Bilder in LaTeX-Dokumenten
PicIns-Benutzerhandbuch Version 3.0

Joachim Bleser   Edmund Lang

TH Darmstadt, Hochschulrechenzentrum

September 1992

Inhaltsverzeichnis

1 Einleitung

Die meisten modernen Textverarbeitungssysteme und Desktop Publishing Systeme bieten die Möglichkeit, Bilder in Dokumente zu integrieren. Die Bilder können beliebig placiert und auf verschiedene Arten umrahmt werden. Als besonders elegant wird das *Umfließen* von Bildern durch die Textzeilen empfunden, wodurch eine enge Bindung der Bilder mit den dazu gehörenden Textpassagen erreicht wird.

TEX und LATEX bieten standardmäßig keine Funktionen dieser Art an. Mit Hilfe des im folgenden beschriebenen Makro-Paketes `PicIns`[1] ist die Integration von Bildern auch in LATEX-Dokumenten in komfortabler Weise möglich.

Wir verstehen dabei unter einem *Bild* eine rechteckig begrenzte Fläche, die aus einem (möglicherweise nicht vorhandenen) Rahmen und einem (möglicherweise leeren) Inhalt besteht.

Der Inhalt eines Bildes kann mit Hilfe von LATEX-Funktionen (z.B. normalem Text, mathematischen Formeln, Tabellen oder Konstrukten, die mit der `picture`-Umgebung erzeugt wurden) oder durch Einbindung extern erzeugter Grafiken definiert werden. Die Problematik der Integration externer Grafiken wird im Abschnitt 5 behandelt.

Die LATEX-Style-Datei `picins.sty` enthält *keine* Funktionen zur Erstellung des Bildinhaltes. Sie stellt lediglich die folgenden LATEX-Kommandos bereit, die zur Placierung der Bilder, zum Freihalten des von den Bildern in Anspruch genommenen Platzes, zur Umrahmung und zum Umfließen der Bilder mit dem Text erforderlich sind:

`\parpic` placiert ein Bild an den Anfang eines Absatzes.

`\hpic` placiert Bilder in einem eigenen Absatz nebeneinander.

`\picskip` beeinflußt das *Umfließen* von Bildern durch Textzeilen.

`\pichskip` steuert den horizontalen Abstand zwischen Bild und Text.

`\shadowthickness` bestimmt die Dicke des Schattens beim Umrahmen von Bildern.

`\dashlength` bestimmt die Länge der Striche bei der gestrichelten Umrahmung.

`\boxlength` bestimmt die Kastentiefe bei der Umrahmung mit einem Kasten.

`\piccaption` erlaubt das Setzen einer Unterschrift.

`\newcaption` formatiert die Unterschrift etwas anders als das original LATEX `\caption`-Kommando.

`\picchangemode` steuert die Bildplacierung bei doppelseitigen Dokumenten.

Weiterhin werden vier neue Umgebungen zur Verfügung gestellt. Sie werden wie gewohnt durch `\begin{`*Umgebung*`}` ...`\end{`*Umgebung*`}` eingeleitet bzw. beendet.

`frameenv` ist eine Umgebung, deren Inhalt umrahmt wird.

`dashenv` ist eine Umgebung, deren Inhalt gestrichelt umrahmt wird.

`ovalenv` ist eine Umgebung, deren Inhalt mit abgerundeten Ecken umrahmt wird.

`shadowenv` ist eine Umgebung, deren Inhalt schattiert umrahmt wird.

[1] Die vorliegende Publikation ist urheberrechtlich geschützt. Inhaltliche Änderungen bedürfen der schriftlichen Genehmigung der Autoren.

2 Bilder am Anfang eines Absatzes: `\parpic`

Das `\parpic`-Kommando placiert ein Bild wahlweise links oder rechts an den *Anfang* eines Absatzes. Der Text, der auch mehrere Absätze umfassen kann, umfließt das Bild.

Syntax : `\parpic(`*breite,höhe*`)(`*x-offset,y-offset*`)[`*optionen*`][`*position*`]{`*Bildinhalt*`}`

Alle Parameter bis auf *Bildinhalt* sind optional.
Beschreibung der Parameter:

breite, höhe: *breite* bestimmt die Breite des Bildes. Die Textzeilen, die das Bild umfließen, werden um den Betrag von *breite* (zuzüglich einem definierten Abstand zwischen Bild und Text) verkürzt. Dadurch bleibt die Gesamtbreite des Absatzes erhalten. Aus der *höhe* des Bildes berechnet sich die Anzahl der Textzeilen, die zum Umfließen des Bildes benötigt werden.

Fehlen diese Angaben, werden für *breite* und *höhe* die Ausdehnungen des kleinsten Rechteckes gewählt, das den Bildinhalt vollständig umschließt (*bounding box*). Dies setzt voraus, daß der Bildinhalt eine echte Höhe und Breite besitzt, was nicht bei allen Methoden zur Einbindung extern erstellter Bilder der Fall ist (s.a. Abschnitt 5.6).

x-offset, y-offset: Durch Angabe dieses Wertepaares ist es möglich, den Bildinhalt innerhalb des Rahmens beliebig in alle Richtungen zu verschieben. Der Bezugspunkt (*Ursprung, Referenzpunkt*) befindet sich in der linken oberen Ecke. Ein positiver *x-offset* verschiebt das Bild nach rechts, ein positiver *y-offset* verschiebt es nach unten. Negative Werte sind möglich (s.a. Abschnitt 5.6).

Fehlen diese Angaben, wird der *Bildinhalt* entsprechend des *position*-Parameters positioniert. Die *offset*-Parameter sind besonders bei der Integration extern erstellter Bilder wichtig (s.a. Abschnitt 5.6).

optionen: Mit Hilfe der *optionen* kann die Position des Bildes in Bezug auf den Absatz und die Umrahmungsart spezifiziert werden:

`l` (*left*)	Das Bild wird auf die linke Seite des Absatzes placiert.
`r` (*right*)	Das Bild wird auf die rechte Seite des Absatzes placiert.
`f` (*frame*)	Das Bild wird mit einer durchgezogenen Linie umrahmt.
`d` (*dash*)	Das Bild wird mit einer gestrichelten Linie umrahmt.
`o` (*oval*)	Die Ecken der Umrahmung sind abgerundet.
`s` (*shadow*)	Das Bild wird mit einem schattierten Rahmen umrahmt.
`x` (*box*)	Das Bild wird mit einem drei-dimensionalen Kasten umrahmt.

Kombinationen aus jeweils einem Positionsparameter und einem Umrahmungsparameter sind erlaubt (z.B. `lf, dr, ro,` ...). Kombinationen aus mehreren Positionsparametern (z.B. `lr, rl`) oder mehreren Umrahmungsparametern (z.B. `os, do,` ...) führen zu Fehlern.

Werden keine *optionen* angegeben, wird das Bild auf die linke Seite des Absatzes placiert und nicht umrahmt.

position : Mit Hilfe des *position*-Parameters kann die Position des *Bildinhalts* innerhalb der Rahmens bestimmt werden. Möglich sind:

`l` (*left*)	Der Bildinhalt wird am linken Rand des Rahmens ausgerichtet.
`r` (*right*)	Der Bildinhalt wird am rechten Rand des Rahmens ausgerichtet.
`t` (*top*)	Der Bildinhalt wird am oberen Rand des Rahmens ausgerichtet.
`b` (*bottom*)	Der Bildinhalt wird am unteren Rand des Rahmens ausgerichtet.

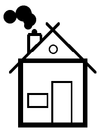
Wenn keine Angabe zur horizontalen Positionierung (`l, r`) vorliegt, wird der Bildinhalt horizontal zentriert. Analog dazu wird der Bildinhalt bei fehlenden Angaben zur vertikalen Ausrichtung (`b, t`) vertikal zentriert. Einander nicht widersprechende Positionen dürfen miteinander verknüpft werden (`lt, lb, rt, rb`). Wenn der *offset*-Parameter und der *position*-Parameter gleichzeitig angegeben werden, hat der *position*-Parameter keine Gültigkeit. Wenn weder *offset-* noch *position*-Parameter angegeben werden, wird der Bildinhalt horizontal und vertikal innerhalb des Rahmens zentriert. Wenn der *position*-Parameter angegeben wird, muß auch der *optionen*-Parameter vorhanden sein (evtl. in der Form `[]`, also z.B. `\parpic(3cm,2cm)[][tr]{`*Bildinhalt*`}`).

Die Positionierung von extern erzeugten Bildern hängt von der gewählten Methode zur Einbindung dieser Bilder ab (s.a. Abschnitt 5).

Bildinhalt : Zuletzt folgt der eigentliche Bildinhalt, d.h. das, was in den freien Platz eingesetzt werden soll. Der Inhalt darf ein beliebiges TEX- oder LATEX-Konstrukt sein, z.B. eine mit der `picture`-Umgebung erstellte Zeichnung, oder auch ein extern erstelltes Bild. Beispiele folgen weiter unten.

