Seite dimensionieren und platzieren, basieren auf der Diagonalkonstruktion.
- Die linke obere Ecke und die rechte untere Ecke des Textblocks liegen dabei immer auf der Seitendiagonalen. Auf diese Weise hat der Textblock die gleiche Proportion wie die Seite. (Hier im Beispiel die Proportion der Quarte 3 : 4.)
- Die rechte obere Ecke des Textblocks muss bei der Diagonalkonstruktion auf der Doppelseitendiagonalen liegen. Auf diese Weise werden die Verhältnisse der Ränder festgelegt.

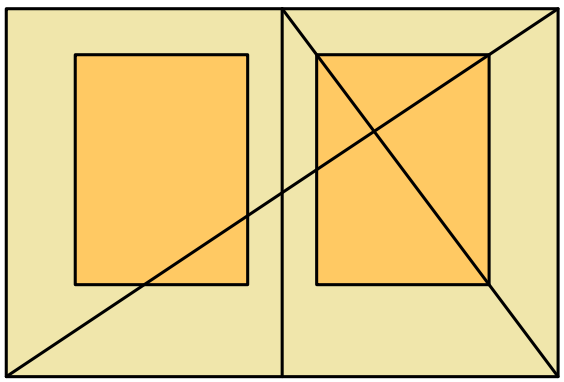
Diagonalkonstruktionen nach Milchsack und Tschichold 285



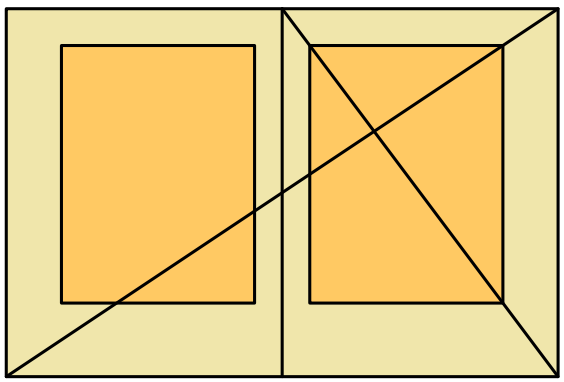
- Unabhängig von der Proportion der Seite liegt bei dieser Konstruktion immer das Verhältnis von Innenrand zu dem zugehörigen Außenrand bei 1 : 2.
- Das Verhältnis der Innenränder zueinander bzw. das Verhältnis der Außenränder ergibt sich aus der Seitenproportionierung.
- Aus der Seitenproportion 3 : 4 ergeben sich hier die Randproportionen 3 : 4 : 6 : 8. (Proportionen aus Milchsacks Hauptgesetz.)
- Bei einer Seitenproportion von 2 : 3 würden wir die Randproportionen 2 : 3 : 4 : 6 erhalten.

Diagonalkonstruktionen nach Milchsack und Tschichold 285

- Die allgemeine Diagonalkonstruktion lässt offen, wie groß der Textblock dimensioniert wird.

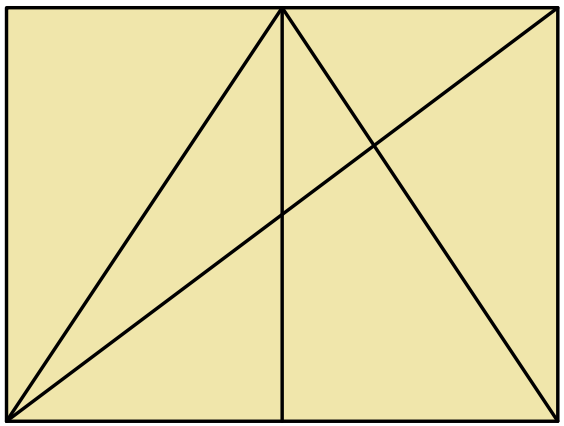


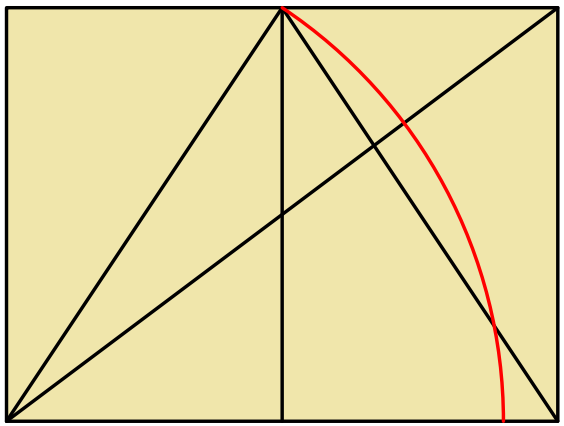
Diagonalkonstruktionen nach Milchsack und Tschichold 285



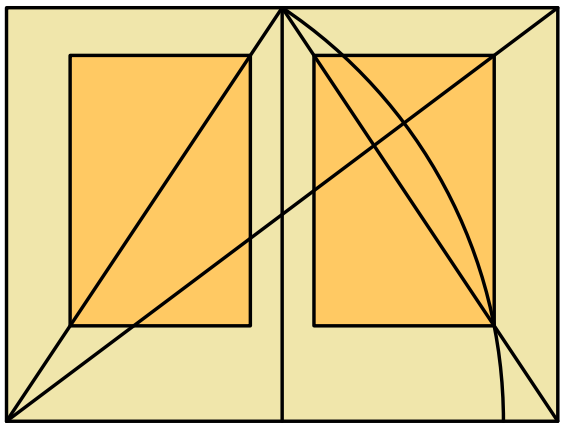
- Die allgemeine Diagonalkonstruktion lässt offen, wie groß der Textblock dimensioniert wird.
- Dafür gibt es entsprechend der Vorbilder aus dem Mittelalter und der Renaissance Vorschläge von Tschichold und van de Graaf.

- Zunächst werden die beiden Seitendiagonalen und eine der Doppelseiten-Diagonalen eingezeichnet.

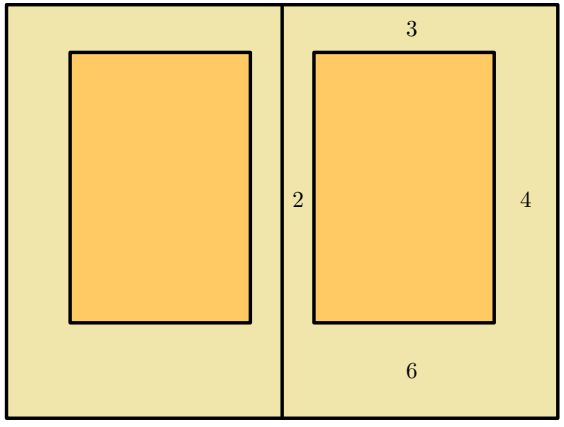




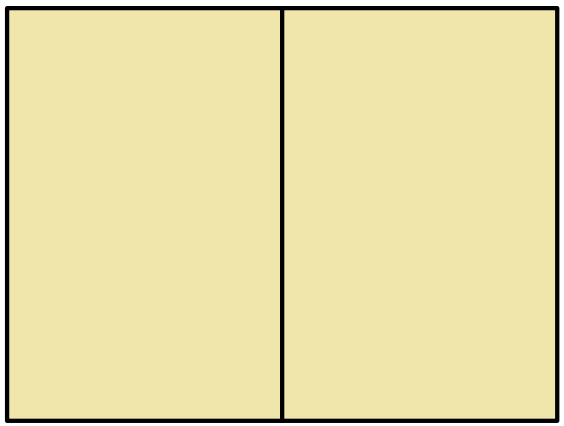
- Zunächst werden die beiden Seitendiagonalen und eine der Doppelseiten-Diagonalen eingezeichnet.
- Dann wird mit dem Zirkel die linke Seitendiagonale abgesteckt und ein entsprechender Kreisbogen auf der rechten Seite gezogen.



- Zunächst werden die beiden Seitendiagonalen und eine der Doppelseiten-Diagonalen eingezeichnet.
- Dann wird mit dem Zirkel die linke Seitendiagonale abgesteckt und ein entsprechender Kreisbogen auf der rechten Seite gezogen.
- Daraus ergeben sich dann die Eckpunkte des Textblocks. Da die linke obere und die rechte untere Ecke auf der rechten Seitendiagonale verlaufen, werden für den Textblock die Seitenproportionen übernommen.

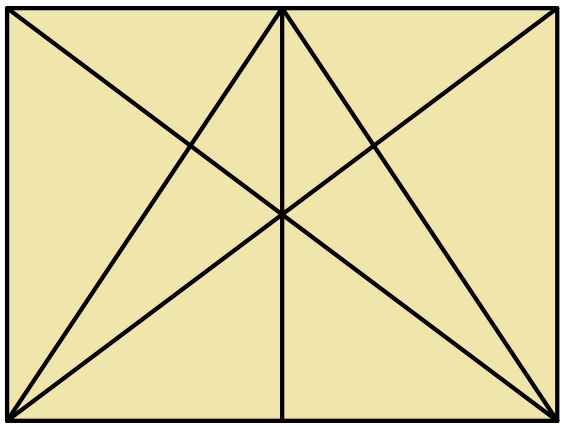


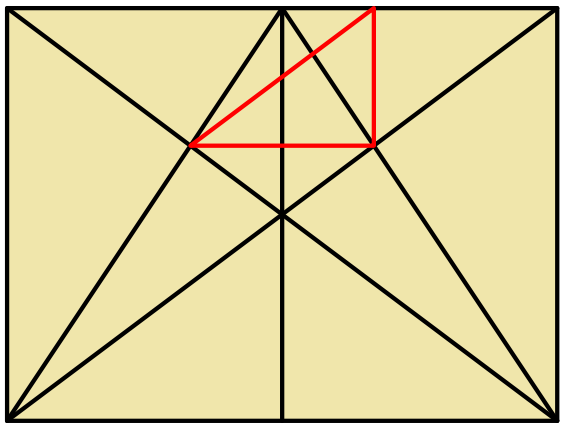
- Zunächst werden die beiden Seitendiagonalen und eine der Doppelseiten-Diagonalen eingezeichnet.
- Dann wird mit dem Zirkel die linke Seitendiagonale abgesteckt und ein entsprechender Kreisbogen auf der rechten Seite gezogen.
- Daraus ergeben sich dann die Eckpunkte des Textblocks. Da die linke obere und die rechte untere Ecke auf der rechten Seitendiagonale verlaufen, werden für den Textblock die Seitenproportionen übernommen.
- Die Ränder haben dann entsprechend Milchsacks Hauptgesetz die Proportionen 2 : 3 : 4 : 6.
- Nachteil: Diese Konstruktion ist nur für die Seitenproportion 3 : 2 sinnvoll.



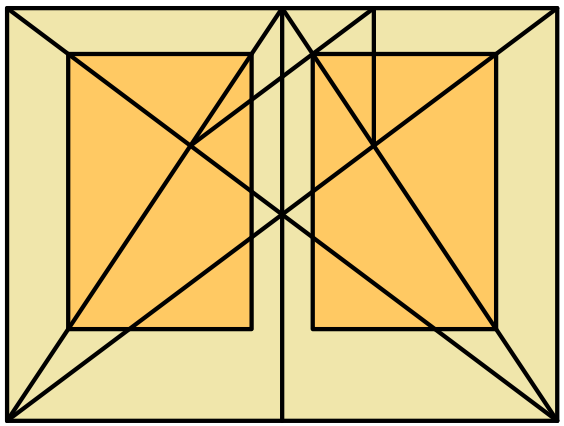
- Der Niederländer Joh. A. van de Graaf publizierte 1946 in der Amsterdamer Zeitschrift Tété ein Konstruktionsverfahren, dass für beliebige vorgegebene Seitenproportionen im Rahmen der Diagonalkonstruktion eine sinnvolle Textblockgröße festlegt.

- Zunächst werden die Diagonalen der Doppelseite und der beiden einzelnen Seiten eingezeichnet.

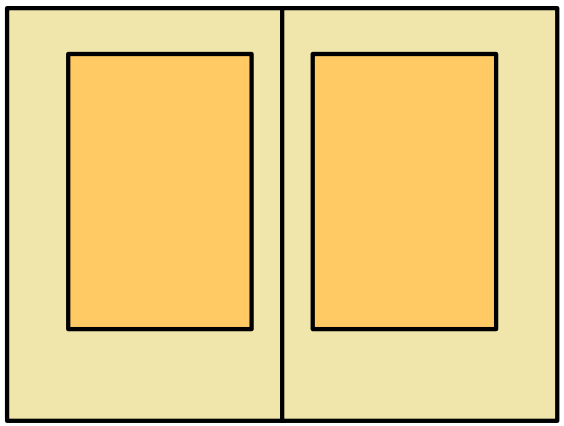




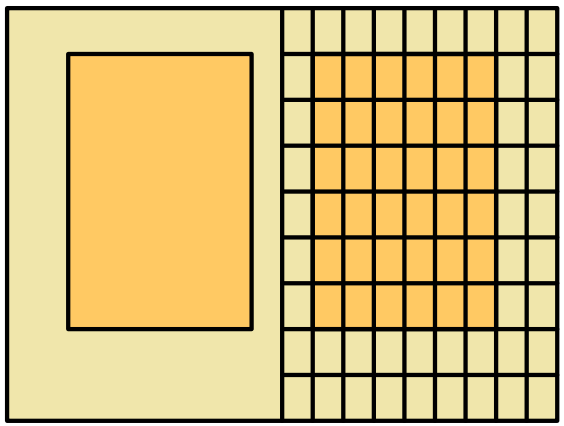
- Zunächst werden die Diagonalen der Doppelseite und der beiden einzelnen Seiten eingezeichnet.
- Ausgehend von den beiden oberen Schnittpunkten wird ein rechtwinkliges Dreieck gebildet, dessen Hypothenuse die rechte Diagonale schneidet.



- Zunächst werden die Diagonalen der Doppelseite und der beiden einzelnen Seiten eingezeichnet.
- Ausgehend von den beiden oberen Schnittpunkten wird ein rechtwinkliges Dreieck gebildet, dessen Hypothenuse die rechte Diagonale schneidet.
- Dieser Schnittpunkt legt die obere linke Ecke des rechten Textblocks fest.

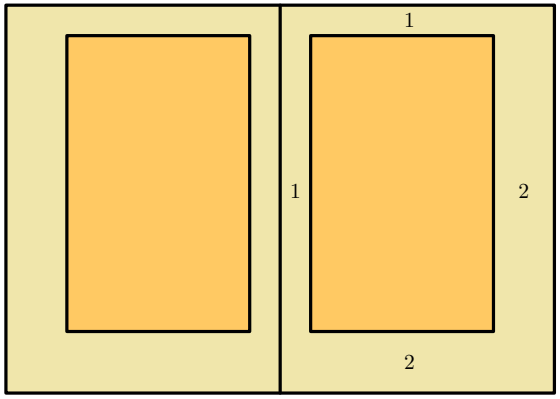


- van de Graaf und auch Jan Tschichold erachten dieses Verfahren als sinnvoll unabhängig von dem vorgebenen Seitenformat. Es ist nach Meinung von Tschichold auch für ISO A4 sinnvoll.
- Der Textblock und die Seiten haben die gleichen Proportionen.
- Innenrand und Außenrand haben das Verhältnis 1 : 2. Das gleiche Verhältnis besteht zwischen dem oberen und dem unteren Rand.
- Die Seitenlängen der Seite stehen im Verhältnis 3 : 2 zu den entsprechenden Seitenlängen des Textblocks.



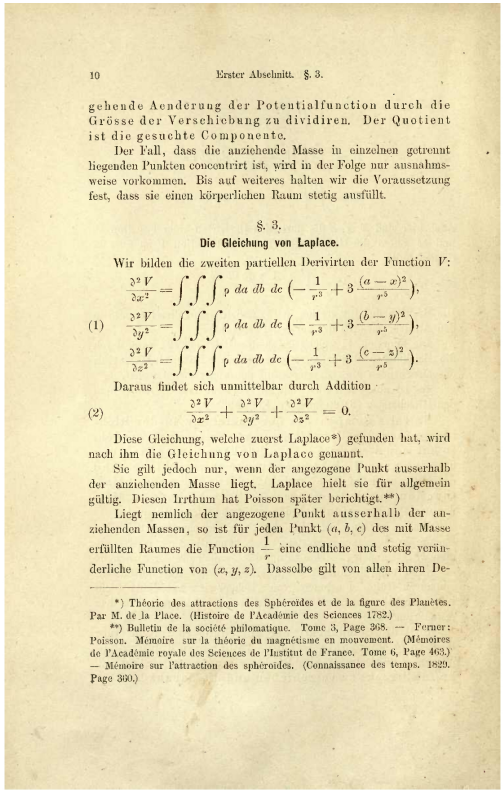
- van de Graaf und auch Jan Tschichold erachten dieses Verfahren als sinnvoll, unabhängig von dem vorgebenen Seitenformat. Es ist nach Meinung von Tschichold auch für ISO A4 sinnvoll.
- Der Textblock und die Seiten haben die gleichen Proportionen.
- Innenrand und Außenrand haben das Verhältnis 1 : 2. Das gleiche Verhältnis besteht zwischen dem oberen und dem unteren Rand.
- Die Seitenlängen der Seite stehen im Verhältnis 3 : 2 zu den entsprechenden Seitenlängen des Textblocks.
- Wenn ein 9×9-Raster für jede Seite aufgelegt wird, dann passt sich der Textblock genau ein.

Goldener Schnitt in ISO A4 nach Bringhurst



- Der goldene Schnitt als Proportion für den Textblock eignet sich im Kontrast zu dem eher disharmonischen Format ISO A4 $(1 : \sqrt{2})$.
- Bringhurst schlägt vor, den oberen und inneren Rand gleich auf $1/9$ der Seitenweite zu dimensionieren.
- Wenn dann der untere Rand im Verhältnis $2 : 1$ zu dem oberen Rand gesetzt wird, ergibt sich ein äußerer Rand, der näherungsweise im Verhältnis $2 : 1$ zum inneren Rand proportioniert ist.
- Umgekehrt könnten natürlich die Randverhältnisse wie angezeigt genau eingehalten werden. Dann wird der goldene Schnitt nur näherungsweise erreicht (etwa $1 : 1,6213$ statt $1 : \phi \approx 1 : 1,6180$).

- Das *wissenschaftliche Lesen* entspricht weitgehend dem *linearen Lesen*. Es erlaubt und fordert aber einige Ergänzungen:
  - Die Zielgruppe sind berufliche Leser, die nicht auf maximalen Lesekomfort angewiesen sind, sondern die auch mit längeren Zeilen zurecht kommen, wenn es darum geht, umfangreiche Inhalte in kompakter Form anzubieten.
  - Strukturierende Elemente wie hierarchische Überschriften und die Verwendung aktiver Auszeichnungen (wie z.B. durch Fettschrift) sollten die Orientierung erleichtern.



10 Erster Abschnitt. §. 3.

gehende Aenderung der Potentialfunction durch die Grösse der Verschiebung zu dividiren. Der Quotient ist die gesuchte Componente.

Der Fall, dass die anziehende Masse in einzelnen getrennt liegenden Punkten concentrirt ist, wird in der Folge nur ausnahmsweise vorkommen. Bis auf weiteres halten wir die Voraussetzung fest, dass sie einen körperlichen Raum stetig ausfüllt.

§. 3.

Die Gleichung von Laplace.

Wir bilden die zweiten partiellen Derivirten der Function V:

$$(1)\quad \begin{aligned} \frac{\partial^2 V}{\partial x^2} &= \int\int\int \rho\, da\, db\, dc \left(-\frac{1}{r^3} + 3\frac{(a-x)^2}{r^5}\right), \\ \frac{\partial^2 V}{\partial y^2} &= \int\int\int \rho\, da\, db\, dc \left(-\frac{1}{r^3} + 3\frac{(b-y)^2}{r^5}\right), \\ \frac{\partial^2 V}{\partial z^2} &= \int\int\int \rho\, da\, db\, dc \left(-\frac{1}{r^3} + 3\frac{(c-z)^2}{r^5}\right). \end{aligned}$$

Daraus findet sich unmittelbar durch Addition

$$(2)\quad \frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} + \frac{\partial^2 V}{\partial z^2} = 0.$$

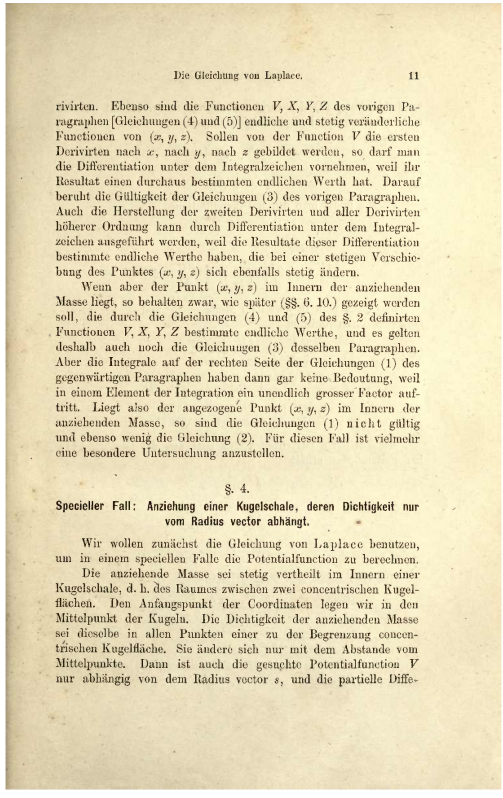
Diese Gleichung, welche zuerst Laplace*) gefunden hat, wird nach ihm die Gleichung von Laplace genannt.

Sie gilt jedoch nur, wenn der angezogene Punkt ausserhalb der anziehenden Masse liegt. Laplace hielt sie für allgemein gültig. Diesen Irrthum hat Poisson später berichtigt.**)

Liegt nemlich der angezogene Punkt ausserhalb der anziehenden Massen, so ist für jeden Punkt (a, b, c) des mit Masse erfüllten Raumes die Function $\frac{1}{r}$ eine endliche und stetig veränderliche Function von (x, y, z). Dasselbe gilt von allen ihren De-

*) Théorie des attractions des Sphéroides et de la figure des Planètes. Par M. de la Place. (Histoire de l'Académie des Sciences 1782.)

**) Bulletin de la société philomatique. Tome 3, Page 388. — Ferner: Poisson. Mémoire sur la théorie du magnétisme en mouvement. (Mémoires de l'Académie royale des Sciences de l'Institut de France. Tome 6, Page 463.) — Mémoire sur l'attraction des sphéroides. (Connaissance des temps. 1829. Page 360.)



Die Gleichung von Laplace. 11

rivirten. Ebenso sind die Functionen V, X, Y, Z des vorigen Paragraphen [Gleichungen (4) und (5)] endliche und stetig veränderliche Functionen von (x, y, z). Sollen von der Function V die ersten Derivirten nach x, nach y, nach z gebildet werden, so darf man die Differentiation unter dem Integralzeichen vornehmen, weil ihr Resultat einen durchaus bestimmten endlichen Werth hat. Darauf beruht die Gültigkeit der Gleichungen (3) des vorigen Paragraphen. Auch die Herstellung der zweiten Derivirten und aller Derivirten höherer Ordnung kann durch Differentiation unter dem Integralzeichen ausgeführt werden, weil die Resultate dieser Differentiation bestimmte endliche Werthe haben, die bei einer stetigen Verschiebung des Punktes (x, y, z) sich ebenfalls stetig ändern.

Wenn aber der Punkt (x, y, z) im Innern der anziehenden Masse liegt, so behalten zwar, wie später (§§. 6. 10.) gezeigt werden soll, die durch die Gleichungen (4) und (5) des §. 2 definirten Functionen V, X, Y, Z bestimmte endliche Werthe, und es gelten deshalb auch noch die Gleichungen (3) desselben Paragraphen. Aber die Integrale auf der rechten Seite der Gleichungen (1) des gegenwärtigen Paragraphen haben dann gar keine Bedeutung, weil in einem Element der Integration ein unendlich grosser Factor auftritt. Liegt also der angezogene Punkt (x, y, z) im Innern der anziehenden Masse, so sind die Gleichungen (1) nicht gültig und ebenso wenig die Gleichung (2). Für diesen Fall ist vielmehr eine besondere Untersuchung anzustellen.

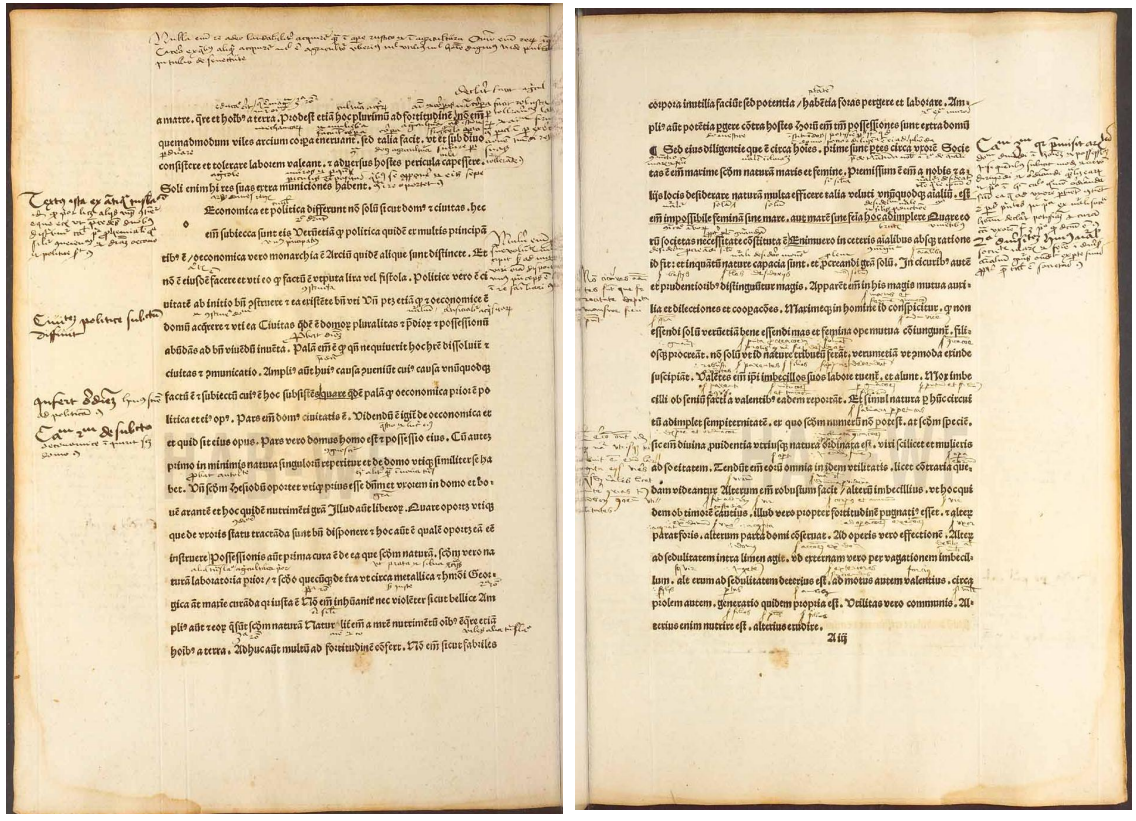
§. 4.

Specieller Fall: Anziehung einer Kugelschale, deren Dichtigkeit nur vom Radius vector abhängt.

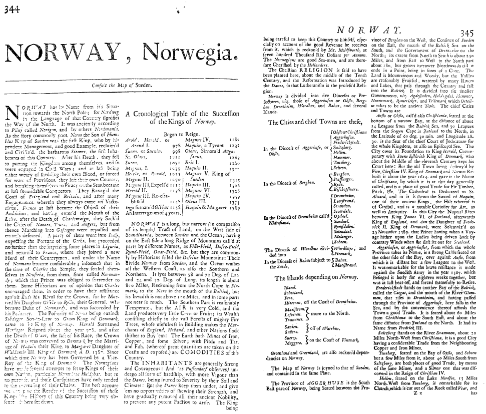
Wir wollen zunächst die Gleichung von Laplace benutzen, um in einem speciellen Falle die Potentialfunction zu berechnen.

Die anziehende Masse sei stetig vertheilt im Innern einer Kugelschale, d. h. des Raumes zwischen zwei concentrischen Kugelflächen. Den Anfangspunkt der Coordinaten legen wir in den Mittelpunkt der Kugeln. Die Dichtigkeit der anziehenden Masse sei dieselbe in allen Punkten einer zu der Begrenzung concentrischen Kugelfläche. Sie ändere sich nur mit dem Abstande vom Mittelpunkte. Dann ist auch die gesuchte Potentialfunction V nur abhängig von dem Radius vector s, und die partielle Diffe-

- Der dargestellte Text ist aus dem 1880 erschienenen Werk *Schwere, Elektricität und Magnetismus* von Bernhard Riemann.
- Hinweis: Auszeichnungen durch Sperrungen werden heute nicht mehr als empfehlenswert erachtet. Zu bevorzugen sind aktive Auszeichnungen (z.B. durch Fettschrift) oder integrierte Auszeichnungen (z.B. durch eine kursive Schrift oder die Verwendung von Kapitälchen).



