Plaatjes in een tekst

Jeroen Fokker & Piet van Oostrum

Dept. of Computer Science
Utrecht University
jeroen@cs.ruu.nl piet@cs.ruu.nl

Abstract

In deze artikel wordt eerst aandacht besteed aan de (beperkte) mogelijkheden die LaTeX zelf biedt voor het maken van plaatjes. Soms zijn die voldoende, en dan is het wel zo gemakkelijk om die te gebruiken. Daarna wordt de rol van PostScript besproken bij het gebruik van plaatjes, al of niet in combinatie met LaTeX. Vervolgens wordt een vijftal tekenprogramma's besproken waarmee interactief ingewikkeldere figuren gemaakt kunnen worden. Tenslotte wordt aangegeven hoe 'bitmaps' (letterlijke afbeeldingen van het scherm van een computer) als plaatje behandeld kunnen worden. Dit laatste is vooral van belang bij het schrijven van handleidingen voor programma's.

Inleiding

Steeds meer uitgevers gebruiken LaTeX voor de opmaak van wetenschappelijke teksten in tijdschriften en boeken. LaTeX is dan ook een uitstekend pakket, vooral voor de opmaak van teksten over wiskunde en informatica. Dat het daarbuiten nog weinig toepassing vindt, is vooral te wijten aan het feit dat er een soort 'programmeurs-denken' voor nodig is om de tekst in te voeren. Voor de doorsneeschrijver van informatica-teksten is dat echter geen probleem, integendeel.

Een zwak punt van LaTeX is echter de mogelijkheid om plaatjes in een tekst op te nemen. Deze is zo beperkt, dat zelfs de meest doorgewinterde LaTeX-gebruiker in zwakkere momenten denkt aan de aanschaf van een Apple of PC met 'desktop-publishing' software. Dat is echter helemaal niet nodig; LaTeX biedt genoeg mogelijkheden om plaatjes in de tekst op te nemen die met andere programma's zijn gemaakt.

In dit artikel wordt eerst aandacht besteed aan de (beperkte) mogelijkheden die LaTeX zelf biedt voor het maken van plaatjes. Soms zijn die voldoende, en dan is het wel zo gemakkelijk om die te gebruiken. Daarna wordt de rol van PostScript besproken bij het gebruik van plaatjes, al of niet in combinatie met LaTeX. Vervolgens wordt een vijftal tekenprogramma's besproken waarmee interactief ingewikkeldere figuren gemaakt kunnen worden. Tenslotte wordt aangegeven hoe 'bitmaps' (letterlijke afbeeldingen van het scherm van een computer) als plaatje behandeld kunnen worden. Dit laatste is vooral van belang bij het schrijven van handleidingen voor programma's. Deze versie van het artikel is aangepast voor LaTeX2$_\varepsilon$. N.B. Een aantal van de behandelde programma's en technieken is specifiek voor Unix.

1 LaTeX

1.1 Figuren in LaTeX

Onder een figuur (*figure*) wordt in LaTeX een stuk van een tekst verstaan die niet op een vaste plaats hoeft te staan. Dat kan een plaatje zijn, maar ook bijvoorbeeld een listing van een programma, of een speciaal stuk tekst. Zo'n figuur kan een onderschrift hebben, en wordt door LaTeX automatisch genummerd. De figuur wordt door LaTeX op een plek gezet waar voldoende ruimte over is. Als een figuur meer ruimte kost dan de resterende ruimte op een pagina, dan wordt de figuur op de volgende pagina gezet.

Een figuur kan gemaakt worden met een `figure`-environment. Alle tekst die tussen `\begin{figure}` en `\end{figure}` staat behoort tot de figuur. In die tekst kan met een aanroep van `\caption` worden aangegeven wat het onderschrift (of bovenschrift) van de figuur is. De makkelijkste manier om een plaatje te maken is wel, om in een `figure`-environment simpelweg ruimte te reserveren door middel van een aanroep van `\vspace`. Een voorbeeld daarvan is (eerst de source-tekst, daaronder het resultaat):

```
\begin{figure}
\vspace{15mm}
\caption{Functioneel wit}
\end{figure}
```

Figure 1: *Functioneel wit*

De tekst 'Figure 1:' wordt door LaTeX zelf toegevoegd. In een Nederlandse tekst is het natuurlijk niet mooi om figuren aan te duiden met Engelse woorden. Een oplossing daarvoor is het `babel` package met de `dutch` optie op te nemen, bijvoorbeeld:

```
\usepackage[dutch]{babel}
```

Ook woorden als 'Contents' worden dan vervangen door 'Inhoudsopgave'. Bovendien worden bij gebruik van deze

optie de Nederlandse afbreekregels gebruikt in plaats van de Engelse en krijgt het " teken een andere betekenis.

Zoals gezegd wordt de nummering van de figuren door LaTeX zelf uitgevoerd. Als je vanuit de tekst daarnaar wilt verwijzen, kun je het figuur een symbolische naam geven met `\label` (na het `\caption` commando). Later kun je dan naar de figuur verwijzen met een aanroep van `\ref`. Bijvoorbeeld:

```
\begin{figure}
\vspace{15mm}
\caption{Mooi plaatje}
\label{fig.mooi}
\end{figure}
In figuur~\ref{fig.mooi} staat
een mooi plaatje.
```

Figuur 3: *Mooi plaatje* In figuur 3 staat een mooi plaatje.

In plaats van met `\vspace` gegenereerde witruimte mag er in de figuur natuurlijk ook iets anders staan. Een figuur kan bijvoorbeeld goed gebruikt worden om een listing weer te geven die niet op een specifieke plaats in de tekst hoeft te staan:

```
\begin{figure}
\small
\begin{verbatim}
    int fac(int n)
    {   return (n==0?1:fac(n-1))
    }
\end{verbatim}
\caption{De faculteit-functie}
\end{figure}
```

```
int fac(int n)
{   return (n==0?1:fac(n-1))
}
```

Figuur 7: *De faculteit-functie*

Met het `\parpic` commando uit het `picins` package is het mogelijk om plaatjes op te nemen die niet de hele breedte van het papier beslaan. Er zijn wel enige beperkingen aan dit commando: zo gaat bijv. het gebruik van een `verbatim` omgeving rechts naast de figuur niet goed, en moet je ook zelf opletten dat de figuur niet op een pagina-overgang terecht komt.

Figuur 1: *Een* `picins` *figuur*

Het `picins` package gebruik je door vooraan in je document op te nemen:

```
\usepackage{picins}
```

Je kunt dan een plaatje invoegen in een alinea door bijvoorbeeld:

```
\piccaption{Een { picins} figuur}
\parpic(5cm,4cm)[s]{Hier de figuur}
```

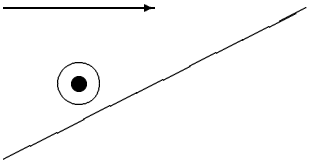
1.2 Plaatjes in LaTeX

Voor het maken van eenvoudige plaatjes, die bestaan uit lijnen, pijlen, cirkels en tekst is in LaTeX het environment `picture` beschikbaar. Een picture zal vaak optreden als de inhoud van een figure. Verplicht is dat echter niet; een picture kan bijvoorbeeld ook los in de tekst staan, bijvoorbeeld in een `quote` of een `center` environment. Zelfs kan een picture midden in een zin worden opgenomen (het is dan wel verstandig om de afmetingen klein te houden, zoals hier:).

Een picture environment begint met de tekst `\begin{picture}`. Direct daarachter moet, tussen ronde haakjes, worden aangegeven wat de breedte en de hoogte van het plaatje is. Alle afmetingen worden gegeven in veelvouden van `\unitlength`. Default is dat 1 punt (ongeveer $\frac{1}{3}$ mm), maar met `setlength` kun je de lengte-eenheid veranderen.