2.1 Die Optionen von `\parpic`

Die meisten der folgenden Beispiele sind umrahmt; dies ist nicht unbedingt erforderlich, verdeutlicht aber die Funktionsweise von `\parpic`. Die Beispiele verwenden Text mit halber Zeilenbreite, um in der rechten Hälfte der Seite die LATEX-Kommandos zur Erzeugung der Beispiele anzudeuten.
Das Kommando `\Karten` steht für `$\clubsuit\diamondsuit\heartsuit\spadesuit$`.
Die Box `\hausboxI` enthält das kleine Häuschen, das mittels der `picture`-Umgebung erstellt wurde. Die Kommandosequenz `\copy\hausboxI` kopiert die Box `\hausboxI` an die entsprechende Stelle *ohne* ihren Inhalt zu löschen (`\box\hausboxI` kopiert die Box und löscht den Inhalt). `\hausbreiteI` und `\haushoeheI` sind Dimensionsangaben und enthalten Breite und Höhe der Box `\hausboxI`. Die Initialisierung der Dimensionsangaben geschieht z.B. durch `\hausbreiteI=\wd\hausboxI`, wodurch `\hausbreiteI` die Breite der Box zugeordnet wird.



Die einfachste Form von `\parpic`. Alle Parameter bis auf *Bildinhalt* wurden weggelassen. Das bedeutet, daß Breite und Höhe des Bildes automatisch berechnet werden.

```
\parpic{\copy\hausboxI}
Die einfachste Form von \parpic.  ...
```

$\clubsuit\diamondsuit\heartsuit\spadesuit$

Die Breite dieses "Bildes" beträgt 3 cm, die Höhe beträgt 1 cm. Da keine *offsets* und auch keine *positionen* angegeben wurden, wird der Bildinhalt automatisch zentriert.

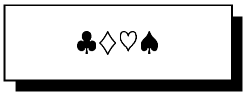
```
\parpic(3cm,1cm)[f]{\Karten} Die Breite
des ''Bildes'' beträgt 3 cm, die Höhe
beträgt 1 cm ...
```

$\clubsuit\diamondsuit\heartsuit\spadesuit$

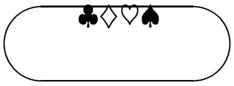
In diesem Beispiel wurde der
Bildinhalt mit Hilfe von *offsets* horizontal und vertikal um 5mm in Bezug auf die linke obere Ecke des Rahmens (*Bezugspunkt*) verschoben. Die Angabe von Offsets ist immer dann sinnvoll, wenn die automatische Positionierung zu unbefriedigenden Ergebnissen führt.

```
\parpic(3cm,1cm)(5mm,5mm)[f]{\Karten}
In diesem Beispiel wurde der Bildinhalt
mit Hilfe von ...
```

Das Bild[2] kann auf Wunsch auch auf die rechte Seite eines Absatzes placiert werden. Dazu ist die Option [r] vorgesehen. Durch die zusätzliche Option [s] wurde dieses Bild schattiert. Beide Optionen werden zusammengezogen.

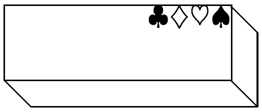



```
\parpic(3cm,1cm)[sr]{\Karten}
Das Bild kann auf Wunsch auch auf die
rechte Seite eines Absatzes placiert
werden.  ...
```



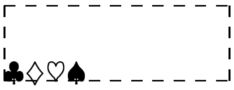
Die Option [o] umrahmt das Bild mit abgerundeten Ecken. Die Position [t] verschiebt den Bildinhalt an den oberen Rand der Umrahmung.

```
\parpic(3cm,1cm)[o][t]{\Karten} Die
Option {\tt[o]} umrahmt das Bild mit
abgerundeten Ecken.  ...
```



Die Option [x] umrahmt das Bild mit einem 3D-Kasten. Die Position [tr] verschiebt den Bildinhalt an die rechte obere Ecke der Umrahmung.

```
\parpic(3cm,1cm)[x][tr]{\Karten} Die
Option {\tt[x]} umrahmt das Bild mit
einem 3D-Kasten.  ...
```



Die Option [d] umrahmt das Bild mit gestrichelten Kanten. Die Position [lb] verschiebt den Bildinhalt an die linke untere Ecke.

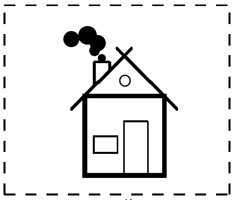
```
\parpic(3cm,1cm)[d][lb]{\Karten} Die
Option {\tt[d]} umrahmt das Bild mit
gestrichelten Kanten.  ...
```

Die Positionierung des Bildinhalts innerhalb des Rahmens erfolgt im Normalfall sehr exakt. Falls dennoch Probleme auftreten, so sind diese meistens dadurch begründet, daß das Konstrukt, das den Bildinhalt darstellt, keine echte Höhe oder Breite aufweist, was häufig bei der Integration extern erstellter Bilder vorkommt (s.a. Abschnitt 5). Probleme treten ebenfalls auf, wenn bei der Definition des Bildinhalts an den Rändern freier Platz erzeugt wird. Dies tritt häufig bei der `picture`-Umgebung auf und führt ebenfalls dazu, daß der sichtbare Teil des Bildinhalts falsch positioniert wird. In solchen Fällen sollte das Bild mit dem *offset*-Parameter in die gewünscht Position verschoben werden.

2.2 Text neben dem Bild

Die Anzahl der Zeilen, die eingerückt neben dem Bild stehen sollen, kann durch das `\picskip`-Kommando beeinflußt werden. Standardmäßig werden so viele Zeilen eingerückt, bis das Bild vollständig umflossen ist. Soll von dieser Vorgabe abgewichen werden, weil z.B. nur ein Absatz neben das Bild gesetzt werden und der nächste Absatz wieder unterhalb des Bildes erscheinen soll, so ist `\picskip`{n} zu verwenden. Das `\picskip`-Kommando beendet den laufenden Absatz.

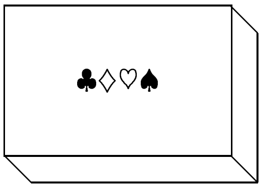
[2] Der schlechte Zeilenumbruch in diesem und anderen Beispielen ist ein Fehler der LaTeX `minipage` und nicht der `PicIns`-Makros



Der nächste Absatz soll nicht mehr eingerückt werden.

```
\parpic(3cm,2.5cm)[d]{\copy\hausboxI}
Der nächste Absatz ...
\picskip{0}
...
```

$n = 0$ (`\picskip{0}`) bedeutet, daß keine weiteren Zeilen mehr eingerückt werden. Der nächste Absatz beginnt unterhalb des Bildes.

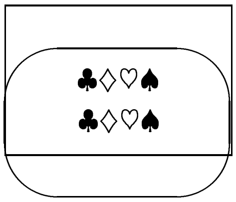


Wird `picskip` ein Wert $n > 0$ übergeben, so werden noch n Zeilen des *nächsten Absatzes* eingerückt. Dies kann in Einzelfällen notwendig sein, in denen die Berechnung der Anzahl einzurückender Zeilen nicht korrekt ist (s. a. Abschnitt "*Probleme*", Seite 19). In einem solchen Fall kann der Text den unteren Teil des Bildes überschreiben, oder es werden mehr Zeilen eingerückt als erforderlich.

```
\parpic(3cm,2cm)[x]{\Karten} Wird {\tt
picskip} ein Wert $n > 0$ übergeben, so
werden noch $n$ Zeilen des {\em
nächsten Absatzes} eingerückt.
\picskip{4}
Dies kann in Einzelfällen notwendig
sein, ...
```

Nach einem `\picskip`-Kommando mit einem Wert $n > 0$ darf *kein* `\par`-Kommando und auch *keine* Leerzeile folgen[3]. Nach einem `\picskip`-Kommando mit $n = 0$ ist ein `\par`-Kommando redundant.

Der Befehl `\picskip{0}` ist insbesondere dann notwendig, wenn zwei `\parpic`-Kommandos mit nur wenig Text aufeinander folgen. Dann tritt folgender unerwünschter Effekt auf:



Absatz 1: Dieser Absatz umfliesst das Bild nicht vollständig.

Absatz 2: Dadurch rutschen beide Bilder ineinander.

Dieser Fehler kann vermieden werden, indem nach dem *Absatz 1* wie oben beschrieben der Befehl `\picskip{0}` ergänzt wird.

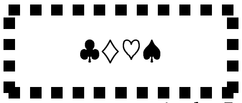
2.3 Linienstärke, Schattendicke, Seitenlänge und Strichlänge

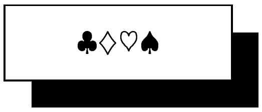
Das LaTeX-Kommando `\linethickness` erlaubt die Variation der Rahmenstärke des `\parpic`-Kommandos mit den Optionen `f,d` oder `s`. Die Änderung gilt solange, bis sie wieder zurückgesetzt wird. Die voreingestellte Rahmenstärke beträgt 0.4pt.

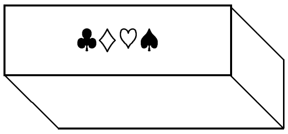
Die Strichlänge für gestrichelte Rahmen kann durch das in `picins.sty` definierte Kommando `\dashlength` verändert werden (z.B. `\dashlength{2pt}`). Voreingestellt ist eine Länge von 4pt.

[3]Die Einrückung der Zeilen wird mit den TeX-Primitiva `\hangindent` und `\hangafter` erreicht. Ein `\par`-Kommando setzt `\hangindent` auf `0pt` zurück, sodaß keine weiteren Zeilen mehr eingerückt würden.

