LaTeX, le graphisme et PostScript

Denis ROEGEL

roegel@loria.fr

1997

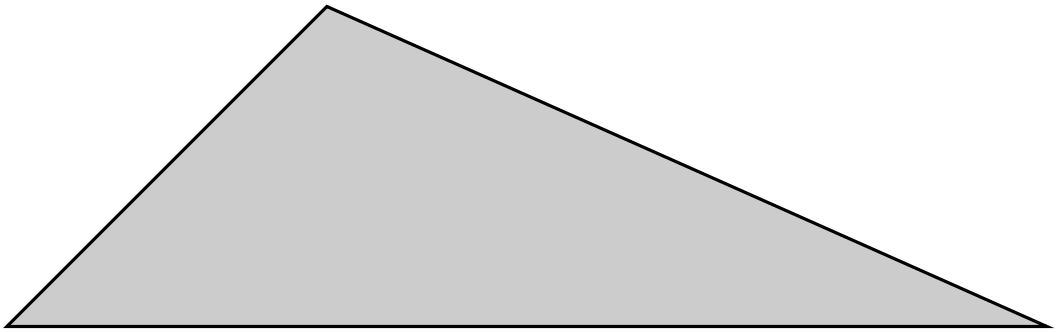
1. Généralités

1.1. Commande `\special` : mécanisme d'échappement

TeX ne comprend pas le PostScript et ce langage n'existait pas quand TeX a été conçu.

Pour que TeX soit le plus général possible, Knuth a introduit un mécanisme d'*échappement* qui permet d'écrire un texte quelconque dans le fichier `.dvi`, ce texte pouvant être ultérieurement interprété par des programmes lisant le format `.dvi`.

Si la commande `\special` commence par `"`, elle permet d'insérer du PostScript littéral dans un fichier `.dvi`. Par exemple,



est obtenu par la commande :

```
\vbox to 100bp{\vss % insertion directe en PostScript
\special{" newpath 0 0 moveto 100 100 lineto 325 0
lineto closepath gsave 0.8 setgray fill grestore stroke}}
```

Chaque *package* graphique qui veut étendre TEX va en général le faire via la commande `\special`.

Une commande `\special` n'a aucun effet sur le positionnement des caractères dans TEX. Si l'on souhaite avoir de la place pour le résultat de la commande `\special`, il faut réserver cette place dans le source TEX, par exemple avec une commande `\vbox` appropriée comme plus haut.

Les commandes insérées dans le fichier `.dvi` sont en général ignorées par la visionneuse `dvi` telle que `xdvi`. Par contre, ces commandes sont prises en compte par les *drivers* (tels `dvips`) qui les reconnaissent.

Au moment de la compilation, LATEX n'inclut pas la figure, mais seulement une commande qui permettra au *driver* de la charger.

La syntaxe à employer au sein d'une commande `\special` peut donc dépendre du *driver* et il est plus simple d'utiliser une interface convenable pour exprimer les actions des commandes `\special` de manière abstraite.

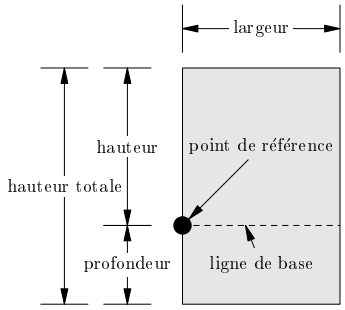
Des interfaces sont fournies par les *packages* `epsf`, `psfig`, `epsfig`, `graphics`, etc. L'interface standard en LATEX est celle du *package* `graphics` (ou de sa variante `graphicx`).

1.2. Boîtes TEX

Un boîte est un objet traité comme une unité.

Chaque boîte est définie par la donnée de son contenu et par quatre attributs :

- un point de référence solidaire du contenu de la boîte ;
- la largeur de la boîte ;
- la hauteur et la profondeur de la boîte.



En général, le contenu d'une boîte sera situé *visuellement* à l'intérieur des limites de la boîte. Toutefois, il y a des cas où le contenu peut dépasser de la boîte (par exemple certaines lettres italiques).

La hauteur totale d'une boîte n'est pas un attribut de la boîte mais est déterminé à partir de la hauteur et de la profondeur de la boîte.

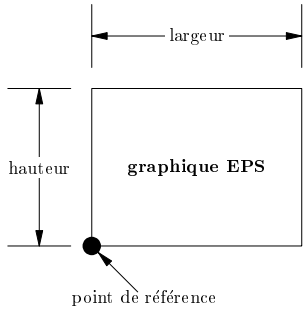
1.3. Manipulation des boîtes

Lorsque plusieurs boîtes sont mises les unes à la suite des autres, elles sont juxtaposées de telle sorte que les lignes de base soient alignées.

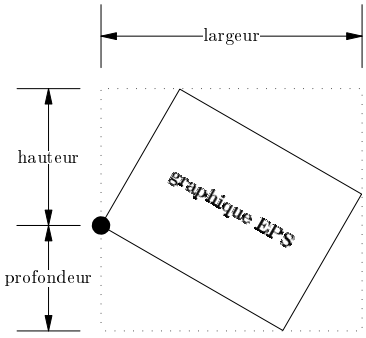
1.4. Boîte correspondant à un fichier EPS

Le point de référence d'un fichier EPS n'ayant pas subi de rotation est le coin inférieur gauche.

- La profondeur est nulle ;
- La hauteur totale est égale à la hauteur.



Après rotation autour du point de référence, les attributs sont les suivants :



Par diverses manipulations, le point de référence peut devenir différent du coin inférieur gauche de la figure non tournée.

1.5. *Bounding Box*

Un fichier EPS comporte non seulement des commandes de dessin PostScript, mais aussi une *Bounding Box* (boîte englobante) qui spécifie la véritable taille du dessin.

