An output routine for an illustrated book: Making the FAO Statistical Yearbook

Boris Veytsman

Abstract

Output routines involving illustrations (“floating bodies” in the \LaTeX\ lingo) are the most complex part of \TeX. For most algorithms used in \TeX, \LaTeX\ and \ConTeXt\ the basic concept is a flow of text, occasionally interrupted by illustrations which can be placed anywhere close to the point they are mentioned. The story is told mainly by the text, and illustrations have a secondary role.

Here we discuss the different case of an *illustrated book*, where the main story is told by the illustrations and their interaction. The simplest examples of such books are art albums. Another (surprising) example is the FAO Statistical Yearbook, where the story is told primarily by maps, charts and tables, while the text has a secondary role.

We describe a concept of a relatively simple output routine for such books and its implementation in \LaTeX.

1 Introduction

A recent report by the \LaTeX\3 team [4] contained the exhortation to engage in “collecting and classifying design tasks”. In this paper we describe a design task and propose a way to solve it. While the code here is \LaTeX\2-specific, we hope the algorithm and concepts may be useful for other formats as well.

Probably one of the most difficult concepts in \TeX\ is illustrations (“floats” in \LaTeX\ nomenclature). They interrupt the galley, and \TeX\ should put them on the page outside of the normal flow, using an asynchronous output routine (OTR). Various OTRs for plain \TeX\ are described in the series of papers by Salomon [5, 6, 7]; the last part deals specifically with insertions, the usual way to typeset illustrations in plain \LaTeX\2 algorithms are described in [1], probably the most complex part of \LaTeX\ code. These algorithms deal with one- or two-column typesetting with illustrations of arbitrary height occupying one or two columns. \ConTeXt\ can deal with a more general situation of $n$-columns with illustrations occupying $m$ columns of text [3].

It should be noted that all these cases assume that the main story is told by the text. Illustrations are put on the pages almost as an afterthought. They do not interact with each other. The only task of the algorithm is to put them somewhere not too far from the point they are mentioned, and without creating too much empty space on the pages.

Boris Veytsman

Figure 1: An example of an art album spread (from [2])

Figure 2: Mock-up spread of FAO Statistical Yearbook 2014

One can imagine the opposite situation: the story is told primarily by the illustrations and their interaction, while the text plays an auxiliary role. In this case the author spends much effort in putting the illustrations exactly where she wants them to be, and the task of the compositor is to put the text in the remaining empty space in a pleasing manner. We will call books created in this manner *illustrated books*, for lack of a better term.

An immediate example of an illustrated book is an art album (Figure 1). The importance of interaction between the pictures is well known to artists; this is why good exhibitions usually have “Hanging Committees” that carefully discuss the order and positions of the pieces. It is evident from Figure 1 that the author first put the illustrations on the pages, and then filled the rest with the text.

A more surprising example is the FAO Statistical Yearbook (Figure 2). The Yearbook uses tables, charts and maps to illustrate the statistical trends. Their positions on the pages is determined by the graphic designer; the text must fill the gaps.

In the rest of the paper we discuss how the design shown in Figure 2 was implemented in \LaTeX. The
code is available in the repository at http://github.com/filippogheri/FAOSYBLaTeXpackage, and a formatted version is available online with this article.

2 User interface

The main unit of the FAO Yearbook is the spread. As shown in Figure 3, it is split into eight quadrants, four per page. The quadrants are denoted by two-letter combinations like ul for upper left and lr for lower right. Lowercase is used for verso (even) pages, and uppercase for recto (odd) pages.

The illustrations have fixed sizes: they can occupy one, two, or four quadrants. Accordingly there are four kinds of illustrations: ‘Single’ ones take one quadrant, ‘Tall’ ones take two quadrants stacked vertically, ‘Wide’ ones take two quadrants stacked horizontally, and ‘Big’ ones take all four quadrants on a page.

The user specifies the illustration type (e.g. chart or map), its size (S, T, W or B), and the upper left quadrant occupied by the illustration. We used \LaTeX environments for this. The name of the environment corresponds to the illustration type, while its mandatory arguments specify its size and position. For example, the code
\begin{chart}{S}{LR} ... \end{chart}
specifies a chart occupying a single lower right quadrant on a recto page, while the code
\begin{map}{W}{ul} ... \end{map}
specifies a map occupying two top quadrants on a verso page.