- Die Doppelseite entstammt dem Werk *Arestotelis Stagirite Philozophorum maximi oeconomicorum libri duo sub gemina translatione*, das 1499 von Martin Landsberg in Leipzig gedruckt wurde.
- Bemerkenswert sind die großen Zeilenabstände, die zwar schlecht für das lineare Lesen sind, sich jedoch als praktisch für die intensive Arbeit mit dem Text erweisen, so dass bequem Notizen und Anmerkungen hinzugefügt werden können.


344

NORWAY, Norwegia.

Conſult the Map of Sweden.

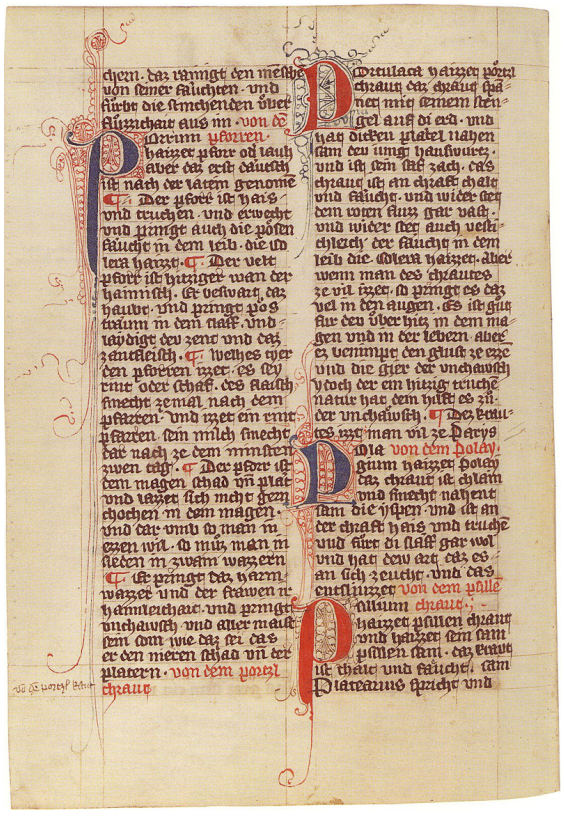
A Cronological Table of the Succeſſion of the Kings of *Norway*.

NORWAY. 345

The Cities and chief Towns are theſe,

The Iſlands depending on *Norway*.

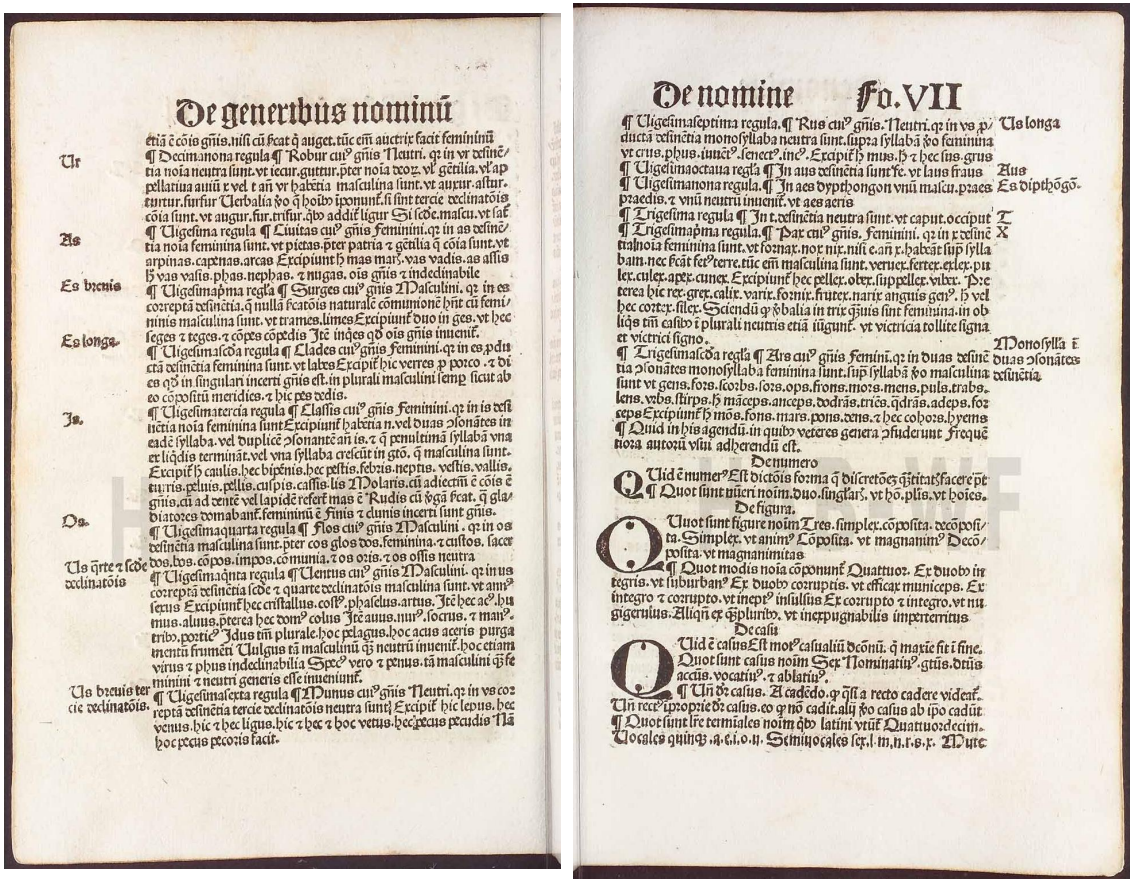
- Ziel des *informierenden Lesens* ist das schnelle Finden interessanter Textteile. Die Überschriften und die Strukturierung sollen eine schnelle Entscheidung ermöglichen, was sich zu lesen lohnt.
- Die zwei Seiten sind aus dem Werk *A system of geography: or, a new & accurate description of the earth in all its empires, kingdoms and states* des Geographen Herman Moll, London 1701.



Konrad von Megenberg,
Buch der Natur, 1350

- Informierendes Lesen soll diagonales Lesen ermöglichen.
- Der Leser soll also nicht genötigt werden, das gesamte Werk durchzulesen.
- Entsprechend sollte jeder Abschnitt deutlich lesbare Überschriften tragen, die einen guten Hinweis auf den Inhalt des Abschnitts liefern.
- Gelegentlich übernehmen auch Bilder die Funktion als Ansprungspunkt für zugehörige Texte.
- Zeitungen, Sachbücher, Handbücher und Ratgeber sind typisch für das informierende Lesen.

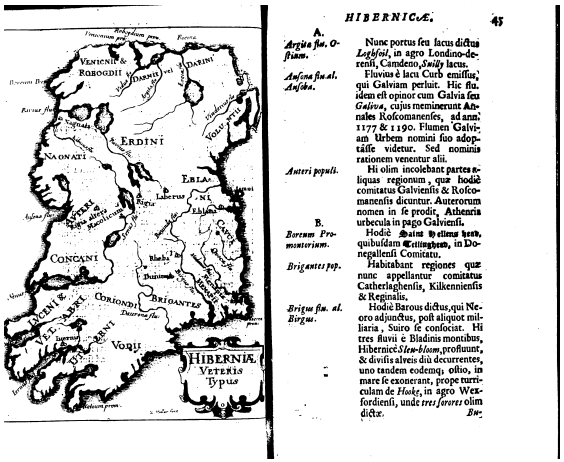
- Der Text wird optisch klar gegliedert. Entsprechend dürfen die Abschnitte klar voneinander abgesetzt werden, auch wenn dies den Textblock auseinanderreißt.
- Aktive Auszeichnungen sind erwünscht. Entsprechend sollten die Überschriften und wichtige Stichworte innerhalb des Textes beispielsweise mit Fettschrift hervorgehoben werden.
- Alternativ kann auch der Rand genutzt werden, um dort zusätzliche Überschriften unterzubringen.
- Ein mehrspaltiger Satz, auch mit drei oder mehr Kolumnen, kann sinnvoll sein.
- Ausführliche Bildunterschriften, die möglicherweise auch mehrspaltig gesetzt werden, sind zugelassen.
- Im übrigen sollte die Typografie der des *linearen Lesens* entsprechen.



De generibus nominū

De nomine Fo. VII

- Die Doppelseite entstammt dem Werk *Grammatica Nicolai perotti cum additionibus regularum. et metrice artis Guarini veronensis perfacundi viri lucidissime perspecta* von Niccolò Perotti (1429–1480), 1501 gedruckt von Heinrich Quentel.
- Bemerkenswert ist hier, wie die Marginalien eingesetzt werden, um grammatikalische Regeln leichter finden zu können.

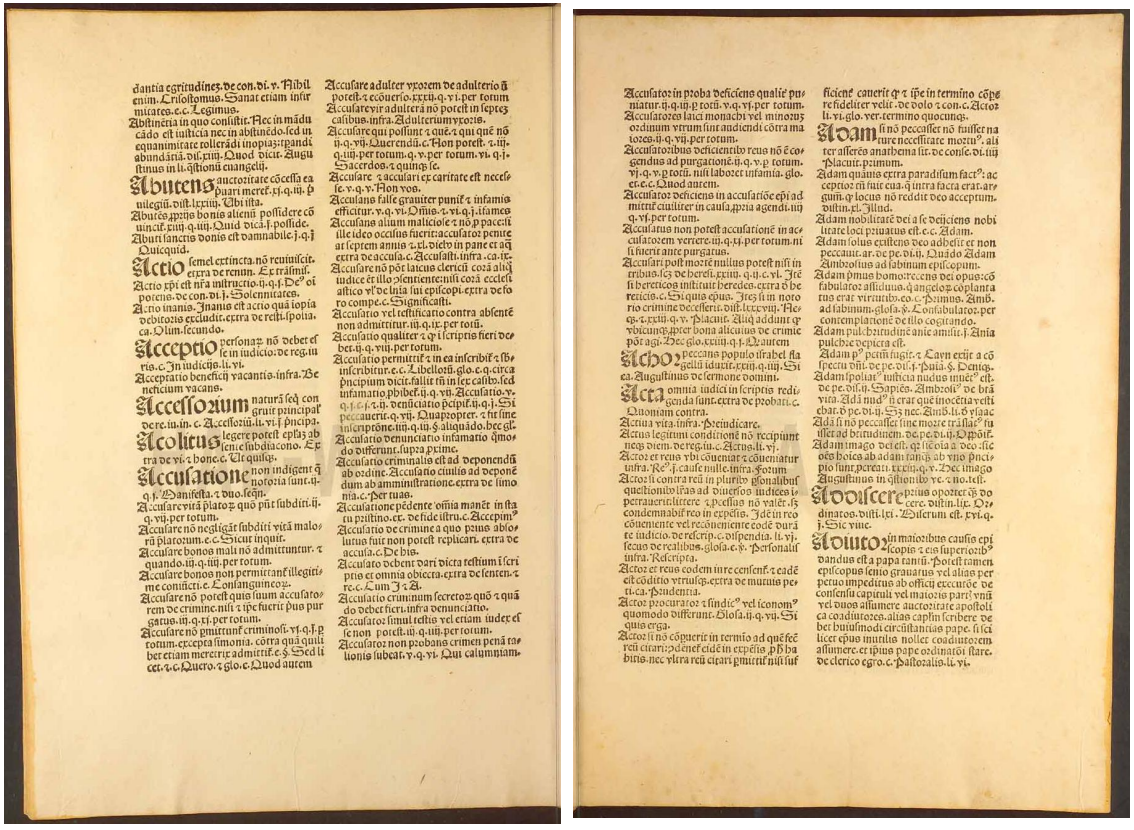


Sir James Ware,
De Hibernia & antiquitatibus ejus disquisitiones, London 1654

- Ziel des *konsultierenden Lesens* ist das schnelle Finden von Informationen zu einem vorgegebenen Stichwort oder sonstigen Schlüssel.
- Damit das Nachschlagen zügig möglich ist, muss der Inhalt entsprechend geordnet bzw. durch ein Register ergänzt werden.
- Typisch ist das konsultierende Lesen für Lexika, Wörterbücher, Nachschlagewerke, Chronologien, Bibliographien und Referenz-Manuals.

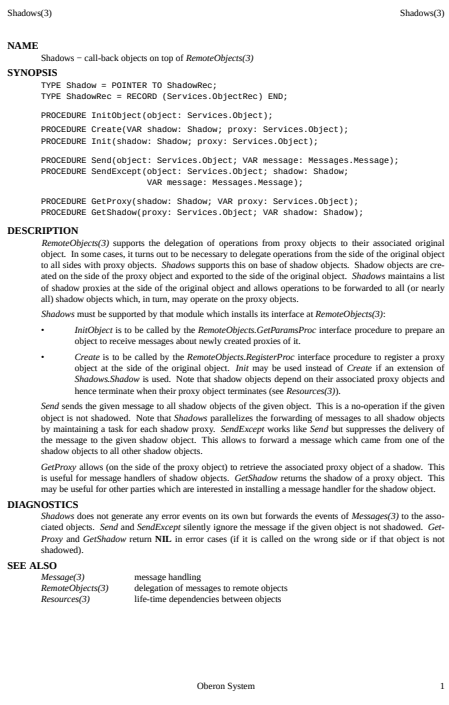
- Die Stichworte müssen so deutlich wie möglich aktiv ausgezeichnet werden.
- Alle anderen Auszeichnungen haben sich dem unterzuordnen und sollten daher möglichst integriert sein.
- Die Suche eines Stichworts sollte sinnvollerweise durch entsprechende Kolumnentitel erleichtert werden.
- Wenn die zu einem Stichwort gehörenden Texte eher kurz sind, empfiehlt sich eine eher kleine Schriftgröße und ein kleiner Zeilenabstand.
- Typisch ist eine gute Ausnutzung des zur Verfügung stehenden Raums und ein mehrspaltiger Satz.

Typografie des konsultierenden Lesens



- Die Doppelseite entstammt dem Werk *Compendium iuris canonici*, das 1499 von Georg Husner in Straßburg gedruckt wurde.
- Bemerkenswert ist hier, dass die Stichworte genau dann besonders groß gesetzt sind, wenn die ersten drei Buchstaben zum ersten Male auftreten.

Kombination des konsultierenden und informierenden Lesens



Shadows(3) Shadows(3)

NAME

Shadows − call-back objects on top of *RemoteObjects(3)*

SYNOPSIS

```
TYPE Shadow = POINTER TO ShadowRec;
TYPE ShadowRec = RECORD (Services.ObjectRec) END;

PROCEDURE InitObject(object: Services.Object);

PROCEDURE Create(VAR shadow: Shadow; proxy: Services.Object);
PROCEDURE Init(shadow: Shadow; proxy: Services.Object);

PROCEDURE Send(object: Services.Object; VAR message: Messages.Message);
PROCEDURE SendExcept(object: Services.Object; shadow: Shadow;
                     VAR message: Messages.Message);

PROCEDURE GetProxy(shadow: Shadow; VAR proxy: Services.Object);
PROCEDURE GetShadow(proxy: Services.Object; VAR shadow: Shadow);
```

DESCRIPTION

RemoteObjects(3) supports the delegation of operations from proxy objects to their associated original object. In some cases, it turns out to be necessary to delegate operations from the side of the original object to all sides with proxy objects. *Shadows* supports this on base of shadow objects. Shadow objects are created on the side of the proxy object and exported to the side of the original object. *Shadows* maintains a list of shadow proxies at the side of the original object and allows operations to be forwarded to all (or nearly all) shadow objects which, in turn, may operate on the proxy objects.

Shadows must be supported by that module which installs its interface at *RemoteObjects(3)*:

- *InitObject* is to be called by the *RemoteObjects.GetParamsProc* interface procedure to prepare an object to receive messages about newly created proxies of it.
- *Create* is to be called by the *RemoteObjects.RegisterProc* interface procedure to register a proxy object at the side of the original object. *Init* may be used instead of *Create* if an extension of *Shadows.Shadow* is used. Note that shadow objects depend on their associated proxy objects and hence terminate when their proxy object terminates (see *Resources(3)*).

Send sends the given message to all shadow objects of the given object. This is a no-operation if the given object is not shadowed. Note that *Shadows* parallelizes the forwarding of messages to all shadow objects by maintaining a task for each shadow proxy. *SendExcept* works like *Send* but suppresses the delivery of the message to the given shadow object. This allows to forward a message which came from one of the shadow objects to all other shadow objects.

GetProxy allows (on the side of the proxy object) to retrieve the associated proxy object of a shadow. This is useful for message handlers of shadow objects. *GetShadow* returns the shadow of a proxy object. This may be useful for other parties which are interested in installing a message handler for the shadow object.

DIAGNOSTICS

Shadows does not generate any error events on its own but forwards the events of *Messages(3)* to the associated objects. *Send* and *SendExcept* silently ignore the message if the given object is not shadowed. *GetProxy* and *GetShadow* return **NIL** in error cases (if it is called on the wrong side or if that object is not shadowed).

SEE ALSO

Message(3)	message handling
RemoteObjects(3)	delegation of messages to remote objects
Resources(3)	life-time dependencies between objects

Oberon System 1

- Die UNIX-Manualseiten adressieren gleichzeitig das konsultierende und das informierende Lesen.
- Zunächst kann über einen Begriff (wie etwa einem Funktions- oder Modulnamen) die zugehörige Dokumentation gefunden werden.
- Innerhalb einer Manualseite wird das diagonale Lesen durch eine weitgehend standardisierte Strukturierung und (bei längeren Texten) kleineren Überschriften unterstützt.

- Didaktisch aufgebaute Lehrbücher haben häufig mehrere Ebenen. Es gibt einführende Texte, zusätzliche Anmerkungen, die sich an Fortgeschrittene richten, Aufgaben und Zusammenfassungen, die den Stoff wieder schnell in Erinnerung bringen.
- Die Texte sind inhaltlich geordnet, so dass sich passend zu einer Thematik alle Ebenen eng beieinander zu finden sind.
- Der Leser möchte aber zu einem Zeitpunkt nur eine oder zwei Ebenen lesen, die er aus dem gesamten Text zu einer Thematik selektieren möchte.
- Neben Lehrbüchern kann das selektierende Lesen auch für kommentierte oder mehrsprachige Ausgaben sinnvoll sein, bei der Text und Kommentar bzw. die Übersetzung eng beieinander stehen.
- Fußnoten unterstützen ebenfalls das selektierende Lesen.

- Die unterschiedlichen Ebenen müssen auf den ersten Blick erkennbar sein.
- Entsprechend müssen die Ebenen typografisch deutlich unterschiedlich gestaltet werden. Denkbar sind
  - unterschiedliche Schriftschnitte (etwa Antiqua vs Grotesk),
  - unterschiedliche Schriftvarianten (etwa normal vs kursiv),
  - unterschiedliche Schriftgrößen,
  - optische Trennelemente, farbige Textunterlegungen oder die Verwendung spezieller Symbole.
- Wichtig ist die konsistente Verwendung eines gleichbleibenden Schemas, das nach Möglichkeit keine umfangreichen Erklärungen benötigt.

- Der erste bekannte Autor, der mit typografischen Maßen arbeitete, war Francesco Torniello (1490–1589) in seinem Werk *Opera del modo de fare le littere maiuscole antique*, das 1517 in Mailand erschien. Hier führte er ein 18x18-Raster ein, wobei die Länge zweier Rasterfelder als Punkt definiert wurden. Dies geriet jedoch in Vergessenheit.
- Die heutige Definition eines Punkts als Maßeinheit in der Typografie geht zurück auf Pierre-Simon Fournier (1712–1768), der 1737 das Werk *Table des proportions qu'il faut observer entre les caractères* veröffentlichte. Fournier definierte dabei ein Punkt als $\frac{1}{864}$ Pariser Fuß. Ins metrische System umgerechnet entspricht ein Punkt nach Fournier etwa 0,34 mm.
- 12 Punkte wurden zu einem Cicero zusammengefasst.

- Firmin Didot (1764–1836) entwickelte das System von Fournier weiter und wechselte vom Pariser Fuß zum *pied de roi* einem zu seiner Zeit gebräuchlicheren Maß. Ein Didot-Punkt entspricht 0,376 mm.
- Hermann Berthold (1831–1904) passte das System dann an das metrische System an. Seiner Definition nach ist ein Punkt genau $\frac{1}{2660}$ Meter (ca. 0,376 mm). Ein Cicero nach Berthold betrug ca. 4,511 mm.
- 1978 wurde der Didot-Punkt ein weiteres Mal neu definiert auf genau 0,375 mm. Ein Cicero entspricht dann genau 4,5 mm.

- In Nordamerika hat sich jedoch das 1886 eingeführte Maß des Pica-Punkts durchgesetzt. Hier entspricht ein Inch 72,27 Punkten, so dass ein Punkt ca. 0,351 mm gleicht. Statt dem Begriff des Cicero wird der des Pica verwendet, der umgerechnet ca. 4,22 mm entspricht.
- Adobe vereinfachte in der digitalen Typografie die Definition des Punkts auf $\frac{1}{72}$ Inch. Das sind ca. 0,0139 Inch und ca. 0,353 mm. Dies wurde zum Standard in der digitalen Typografie, weswegen er auch DTP-Punkt genannt wird. Ein Pica entspricht dann genau $\frac{1}{6}$ Inch; das sind ca. 4,23 mm. (In TEX entspricht ein *pt* jedoch noch der klassischen Definition und *bp* – für *big point* stehend – folgt der Definition von Adobe.)

Der Punkt wird primär für die Messung der Schriftgröße herangezogen. Traditionell werden die einzelnen Schriftgrößen mit Bezeichnungen verbunden:

	Didot-Punkt / deutsch	Pica-Punkt / englisch
3 pt	Brillant	Excelsior
4 pt	Diamant	Brilliant
5 pt	Perl	Pearl
6 pt	Nonpareille	Nonpearl
7 pt	Kolonel	Minion
8 pt	Petit	Brevier
9 pt	Borgis	Bourgeois
10 pt	Korpus/Garmond	Long Primer
11 pt	Rheinländer	Small Pica
12 pt	Cicero	Pica
14 pt	Mittel	English
16 pt	Tertia	Columbian

- Die Schriftgrößen in Punkten orientieren sich an den Kegelgrößen des Bleisatzes. In der digitalen Typografie wurden diese Größen in etwa beibehalten, obwohl sie nicht ableiten lassen von der Bounding-Box oder den sonstigen metrischen Angaben eines Schriftschnitts.
- Deswegen gibt es einige Maße, die sich direkt an den Schriftschnitten einzelner Buchstaben orientieren:
  - Die **Versalhöhe** gibt die typische Höhe eines Großbuchstabens (etwa der des »H«) an.
  - Die **Vertikalhöhe** spezifiziert die maximale vertikale Ausdehnung, die sich etwa bei der Buchstabenkombination »hp« messen lässt.
  - Die **x-Höhe** misst die Höhe der Kleinbuchstaben, wobei typischerweise das »x« selbst genommen wird.

- Gegeben seien
  - zu formatierender Text,
  - die Metriken und weiteren Tabellen der benötigten Schriftschnitte,
  - diverse Rahmenparameter (z.B. der Satzspiegel) und
  - Trennungstabellen.
- Zu erzeugen ist eine druckbare Ausgabe.

Um diese Probleme zu lösen, werden Algorithmen und Datenstrukturen benötigt,

- um den Text in Wörter zu gliedern (zusammenhängende Zeichenfolge ohne Leerraum, die einheitlich gesetzt ist),
- geeignete Trennungsstellen in Wörtern zu finden,
- um unmittelbar aufeinanderfolgende Eingabezeichen (also Wörter) in eine Folge von Ausgabezeichen abzubilden (z.B. Berücksichtigung von Ligaturen) und diese relativ zueinander zu positionieren (Kerning),
- um Paragraphen in Zeilen zu zerlegen und
- um eine Folge von Paragraphen in Seiten aufzuteilen.

Voraussetzung für die Algorithmen sind geeignete Datenstrukturen und Schnittstellen, die für einen ausgewählten Schriftschnitt die benötigten Metriken und Tabellen bereitstellen.

- FreeType bietet eine Schnittstelle zu Schriftschnitten und für die Rasterisierung. Eine Unterstützung des Kerning gehört nicht dazu. Das liegt daran, dass die Bibliothek sich auf die Rasterisierung konzentriert, wobei Kerning keine Rolle mehr spielt.
- HarfBuzz ist eine Bibliothek, die deutlich über das einfache Kerning hinausgeht und entsprechend der modernen Schrifttechnologien in der Lage ist, Zeichen einer Zeichenfolge zu ersetzen, zu kombinieren und zu platzieren. Solche Werkzeuge werden *shaping engines* genannt.

Für Java gibt es an mehreren Stellen Bibliotheken, die mit Metriken arbeiten:

- Die Klasse *java.awt.FontMetrics* bietet eine Methode namens *charWidth* an, unterstützt jedoch kein Kerning. Zwar lässt sich seit Java 6 die Unterstützung von Kerning und Ligaturen einschalten, aber die Bibliothek gewährt keinen Zugang zu den entsprechenden Datenstrukturen. Ferner betrifft dies nur von Java unmittelbar unterstützte Schriften, die auch dargestellt werden können.
- Das FOP-Projekt (Apache Formatting Objects Processor), das sich zum Ziel setzt, XML-Repräsentierungen in PDF abzubilden, unterstützt das Laden diverser Schriftrepräsentationen, darunter auch Adobe Type 1.
- Das sfntly-Projekt von Google bietet Schnittstellen für OpenType und TrueType. Jedoch sehe ich da noch keine unmittelbare Unterstützung für das Kerning.

FontMetrics.java

```
package de.uniulm.mathematik.typo.afm;

public interface FontMetrics {
   public String getName();
   public String getName(int code);
   public int minCode();
   public int maxCode();
   public int length();
   public boolean defined(int code);
   public int[] getBoundingBox();
   public int[] getBoundingBox(int code);
   public int getVersalHeight();
   public int getXHeight();
   public int getDescender();
   public int getAscender();
   public int getWidth(int code);
   public int getKerning(int code1, int code2);
}
```

- Ein Schriftschnitt kommt mit einer Kodierung. Kodiert wird mit ganzen Zahlen aus dem Bereich *minCode()* bis *maxCode()*, wobei naiverweise angenommen wird, dass die Ordinalwerte druckbarer ASCII-Zeichen direkt verwendet werden können.
- Ein Unterstützung alternativer Kodierungen (manche Schriftschnitte haben Zeichen, die nur über Umkodierungen erreichbar sind) fehlt.
- Für jedes einzelne Zeichen kann die Weite (*getWidth()*), die Bounding-Box und der Name abgefragt werden. (Namen sind die Voraussetzung für Umkodierungen.)
- Die Unterstützung für Ligaturen und die Unterstützung für kombinierte Zeichen fehlen. (Hierfür wäre eine fortgeschrittene *shaping engine* notwendig.)

Beispielhaft wird diese Schnittstelle implementiert durch eine Unterstützung der AFM-Dateien (Adobe Font Metrics), die folgende Vorteile haben:

- Das Format der AFM-Dateien ist öffentlich spezifiziert, siehe die *Adobe Font Metrics File Format Specification*.
- Da die AFM-Dateien nicht-binär als ASCII-Text repräsentiert sind, die einer gewissen Syntax genügen, können sie sowohl bequem direkt gelesen werden als auch mit überschaubaren Aufwand syntaktisch analysiert und in passende Datenstrukturen überführt werden.

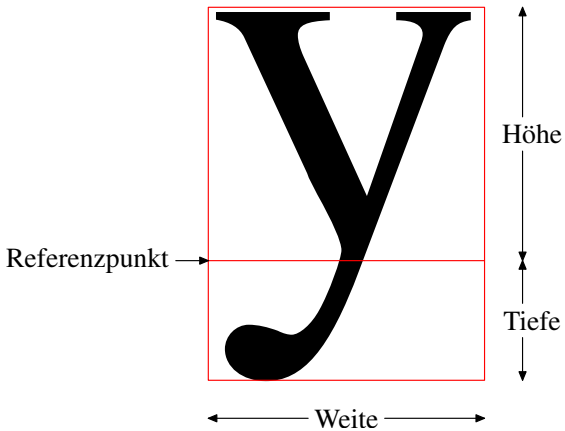
TestAV.java

```
public class TestAV {
   private final static String AFM_INPUT =
      "/usr/local/texlive/2015/texmf-dist/fonts/afm/adobe/times/ptmr8a.afm";
   public static void main(String[] args) {
      try {
         FontMetrics fm = new AdobeFontMetrics(AFM_INPUT);
         System.out.println("width of A = " + fm.getWidth('A'));
         System.out.println("width of V = " + fm.getWidth('V'));
         System.out.println("kerning = " + fm.getKerning('A', 'V'));
      } catch (Exception e) {
         e.printStackTrace();
      }
   } // main
} // class TestFontMetrics
```

- Die Klasse *AdobeFontMetrics* implementiert die Schnittstelle *FontMetrics* und bietet dabei zwei Konstruktoren an, die entweder die Angabe eines Dateinamens (wie hier) oder eines *BufferedReader* unterstützen.

```
clonard$ TestAV
width of A = 722
width of V = 722
kerning = -135
clonard$
```

- Es wurde hier die Konvention von Adobe übernommen, mit Einheiten zu arbeiten, die einem 1/1000 Punkt entsprechen.
- Das erlaubt es, weitgehend mit ganzzahligen Werten zu arbeiten.
- TEX macht das ähnlich, arbeitet aber mit Einheiten, die $\frac{1}{65536}$ pt entsprechen, wobei in TEX die Einheit pt dem nordamerikanischen Pica-Punkt entspricht.



