METAFONT

Seminar im Grundstudium
Kurven und Flächen
Matthias Ott

20.Januar 1995

Inhaltsverzeichnis

1 Die Grundidee

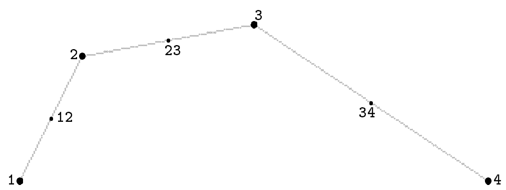
METAFONT wurde vom selben Author wie TEX also von Donald E. Knuth programmiert und entworfen. Es wurde in WEB (einem Pascal-Dialekt) geschrieben, was bedeutet, daß es genauso wie TeX leicht portierbar ist und deshalb universell verwendbar.
METAFONT basiert auf der Idee, die Zeichen eines Zeichensatzes ähnlich wie im wirklichen Leben zu zeichnen. Das bedeutet, daß die einzelnen Zeichen, die man mit METAFONT entwirft, mit einem vorher definierten Stift gezeichnet werden.

2 Der Zusammenhang mit 'Kurven und Flächen'

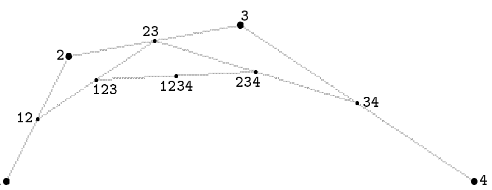
Der Linienalgorithmus basiert auf dem Bernstein-Algorithmus für Polynome 3.Ordnung:

$$z(t) = (1-t)^3 z_1 + 3(1-t)^2 t z_2 + 3(1-t)t^2 z_3 + t^3 z_4$$

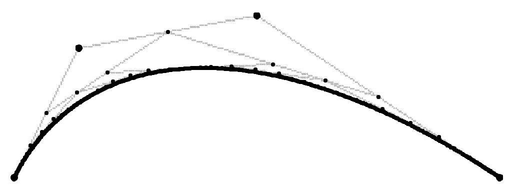
$\Rightarrow$ allgemeinste Eingabe: `draw z1..control z2 and z3..z4`
Man beginnt mit vier Punkten (z_1, z_2, z_3, z_4) und berechnet dann die 3 Mittelpunkte $z_{12} = \frac{1}{2}[z_1, z_2], z_{23} = \frac{1}{2}[z_3, z_3], z_{34} = \frac{1}{2}[z_3, z_4]$:



Diesen Punkte verbindet man, und fährt mit dem Algorithmus fort:



Der Prozess konvergiert dann gegen die gewünschte Kurve:



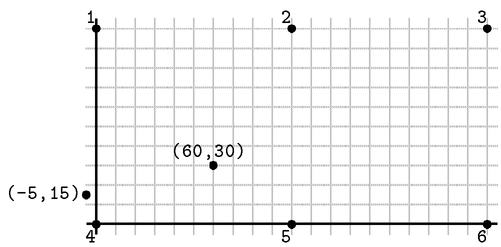
3 Das Programm

3.1 Allgemeines

METAFONT ist eine normale Programmiersprache, auf Befehlen basierend. Tatsächlich besteht sie aus wenigen Grundbefehlen, der Rest wird durch Makros gebildet.
Der mathematische Interpreter von METAFONT ist sehr leistungsfähig, er kann Gleichungssysteme mit mehreren Umbekannten lösen.
⇒ z.B. $y = 10$ stellt eigentlich keine Zuweisung dar, diese müßte explizit mit $y := 10$ geschehen.

3.2 Werteeingabe

Das Koordinatensystem ist auf Pixel ausgerichtet, den Ursprung links unten:



⇒ Bei doppelter Druckerauflösung müssen die Koordinaten verdoppelt werden.
Zahlenangaben brauchen keine ganze Zahlen sein, der kleinste Bruchteil ist $\frac{1}{65536} = 2^{-16} \equiv$ **epsilon** . Intern werden alle Zahlen auf ein Vielfaches von $\frac{1}{65536}$ gerundet. Die größte Variablenzahl ist $2^{12} - \frac{1}{65536} \equiv$ **infinity** . Bei der Positionierung der Punkte werden die Werte auf eine ganze Zahl gerundet. Für Zwischenergebnisse sind interne Werte $-2^{15} \leq x \leq 2^{15} - epsilon$ erlaubt.
Werte können in der Form $(x_i, y_i) = (a, b)$ angegeben werden. Ein mit **z** beginnender Variablenname steht als Abkürzung für (x_i, y_i).
⇒ $z_1 = (0, 0)$ ist erlaubt, $0.5[z_1, z_2] \equiv (0.5(x_1 + x_2), 0.5(y_1 + y_2))$
Weitere Längeneinheiten:

Maß	*Name*	*Umrechnung*	*Verbreitung*
pt	Punkt	72.72pt = 1in	Druckwesen USA und UK
pc	Pica	1pc = 12pt	dto.
bp	Big Point	72bp = 1in	dto.
dd	Didot Punkt	1157dd = 1238pt	Druckwesen Europa
cc	Cicero	1cc = 12dd	dto.
in	Inch	1in = 2.54cm	USA und UK
cm	Zentimeter	100cm = 1m	Europa
mm	Millimeter	100mm = 1cm	dto.

Aber: Auch diese Werte sind pixelbezogen. Für absolute Maßangaben unabhängig jeder Druckerauflösung ist den Maßeinheiten ein #-Zeichen anzuhängen. Dies gilt für allen obigen Längeneinheiten.

3.3 Punkte

Für die Definition von Punkten gibt es mehrere Voraussetzungen:

- Die Variablen x und y stehen für die beiden Koordinatenachsen;
- Ein Punkt kann in der Form $z = (x, y)$ definiert werden;
- Für verschiedene Punkte kann ein Vektor verwendet werden;
  Hierbei gilt:
  `x[1]` $\equiv$ `x.1` $\equiv$ `x1`
  Es kann eine linke und rechte Ecke angegeben werden:
  `x[1][r]` $\equiv$ `x.1.r` $\equiv$ `x1r`
  Ein mehrdimensionaler Vektor ist möglich:
  `x[1][1][r]` $\equiv$ `x.1.1.r` , aber: $\neq$ `x11r`

3.4 Bearbeitungsmodi

Es existieren standardgemäß vier verschiedene Bearbeitungsmodi:
`proof, smoke, lowres, localfont`
`mode=proof:` jeder Buchstabe stark vergrößert, eigene Seite, grau getönt, mit Eingabekontrollpunkten, mit Gitternetz
`mode=smoke:` keine Kontrollpunkte, schwarz, kein Gitter, 4 Ecksymbole
`mode=lowres:` 200 Pixel/Zoll
`mode=localfont:` Standardanpassung an den lokalen Drucker

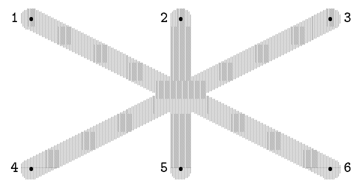
Die Größe kann auch wie folgt verändert werden: `mag=`*vergr-wert* : $< 1.0 \Rightarrow$ Verkleinerung
`mag=magstep` $n : x = 1.2^n$

Bei einem Programmdurchlauf entsteht ein gf-File (Generic Font) und ein tfm-File (TEX-Font-Metric). Mit **gftodvi** kann das gf-File in ein dvi-File und mit **gftopk** in ein von LATEX ladbares Font-File umgewandelt werden.

4 Zeichnen

4.1 Linien und Kurven

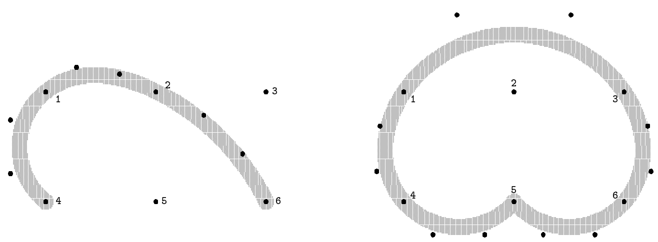
Befehle zum Zeichnen von Linien:
`draw` $z_1..z_2$ oder `draw` $z_1 - -z_2$
. . steht für das Verbinden zweier isolierter Punkte, bei - - können mehrere Punkte angegeben werden.