Celle-ci est donnée dans un commentaire du fichier EPS, par exemple :

```
%!PS-Adobe-2.0 EPSF-2.0
%%Creator: dvipsk 5.58f
%%Title: table-cmr10.dvi
%%BoundingBox: 71 292 541 716
%%EndComments
```

Les quatre entiers de la *Bounding Box* (BB) sont :

- l'abscisse du coin inférieur gauche de la BB ;
- l'ordonnée du coin inférieur gauche de la BB ;
- l'abscisse du coin supérieur droit de la BB ;
- l'ordonnée du coin supérieur droit de la BB.

Coordonnées exprimées en points PostScript, égaux à $1/72$ de pouce.

Largeur du fichier EPS précédent : 541-71=470 points PS

Hauteur du fichier EPS précédent : 716-292=424 points PS

En TEX, un `bp` (*big point*) correspond à un point PS et un `pt` est égal à $1/72.27$ de pouce.

1.6. Format PostScript Encapsulé

Un fichier au format EPS ne peut pas comporter n'importe quelle commande PS. Par exemple, les commandes `clear`, `letter`, `a4`, etc, ne sont pas autorisées.

Un fichier d'une page sans le commandes interdites peut être converti en EPS avec des programmes tels que `ps2epsi`. La BB peut aussi être calculée avec `bbfig` ou mesurée directement.

1.7. Formats PostScript non standard

Certains logiciels créent des fichiers PS non standard et il existe en général des programmes pour corriger ces fichiers. Les fichiers PS issus de FrameMaker peuvent par exemple être corrigés au moyen du script `fixfm5`.

2. Commandes du *package* graphics

Les *packages* `graphics` et `graphicx` sont chargés en indiquant le *driver* en option, par exemple :

```
\usepackage[dvips]{graphics}
```

La seule différence entre `graphics` et `graphicx` est d'ordre syntaxique.

Le *package* `graphicx` comporte cinq commandes principales :

```
\includegraphics[options]{filename}
\rotatebox[options]{angle}{argument}
\scalebox{h-scale}[v-scale]{argument}
\resizebox{width}{height}{argument}
\resizebox*{width}{totalheight}{argument}
```

2.1. Commande `\includegraphics`

`\includegraphics[`⟨*options*⟩`]{`⟨*nom de fichier*⟩`}`

L'usage le plus simple consiste à écrire

```
\documentclass{article}
\usepackage{graphicx}
\begin{document}
\includegraphics{file.eps}
\end{document}
```

L'inclusion se fait à la taille normale.

2.1.1. Modification de la largeur

```
\includegraphics[width=3in]{file.eps}
```

inclut `file.eps` de telle sorte que la largeur soit 3 pouces.

```
\includegraphics[width=\textwidth]{box.eps}
\includegraphics[width=0.80\textwidth]{box.eps}
```

Avec le *package* `calc`, on peut écrire :

```
\includegraphics[width=\textwidth-2.0in]{box.eps}
```

2.1.2. Modification de la hauteur

En général, c'est la hauteur totale (`totalheight`) et non la hauteur (`height`) qui interviendra dans les options des commandes `\includegraphics`.

2.1.3. Justification

Une figure est placée là où la commande est déclarée.

Pour centrer une figure, il suffit de l'inclure dans l'environnement `center` :

```
\begin{center}
\includegraphics[width=2in]{box.eps}
\end{center}
```

On peut aussi utiliser `\centering` dans un environnement `figure` :

```
\begin{figure}
  \centering
  \includegraphics[width=2in]{box.eps}
\end{figure}
```

ou

```
\begin{figure}
  \begin{center}
  \includegraphics[width=2in]{box.eps}
  \end{center}
\end{figure}
```

(l'environnement `center` produit de l'espace supplémentaire)

2.2. Commande `\scalebox`

`\scalebox{`⟨*h-scale*⟩`}[`⟨*v-scale*⟩`]{`⟨*argument*⟩`}`

La hauteur est multipliée par `v-scale` qui est optionnel est égal à `h-scale` sinon. La largeur est multipliée par `h-scale`. Des valeurs négatives permettent de faire des réflexions.

2.3. Commande `\resizebox`

`\resizebox{`⟨*largeur*⟩`}{`⟨*hauteur*⟩`}{`⟨*argument*⟩`}`

`\resizebox*{`⟨*largeur*⟩`}{`⟨*hauteur totale*⟩`}{`⟨*argument*⟩`}`

Ces commandes changent la taille d'un objet pour lui donner une taille cible. Si `!` est employé dans l'un des paramètres, l'aspect est conservé.

`\height`, `\depth`, `\totalheight` et `\width` peuvent être utilisés pour faire référence aux dimensions d'origine. Exemple :

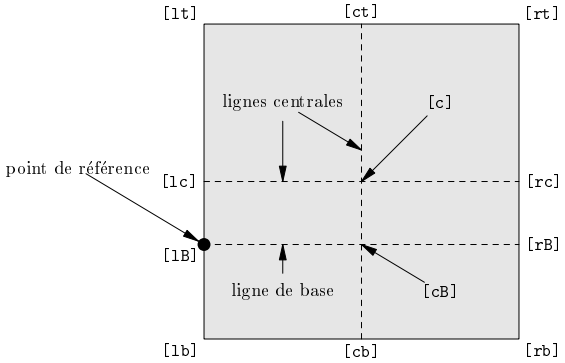
```
\resizebox{2in}{\height}{argument}
```

2.4. Commande `\rotatebox`

`\rotatebox[`⟨*options*⟩`]{`⟨*angle*⟩`}{`⟨*argument*⟩`}`

Cette commande permet de faire faire une rotation d'une angle donné, et autour d'un point donné à une boîte. Par défaut, la rotation se fait autour du point de référence.