The user writes down the code for the illustrations and the text, and \LaTeX typesets them according to the chosen pattern. The command \clearpage typesets all illustrations and text obtained so far.

3 Algorithms

In this section we describe the algorithms used to typeset the book.

The main problem is when to start output. If we had just illustrations, then the answer would be simple: as soon as we have enough illustrations for the full page. This is the approach used by Dave Walden in his photo album macros [8]. However, since we have illustrations and text, we are in a more complex situation. We need to check whether we have enough text to fill the gaps. This is done by page builder. There are ways to inform the page builder about the space needed by illustrations [5]. However they assume that all illustrations should be put on the page being built. In our case we may have both illustrations for the current page and illustrations for following pages. Only the OTR knows which illustrations belong to the current page, but the OTR is started asynchronously by the page builder. Thus our algorithms must include communication between the page builder and the OTR.

Each of the environments described in Section 2 adds its contents to the bottom of an illustration box. There are 18 such boxes corresponding to all valid combinations of illustration size and position (an attempt to insert, e.g. a Tall illustration starting at the lower left quadrant produces an error since this combination is not valid). We use \vsplit to extract the top (oldest) illustration from the box.

We follow the basic idea of [1] for two-column typesetting. The page builder starts the OTR whenever a column of text is formed. It is the job of the OTR to determine whether we are at the first or second column, and proceed accordingly. One can imagine the OTR having two stages: the first deals with a first column from the page builder, and the second has two columns to work with.

So at the first stage we have a column of text. We also know whether this column is the first or the second, and whether we are on a recto or a verso page. Thus we can check whether we already have illustrations in the quadrants for this page.

First, it can happen that the current page is completely covered by Wide or Big illustrations. In this case we do not need to put any text on the page, and simply output the illustrations. Note that this should happen only when we typeset the first column — otherwise we have a full column of text which belongs to a wrong page: recto or verso.

If after this test we are still inside the OTR, then we are free to form a column. Again, it may happen that this column is completely taken by illustrations; in this case we return the text to the page builder and send illustrations to the second stage.

Now we are at the most interesting part of the algorithm. We have text and possibly illustrations to mix in the column. However, is the height of the text box right? Possibly not: the page builder might think that there were no illustrations and not correct for them. Fortunately, \LaTeX provides a global parameter \vsize, which reflects the page builder’s idea about the required text height. So we can calculate the required height in the OTR and

An output routine for an illustrated book: Making the FAO Statistical Yearbook
Algorithm 1: OTR, first stage
if have Big or both top & bottom Wide illustrations then
  if second column then
    \ Error
    Send the illustrations to the special OTR;
    Send text back to page builder
  if have Tall or both top & bottom Single illustrations then
    Form a column from the illustrations;
    Send the column to the second stage;
    Send the text back to page builder
  Calculate column height;
  if column height equals \vsize then
    Add illustrations to the column;
    Send the column to the second stage
  else
    Change \vsize;
    Send text back to page builder;
    Leave OTR

Algorithm 2: OTR, second stage
if first column then
  Save column
else
  Add first column and wide illustrations,
  add decorations and ship the page out
Reset \vsize; Leave OTR

Algorithm 3: OTR, special case
Put illustrations on the page;
Add decorations and ship the page out;
Reset \vsize;
Leave OTR

Algorithm 4: \clearpage
while some illustration boxes are not empty do
  Call OTR

Acknowledgments

This work would have been impossible without great and patient people at FAO UN: Filippo Gheri, Amy Heyman, Shira Fano, and many others.

I am grateful to Hans Hagen and Frank Mittelbach for the discussion of Con\TeXt and \LaTeX\ float routines and to Dave Walden for letting me know about his paper.

As always, the participants of the TUG meeting gave me many interesting comments and suggestions.

References


\begin{itemize}
\item Boris Veytsman
  \begin{itemize}
  \item George Mason University
  \item borisv (at) lk (dot) net
  \item \url{http://borisv.lk.net}
  \end{itemize}
\end{itemize}