In een picture environment staan gewoonlijk alleen aanroepen van de macro `\put`. Direct achter `\put` staan de coördinaten waar iets neergezet moet worden, daarachter staat tussen accolades wàt er neergezet moet worden. Dat kan bijvoorbeeld een lijn of een cirkel zijn. Een voorbeeld van de opbouw van een picture volgt hieronder.

```
\setlength{\unitlength}{1mm}
\begin{picture}(40,20)
  \put(10,10){\circle{6}}
  \put(10,10){\circle*{2}}
  \put(0,0){\line(1,2){40}}
  \put(0,20){\vector(1,0){20}}
\end{picture}
```



De volgende objecten kunnen optreden als parameter van `\put`:

- `circle{`*diam*`}` — een cirkel met gegeven diameter wordt neergezet met het middelpunt op de plaats die achter `\put` is gespecificeerd (maximum diameter is 40 punt).
- `circle*{`*diam*`}` — de cirkel wordt zwart opgevuld (maximum diameter is 15 punt).
- `line(`*x*`,`*y*`){`*breedte*`}` — een lijn met (x, y) als richtingsvector met een gegeven breedte (als de lijn vertikaal staat wordt niet de breedte maar de hoogte opgegeven).
- `vector(`*x*`,`*y*`){`*breedte*`}` — een pijl in de gegeven richting met de gegeven breedte.

- `oval(x,y)[`*deel*`]` — een ovaal dat past in een rechthoek met gegeven afmetingen. De parameter *deel* bepaalt (indien aanwezig) welk deel van het ovaal getekend moet worden: zet hier één (voor een half ovaal) of twee (voor een kwart ovaal) van de letters `t` (top), `b` (bottom), `l` (left) of `r` (right).
- `makebox(x,y)[`*positie*`]{`*tekst*`}` — de gegeven tekst wordt in een onzichtbare rechthoek met gegeven afmetingen gezet, waarbij de *positie* aangeeft waar de tekst wordt gezet: `l` (tegen de linkerkant), `r` (tegen de rechterkant), `t` (tegen de bovenkant), `b` (tegen de onderkant), `rb` (tegen de rechteronderhoek), enz.
- `framebox(x,y)[` *positie*`]{`*tekst*`}` — als `makebox`, alleen is het rechthoekige kader nu ook zichtbaar.

Er zijn nogal wat beperkingen aan het picture-environment[1]. De schuinte van lijnen is bijvoorbeeld aan voorwaarden gebonden: de x en y waarmee de richtingsvector van een lijn wordt gespecificeerd moeten tussen -6 en 6 liggen. Voor pijlen (`vectors`) is dit zelf tussen -4 en 4. Bovendien zijn, behalve de kwartcirkels die door `oval` worden geproduceerd, ronde lijnen niet mogelijk. Tenslotte zijn alle tekeningen *lijn*tekeningen; het is niet mogelijk om arceringen aan te brengen.

Een belangrijk voordeel van het picture-environment is dat het plaatje in de tekst kan worden opgenomen, en gegarandeerd overal kan worden uitgeprint waar men LaTeX verstaat. Ook zijn alle teksten in het plaatje netjes in hetzelfde lettertype als de rest van de tekst, en kunnen bijvoorbeeld formules worden opgenomen in het plaatje.

1.3 Macropakketten voor LaTeX

Om het maken van LaTeX-pictures wat eenvoudiger te maken zijn er macropakketten beschikbaar die extra commando's bieden om plaatjes te maken. De naam van zo'n macropakket moet worden opgenomen tussen de accolades in het `\usepackage` commando.

De volgende macro-pakketten kunnen handig zijn:

epic (een afkorting van 'extended picture'): hiermee kunnen eenvoudig allerlei soorten stippellijnen worden gemaakt. Bovendien kan een lijnfiguur worden gespecificeerd door de achtereenvolgende hoekpunten op te sommen, zonder voor ieder lijnstuk apart de richtingsvector te hoeven berekenen. De beperking dat de richtingscoëfficiënt van een lijn een simpele breuk moet zijn blijft; specificeer je een lijn die dat niet heeft, dan wordt deze benaderd met korte lijntjes die dat wel hebben.

eepic Dit is uitbreiding van `epic` waarbij de beperkingen opgeheven zijn.

bezier dit (LaTeX2.09) macropakket biedt een macro `\bezier` waarmee kromme lijnen getekend kunnen worden (Bézier-curves, om precies te zijn). Deze curves worden benaderd door een heleboel punten naast elkaar te zetten, dus echt mooi wordt het niet. Bovendien mogen de lijnen niet te lang zijn, want bij meer dan 1000 punten gaat TeX out of memory. Voor een enkel krom lijntje is het pakket echter wel bruikbaar. In LaTeX2ε is standaard het commando `\qbezier` beschikbaar.

xypic met dit pakket kunnen relatief eenvoudig het soort diagrammen getekend worden waar wiskundigen dol op zijn: formules met pijlen ertussen waarbij bijschriften staan.

Het voert te ver om deze pakketten hier uitgebreid te behandelen; er is meer over te lezen in de bijbehorende documentatiefiles.

1.4 Gebruik van dvi-files

De uitvoer van LaTeX is een `.dvi`-file. Zo'n file is 'device independent'; hij kan bijvoorbeeld bekeken worden op het scherm, of afgedrukt op de printer. Bovendien kan een dvi-file worden getransformeerd in een andere dvi-file.

De volgende programma's zijn beschikbaar om dvi-files te verwerken:

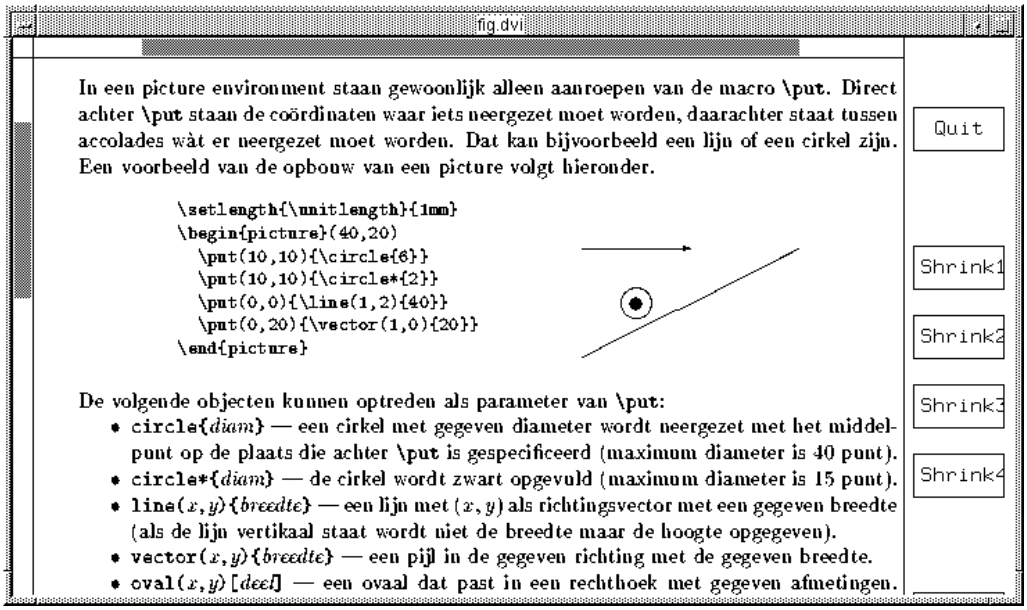
xdvi Hiermee kan een dvi-file op het scherm bekeken worden. Na het geven van de opdracht `xdvi aap.dvi` verschijnt een window waarin de eerste pagina van de file `aap.dvi` te zien is (zie figuur 2). Als de pagina te groot is om in het window te passen, kan er met scrollbars door gebladerd worden. Naast de pagina zijn een aantal buttons te zien waarmee door de tekst gebladerd kan worden. Blader-commando's kunnen ook via het toetsenbord gegeven worden:

`d`	scroll down
`u`	scroll up
`n`	next page
`p`	previous page
getal `g`	go to page *getal*
`q`	quit