Douze points particuliers sont identifiés dans une boîte :



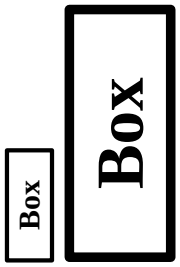
On peut par exemple écrire `\rotatebox[origin=c]{50}{\includegraphics{file.eps}}` pour effectuer une rotation de 50 degrés autour du centre de la BB.

3. Alignement de graphiques EPS

3.1. Ordre des options

Les options sont évaluées de la gauche vers la droite :

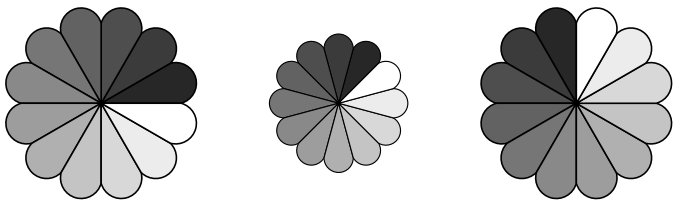
```
\begin{center}
  \includegraphics[angle=90,totalheight=0.5in]{figures.5}
  \includegraphics[totalheight=0.5in,angle=90]{figures.5}
\end{center}
```



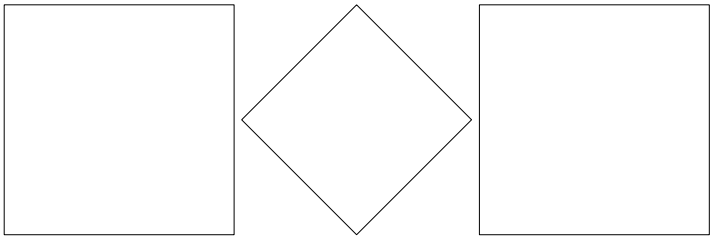
3.2. Taille, BB et objet

Lorsqu'une taille est spécifiée, c'est celle de la BB, pas de l'objet. Cela peut donner des résultats paradoxaux :

```
\begin{center}
  \includegraphics[totalheight=1in]{rosette.ps}
  \includegraphics[angle=45,totalheight=1in]{rosette.ps}
  \includegraphics[angle=90,totalheight=1in]{rosette.ps}
\end{center}
```



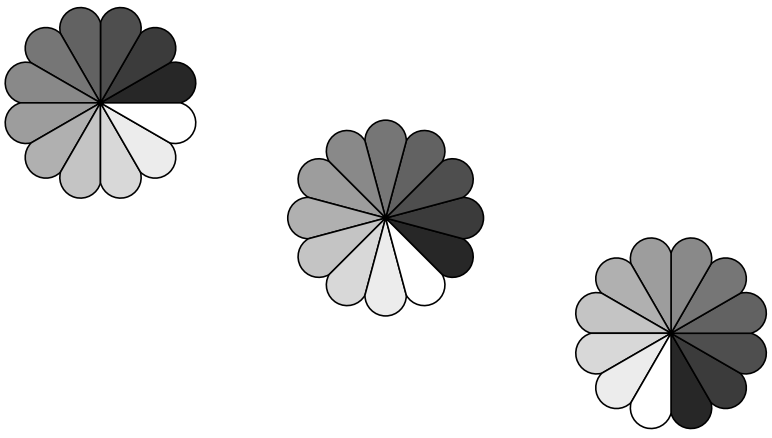
L'examen des BB explique la situation :



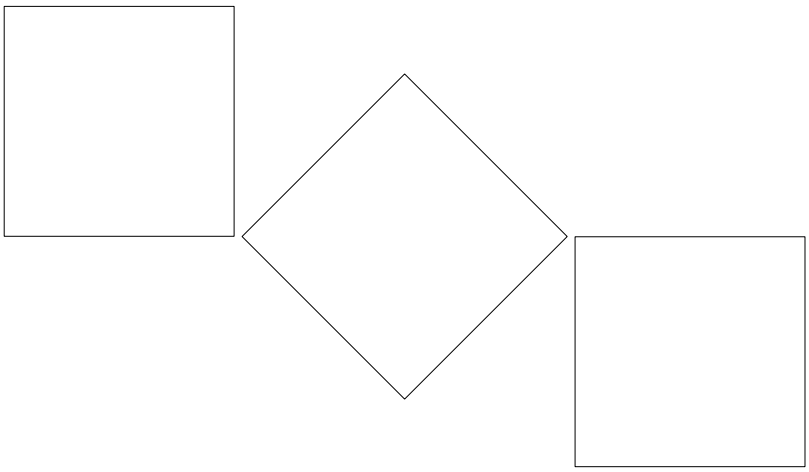
3.3. Alignement d'objets tournés

Comme l'alignement se fait sur le point de référence, on peut aussi avoir des surprises :

```
\begin{center}
  \includegraphics[totalheight=1in]{rosette.ps}
  \includegraphics[totalheight=1in,angle=-45]{rosette.ps}
  \includegraphics[totalheight=1in,angle=-90]{rosette.ps}
\end{center}
```



L'examen des BB donne :

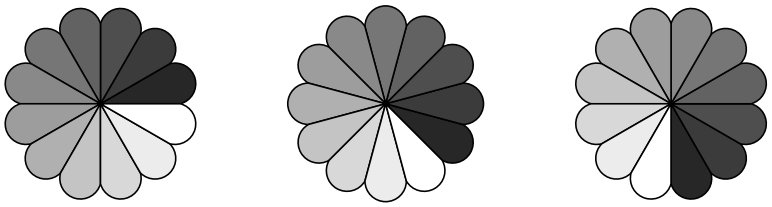


(l'espace entre les BB est dû à l'espace à la fin des lignes `\includegraphics`)

Si on souhaite l'alignement par rapport au centre, on peut utiliser l'option `origin` :

```
\begin{center}
  \includegraphics[totalheight=1in]{rosette.ps}
  \includegraphics[totalheight=1in,origin=c,angle=-45]{rosette.ps}
  \includegraphics[totalheight=1in,origin=c,angle=-90]{rosette.ps}
\end{center}
```

qui produit :