Die drei folgenden Beispiele wurden innnerhalb einer `enumerate` Umgebung erstellt und sind deshalb durchnummeriert. Das `\parpic`-Kommando und das `\hpic`-Kommando lassen sich innerhalb von (auch tiefer verschachtelten) Listen verwenden. Nur von der Verwendung innerhalb einer `description`-Liste ist abzuraten, da hierbei der Itemtext das Bild oder den eigentlichen Text überschreiben kann.

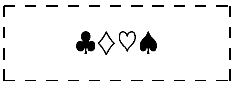
1. 

Die Kombination von `dashlength` und `linethickness` erlaubt weitere Gestaltungsmöglichkeiten. Es sollte dann aber unbedingt darauf geachtet werden, daß Bildhöhe und Bildbreite ein ganzzahliges Vielfaches von `dashlength` sind. In diesem Beispiel wurden `\dashlength{4pt}` und `\linethickness{4pt}` gewählt. Diese Werte sind auch voreingestellt. Die Strichlänge kann selbstverständlich auch unabhänig von der Linienstärke verändert werden.
2. 

Bei schattierten Rahmen kann die *Schattendicke* frei gewählt werden. Das in `picins.sty` definierte Kommando `\shadowthickness` arbeitet analog zum LaTeX-Kommando `\linethickness`. In diesem Beispiel wurde `\shadowthickness{10pt}` gewählt. Voreingestellt ist eine Schattendicke von 4pt.
3. 

Die Seitenlänge eines Kastenrahmens, d.h. die Tiefe des Kastens, kann mit dem Kommando `\boxlength` verändert werden. Dabei ist aber zu beachten, daß LaTeX geneigte Linien nicht in beliebiger Länge darstellen kann. Bei bestimmten Tiefen-Angaben werden daher die schrägen Seitenlinien nicht dargestellt. In diesem Beispiel wurde `\boxlength{20pt}` gewählt. Voreingestellt ist eine Tiefe von 10pt.

2.4 Abstand zwischen Bild und Text



Der horizontale Abstand zwischen Bild und Text kann durch `\pichskip{`*dim*`}` variiert werden (das *h* in `\pichskip` steht für *horizontal*). Voreingestellt ist ein Abstand von $1em$. In diesem Beispiel beträgt er $3em$.

```
\pichskip{3em}
\parpic(3cm,1cm)[d]{\Karten} Der
horizontale Abstand zwischen ...
```

2.5 Unterschriften

Wie bereits oben erwähnt, muß der *Bildinhalt* nicht unbedingt eine Zeichnung sein. Als Beispiel mag eine mathematische Formel dienen :

$$V = \left| \begin{array}{lcl} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \end{array} \right|$$

Abbildung 1: Das Volumen $V = (abc)$ des von den Vektoren a, b, c aufgespannten Parallelepipedes berechnet sich nach der links stehenden Formel.

```
\piccaptionside
\piccaption{Das Volumen $V = (abc)$...}
\setbox\tmpbox=\hbox{$ V = \left|
\begin{array}{lcl}
a_x & a_y & a_z\\
b_x & b_y & b_z\\
c_x & c_y & c_z
\end{array} \right| $}
\parpic{\box\tmpbox}
```

Wie in diesem Beispiel zu sehen ist, können Bilder mit Unterschriften versehen werden. Die Position, an die die Unterschrift gesetzt wird, kann durch 4 Befehle gesteuert werden:

1. `piccaptionoutside` positioniert *unterhalb* des Bildes und *außerhalb* des Rahmens (dies ist zugleich die Voreinstellung).
2. `piccaptioninside` positioniert *unterhalb* des Bildes, aber *innerhalb* des Rahmens.
3. `piccaptionside` positioniert *neben* das Bild, vertikal in Bezug zur Bildhöhe zentriert.
4. `piccaptiontopside` positioniert *neben* das Bild, aber an der Bildoberkante ausgerichtet.

Achtung: Die Unterschrift muß *vor* dem \parpic-Kommando definiert werden. Sie erscheint trotzdem unter (bzw. neben) dem Bild.

Durch das `\piccaption`-Kommando werden die Zähler für Abbildungen und Tabellen in derselben Weise erhöht, wie durch das original LaTeX-Kommando `\caption` innerhalb von `figure`- oder `table`-Umgebungen.

♣♢♡♠

Abbildung 2: Kartensymbole

Am gebräuchlichsten dürfte wohl der erste Fall sein, also die Unterschrift unter dem Bild und außerhalb des Rahmens. Wurde keine Rahmenoption gewählt, unterscheiden sich der erste und der zweite Fall nicht.

```
\piccaptionoutside
\piccaption{Kartensymbole}
\parpic(3cm,1cm)[f]{\Karten} Am
gebräuchlichsten dürfte wohl der erste
Fall sein, ...
```

2.6 Doppelseitige Dokumente

Mit den `\parpic`-Optionen `[l]` oder `[r]` werden Bilder links oder rechts an den Absatzanfang gesetzt, unabhängig davon, ob die aktuelle Seite eine linke oder eine rechte Seite ist. Mit Hilfe des `\picchangemode`-Modus, der in erster Linie für `twoside` Dokumente gedacht ist, sich aber auch in allen anderen Stil-Optionen verwenden läßt, kann man die Bilder in Abhängigkeit von "linke oder rechte Seite" positionieren. Das Kommando `\picchangemode` beginnt und das Kommando `\nopicchangemode` beendet diesen speziellen Modus. Die Wirkungsweise von `\picchangemode` ist die folgende:
Auf *ungeraden* Seiten (dies sind im allgemeinen *rechte* Seiten) wird das Bild *entsprechend* der Vorgabe positioniert (also `[l]` setzt das Bild auf die *linke* Seite des Absatzes).
Auf *geraden* Seiten (dies sind im allgemeinen *linke* Seiten) wird das Bild *entgegengesetzt* der Vorgabe positioniert (also `[l]` setzt das Bild auf die *rechte* Seite des Absatzes).

3 Bilder zwischen Absätzen: `\hpic`

Das `\parpic`-Kommando wird benutzt, um Bilder und Text zu mischen. Für Bilder, die ohne Text zwischen den Absätzen stehen sollen, stellt `PicIns` das `\hpic`-Kommando zur Verfügung. Im Gegensatz zu `\parpic` lassen sich mit `\hpic` auch mehrere Bilder nebeneinander setzen.

Syntax: `\hpic`(*breite,höhe*)(*x-offset,y-offset*)[*optionen*][*position*]{*Bildinhalt*}

Alle Parameter bis auf *Bildinhalt* sind optional.
Beschreibung der Parameter:

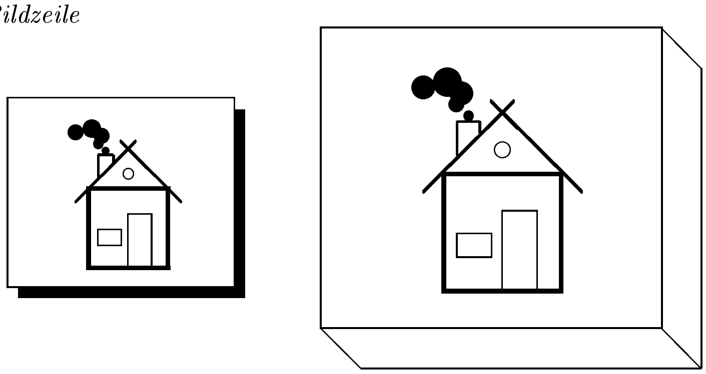
breite, höhe, x-offset, y-offset, position und *Bildinhalt* bedeuten dasselbe wie unter `\parpic` auf Seite 3 beschrieben.

An *optionen* stehen zu Verfügung :

`t` (*top*)	Aufeinanderfolgende Bilder werden an der Oberkante bündig ausgerichtet.
`b` (*bottom*)	Aufeinanderfolgende Bilder werden an der Unterkante bündig ausgerichtet.
`f` (*frame*)	Das Bild wird umrahmt.
`d` (*dash*)	Die Umrahmung des Bildes ist gestrichelt statt durchgezogen.
`o` (*oval*)	Die Ecken der Umrahmung sind abgerundet.
`s` (*shadow*)	Das Bild wird umrahmt und schattiert.
`x` (*box*)	Das Bild wird als Box umrahmt.

Ohne Positionsparameter `t` und `b` werden mehrere aufeinanderfolgende Bilder vertikal zentriert, ohne Umrahmungsparameter werden Bilder nicht umrahmt.

Die *Bildzeile*



wurde folgendermaßen erzeugt:

```
\centerline{%
\hpic(3cm,2.5cm)[s]{\copy\hausboxI}%
\hfil%
\boxlength{15pt}%
\hpic(4.5cm,4cm)[x]{\copy\hausboxII}%
\hfil%
\shadowthickness{4pt}%
\hpic(3cm,2.5cm)[s]{\copy\hausboxI}%
}
```

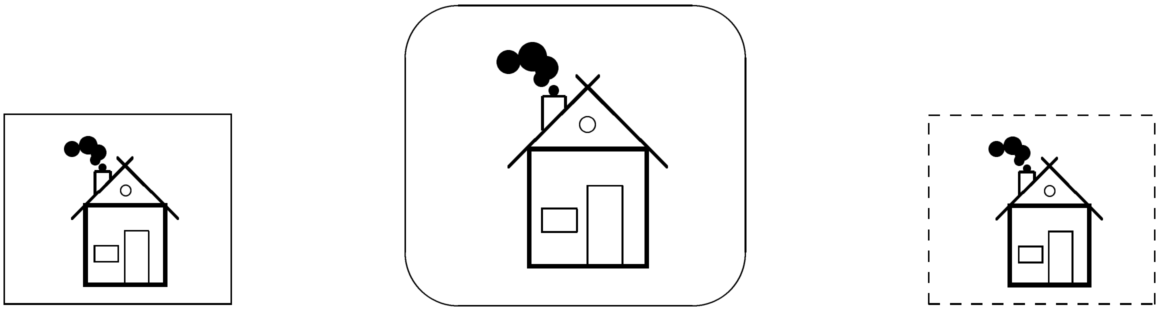


Abbildung 3: Wirkungsweise der Optionen `[fb]`, `[ob]` und `[db]`, zugleich Demonstration des `\newcaption` Kommandos, das den Text der Unterschrift *linksbündig* formatiert.

