

# L<sup>A</sup>T<sub>E</sub>X-Paragraphs Floating around Figures

Thomas Kneser  
Gesellschaft für wissenschaftliche  
Datenverarbeitung Göttingen  
Am Fassberg  
D-3400 Göttingen, FRG  
Bitnet: tkneser@dgogwdg1

## Abstract

A L<sup>A</sup>T<sub>E</sub>X-environment is presented which has the following property: Figures which are substantially narrower than `\textwidth` are placed automatically right or left justified within one or more paragraphs. Such figures can be set as easily and in the same way as L<sup>A</sup>T<sub>E</sub>X's standard figures.

## When to Use the FLOATFIG Style Option

Figures often do not fill the full page width. If the width of such figures is only half of the page width or even less, lines of text should be set beside the figures, or—from another point of view—figures should “float” in paragraphs. Figures 1 and 2 show examples of such “Floating Figures”.<sup>1</sup> They can be set by using our FLOATFIG style option.

The macros which make up the FLOATFIG style option are based on PLAIN- $\TeX$  macros developed by Thomas Reid (*TUGboat* Vol. 8 # 3 page 315), who showed how to set figures *right justified* with paragraphs floating around them. For such layout Reid chose the term “Floating Figures”.

This choice could cause confusion, when we adapt these macros for L<sup>A</sup>T<sub>E</sub>X, since Leslie Lamport uses the term “float” for objects which are realized as  $\TeX$ -`\inserts`. While Lamport's floats are floating in the “main vertical list”, floats introduced by Reid are floating within paragraphs. For what follows we adopt the latter definition.

## How to Use It

The Floating Figures style option is fully compatible with L<sup>A</sup>T<sub>E</sub>X's standard figure facility:

1. Floating Figures and standard figures may be requested in any sequence,
2. Floating Figures can be captioned like standard figures,

<sup>1</sup> The pages shown are a resetting of some pages from HERMANN WEYL'S book on Symmetry (German edition, Birkhäuser Verlag 1955)

3. Captioned Floating Figures are inserted in the list of figures which may be printed by the standard `\listoffigures` command.

A Floating Figure may be requested as follows:

```
\documentstyle[floatfig]{article}
\begin{document}
\initfloatingfigs
.
.
\begin{floatingfigure}{5.6cm}
\vspace{6.0cm}
% optional !
\caption{Intermolecular potential K-Xe}
\end{floatingfigure}
.
.
\end{document}
```

It is essential that `\initfloatingfigs`, which initializes the `floatingfigure` environment, follows `\begin{document}` immediately. Otherwise some formatting errors will occur.

A Floating Figure may only be requested in vertical mode, that is between paragraphs. “5.6cm” in our example specifies the width of the figure space.

A Floating Figure will be set as soon as possible after the request for it has been encountered by L<sup>A</sup>T<sub>E</sub>X. That means, it will be tested whether there is enough vertical space on the current page; if not, the figure will be moved to the next page.

Floating Figures are set *alternating*, that is on the right hand side on odd and on the left hand side on even numbered pages.

## Restrictions

1. The FLOATFIG style option may not be combined with the TWOCOLUMN style option,

2. A Floating Figure cannot appear in a paragraph which begins on top of a page.

## How It works

We have extended the macros designed by Reid with regard to:

1. fitting them into the L<sup>A</sup>T<sub>E</sub>X context,
2. justifying Floating Figures right and left, and
3. the generation of warning messages for “collisions” of two Floating Figures.

**Fitting into the L<sup>A</sup>T<sub>E</sub>X context** The PLAIN-<sub>T</sub>E<sub>X</sub> implementation by Reid is based on a redefined `\output` routine:

```
\edef\oldoutput{\the\output}%
\output={\the\outputpretest
\ifoutput\oldoutput\fi}
\outputpretest={\outputtrue}
```

If a Floating Figure is requested, the content of the `\outputpretest` token register then decides:

1. if there is enough vertical space to set the Floating Figure,
2. if setting of another Floating Figure is already in progress, or
3. if indeed the current page is ready to be sent to the DVI file.

T<sub>E</sub>X has to deal with more than one paragraph until a Floating Figure is completely processed. During this process, the redefined `\output` routine is called at the beginning of *every* paragraph; this is done indirectly by expanding the control sequence `\tryfig`. Therefore, the `\everypar` token list is prepared by the following command sequence:

```
\edef\oldeverypar{\the\everypar}
\everypar={\tryfig\oldeverypar}
```

Now `\tryfig` triggers the (modified) `\output` routine, which then makes the decisions mentioned above.