De même,

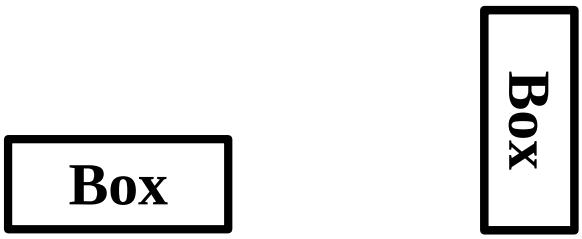
```
\begin{center}
\includegraphics[width=1in]{box.eps}
\hspace{1in}
\includegraphics[width=1in,angle=-90]{box.eps}
\end{center}
```

donne :

Les bas des figures peuvent être alignés en effectuant la rotation autour du coin inférieur droit :

```
\begin{center}
\includegraphics[width=1in]{box.eps}
\hspace{1in}
\includegraphics[width=1in,origin=br,angle=-90]{box.eps}
\end{center}
```

qui donne :

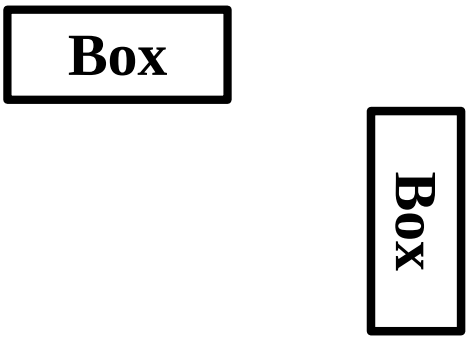


3.4. Alignement dans des environnements `minipage`

Les options `[b]` ou `[t]` ne produisent pas forcément ce à quoi on s'attend dans le cas de figures :

```
\begin{center}
  \begin{minipage}[b]{.25\textwidth}
    \centering \includegraphics[width=1in]{box.eps}
  \end{minipage}
  \begin{minipage}[b]{.25\textwidth}
    \centering \includegraphics[width=1in,angle=-90]{box.eps}
  \end{minipage}
\end{center}
```

donne :



L'option `[b]` n'aligne pas les bas des figures mais les *lignes de base* des premières lignes du bas. Lorsqu'il n'y a qu'une ligne, les options `[t]` et `[b]` ont le même effet.

3.4.1. Alignement des bas de `minipage`

Il suffit d'ajouter une ligne d'epaisseur nulle au bas de chaque environnement `minipage` :

```
\begin{center}
  \begin{minipage}[b]{.25\textwidth}
    \centering \includegraphics[width=1in]{box.eps}
    \par\vspace{0pt}
  \end{minipage}
  \begin{minipage}[b]{.25\textwidth}
    \centering \includegraphics[width=1in,angle=-90]{box.eps}
    \par\vspace{0pt}
  \end{minipage}
\end{center}
```

qui produit :

3.4.2. Alignement des hauts de `minipage`

De manière similaire, on peut ajouter une ligne d'épaisseur nulle au haut des environnements `minipage` :

```
\begin{center}
  \begin{minipage}[t]{.25\textwidth}
    \vspace{0pt}
    \centering \includegraphics[width=1in]{box.eps}
  \end{minipage}
  \begin{minipage}[t]{.25\textwidth}
    \vspace{0pt}
    \centering
    \includegraphics[width=1in,angle=-90]{box.eps}
  \end{minipage}
\end{center}
```

qui produit :

4. Inclusion de fichiers compressés

Le *driver* `dvips` peut exécuter une opération sur un fichier avant insertion, par exemple :

- décompression de fichiers compressés ;
- changement de format de figures, ce qui permet d'inclure des figures qui ne sont pas au format PS.

Les commandes `\DeclareGraphicsRule` et `\DeclareGraphicsExtensions` permettent de spécifier comment LaTeX traite des fichiers inclus avec `\includegraphics`.

4.1. Exemple

Pour utiliser un fichier compressé, il faut avoir sa BB dans un fichier séparé, car ce n'est pas TeX qui fait la décompression, mais `dvips`. Les étapes sont les suivantes :

1. créer un fichier EPS, par exemple `fichier.eps` ;
2. mettre la BB dans un autre fichier, par exemple `fichier.eps.bb` ;
3. compresser le fichier avec `gzip` ;

4. inclure la commande `\DeclareGraphicsRule` appropriée avant `\includegraphics` :

```
\documentclass[dvips]{article}
\usepackage{graphicx}
\begin{document}
  \DeclareGraphicsRule{.eps.gz}{eps}{.eps.bb}{‘gunzip -c #1}
  \begin{figure}
    \centering
    \includegraphics[width=3in]{fichier.eps.gz}
    \caption{Fichier EPS compress\’e}
    \label{fig:compress}
  \end{figure}
\end{document}
```

4.2. Commande `\DeclareGraphicsRule`

`\DeclareGraphicsRule{`⟨*ext*⟩`}{`⟨*type*⟩`}{`⟨*fichier de bb*⟩`}{`⟨*commande*⟩`}`

- *ext* est l’extension du fichier compressé ;
- *type* est le type du fichier décompressé ;
- *fichier de bb* est l’extension du fichier contenant la BB ;
- *commande* est la commande à appliquer ; la commande doit être précédée par une cote inversée (*backquote*).

Ainsi, `\DeclareGraphicsRule{.eps.gz}{eps}{.eps.bb}{‘gunzip -c #1}` spécifie qu’un fichier avec l’extension `.eps.gz` est traité comme un fichier EPS compressé où la BB est stockée dans le fichier d’extension `.eps.bb` et la commande `gunzip -c` décompresse le fichier.

Un certain nombre de règles, dont celle qui a été utilisée, sont déjà définies dans le fichier `dvips.def`.