Diese *Bildzeile* entstand analog zur obigen, doch wurden als Optionen `[fb]`, `[ob]` und `[db]` verwendet. Dadurch erscheinen die Bilder umrahmt, gestrichelt und oval umrahmt und an der Unterkante bündig ausgerichtet. Die Ausrichtung am linken bzw. rechten Seitenrand wurde durch die Kommandos `\hfill` (anstelle von `\hfil`) zwischen den einzelnen Bildern erreicht.

Innerhalb einer *Bildzeile* sollten die Positionsparameter `t` und `b` nicht gemischt verwendet werden, da sonst unvorhersagbare Ergebnisse auftreten können.
Im Anschluß an die *Bildzeile* wird der Text unterhalb des *tiefsten* Bildes fortgesetzt.
Bildunterschriften für *Bildzeilen* können mit dem LaTeX-Kommando `\caption` oder dem `PicIns`-Kommando `\newcaption` erzeugt werden, jedoch nicht mit dem `\piccaption`-Befehl. Das `\newcaption`-Kommando unterscheidet sich von dem LaTeX-Kommando `\caption` durch die Ausrichtung von Bildunterschriften, die sich über mehrere Zeilen erstrecken. Bei `\newcaption` werden die Zeilen der Bildunterschrift *untereinander* ausgerichtet (s.a. die Unterschrift der vorherigen Abbildung).

4 Umrahmte Umgebungen

Um Textteile besonders hervorzuheben, kann man sie in eine oder in Kombinationen mehrerer der neuen Umgebungen einschließen (s. S. 2). Optional kann für die Umgebung eine *Breite* definiert werden (ohne Breitenangabe wird die aktuelle Zeilenbreite bzw. bei zweispaltigem Text die aktuelle Spaltenbreite verwendet). Verschachtelungen mit anderen Umgebungen sind möglich.

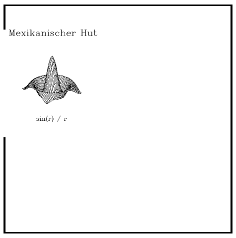
Syntax: `\begin`{*Umgebungsname*}[*Breite*] ··· `\end`{*Umgebungsname*}

Als *Umgebungsnahme* kann eine der Angaben `frameenv`, `dashenv`, `ovalenv` oder `shadowenv` eingesetzt werden.

Das folgende Beispiel zeigt eine schmale, gestrichelt umrahmte Umgebung innerhalb einer oval umrahmten Umgebung.

1. `\begin{ovalenv}`
2. `\begin{center}`
3. `\begin{dashenv}[9cm]`
4. `\begin{enumerate}`
5. es folgen viele Punkte (`\item`)
6. `\end{enumerate}`
7. `\end{dashenv}`
8. `\end{center}`
9. `\end{ovalenv}`

Besonders interessant dürfte die Kombination der `PicIns`-Makros mit der `figure`-Umgebung sein. Durch die Umrahmung wird die Zugehörigkeit von Bild und Text besonders deutlich. Ein Beispiel :



$$f(x,y) = \frac{\sin(\sqrt{x^2+y^2})}{\sqrt{x^2+y^2}}$$

Dieser *Mexikanische Hut* wurde mit dem Programm `PlotIt` berechnet, im PCL-Format des HP-Laserjet gespeichert, danach in das MSP-Format konvertiert und als externes Bild in das LaTeX-Dokument eingefügt (s.a. Abschnitt 5). Wir sehen ihn in Abb. 5 noch einmal etwas vergrößert. Der Bildinhalt ist nur zu sehen, wenn das Dokument mit dem Treiber DVIHPLJ der DVIDRV-Familie aus emTeX oder mit dem Treiber DVIHP (bzw. DVIKYO) der Treiberfamilie des Instituts für theoretische Informatik der TH Darmstadt ausgedruckt wurde. In allen anderen Fällen ist nur ein leerer Rahmen zu sehen.
Das hier gezeigte Beispiel ist eine `figure`-Umgebung, die eine `shadowenv`-Umgebung enthält. Darin wurde nach dem ersten Absatz durch das `\parpic`-Kommando das Bild mit dem mexikanischen Hut definiert. Die Formel ist der Inhalt des Absatzes, der neben den Bild erscheint.
Der Abstand zwischen Text und Umrahmung innerhalb der Umgebungen wird durch `\fboxsep` gesteuert.

Abbildung 4: Mexikanischer Hut (klein). In Abb. 5 ist er nochmal etwas größer zu sehen. Diese Unterschrift wurde mit dem `\newcaption`-Kommando erzeugt, das bewirkt, daß die Unterschrift-Zeilen *untereinander* ausgerichtet werden.

Kombinationen mit anderen Umgebungen sind ebenso möglich (s.a. Abb. 4, Seite 11). In diesem Beispiel wurde die Rahmendicke durch `\linethickness{4pt}` verändert.

5 Integration extern erstellter Bilder

Wie bereits erwähnt wurde, kann der Inhalt eines Bildes durch beliebige LaTeX-Konstruktionen definiert werden. Zur Erzeugung von Grafiken steht in LaTeX die `picture`-Umgebung zur Verfügung, die einige einfache Funktionen zur Darstellung von Linien, Kreisen und Kreisbögen enthält. Erweiterte Möglichkeiten bieten Makro-Pakete wie `pictex.tex` (Erweiterung der `picture`-Umgebung von LaTeX), `chemstru.tex` (Erzeugung chemischer Strukturformeln) oder `bezier.sty` (Erzeugung von Bezier-Kurvenlinien).

Diese in das LaTeX-System integrierten Funktion sind portabel und systemunabhängig, d.h. sie können mit jeder LaTeX-Implementierung in eine DVI-Datei umgesetzt werden und mit jedem TeX-

Bildschirmtreiber oder Druckertreiber auf den entsprechenden Ausgabegeräten ausgegeben werden. Außerdem werden zur Beschriftung der Grafiken die LaTeX-internen Schriftarten verwendet, wodurch sich die Grafiken sehr gut in den umgebenden Text einpassen. Da das LaTeX-System aber nicht zur Erzeugung von Grafiken, sondern zur Erzeugung von Texten konzipiert wurde, sind all diese Lösungen sehr aufwendig und schwerfällig und daher nur bedingt zur Darstellung von Grafiken geeignet. Die Nachteile liegen in dem zum Teil extrem großen Zeit- und Speicherbedarf für die Umsetzung der Grafiken durch LaTeX (viele LaTeX-Implemetierungen, insbesondere für MS-DOS, sind nicht in der Lage, sehr große oder komplexe Grafiken mit den oben genannten Makro-Paketen zu bearbeiten) sowie im geringen Funktionsumfang und der mangelnden Benutzerfreundlichkeit verglichen mit den vielen speziellen Grafiksystemen, die es für nahezu alle Betriebssysteme gibt.

Es liegt daher nahe, Grafiken, die mit speziellen Grafikprogrammen erstellt wurden, in TeX- oder LaTeX-Dokumente zu integrieren. Bekannte Beispiele aus dem MS-DOS Bereich sind

`AutoCad, AutoSketch, UniCAD` zur Erstellung von Konstruktionszeichnungen,

`PlotIT, MathCad, Matlab, Mathematica, GNUPlot` zur Erstellung mathematischer Grafiken,

`Harvard Graphics, Lotus 1-2-3, MS-Excel` zur Erstellung von Geschäfts- und Präsentations-Grafiken,

`MS-Paint, PC-Paintbrush` zur Erstellung pixelorientierter Zeichnungen,

`Arts&Letters, Designer, CorelDraw` zur Erstellung vektororientierter Zeichnungen mit hohen Anforderungen an die Qualität von Grafik- und Text-Objekten.

`IDL` zur Darstellung und Manipulation beliebiger grafischer Objekte.

Etliche dieser Programme (z.B. `AutoCad, Mathematica, Gnuplot, CorelDraw, IDL`) sind auch für viele Unix-Systeme verfügbar. Darüber hinaus existieren für UNIX eine Reihe weiterer leistungsfähiger Grafik-Systeme (z.B. `AVS, XFig` u.a.).

In der letzten Zeit wurden eine Reihe unterschiedlicher Ansätze für die Integration extern erstellter Grafiken in TeX- und LaTeX-Dokumente entwickelt. Die interessantesten Ansätze, für die es zum Teil auch schon Lösungen gibt, werden in den folgenden Abschnitten vorgestellt.