- Die fundamentale Datenstruktur für die Verarbeitung von Texten sind Schachteln.
- Jedes Zeichen wird nur in Form einer Schachtel repräsentiert, die eine gewisse Weite, Höhe und Tiefe hat.
- Auf der Höhe des Referenzpunkts verläuft die Basislinie.
- Die Angaben werden der Metrik entnommen, die zu der Schriftform gehört.
- Eine Schachtel ist jedoch keine strenge Bounding-Box, d.h. es ist sehr wohl möglich, dass die Darstellung eines Zeichens die Grenzen der Schachtel überschreitet.
- Die Gestaltung des Zeichens ist uns hier ohnehin nicht bekannt.

- Im horizontalen Modus werden die Schachteln horizontal auf der Höhe der Basislinien aneinandergereiht.
- Aneinandergereihte Schachteln werden wiederum als Schachteln betrachtet.
- Längere horizontale Schachtelketten, die zu einem Paragraphen gehören, müssen an geeigneten Stellen auseinandergebrochen werden.
- Ein gebrochener Paragraph ist dann eine vertikale Aneinanderreihung horizontaler Schachtelketten.
- Eine Folge von vertikalen Schachtelketten muss an geeigneten Stellen in Seiten gebrochen werden.
- Ziel typografischer Algorithmen ist es, eine Texteingabe in ein geeignetes Schachtelsystem zu überführen.

Neben den regulären Schachteln werden für die typografischen Algorithmen noch spezielle Schachteln benötigt, die der Analyse der zur Verfügung stehenden Spielräume und Trennungsmöglichkeiten dienen:

- Statt Leerzeichen mit fester Länge werden **Dehnfugen** (*glue*) verwendet, die im Normalfall einen Leerraum passend zum gewählten Schriftschnitt einnehmen, bei Bedarf aber auch etwas gestaucht oder gestreckt werden können.
- **Sollbruchstellen** markieren Stellen, bei denen eine horizontale Aneinanderreihung von Schachteln gebrochen werden kann. Im Falle von Dehnfugen ist es offensichtlich. Es gibt aber auch potentielle Trennungen, bei denen ein Trennungszeichen eingefügt wird und die einen Strafwert besitzen, weswegen Knuth sie *penalties* nannte.

Item.java

```
public interface Item {
   public final int INFINITY = Integer.MAX_VALUE / 2;
   public boolean isPenalty(); public boolean isGlue();
   public boolean isBox();
   public abstract int getWidth();
   public int getStretchability(); public int getShrinkability();
   public void shrink(int units); public void stretch(int units);
   public int getHeight(); public int getDepth();
   public int getPenalty(); public boolean getPenaltyFlag();
   public StringBuffer genPostScript(PostScriptContext context);
}
```

- Dies ist die abstrakte Schnittstelle für Schachteln einschließlich spezieller Varianten, die im folgenden verwendet wird. (Es fehlt die Unterstützung vertikaler Dehnfugen.)

isPenalty()	handelt es sich um eine Sollbruchstelle?
isGlue()	handelt es sich um eine Dehnfuge?
isBox()	handelt es sich um eine normale Schachtel?
getWidth()	horizontale Weite dieser Schachtel
getStretchability()	um wieviel Einheiten darf gedehnt werden?
getShrinkability()	um wieviel Einheiten darf gestaucht werden?
stretch()	Auseinanderziehen der Schachtel
shrink()	Zusammenstauchen der Schachtel
getHeight()	Höhe der Schachtel oberhalb der Basislinie
getDepth()	Tiefe der Schachtel unterhalb der Basislinie
getPenalty()	Je größer der Wert ist, umso unschöner ist eine Brechung an dieser Stelle
getPenaltyFlag()	konsekutive Trennungen an Stellen mit gesetztem Flag sind besonders unschön
genPostScript()	Generierung von PostScript für die Schachtel, wobei die Anfangsposition am Basispunkt liegt und der Endpunkt um die Weite nach rechts versetzt liegt.

- Diese Repräsentierung orientiert sich weitgehend an die des Artikels von Donald E. Knuth: *Breaking Paragraphs Into Lines*, der ursprünglich in 1981 in *Software–Practice and Experience* erschienen ist und innerhalb des Buchs *Digital Typography* 1999 nachgedruckt wurde.
- Hinzugekommen ist jetzt nur die Generierung von PostScript, das für unsere Beispiele die bequemste Ausgabe-Möglichkeit darstellt.
- Ein Knuth folgender Ansatz findet sich auch in der FOP-Klassenbibliothek im Paket *org.apache.fop.layoutmgr* mit der abstrakten Basisklasse *KnuthElement* und den davon abgeleiteten Klassen *KnuthBox*, *KnuthGlue* und *KnuthPenalty*.

Box.java

```
public abstract class Box implements Item {
   final public boolean isPenalty() { return false; }
   final public boolean isGlue() { return false; }
   final public boolean isBox() { return true; }
   public abstract int getWidth();
   public int getStretchability() { return 0; }
   public int getShrinkability() { return 0; }
   public void shrink(int units) {}
   public void stretch(int units) {}
   public abstract int getHeight();
   public abstract int getDepth();
   final public int getPenalty() { return Item.INFINITY; }
   public boolean getPenaltyFlag() { return false; }
   public abstract StringBuffer genPostScript(PostScriptContext context);
}
```

- *Box* ist eine abstrakte Klasse auf Basis der Schnittstelle *Item* für reguläre Schachteln ohne spezielle Trennungs- oder Dehnungseigenschaften.

GlyphBox.java

```
public class GlyphBox extends Box {
   private FontMetrics fm;
   private int code; private int size;
   private int[] bbox;

   public GlyphBox(FontMetrics fm, int size, int code) {
      this.fm = fm; this.size = size; this.code = code;
      bbox = fm.getBoundingBox(code);
   }
   public int getWidth() { return fm.getWidth(code) * size; }
   public int getHeight() { return bbox[3] * size; }
   public int getDepth() { return bbox[1] * size; }
   public StringBuffer genPostScript(PostScriptContext context) {
      return context.addTo(fm, size, code);
   }
}
```

- Für ein einzelnes Zeichen genügen uns die Infos aus der zugehörigen Metrik.

PostScriptContext.java

```
public class PostScriptContext {
   public PostScriptContext();
   public FontMetrics getCurrentFont(); public int getCurrentSize();
   public boolean insideString(); public StringBuffer closeString();
   public StringBuffer gsave(); public StringBuffer grestore();
   public StringBuffer switchTo(FontMetrics fm, int size);
   public StringBuffer addTo(FontMetrics fm, int size, int code) {
} // class PostScriptContext
```

- Um nicht vor jedem Zeichen den aktuellen Schriftschnitt auszuwählen und mehrere hintereinander kommende Einzelzeichen ohne Kerning zusammengefasst ausgeben zu müssen, steht mit *PostScriptContext* eine Klasse zur Verfügung, die diese Koordinierung übernimmt.

PostScriptContext.java

```
public StringBuffer switchTo(FontMetrics fm, int size) {
   StringBuffer result = new StringBuffer("");
   if (newcontext || fm != currentFont || size != currentSize) {
      result.append(closeString()); result.append("/");
      result.append(fm.getName()); result.append(" findfont ");
      result.append(size); result.append(" scalefont setfont\n");
      currentFont = fm; currentSize = size; newcontext = false;
   }
   return result;
}
```

- *switchTo()* generiert PostScript-Text, der den gewünschten Schriftschnitt auswählt, falls dieser sich verändert haben sollte.

PostScriptContext.java

```
public StringBuffer addTo(FontMetrics fm, int size, int code) {
   StringBuffer result = new StringBuffer("");
   result.append(switchTo(fm, size));
   if (!inString) {
      inString = true; result.append("(");
   }
   if (code == '(' || code == ')' || code == '\\') {
      result.append('\\');
      result.append(Integer.toOctalString(code));
   } else {
      result.append((char) code);
   }
   return result;
}
```

- *addTo()* fügt ein einzelnes Zeichen hinzu. Falls wir uns bereits in einer PostScript-Zeichenkette befinden, muss dabei nur das Zeichen selbst ausgegeben werden.

Schachteln für das Kerning

KerningBox.java

```
public class KerningBox extends Box {
   private int kerning;

   public KerningBox(int kerning) { this.kerning = kerning; }
   public int getWidth() { return kerning; }
   public int getHeight() { return 0; }
   public int getDepth() { return 0; }

   public StringBuffer genPostScript(PostScriptContext context) {
      StringBuffer result = context.closeString();
      result.append((float) kerning / 1000);
      result.append(" 0 rmoveto\n");
      return result;
   }
}
```

- Das Kerning wird durch Schachteln repräsentiert, die nur eine Weite haben, die im Falle des Zusammenrückens negativ ist.
- Umgesetzt in PostScript wird das Kerning durch eine relative Bewegung mit *rmoveto*.

Glue.java

```
public class Glue implements Item {
   private int originalWidth; private int width;
   private int stretchability; private int shrinkability;

   private int intoRange(int value) {
      if (value > Item.INFINITY) {
         return Item.INFINITY;
      } else if (value < - Item.INFINITY) {
         return -Item.INFINITY;
      } else {
         return value;
      }
   }
   public Glue(int width, int stretchability, int shrinkability) {
      this.width = intoRange(width);
      originalWidth = this.width;
      assert(stretchability >= 0);
      this.stretchability = intoRange(stretchability);
      assert(shrinkability >= 0);
      this.shrinkability = intoRange(shrinkability);
   }
   // ...
}
```

Glue.java

```
final public boolean isGlue() { return true; }
final public boolean isPenalty() { return false; }
final public boolean isBox() { return false; }
public int getWidth() { return width; }
public int getStretchability() { return stretchability; }
public int getShrinkability() { return shrinkability; }
public void stretch(int units) { /* ... */ }
public void shrink(int units) { /* ... */ }
public int getHeight() { return 0; }
public int getDepth() { return 0; }
public int getPenalty() { return 0; }
final public boolean getPenaltyFlag() { return false; }

public StringBuffer genPostScript(PostScriptContext context) {
   StringBuffer result = context.closeString();
   if (width != 0) {
      result.append((double) width / 1000);
      result.append(" 0 rmoveto\n");
   }
   return result;
}
```

Penalty.java

```
public class Penalty implements Item {
   private int penalty;
   private boolean flag;

   private int intoRange(int value) {
      if (value > Item.INFINITY) {
         return Item.INFINITY;
      } else if (value < - Item.INFINITY) {
         return -Item.INFINITY;
      } else {
         return value;
      }
   }

   public Penalty(int penalty, boolean flag) {
      this.penalty = intoRange(penalty); this.flag = flag;
   }
   // ...
}
```

Penalty.java

```
final public boolean isPenalty() { return true; }
final public boolean isGlue() { return false; }
final public boolean isBox() { return false; }
public int getWidth() { return 0; }
public int getStretchability() { return 0; }
public int getShrinkability() { return 0; }
public void shrink(int units) {}
public void stretch(int units) {}
public int getHeight() { return 0; }
public int getDepth() { return 0; }
public int getPenalty() { return penalty; }
public boolean getPenaltyFlag() { return flag; }

public StringBuffer genPostScript(PostScriptContext context) {
   return new StringBuffer("");
}
```

HorizontalBox.java

```
public class HorizontalBox extends Box {
   private LinkedList<Item> items;
   private int width; private int height; private int depth;

   public HorizontalBox() {
      items = new LinkedList<Item>();
      width = 0; height = 0; depth = 0;
   }
   public void add(Item item) {
      items.addLast(item);
      width += item.getWidth();
      if (item.getHeight() > height) {
         height = item.getHeight();
      }
      if (item.getDepth() > depth) {
         depth = item.getDepth();
      }
   }
   // ...
}
```

HorizontalBox.java

```
public class HorizontalBox extends Box {
   // ...

   public int getWidth() { return width; }
   public int getHeight() { return height; }
   public int getDepth() { return depth; }

   public StringBuffer genPostScript(PostScriptContext context) {
      StringBuffer result = new StringBuffer("");
      for (Item item:items) {
         result.append(item.genPostScript(context));
      }
      return result;
   }
}
```

- Horizontale Schachteln akkumulieren einzelne Schachteln hintereinander.
- Eine horizontale Schachtel wird später nicht mehr aufgebrochen. Sie entsteht vielmehr als Resultat eines zeilenbrechenden Algorithmus.

Vertikal zusammengesetzte Schachteln

VerticalBox.java

```
public class VerticalBox extends Box {
   private LinkedList<Item> items;
   private int width; private int height; private int depth;
   private int baselineskip;

   public VerticalBox(int baselineskip) {
      items = new LinkedList<Item>();
      width = 0; height = 0; depth = 0;
      this.baselineskip = baselineskip;
   }

   public void add(Item item) {
      items.addLast(item);
      if (item.getWidth() > width) {
         width = item.getWidth();
      }
      height += baselineskip; depth = item.getDepth();
   }
   public int getWidth() { return width; }
   public int getHeight() { return height; }
   public int getDepth() { return depth; }
   // ...
} // class VerticalBox
```

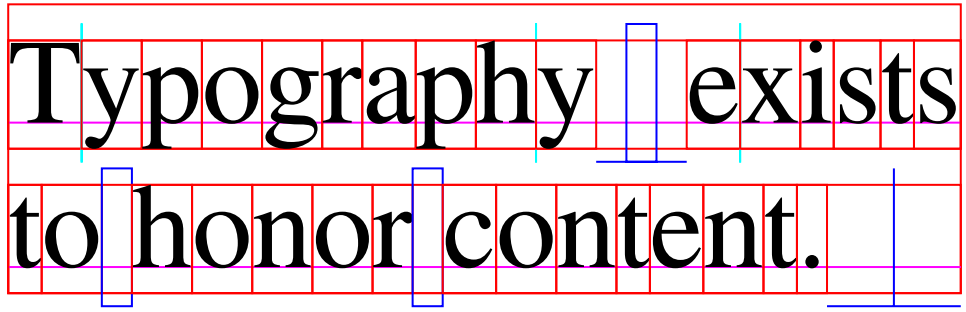
VerticalBox.java

```
public VerticalBox(int baselineskip) {
   items = new LinkedList<Item>();
   width = 0; height = 0; depth = 0;
   this.baselineskip = baselineskip;
}

public void add(Item item) {
   items.addLast(item);
   if (item.getWidth() > width) {
      width = item.getWidth();
   }
   height += baselineskip; depth = item.getDepth();
}
```

- Der *baselineskip* legt den vertikalen Abstand zwischen den Basislinien der einzelnen Zeilen fest.
- Das ist genau das, was bei einem Paragraphen gewünscht wird. Für andere Zwecke werden andere vertikale Schachtelsysteme benötigt, die sich an den vertikalen Höhen der einzelnen Schachteln orientieren.

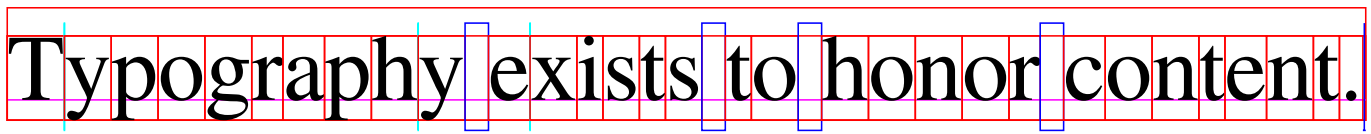
VerticalBox.java

```
public StringBuffer genPostScript(PostScriptContext context) {
   double bskip = (double) baselineskip / 1000;
   StringBuffer result = new StringBuffer("");
   if (items.size() > 0) {
      int i = 0;
      for (Item item:items) {
         result.append(context.gsave());
         if (i < items.size()-1) {
            result.append("0 ");
            result.append(bskip * (items.size()-1-i));
            result.append(" rmoveto\n");
         }
         result.append(item.genPostScript(context));
         result.append(context.closeString());
         result.append(context.grestore());
         ++i;
      }
      result.append((double) width / 1000);
      result.append(" 0 rmoveto\n");
   }
   return result;
}
```



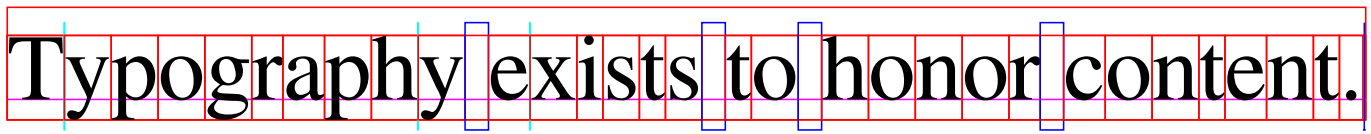
- Paragraphen werden durch vertikale Schachteln repräsentiert, bei denen horizontale Schachteln die einzelnen Zeilen einpacken, die wiederum aus den aneinandergereihten Zeichen, Kerning-Schachteln und Dehnfugen bestehen. Jede der Zeilen wurde an die gewünschte Paragraphenweite angepasst.
- Reguläre Schachteln sind in dieser Darstellung rot, Dehnfugen blau und Kerning-Schachteln hellblau. Die Schachteln der Zeichen wurden hier normalisiert (d.h. mit einheitlicher Tiefe und Höhe versehen), damit der Abstand zwischen der untersten Basislinie und der unteren Kante des Paragraphenblocks unabhängig davon ist, ob die unterste Zeile nach unten ragende Zeichen wie etwa ein »g« enthält oder nicht.

Ausgangssituation bei der Zerlegung eines Paragraphen 338

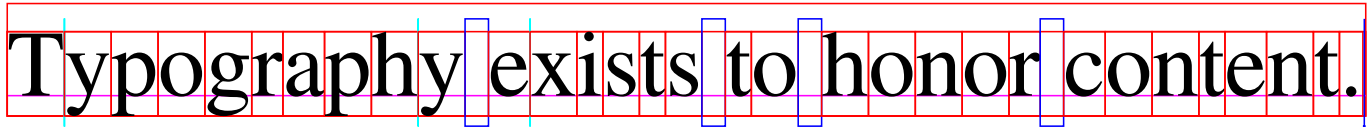


- Bevor eine Zerlegung eines Paragraphen beginnen kann, benötigen wir eine Datenstruktur, bei der alle Schachteln aufgereiht sind.
- Dies kann noch keine reguläre horizontale Schachtel sein, da horizontale Schachteln nicht mehr aufgebrochen werden können.

Ausgangssituation bei der Zerlegung eines Paragraphen 339



- Zu den einzelnen Elementen gehören
  - reguläre Schachteln, die beispielsweise ein Zeichen darstellen oder einen individuellen Abstand zwischen zwei aufeinanderfolgenden Zeichen regeln (Kerning),
  - Dehnfugen, die einen dehn- oder stauchbaren Leerraum repräsentieren ohne weitere sichtbare Darstellung und
  - Sollbruchstellen mit einem Strafwert als Häßlichkeitsmaß einer möglichen Trennung an dieser Stelle. Sollbruchstellen gibt es nach einer Folge von Binde- oder Gedankenstriche und bei potentiellen Trennstellen innerhalb von Wörtern.



- Nach der Definition von Donald E. Knuth gibt es genau zwei Arten zulässiger Trennungsstellen:
  - Sollbruchstellen mit einem Strafwert $< \infty$ und
  - Dehnungsfugen, die unmittelbar einer regulären Schachtel folgen.
- Die Eingabe des Zerlegungsalgorithmus kann auch als Folge von nicht trennbaren Schachteln betrachtet werden, denen jeweils eine zulässige Trennungsstelle folgt.
- Eine Sequenz nicht trennbarer Schachteln wird im folgenden Wort genannt.
- Damit Wörter immer durch eine zulässige Trennungsstelle terminiert werden, muss am Ende der Sequenz noch eine Trennungsstelle hinzugefügt werden. Hierfür bietet sich eine unendlich dehnbare Dehnfuge mit einer regulären Weite von 0 an.

HorizontalSequence.java

```
public interface HorizontalSequence extends Iterable<Item> {
   public void add(Item item);
   public void add(HorizontalSequence hseq);
   public Width getWidth();
   public IterableIterator<HorizontalSequence> getWords();
   public Item getFollowingBreakpoint();
   public IterableIterator<Item> getGlueItems();
   public HorizontalSequence clone();
} // class HorizontalSequence
```

- Sequenzen kann nur etwas hinzugefügt werden (entweder einzelne Schachteln oder andere Sequenzen).
- Der Datentyp *Width* vereinigt die reguläre Weite mit Hinweisen auf die Dehn- und Stauchbarkeit. Die Methode *getWidth()* liefert diese aufsummiert zurück für alle enthaltenen Schachteln.

Schnittstelle für Schachtel-Sequenzen

HorizontalSequence.java

```
public interface HorizontalSequence extends Iterable<Item> {
   public void add(Item item);
   public void add(HorizontalSequence hseq);
   public Width getWidth();
   public IterableIterator<HorizontalSequence> getWords();
   public Item getFollowingBreakpoint();
   public IterableIterator<Item> getGlueItems();
   public HorizontalSequence clone();
} // class HorizontalSequence
```

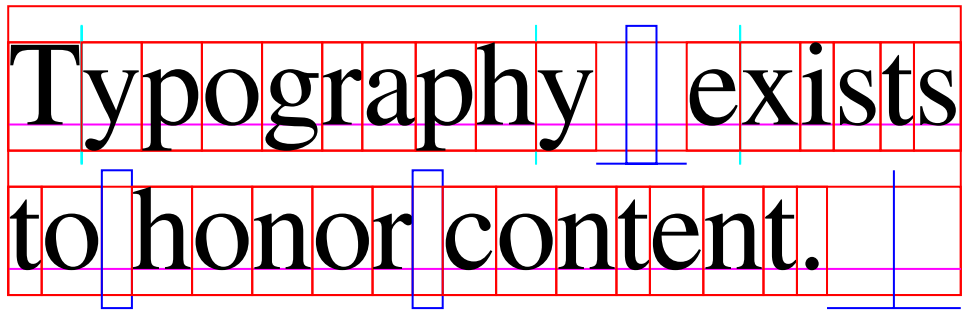
- Da diese Schnittstelle die Schnittstelle für *Iterable* beinhaltet, ist das Durchiterieren der einzelnen Schachteln möglich.
- Der Iterator *getWords* erlaubt es, über alle enthaltenen Worte zu iterieren. Wörter werden ebenfalls als Sequenzen repräsentiert, jedoch ohne die Trennungsstelle. Herausgegriffene Wörter erlauben den Zugriff auf die folgende Trennungsstelle mit der Methode *getFollowingBreakpoint()*.
- Wenn eine Sequenz an eine gegebene Weite durch Stauchen oder Dehnen anzupassen ist, dann liefert die Methode *getGlueItems()* die Dehnfugen.

- Für die Schnittstelle *HorizontalSequence* gibt es zwei Implementierungen: *SimpleHorizontalSequence* und die normalerweise nicht sichtbare *SimpleHorizontalSequence.SubSequence*.
- Die explizite Unterstützung von Subsequenzen erlaubt die Vermeidung der Duplikation von Datenstrukturen. Das hat zur Konsequenz, dass die Iteration mit *getWords* einschliesslich der Erzeugung der Subsequenzen für die einzelnen Worte nur einen Aufwand hat, der linear von der Zahl der Wörter abhängt und nicht deren Länge.
- Wenn Subsequenzen durch die *add*-Methoden verlängert werden, verändern sie nicht die zugrundeliegende Sequenz.

HorizontalFitter.java

```
public class HorizontalFitter {
   public static void fit(HorizontalSequence hseq, int width);
} // class HorizontalFitter
```

- Die Klasse *HorizontalFitter* offeriert nur eine statische Methode *fit()*, die versucht, die gegebene horizontale Sequenz an die vorgegebene Weite anzupassen, indem die zur Verfügung stehenden Dehnfugen angepasst werden.
- Wenn die Sequenz zu dehnen ist und die Sequenz Dehnfugen mit unendlich großen Dehnkapazitäten besitzt, dann wird die notwendige Dehnung gleichmäßig über alle unendlich dehnbaren Dehnfugen verteilt.
- Ansonsten wird die notwendige Dehnung oder Stauchung entsprechend der jeweiligen Dehn- oder Stauchkapazitäten gleichmäßig bei allen Dehnfugen durchgeführt.



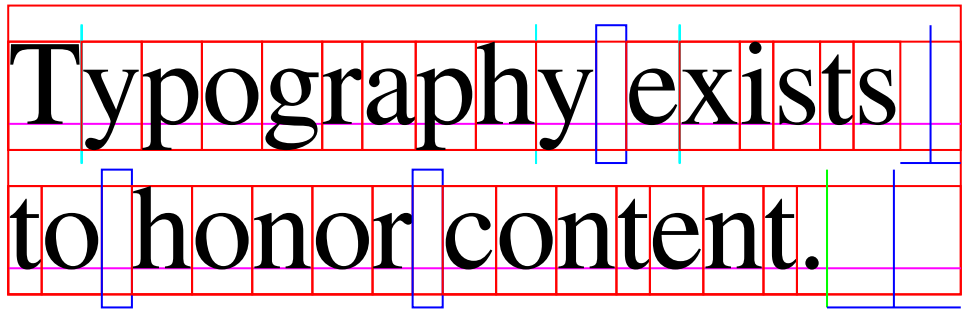
- Eine unendlich dehnbare Dehnfuge ist insbesondere am Ende eines Paragraphen sinnvoll, damit die letzte Zeile nicht bis auf die Paragraphenweite ausgedehnt wird.
- Wie an diesem Beispiel zu sehen ist, wurden die normalen Dehnfugen zwischen den Wörtern der letzten Zeile nicht ausgedehnt, sondern nur die unendlich dehnbare Dehnfuge am Ende des Paragraphen.

Schnittstelle für zeilenbrechende Algorithmen 346

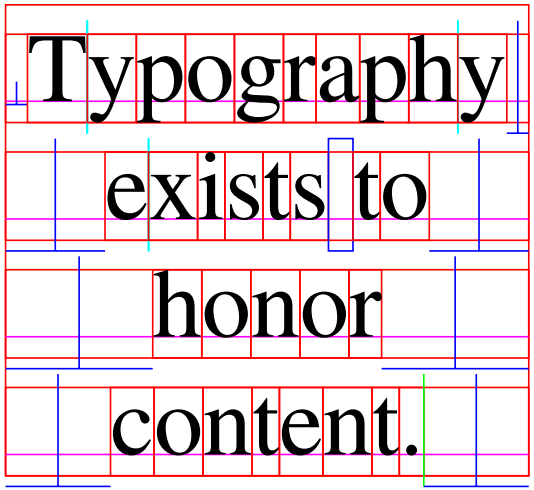
LineBreaker.java

```
public interface LineBreaker {
   public void setLineWrapper(SequenceWrapper wrapper);
   public void setHorizontalBoxWrapper(ItemWrapper hboxwrapper);
   public Item breakParagraph(HorizontalSequence hseq,
      int parwidth, int baselineskip);
} // interface LineBreaker
```

- Da es mehrere zeilenbrechende Algorithmen gibt, ist eine allgemeine Schnittstelle hierfür sinnvoll.
- Der eigentliche Algorithmus wird verpackt durch die Methode *breakParagraph()*, die eine horizontale Sequenz in Zeilen bricht, die einzelnen Zeilen an die vorgegebene Weite anpasst und diese dann mit dem gegebenen Zeilenabstand in einer vertikalen Box übereinander stapelt.
- Optional kann mit *setLineWrapper()* eine Bearbeitung einer Zeilensequenz vor der Anpassung an die vorgegebene Weite eingebaut werden. Ferner ist mit *setHorizontalBoxWrapper()* eine Nachbearbeitung der horizontalen Schachteln für die einzelnen Zeilen möglich.



- Mit *setLineWrapper()* ist es möglich, vom traditionellen Blocksatz abzuweichen.
- Wenn der Wrapper an die horizontalen Sequenzen jeweils eine Dehnfuge mit unendlicher Dehnkapazität anfügt, erhalten wir einen aufgerauhten rechten Rand.
- Der grüne Strich repräsentiert hier eine Sollbruchstelle, die an das Ende des Paragraphen angehängt wurde. Auf diese Weise wird auch das letzte Wort mit einer Trennstelle terminiert, wenn zu Beginn die sonst übliche unendlich dehnbare Dehnfuge am Ende des Paragraphen fehlt.