4.3. Commande `\DeclareGraphicsExtensions`

La commande `\DeclareGraphicsExtensions` permet de spécifier les extensions à tester si un `\includegraphics` est donné sans extension. Par défaut, on a l'équivalent de :

```
\DeclareGraphicsExtensions{.eps,.ps,.eps.gz,.ps.gz,.eps.Z}
```

Par conséquent, `\includegraphics{fichier}` fera d'abord chercher `fichier.eps`, puis `fichier.ps`, etc.

4.4. Inclusion de fichiers non EPS

Ces fichiers doivent être convertis.

Exemple d'inclusion de fichier GIF :

```
\DeclareGraphicsRule{.gif}{eps}{.gif.bb}{`giftoppm #1 | ppmtopgm |pgmtops}
\includegraphics[width=3in]{fichier.gif}
```

5. *Package* `psfrag` : PostScript annoté par du texte

L'inconvénient de beaucoup de programmes externes à TEX est qu'ils ne permettent pas d'inclure des étiquettes écrites en TEX. Le *package* `psfrag` permet de spécifier des substitutions de symboles dans une fichier EPS.

Pour utiliser `psfrag`, on procède comme suit :

- inclure `\usepackage{psfrag}` dans le préambule ;
- dans le document, utiliser des commandes `\psfrag` pour spécifier le texte EPS à remplacer ;
- utiliser la commande `\includegraphics` normalement.

La syntaxe de `\psfrag` est :

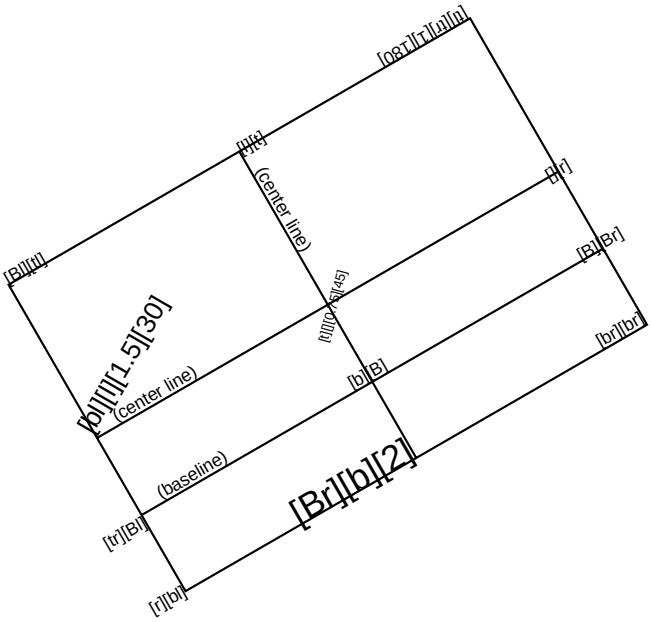
`\psfrag{`⟨*textePS*⟩`}[`⟨*posn*⟩`][`⟨*PSposn*⟩`][`⟨*scale*⟩`][`⟨*rot*⟩`]{`⟨*texte*⟩`}`

- *textePS* est le texte à remplacer dans le fichier EPS ;
- *posn* est la position de point de placement par rapport au nouveau texte ; par défaut, c'est `Bl` ;
- *PSposn* est la position de point de placement par rapport à l'ancien texte ; par défaut, c'est `Bl` ;

– *scale* est le facteur d'augmentation de la taille ; il est préférable d'utiliser `\small`, `\large`, etc., dans les commandes `\psfrag` ;

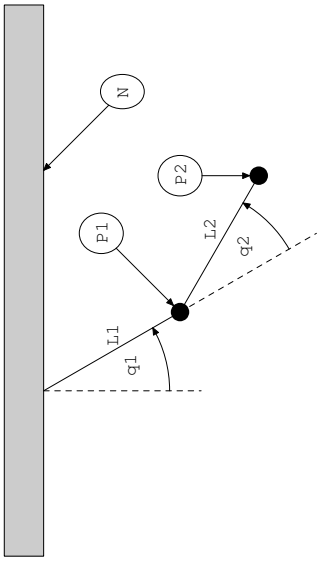
– *rot* est l'angle de rotation des étiquettes, ce qui peut être utile avec des programmes qui ne permettent de créer des fichiers EPS que avec des étiquettes horizontales.

5.1. Options de placement



5.2. Exemple

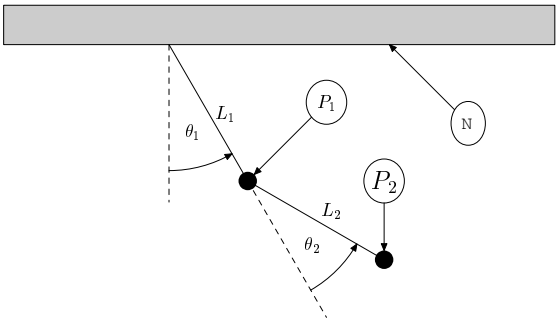
`\includegraphics{pend.eps}` inclut le fichier `pend.eps` sans aucun remplacement, ce qui donne :



Toutefois, si l'on écrit :

```
\psfrag{q1}{$\theta_1$}
\psfrag{q2}{$\theta_2$}
\psfrag{L1}{$L_1$}
\psfrag{L2}{$L_2$}
\psfrag{P1}[][]{$P_1$}
\psfrag{P2}[][]{\Large $P_2$}
\includegraphics{pend.eps}
```

on obtient :



- toutes les étiquettes n'ont pas besoin d'être remplacées ;
- `\psfrag{pi}{$\pi$}` remplace `pi`, mais pas `pi/2`, etc.

6. Inclusion d'un fichier PS de multiples fois

Lorsqu'une figure est chargée plusieurs fois, par exemple un logo, son code apparaît de multiples fois et on peut chercher à obtenir quelque chose de plus efficace.