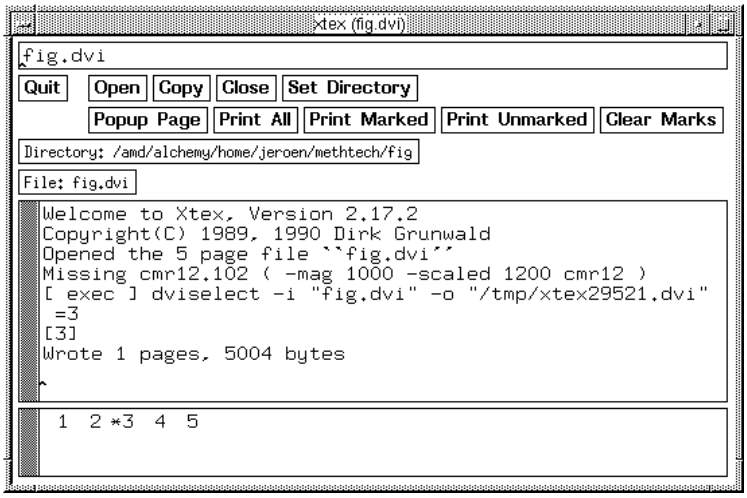
xtex Dit is een andere previewer voor dvi-files. Het programma maakt twee nieuwe windows: een window waarin de pagina verschijnt, en een apart controlewindow (zie figuur 3). Handig van deze previewer is vooral de mogelijkheid om pagina's te markeren. Dat kan door op het betreffende paginanummer (onderin het controlewindow) te klikken, of door naar de pagina te bladeren en op `m` te drukken. In de lijst met paginanummers verschijnt een sterretje voor de geselecteerde pagina(s). De geselecteerde pagina's kunnen geprint worden door op de button 'print marked' te drukken. Er zijn geen buttons om te bladeren, maar het programma kent dezelfde toetsenbord-commando's als `xdvi`.

dvips Hiermee kan een dvi-file worden omgezet in PostScript (zie hoofdstuk 2). PostScript-files kunnen op een groot aantal (laser)printers geprint worden. Deze conversie kan handig zijn als een file opgestuurd moet worden naar een plaats waar LaTeX niet is ingeburgerd, en waar men dus geen dvi-files kan verwerken.

[1] In de toekomst zal er een LaTeX2ε package `pict2e` beschikbaar komen waarmee deze beperkingen opgeheven kunnen worden.



Figuur 2: *xdvi, een previewer voor dvi-files*



Figuur 3: *xtex, controlewindow van een alternatieve dvi-previewer*

dviselect Met dit programma kunnen pagina's uit een dvi-file worden geselecteerd en in een andere dvi-file opgeslagen.

dvidvi Hiermee kunnen pagina's worden geselecteerd, in een andere volgorde gezet, of worden samengevoegd tot grotere pagina's. Dit commando kan bijvoorbeeld worden gebruikt om zonder knippen en plakken een A5-boekje te genereren. Zie de manual.

2 PostScript

2.1 Eigenschappen van PostScript

PostScript is een taal waarmee de opmaak van een pagina beschreven kan worden. PostScript wordt veel gebruikt om printers te besturen, en is daarom ook een geliefd formaat voor tekenprogramma's. Behalve afbeeldingen kan in een PostScript-plaatje ook tekst worden opgenomen.

PostScript is veel flexibeler dan LaTeX voor het maken van plaatjes. Zo zijn er bijvoorbeeld de volgende dingen mogelijk:

- lijnen in alle richtingen
- gebogen lijnen
- geroteerde tekst
- gearceerde oppervlaktes
- *bitmaps* (1-op-1-afbeelding van een grafisch scherm)

PostScript heeft echter minder kennis van tekstopmaak dan LaTeX. Het is bijvoorbeeld niet mogelijk om automatisch woorden over regels te verdelen, en regels over pagina's. Voor lange stukken tekst is LaTeX dus geschikter, voor ingewikkelde figuren is PostScript geschikter.

Een PostScript-file is een gewone tekstfile, en kan dus worden aangemaakt en bekeken met de tekstverwerker. Hoewel dit in principe mogelijk is, is het meer gebruikelijk om PostScript-files te laten *genereren* door andere programma's.

PostScript is in feite een programmeertaal, waarbij het executeren van een programma tot gevolg heeft dat de layout van een pagina wordt bepaald. Het is niet nodig om te weten hoe deze taal is opgebouwd om PostScript-verwerkende programma's te kunnen gebruiken. Kennis van de taal is alleen nodig als je met de hand (of liever gezegd: met de editor) PostScript-programma's wilt schrijven. Ook is deze kennis natuurlijk nodig als je zelf programma's wilt schrijven die PostScript genereren. Een inleiding in de taal volgt in paragraaf 2.4.

2.2 Inclusie in LaTeX

PostScript-plaatjes kunnen in een LaTeX-document worden opgenomen. Het nadeel daarvan is dat de resulterende dvi-file niet echt device-independent meer is: hij kan alleen nog maar worden gebruikt op devices die ook PostScript kennen. Dit hoeft geen groot bezwaar te zijn, omdat dvi-files vaak worden afgedrukt op een PostScript-laserprinter. De dvi-file wordt daartoe met het programma `dvips` omgezet in PostScript, en de toe te voegen PostScript-illustratie kan daarbij naadloost worden ingelast. De zogenaamde eps (zie beneden) plaatjes kunnen met xdvi gepreviewd worden, voor andere pstscript plaatjes kan dit wel eens een probleem zijn.

Een probleem is dat ten tijde van de conversie TeX → dvi bekend moet zijn hoeveel ruimte er opengelaten moet worden. Je moet in het algemeen een PostScript-tekst geheel interpreteren om te weten hoe groot een tekening is. Zo veel kennis van PostScript heeft LaTeX niet, dus moet hier iets anders op gevonden worden.

Net als in veel andere talen kan in een PostScript-programma commentaar worden opgenomen. Een regel commentaar begint met een procent-teken en loopt tot het einde van de regel. Een veelgebruikte conventie is om een speciale commentaar-regel in een PostScript-tekst op te nemen:

```
%%BoundingBox: 75 435 523 743
```

Hiermee wordt gespecificeerd wat de 'begrenzende rechthoek' van de tekening is. De eerste twee getallen zijn de coordinaten van de linker-onderhoek de andere twee getallen de coordinaten van de rechter-bovenhoek (in de eenheid *point*, dit is ongeveer $\frac{1}{3}$mm). Veel programma's die PostScript genereren nemen daarin een bounding-box commentaarregel in op. Zelf kan je natuurlijk ook zo'n regel toevoegen aan een PostScript-tekst.

PostScript-plaatjes waarin een bounding box wordt gespecificeerd heten *encapsulated PostScript*, afgekort *eps*. Aan een LaTeX-tekst kun je encapsulated-PostScript plaatjes toevoegen. Om de afmeting te bepalen, hoeft LaTeX alleen de regel waarin `%%Bounding Box` staat op te zoeken.