5.1 Konvertierung von Grafiken in TeX-Makro-Definitionen

Wegen der oben beschriebenen Unhandlichkeit der LaTeX-internen Lösungen zur Beschreibung von Grafiken wäre es von Vorteil, wenn man extern erstellte Grafiken automatisch in die entsprechenden TeX-Makroaufrufe umsetzen könnte. Dadurch würden die Portierbarkeit und Geräteunabhängigkeit erhalten, aber es müßten auch die oben angesprochenen Nachteile dieser Lösung in Kauf genommen werden (lange Bearbeitungszeit durch LaTeX, Speicherplatzproblematik).

Dieser Lösungsansatz ist als einziger der hier beschriebenen Ansätze systemunabhängig und gleichzeitig vollständig portabel und einfach anzuwenden. Bekannte Implementierungen dieses Ansatzes sind das unter MS-DOS laufenden Programm `TEXCAD` (`TEXCAD` ist ein Zeichenprogramm, das die Erstellung einfacher Bilder ermöglicht. Die Bildbeschreibungen werden als LaTeX-Kommandos in einer Datei ablegt) oder die unter MS-DOS und Unix zur Verfügung stehenden Programme `GNUPlot` und `XFig`.

5.2 Konvertierung von Grafiken in METAFONT-Definitionen

Diese Methode hat ebenfalls zum Ziel, die Systemunabhängigkeit bei der Integration extern erstellter Grafiken zu erhalten. Dabei sollen aber gleichzeitig die Nachteile der internen LaTeX-Lösungen und des unter 5.1 beschriebenen Ansatzes vermieden werden. Das Hilfsmittel für diesen Lösungsweg ist das METAFONT-System. METAFONT ist ein Programm, mit dessen Hilfe Grafikobjekte beschrieben und in das Bitrasterformat für beliebige Rasterausgabegeräte (vom Fotobelichter mit über 2000 dots per inch (dpi) Auflösung über die gängigen Laserdrucker mit 300 dpi, Nadel- und Tintenstrahldrucker mit 180-360 dpi bis zu Grafikbildschirmgeräten mit etwa 100 dpi Auflösung) konvertiert werden können. METAFONT wurde speziell für die Beschreibung einer besonderen Klasse von Grafikobjekten, nämlich die Klasse der Schriftzeichen beliebiger Schriftarten, entwickelt. Da das Programm aber keine relevanten Einschränkungen hinsichtlich der Form, der Struktur oder der Größe der Schriftzeichen enthält, kann es ebenso gut für die Beschreibung beliebiger allgemeiner Grafikobjekte verwendet werden.

METAFONT kann logisch zusammengehörende Grafikobjekte in sogenannten Zeichensätzen zusammenfassen. Es erzeugt für jeden Zeichensatz eine TFM-Datei, die im wesentlichen die Größenangaben für die in dem Zeichensatz definierten Zeichen (Grafikobjekte) enthält sowie eine GF-Datei, die das Bitraster für alle Zeichen des Zeichensatzes (d.h. das optische Erscheinungsbild) in der vorgewählten Auflösung beschreibt. Die TFM-Datei wird vom TEX- bzw. LATEX-System benötigt, um die Zeichen im Dokument im richtigen Abstand nebeneinander und untereinander anzuordnen. Die GF-Datei dient als Grundlage für die Bildschirm- und Druckerausgabetreiber, die die Zeichen der einzelnen Zeichensätze gemäß ihrer optischen Gestalt und ihrer Größe darstellen. Manche TEX-Treiberprogramme können GF-Dateien direkt interpretieren, andere benötigen ein Bitrasterformat, das direkt aus dem von METAFONT erzeugten GF-Format abgeleitet werden kann. Die gängigsten von TEX-Treibern verwendeten Bitrasterformate sind das kompakte PK-Format und das speicheraufwendige, aber einfach zu interpretierende PXL-Format. In jedem Fall ist es nötig, daß die in einem TEX- oder LATEX-Dokument angesprochenen Zeichensätze in einem Bitrasterformat vorliegen, das der Geräteauflösung des gewünschten Ausgabegerätes entspricht. Der besondere Vorteil von METAFONT liegt darin, daß die Zeichen eines Zeichensatzes nur einmal beschrieben werden müssen und dann für jedes beliebige Rasterausgabegerät und in jeder beliebigen Vergrößerung zur Verfügung gestellt werden können.

Dieser Vorteil soll auch bei dem in diesem Abschnitt beschriebenen Ansatz zur Integration extern erstellter Grafiken genutzt werden. Nach der Konvertierung der Grafik in eine für METAFONT verständliche Beschreibung und die Bereitstellung der TFM- und GF-Dateien durch METAFONT kann die Grafik wie jedes andere Zeichen eines TEX-Zeichensatzes an beliebigen Stellen innerhalb eines Dokumentes verwendet werden. Da die Zeichen von TEX als unteilbare Objekte aufgefaßt werden, unterliegen solche konvertierten Grafiken dem normalen Zeilen- und Seitenumbruch und sind völlig frei innerhalb des Dokumentes verschiebbar. Wegen der geräteunabhängigen METAFONT-Beschreibung können einmal definierte Grafiken in das Bitrasterformat für beliebige Ausgabegeräte konvertiert werden. Der gesamte Zeitaufwand wird bei der Interpretation der Beschreibung und der Umrechnung ins Bitrasterformat innerhalb des METAFONT-Programms aufgewendet. Bei der späteren (im allgemeinen häufigen) Verwendung der Grafiken als Zeichen eines speziellen Zeichensatzes im TEX- oder LATEX-Dokument tritt dann im Gegensatz zu der im Abschnitt 5.1 dargestellten Methode keinerlei Zeitverlust oder Speicherengpaß auf.

Leider ist aber auch dieser Ansatz nicht frei von einigen relevanten Nachteilen. Die gesamte Prozedur von der Konvertierung der externen Grafik in das METAFONT-Format über die Erzeugung der TFM- und GF-Dateien (gegebenenfalls für mehrere Ausgabegeräte), die Verteilung der TFM- und GF-Dateien (und eventuelle vorherige Konvertierung der GF-Dateien in ein Format, das der betreffende Ausgabetreiber versteht) an die Stellen, wo sie von TEX, LATEX bzw. von den Ausgabetreibern erwartet werden, und die Einbindung als Zeichen der von METAFONT erzeugten Zeichensätze in das Dokument ist eine aufwendige Angelegenheit und kann nicht vollständig automatisiert werden. Für ungeübte Benutzer ist der gesamte Vorgang sicher nicht akzeptabel. Außerdem steht nicht allen TEX- oder LATEX-Anwendern das METAFONT-Programm zur Verfügung. Schließlich haben viele der häufig verwendeten TEX-Ausgabetreiber Probleme bei der Ausgabe von großen Zeichen, was die gesamte Methode für die Einbindung großer Grafiken zunichte machen würde. Hier könnte man sich dadurch behelfen, daß große Grafiken nicht in ein einziges Zeichen umgewandelt werden, sondern in viele Zeichen eines Zeichensatzes, die dann nur in der richtigen Reihenfolge zusammengesetzt werden müssen, um die Ursprungsgrafik wieder korrekt darzustellen. Dieses Verfahren verschärft aber wieder die oben beschriebene Problematik der Anwendbarkeit der Prozedur. Ein weiterer, unter Umständen sehr einschränkender Nachteil ergibt sich daraus, daß es im Allgemeinen nicht möglich ist, Grafiken aus einem Bitrasterformat in eine äquivalente METAFONT-Beschreibung umzuwandeln. Da die METAFONT-Beschreibungen selbst Vektorbeschreibungen sind, müssen auch die Grafikformate, aus denen sie erzeugt werden sollen, Vektorformate sein. Dadurch wird praktisch die gesamte Klasse von Grafiken, die über Scanner gewonnen werden, von diesem Verfahren ausgeschlossen. Als letzter Problempunkt soll die mangelnde Portierbarkeit von Dokumenten, die diese Variante der Grafikeinbindung nutzen, angesprochen werden. Es ist in der Regel kein Problem, TEX-Dokumente auf der Basis von DVI-Dateien an andere Personen weiterzugeben, da es heute üblich ist, daß die TEX-Ausgabetreiber Zugriff zu den PXL- oder PK-Dateien für alle Standard-TEX-Zeichensätze in der benötigten Geräteauflösung haben. Der Druckvorgang ist automatisiert, so daß die Benutzer, die ein TEX-Dokument ausdrucken wollen, nur ein bestimmtes Kommando aufrufen müssen. Wenn nun aber das auszudruckende Dokument auf Zeichensätze zugreift, die nicht Bestandteil der Standard-TEX-Zeichensätze sind (wie im hier diskutierten Fall der Einbindung extern erstellter Grafiken), so müssen diese Zeichensatzbeschreibungsdateien (d.h. die PXL- oder PK-Dateien) an den Stellen zur Verfügung gestellt werden, an denen sie von dem betreffenden Ausgabetreiber erwartet werden. Dies erfordert eine genaue Kenntnis der vom TEX-System verwendeten Dateiorganisation und wird außerdem von den einzelnen Treiberprogrammen unterschiedlich gehandhabt, so daß ungeübte Benutzer kaum in der Lage sind, diesen Vorgang korrekt auszuführen.