Il y a quatre méthodes courantes pour inclure un fichier PS plusieurs fois :

1. utiliser `\includegraphics` pour chaque figure ; ceci a l'inconvénient que la figure doit être lue à chaque fois et figure à chaque fois dans le fichier PostScript final ;
2. sauvegarder le résultat de `\includegraphics` dans une boîte :
   ```
   \newsavebox\mongraphique
   \sbox\mongraphique{\includegraphics{fichier.eps}}
   ```
   puis l'utiliser avec `\usebox{\mongraphique}` ; le fichier n'est lu qu'une fois, mais le code figure néanmoins plusieurs fois dans le fichier PostScript final ;
3. si le fichier EPS contient un graphique vectoriel, il est possible de créer une commande PostScript qui dessine le graphique ; il suffit alors d'appeler la commande plusieurs fois, mais le code du dessin n'apparaît qu'une fois ;
4. comme dans le précédent cas, mais en incluant la commande dans une boîte.

6.1. Définition d'une commande PostScript

Un fichier graphique vectoriel doit être découpé en deux fichiers, l'un définissant le dictionnaire PostScript et les commandes graphiques, l'autre incluant les informations de l'en-tête et l'utilisation de la commande précédemment définie.

6.1.1. Exemple : EPS de `xfig`

Un fichier EPS produit par `xfig` a l'aspect suivant :

```
%!PS-Adobe-2.0 EPSF-2.0
%%Title: unnamed.fig
%%Creator: fig2dev Version 3.1 Patchlevel 2
%%CreationDate: Sun Mar 30 22:34:36 1997
%%For: roegel@bellange.loria.fr (Denis B. Roegel)
%%Orientation: Landscape
%%BoundingBox: 0 0 189 194
%%Pages: 0
%%BeginSetup
%%IncludeFeature: *PageSize A4
%%EndSetup
%%EndComments
/$F2psDict 200 dict def
$F2psDict begin
$F2psDict /mtrx matrix put
```

```
/col-1 {0 setgray} bind def
/col0 {0.000 0.000 0.000 srgb} bind def
...
/$F2psBegin {$F2psDict begin /$F2psEnteredState save def} def
/$F2psEnd {$F2psEnteredState restore end} def
%%EndProlog
$F2psBegin
10 setmiterlimit
...
$F2psEnd
rs
```

Le fichier EPS contient en général trois parties :

1. L'en-tête qui commence par `%` ;
2. Le prologue qui commence par `/$F2psDict 200 dict def` et s'achève avec `%%EndProlog` ; `$F2psDict` est le nom du dictionnaire, et d'autres noms que celui-ci peuvent être utilisés ;
3. La dernière partie contient les commandes utilisées pour tracer les graphiques.

6.1.2. Création de deux nouveaux fichiers

Supposons que le fichier qui précède est appelé `fic.eps`. On crée alors les fichiers `fic.h` et `fic.ps` où `fic.h` contient :

```
/$F2psDict 200 dict def
$F2psDict begin
$F2psDict /mtrx matrix put
/col-1 {0 setgray} bind def
/col0 {0.000 0.000 0.000 srgb} bind def
...
/$F2psBegin {$F2psDict begin /$F2psEnteredState save def} def
/$F2psEnd {$F2psEnteredState restore end} def
%%EndProlog

/MaFigure{
$F2psBegin
10 setmiterlimit
...
$F2psEnd
} def
```

et `fic.ps` contient :

```
%!PS-Adobe-2.0 EPSF-2.0
%%Title: unnamed.fig
%%Creator: fig2dev Version 3.1 Patchlevel 2
%%CreationDate: Sun Mar 30 22:34:36 1997
```

```
%%For: roegel@bellange.loria.fr (Denis B. Roegel)
%Magnification: 1.00
%%Orientation: Landscape
%%BoundingBox: 0 0 189 194
%%Pages: 0
%%BeginSetup
%%IncludeFeature: *PageSize A4
%%EndSetup
%%EndComments
$F2psDict begin MaFigure end
```

6.1.3. Utilisation de la figure sauvegardée

La figure peut être utilisée avec :

```
\documentclass{article}
\usepackage{graphicx}
\special{header=fic.h}
\begin{document}
  ...
  \includegraphics[width=2in]{fic.ps}
  ...
  \includegraphics[totalheight=1in]{fic.ps}
  ...
\end{document}
```

Le fichier `fic.eps` n'est donc plus utilisé.

On peut ensuite, si on le souhaite, sauver la commande `\includegraphics` dans une boîte comme vu plus haut.

6.2. Graphiques dans les en-têtes ou pieds de page

On peut utiliser le *package* `fancyhdr` de Piet van Oostrum. Exemple :

```
\documentclass{article}
\usepackage{fancyhdr,graphicx}
\renewcommand{\headheight}{0.6in} %% doit etre assez grand pour le dessin
\renewcommand{\textheight}{7.5in}
% definition de commande PS
\special{header=fic.h}
% sauvegarde du dessin dans une boite
\newsavebox\mongraphique
\sbox\mongraphique{\includegraphics[totalheight=0.5in]{fic.ps}}
\pagestyle{fancy}
\fancyhead{} % efface tous les en-tetes
\fancyhead[L]{\usebox{\mongraphique}}
\fancyfoot{} % efface tous les pieds de page
\fancyfoot[C]{\thepage}
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}
\begin{document}
...
\end{document}
```

7. Commandes LaTeX autour de l'environnement `figure`

7.1. Emploi de l'environnement `figure`

```
\begin{figure}
  \centering
  \includegraphics[totalheight=2in]{graph.eps}
  \caption{Ceci est un graphique EPS}
  \label{fig:graph}
\end{figure}
```