Het toevoegen van een PostScript-plaatje aan een LaTeX-tekst gebeurt als volgt. In het LaTeX-document moet je het package `epsf` ('encapsulated PostScript figure') `epsf` gebruiken, dus bijvoorbeeld

```
\usepackage{epsf}
```

Het plaatje kan dan in een LaTeX-box geplaatst worden met de macro-aanroep `\epsfbox{`*filenaam*`}`. dit kan (voor hele kleine plaatjes) midden in de regel gebeuren, maar meestal zal deze aanroep in een `center` of `quote` environment staan. In dat geval moet er ook nog `\leavevmode` voor staan, om TeXnische redenen. Dus:

```
\begin{quote}
\leavevmode\epsfbox{plaatje.ps}
\end{quote}
```

Als in de PostScript-file onverhoopt geen Bounding box staat gedefinieerd, kun je die als optionele parameter meegeven aan `\epsfbox`, bijvoorbeeld

```
\epsfbox[0 0 480 220]{file.ps}
```

Het plaatje krijgt z'n 'natuurlijke grootte'. Je kunt de grootte beïnvloeden door de macro `\epsfsize` te herdefiniëren. Deze macro heeft twee parameters: de natuurlijke x-afmeting en de natuurlijke y-afmeting. Hij moet opleveren wat de gewenste x-afmeting in het eindresultaat is. De y-afmeting wordt vanzelf meegeschaald. Bijvoorbeeld om een plaatje te halveren:

```
\renewcommand{\epsfsize}[2]{0.5#1}
```

Of om een plaatje kleiner te maken als het te breed is:

```
\renewcommand{\epsfsize}[2]{%
          \ifnum#1>\hsize\hsize\else#1\fi}
```

In LaTeX2$_\varepsilon$ kan je het standaard package `graphics` nemen en dan `\includegraphics{`*filenaam*`}` gebruiken. Als optionele argumenten kunnen opgegeven worden `[`x_1, y_1`]` en `[`x_2, y_2`]` voor resp. de coordinaten van de linkeronderhoek en de rechterbovenhoek (of alleen de rechterbovenhoek als de linkeronderhoek `[0,0]` is).

2.3 PostScript-verwerkende programma's

De volgende programma's hebben PostScript als invoer en/of als uitvoer:

ghostview dit is een programma waarmee PostScript-programma's op het scherm kunnen worden bekeken. Als het een document met meerdere pagina's betreft, kun je daardoorheen bladeren door in het window waar je het programma aanroept op return te drukken. Dit programma is ideaal om PostScript-programma's te debuggen zonder al te veel papier te gebruiken. Het programma kan ook gebruikt worden om te controleren of PostScript-plaatjes op de goede plaats in LaTeX-files terecht zijn gekomen als de previewer niet in staat is dit te laten zien. Daartoe moet je de dvi-file eerst met `dvips` converteren naar een ps-file, die je vervolgens met `ghostview` kunt bekijken.

a2ps dit programma maakt van een gewone tekstfile een PostScript-file, waarbij de tekst in twee kolommen is ingedeeld, de pagina's van een header en een nummering worden voorzien, enzovoort. Je hebt dit programma meestal niet nodig omdat het standaard

door `laser` wordt gebruikt, maar het kan expliciet worden aangeroepen om extra opties mee te geven (bijvoorbeeld om de header-regel weg te laten, de regels te nummeren, of om hele lange regels weer te geven).

dvips dit programma converteert een dvi-file naar PostScript. Het wordt automatisch gebruikt bij het laseren van een dvi-file.

pstops met dit programma kan een PostScript-programma worden geconverteerd in een ander PostScript-programma. Er kunnen bijvoorbeeld pagina's mee worden geselecteerd of worden samengevoegd. Een veel gebruikte optie is om twee PostScript-pagina's naast elkaar op één vel te zetten. De wat kryptische commandoregel hiervoor is

```
pstops '2:0L@.7(21cm,0)+1L@.7
  (21cm,14.85cm)' input.ps output.ps
```

Om de output van `dvips` te verwerken moeten de `0` en de `14.85` worden omgewisseld.

idraw dit is een programma om interactief PostScript-tekeningen te maken. Het programma kan zijn eigen uitvoer opnieuw inlezen, maar kan niet gebruikt worden om willekeurige PostScript-plaatjes aan te passen (zie paragraaf 3.1).

xgrabsc met dit programma kan een 'snapshot' van (een deel van) het scherm gemaakt worden, o.a. in PostScript-formaat (zie paragraaf 4.3).

2.4 PostScript als taal

Een PostScript-programma is opgebouwd uit operatoren die worden toegepast op parameters. Er is een groot aantal operatoren standaard aanwezig, en het is ook mogelijk om zelf nieuwe operatoren te definiëren. Omdat ook keuze en herhaling/recursie mogelijk zijn, is PostScript een echte programmeertaal.

Het voeden van parameters aan operatoren gebeurt via een stack. Bij het executeren van een PostScript-programma wordt elk ding dat geen operator is op de stack gezet. Een operator plukt de parameters die hij nodig heeft van de stack. Daardoor moeten alle expressies in *postfix*-notatie worden geschreven: eerst de parameters (die op de stack worden gezet), en dan de operator (die ze er weer af haalt).

Naast de stack wordt een belangrijke rol gespeeld door de *current page*, waarop de tekening wordt opgebouwd, het *current path*, bestaande uit een aantal lijnstukken (die niet noodzakelijkerwijs aan elkaar vast hoeven te zitten), en het *current point*. Het gebruik al deze zaken wordt gedemonstreerd in het volgende eenvoudige, doch complete PostScript-programma:

```
100 200 moveto
150 230 lineto
150 250 lineto
stroke
showpage
```

De operator `moveto` pakt twee parameters van de stack, en zorgt ervoor dat het *current point* hier komt te staan. De operator `lineto` voegt een lijnstuk van het huidige punt naar het nieuwe gespecificeerde punt toe aan het *path*. Bovendien wordt dit nieuwe punt het huidige punt. De operator `stroke` zorgt ervoor dat het *path* als lijn-tekening op de huidige pagina wordt gezet. De operator `showpage` tenslotte laat de huidige pagina zien (op papier of scherm), en begint met een schone lei.

Behalve de genoemde operatoren die een grafische aktie uitvoeren, zijn er ook operatoren voor het meer conventionele rekenwerk. Ook deze operatoren worden in postfix-notatie opgeschreven. De expressie $1 + 2 * 3 + 4$ ziet er daarom als volgt uit:

```
1 2 3 mul add 4 add
```

Het leren van PostScript bestaat voor een deel uit het handig worden in het gebruik van deze stack-gebaseerde notatie. Verder is het natuurlijk belangrijk om te weten welke operatoren er beschikbaar zijn.

Hieronder volgt een overzicht van een aantal operatoren. De lijst is niet compleet, maar geeft wel een idee van wat er zoal mogelijk is. In de tabellen wordt voor de operator aangegeven wat de parameters zijn, en erachter wat de operator op de stack achterlaat (of – als de parameter 'verbruikt' wordt).

- Operatoren om de stack te manipuleren

x	`pop`	–	verwijder bovenste element
$x\ y$	`exch`	$y\ x$	verwissel bovenste twee elementen
x	`dup`	$x\ x$	verdubbel bovenste element
$x_1 \ldots x_n\ n$	`copy`	$x_1 \ldots x_n\ x_1 \ldots x_n$	kopieer elementen

- Rekenkundige operatoren

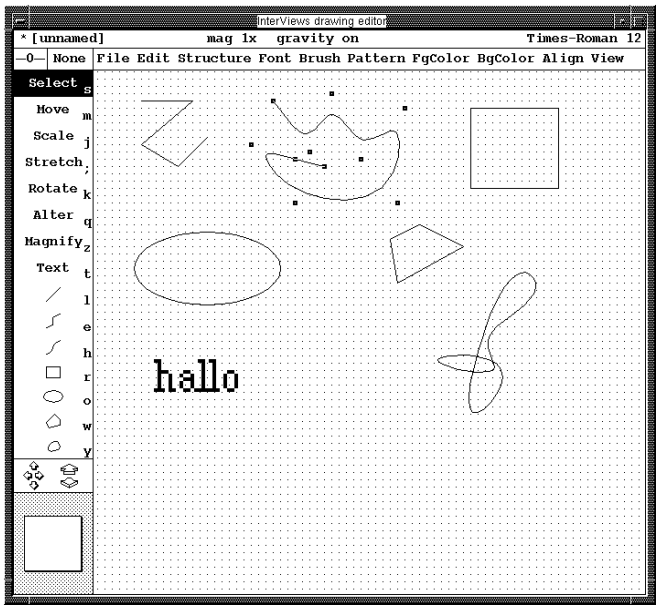
$x\ y$	`add`	z	optellen
$x\ y$	`sub`	z	aftrekken
$x\ y$	`mul`	z	vermenigvuldigen
$x\ y$	`div`	z	delen
$x\ y$	`idiv`	z	integer delen
$x\ y$	`mod`	z	rest bij deling
x	`abs`	z	absolute waarde
x	`neg`	z	omgekeerde
x	`sqrt`	z	wortel
x	`sin`	z	sinus (x in graden)
	...		