Unter dem Namen HPtoXX ist ein Konvertierungsprogramm verfügbar, das Grafiken, die in dem sehr weit verbreiteten HPGL-Format vorliegen, in eine äquivalente METAFONT-Beschreibung umwandelt (HPtpXX ist aus dem ebenfalls noch verfügbaren, älteren Programm HPtoMF entstanden). Das Programm kann über die Deutschsprachige Anwendervereinigung TEX e.V. (DANTE) oder auf elektonischem Weg über die TEX-Server in Heidelberg oder Stuttgart bezogen werden. Es ist unter dem Betriebssystem MS-DOS und SUN-OS ablauffähig, Hinweise für die Verwendung des Programms können der Benutzeranleitung entnommen werden.

5.3 Konvertierung von Grafiken in TEX-Zeichensätze

Dieser Ansatz kürzt das im Abschnitt 5.2 beschriebene Verfahren erheblich ab, indem er aus externen Grafikformaten direkt TFM- und PK- bzw. PXL-Dateien erzeugt. Dabei entfällt der *Umweg* über METAFONT, womit aber auch die Systemunabhängigkeit auf der Strecke bleibt (d.h. TEX- Dokumente, die Grafiken mit dieser Methode einbinden, können nur auf solchen Geräten ausgegebem werden, deren Rasterauflösung den Vorgaben der PK- bzw. PXL-Dateien entspricht; im Gegensatz zur METAFONT-Methode ist es ohne das Konvertierungsprogramm nicht möglich, PK- oder PXL-Dateien für andere Auflösungen zu erzeugen). Alle anderen Schritte, die für die Integration der Grafiken in die Dokumente nötig sind, entsprechen dem in Abschnitt 5.2 beschriebenen Verfahren. Dabei treten auch die gleichen Schwierigkeiten und Probleme auf, die bereits im Abschnitt 5.2 beschrieben wurden.

Auch für diesen Ansatz gibt es Implementierungen, die unter MS-DOS laufen. Das Programm RUMGRAPH kann aus dem im MS-DOS Bereich sehr häufig verwendeten Bitrasterformat PCX die von TEX, LATEX und den TEX-Treibern benötigten TFM- und PXL-Dateien erzeugen. Für Treiber, die nur das PK-Format interpretieren können, müssen die PXL-Dateien mit ebenfalls frei verfügbaren Konvertierungsprogrammen in das PK-Format umgesetzt werden. Das leistungsfähigere Programm BM2FONT kann mehrere Bitrasterformate interpretieren und generiert direkt das PK-Format. Beide Produkte können über die Deutschsprachige Anwendervereinigung TEX e.V. (DANTE) oder die bereits erwähnten TEX-Server bezogen werden.

5.4 Einbindung externer Grafiken über das \special-Kommando

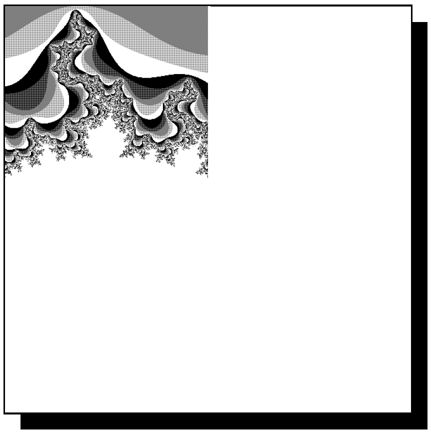
Mit Hilfe des TEX-Kommandos `\special` ist es möglich, Informationen aus einem TEX- oder LATEX-Dokument ungefiltert an den Bildschirm- oder Druckertreiber weiterzugeben. Auf diesem Weg könnte man dem Ausgabetreiber den Namen einer extern erstellten Grafikdatei mitteilen. Der Treiber könnte den Inhalt der Grafikdatei interpretieren und die Grafik in das auszugebende TEX- oder LATEX-Dokument einfügen. Zur Zeit sind mehrere TEX-Druckertreiber verfügbar, die diese Funktionen unterstützen (z.B. die an der TH Darmstadt entwickelten Treiber für HP-LaserJet-kompatible Laserdrucker[4], die Treiber der DVIDRV-Familie aus emTEX oder der weit verbreitete Treiber DVIPS für POSTSCRIPT-Drucker).

Die einzubindenden externen Grafiken müssen als Dateien in einem Format vorliegen, das entweder vom Treiber (treiberspezifisches Grafikformat) oder vom Drucker (druckerspezifisches Grafikformat) interpretiert werden kann. Als treiberspezifische Grafikformate sind vor allem solche Formate interessant, die von vielen Grafikprogrammen erzeugt werden können. Die Grafiken, die in einem treiberspezifischen Format vorliegen, werden vom Treiber interpretiert und in ein für den entsprechenden Drucker passendes, druckerspezifisches Grafikformat umgewandelt. Der Vorteil dieses Verfahrens liegt darin, daß der Treiber die einzubindenden Grafiken noch in der Größe skalieren und im Aussehen verändern kann (z.B. Ausblenden oder Abschneiden bestimmter Teile der Grafik), bevor sie in das TEX-Dokument eingefügt werden. Die Treiber der DVIDRV-Familie können die bei MS-DOS-Programmen weit verbreiteten Grafikformate MSP und PCX verarbeiten. Nach unser Information sind zur Zeit leider keine TEX-Druckertreiber verfügbar, die das bekannte Standard-Grafikformat `HPGL` für Vektorgrafiken interpretieren können.

Andere Druckertreiber (z.B. die oben angeführten Treiber der TH Darmstadt) unterstützen die Einbindung von Grafiken, die in einem *druckerspezifischen* Grafikformat vorliegen. In diesem Fall liest der Treiber die Grafikdatei und gibt sie ohne weitere Interpretation an den Drucker weiter. Dabei ist aber darauf zu achten, daß die Grafikdateien keine Druckersteuerbefehle enthalten, die den vom TEX-Treiber eingestellten Druckermodus verändern. Dazu gehören insbesondere Kommandos, die einen Seitenvorschub erzwingen, den Drucker in seinen Grundzustand zurücksetzen oder andere Druckersteuerkommandos, die eine vom Treiber vorgenommene Einstellung verändern. Außerdem dürfen die in der Grafikdatei enthaltenen Steuerkommandos keine absolute Drucker-Positionierung (d.h. relativ zur linken oberen oder linken unteren Ecke der Druckseite), sondern nur relative Drucker-Positionierung (d.h. relativ zur aktuellen Druckposition) vornehmen. Nur so kann eine beliebige Verschiebbarkeit der Grafik innerhalb einer Seite gewährleistet werden.

Für das von den Laserdruckern der HP Laserjet-Familie und den dazu kompatiblen Druckern verwendete Druckerformat PCL (das einen de-facto Standard darstellt) steht das vom Hochschulrechenzentrum der TH-Darmstadt entwickelte Konvertierungsprogramm `PCLRel` zur Verfügung. Dieses Programm läuft unter MS-DOS und modifiziert PCL-Dateien derart, daß sie direkt in TEX- bzw. LATEX-Dokumente eingebunden werden können (Seitenvorschub-Kommandos und andere Kommandos, die den Druckermodus verändern, werden gelöscht; der Modus für absolute Drucker-Positionierung wird in den Modus für relative Drucker-Positionierung geändert). Das Programm kann die meisten (aber nicht alle) der von den bekannten Grafik-Programmen erzeugten PCL-Dateien bearbeiten.

[4] weitere Informationen können über das Hochschulrechenzentrum oder das Institut für theoretische Informatik der TH Darmstadt bezogen werden



Wenn Sie den Laser-Drucker-Treiber DVIHPLJ der DVIDRV-Treiberfamilie oder den Treiber DVIHP (bzw. DVIKYO) der Treiberfamilie des Instituts für theoretische Informatik der TH Darmstadt verwenden, so sollten Sie links einen Ausschnitt eines "Apfelmännchens" sehen. Wenn nicht, sehen Sie nur einen Schatten dieser Pracht.

Die Syntax des `\special`-Kommandos zur Definition einer externen Grafik-Datei wird durch den jeweiligen Druckertreiber vorgegeben. Bei den Treibern DVIHP und DVIKYO lautet das Kommando zur Integration einer PCL-Grafik-Datei `\special{HP:INCLUDE:`*Grafikdatei*`}`. Bei den Treibern der DVIDRV-Familie lautet das Kommando zum Einfügen einer MSP- oder PCX-Grafik-Datei `\special{em:graph` *Grafikdatei*`}`. Da es für diese Syntax noch keine weltweit einheitlichen Regeln gibt, ist eine Änderung in einer späteren Version der Treiber sehr wahrscheinlich. Damit in einem solchen Fall nicht alle TeX- bzw. LaTeX-Dokumente, die externe Grafiken einbinden, geändert werden müssen, sollte die Syntax des `\special`-Kommandos **unbedingt** in einer eigenen Kommando-Definition (TeX- bzw. LaTeX-Makro) *versteckt* werden. Die Kommando-Definitionen für die verschiedenen Drucker-Treiber bzw. druckerspezifischen Grafikformate sollten in einer eigenen Makro-Datei gesammelt werden. Erfolgt dann irgendwann eine Änderung der Syntax des `\special`-Kommandos oder soll eine bestimmtes Dokument, das externe Grafiken enthält, mit einem anderen Druckertreiber ausgedruckt werden, so muß nur diese Makro-Datei geändert werden und die TeX- bzw. LaTeX-Eingabedateien bleiben unverändert. Danach muß das betreffende Dokument mit TeX bzw. LaTeX neu umgesetzt werden und kann anschließend mit dem gewünschten Druckertreiber ausgedruckt werden. Als Beispiel für eine Makro-Datei, die Kommando-Definitionen zur Einbindung mehrerer druckerspezifischer Grafikformate für verschiedene Druckertreiber enthält, ist die Datei `extpic.sty` in der Auslieferung von `PicIns` enthalten.

