The PracTEX Journal, 2008, No. 3
Article revision 2008/11/16

Writing posters in LaTeX

T. Morales de Luna

Email morales@anamat.cie.uma.es
Address Dpto. Análisis Matemático
Facultad de Ciencias
Universidad de Málaga
Málaga 29071
Spain

Abstract LaTeX is an excellent editor for the creation of poster presentations. When writing a poster with LaTeX, several options are available. Here we would like to present some of these options and in particular the *a0poster* class and Brian Amberg's poster template. We shall introduce the basics as well as some useful packages and techniques to make your poster look *nice*. You can even choose to write your poster sequentially or up from different text blocks positioned absolutely or relatively within the page

Contents

1 The *a0poster* class

1.1 Learning the basics

The *a0poster* package [1] has been developed by Gerlinde Kettl and Matthias Weiser and allows to write posters in different sizes in an easy way. The package is composed of the following files:

`a0poster.cls`	Defines the class file
`a0size.sty`	Defines the font sizes
`a0_eng.tex`	Manual in English
`a0.tex`	Manual in German

In theory it allows you to define posters of size A4 to A0b, but it practice there is some problem with scaling when we define posters of size A4 to A1. To my knowledge, this problem still exists so we are going to use just the A0 size and a workaround will be shown in section 1.3 to obtain smaller sizes.

a0poster is a class like *article*, *book* and many others in LaTeX and is used in a similar way. Therefore, the beginning of a TeX file would be something like

```
\documentclass[portrait,a0,final]{a0poster}
\begin{document}
Write poster here
\end{document}
```

Now, you can write the content of your poster as you would write any other document in LaTeX. If you prefer your poster to be in landscape mode, just replace the option *portrait* with *landscape*.

All that remains to be done is to compile your document in the usual way using LaTeX and dvips.

```
latex your_doc.tex
dvips your_doc
```

Although pdfLaTeX can also be used, we prefer LaTeX here in order to use some nice effects of the package *PStricks* described in next section.

The only disadvantage of the *a0poster* class is that you have to think carefully the position of text and graphics because the poster is written sequentially. We will learn how to make posters from different text blocks that are absolutely or relatively positioned on the main page in sections 2 and 3.

1.2 Adding some eye candy to your poster

Now that we know how to write a poster, we would like to add some colors and other effects so that the final poster looks *nice*.

1.2.1 Dividing the poster into columns

First, let us divide the page in several columns. This could be done using the package *multicol* but we prefer here to use the *minipage* environment instead, as this will be more versatile and really useful in many situations. The environment *minipage* allows to virtually include a small page inside your document. You can think of it as a small text box placed in your document. The syntax of the *minipage* environment is as follows

```
\begin{minipage}{length}
  Write here
\end{minipage}
```

where *length* is a given length. One suitable way to define the length of the *minipage* is to use a fraction of the text width. Thus, suppose that you want to define a *minipage* of size equal to the half of the available line width, just write

```
\begin{minipage}{0.5\linewidth}
  Write text here
\end{minipage}
```

So, to make a two-column poster (which makes each column of size $1/2 = 0.5$ the available text width), we write

```
\begin{minipage}{0.49\linewidth}
  First column
\end{minipage}
\begin{minipage}{0.49\linewidth}
  Second column
\end{minipage}
```

Take care not to add extra blank lines between the end of one *minipage* and the beginning of the following or you will not obtain the desired effect. Usually

it is better not to use all the line space available otherwise we obtain one *minipage* under the other one. That's the reason we use 0.49 times the line width.

You can include a *minipage* environment within another *minipage* environment so that the possibilities are endless. For instance, the following text

```
\begin{center}
\begin{minipage}{0.8\linewidth}
  \hrule

   **********Writing two columns**********
  \begin{minipage}{0.3\linewidth}
    This is the first column
  \end{minipage}
  \hspace{0.09\linewidth}
  \begin{minipage}{0.6\linewidth}
    This is the second column and it is subdivided into three columns

    \begin{minipage}{0.3\linewidth}
      One
    \end{minipage}
    \begin{minipage}{0.3\linewidth}
      Two
    \end{minipage}
    \begin{minipage}{0.3\linewidth}
      Three
    \end{minipage}
  \end{minipage}
  \hrule
\end{minipage}
\end{center}
```

produces

**********Writing two columns**********

This is the first column

This is the second column and it is subdivided into three columns

One Two Three

1.2.2 Adding some color

Previous sections should allow you to already obtain a suitable poster for your needs, but we would like our poster to have some nice colors and other effects. We are going to include two new packages by writing

`\usepackage{pstricks,pst-grad}` before the `\begin{document}` line.