- Wenn durch den Wrapper für die einzelnen Zeilen jeweils am Anfang und Ende jeweils eine unendlich dehnbare Dehnfuge hinzugefügt wird, erhalten wir eine zentrierte Formatierung.

Basisklasse für zeilenbrechende Algorithmen

BasicLineBreaker.java

```
public abstract class BasicLineBreaker implements LineBreaker {
   protected SequenceWrapper wrapper;
   protected ItemWrapper hboxwrapper;

   public BasicLineBreaker() {
      wrapper = null; hboxwrapper = null;
   }

   public void setLineWrapper(SequenceWrapper wrapper) {
      this.wrapper = wrapper;
   }

   public void setHorizontalBoxWrapper(ItemWrapper hboxwrapper) {
      this.hboxwrapper = hboxwrapper;
   }

   // ...

   public abstract Item breakParagraph(HorizontalSequence hseq,
      int parwidth, int baselineskip);
} // class BasicLineBreaker
```

BasicLineBreaker.java

```
protected void add(VerticalBox vbox,
      HorizontalSequence line, int parwidth) {
   if (wrapper != null) {
      line = wrapper.wrap(line);
   }
   HorizontalFitter.fit(line, parwidth);
   Item hbox = new HorizontalBox(line);
   if (hboxwrapper != null) {
      hbox = hboxwrapper.wrap(hbox);
   }
   vbox.add(hbox);
}
```

- Die *add*-Methode wird von den abgeleiteten Klassen aufgerufen, um eine fertiggestellte *HorizontalSequence* an eine *VerticalBox* anzuhängen.
- An dieser Stelle werden die beiden Wrapper aufgerufen: *wrapper.wrap* passt ggf. die Formatierung an (Blocksatz, aufgerauht, zentriert etc) und der *hboxwrapper* dient nur u.U. der Visualisierung des Schachtelsystems.

- Gegeben ist eine horizontale Sequenz, die in einzelne Wörter zerlegt ist.
- Für den Algorithmus wird zusätzlich eine horizontale Sequenz verwaltet, die für die aktuelle Zeile steht. Diese ist zu Beginn leer.
- Für jedes Wort aus der horizontalen Sequenz wird überprüft, ob sich dieses noch auf die aktuelle Zeile quetschen lässt, d.h. ob der bereits vorhandene Zeileninhalt und die dazwischenliegende Trennstelle und das neue Wort bei einer maximalen Stauchung noch in eine Zeile passt.
- Wenn das neue Wort hineinpasst, dann wird das Wort in die aktuelle Zeile mitsamt der davorliegenden Trennstelle hinzugefügt. Die Schleife wird danach fortgesetzt.
- Wenn es nicht hineinpasst, wird die bislang vollendete Zeile zu den fertiggestellten Zeilen hinzugefügt (in eine vertikale Schachtel) und die aktuelle Zeile mit dem neuen Wort initialisiert. Danach geht es in der Schleife weiter.
- Am Ende der Schleife wird die aktuelle Zeile noch hinzugefügt, falls sie nicht leer ist.

FirstFitLineBreaker.java

```
public class FirstFitLineBreaker extends BasicLineBreaker {
   public Item breakParagraph(HorizontalSequence hseq,
         int parwidth, int baselineskip) {
      VerticalBox vbox = new VerticalBox(baselineskip);

      HorizontalSequence line = null; // current line
      Item bp = null; // last breakpoint
      // ... for loop over all words of hseq ...
      if (line != null) {
         line.add(bp);
         add(vbox, line, parwidth);
      }
      return vbox;
   }
} // class FirstFitLineBreaker
```

- Zu Beginn wird eine neue *VerticalBox* erzeugt, dann werden in der **for**-Schleife dieser sukzessive Zeilen hinzugefügt (siehe folgende Folie) und schließlich der Rest, sofern vorhanden, angehängt.

FirstFitLineBreaker.java

```
for (HorizontalSequence word: hseq.getWords()) {
   if (line == null) {
      line = word;
   } else {
      Width sum = new Width(line.getWidth());
      sum.add(bp); sum.add(word.getWidth());
      if (sum.getWidth() > parwidth) {
         if (sum.getWidth() - sum.getShrinkability()
               <= parwidth) {
            line.add(bp); line.add(word);
            add(vbox, line, parwidth); line = null;
         } else {
            add(vbox, line, parwidth); line = word;
         }
      } else {
         line.add(bp); line.add(word);
      }
   }
   bp = word.getFollowingBreakpoint();
}
```

- Der Best-Fit-Algorithmus bemüht sich um lokale Minima bezüglich der Häßlichkeit.
- Die Häßlichkeit wird danach bewertet, wie dramatisch der zur Verfügung stehende Spielraum ausgenutzt wird.
- Ein Wert von 0 bedeutet, dass nichts verändert werden muss. Wenn ein Wert von 1 erreicht wird, dann wird der Spielraum vollständig ausgenutzt. Wenn der Wert von 1 überschritten wird, dann überstrapazieren wir die vorhandene Flexibilität.
- Der Best-Fit-Algorithmus vergleicht alle Trennungskandidaten am nächsten Zeilenende und vergleicht sie anhand des Maßes.
- Best-Fit liefert nicht unbedingt bessere Resultate als First-Fit, da möglicherweise das globale Minimum erst dann erreicht wird, wenn zunächst eine etwas häßlichere Variante gewählt wird.

BestFitLineBreaker.java

```
public class BestFitLineBreaker extends BasicLineBreaker {
   public Item breakParagraph(HorizontalSequence hseq,
         int parwidth, int baselineskip) {
      VerticalBox vbox = new VerticalBox(baselineskip);

      HorizontalSequence line = null; // current line
      HorizontalSequence candidate = null;
      HorizontalSequence nextline = null; // beginning of next line
      double badness = 0; // of candidate
      Item bp = null; // last breakpoint
      // ... for loop over all words ...
      if (line != null) {
         line.add(bp);
         add(vbox, line, parwidth);
      }
      return vbox;
   }
} // class BestFitLineBreaker
```

BestFitLineBreaker.java

```
    for (HorizontalSequence word: hseq.getWords()) {
       if (line == null) {
          line = word;
       } else {
          line.add(bp); line.add(word);
       }
       if (candidate != null) {
          if (nextline == null) {
             nextline = word;
          } else {
             nextline.add(bp); nextline.add(word);
          }
       }
       Width width = line.getWidth();
       // ... analysis ...
       bp = word.getFollowingBreakpoint();
    }
```

- Zu dem aktuellen Kandidaten *candidate* gehört, falls definiert, auch *nextline* (was gehört in die nächste Zeile) und *badness* (gemessener Wert).

BestFitLineBreaker.java

```
if (width.getWidth() - width.getShrinkability() > parwidth) {
   if (candidate == null) {
      add(vbox, line, parwidth); line = null;
   } else {
      add(vbox, candidate, parwidth);
      line = nextline; candidate = null; nextline = null;
   }
} else if (width.getWidth() + width.getStretchability() >= parwidth) {
   double newbadness;
   if (width.getWidth() < parwidth) {
      newbadness = ((double) parwidth - width.getWidth()) /
         width.getStretchability();
   } else {
      newbadness = ((double) width.getWidth() - parwidth) /
         width.getShrinkability();
   }
   if (candidate == null || newbadness < badness) {
      candidate = line.clone(); nextline = null; badness = newbadness;
   }
} else {
   candidate = line.clone(); nextline = null;
   badness = Item.INFINITY;
}
```

Überblick der Struktur einer horizontalen Sequenz nach Knuth

- Eine horizontale Sequenz aus m Elementen kann durch die folgenden sechs Folgen repräsentiert werden:

$t_1 \ldots t_m$	Typ des Elements: *box*, *glue* oder *penalty*.
$w_1 \ldots w_m$	Die Weite des Elements.
$y_1 \ldots y_m$	Der Ausdehnungsspielraum bei Dehnfugen (*glue*), bei anderen Elementen 0.
$z_1 \ldots z_m$	Der Schrumpfungsspielraum bei Dehnfugen (*glue*), bei anderen Elementen 0.
$p_1 \ldots p_m$	Der Strafwert bei Sollbruchstellen, bei anderen Elementen 0.
$f_1 \ldots f_m$	Der Schalter bei Sollbruchstellen, bei anderen Elementen 0.

- Gegeben ist eine Folge $\{l_i\}$ von gewünschten Zeilenlängen. Normalerweise gilt $l_1 = l_2 = \ldots$, aber im Falle von Illustrationen oder anderen ungewöhnlichen Randbedingungen können diese voneinander abweichen.
- Wenn für die Zeile j die Elemente $a_j \ldots b_j$ in Betracht gezogen werden, dann ergeben sich aufsummiert folgende Werte:

$$
\begin{aligned}
L_j &= \sum_{i=a_j}^{b_{j-1}} w_i + \begin{cases} w_{b_j} & \text{falls } t_{b_j} = \mathit{penalty} \\ 0 & \text{sonst} \end{cases} \\
Y_j &= \sum_{i=a_j}^{b_{j-1}} y_i \\
Z_j &= \sum_{i=a_j}^{b_{j-1}} z_i
\end{aligned}
$$

- In Abhängigkeit von L_j und l_j lässt sich ein Justierverhältnis r_j bestimmen:
  - Falls $L_j = l_j$, dann passt die Zeile ganz genau und es sei $r_j = 0$
  - Falls $L_j < l_j$, dann ist die Zeile zu kurz und es sei $r_j = \frac{l_j - L_j}{Y_j}$, falls $Y_j > 0$, und undefiniert andernfalls.
  - Falls $L_j > l_j$, dann hat die Zeile Überlänge und es sei $r_j = \frac{l_j - L_j}{Z_j}$, falls $Z_j > 0$, und undefiniert andernfalls.
- Beispiel: Falls $r_j = \frac{1}{2}$, dann muss der Ausdehnungsspielraum zur Hälfte ausgenutzt werden.
- Die einzelnen Weiten aller Dehnfugen innerhalb der Zeile sind dann auf $w_i + r_j y_i$ zu setzen, falls $r_j \geq 0$ und $w_i + r_j z_i$, falls $r_j < 0$.

- Bereits 1963 schlug C. J. Duncan einen Algorithmus vor, der eine Zerlegung akzeptierte, sobald $|r_j| \leq 1$ galt. In diesem Falle haben wir eine Lösung, die im Rahmen der vorgesehenen Spielräume für Ausdehnungen und Schrumpfungen liegt.
- Da es jedoch häufig mehrere solcher Lösungen gibt, lohnt es sich, diese zu vergleichen. Auch falls es keine Lösung gibt, ist es wohl im Notfall sinnvoll, die am wenigsten schlechte Lösung auszusuchen.
- Knuth hat für TEX folgendes Maß als für sinnvoll befunden, das die Unschönheit einer Zeile bewertet:

$$\beta_j = \begin{cases} \infty, & \text{falls } r_j \text{ undefiniert oder } r_j < -1 \\ \lfloor 100|r_j|^3 + .5 \rfloor & \text{sonst} \end{cases}$$

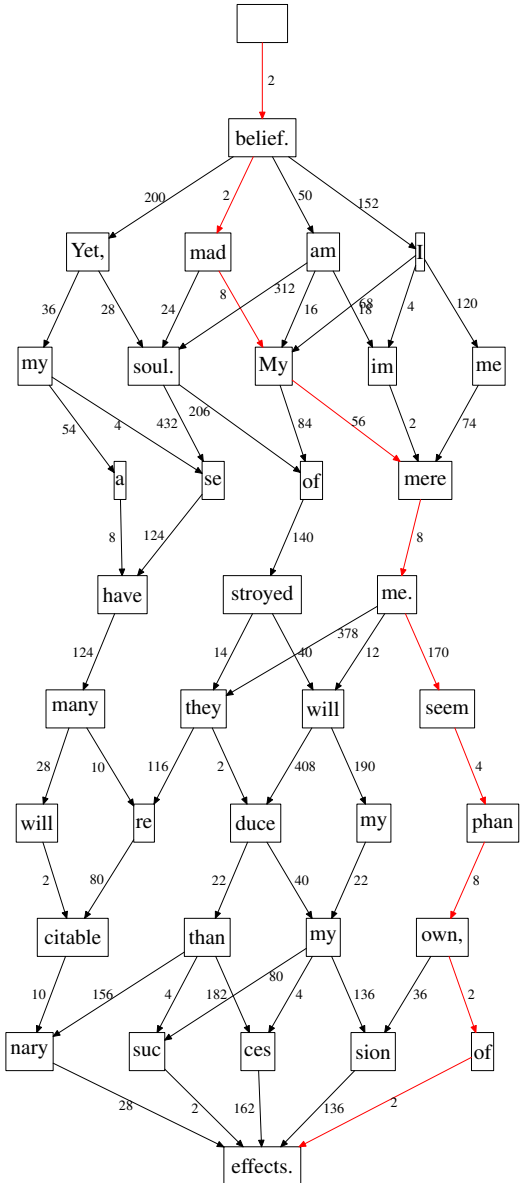
- Bei einer Aggregierung der Unschönheiten der einzelnen Zeilen geht der Strafwert der Sollbruchstelle ein: $\pi_j = p_{b_j}$
- Ferner sei α_j positiv (z.B. 3000), falls die j-te Zeile von zwei Sollbruchstellen eingefasst ist, bei der die Schalter jeweils auf 1 stehen. (Aufeinanderfolgende Trennungen von Wörtern sind zu vermeiden.)
- Darauf aufbauend definiert Knuth folgendes Maß für die Unschönheit einer Zeile:

$$\delta_j = \begin{cases} (1+\beta_j+\pi_j)^2+\alpha_j & \text{falls } \pi_j \geq 0 \\ (1+\beta_j)^2-\pi_j^2+\alpha_j & \text{falls } -\infty < \pi_j < 0 \\ (1+\beta_j)^2+\alpha_j & \text{falls } \pi_j = -\infty \end{cases}$$

- Das Maß für die Unschönheit einer Zerlegung ergibt sich dann aus der Summe der δ_j.

- Knuth versuchte, mit seinem Maß folgende Ziele umzusetzen:
  - Wenn eine einzelne Zeile unvermeidlich schlecht ist, sollen die anderen Zeilen dennoch möglichst schön gehalten werden.
  - Die Unschönheit wächst mit den Strafwerten der Sollbruchstellen π_j. Bei negativen Strafwerten wird ein Bruch an der Stelle gerne gesehen. Allerdings wird bei $\pi_j = -\infty$ dies ignoriert, da hier ein Bruch unvermeidlich ist.
  - Da jeweils 1 dazu addiert wird, werden bei ansonsten annäherungsweise gleich guten Lösungen diejenigen mit weniger Zeilen bevorzugt.

Zerlegung eines Paragraphen als Optimierungsproblem 364



- Optimierungsproblem: Finde die Zerlegung mit der minimalen Summe der Unschönheitsmaße.
- Das Problem kann als gerichteter, antizyklischer Graph (DAG) dargestellt werden, bei dem die möglichen Bruchstellen als Knoten repräsentiert werden und die zugehörigen Unschönheitsmaße der jeweils vorangehenden Zeile als Kostenwert.
- Einen extra Knoten gibt es für den Anfang und ebenso gibt es nur einen Endknoten.
- Das linke Beispiel ist ein zusammengekürzter Graph, bei dem Kanten mit extremem Unschönheiten herausgefiltert wurden.

- Gegeben sei ein Paragraph mit $n+1$ Sollbruchstellen, wobei die letzte Sollbruchstelle unvermeidlich ist.
- Zulässige Zerlegungen sind dann Teilmengen aller Sollbruchstellen, die die letzte Sollbruchstellstelle enthalten.
- Dann gibt es insgesamt $\sum_{i=0}^{n} \binom{n}{i}$ mögliche Zerlegungen.
- Der Aufwand, sämtliche zulässige Zerlegungen zu untersuchen, wäre extrem.
- Wenn es als graphentheoretisches Problem betrachtet wird (Finden des kürzesten Wegs), dann lässt sich der Aufwand unter Verwendung des Algorithmus von Dijkstra auf $O(n^2)$ begrenzen. Dies ist immer noch sehr aufwendig.

- Knuth hat gezeigt, dass sich der Aufwand auf $O(n * m)$ begrenzen lässt, wobei dann m dann der Zahl der zu erwartenden Bruchstellen pro Zeile entspricht.
- Der Wert m lässt sich noch weiter reduzieren, wenn
  - Trennstellen in Wörtern erst dann betrachtet werden, wenn klar ist, dass sie benötigt werden, und
  - Zerlegungen mit extrem hohen Unschönheitsmaßen an einem Zeilenbruch von vorneherein aussortiert werden.
- In der Praxis bedeutet dies, dass der Aufwand sich auf $O(n)$ beschränkt, d.h. mit linearem Aufwand eine Zerlegung eines Paragraphen erfolgen kann. Voraussetzung dafür ist allerdings, dass Paragraphen nicht ungewöhnlich lang werden.

- Der gesamte Paragraph wird in einem Durchgang sequentiell bearbeitet.
- Es gibt eine Menge der Bruchstellen (zu Beginn nur der Anfang des Paragraphen). Eine Teilmenge davon sind die aktiven Bruchstellen (zu Beginn ebenfalls nur der Anfang des Paragraphen).
- Bei jeder Sollbruchstelle beim sequentiellen Durchgang wird die Unschönheit der Zeile gemessen ausgehend von jeder aktiven Bruchstelle bis zur neuen Sollbruchstelle.
- Wenn keine aktive Bruchstelle gefunden wird mit $|r| \leq 1$, dann wird die Sollbruchstelle nicht weiter beachtet.
- Andernfalls wird die Sollbruchstelle in die Liste der aktiven Bruchstellen aufgenommen. Dabei wird notiert, zu welcher vorherigen aktiven Bruchstelle δ minimal war.
- Aktive Bruchstellen, die zur aktuellen Sollbruchstelle einen Wert $r < -1$ haben, werden aus der Liste der aktiven Bruchstellen entfernt.
- Inaktive Bruchstellen, auf die keine aktive Bruchstelle verweist, können entfernt werden (Sackgassen).

First-Fit:

For the most wild, yet most homely narrative which I am about to pen, I neither expect nor solicit belief. Mad indeed would I be to expect it, in a case where my very senses reject their own evidence. Yet, mad am I not--and very surely do I not dream. But to-morrow I die, and to-day I would unburden my soul. My immediate purpose is to place before the world, plainly, succinctly, and without comment, a series of mere household events. In their consequences, these events have terrified--have tortured--have destroyed me. Yet I will not attempt to expound them. To me, they have presented little but horror--to many they will seem less terrible than baroques. Hereafter, perhaps, some intellect may be found which will reduce my phantasm to the commonplace--some intellect more calm, more logical, and far less excitable than my own, which will perceive, in the circumstances I detail with awe, nothing more than an ordinary succession of very natural causes and effects.

Total-Fit:

For the most wild, yet most homely narrative which I am about to pen, I neither expect nor solicit belief. Mad indeed would I be to expect it, in a case where my very senses reject their own evidence. Yet, mad am I not--and very surely do I not dream. But to-morrow I die, and to-day I would unburden my soul. My immediate purpose is to place before the world, plainly, succinctly, and without comment, a series of mere household events. In their consequences, these events have terrified--have tortured--have destroyed me. Yet I will not attempt to expound them. To me, they have presented little but horror--to many they will seem less terrible than baroques. Hereafter, perhaps, some intellect may be found which will reduce my phantasm to the commonplace--some intellect more calm, more logical, and far less excitable than my own, which will perceive, in the circumstances I detail with awe, nothing more than an ordinary succession of very natural causes and effects.

Best-Fit:

For the most wild, yet most homely narrative which I am about to pen, I neither expect nor solicit belief. Mad indeed would I be to expect it, in a case where my very senses reject their own evidence. Yet, mad am I not--and very surely do I not dream. But to-morrow I die, and to-day I would unburden my soul. My immediate purpose is to place before the world, plainly, succinctly, and without comment, a series of mere household events. In their consequences, these events have terrified--have tortured--have destroyed me. Yet I will not attempt to expound them. To me, they have presented little but horror--to many they will seem less terrible than baroques. Hereafter, perhaps, some intellect may be found which will reduce my phantasm to the commonplace--some intellect more calm, more logical, and far less excitable than my own, which will perceive, in the circumstances I detail with awe, nothing more than an ordinary succession of very natural causes and effects.

Total-Fit:

For the most wild, yet most homely narrative which I am about to pen, I neither expect nor solicit belief. Mad indeed would I be to expect it, in a case where my very senses reject their own evidence. Yet, mad am I not--and very surely do I not dream. But to-morrow I die, and to-day I would unburden my soul. My immediate purpose is to place before the world, plainly, succinctly, and without comment, a series of mere household events. In their consequences, these events have terrified--have tortured--have destroyed me. Yet I will not attempt to expound them. To me, they have presented little but horror--to many they will seem less terrible than baroques. Hereafter, perhaps, some intellect may be found which will reduce my phantasm to the commonplace--some intellect more calm, more logical, and far less excitable than my own, which will perceive, in the circumstances I detail with awe, nothing more than an ordinary succession of very natural causes and effects.

- Für einen Blocksatz mit einer einheitlichen Farbe des Texts ist die Möglichkeit, Wörter bei Bedarf trennen zu können, essentiell.
- Einfache Trennungsregeln stehen nicht zur Verfügung oder liefern zuviele Falschtrennungen oder finden zuwenig Trennungsstellen.
- Die erste TEX-Implementierung von 1977 entfernte zunächst die Nachsilbe, die Vorsilbe und suchte dann nach Folgen von Vokal-Konsonant-Konsonant-Vokal und trennte dann zwischen den beiden Konsonanten. Damit werden jedoch nur ca. 40% der zulässigen Trennstellen erfasst und es bleiben Fehler wie etwa in „prog-ram“, die von längeren Ausnahmenlisten erfasst werden müssen.
- Einzelne Wörterbücher nennen umfassend Trennstellen (wie etwa Webster's oder der Duden). Diese Listen sind jedoch recht umfangreich und trotz ihres Umfangs nicht umfassend. Dies gilt insbesondere für Sprachen, die Zusammensetzungen erlauben und zahlreiche Beugungen (Flexionen) vorsehen wie etwa die deutsche Sprache.

- Gegeben sei eine Liste mit Trennstellen für einen umfangreichen Wortschatz. (Bei TEX wurde hier das *Merriam-Webster Pocket Dictionary* von 1974 mit ca. 50.000 Einträgen verwendet. Für die deutsche Sprache gibt es eine von Werner Lemberg gepflegte Liste mit 472.479 Einträgen, siehe `http://repo.or.cz/w/wortliste.git`.)
- Gesucht werden
  - ein sprachunabhängiges Regelsystem,
  - ein Verfahren, dass das Regelsystem für eine Sprache mit Hilfe von einer umfangreichen Trennliste automatisiert konfiguriert, so dass die Zahl der erzeugten Regeln möglichst minimal ist und
  - Datenstrukturen und Algorithmen, die mit geringem Laufzeit- und Speicheraufwand die möglichen Trennstellen eines Wortes finden, selbst wenn es nicht auf der Liste enthalten war.
- Ziel sollte es sein, dass die Mehrheit der korrekten Trennstellen gefunden wird und nur marginal wenige falsche Trennstellen geliefert werden, die notfalls mit Hilfe von Ausnahmelisten behandelt werden können.

- Das Regelsystem betrachtet nur vier aufeinanderfolgende Buchstaben $b_1 \ldots b_4$, um zu beurteilen, ob zwischen b_2 und b_3 getrennt werden kann.
- Da eine Tabelle mit $26^4 = 456976$ Einträgen zu umfangreich ist, gibt es stattdessen drei Tabellen für die Buchstabenpaare (b_1, b_2), (b_2, b_3) und (b_3, b_4), die jeweils die Wahrscheinlichkeiten angeben, dass hinter, zwischen und vor dem jeweiligen Buchstabenpaar in der gegebenen Wortliste getrennt wird.
- Die drei Werte aus den Tabellen werden miteinander multipliziert und es werden alle Trennstellen berücksichtigt, bei denen das Produkt einen vorgegebenen Mindestwert erreicht. (Hierbei wird ignoriert, dass die drei Wahrscheinlichkeiten nicht unabhängig voneinander sind.)
- Franklin Liang stellte in seiner Untersuchung jedoch fest, dass mit diesem Verfahren nur ca. 40% der Trennstellen gefunden werden bei einer Fehlerquote von 8%.

- Flexibler als die Wahrscheinlichkeitstabellen für die Buchstabenpaare sind Trennungsmuster, da sie beliebig lang sein können und auch gängige Vor- und Nachsilben berücksichtigen können.
- Beispiele für Trennungsmuster der englischen Sprache aus der Arbeit von Franklin Liang:

  .in-d .in-s .in-t .un-d b-s -cia con-s con-t e-ly erl-l er-m
  ex- -ful it-t i-ty -less l-ly -ment n-co -ness n-f n-l n-si n-v om-m
  -sion s-ly s-nes ti-ca x-p

  (Der Punkt repräsentiert jeweils den Anfang oder das Ende eines Worts.)
- Beispiele für die deutsche Sprache:

  .es-p .ob-l a-bl an-kl eu-e e-z ge-s g-q h-d h-h i-che i-d ll-b o-ra.

- Auch wenn Trennungsmuster viele Fälle korrekt erfassen, so bleibt doch eine signifikante Zahl von Fällen, bei denen falsch getrennt wird.
- Beispiel: Obwohl die Regel *-tion* in vielen Fällen zutrifft, darf „cation“ nicht getrennt werden.
- Wenn Ausnahmen nur über eine explizite Ausnahmeliste geregelt werden können, wird sie zu umfangreich.
- Deswegen ist es sinnvoll, auch Ausnahmeregeln in Form von Regeln zu formulieren. Als Ausnahmeregel würde sich hier *.cat* empfehlen.
- Das System lässt sich fortsetzen mit Ausnahmen der Ausnahmeregeln etc.

- Das Regelsystem besteht aus einer beliebigen Zahl von Trennungsmustern und einer Festlegung der Variablen *leftmin* und *rightmin*. (Die beiden Variablen geben den Mindestumfang eines Wortteils an, der vorne oder am Ende des Worts abgetrennt werden kann. In der englischen Sprache sind beide Werte 2, in der deutschen Sprache kann *leftmin* auf 1 gesetzt werden.)
- Jedes Trennungsmuster besteht aus Buchstaben, dem Punkt "." als Zeichen für den Wortanfang oder das Wortende und Ziffern, die Trennstellen bewerten.
- Ziffern repräsentieren den Rang einer Trennstelle in einer Regel. Ungerade Ränge (beginnend ab 1) befürworten eine Trennung, gerade Ränge (beginnend ab 2) stehen für eine Unterdrückung einer Trennstelle.
- Für ein zu trennendes Wort werden sämtliche zutreffenden Regeln berücksichtigt. Wenn sich mehrere Regeln für eine potentielle Trennregel einander widersprechen, dann setzt sich die Regel mit dem höheren Rang für die Trennstelle durch.

- Zu trennen ist das Wort „typography“. Folgende Trennungsmuster bei TEX treffen darauf zu (siehe Datei *hyphen.tex*):
  - *1ty*
  - *y3po*
  - *5po4g*
  - *1gr*
  - *4graphy*
  - *3raphy*
  - *1phy*
- Das Resultat ist „ty-pog-ra-phy“.
- Zu sehen ist hier, dass normalerweise durchaus vor „gr“ getrennt wird (4. Regel), dies in diesem konkreten Fall jedoch durch zwei Ausnahmeregeln unterbunden wird: *5po4g* und *4graphy* – in beiden Fällen schlägt der Rang 4 den Rang 1.
- Die vorgeschlagene Trennung entspricht genau der, die von Merriam-Webster vorgegeben wird.

- Jede Trennungsregel enthält mindestens einen Buchstaben.
- Entsprechend müssen bei einem Wort der Länge n insgesamt $\frac{n(n+1)}{2}$ Teilzeichenfolgen untersucht werden, was einem Aufwand von $O(n^2)$ entspricht.
- Dieser Aufwand multipliziert sich mit dem Aufwand, nach einer Regel für eine Zeichenfolge zu suchen.
- Das entspricht zumindest der einfachsten Vorgehensweise, die alle Regeln beispielsweise über eine *HashMap* zugänglich macht.
- Alternativ wäre es denkbar, einen endlichen Automaten für ein Regelsystem zu erzeugen. Dann reduziert sich der Aufwand auf $O(n)$. Allerdings wäre der Automat ziemlich umfangreich. (LuaTEX arbeitet inzwischen mit einem endlichen Automaten.)
- Eine weitere Alternative sind Tries.