- Operatoren om een pad op te bouwen

–	`newpath`	–	begin een nieuw pad
–	`currentpoint`	$x\ y$	zet coördinaten huidige punt op stack
$x\ y$	`moveto`	–	verplaats 'huidige punt'
dx dy	`rmoveto`	–	verplaats 'huidige punt' relatief
$x\ y$	`lineto`	–	trek lijn
dx dy	`rlineto`	–	trek lijn relatief
–	`closepath`	–	maak pad gesloten
$x\ y\ r\ a\ b$	`arc`	–	cirkelboog naar (x, y), straal r tussen hoek a en b
$x_1\ y_1\ x_2\ y_2\ x_3\ y_3$	`curveto`	–	Bézier curve naar p_3 met p_1 en p_2 als controlepunt

- Operatoren om een pad te gebruiken

–	`stroke`	–	trek lijnen langs het pad
–	`fill`	–	kleur het pad in
x	`setlinewidth`	–	verander de dikte van de lijnen
x	`setgray`	–	verander de grijstint van lijnen en vlakken



Figuur 4: *idraw, een interactief tekenprogramma*

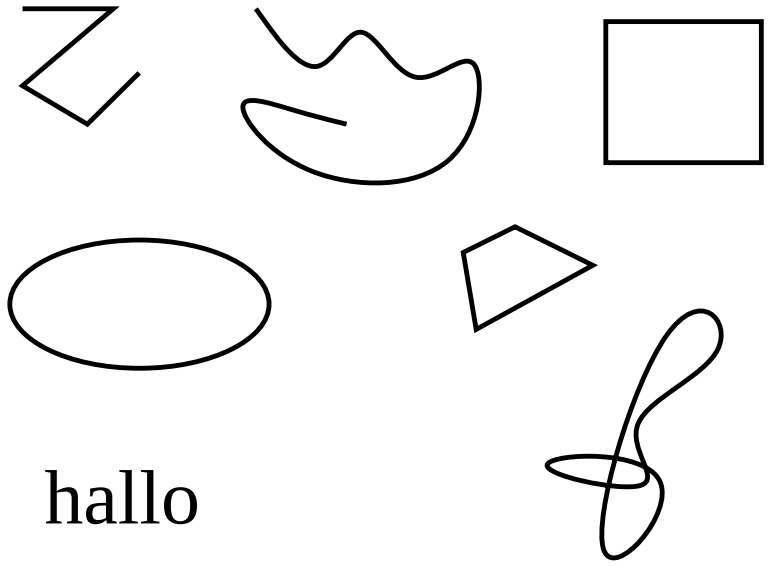
3 Tekenprogramma's

3.1 idraw: Een tekenprogramma

De makkelijkste manier om een plaatje te maken is om hiervoor een interactief tekenprogramma te gebruiken. Een van de mogelijkheden is het programma `idraw`. Na een tijdje gebruik kan het scherm er uitzien als in figuur 4. Het grootste deel van het window wordt gebruikt om het plaatje weer te geven. Erboven staat en menubalk, links een 'itembox'. Linksonder staat een situatie-diagram, dat aangeeft welk deel van het plaatje 'in beeld' is. Door het witte vlak hierop te schuiven, wordt een ander deel van het plaatje zichtbaar. Met de vier pijltjes boven het situatiediagram kan ook worden geschoven. De twee grote pijltjes worden gebruikt om in- en uit te zoomen op een detail van de tekening.

Door op een item in de itembox te klikken (of door een letter op het toetsenbord te tikken) kunnen verschillende soorten objecten getekend worden: rechte lijn, stuksgewijs rechte lijnen, curve, rechthoek, ovaal, gesloten polygon, gesloten curve, of tekst. Bij de figuren die uit meerdere delen bestaan (zoals curves) wordt het laatste punt gespecificeerd door op de rechter muisknop te klikken.

Aan het laatst getekende object zitten zwarte vierkantjes. Een object kan verschoven worden door 'move' te selecteren, en hem met de muis bij zo'n vierkantje op te pakken. Met 'scale' kan een figuur groter of kleiner gemaakt worden, met 'stretch' worden uitgerekt in één richting. 'Rotate' draait een figuur, en met 'alter' kunnen (bijvoorbeeld in een curve) nog hoekpunten worden verplaatst. Transformaties kunnen op meerdere objecten tegelijk werken (bijvoorbeeld om ze allemaal evenveel te vergroten) door ze



Figuur 5: *uitvoer van idraw op de printer*

aan te klikken terwijl de shift-toets is ingdrukt. Nog handiger is het om meerdere geselecteerde objecten te groeperen met het commando ‘group’ in het ‘structure’ menu, en ze daarna als één geheel te behandelen. Daarna kunnen ze desgewenst weer worden losgmaakt met ‘ungroup’.

De tekening is een vector-tekening, dat wil zeggen dat alle objecten opgeslagen worden in de vorm van hoekpunten en dergelijke. Objecten die elkaar overlappen kunnen daarom ook weer worden gescheiden. De afbeelding op het scherm is een ruwe benadering van het uiteindelijke resultaat: dit kan veel mooier zijn (vooral wat betreft teksten en curves). Het resultaat, zoals dat op de printer wordt afgedrukt, is te zien in figuur 5.

Het stippenpatroon op de achtergrond wordt niet afgedrukt. Het is bedoeld als hulpmiddel bij het tekenen. Met het ‘align’-menu kan worden ingesteld dat alle hoekpunten moeten samenvallen met een rasterpunt (‘align to grid’). Daarmee kun je nette tekeningen maken. Met de andere menu’s kunnen dingen worden ingesteld als lijndikte, invulkleur van gesloten objecten, lettertype van teksten, en pijlkoppen aan lijnuiteinden

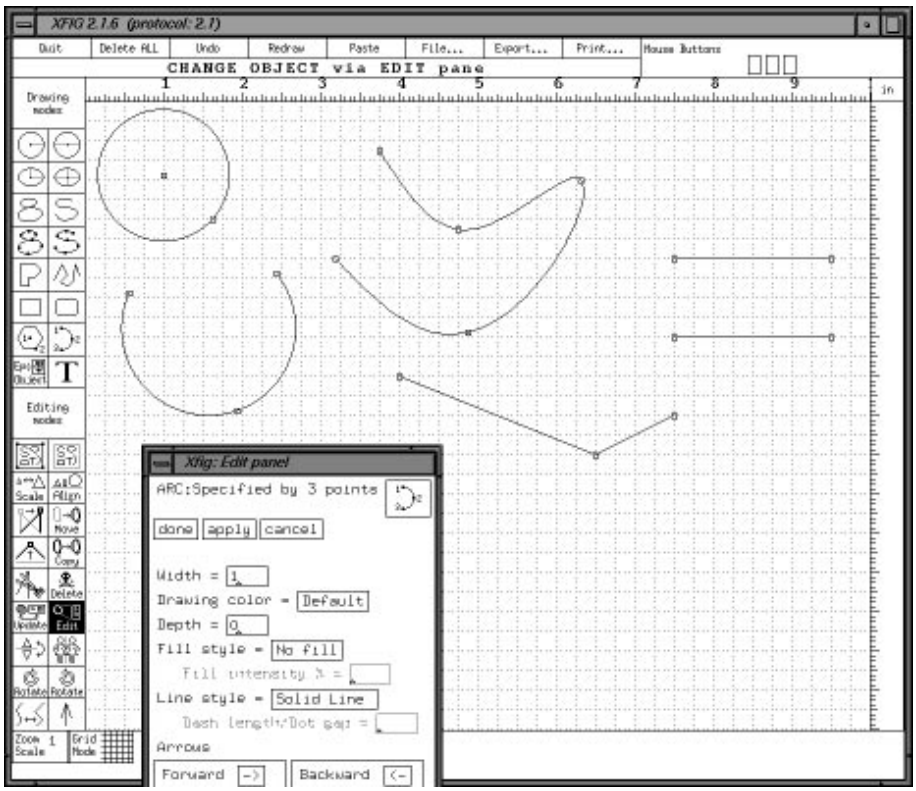
3.2 xfig: nog een tekenprogramma

Een ander tekenprogramma is `xfig`. Zoals blijkt uit de ikoontjes in figuur 6 heeft dit programma ongeveer dezelfde mogelijkheden als `idraw`.