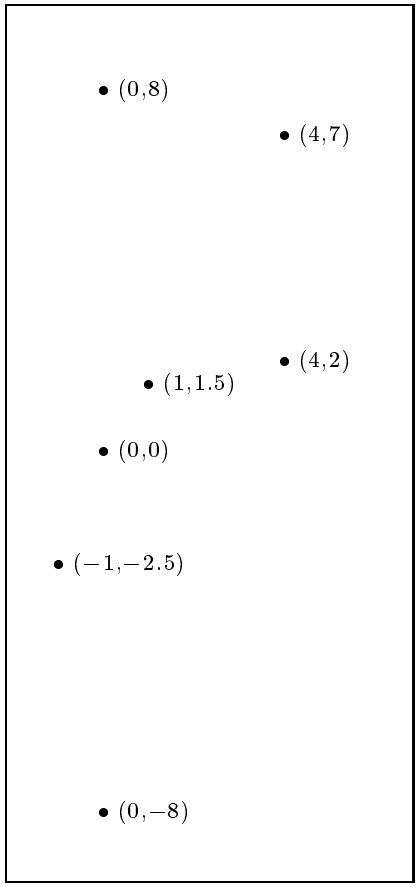
7.2. Espacement vertical autour de la légende

L'espacement peut être réglé par `\abovecaptionskip` et `\belowcaptionskip`.

```
\begin{figure}
  \setlength{\abovecaptionskip}{0pt}
  \setlength{\belowcaptionskip}{10pt}
  \centering \caption{L\'egende au-dessus}
  \includegraphics[width=1in]{box.eps}
\end{figure}
```

7.3. Autres possibilités

Plusieurs autres *packages* permettent d'encadrer des figures de diverses manières, permettent de personnaliser les légendes, etc. On trouvera tous les détails dans l'article de Keith Reckdahl.



8. Production de figures au sein de LaTeX

8.1. Graphiques en TeX pur

Dans le TeXbook, Knuth donne l'exemple suivant :

```
\newdimen\unit
\newbox\ursa
\def\point#1 #2 {\rlap{\kern#1\unit
    \raise#2\unit\hbox{$
      \scriptstyle\bullet\;(#1,#2)$}}}
\unit=\baselineskip
\setbox\ursa=\vtop{\hrule
  \hbox{\vrule height10\unit depth9.4\unit \kern2\unit
    \hbox{\unit=\baselineskip
      \point 0 0 % Alioth (Epsilon Urs\ae\ Majoris)
      \point 0 8 % Dubhe (Alpha Urs\ae\ Majoris)
      \point 0 -8 % Alkaid(Eta Urs\ae\ Majoris)
      \point -1 -2.5 % Mizar (Zeta Urs\ae\ Majoris)
      \point 4 7 % Merak (Beta Urs\ae\ Majoris)
      \point 4 2 % Phekda (Gamma Urs\ae\ Majoris)
      \point 1 1.5 % Megrez (Delta Urs\ae\ Majoris)
      }%
    \kern7\unit \vrule}\hrule}

\box\ursa
```

qui produit :

8.2. Environnement picture

est produit par :

```
\newcounter{cms}
\setlength{\unitlength}{1mm}
\begin{picture}(50,39)
\put(15,20){\circle{6}}
\put(30,20){\circle{6}}
\put(15,20){\circle*{2}}
\put(30,20){\circle*{2}}
\put(10,24){\framebox(25,8){voiture}}
\put(10,32){\vector(-2,1){10}}
\multiput(10,7)(10,0){5}{\addtocounter{cms}{1}
\makebox(0,0)[b]{\arabic{cms}}}
\put(0,7){\makebox(0,0)[bl]{cm}}
\multiput(1,0)(1,0){49}{\line(0,1){2,5}}
\multiput(5,0)(10,0){5}{\line(0,1){3,5}}
\thicklines
\put(0,0){\line(1,0){50}}
\multiput(0,0)(10,0){6}{\line(0,1){5}}
\end{picture}
```

8.3. *package* graphpap

Le *package* graphpap permet d'afficher une grille qui facilite le tracé de dessins dans l'environnement picture :


```
\setlength{\unitlength}{0.2mm}
\graphpaper(-75,0)(200,100)
```

8.4. pictex

```
$$\beginpicture     % Structural formula of \carma

\setcoordinatesystem units <10mm,10mm> % niet veranderen
\setplotarea x from 0 to 7, y from -1.5 to 1
\normalgraphs
\setlinear

\put {$H_3C$}   [br] at 1.6  0.5
\put {$H_3C$}   [tr] at 1.6 -0.5
\put {N---C}    [l]  at 2.0  0.0
\put {O}        [bl] at 3.3  0.5
\put {$O^-$}    [tl] at 3.3 -0.5
\put {$H_2N^+$} [l]  at 5.0  0.0
\put {$CH_3$}   [bl] at 6.0  0.5
\put {$CH_3$}   [tl] at 6.0 -0.5

\plot 1.6   0.5   2.0   0.1 /
\plot 1.6  -0.5   2.0  -0.1 /
\plot 3.25  0.5   2.95  0.2 /
\plot 3.35  0.5   3.05  0.2 /
\plot 3.3  -0.5   3.0  -0.2 /
\plot 6.0   0.5   5.6   0.2 /
\plot 6.0  -0.5   5.6  -0.2 /

\put {dimethyl ammonium carbonate} at 3.7 -1.3

\endpicture$$
```

donne :

dimethyl ammonium carbonate

Le *package* `pictex` est très puissant, mais aussi très gourmand en mémoire et en temps de calcul.

8.5. xypic

Le *package* xypic de Kristoffer Rose est particulièrement adapté à des diagrammes complexes avec diverses sortes de flèches. Bien souvent les diagrammes s'appuient sur une structure matricielle.