- Der Name Trie leitet sich ab von *re-trie-val*. Die Datenstruktur wurde zuerst von Briandais (1959) und Fredkin (1960) beschrieben. Der Begriff geht auf Fredkin zurück.
- Anders als bei assoziativen Arrays (Maps) wird davon ausgegangen, dass Schlüssel sich definieren als eine Sequenz von Zeichen aus einem endlichen Alphabet.
- Entsprechend erfolgt der Zugriff zeichenweise. An jedem erreichten Punkt können wir dabei auf eine Regel stoßen, die Suche aber noch fortsetzen. Die Suche wird abgebrochen, sobald eine Zeichenfolge als Anfang eines Schlüssels nicht vorkommen kann.
- Tries können durch Baumstrukturen repräsentiert werden. Baumknoten können auf Regeln verweisen und die Zahl der möglichen Unterbäume zu einem Baumknoten ist nur durch den Umfang des Alphabets begrenzt.

TrieReader.java

```
package de.uniulm.mathematik.tries;

public interface TrieReader<TrieInfo> {

   public interface TriePointer<TrieInfo> {
      public TriePointer<TrieInfo> descend(int code);
      public TrieInfo getInfo();
   }

   public TriePointer<TrieInfo> getRoot();
   public int getNumberOfEntries();
   public int getNumberOfNodes();
}
```

- Der Typparameter *TrieInfo* repräsentiert hier den Typ der durch die Datenstruktur auffindbaren Objekte (hier: Trennungsregeln).
- Eine Suche beginnt mit dem Aufruf von *getRoot*, das einen Zeiger (*TriePointer*) in die Datenstruktur liefert. An jedem Punkt lässt sich mit *getInfo* abfragen, ob ein Objekt zu finden ist und ein weiterer Abstieg ist mit *descend* möglich, das *null* zurückliefert, wenn die Zeichenfolge als Anfang eines Schlüssels nicht vorkommen kann.

HashedTrie.java

```
public class TrieNode<TrieInfo> implements TrieIterator<TrieInfo> {
   protected TrieInfo info;
   protected HashMap<Integer, TrieNode<TrieInfo> > subnodes;
   protected TrieNode() {
      info = null;
      subnodes = new HashMap<Integer, TrieNode<TrieInfo> >();
   }
   public TrieNode<TrieInfo> descend(int code) {
      return subnodes.get(new Integer(code));
   }
   public TrieInfo getInfo() { return info; }
   public Iterator<Integer> iterator() {
      return subnodes.keySet().iterator();
   }
}
```

- Die einem Baumknoten untergeordneten Unterbäume werden hier mit einer *HashMap* verwaltet.
- Franklin Liang hat für die TEX-Implementierung eine komprimierte Datenstruktur (*packed tries*) entwickelt, die noch effizienter ist und insbesondere nur sehr wenig Speicherplatz benötigt. (Allerdings lässt sich das in Java nicht in gleicher Weise umsetzen.)

HashedTrie.java

```
public class HashedTrie<TrieInfo>
   implements TrieReader<TrieInfo>, TrieConstructor<TrieInfo> {

   // public class TrieNode<TrieInfo> ...

   private TrieNode<TrieInfo> root;
   private int numberOfNodes; private int numberOfEntries;

   public HashedTrie() {
      root = new TrieNode<TrieInfo>();
      numberOfNodes = 0; numberOfEntries = 0;
   }

   public TrieNode<TrieInfo> getRoot() {
      return root;
   }

   // public void insert(String word, TrieInfo info) ...

   public int getNumberOfEntries() { return numberOfEntries; }
   public int getNumberOfNodes() { return numberOfNodes; }
}
```

HashedTrie.java

```
public void insert(String word, TrieInfo info) {
   assert info != null;
   TrieNode<TrieInfo> node = root;
   for (int index = 0; index < word.length();
         index = word.offsetByCodePoints(index, 1)) {
      int ch = word.codePointAt(index);
      Integer key = new Integer(ch);
      TrieNode<TrieInfo> subnode = node.subnodes.get(key);
      if (subnode == null) {
         subnode = new TrieNode<TrieInfo>();
         node.subnodes.put(key, subnode);
         ++numberOfNodes;
      }
      node = subnode;
   }
   node.info = info;
   ++numberOfEntries;
}
```

HyphenationPoint.java

```
public interface HyphenationPoint {
   public int getPosition();
   public int getValue();
}
```

- Die Trennungsposition ist relativ zum Beginn der Trennungsregel und *getValue* liefert den zugehörigen Rang.

HyphenationRule.java

```
public interface HyphenationRule
   extends Iterable<HyphenationPoint> {
}
```

- Eine Trennungsregel besteht dann nur noch aus einer Folge von Trennungspositionen und zugehörigen Rängen.
- Die Buchstabenfolge der Regel gehört zum Schlüssel und wird in dieser Datenstruktur nicht wiederholt.

Hyphenator.java

```
public int[] hyphenate(String s) {
   // extract array of codepoints
   // where upper case letters are mapped to lower case
   String word = "." + s + ".";
   int len = word.codePointCount(0, word.length());
   int[] codepoints = new int[len];
   for (int i = 0; i < word.length(); i = word.offsetByCodePoints(i, 1)) {
      codepoints[i] = word.codePointAt(i);
      if (Character.isUpperCase(codepoints[i])) {
         codepoints[i] = Character.toLowerCase(codepoints[i]);
      }
   }
   // find matching hyphenation patterns and apply them to value[] ...
   // create array with hyphenation positions ...
}
```

- Ein zu trennendes Wort wird mit zwei Punkten ergänzt, damit auch die Trennregeln berücksichtigt werden, die sich nur auf den Anfang oder das Ende eines Worts beziehen.
- Das Eingabealphabet wird standardisiert, indem hier Groß- auf Kleinbuchstaben abgebildet werden. Es ist hier auch mehr denkbar wie etwa die Wegnahme von Akzenten.

Hyphenator.java

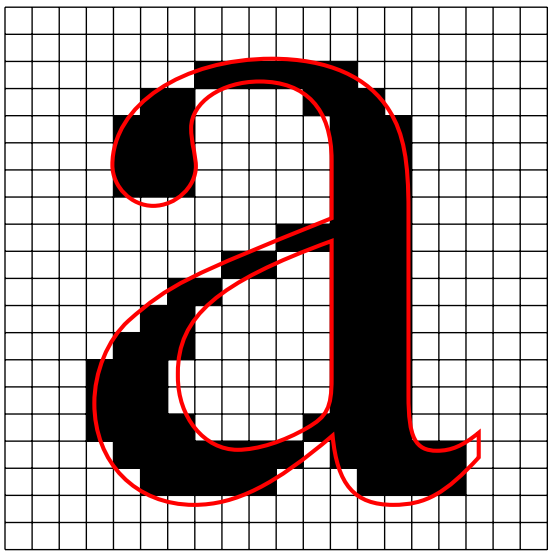
```
// find matching hyphenation patterns and apply them to value[]
int[] value = new int[len+1];
for (int start = 0; start+leftmin < len; ++start) {
   int pos = start;
   for (TrieReader.TriePointer<HyphenationEntry>
         ptr = patterns.getRoot();
         ptr != null;
         ptr = pos < len? ptr.descend(codepoints[pos++]): null) {
      HyphenationEntry rule = ptr.getInfo();
      if (rule != null) {
         for (HyphenationPoint point: rule) {
            int index = point.getPosition() + start - 1;
            if (index >= leftmin && index + rightmin < len-1) {
               int val = point.getValue();
               if (val > value[index]) {
                  value[index] = val;
               }
            }
         }
      }
   }
}
```

Hyphenator.java

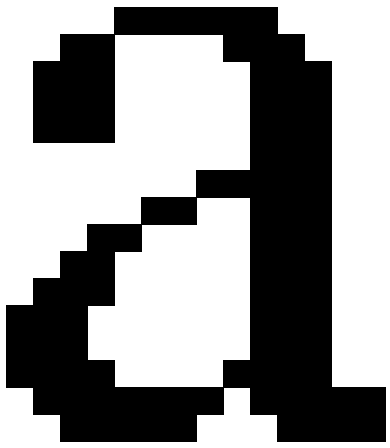
```
// create array with hyphenation positions
int count = 0;
for (int i = 0; i <= len; ++i) {
   if (value[i] % 2 == 1) {
      ++count;
   }
}
if (count == 0) {
   return null;
}
int[] result = new int[count]; int index = 0;
for (int i = 0; i <= len; ++i) {
   if (value[i] % 2 == 1) {
      result[index++] = i;
   }
}
return result;
```

- Der Trennungsalgorithmus von Franklin Liang ist bis heute das wichtigste Trennungsverfahren, das auch außerhalb von TEX bei anderen Programmen wie etwa *troff* oder *QuarkXPress* eingesetzt wird.
- Verschiedene Verbesserungen wurden jedoch bzw. bereits umgesetzt (in Ω_2), aber von LuaTEX bislang nicht übernommen:
  - Trennungsregeln sollten Gewichtungen vorsehen, damit gute Trennstellen von weniger guten unterschieden werden können. In der deutschen Sprache sollte eher die Trennung „Trennungs-regel“ vor „Trennungsre-gel“ bevorzugt werden. (Dies könnte in den Strafwert der entsprechenden Sollbruchstelle einfließen.)
  - Bei gleich guten Trennungsstellen sollten die Trennungsstellen in der Mitte eines Worts bevorzugt werden.
  - Mehrdeutigkeiten sollten nicht getrennt werden: „Wach-stube“ vs. „Wachs-tube“.

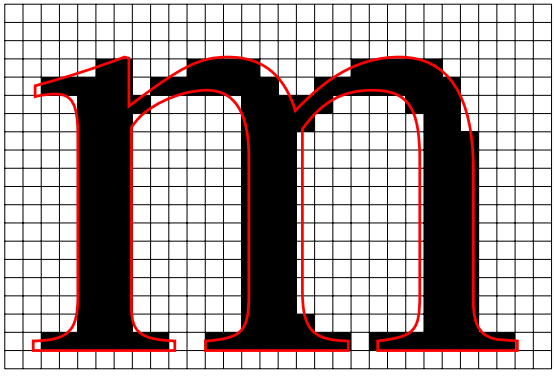
- Am Ende sind die geometrisch definierten Flächen in ein Raster gegebener Größe und Ausformung zu überführen, um sie auf dem Bildschirm oder dem Drucker darstellen zu können.
- Im einfachsten Falle handelt es sich dabei um ein Schwarz/Weiß-Raster. Ansonsten sind Grauabstufungen oder auch Farben möglich, wobei unterschiedliche Farbräume üblich sind.
- Idealerweise sind Rasterpunkte symmetrisch (etwa rotationssymmetrische Kreise oder achsensymmetrische Quadrate). Jedoch kommen auch Ellipsen oder Rechtecke vor. Auch ist es (TFT-Bildschirme!) denkbar, dass ein Pixel aus mehreren Sub-Pixeln besteht, die unterschiedliche Farben darstellen und nur benachbart sind.
- METAFONT unterstützt nur Schwarz/Weiß-Raster, kann aber mit Verzerrungen umgehen. PostScript unterstützt in jedem Falle Farben, selbst wenn das Ausgabe-Gerät nur ein Schwarz/Weiß-Raster anbietet.

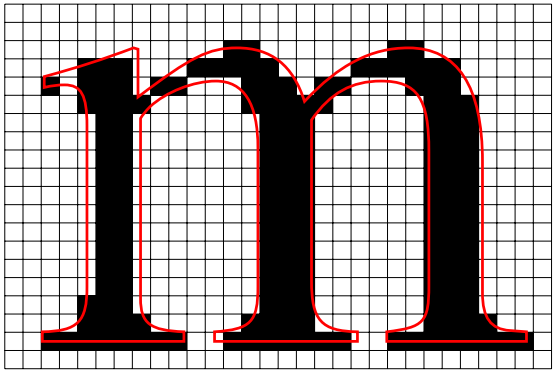


- Gegeben sei eine Kurve, die schwarz auszufüllen ist.
- Ferner sei eine Abbildung gegeben, die das verwendete Koordinatensystem auf das Raster abbildet. Idealerweise sei angenommen, dass die Rasterfelder quadratisch sind.
- Dann werden genau die Rasterfelder schwarz, deren Mittelpunkt (im Ausgangskoordinatensystem) innerhalb des Pfades liegt.

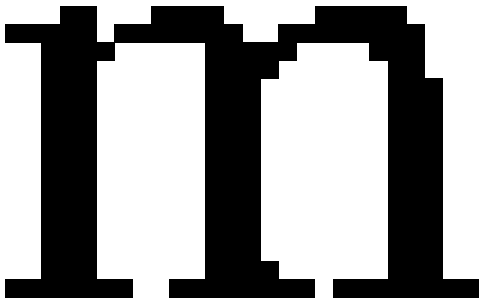


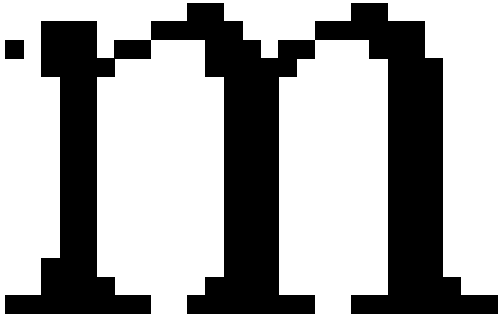
- Gegeben sei eine Kurve, die schwarz auszufüllen ist.
- Ferner sei eine Abbildung gegeben, die das verwendete Koordinatensystem auf das Raster abbildet. Idealerweise sei angenommen, dass die Rasterfelder quadratisch sind.
- Dann werden genau die Rasterfelder schwarz, deren Mittelpunkt (im Ausgangskoordinatensystem) innerhalb des Pfades liegt.



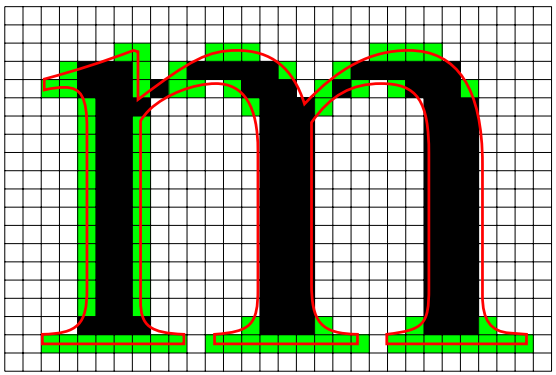


- Die Problematik dieses Verfahrens liegt in der hohen Empfindlichkeit gegen Verschiebungen.
- Keine der beiden Rasterungen für das kleine m ist optimal. Die untere Variante ist aber katastrophal im Vergleich zu der ersten, weil sie um eine halbe Rasterweite nach rechts und nach oben verschoben worden ist.
- In der oberen Variante ist der rechte Zwischenraum breiter als der linke.
- In der unteren Variante ist der linke Schaft deutlich schmaler als die rechten beiden Schäfte. Ferner sind die Serifen deutlich unregelmäßig, und die linke obere Serife ist sogar vom Rest des Buchstabens abgetrennt.



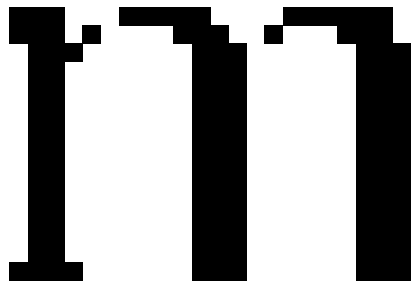


- Die Problematik dieses Verfahrens liegt in der hohen Empfindlichkeit gegen Verschiebungen.
- Keine der beiden Rasterungen für das kleine m ist optimal. Die untere Variante ist aber katastrophal im Vergleich zu der ersten, weil sie um eine halbe Rasterweite nach rechts und nach oben verschoben worden ist.
- In der oberen Variante ist der rechte Zwischenraum breiter als der linke.
- In der unteren Variante ist der linke Schaft deutlich schmaler als die rechten beiden Schäfte. Ferner sind die Serifen deutlich unregelmäßig, und die linke obere Serife ist sogar vom Rest des Buchstabens abgetrennt.

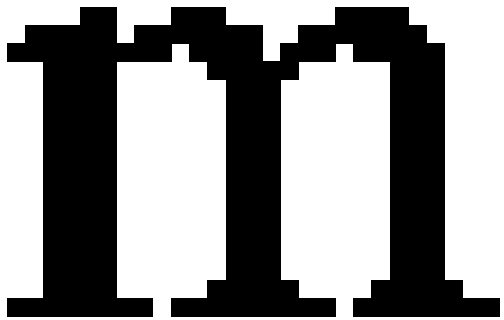


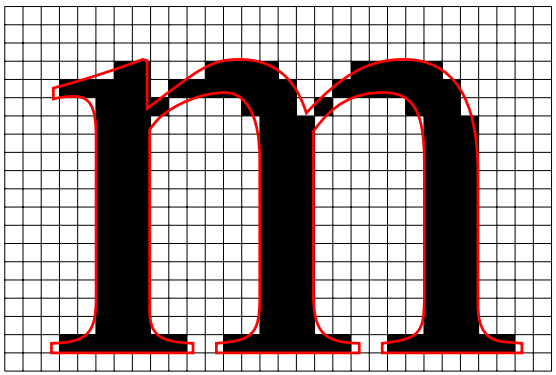
- Die Rasterung einer Fläche ist besonders dort problematisch, wo ihr Rand tangential zu waagrechten oder senkrechten Linien verläuft.
- Wenn diese Linien in etwa der Mitte eines Rasterfeldes verlaufen, ist die Rasterung extrem instabil. Geringste Verschiebungen führen dann dazu, dass ganze Spalten oder Zeilen von Rasterfeldern hinzukommen oder verschwinden.
- Erstrebenswert ist es daher, die auszufüllende Kurve von Anfang an so zu konfigurieren, dass die gefährdeten Ränder entlang der Rastergrenzen verlaufen.
- Links sind alle »Wackelkandidaten« grün gefärbt.

- Variante ohne »Wackelkandidaten«.

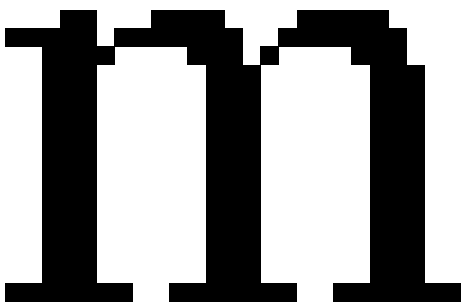


- Variante mitsamt allen »Wackelkandidaten«.

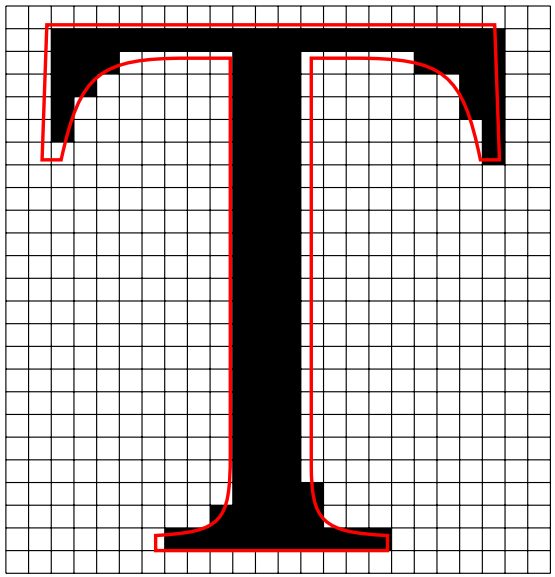




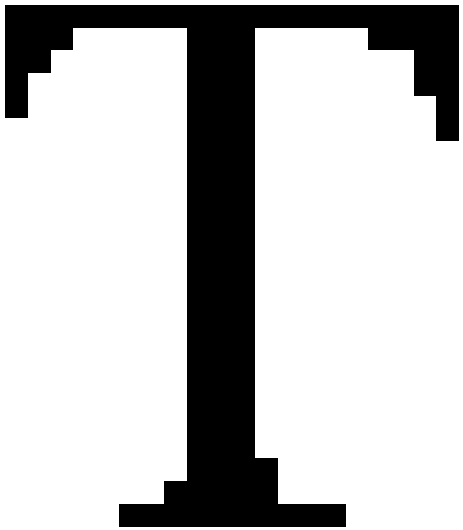
- Bei dieser Version des m wurden die Ränder der drei Schäfte auf die Rastergrenzen ausgerichtet.
- Das war nur möglich, indem alle drei Schäfte etwas zusammenrückten.
- Eine gute Rasterung lässt sich nur erreichen, wenn die Form in geeigneter Weise an die Rasterung angepasst wird.
- METAFONT versucht dies zu automatisieren, wenn die Variable `autoround` einen positiven Wert besitzt.



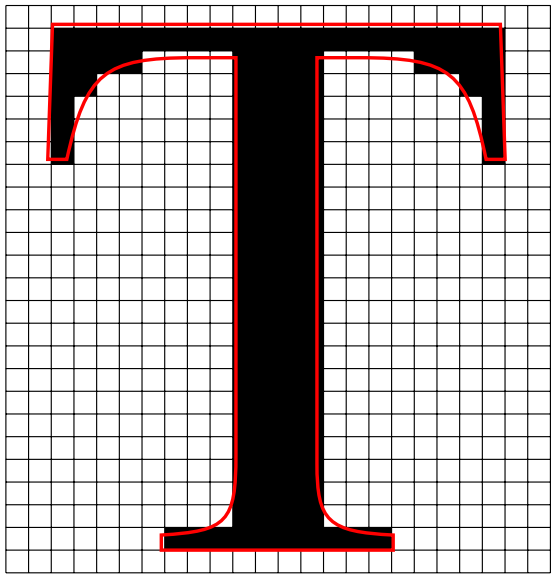
- Bei dieser Version des m wurden die Ränder der drei Schäfte auf die Rastergrenzen ausgerichtet.
- Das war nur möglich, indem alle drei Schäfte etwas zusammenrückten.
- Eine gute Rasterung lässt sich nur erreichen, wenn die Form in geeigneter Weise an die Rasterung angepasst wird.
- METAFONT versucht dies zu automatisieren, wenn die Variable `autoround` einen positiven Wert besitzt.



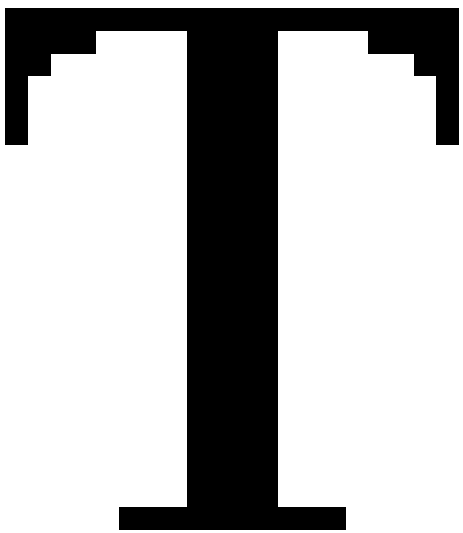
- Selbst dann, wenn bei symmetrisch gestalteten Formen beide Seiten getrennt betrachtet akzeptabel sind, würde das Auge eine Asymmetrie sofort wahrnehmen.
- Beim T liegt hier das Problem nicht nur darin, dass die rechte Seite des Schafts instabil ist. Es fällt auch auf, dass die oberen Serifen ungleichmäßig gestaltet sind.



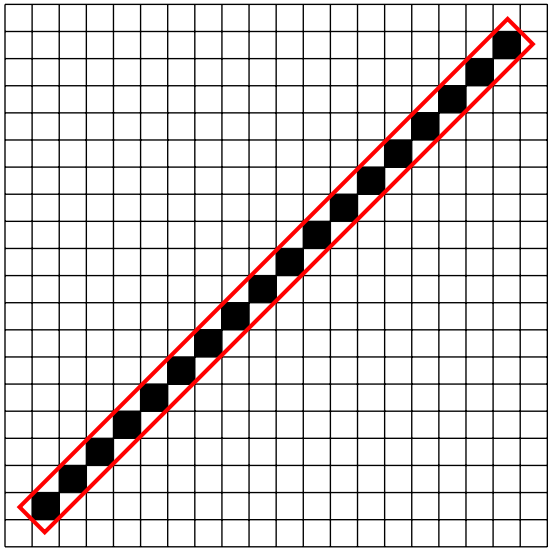
- Selbst dann, wenn bei symmetrisch gestalteten Formen beide Seiten getrennt betrachtet akzeptabel sind, würde das Auge eine Asymmetrie sofort wahrnehmen.
- Beim T liegt hier das Problem nicht nur darin, dass die rechte Seite des Schafts instabil ist. Es fällt auch auf, dass die oberen Serifen ungleichmäßig gestaltet sind.



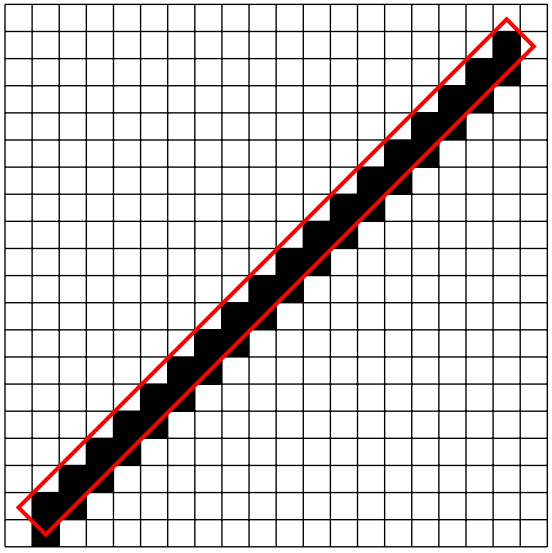
- Das Problem kann nur zuverlässig gelöst werden, wenn die Symmetrie-Achse der Form so gelegt wird, dass sie durch die Mittelpunkte der Rasterfelder verläuft.
- Dessen ungeachtet sind weiterhin Ränder der Form mit horizontalen oder senkrechten Tangenten auf die Rasterkanten auszurichten, auch wenn dies zur Umgestaltung der Form führt.



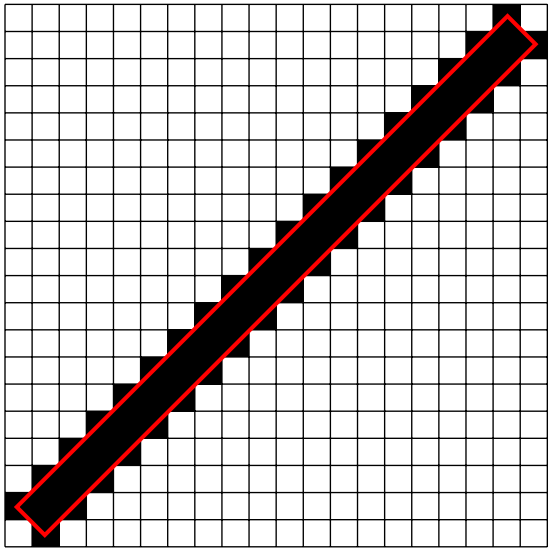
- Das Problem kann nur zuverlässig gelöst werden, wenn die Symmetrie-Achse der Form so gelegt wird, dass sie durch die Mittelpunkte der Rasterfelder verläuft.
- Dessen ungeachtet sind weiterhin Ränder der Form mit horizontalen oder senkrechten Tangenten auf die Rasterkanten auszurichten, auch wenn dies zur Umgestaltung der Form führt.



- Auch diagonal geführte Linien können Instabilitäten aufweisen.
- Diese Diagonale hat eine Breite von etwa $r\sqrt{2}$, wobei r für die Rasterseitenlänge steht.
- So sieht das Ergebnis aus, wenn der Rand der Figur nicht berücksichtigt wird, d.h. wenn ein Rasterfeld-Mittelpunkt genau auf dem Rand liegt, wird er nicht mitgezeichnet.

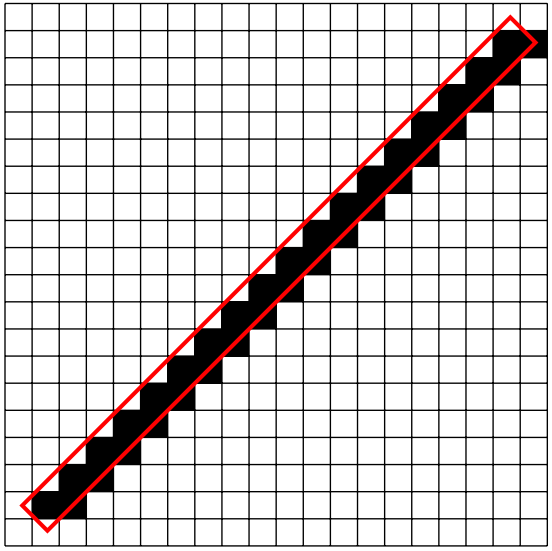


- Wird die Kurve minimal nach unten verschoben, erhalten wir über das Doppelte der vorherigen Rasterpunkte.