PStricks is a set of macros that allows the inclusion of PostScript drawings directly and *pst-grad* is used to fill with colour gradients, using *PStricks*.

With the aid of this package, we can now define any color we want using the RGB scale, just use in the preamble the folowing syntax

```
\newrgbcolor{colorname}{r g b}
```

where *r,g,b* are numbers between 0 and 1 that express the quantity of red, green and blue we add to create our color. For example

```
\newrgbcolor{lightviolet}{0.8 0.3 0.7}
```

We can now obtain any word by writing `{\lightviolet any word}`. The gray-scales *black, darkgray, gray, lightgray,* and *white*, and the colors *red, green, blue, cyan, magenta,* and *yellow* are predefined in *PStricks*.

You may also want to have a look to the package *xcolor* which provides easy driver-independent access to several kinds of colors, tints, shades, tones, and mixes of arbitrary colors already predefined in the package. The add-on package *dvips* with the option *dvipscolor* defines also some extra colors and may be useful as well.

Once we have defined our favourite colors, let as introduce a fancy way to include titles. For this purpose we may define in the preamble the new command *fancytitle*

```
\newcommand{\fancytitle}[1]{
  \begin{center}
    \psshadowbox[linewidth=2mm,framearc=0.1,linecolor=blue,
                 fillstyle=gradient,gradangle=0,gradbegin=white,
                 gradend=lightviolet,gradmidpoint=1.0,framesep=1em]
                 {#1}
   \end{center}
   \vspace{0.015\textheight}
}
```

This will allow to use the command `\fancytitle{My title}` anywhere within the body of the document, which will give

My title

You can try changing the options passed to *psshadowbox* in the command definition in order to obtain different results. In particular,

linewidth	Gives the width of the line that surrounds the box
framearc	Increase in order to have the edges more rounded
linecolor	This is the color of the edges of the box
gradbegin, gradend	These are the colors used to fill the box

The command *psshadowbox* can also be used to write some text inside a box. For example,

```
\psshadowbox[linewidth=1mm,framearc=0.1,linecolor=red,framesep=1em]{
     \begin{minipage}[h]{0.5\linewidth}
       {\bf Fermat's last theorem}

       If an integer $n$ is greater than $2$, then the equation
       $$ a^n+b^n=c^n $$
       has no solutions in non-zero integers $a$, $b$, and $c$.
     \end{minipage}
 }
```

will give

Fermat's last theorem
If an integer n is greater than 2, then the equation

$$a^n + b^n = c^n$$

has no solutions in non-zero integers a, b, and c.

You can even define a background color for the poster. Just write the following after `\begin{document}`

```
\psframe[fillstyle=gradient,framearc=0.05,linewidth=1mm,gradmidpoint=0,
 gradbegin=cgradbegin,gradend=cgradend](0,0)(1.\textwidth,-1.\textheight)
```

where the colors *cgradbegin, cgradend* are colors already defined.

Combining all these techniques, we can obtain posters like the one shown in Figure 1

PStricks allows many other possibilities. We refer to [2] for more information.

1.3 Obtaining smaller size posters

As we have said, the option for making posters from A4 to A1 does not seem to work properly. Nevertheless, we can always rescale the poster to smaller size when printing.

There is also a workaround to exactly obtain the desired size.