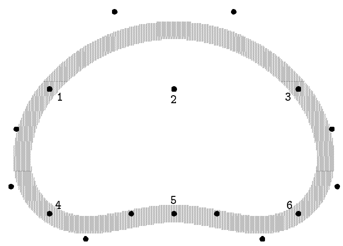
wurde erzeugt mit:

```
z1=(0,100);
z2=(100,100);
z3=(200,100);
z4=(0,0);
z5=(100,0);
z6=(200,0);
draw z1..z6;
draw z2..z5;
draw z3..z4;
```

Werden mit . . mehr als 2 Punkte verbunden, so entsteht eine Kurve.



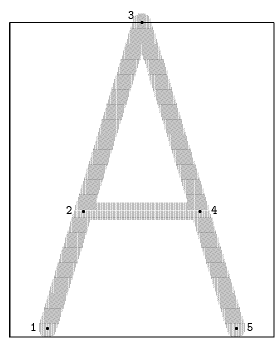
Geschlossene Kurven werden durch Anhängen von `cycle` gebildet.



Gefüllte Flächen:
Diese werden mit `fill` erzeugt, der Kurvenzug *muß* dann mit `cycle` enden. Hier ist die Beschaffenheit des Zeichenstiftes belanglos. Zum Zeichnen einer Fläche mit dem aktuellen Stift muß `filldraw` verwendet werden.
Punkte:
Mit `drawdot` z wird ein Punkt mit dem aktuellen Stift gezeichnet.
All diese Befehle können durch die Vorsilbe `un` zum Löschen verwenden werden.
Die Richtung der Kurve:
Das Anfügen von [dir α] an einen Punkt beinflußt die Richtung. Hier existieren spezielle Makros: `right, up, left, down`

4.2 Mathematischer und realer Punkt

Ein realer Punkt mit einen kreisförmigen Stift von 10pt Radius und der Koordinaten (x, y) ergibt einem gefüllten Kreis mit Radius 10pt und Mittelpunkt (x, y).Solche Punkte liegen deshalb manchmal außerhalb der Begrenzungen.
$\Rightarrow$ Die Befehle `top, bot, lft, rt` verhindern dies:




```
bot z1=(10,0); z5=z1+(80,0);
z3=(1/2[x1,x5],150);
z2=0.4[z1,z3]; z4=0.4[z5,z3];
```

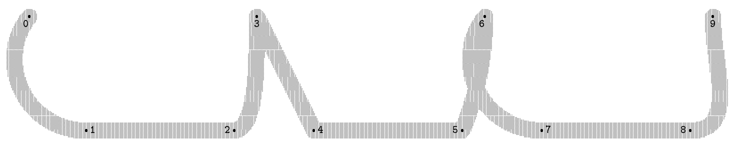
```
draw z1..z3; draw z3..z5; draw z2..z4;
```

4.3 Krümmung und Spannung

`curl` a: die Krümmung der Kurve an einem Punkt; großer Wert $\Leftrightarrow$ starke Krümmung;
`tension` a: Spannung der Kurve zwischen 2 Punkten; $> 1 \Leftrightarrow$ stärkere Spannung
`tension` a `and` b: ermöglicht die Eingabe von verschiedenen Spannungswerten an den Enden
`..` steht für `..tension 1 and 1..`, und `..tension` a`..` für `..tension` a `and` a`..`

4.4 Allgemeine Makros-Definitionen

`def -- = {curl 1}..{curl 1} enddef;` : Verbinden zweier Punkte immer mit einer geraden Linie, da Krümmung an beiden Punkten gleich Null
`def --- = .. tension infinity .. enddef;`: 'gespannte' Linie, fast das gleiche wie `--`, die Linie ist nur nicht ganz gerade, sondern so gebogen, daß sie an den Zwischenpunkten 'rund' übergeht.
`def ... = .. tension atleast 1 .. enddef;`: 'gebundene' Kurve mit einer Spannung zwischen zwei Punkten von mindestens 1
`&` Verbinden von 2 unabhängigen Pfaden mit gemeinsamen Endpunkt, keine Beeinflussung gegenseitig;
Beispiel der verschiedenen Möglichkeiten:



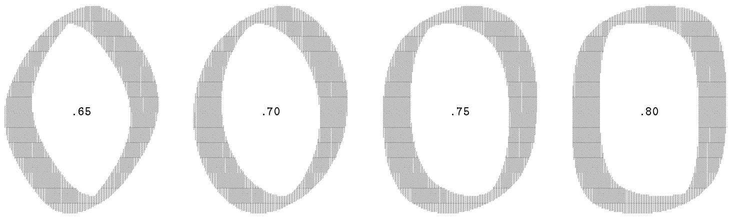
wurde erzeugt durch:

```
z0=(0,100); z1=(50,0); z2=180,0);
for n = 3 upto 9: z[n]=z[n-3]+(200,0); endfor
draw z0..z1---z2...{up}z3
& z3..z4--z5...{up}z6
& z6...z7---z8..{up}z9;
```

4.5 Spezielle Zeichenbefehle

Vordefinierte Makros für Kreise, Quadrate und Ellipsen:
`[`*quarter, half, full*`]circle [rotated` *winkel*`][scaled` *maß*`][shifted` (x, y)`]`

`superellipse` (*rechts*, *oben*, *links*, *unten*, *wert*)
zum Zeichnen einer Ellipse, die in Abhängigkeit von *wert* von einer echten Ellipse abweicht:



4.6 Der Zeichenstift

Eine Linie wird mit einem Stift einer bestimmten Dicke gezogen. Dieser Stift kann in METAFONT sehr frei bestimmt und abgeändert werden. Es existieren verschiedene Standardstifte:

`pencircle [xscaled` *a*, `yscaled` *b*`][rotated` *c*`]`

ergibt einen kreisförmigen Stift, der skaliert und rotiert werden kann. Die Standardgröße beträgt .2pt.

`pensquare [xscaled` *a*, `yscaled` *b*`][rotated` *c*`]`

ergibt einen viereckigen Stift.

`penrazor [scaled` *a*`][rotated` *b*`]`

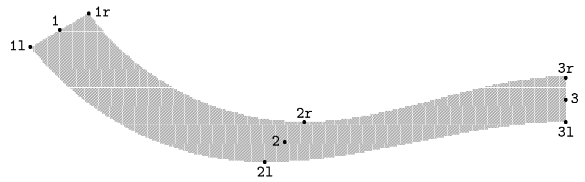
ist ein linienartiger Stift mit der Ausdehnung von einem Pixel in y-Richtung.
Weitere Zeichenstifte können mit `makepen` individuell entworfen werden.

Benutzung dieser Stifte:
Das Auswählen erfolgt mit `pickup` *pen*, der Stift kann mit *pen-name* `:= savepen` abgespeichert werden. Das Laden dieses Stiftes erfolgt mit `pickup` *pen-name*.

4.7 Flächen und Pseudozeichenstifte

Statt mit einem tatsächlichen Stift zu arbeiten, kann die Abmessung und die Neigung in bestimmten Punkten angegeben werden. METAFONT zeichnet dann eine gefüllte Fläche durch diese Punkte:
`penpos`α (*grösse*, *winkel*)
definiert die Dicke und die Neigung eines fiktiven Stiftes im Punkt α,
`penstroke` z_{1e} `..` z_{2e},

verbindet die einzelnen Punkte:




```
penpos1(1.2pt,30); penpos2(1.0pt,45); penpos3(0.8pt,90);
z1=(0,2pt); z2=(4pt,0); x3=9pt; y3l=y2r;
penstroke z1e..z2e{right}..z3e{right};
```

5 Die Programmierung

5.1 Variablen

Die verschiedenen Variablentypen von METAFONT:
`boolean`
`string`
`path`
`pen`
`picture`
`pair`
`transform`
`numeric`

Manchmal ist es notwendig, Variablen vom Typ *numeric* neu zu initialisieren, z.B. innerhalb des gleichen Programmes bei voneinander unabhängigen Relationen.

5.2 Schleifen und Verzweigungen

`for` a=v_1 `upto` v_3: *Schleifenrumpf;* `endfor`
oder
`for` a=v_1 `step` v_2 `until` v_3: *Schleifenrumpf;* `endfor`

stellt eine Schleife von a mit Anfangswert v_1 bis v_3 und optional mit Schrittgrösse v_2 dar.
Konditionalanweisungen:

`if` a_1: a_2

`elseif` b_1`:` b_2 ....
`else` c
`fi`

Es können beliebig viele `elseif` verwendet werden.