Die externen Bilder in diesem Dokument wurden durch die in der Makro-Datei `extpic.sty` definierten Kommandos `\hpinc` (für die Treiber DVIHP und DVIKYO) bzw. `\eminc` (für die Treiber der DVIDRV-Familie) in das Dokument aufgenommen. Damit lautet das vollständige Kommando für die Darstellung des obigen Bildes

```
\parpic(5.4cm,5.4cm)(0pt,0pt)[s]{\hpinc{mandel.pcl}}
```

bzw.

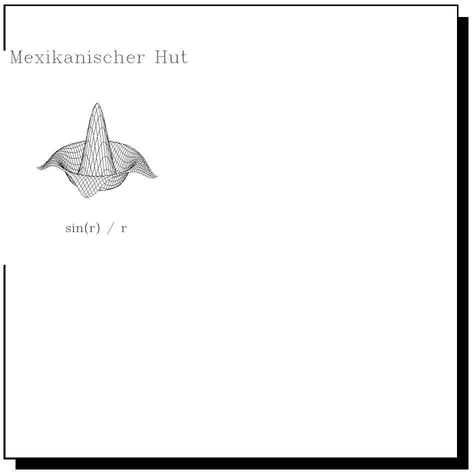
```
\parpic(5.4cm,5.4cm)(0pt,0pt)[s]{\eminc{mandel.msp}}.
```

Die Methode der Einbindung extern erstellter Grafiken über das `\special`-Kommado ist in vielen Fällen die einfachste und schnellste Integrationsmethode. Da sie aber vollständig auf den Ausgabetreiber und die vom Treiber unterstützten Ausgabegeräte abgestimmt ist, kann damit keinerlei Geräteunabhängigkeit und Portabilität erreicht werden. TeX-Dokumente und die dazugehörenden Grafiken in externen Formaten können nur dann an anderer Stelle ausgedruckt werden, wenn dort der gleiche Ausgabetreiber und das gleiche Ausgabegerät vorhanden sind.

Der wesentliche Nachteil dieser Methode liegt darin, daß zum Zeitpunkt der Umsetzung des Dokuments mit TeX bzw. LaTeX nichts über die Größe des Bildes bekannt ist (da das Bild erst vom Ausgabetreiber eingefügt wird). Die Funktionen der `PicIns`-Kommandos `\parpic` und `\hpic` zur automatischen Größenbestimmung und Zentrierung des Bildinhalts können daher nicht angewendet werden (s.a. Abschnitt 5.6).

5.5 Konvertierungshilfen

Die meisten der bekannten Grafik-Programme speichern die erzeugten Grafiken in einem eigenen programmspezifischen Format. Auf diese Weise haben sich im Laufe der Zeit sehr viele unterschiedliche



Wie versprochen erscheint hier noch einmal der *Mexikanische Hut*, diesmal aber ein wenig größer. Voraussetzung ist, daß Sie den Druckertreiber DVIHPLJ der Treiberfamilie DVIDRV oder den Druckertreiber DVIHP (bzw. DVIKYO) aus der Treiberfamilie des Instituts für theoretische Informatik der TH Darmstadt benutzen.

Abbildung 5: *Mexikanischer Hut* (groß)

Grafikformate herausgebildet, die untereinander völlig inkompatibel sind. Einige dieser Formate haben sich zu Quasi-Standards entwickelt und die neueren Versionen vieler Grafikprogramme sind in der Lage, ihre Grafiken auch in einigen dieser Standardformate abzulegen.

In den Fällen, in denen dies nicht machbar ist, besteht die Möglichkeit, die Grafik in einem anderen Format abzulegen, das dann von einem der vielen auf dem Markt befindlichen Grafik-Konvertierungsprogramme in das benötigte Format umgewandelt werden kann. Besonders günstig ist es, die Grafiken in einem gebräuchlichen Vektorformat zu speichern, da sie dann bei der späteren Weiterbearbeitung noch auf die gewünschte Größe skaliert werden können. Bei Bitrasterformaten ist eine nachträgliche Skalierung gar nicht oder nur um den Preis großer Qualitätsverluste möglich.

Die bekanntesten Vektorformate sind die Formate HPGL der Firma Hewlett Packard und POSTSCRIPT der Firma Adobe. Für beide Formate gibt es mehrere Konvertierungsprogramme, die aus den Ausgangsformaten andere Vektorformate bzw. Bitraster- oder Druckerformate erzeugen können. Als Beispiele seien das unter MS-DOS und UNIX laufenden Programme GHOSTCRIPT (für das POSTSCRIPT-Format) und das unter MS-DOS laufende Programm HIJAAK (für das HPGL-Format) genannt. Beide Programme sind kommerziell verfügbare Produkte. Daneben gibt es das bereits erwähnte *public domain* Produkt HPtoXX sowie das hervorragende Shareware-Programm PRINTGL zur Konvertierung von HPGL-Grafiken in Bitrasterformate. Die Programme können bei der Umwandlung des Vektorformats in ein Bitrasterformat (zu den Bitrasterformaten zählen auch die meisten Druckerformate) die Auflösung des Bitrasters (d.h. die Anzahl der Rasterpunkte pro Längeneinheit) berücksichtigen, so daß es möglich ist, die selbe Grafik für verschiedene Ausgabegeräte zu erzeugen. Auf diesem *Umweg* kann man auch für die im Abschnitt 5.3 beschriebene Methode die Systemunabhängigkeit gewährleisten.

Auch für die meisten Bitrasterformate stehen verschiedene Konvertierungsprogramme zur Verfügung. Das oben erwähnte Programm HIJAAK kann die meisten bekannten Bitrasterformate ineinander überführen. Das *public domain* Programm PCLtoMSP konvertiert Grafiken aus dem PCL-Format in die Formate PCX oder MSP. Da praktisch jedes Grafikprogramm im MS-DOS Bereich das PCL-Rasterformat erzeugen kann, sollte es für die allermeisten Grafiken eine Möglichkeit der Integration in ein TEX- oder LATEX-Dokument geben.

5.6 Verschieben des Bildinhalts innerhalb des Rahmens

Die Beispiele in Abschnitt 2 haben gezeigt, daß die Lage des Bildinhalts innerhalb des Rahmens von unterschiedlichen Faktoren abhängt. Die automatische Größenbestimmung und Ausrichtung des Bildinhaltes innerhalb des Rahmens kann nur bei Konstrukten angewendet werden, die eine definierte Höhe und Breite haben. Dies ist bei extern erzeugten Bildern nur dann der Fall, wenn sie mit den Methoden 5.1 – 5.3 eingebunden werden. Bilder, die über das `\special`-Kommando integriert werden, haben keine definierte Größe. Für solche Bilder muß die Bildgröße explizit angegeben werden und die Ausrichtung muß über den *offset*-Parameter erfolgen, wobei die linke obere Ecke des Rahmens den Bezugspunkt darstellt.

Die linke obere Ecke stellt jedoch nur in Ausnahmefällen den korrekten Einfügepunkt für den Bildinhalt dar (nämlich dann, wenn der Bildinhalt ausgehend vom Bezugspunkt nach unten und nach rechts gezeichnet wird und außerdem den gesamten Rahmen ausfüllt). In allen anderen Fällen ist mittels der `\parpic`-optionen *x-offset* und *y-offset* eine Verschiebung des Bildinhalts in die geeignete Richtung vorzunehmen, so daß der Bildinhalt korrekt innerhalb des Rahmens zentriert ist. Hierbei wird es in vielen Fällen nötig sein, einen Probeausdruck mit den einzufügenden Bildern zu erstellen und anschließend durch Abmessen die Verschiebewerte zu bestimmen.

Durch die Definition von negativen Verschiebewerten ist es möglich, auch solche Bilder zu integrieren, die am oberen bzw. linken Rand leeren Platz enthalten, der sich wie eine Verschiebung des Bildinhaltes nach unten bzw. nach rechts auswirkt. Solche *Freiräume* kommen bei extern erstellten Bildern häufig vor, da die Grafikprogramme in der Regel davon ausgehen, daß ein Bild auf einer eigenen Ausgabeseite ausgedruckt werden soll und das Bild daher in die Mitte der Seite zentrieren. Einige der erwähnten Konvertierungsprogramme sind in der Lage, beliebige Ausschnitte des Bildes zu bearbeien. Damit ergibt sich die Möglichkeit, unerwünschte Teile eines Bildes zu eliminieren.

Zur Kontrolle der korrekten Position der eingebundenen Bilder eignen sich Preview-Programme (das sind TEX-Ausgabe-Treiber für die Darstellung der TEX-Dokumente an einem Bildschirm), die die integrierten Bilder direkt anzeigen können (als Beispiel seien die Preview-Programme aus dem emTEX-Paket genannt, die Grafiken im PCX- bzw. MSP-Format am Bildschirm darstellen). Hierbei ist jedoch zu beachten, daß die Auflösung der Grafik-Dateien der Auflösung der verwendeten PK- oder PXL-Dateien entspricht. Liegen die Grafik-Dateien beispielsweise in einer Auflösung von 300 dpi vor, so müssen im Preview-Programm statt der üblicherweise verwendeten PK- oder PXL-Dateien (100 dpi Auflösung) die PK- bzw. PXL-Dateien mit 300 dpi Auflösung verwendet werden.