Adopting this concept when using the macros in the L<sup>A</sup>T<sub>E</sub>X context, we are faced with the following problems:

1. At the time `FLOATFIG.STY` is read in, the `\output` routine is still undefined, and remains undefined until `\begin{document}` is expanded; so the redefinition of the `\output` routine has to be done after `\begin{document}` by the command `\initfloatingfigs` (see section “Known Problems” below).
2. There are situations where L<sup>A</sup>T<sub>E</sub>X decides to redefine the `\everypar` token list without saving the former content; this occurs, for instance, when expanding a `\section` control

sequence. We overcome this by redefining `\everypar` whenever the `\floatingfigure` environment is entered. So to avoid problems, a Floating Figure should be requested early enough before any sectioning control sequence (see also subsection “Misleading collision warnings”). Furthermore, the conflicting definitions of `\everypar` are the reason why Floating Figures cannot move across section boundaries.

**Justifying figures right and left** The problem to be solved is whether a particular figure has to be set left or right justified. This decision has to be made according to the value of the page count (left if even, right if odd). This is because we are dealing with the well known problem of associating a certain part of input text with the number of the page on which it will be finally set .

As pointed out by Donald Knuth in *The T<sub>E</sub>Xbook*, this association is made at `\output` routine time. Therefore the problem is not so hard to solve, since in Reid’s version there is already a modified `\output` routine which decides if a particular figure will fit on the current page. As a by-product of this decision one easily gains the information “odd” or “even” for the page count of the current page. So our problem is reduced to the following simple decision:

```
\ifodd\count0 %
\hbox to \hsize{\hss\copy\figbox}%
\global\oddpagetrue
\else% leftsetting
\hbox to \hsize{\copy\figbox\hss}%
\global\oddpagesfalse
\fi% \ifodd\count0
```

**Collisions of Floating Figures** We define a collision as a situation where:

1. a Floating Figure is requested before a predecessor has been finished, or
2. some sectioning is requested before a Floating Figure has been finished.

While the `FLOATFIG` style option cannot avoid such collisions, it will recognize them. For diagnostic purposes we have therefore defined the switch `\iffigprocessing` and another count register called `\ffigcount`. This count register is used to attach a sequence number to each Floating Figure, so they can be identified uniquely in collision warning messages. These sequence numbers are not to be confused with the figure count maintained by standard L<sup>A</sup>T<sub>E</sub>X.

## Known Problems

**Need of Initialization** The present version of the style option needs to be initialized by the control sequence `\initfloatingfigs`, as mentioned above.

We hope a later version will initialize itself when the first request for a Floating Figure is encountered. One problem with such an automatic initialization seems to be that one is grouped down, being inside a  $\text{\LaTeX}$  environment. So far, I must confess, we have failed to make the respective settings `\global`.

**Misleading collision warnings** As mentioned above, Floating Figures do not move across section boundaries. If a Floating Figure is requested near the end of a section, the actual figure will be truncated, if it does not fit into the current section.

If this has occurred, for instance, with Floating Figure number 4, a collision will be reported when a request for Floating Figure number 5 is encountered. But the message will tell us that there is a problem with figure 4 without further explanation. So one has to check in each case that the problem is *not* caused by collision with figure 5 but with a section heading. What is even worse: if there is no Floating Figure 5, but only a truncated 4, one gets no warning message at all! At the moment, one can only check this on the printout. We pointed out in section "Fitting into the  $\text{\LaTeX}$ -Context" why this unfortunate situation cannot easily, if at all, be remedied.

A minor blemish is due to a retardation caused by the `\everypar` mechanism. Floating Figures in consecutive paragraphs seem to be disapproved. The warning messages generated in this connection can be ignored.

## Conclusions

Working on `FLOATFIG.STY` we had some unexpected problems which were caused by  $\text{\LaTeX}$ 's somewhat unsafe assignments to the `\everypar` token list on the one hand, and by the fact that the use of this token list is fundamental to the algorithm designed by Thomas Reid on the other hand.

We did expect problems in fitting the Floating Figures in with  $\text{\LaTeX}$ 's handling of figure captions: that is, to achieve a single figure caption numbering for standard figures and Floating Figures and to get both types listed together by the `\listoffigures` control sequence. But this problem was easily solved by the following local definition within the `FLOATFIG` environment:

```
\def\@capttype{figure}
```

Obviously this is due to the fact that  $\text{\LaTeX}$ 's caption apparatus is thoroughly parameterized.

The `FLOATFIG.STY` file is stored in the EARN/BITNET listserver in Heidelberg; send the command:

```
GET FLOATFIG ZOOUUE LATEXSTY
```

to `LISTSERV` at `DHDURZ1` to obtain a copy of the style option file.