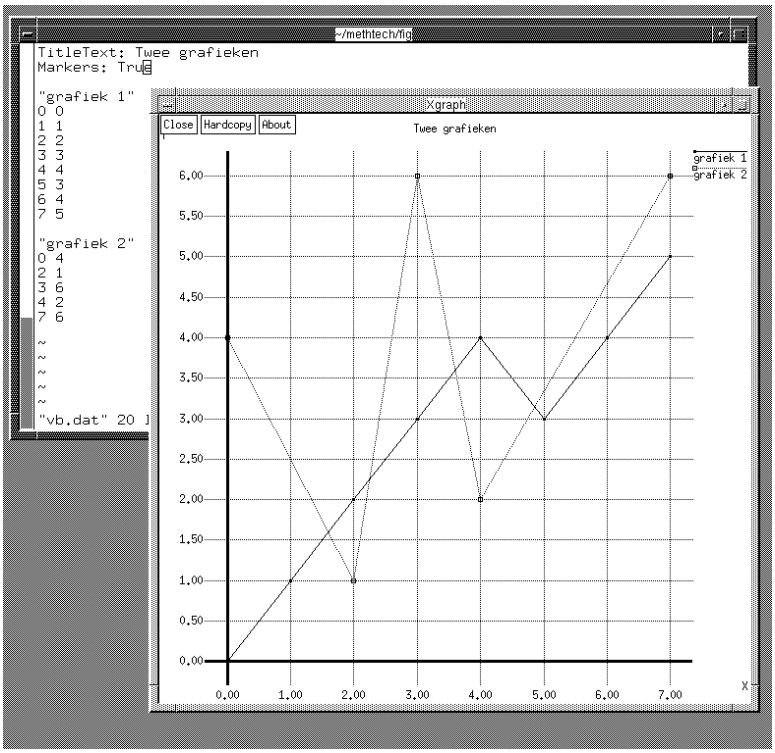
Een extra eigenschap van `xfig` is dat het programma een LaTeX picture environment kan genereren. Er mogen dan natuurlijk geen dingen gebruikt worden die in zo’n environment onmogelijk zijn, zoals gebogen lijnen. Er zijn aparte ikoontjes beschikbaar om je te helpen om alleen maar lijnen te tekenen met door LaTeX toegestane richtingscoëfficiënt. Bovendien heeft `xfig` de mogelijkheid om de tekst in een plaatje in een LaTeX lettertype af te drukken i.p.v. een PostScript lettertype. Dit geeft een betere uniformiteit in je document en ook de mogelijkheid om formules e.d. in de tekst op te nemen.

De plaatjes worden door het programma opgeslagen in een privé-formaat. Deze files kunnen worden geconverteerd naar PostScript en, binnen de genoemde beperkingen, naar LaTeX met de `export` optie.

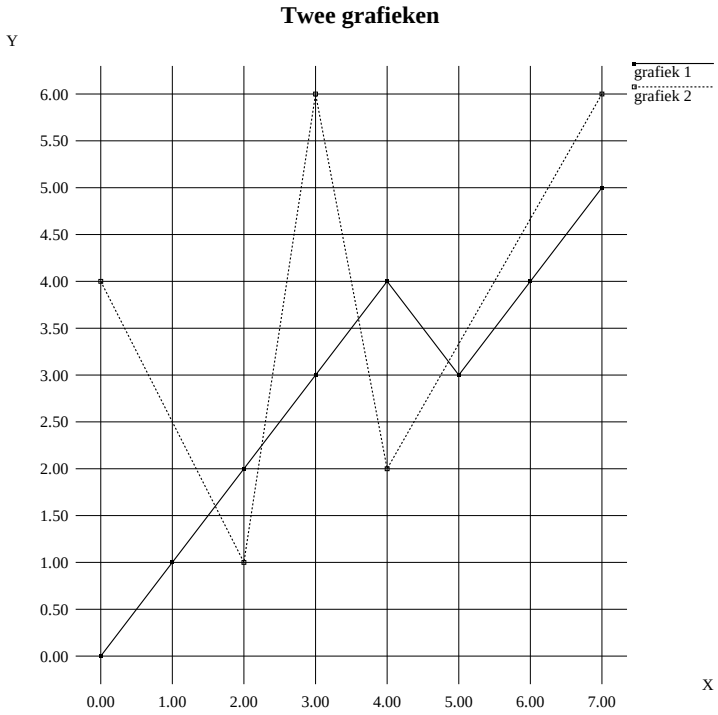
`xfig` kan ook eps plaatjes opnemen maar deze niet meer wijzigen. Wel kunnen elementen aan een tekening toegevoegd worden, waardoor eps plaatjes uit een andere bron geannoteerd kunnen worden.



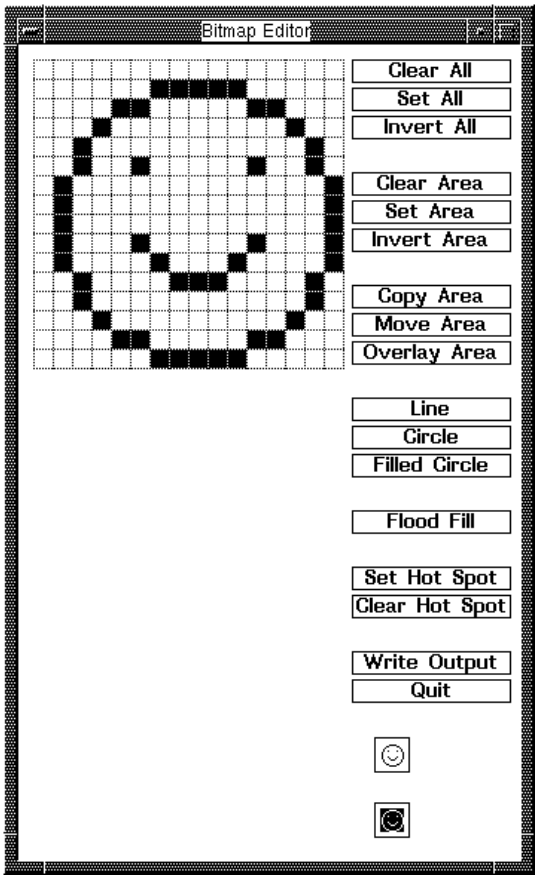
Figuur 6: *xfig, een tweede interactief tekenprogramma*



Figuur 7: *xgraph, grafieken maken van meetgegevens*



Figuur 8: *PostScript-uitvoer van xgraph*



Figuur 9: *bitmap, ontwerpen van kleine bitmaps*

3.3 Ipe: Integrated Picture Environment

Ipe is een recent tekenprogramma dat de opmerkelijke eigenschap heeft dat het interne formaat waarin tekeningen opgeslagen worden zowel een geldige PostScript file als een geldig stuk LaTeX is. De ipe-files kunnen zonder meer in een LaTeX document worden opgenomen. Ipe splitst het tekstgedeelte van het plaatje van het grafische deel. Voor de tekst worden gewone LaTeX-commando's gebruikt, voor het grafische deel PostScript. Verder heeft ipe de mogelijkheid om bestaande PostScript files in te lezen en te bewerken.