Die logische Verknüpfung mehrerer Ausdrücke erfolgt mit `and, or, not`.
Weitere mathematische Befehle:
`max(`a_1`,`a_2`,...)`
`min(`a_1`,`a_2`,...)`
`floor` *zahl*
`ceiling` *zahl*
`round` *zahl*
m `mod` n
`abs` *zahl*
`length(`x, y`)`
Mit dem `++` Operator können Ausdrücke wie $\sqrt{a^2+b^2}$ als `a++b` eingeben werden, analog `a--b` für $\sqrt{a^2-b^2}$

6 Entwerfen eines Zeichensatzes

6.1 Grundvoraussetzungen

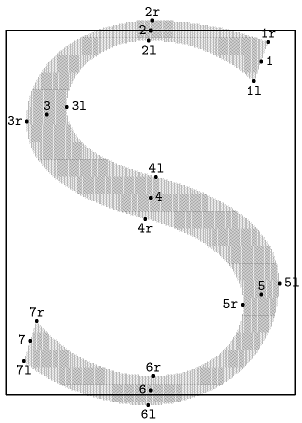
Durch den Befehl
`beginchar (` *symbol, breite#, höhe#, tiefe#*`);`
`endchar;`
wird einer Zeichenkonstruktion ein Platz im Zeichensatz und eine umgebende Box zugeordnet. LaTeX verwendet beim Setzen der Zeichen diese umgebende Box, die mit *breite#, höhe#, tiefe#* definiert wird.
symbol ist die Platzierung des Zeichens im Zeichensatz.

6.2 Ligaturen und Italic-Korrekturen

Der Abstand zwischen zwei Zeichen ist durch die umgebenden Boxen festgelegt. Bei manchen Zeichenkombinationen soll von diesem Wert abgewichen werden. z.B. 'VA' statt 'VA' ⇒
`ligtable "`z_1`" : "`z_2`": ...: "`s_1`" kern` m_1`, "`s_2`" kern` m_2`, ...;`
Dies bedeutet: Folgt auf die Zeichen z_1,z_2, ... als nächstes Zeichen s_1, so wird ein weiterer Zwischenraum der Größe m_1 angefügt, usw.. Dieser Befehl muß außerhalb des Zeichenmakros stehen.
Andere Zeichenkombinationen werden durch das Ersetzen mit einer eigenen Ligatur realisiert. Dies erfolgt bei fi, ff oder fl z.B. durch:
`ligtable "f": "i" =: oct"014", "f" =: oct"013", "l" =: oct"015";`
Zusätzlich muß beim Umschalten von geneigter auf gerader Schrift beachten

werden, daß der Abstand hier meist zu klein wird. Dies kann durch `italcorr` *maß#* korrigiert werden.

6.3 Weiterführendes Beispiel




```
mode_setup;
clear_all;
em#:=10pt#; cap#:=7pt#;
slab#:=.8pt#;
stem#:=.9pt#;
thin#:=.8pt#;
thick#:=5/6pt#;
o#:=1/5pt#;
define_corrected_pixels(o);
define_pixels(em,cap);
define_blacker_pixels(thin,thick,slab,stem);
beginchar(25,5/9em#,cap#,0); % ´´The letter S´´;
penpos1(slab,70);
penpos2(.5slab,80);
penpos3(.5[slab,thick],200);
penpos5(.5[slab,thick],210);
penpos6(.7slab,80);
penpos7(.25[slab,thick],72);
x1=x5; y1r=.94h+o;
x2=x4=x6=.5w; y2r=h+o; y4=.54h; y6l=-o;
x3r=.04em; y3=.5[y4,y2];
x5l=w-.03em; y5=.5[y4,y6];
.5[x7l,x7]=.04em; y7l=.12h-o;
path trial; trial=z3{down}..z4..{down}z5;
pair dz; dz=direction 1 of trial;
```

```
penpos4(thick,angle dz-90);
penstroke z1e..z2e{left}..z3e{down}..z4e{dz}..z5e{down}..z6e{left}..z7e;
labels.lft(2,3r,5r,6,7);
labels.rt(1,3l,4,5l);
labels.top(1r,2r,3,4l,5,6r,7r);
labels.bot(1l,2l,4r,6l,7l);
endchar;
```

Literatur

[1] Donald E. Knuth: The Metafontbook, 1991

[2] H. Kopka: Latex Ergänzungen, 1995

[3] G. Müllejans: Metafont, 1992