Voici un petit exemple :

```
$$\diagramcompileto{dt-font-files}
   &{}\framed<5pt>\Text{Texte source\\ (\extname{.tex})}
   \ddto|{\ncmd{tex}}\dlto|{\ncmd{tex}} &{}
   \framed<5pt>\Text{Fonte PS\\
      (\extname{.pfa}, \extname{.pfb}, \extname{.afm})}
   \dto|{\ncmd{afm2tfm}}&{}
   \framed<5pt>\Text{Fonte source\\ \nfont{cmr10.mf}}
   \dlto|{\ncmd{mf}}\dto|{\ncmd{mf}}\drto|{\ncmd{mf}}\cr
         % fin de la premiere ligne
  {}\extname{.log}, etc.&{}
   &{}\mathtt{cmr10.tfm}\dlto|{\ncmd{tex}}
         \dto|{\ncmd{dvips}}
   &{}\mathtt{cmr10.300gf}
         \dto|{\ncmd{gftopk}}&{}\extname{.log}\cr
         % fin de la seconde ligne
   &{}\extname{.dvi}\rto|{\ncmd{dvips}}
                  \drto|{\ncmd{xdvi}}
   &{}\framed<5pt>{\extname{.ps}}&{}
   {}\mathtt{cmr10.300pk}\lto|{\ncmd{dvips}}
                        \dlto|{\ncmd{xdvi}}&{}\cr
         % fin de la troisieme ligne
   &&{}\framed<5pt>{\hbox{\'ecran}}\cr
\enddiagram$$
```

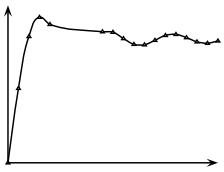
qui se traduit en :

8.6. *package* pstricks

Le *package* pstricks de Timothy Van Zandt fournit de nombreuses commandes qui se traduisent toutes en PostScript, et offrent de ce fait beaucoup plus de possibilités que le simple environnement picture.

```
{\psset{xunit=.2cm,yunit=1.5cm}
\pspicture(0,0)(20,2.2)
        \savedata{\mydata}[
          {{0, 0}, {1., 0.946083}, {2., 1.60541}, {3., 1.84865}, {4., 1.7582},
            {9., 1.66504}, {10., 1.65835}, {11., 1.57831}, {12., 1.50497},
            {13., 1.49936}, {14., 1.55621}, {15., 1.61819}, {16., 1.6313},
            {17., 1.59014}, {18., 1.53661}, {19., 1.51863}, {20., 1.54824}}]
        \dataplot[plotstyle=curve,showpoints=true,
            dotstyle=triangle]{\mydata}
        \psline{<->}(0,2)(0,0)(20,0)
\endpspicture}
```

produit :

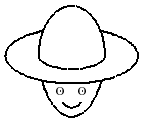


8.7. DraTex

Le *package* DraTex de Eitan Gurari définit un langage pour le tracé de figures :

```
\Draw(0.6pt,0.6pt)
  \DrawOvalArc(30,45)(0,180)
  \DrawOvalArc(30,12)(180,360)
  \DrawOvalArc(60,25)(140,400)
  \Move(0,5)
  \DrawOvalArc(30,60)(225,315)
  \DrawOvalArc(20,52)(260,280)
  \Move(-10,-37) {\Text(--$\cdotp$--)}  \Text(--o--)
  \Move(20,0)    {\Text(--$\cdotp$--)}  \Text(--o--)
\EndDraw
```

produit



Voir sur le LN pour plus de détails.

9. Outils annexes

De nombreux programmes peuvent produire une sortie utilisable par LaTeX ; très souvent, ce sera du PostScript.

9.1. xfig

Peut produire du PostScript, et divers autres formats.

9.2. tgif

Peut produire du PostScript.

9.3. gnuplot

Peut produire du PostScript, de l'environnement picture de LaTeX, etc. Une documentation est disponible sur le LN.

9.4. bm2font

Permet de convertir une image *bitmap* en caractère de police.

9.5. metapost

Metapost (de John Hobby) figure en standard dans les distributions TEX à partir de web2c 7.0.

`Metapost` est une variante de `metafont` et produit non pas des caractères de polices, mais des fichiers PostScript. L'un des points forts de `metapost` est d'être fort bien intégré avec TEX.

L'un des premiers dessins de cette série de transparents a été produit avec :

```
beginfig(1);
numeric u;u=1.5cm;
ahlength:=.2u;ahangle:=25;
z1=origin;z2-z1=(2u,0);z3-z1=(0,u);z4-z3=z2-z1=z6-z5;
z6-z4=2(z4-z2);
fill z1--z2--z6--z5--cycle withcolor (.9,.9,.9);
draw z1--z2--z6--z5--cycle;
dbltextarrow(z5+(0,.5u),z6+(0,.5u),btex largeur etex);
...
pickup pencircle scaled 10pt;
drawdot z3;
pickup pencircle scaled .4pt;
label(btex point de r\'ef\'erence etex,.5[z3,z6]);
drawarrow .5[z3,z6]--z3 cutafter circlepath(10pt,z3)
       cutbefore circlepath(20pt,.5[z3,z6]);
draw z3--z4 dashed evenly;
label(btex ligne de base etex,.5[z3,z2]);
drawarrow .5[z3,z2]--.4[z3,z4]
      cutbefore circlepath(20pt,.5[z3,z2]);
endfig;
```

Pour en savoir plus

- LATEX — A document preparation system de Leslie Lamport
- The LATEX Companion, de Goossens, Mittelbach et Samarin
- The LATEX Graphics Companion, de Goossens, Mittelbach et Rahtz
- The TEXbook, de Donald Knuth
- Keith Reckdahl, Using EPS Graphics in LATEX 2ε Documents, disponible sur le LN

Documentations accessibles sur le LATEX Navigator (LN) (`http://www.loria.fr/tex`), en particulier le guide local du site LORIA.

Serveurs CTAN (miroir : `ftp.loria.fr/pub/unix/tex/ctan`)