3.4 xgraph: Grafieken maken

Voor het maken van grafieken, bijvoorbeeld voor de weergave van meetresultaten, zijn de interactieve tekenprogramma's niet altijd even handig. Hiervoor is een apart programma beschikbaar: `xgraph`. Invoer van dit programma is een tekstfile waarin de coördinaten van de te plotten punten staan, en enkele andere parameters. Schaalverdeling en bijschriften bij de assen worden automatisch verzorgd (zie figuur 7). Met de muis kun je nog inzoomen op een interessant detail.

Door op de knop 'hardcopy' te drukken kan een plaatje worden bewaard. Het uitvoer-formaat kan geschikt worden gemaakt voor `idraw`, bijvoorbeeld om daarmee nog wat details te veranderen. Ook kan direct PostScript worden gegenereerd. Om 'encapsulated' PostScript te krijgen, bijvoorbeeld om het plaatje op te nemen in een LaTeX-tekst, moet de optie 'include in document' gekozen worden (zie figuur 8).

3.5 Gnuplot

Gnuplot is ook een pakket om grafieken te maken. Als invoer van gnuplot wordt een file van coordinaten gebruikt, maar gnuplot kan ook de grafiek van een formule tekenen. Gnuplot is verkrijgbaar voor vele systemen en kan ook uitvoer genereren in verschillende formaten, o.a. LaTeX picture formaat en PostScript.

4 Bitmaps

4.1 Bitmap-formaten

De tot nu toe besproken tekenprogramma's zijn bedoeld om vector-tekeningen te maken. Het voordeel van dit soort tekeningen is dat ze later eenvoudig kunnen worden aangepast, en dat ze onafhankelijk zijn van de resolutie van het gebruikte apparaat. Zo blijken tekeningen op de printer vaak mooier te zijn dan ze op het scherm leken.

Als je een afbeelding wilt maken van het beeldscherm, is de meer aangewezen weg een *bitmap*. Elk puntje op het scherm verschijnt dan als één puntje in de afbeelding. In zo'n bitmap kunnen geen losse objecten meer worden onderscheiden: dingendie over elkaar heen getekend worden, zijn nooit meer te scheiden.

Er is een groot aantal formaten in gebruik om een bitmap te beschrijven. Sommige zijn alleen voor zwart-wit plaatjes, andere kunnen ook voor kleurenplaatjes gebruikt worden. Sommige slaan elke acht pixels op in één byte, andere passen slimme compressietechnieken toe om grote egale vlakken efficiënter op te slaan.

Het standaardformaat voor X-windows is het *xbm*-formaat ('X-bitmap'). Voordeel hiervan is dat het met een editor te bekijken is, nadeel is dat het erg veel geheugen kost: bijna 1 byte per pixel. Voor kleine plaatjes is dat echter niet zo erg.

Een compacter formaat om bitmaps op te slaan is *gif* ('graphics interchange format'). Op pc's is worden de formaten *pcx, bmp* en *img* veel gebruikt. Dit zijn alledrie gecomprimeerde formaten.

4.2 Bitmaps maken en gebruiken

Met het programma `bitmap` kunnen kleine bitmaps in xbm-formaat worden gemaakt en veranderd. Het gebruik van het programma wijst zichzelf; een impressie is te zien in figuur 9.

Als je probeert een grote bitmap met dit programma te editten, worden de vierkantjes zo klein dat je ze niet goed meer kunt aanklikken. Het programma is dan ook in eerste instantie bedoeld voor het ontwerpen van cursors en kleine patroontjes.

Een programma dat gebruik kan maken van zo'n bitmap is `xsetroot`. Hiermee wordt de achtergrond van het X-scherm veranderd. Dit wordt gedemonstreerd in figuur 10.

De neutrale toestand krijg je weer terug met

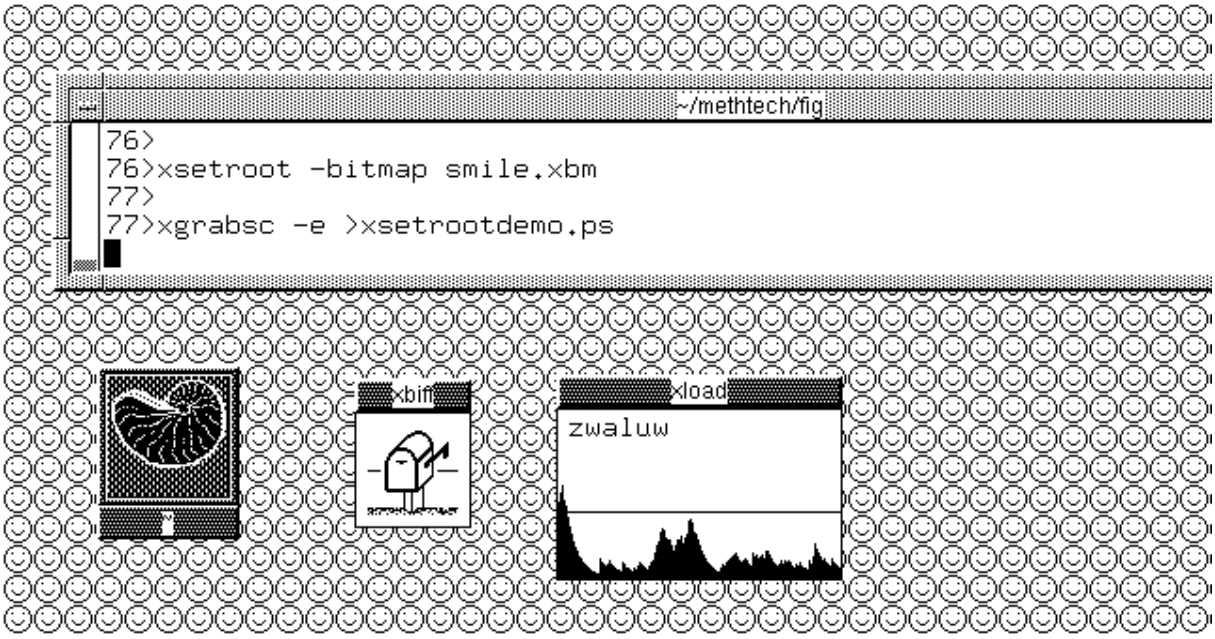
```
xsetroot -grey
```

Er is een conversieprogramma om van een xbm-file een tekstfile te maken waarin elk pixel met een sterretje of een spatie wordt weergegeven. Dit programma heet `bmtoa`. De omgekeerde transformatie wordt uitgevoerd door `atobm`.