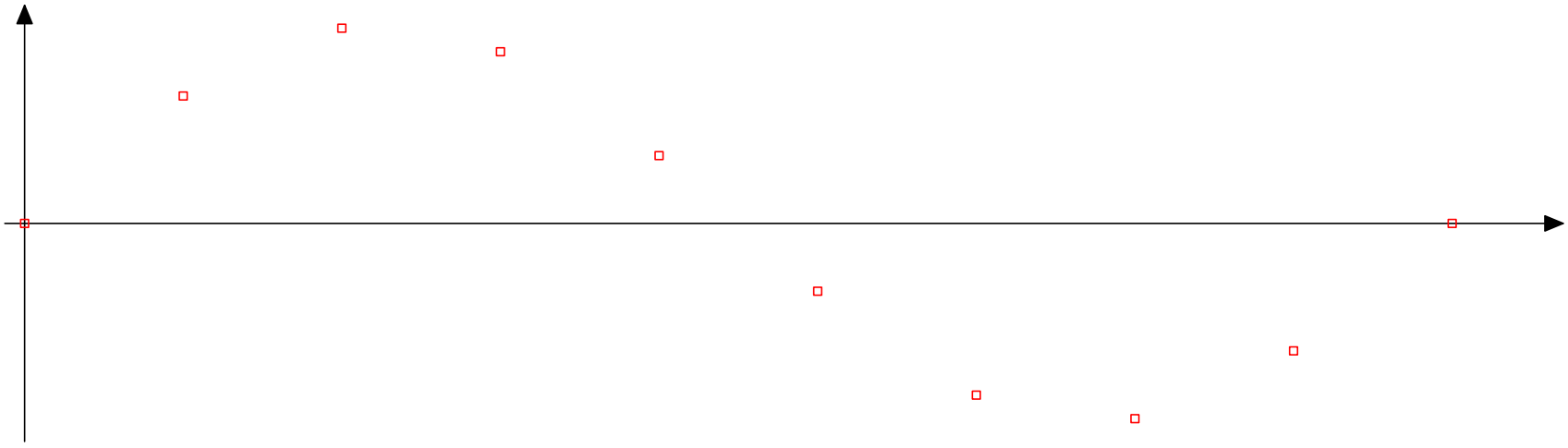


- Wenn der Rand mit berücksichtigt wird bzw. die Breite der Form minimal vergrößert wird, erhalten wir über die dreifache Menge an schwarz gefärbten Rasterpunkten im Vergleich zur Ausgangssituation.

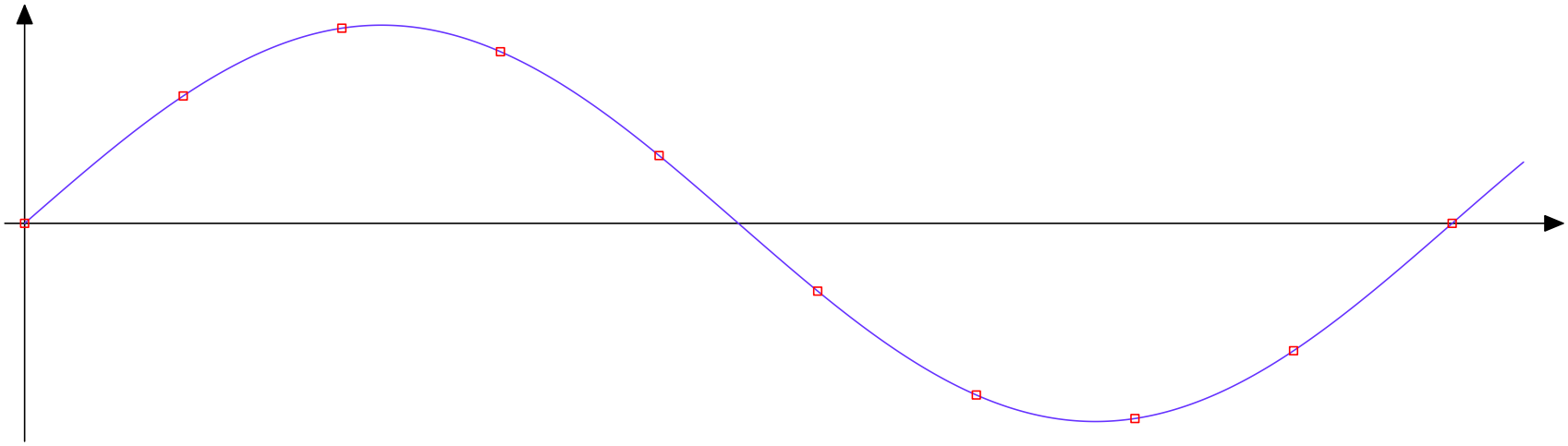
Lösungsansatz für Diagonalen



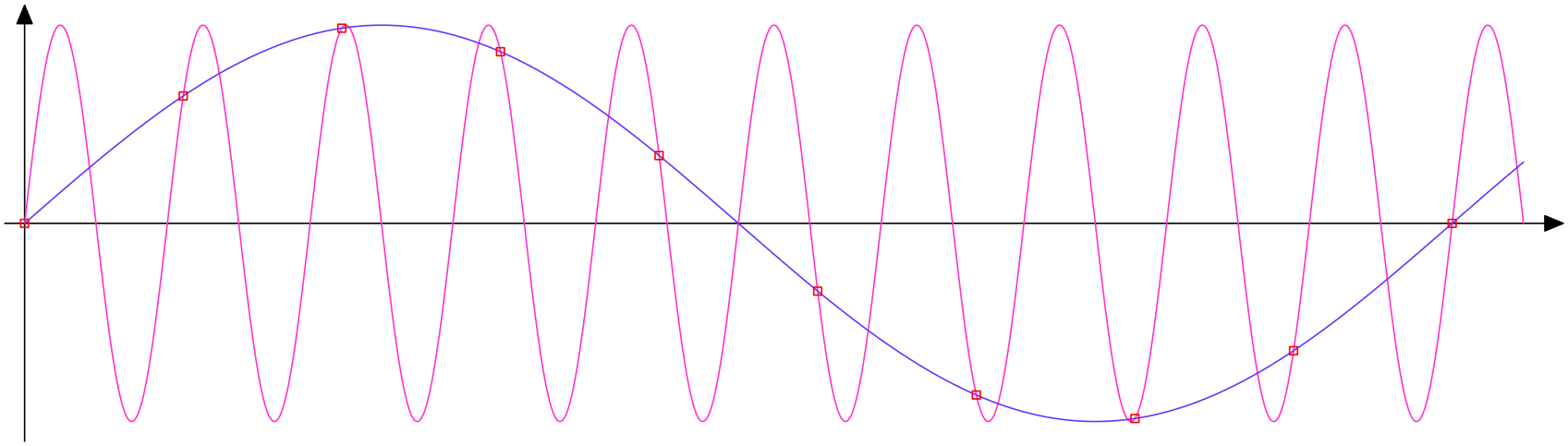
- METAFONT versucht die Frage zu vermeiden, ob ein Rasterfeld zu schwärzen ist, wenn dessen Mittelpunkt genau auf dem Rand der Figur liegt.
- Die Lösung besteht darin, dass METAFONT implizit alle Punkte um $(\delta, \delta\epsilon)$ verschiebt, wobei δ eine sehr kleine Größe ist und $\delta\epsilon < \frac{\delta}{2}$ ist.
- Dies ist auch hilfreich, wenn nicht Figuren zu füllen sind, sondern entlang ihres Randes mit einem Stift zu zeichnen sind. Hier würde eine Kurve $c(t) = (t, t)$ bei $t = \frac{1}{2}$ bei einer Rundung auf ganze Zahlen von $(0, 0)$ auf $(1, 1)$ springen, obwohl die beiden zugehörigen Rasterfelder sich nur an den Eckfeldern berühren. Wenn die obige Verschiebung hinzukommt, dann kommt auch das Rasterfeld $(1, 0)$ bei $t = \frac{1}{2} - 2\delta\epsilon$ hinzu.



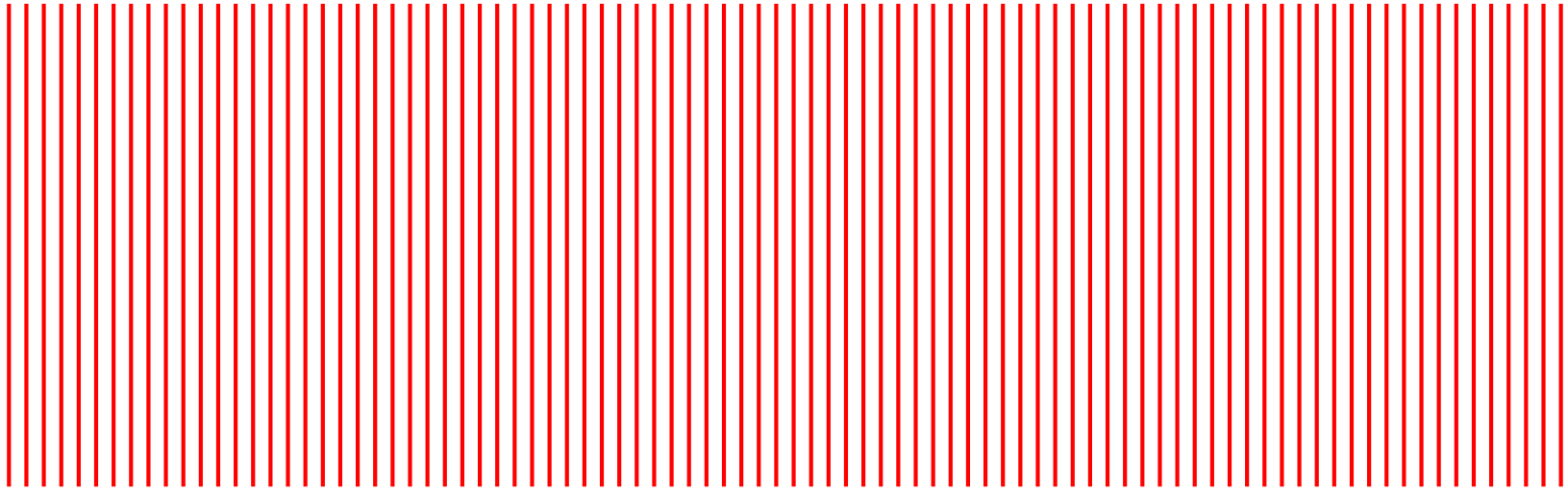
- Ein Problem der Signalverarbeitung: Ein zu digitalisierendes Signal wird zu bestimmten Zeitpunkten gemessen. Aus den Messwerten ist danach eine Näherung des ursprünglichen Signals zu rekonstruieren.



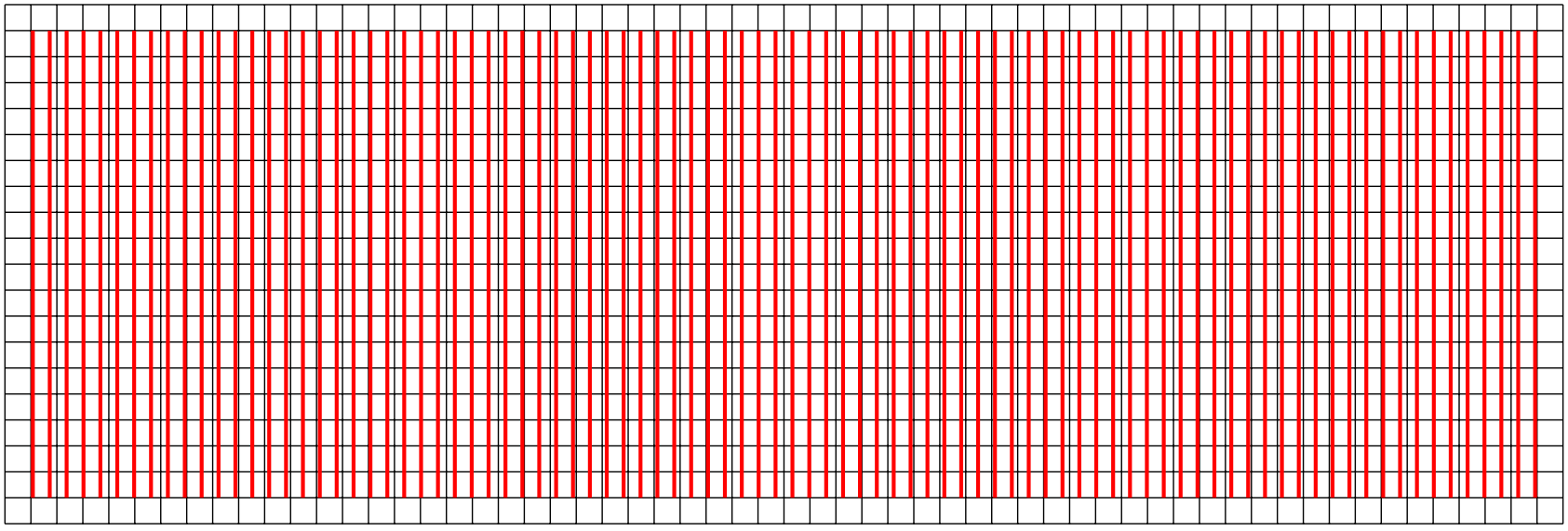
- Durch die gemessenen Punkte passt die dargestellte Sinus-Kurve.



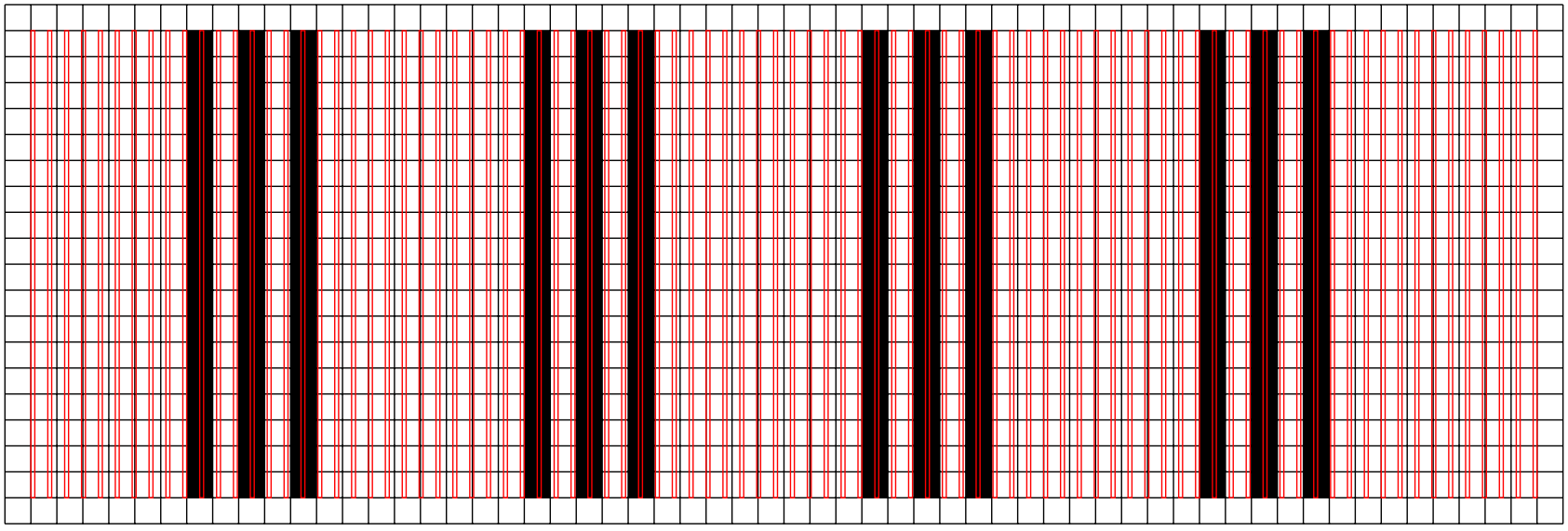
- Aber gleichzeitig passt auch eine sehr viel höherfrequentige Sinus-Kurve durch die gleichen Messwerte.
- Die Signale, die neben dem originalen Signal zu den gleichen Messwerten führen und damit alternativ rekonstruiert werden können, werden Aliase genannt.
- Der Begriff entstand bei dem 1919 entwickelten Überlagerungsempfänger und beschrieb den Effekt, dass ein Sender an zwei beim Oszillator einstellbaren Frequenzen zu hören ist. Im Deutschen wird dafür auch der Begriff Spiegelfrequenz verwendet.



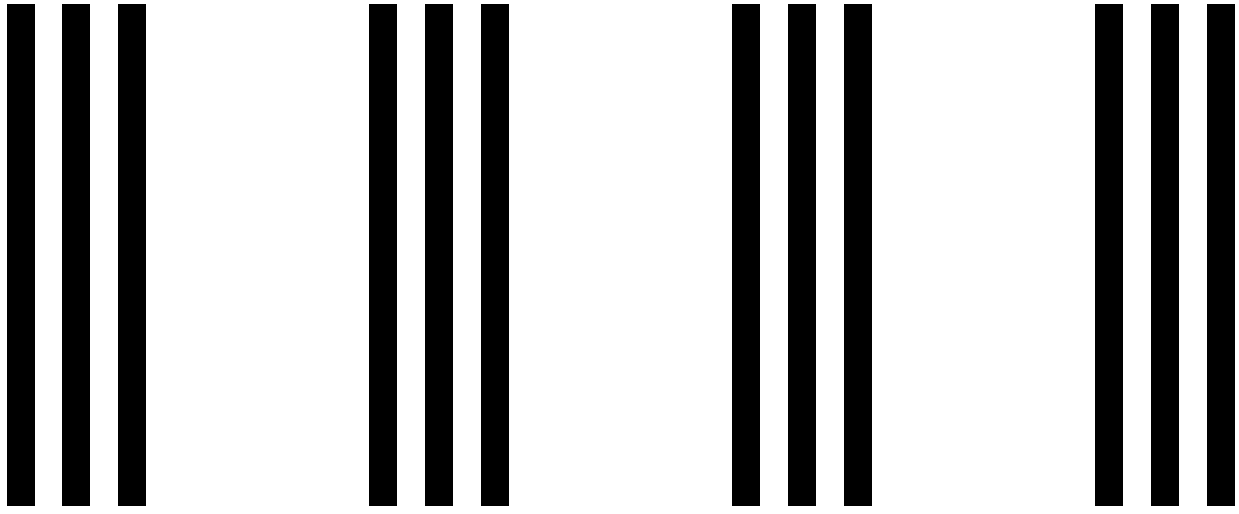
- Die gleiche Problematik findet sich auch bei der Rasterung zweidimensionaler Flächen.
- Gegeben sei ein sehr feines Linienmuster.



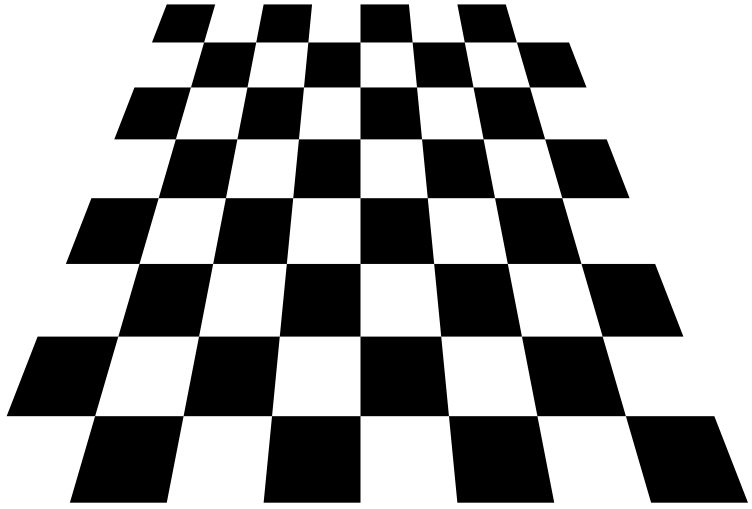
- Die gleiche Problematik findet sich auch bei der Rasterung zweidimensionaler Flächen.
- Gegeben sei ein sehr feines Linienmuster, das mit einem vergleichsweise groben Gitter gerastert wird.



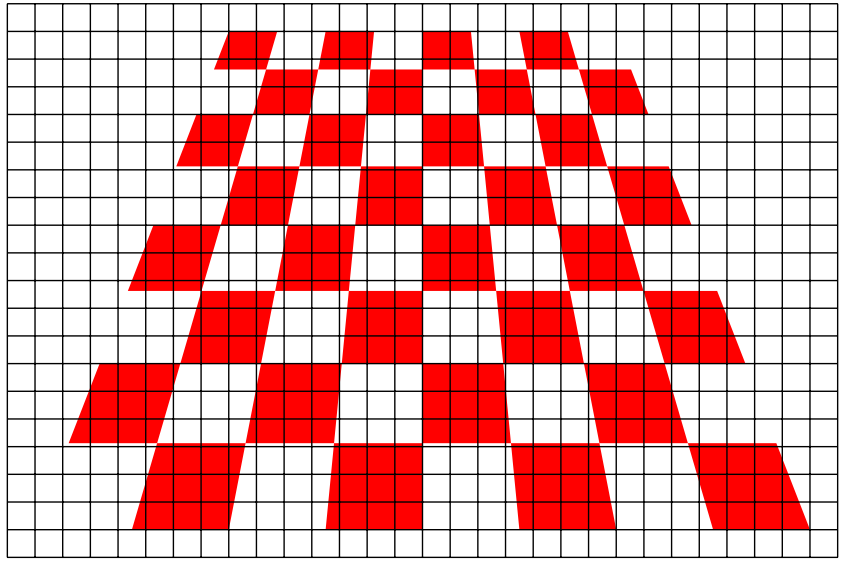
- Die gleiche Problematik findet sich auch bei der Rasterung zweidimensionaler Flächen.
- Gegeben sei ein sehr feines Linienmuster, das mit einem vergleichsweise groben Gitter gerastert wird.
- Dann entsteht ein neues Linienmuster, das keine Ähnlichkeit mit dem ursprünglichen Muster hat.



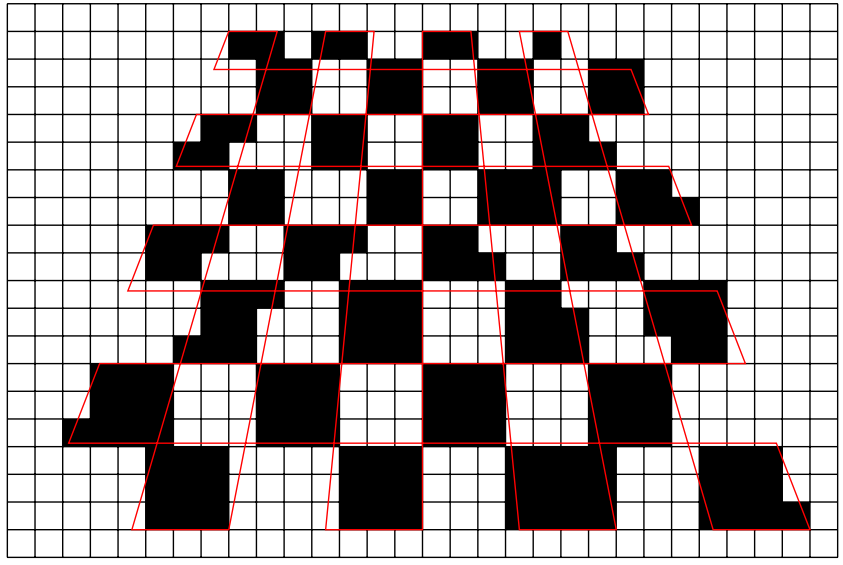
- Bei dem neu entstandenen Linienmuster handelt es sich um einen Alias des Ausgangs-Musters in Bezug auf die gegebene Rasterung.
- Der Begriff »moiré« kommt aus dem Französischen und bezeichnet einen Seidenstoff, der sich durch Gittermuster auszeichnet.



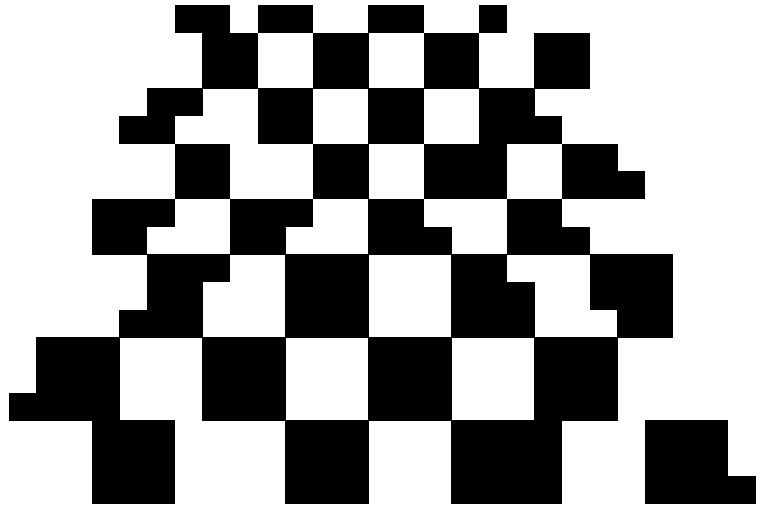
- Gegeben sei ein Schachbrett-Muster, das perspektivisch dargestellt ist mit immer kleiner werdenden Flächen.
- So etwas ist immer eine Herausforderung für eine Rasterung, da das Ausrichten und geringfügige Verformen hier nicht weiterhelfen...



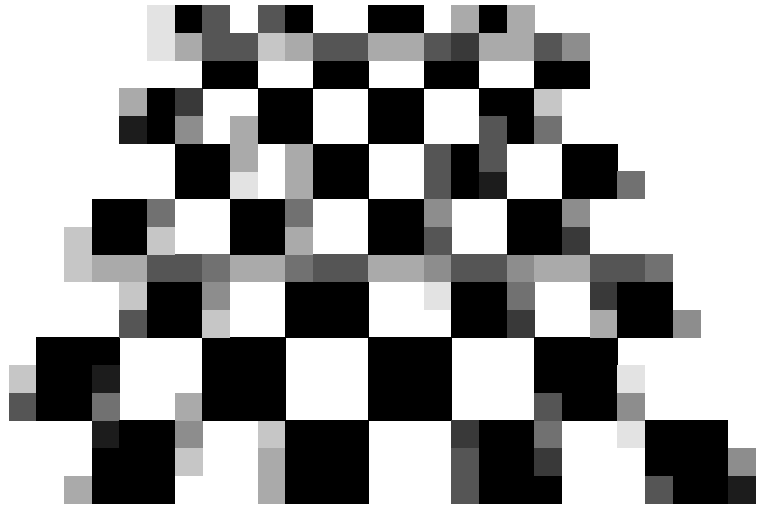
- Gegeben sei ein Schachbrett-Muster, das perspektivisch dargestellt ist mit immer kleiner werdenden Flächen.
- So etwas ist immer eine Herausforderung für eine Rasterung, da das Ausrichten und geringfügige Verformen hier nicht weiterhelfen...



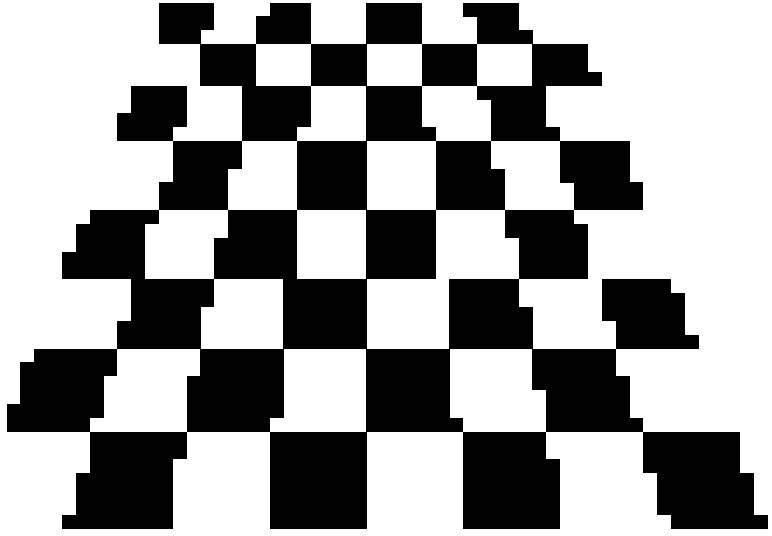
- Wenn das Schachbrett-Muster auf einfache Weise gerastert wird, ist wohl noch die vorderste Reihe in Ordnung. Wenn das Muster jedoch zu klein wird, ist es nach der Rasterung kaum noch zu erkennen.