5.7 Vergleich der verschiedenen Integrationsansätze

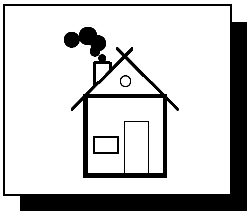
Die in den letzten Abschnitten beschriebenen Ansätze zu Integration von extern erstellten Grafiken in TEX- und LATEX-Dokumente unterscheiden sich zum Teil erheblich voneinander. Dies trifft in abgeschwächtem Maße auch auf unterschiedliche Implementierungen für die einzelnen Ansätze zu. Aus diesem Grund sollen in diesem Abschnitt die spezifischen Vor- und Nachteile der skizzierten Methoden tabellarisch zusammengefaßt werden (s.a. Tabelle 1 auf Seite 22). Dabei sollen alle für den Benutzer relevanten Punkte bei der Einbindung extern erstellter Grafiken in ein TEX- oder LATEX-Dokument bewertet werden. Abhängig von der Wahl der Methode sind die folgenden Schritte auszuführen:

- Konvertierung der Grafiken aus dem vom Grafikprogramm erstellten externen Format in ein Format, das von der ausgewählten Methode weiterverarbeitet werden kann. Dieser Schritt vereinfacht sich, wenn das Grafikprogramm bereits das benötigte Format und die gewünschte Größe der Grafik liefern kann. Wenn die Grafik für verschiedene unterschiedliche Ausgabegeräte zur Verfügung gestellt werden soll (z.B. Drucker- und Bildschirmausgabe), ist der Schritt entsprechend oft zu wiederholen.

- Bearbeiten der im letzten Schritt zur Verfügung gestellten Grafikdateien mit den Programmen der ausgewählten Methode. Dieser Schritt entfällt bei einigen der vorgestellten Ansätze.
- Einbindung der aufbereiteten Grafiken in das TeX- oder LaTeX-Dokument. Dazu gehört das Ansprechen der Grafiken als TeX-Makro (Ansatz 5.1), als Zeichen eines neuen Zeichensatzes (Ansatz 5.2 und 5.3) oder als `\special`-Kommando (Ansatz 5.4), die Ausrichtung der Grafik innerhalb des vorgesehenen Bereichs sowie bei den Ansätzen 5.2 und 5.3 das Kopieren der erzeugten TFM-Datei an die vom TeX-System geforderte Stelle. Gerade der letzte Punkt kann große Probleme verursachen, wenn ein Benutzer mit einem zentral installierten TeX-System arbeitet (an einem Mehrbenutzersystem oder in einem Netzwerk). In diesem Fall gibt es oft keine Möglichkeit, eigene Zeichensatzdateien in das System einzubringen.
- Umsetzung des Dokumentes mit TeX oder LaTeX. Hierbei können bei einigen Ansätzen signifikante Zeit- und Speicherplatzprobleme auftreten.
- Bereitstellung der für die Einbindung aufbereiteten Grafiken für den TeX-Ausgabetreiber. Hier kann es bei den Ansätzen 5.2 und 5.3 zu den bereits beschriebenen organisatorischen Problemen kommen.
- Ausdrucken des Dokumentes inklusive der eingebundenen Grafiken, wobei manche Ausgabetreiber Probleme mit sehr großen Zeichen haben.

6 Probleme

Etwaige Probleme, die bei der Benutzung von `PicIns` auftreten können, sollen nicht verschwiegen werden. Die Berechnung der Anzahl der Zeilen, die *neben* das Bild passen (und somit eingerückt werden müssen) erfolgt unter der Annahme, daß die *Schriftgröße* zumindest während eines Absatzes konstant bleibt.



Dies ist nicht mehr gegeben, wenn z.B. innerhalb eines Absatzes von `\normelsize` nach `\huge` gewechselt wird.

Es ist aber erlaubt, einen ganzen Absatz in einer anderen Schriftgröße zu schreiben.

Sollte es hierbei Probleme geben, kann man diesen mit dem Kommando `\picskip`{*n*} entgegenwirken.

"Klebt" der unterhalb des Bildes stehende Text zu dicht an dem Bild, so liegt das meist daran, daß die Bildhöhe in etwa einem Vielfachen der Zeilenhöhe entspricht. Als Gegenmaßnahme kann die Bildhöhe um wenige Punkte vergrößert oder, wenn dies nicht akzeptabel ist, das `\picskip`-Kommando verwendet werden.

Alle vorgestellten Kommandos und Definitionen können auch im `\twocolumn`-Modus verwendet werden. Der Einsatz von `\parpic` ist jedoch nur dann sinnvoll, wenn das Bild so schmal ist, das noch bequem Text daneben paßt. Es muß aber mit etlichen `overfull` oder `underfull \hboxen` gerechnet werden.

7 Versions-Historie

- Version 1.0
  Definition des `\parpic`- und des `\hpic`-Kommandos mit den Umrahmungsoptionen `f` und `s`.
  Definition des `\picchangemode` (Wechselmodus).

- Version 1.1
  Korrektur des `\picchangemode`.
  Einführung der Optionen `d` und `o` für `\parpic` und `\hpic`.
  Definition der Umgebungen `frameenv` und `shadowenv`.
- Version 1.2
  Korrektur der Umgebungen, Einführung zweier weiterer Umgebungen `dashenv` und `ovalenv` mit optionaler Breitenangabe.
- Version 1.3
  Korrektur der `d`-Option von `\parpic` und `\hpic`.
  Korrektur der `dashenv`-Umgebung.
- Version 1.4
  Vollständige Überarbeitung aller Optionen von `\parpic` und `\hpic`.
  Einführung des `\piccaption-` und des `\pichskip`-Kommandos.
- Version 1.5
  Korrektur der Schattenumrahmung und Dokumentation von `PicIns`
- Version 2.0
  Vollständige Überarbeitung und Umstrukturierung von `picins.sty`.
  Korrektur der fehlerhaften Positionierung von Bildern am unteren Rand einer Seite.
  Korrektur von Umbruch-Problemen bei Dokumenten mit verändertem Zeilenabstand.
  Neue Option `[x]` für `\parpic` und `\hpic`.
  Option für automatisches Positionieren des Bildes für `\parpic` und `\hpic` innerhalb des Rahmens.
  `\parpic` und `\hpic` sind nun auch innerhalb von LIST-Umgebungen (wie z.B. der `itemize` oder der `quote` Umgebung) verwendbar.
- Version 2.1 (Nov. 91)
  Fehlerkorrekturen (wenn zwischen dem `\parpic`-Kommando und dem anschließenden Text Leerzeilen auftraten, wurde keine korrekter Seitenumbruch erzeugt).
- Version 3.0 (Juli 92)
  Erweiterung der Möglichkeiten zum Setzen von Unterschriften.
  Korrektur der Absatzumbrechung und von `\parpic`.
  Korrektur von `\picskip` für `figures` und `minipages`.

8 Noch Fragen?

Wenn Probleme oder Fehler bei der Benutzung von `PicIns` auftreten, oder wenn Sie Kritik und Anregungen zu `PicIns` äußern wollen, so wenden Sie sich bitte an

Dr. Edmund Lang
TH Darmstadt
Hochschulrechenzentrum
Petersenstraße 30
6100 Darmstadt
Bundesrepublik Deutschland
Tel: 06151/16-3458
E-Mail: lang@hrz.th-darmstadt.de

Hier erhalten Sie auch weitere Informationen über neuere Versionen von `PicIns`.

	Makro-Methode	METAFONT-Methode	Zeichensatz-Methode	`\special`-Methode
Konvertierung in das von der Methode benötigte Format	geringer bis mittlerer Aufwand	geringer bis mittlerer Aufwand	geringer bis mittlerer Aufwand	geringer bis mittlerer Aufwand
Bearbeiten der Grafik mit den Programmen der Methode	entfällt	mittlerer Aufwand; METAFONT wird vorausgesetzt	mittlerer Aufwand	entfällt
Einbindung der Grafik in das Dokument	mittlerer Aufwand	relativ großer Aufwand (unter Umständen ist die Methode nicht anwendbar)	relativ großer Aufwand (unter Umständen ist die Methode nicht anwendbar)	mittlerer Aufwand
Umsetzung des Dokuments mit TEX oder LATEX	mitunter sehr großer Zeitaufwand; manche TEX-Installationen können große Grafiken nicht bearbeiten	kein Aufwand	kein Aufwand	kein Aufwand
Bereitstellen der Grafik für den Ausgabetreiber	kein Aufwand	mittlerer Aufwand (unter Umständen ist die Methode nicht anwendbar)	mittlerer Aufwand (unter Umständen ist die Methode nicht anwendbar)	geringer Aufwand
Ausdrucken des Dokumentes	mitunter sehr großer Zeitaufwand	geringer bis mittlerer Zeitaufwand; manche Treiber können sehr große Zeichen nicht drucken.	geringer bis mittlerer Zeitaufwand; manche Treiber können sehr große Zeichen nicht drucken.	geringer bis mittlerer Zeitaufwand
Systemunabhängigkeit	ja	bedingt	bedingt	nein
Portabilität	ja	bedingt	bedingt	nein

Tabelle 1: Vergleich der Methoden zur Einbindung externer Grafiken