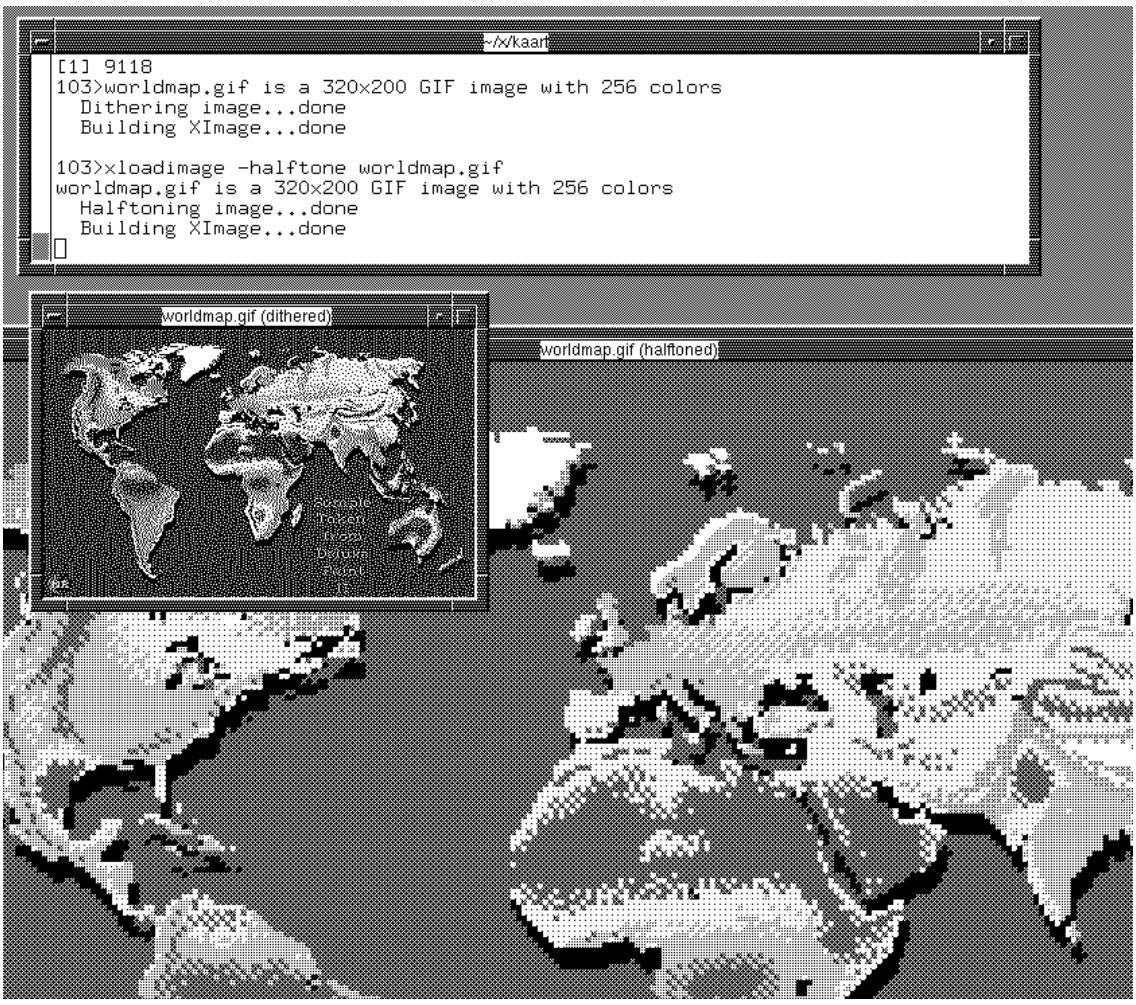
Bitmaps in alle genoemde formaten kunnen worden bekeken met de programma's `xloadimage` of `xv`. Het programma probeert (meestal met succes) zelf te raden wat het type is van de bitmap. Zwartwit plaatjes worden direct afgebeeld; kleurenplaatjes worden zonodig naar zwartwit geconverteerd. Daarbij zijn er twee mogelijkheden: 'dithering', waarbij het plaatje even groot blijft, en 'halftoning', waarbij het plaatje 4×4 keer zo groot wordt. Beide worden getoond in figuur 11.

4.3 Scherm-snapshots

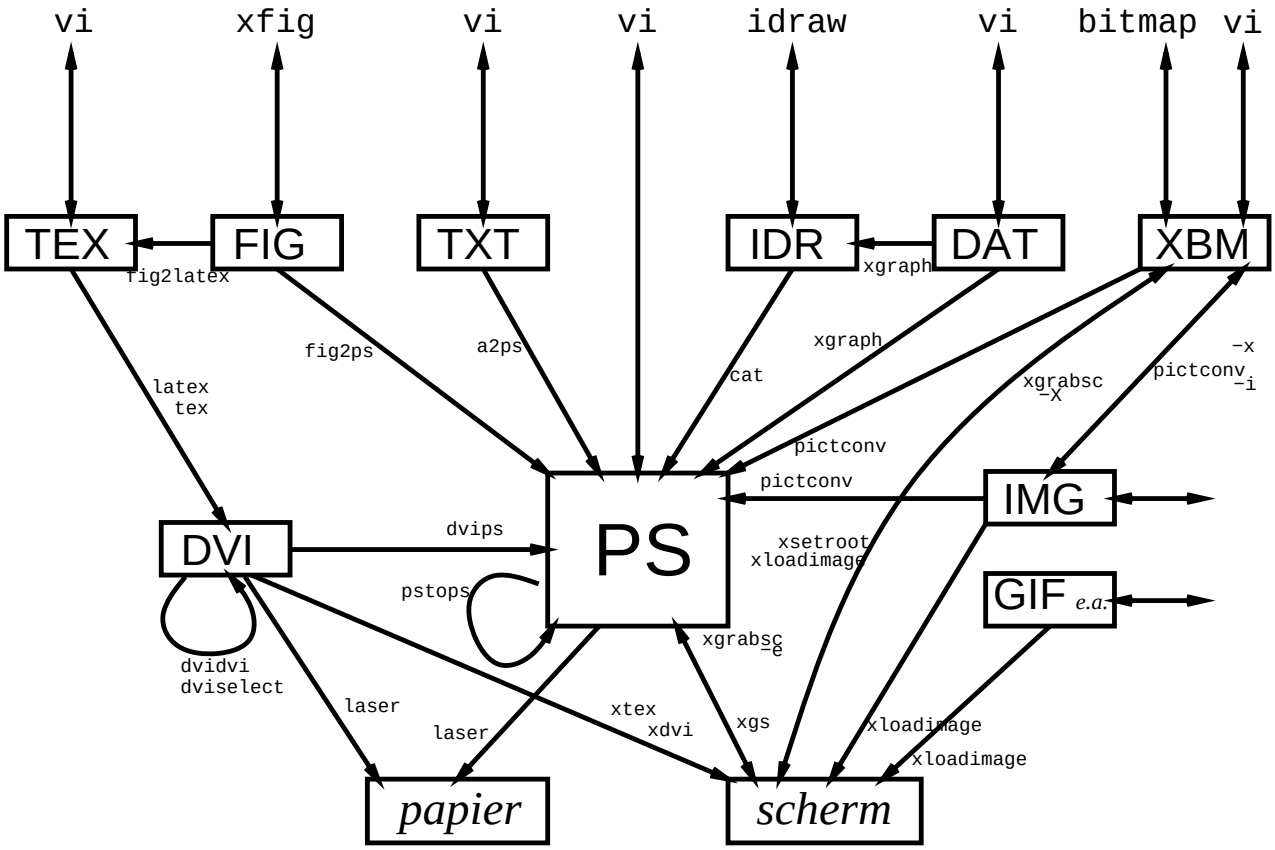
Vaak wil je in een handleiding van een programma een 'snapshot' van het scherm neerzetten. Met het programma `xgrabsc` kun je daartoe een bitmap maken van een deel van het scherm. Bijna alle afbeeldingen in deze tekst zijn met `xgrabsc` gemaakt. Dit commando kun je in een ander window geven. Met de muis kun je daarna een deel van het scherm uitknippen. Het programma kent opties voor de vorm van de output: `-eps` voor encapsulated PostScript, `-bm` voor xbm-formaat. Dat laaste wordt al snel erg groot.



Figuur 10: *xsetroot, instellen van de window-achtergrond*



Figuur 11: *xloadimage, met twee manieren om kleurenplaatjes ZW weer te geven*



Figuur 12: *Overzicht van conversieprogramma's*

Normaliter wordt het gewenste deel van het scherm aangegeven door met de muis een rechthoek te trekken. Het is ook mogelijk om een compleet window te tonen. Hiervoor is de optie `-click`. Vaak wil je voordat het snapshot genomen wordt nog iets prepareren aan het window, bijvoorbeeld het window activeren. Je kunt dan de `-s` optie geven:

```
xgrabsc -eps -click -s 5> scherm.ps
```

Voor afbeeldingen van kleurenschermen zijn er ook opties om de kleuren naar zwartwit te converteren.

In combinatie met `xloadimage` kan `xgrabsc` gebruikt worden om files uit allerlei grafische formaten om te zetten naar PostScript, om ze te kunnen opnemen in een LaTeX-tekst.
Een ander programma voor snapshots is `xwpick`. Dit kan verschillende uitvoerformaten leveren en geeft i.h.a. kleinere files als output.

5 Samenvatting

In figuur 12 wordt van (bijna) alle besproken conversieprogramma's aangegeven tussen welke filetypen ze werken.