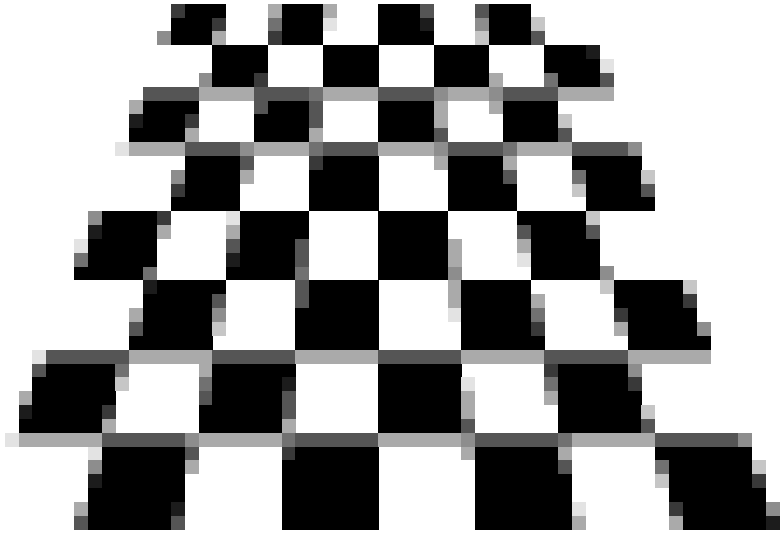
- Wenn das Schachbrett-Muster auf einfache Weise gerastert wird, ist wohl noch die vorderste Reihe in Ordnung. Wenn das Muster jedoch zu klein wird, ist es nach der Rasterung kaum noch zu erkennen.



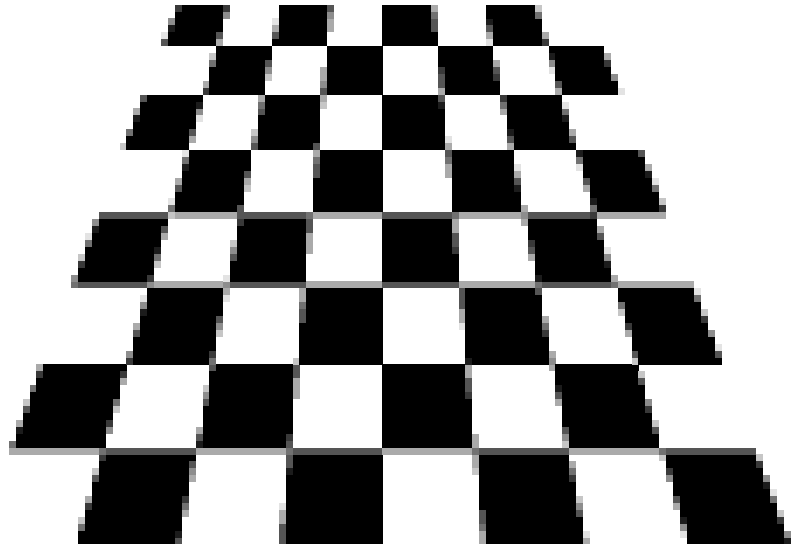
- Wenn auch Graustufen neben Schwarz und Weiß zur Verfügung stehen, bietet es sich an, bei nur teilweise bedeckten Rasterfeldern einen Grauwert zu verwenden.
- In diesem Beispiel wurden in jedem Rasterfeld insgesamt 9 Punkte getestet, ob sie in die Fläche fallen oder nicht. Der Grauwert wurde dann entsprechend des Anteils der in die Kurve fallenden Punkte gewählt.
- Das Verfahren, Graustufen (bzw. Mischfarben) bei teilweise gefüllten Rasterfeldern zu verwenden, wird Anti-Aliasing genannt.



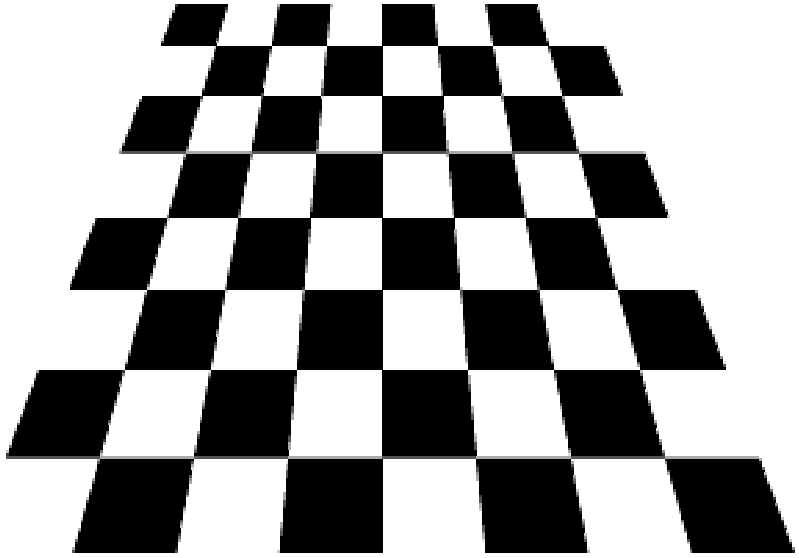
- Wenn wir die Rasterung verdoppeln, können wir das ursprüngliche Muster besser erkennen. Aber es sieht immer noch etwas holprig aus.



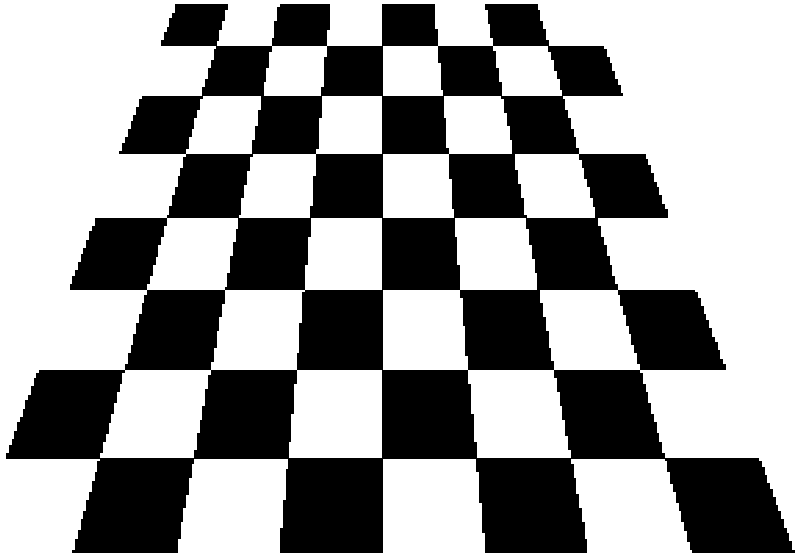
- Anti-Aliasing erleichtert hier dem Auge das Wiedererkennen des bekannten Schachbrettmusters, da die holprigen Kanten wegfallen.
- Das funktioniert aber nur, wenn die Rasterung so fein ist, dass die Grautöne nicht mehr bewusst wahrgenommen werden.



- Unser Auge rastert ebenfalls.
- Ziel ist es daher, ein mit der Rasterung des Mediums darstellbares Bild zu finden, das in Bezug auf das Auge ein Alias zum originalen ungerasterten Bild ist.
- Die Bildschirmauflösung reicht dafür in der Regel aus, jedoch ist Anti-Aliasing unverzichtbar.



- Noch einmal etwas feiner gerastet **mit** Anti-Aliasing.



- Noch einmal etwas feiner gerastet **ohne** Anti-Aliasing.

- Bei GhostScript kann der Anti-Aliasing-Algorithmus getrennt für Schriften und Grafiken eingestellt werden:

  `-dTextAlphaBits=`n mit $n = 1, 2$ oder 4
  `-dGraphicsAlphaBits=`n mit $n = 1, 2$ oder 4

- Die Zahl der Messpunkte beträgt dann 2^n pro Rasterfeld. Bei $n = 1$ ist entsprechend Anti-Aliasing ausgeschaltet.

- Das Format der Type-1-Schriften wurde von Adobe zusammen mit PostScript entwickelt und ist seit 1985 in Nutzung.
- Anders als bei PostScript wurde jedoch das Format der Type-1-Schriften erst 1990 veröffentlicht.
- 1994 wurde das Format als Teil 3 des ISO-Standards 9541 unter dem Titel *Glyph Shape Representation* als Standard akzeptiert.
- Trotz der Konkurrenz durch TrueType sind die Type-1-Schriften bis heute auf dem Markt präsent.
- OpenType ist die Nachfolge-Technologie, die sowohl von Adobe als auch Microsoft unterstützt wird. OpenType unterstützt dabei sowohl die Type-1- als auch die TrueType-Technologie und fügt noch einige Erweiterungen hinzu.
- Literatur: *Adobe Type 1 Format* und Yannis Haralambous: *Fonts & Encodings*, O'Reilly.

Grundsätzlich ähneln Type-1-Schriften weitgehend den Type-3-Schriften, abgesehen davon, dass

- nicht die Standard-Operatoren von PostScript zur Beschreibung der Kurven verwendet werden, dass
- zusätzlich Hinweise zur Optimierung der Rasterung integriert sind (*hints*) und dass
- das Format größtenteils binär und möglicherweise sogar verschlüsselt ist (auf Basis des *eexec*-Operators, der eine einfache Stromchiffre verwendet, deren Algorithmus und von Adobe verwendeter Schlüssel inzwischen öffentlich sind).

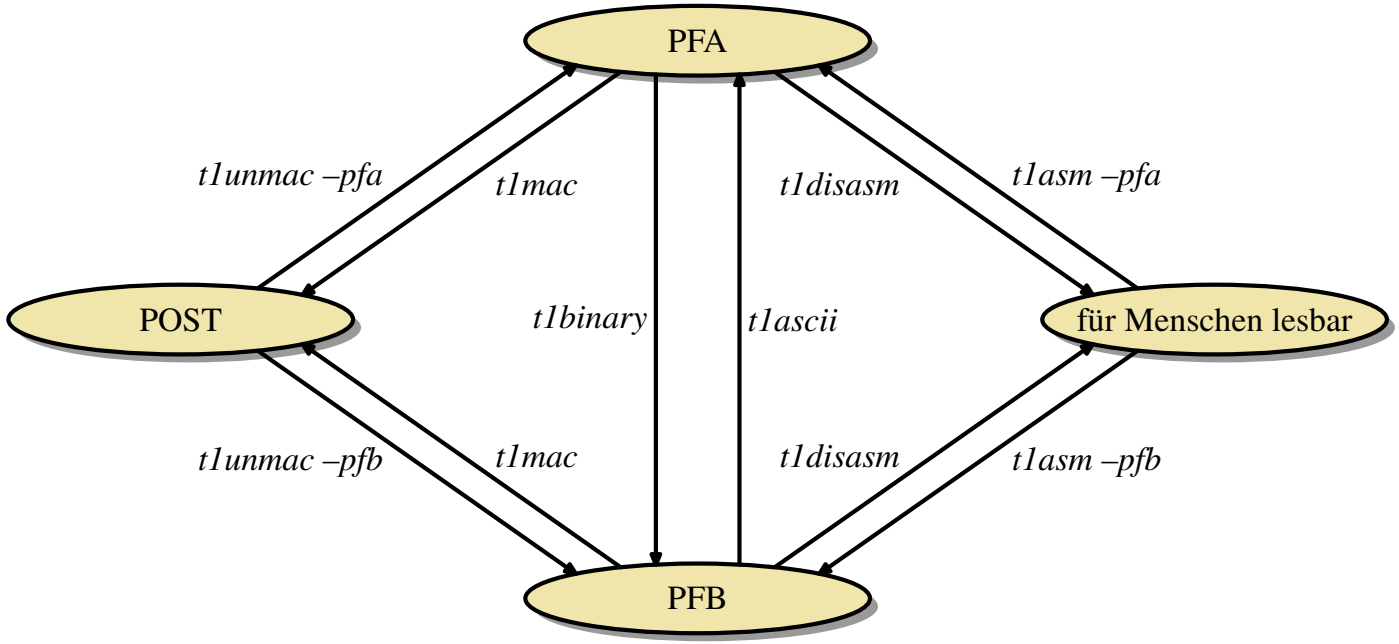
Zu einem Type-1-Schriftschnitt gehören folgende vier Komponenten:

- Ein öffentliches assoziatives Array, das dem eines Type-3-Schriftschnitts ähnelt und in jedem Falle als ASCII-Text abgelegt wird,
- ein privates assoziatives Array, das die Hinweise zur Rasterung aufnimmt und das nur in binärer und möglicherweise sogar nur in verschlüsselter Form vorliegt,
- Prozeduren und
- die Beschreibung der Kurven, die wiederum nur in binärer und möglicherweise sogar in verschlüsselter Form vorliegen.

Varianten des Formats: PFA, PFB und POST

- Unter Unix sind die Formate PFA (A wie ASCII) und PFB (B wie binär) üblich. Bei PFB werden die binären Teile hexadezimal dargestellt.
- Für den Macintosh gibt es das POST-Format.
- Alle drei Varianten lassen sich dank den Werkzeugen von Lee Hetherington und Claire Connelly ineinander überführen.

Konvertierungen zwischen den Formaten



- Konvertierungswerkzeuge von `http://www.lcdf.org/type/`

utmr8a.disasm

```
%!PS-AdobeFont-1.0: NimbusRomNo9L-Regu 1.05
% ...
12 dict begin
/FontName /NimbusRomNo9L-Regu def
% ... more contents of the public dictionary ...
currentdict end
currentfile eexec
dup /Private 15 dict dup begin
% ... private dictionary ...
end
readonly put
noaccess put
dup /FontName get exch definefont pop
mark currentfile closefile
cleartomark
```

- Hier und im folgenden werden Auszüge des Times-Roman-Schriftschnitts von URW++ präsentiert, der unter der GPL zur Verfügung steht. Betrachtet wird der Text, den *t1disasm* generiert hat.

Schlüssel	Typ	Bedeutung
FontType	integer	1
FontMatrix	array	Transformations-Matrix für die Kurven innerhalb der Definitionen für die Zeichen; typisch ist ein 1000 × 1000-Koordinatensystem
FontInfo	dict	assoziatives Array mit weiteren Feldern, die den Schriftschnitt beschreiben
Encoding	dict	bildet Werte aus dem Bereich $[0, 255]$ in Namen für die einzelnen Zeichen ab
FontBBox	array	Bounding-Box aller übereinander gezeichneter Zeichen
CharStrings	dict	Kurven der einzelnen Zeichen

Schlüssel	Typ	Bedeutung
FontName	name	Name des Schriftschnitts
PaintType	integer	1 (fill) oder 2 (stroke)
Private	dict	Hinweise zur Rasterung
UniqueID	integer	Eindeutige Zahl zwischen 0 und $2^{24} - 1$
XUID	array	Alternativ zu UniqueID: Eindeutige Folge von Zahlen, wobei die erste von Adobe vergeben wird

utmr8a.disasm

```
/FontName /NimbusRomNo9L-Regu def
/PaintType 0 def
/WMode 0 def
/FontBBox {-168 -281 1000 924} readonly def
/FontType 1 def
/FontMatrix [0.001 0.0 0.0 0.001 0.0 0.0] readonly def
/Encoding StandardEncoding def
/UniqueID 5020931 def
```

- Die Einträge für `/CharStrings` und `/Private` kommen erst nachträglich hinzu.

Einträge im öffentlichen assoziativen Array 410

utmr8a.disasm

```
/FontInfo 10 dict dup begin
/version (1.05) readonly def
/Notice ((URW)++,Copyright 1999 by (URW)++ ...) readonly def
/Copyright (Copyright (URW)++ ...) readonly def
/FullName (Nimbus Roman No9 L Regular) readonly def
/FamilyName (Nimbus Roman No9 L) readonly def
/Weight (Regular) readonly def
/ItalicAngle 0.0 def
/isFixedPitch false def
/UnderlinePosition -100 def
/UnderlineThickness 50 def
end readonly def
```

- `/FontInfo` gehört mit zum öffentlichen assoziativen Array und ist selbst eines.

- Im Prinzip können die Namen der einzelnen Zeichen, die in `/Encoding` aufgenommen werden, beliebig gewählt werden.
- In Bezug auf PDF kann das ein Problem sein: Wenn interaktiv nach einem Text in einem Dokument gesucht wird, wie kann festgestellt werden, welches Zeichen im Schriftschnitt welchem Zeichen (etwa in Unicode) entspricht?
- Die Lösung besteht darin, dass die Namen der einzelnen Zeichen gewissen Konventionen entsprechen müssen.
- Das wurde ursprünglich festgelegt über ISO 10036 zu Zeiten als Unicode noch nicht zur Verfügung stand. Siehe `http://10036ra.org/`
- Inzwischen wurde die Adobe Glyph List (AGL) eingefroren: `https://github.com/adobe-type-tools/agl-aglfn/blob/master/glyphlist.txt`
- Neue Namen werden entsprechend der *Adobe Glyph List Specification* gebildet, die sich jeweils einer Sequenz von Unicode-Codepoints zuordnen lassen.

utmr8a.disasm

```
currentfile eexec
```

- `currentfile` ist ein Operator, der einen Verweis auf die aktuelle Eingabequelle (mit dem Type-1-Schriftschnitt) zurückliefert.
- `eexec` nimmt den Verweis auf eine Eingabedatei, um ihn die damit referenzierte Eingabe zu entschlüsseln und anschließend auszuführen.
- In einer PFA-Datei ist bis zu diesem Punkt (abgesehen von den ersten 6 Bytes) alles Klartext und erst danach beginnt das eigentliche binäre Format.

Zum assoziativen Array `/Private` gehören folgende Komponenten:

- Globale Hinweise zur Rasterung,
- mehrere historische Parameter,
- einige Operatoren, die für die Prozeduren benötigt werden,
- die Prozeduren selbst (`/Subrs`) und
- weitere Prozeduren (`/OtherSubrs`).

Zu den binären Daten gehört ferner noch `/CharStrings`, das jedoch erst nachträglich in das übergeordnete assoziative Array eingeordnet wird. Es enthält die Kurvendefinitionen der einzelnen Zeichen einschließlich individueller Hinweise zur Rasterung.

utmr8a.disasm

```
/BlueValues [-20 0 450 470 662 682] def
/BlueScale 0.039625 def
/StdHW [30] def
/StdVW [85] def
/StemSnapH [30 38 43 53 60 73] def
/StemSnapV [78 85 91 103 109 115] def
```

- `/BlueValues` spezifiziert die y-Koordinaten für die wichtigsten Höhenlinien.
- Die optionalen Parameter `/BlueScale` und `/BlueShift` spezifizieren, wie weit die Ausrichtung aufgrund der Höhenlinien erfolgt.
- `/StdHW` und `/StemSnapH` spezifizieren die wichtigsten horizontalen Strichstärken. Analog gibt es `/StdVW` und `/StemSnapV` für die vertikalen Strichstärken.

- Die `/BlueValues` spezifizieren Höhenbereiche für den gesamten Schriftschnitt.
- Sie werden typischerweise genutzt, um die Basislinie zu markieren, die x-Höhe und die Versalhöhe.
- Glatt abschließende Versalien wie etwa das „H“ tangieren den Bereich, gehen aber nicht in diesen hinein.
- Bei runden Zeichen (wie etwa beim „O“ oder „o“) oder spitz abschließenden (wie beim „h“) dringt der Pfad in den markierten Bereich hinein (Überschuss), geht aber über die andere Linie nicht hinaus.

utmr8a.disasm

```
/password 5839 def
/MinFeature {16 16} def
```

- Diese Parameter sind entsprechend der Spezifikation von Adobe aus historischen Gründen notwendig.
- Eine weitere Erklärung gibt es nicht dazu. Entsprechend finden sie sich in dieser Form in jedem Type-1-Schriftschnitt.

utmr8a.disasm

```
/RD {string currentfile exch readstring pop} executeonly def
/ND {noaccess def} executeonly def
/NP {noaccess put} executeonly def
```

- Diese Operatoren müssen ebenfalls innerhalb des `/Private`-Arrays definiert werden.
- `/ND` bzw. `/NP` fügen eine Prozedur in ein assoziatives Array bzw. ein reguläres Array ein und beschränken den Zugriff.

utmr8a.disasm

```
/Subrs 251 array
dup 0 {
   3 0 callothersubr pop pop setcurrentpoint return
} NP
dup 1 {
   0 1 callothersubr return
} NP
dup 2 {
   0 2 callothersubr return
} NP
dup 3 {
   return
} NP
% ...
ND
```

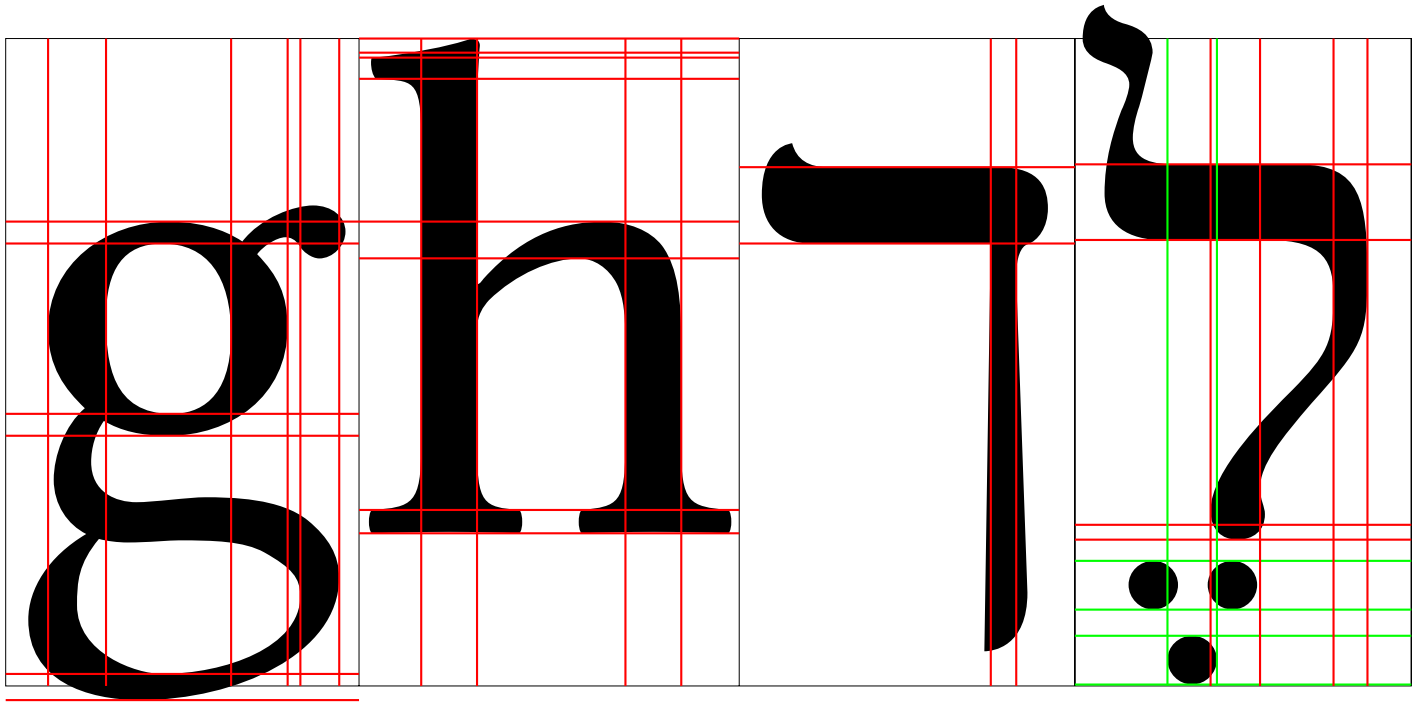
- Die Prozeduren sind in einem regulären Array und somit nur über die jeweilige Nummer erreichbar.
- Die Prozeduren 0 bis 3 sind vordefiniert. Der entsprechende Text wird deswegen normalerweise ignoriert.
- Mit `callothersubr` erfolgt ein Aufruf der anderen Prozeduren. Mit `return` endet die Ausführung einer Prozedur. Ein `return` ist zwingend.

- Der Operator `hsbw` mit den beiden Parametern *sbx* und *wx* spezifiziert mit (*sbx*, 0) den am weitesten links liegenden Punkt der Kurve und den nachfolgenden Weiterbewegungsvektor mit (*wx*, 0).
- Ferner wird die aktuelle Position auf (0, *sbx*) gesetzt. Diese Position geht jedoch nicht in die Kurve selbst ein. Diese muss zwingend mit einer relativen Bewegung beginnen.
- Dies muss der erste Operator sein.
- Alternativ kann `sbw` verwendet werden, falls der Weiterbewegungsvektor eine von 0 abweichende y-Komponente hat.

Operator	Beschreibung
dx dy rmoveto	bewegt sich relativ zur aktuellen Position um (*dx*, *dy*)
dx hmoveto	ist äquivalent zu *dx* `0 rmoveto`
dy vmoveto	ist äquivalent zu `0` *dy* `rmoveto`
dx dy rlineto	Linie mit relativen Koordinaten
dx hlineto	horizontale Linie
dy vlineto	vertikale Linie
dx1 dy1 dx2 dy2 dx3 dy3 rcurveto	Bézier-Kurve, wobei alle Koordinaten relativ zum Ausgangspunkt genommen werden
dx1 dx2 dy2 dy3 hvcurveto	äquivalent zu *dx1* `0` *dx2 dy2* `0` *dy3* `rcurveto`
dy1 dx2 dy2 dx3 vhcurveto	äquivalent zu `0` *dy1 dx2 dy2 dx3* `0 rcurveto`
closepath	Ende einer Kurve; mehrere Kurvendefinitionen sind zulässig; der aktuelle Punkt bleibt bestehen
endchar	Ende einer Zeichendefinition

Operator	Beschreibung
hstem	Spezifikation eines horizonalen Strichs mit der unteren y-Koordinate und der Höhe
vstem	Spezifikation eines vertikalen Schafts mit der niedrigeren x-Koordinate und der Breite
vstem3	adressiert die Problematik des »m«, indem mit insgesamt 6 Parametern die Angaben für drei einzelne `vstem`-Operatoren zusammengefasst werden.
hstem3	analog zu vstem3, jedoch horizontal

Operatoren für individuelle Hinweise zur Rasterung 422



- Das Beispiel zeigt alle mit *hstem* oder *vstem* markierten vertikalen und horizontalen Schäfte der Buchstaben „g“, „h“, dem Kapf (in der Variante für das Ende eines Worts) und das Lamed mit dem Vokalzeichen Segol (drei Punkte), das als Akzent darunter gesetzt wird.
- Schriftschnitt: *LibertinusT1Math*

utmr8a.disasm

```
2 index /CharStrings 316 dict dup begin
% ...
/a {
   % ...
} ND
% ...
/.notdef {
   % ...
} ND
end
```

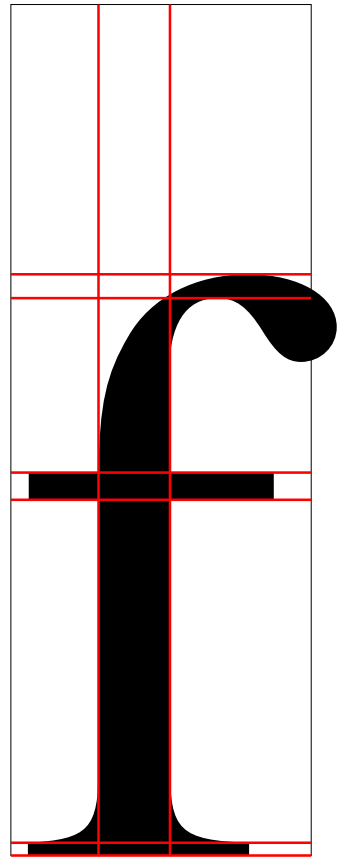
- Da die Definition textuell in der Deklaration von `/Private` eingebettet ist, muss mit `2 index` explizit das übergeordnete assoziative Array referenziert werden (das immer noch auf dem Stack an genannter Position liegt).
- Die einzelnen Prozeduren enthalten jeweils individuelle Hinweise zur Rasterung und die Kurvendefinition. Hierbei dürfen nur die speziellen Operatoren verwendet werden.
- Die Namen müssen mit denjenigen übereinstimmen, die in `/Encoding` referenziert werden.

Definition des Schnitts für den Buchstaben f 424

utmr8a.disasm

```
/f {
   20 333 hsbw
   0 15 hstem 418 32 hstem 655 28 hstem 83 84 vstem
   289 450 rmoveto
   -123 hlineto 116 vlineto
   58 19 31 38 vhcurveto
   21 0 14 -10 18 -29 rrcurveto
   16 -26 12 -10 17 0 rrcurveto
   23 19 18 23 hvcurveto
   36 -44 26 -60 vhcurveto
   -62 0 -53 -27 -26 -46 rrcurveto
   -26 -44 -8 -36 -1 -80 rrcurveto
   -82 hlineto -32 vlineto 82 hlineto
   -314 vlineto
   0 -73 -11 -12 -72 -4 rrcurveto
   -15 vlineto 260 hlineto 15 vlineto
   -82 3 -11 11 0 75 rrcurveto
   314 vlineto 122 hlineto
   closepath endchar
} ND
```

Definition des Schnitts für den Buchstaben f



- Die Ursprünge der TrueType-Schriften liegen in den Bemühungen von Apple und Microsoft, unabhängig von der Schrifttechnologie von Adobe zu werden.
- Dies war insbesondere für die Rasterung von Schriften am Bildschirm relevant.
- Die ursprüngliche Fassung geht auf den finnischen Entwickler Sampo Kaasila zurück, der für Apple arbeitete.
- Apple integrierte TrueType-Schriften zuerst 1991 bei MacOS 6; Microsoft folgte 1992 bei Windows 3.1.
- Beide arbeiteten unabhängig voneinander weiter, was zu zahlreichen Inkompatibilitäten geführt hat.

- Das Dateiformat ist vollumfänglich binär, aber vollständig dokumentiert.
- Dokumentation gibt es von Apple und von Microsoft, die nicht deckungsgleich sind:
  - `http://developer.apple.com/fonts/TTRefMan/index.html`
  - `http://www.microsoft.com/typography/otspec/default.htm`

- Es gibt von Just van Rossum ein frei verfügbares Werkzeug, mit dessen Hilfe TTF-Dateien in ein XML-Format und wieder zurück überführt werden können:
  `https://github.com/fonttools/fonttools`
- Unter Debian steht das Werkzeug über das Paket *fonttools* zur Verfügung.
- Dies wird im folgenden verwendet werden, um einzelne Teile einer TrueType-Datei zu betrachten und zu analysieren.

```
hochwanner$ ttx -l TimesNewRoman.ttf
Listing table info for "TimesNewRoman.ttf":
    tag      checksum   length   offset
    ----   ----------  -------  -------
    LTSH   0x814d6df0       663      316
    OS/2   0xd70a8dc7        86      980
    PCLT   0x59d381e3        54     1068
    VDMX   0x4e236882      4500     1124
    cmap   0x80a4ee32      1750     5624
    cvt    0xf3deda81      1990     7376
    fpgm   0x3775a72f      1395     9368
    gasp   0x00180009        16    10764
    glyf   0x784bf0ce    144434    10780
    hdmx   0x756f659c     15944   155216
    head   0xc17591b0        54   171160
    hhea   0x0f0308af        36   171216
    hmtx   0xb6f7a4a5      2636   171252
    kern   0xa677acd1      5220   173888
    loca   0x03143262      2640   179108
    maxp   0x08fa069b        32   181748
    name   0xf80cc1bf      4881   181780
    post   0x9141e4c3      4898   186664
    prep   0xc6afb762      3874   191564

hochwanner$
```

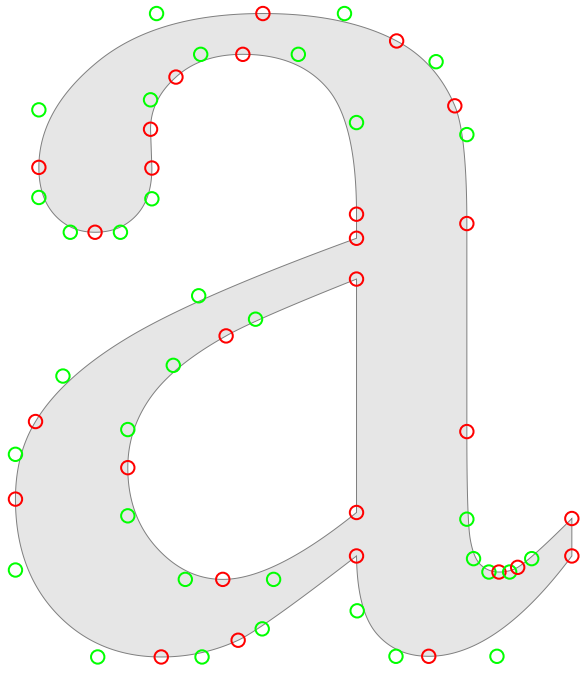
- Das TrueType-Dateiformat sieht auf der obersten Ebene ein Verzeichnis einzelner Tabellen vor. Die Tabellennamen haben dabei bis zu vier Zeichen und belegen einen zusammenhängenden Abschnitt der Datei.
- Die wichtigsten Tabellen sind *head* (globale Definitionen), *cmap* (Abbildung zwischen Zeichensätzen und den im Schriftschnitt enthaltenen Zeichen), *glyf* (Kurvendefinitionen der einzelnen Zeichen zusammen mit Instruktionen zur ihrer Rasterung), *hhea*, *OS/2* und *hmtx* (globale und individuelle Metriken für den horizontalen Satz) und *kern* (Kerning-Tabellen).

Kurvendefinition bei TrueType

TimesNewRoman.ttx

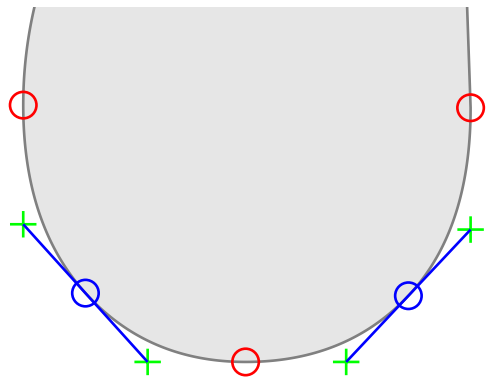
```
<TTGlyph name="a" xMin="73" yMin="-19" xMax="905" yMax="943">
  <contour>
    <pt x="583" y="132" on="1"/>
    <pt x="442" y="23" on="0"/>
    /* ... */
    <pt x="584" y="50" on="0"/>
  </contour>
  <contour>
    <pt x="583" y="197" on="1"/>
    <pt x="583" y="546" on="1"/>
    <pt x="432" y="486" on="0"/>
    <pt x="388" y="461" on="1"/>
    <pt x="309" y="417" on="0"/>
    <pt x="241" y="321" on="0"/>
    <pt x="241" y="264" on="1"/>
    <pt x="241" y="192" on="0"/>
    <pt x="327" y="97" on="0"/>
    <pt x="383" y="97" on="1"/>
    <pt x="459" y="97" on="0"/>
  </contour>
  <instructions><assembly>
      /* ... instructions for rasterization ... */
    </assembly></instructions>
</TTGlyph>
```

Kurvendefinition bei TrueType



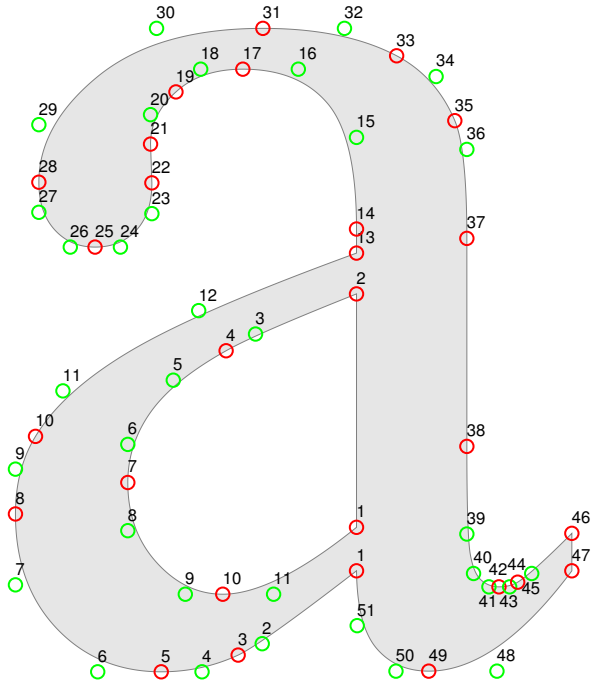
Buchstabe „a“ bei *TimesNewRoman*

- Die Kurvendefinition besteht aus beliebig vielen einzelnen jeweils geschlossenen Kurven (*contour*).
- Jede Kurvenspezifikation besteht aus beliebig vielen Punkten, die entweder auf der Kurve liegen ($on = 1$, hier rot) oder außerhalb ($on = 0$, hier grün).
- Daraus ergibt sich dann eine Folge quadratischer Bézier-Kurven und gerader Linien.

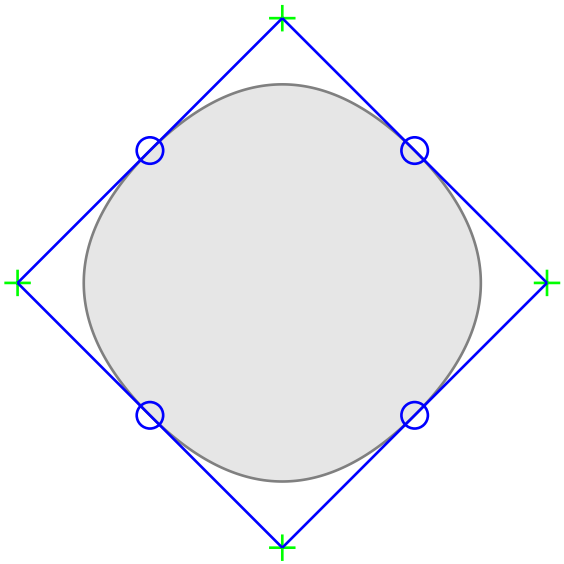


- Wenn mehrere Punkte außerhalb der zu zeichnenden Kurve aufeinanderfolgen ($on = 0$, in der Zeichnung grün), dann wird jeweils implizit genau dazwischen ein Punkt auf der Kurve gewählt (in der Zeichnung blau).
- Die jeweiligen Verbindungslinien (ebenfalls blau) zwischen zwei aufeinanderfolgenden Außenpunkten sind dann tangential zu der zu zeichnenden Kurve.

Kurvendefinition bei TrueType

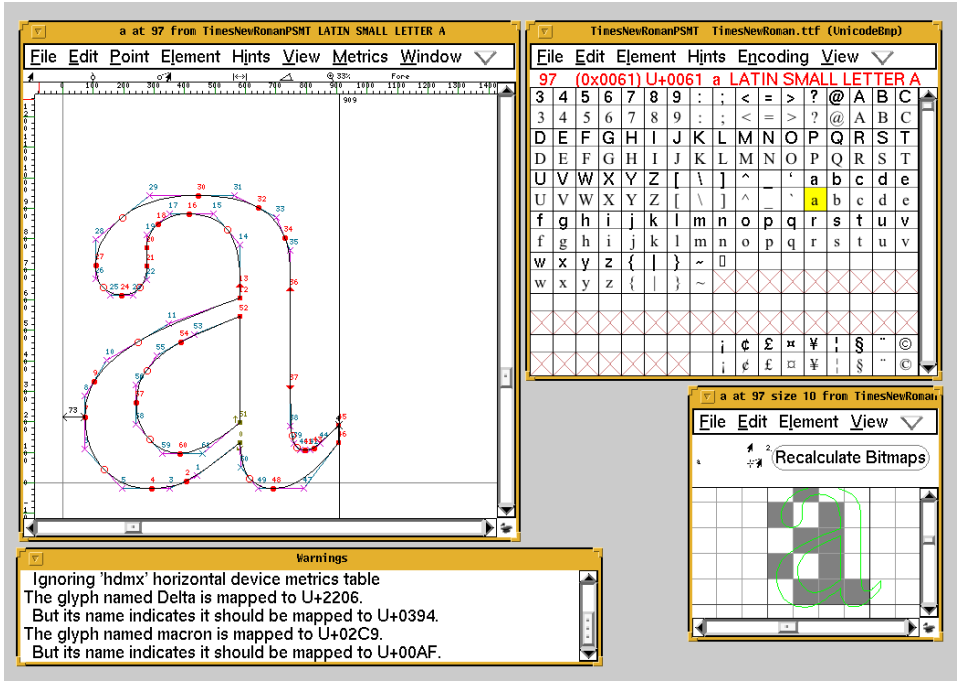


- Die einzelnen Punkte sind so aneinanderzureihen, dass die zu füllende Fläche entsprechend der gewählten Richtung immer rechts liegt.
- Außenlinien werden entsprechend im Uhrzeigersinn, Augen gegen den Uhrzeigersinn gezeichnet.
- Gefüllt wird anschließend entsprechend der *non-zero winding number rule*.

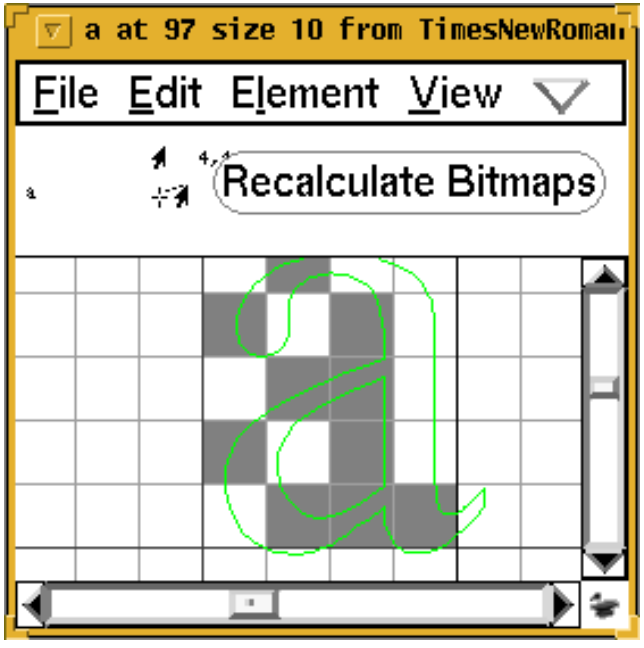


- Prinzipiell sind auch Kurvendefinitionen zulässig, bei denen alle Punkte außerhalb der Kurve liegen.
- Das Beispiel demonstriert dies an den Punkten $(200, 300)$, $(300, 400)$, $(400, 300)$ und $(300, 200)$, die ein Quadrat bilden.

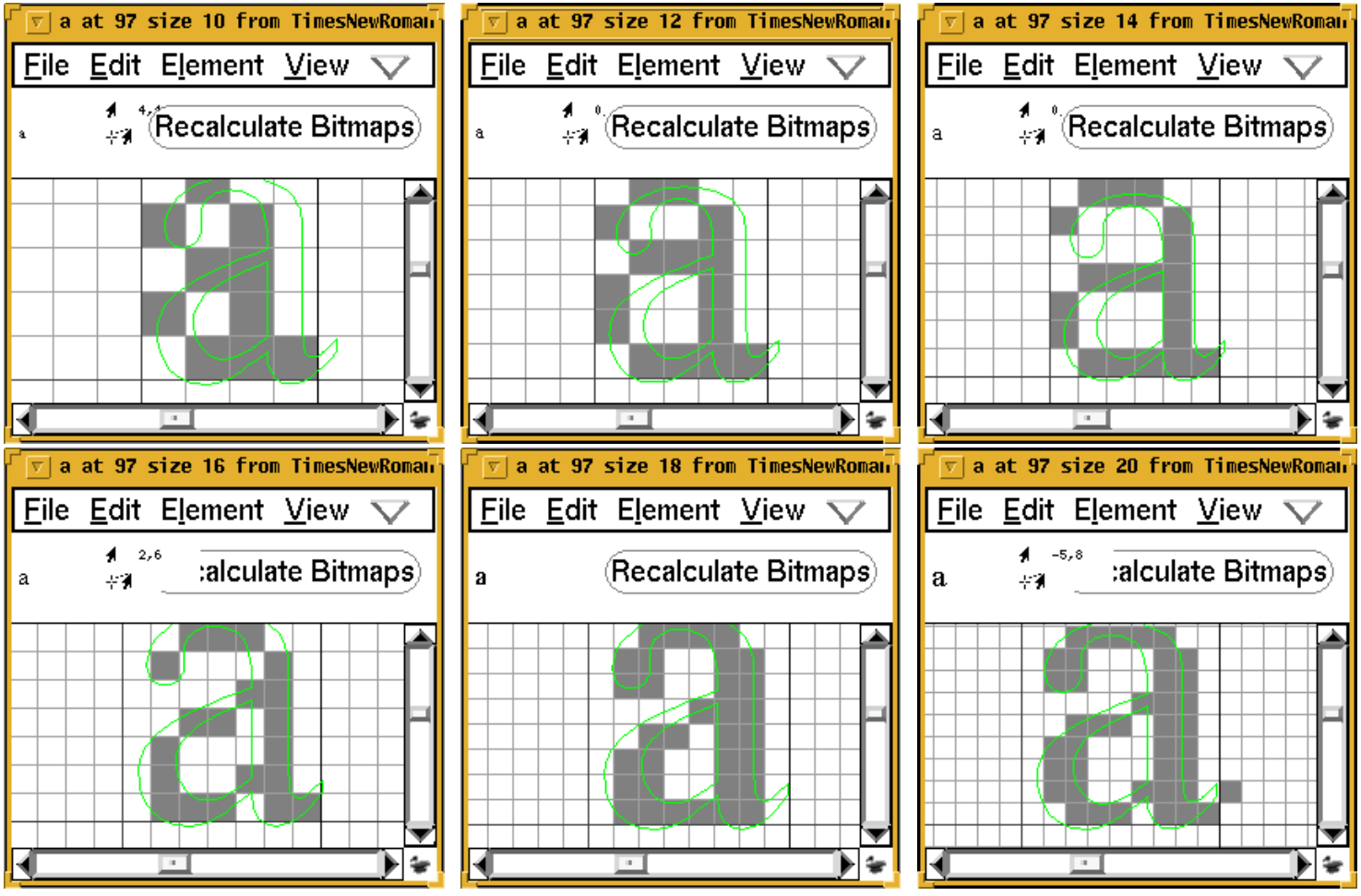
- Für die Modellierung erscheinen kubische Bézier-Kurven einfacher, da die beiden Tangenten auf sehr einfache Weise unabhängig voneinander gewählt werden können.
- Bei quadratischen Kurven sind daher mehr einzelne Bézier-Kurven notwendig.
- Die quadratischen Bézier-Kurven lassen sich jedoch sehr viel effizienter berechnen. Insbesondere vereinfacht sich die Rasterisierung.
- Auch werden unerwünschte Knicks und Scheitelpunkte bei quadratischen Kurven zuverlässig vermieden.



- Zur Visualisierung und insbesondere zum Testen der Rasterisierung empfiehlt sich der Einsatz des freien Software-Pakets *fontforge*.
- `http://fontforge.github.io/`
- Bei Debian steht es als Paket (*fontforge*) zur Verfügung und unter dem gleichen Namen gibt es das Paket auch bei den Macports.

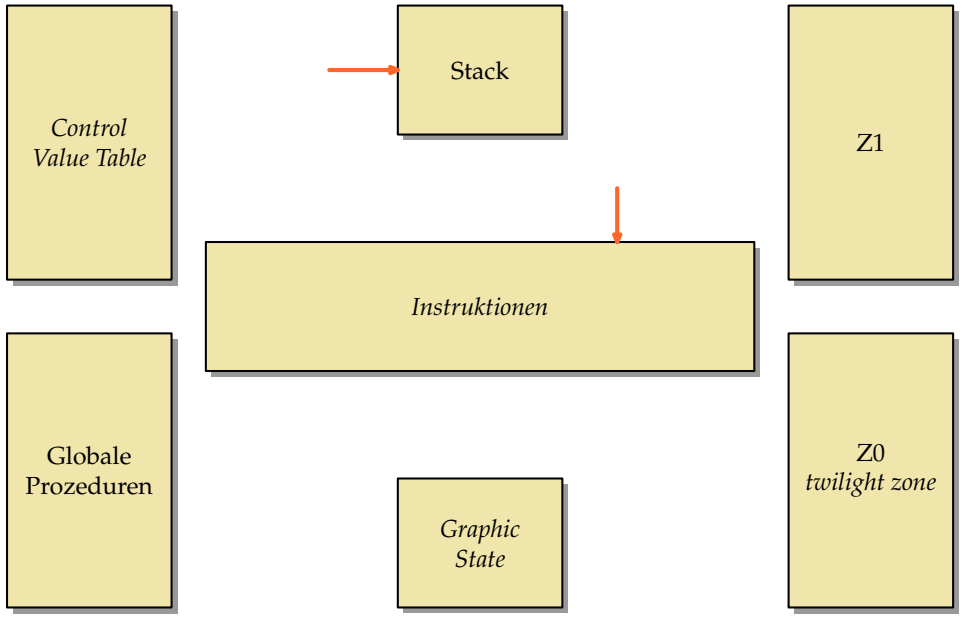


- Anders als bei Type1-Schriften ist die Rasterung bei TrueType-Schriftschnitten frei programmierbar.
- Das Programm kann in Abhängigkeit von der Rasterung sämtliche Punkte frei verschieben und auch ganze Teile (wie etwa Serifen) zum Verschwinden bringen, wenn die Auflösung zu gering ist.
- In dem gezeigten Beispiel ist das „a“ trotz der Rasterung auf 4×5 Felder noch klar zu erkennen.


a at 97 size 10 from TimesNewRoman
File Edit Element View
Recalculate Bitmaps
a at 97 size 12 from TimesNewRoman
File Edit Element View
Recalculate Bitmaps
a at 97 size 14 from TimesNewRoman
File Edit Element View
Recalculate Bitmaps
a at 97 size 16 from TimesNewRoman
File Edit Element View
a at 97 size 18 from TimesNewRoman
File Edit Element View
Recalculate Bitmaps
a at 97 size 20 from TimesNewRoman
File Edit Element View

- Zu den TrueType-Formatbeschreibungen gehört eine virtuelle Maschine, die frei programmiert werden kann, um die Kontrollpunkte der Kurve an eine Rasterung anzupassen.
- Jedem einzelnen Zeichen kann eine entsprechende Prozedur beigefügt werden. Darüber hinaus werden globale Prozeduren unterstützt, die von den für ein einzelnes Zeichen zuständigen Prozeduren aufgerufen werden können.
- Für die globalen Prozeduren gibt es die Tabellen *fpgm* und *prep*. In *cvt* können globale Variablen zusammen mit ihren Initialwerten spezifiziert werden.
- Das TrueType-Format enthält nur den Code der virtuellen Maschine, der in dieser Form nur schwer lesbar ist.

Virtuelle Maschine für die Instruktionen



- Die virtuelle Maschine führt die in einem separaten Speicher abgelegten Instruktionen aus.
- Zugänglich ist ein Stack für Berechnungen und Parameterübergabe, globale, bereits vorinitialisierte Variablen (CVT), der aktuelle Grafikzustand und die in Zonen Z1 und Z0 organisierten Kontrollpunkte der Kurven.

Folgende Datentypen werden unterstützt:

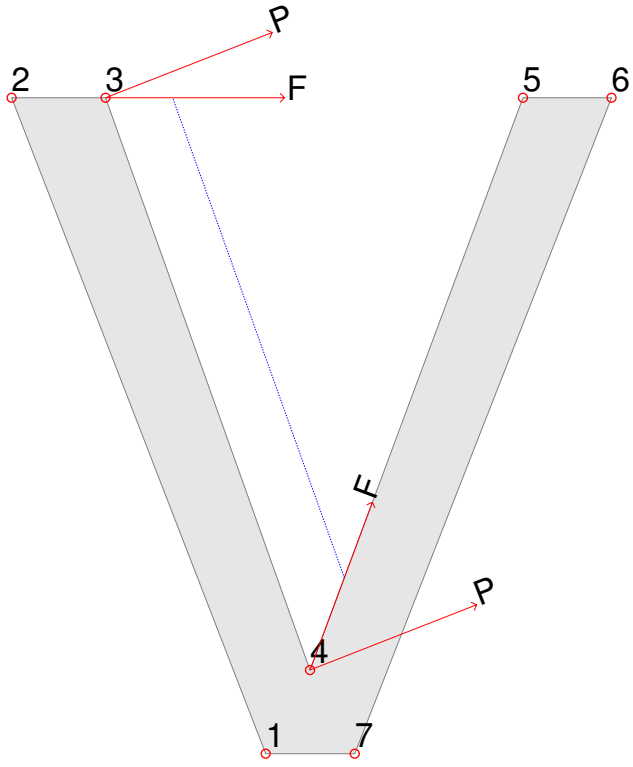
- *long* und *ulong*: ganze Zahlen, jeweils 32 Bit (für die Indizierung, insbesondere von Punkten, und für Boolean-Werte)
- *f26.6*: Zahl in Fixpunkt-Darstellung mit 26 binären Stellen vor dem Komma und 6 binären Stellen nach dem Komma (wird zur Repräsentierung der Koordinaten und von Distanzen verwendet)
- *f2.14*: Fixpunkt-Darstellung für Koordinaten normierter Richtungsvektoren (für Freiheits- und Projektionsvektor)

Der Stack besteht nur aus 32-Bit-Wörtern. Arithmetische Operationen gibt es nur für Werte des Typs *f26.6*.

- Alle Kontrollpunkte der einzelnen Kurven sind entsprechend der vorgegebenen Reihenfolge durchnummeriert (beginnend ab 1) und werden über eine Zone und einen Index ausgewählt.
- Für jeden Punkt gibt es die Koordinaten und die Information, ob dieser auf oder außerhalb der Kurve liegt.
- In der Zone *Z1* sind alle vorgegebenen Punkte zu finden.
- Die Zone *Z0* (auch *twilight zone* genannt) ist zu Beginn leer, kann aber frei verwendet werden.

Ziel der Instruktionen

- Ziel der Instruktionen ist es, die Punkte in *Z1* an das vorgegebene Raster anzupassen.
- Alle Punkte können individuell verschoben werden. Es ist auch möglich, die Eigenschaft zu verändern, ob ein Kontrollpunkt auf der Kurve oder außerhalb davon liegt.
- Das Verschieben der Punkte unterliegt dem Rastergitter und dem aktuellen graphischen Zustand.
- Wenn einige Punkte verschoben worden sind, können die dazwischenliegenden Punkte in dazu passender Weise nachträglich interpoliert werden.
- Wenn die Instruktionen abgearbeitet sind, kommt der klassische Rasterisierungs-Algorithmus zum Zuge, optional mit Erweiterungen, die einige Sonderfälle besser behandeln oder Anti-Aliasing unterstützen.



Nach einer Grafik von H. Schwarz
aus P. Karow: *Digitale Schriften*

- Zum graphischen Zustand gehört der Freiheits- und der Projektionsvektor.
- Distanzen werden immer nur relativ zum Projektionsvektor gemessen oder interpretiert. Die Distanzen haben daher auch ein Vorzeichen, das wahlweise berücksichtigt wird oder nicht.
- Punkteverschiebungen erfolgen immer nur in Richtung des Freiheitsvektors.
- Im Beispiel wird der Projektionsvektor orthogonal zu der Strecke $\overline{P_3P_4}$ gewählt, dann eine feste Distanz gewählt, dann P_3 entlang dem Freiheitsvektor $\overline{P_2P_3}$ verschoben, danach P_4 entlang von $\overline{P_4P_5}$.

Wenn Punkte entlang dem Freiheitsvektor verschoben werden, dann kann die tatsächlich gewählte Verschiebungsdistanz von den Rundungsmodi und dem Raster abhängig gemacht werden.
Konkret lassen sich bei den Rundungsmodi, die zum graphischen Zustand gehören, drei Parameter einstellen:

- Periode: bei 1 werden die Gitterpunkte angesprungen, bei 0.5 die Halbgitterpunkte, bei 2 nur jeder zweiter Gitterpunkt.
- Phase: bei 0 werden genau die Gitterpunkte genommen, bei größeren Werten werden die angesprungenen Punkte entsprechend zum Gitter verschoben.
- Schwelle: regelt, wohin gerundet wird: bei 1 immer zum kleineren Wert, bei 0.5 zum näher gelegenen Wert.

Die Instruktionen verteilen sich über mehrere Tabellen:

fpgm Globale Prozeduren und Instruktionen, die eingerichtet bzw. ausgeführt werden, sobald das erste Zeichen zu rastern ist.

prep Instruktionen, die jedes Mal erneut ausgeführt werden, wenn die Schriftgröße oder die Transformationsmatrix verändert wird.

cvt Initialisierte Tabelle mit globalen Werten (*control value table*).

glyf Bei den einzelnen Zeichen sind nicht nur die Kurven und die Metriken eines einzelnen Zeichens gegeben, sondern auch die individuellen Instruktionen.

In der *cmap*-Tabelle können verschiedene Kodierungstabellen enthalten sein, jeweils mit verschiedenen Kombinationen aus *platformID* und *platEncID*:

- *platformID* = 0: Unicode und dadurch platformunabhängig
- *platformID* = 1: MacOS mit diversen Kodierungen in Abhängigkeit von einer Sprache bzw. einem Schriftsystem
- *platformID* = 2: Microsoft Windows mit diversen Kodierungen in Abhängigkeit von einer Sprache bzw. einem Schriftsystem

Die Tabellen bilden jeweils Codes aus den jeweiligen Tabellen in Namen ab, die in der *glyf*-Tabelle gefunden werden.