First you have to know that when you generate your `poster.dvi` with the command `latex poster`, you also generate a file called `a0header.ps` with the content:

```
%%BeginFeature *PageSize ISOA0/ISO A0
2 dict dup /PageSize [2380 3368] put dup /ImagingBBox null put
setpagedevice
%%EndFeature
```

This file (`a0header.ps`) is linked to `poster.dvi` and contains information related to the size of the poster. Now, suppose we want a A1 size poster. Before rescaling our poster, modify the `a0header.ps` file into:

```
%%BeginFeature *PageSize ISOA1/ISO A1
2 dict dup /PageSize [1684 2380] put dup /ImagingBBox null put
setpagedevice
%%EndFeature
```

Where we have changed the size of the page in bp (big points). See Table 1. Then, we can proceed with the scaling step:

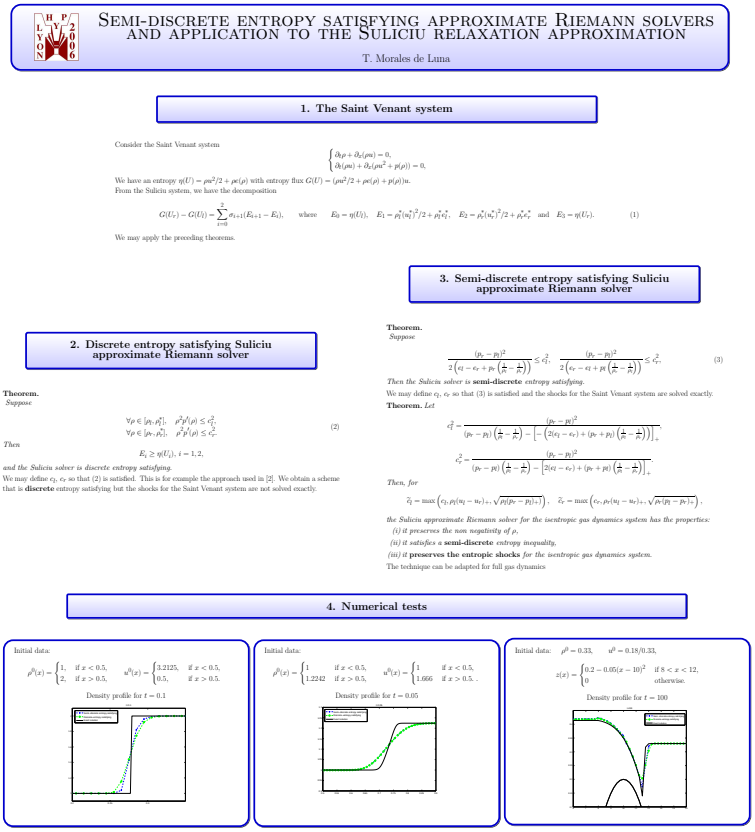

Figure 1: A poster Example

```
dvips -O '-6bp,-6bp' -T595mm,841mm -x694 poster.dvi -o poster.a1.ps
```

Where:

-T595mm,841mm : the new poster dimension in mm. Here the poster is in portrait mode. To rescale a landscape poster, use: -T841mm,595mm.

-x694 : the scaling factor. 1000 is a scaling factor of 1 (no change). From A0 to A1, the scaling factor is $1/\sqrt{(2)} = 0.707$ (See Table 2). To avoid to create new set of fonts (and to save space), it is recommended to use the closest standard LaTeX magnification factor (694 here). Note that from A1 to A0, use 1440. Have a look at Table 3. Read also [3] for more information.

-O '-6bp,-6bp' : the ps file obtained with dvips includes a margin of 1in = 72bp by default. In order to center the poster, we change the paper offset with the corresponding value given by Table 4.

poster.dvi : the input file (in A0 size)

-o poster.a1.ps : the output file

To obtain the PDF file, just run ps2pdf:

```
ps2pdf poster.a1.ps poster.a1.pdf
```

Table 1: Paper Size

	w bp	**h bp**	**w in**	**h in**	**w mm**	**h mm**
A0b	2594	3370	36,03	46,81	915	1189
A0	2380	3368	33,06	46,78	840	1188
A1	1684	2380	23,39	33,06	594	840
A2	1190	1684	16,53	23,39	420	594
A3	842	1190	11,69	16,53	297	420
A4	595	842	8,26	11,69	210	297
A5	421	595	5,85	8,26	149	210
A6	298	421	4,14	5,85	105	149
A7	211	298	2,93	4,14	74	105
A8	149	211	2,07	2,93	53	74
A9	106	149	1,47	2,07	37	53
A10	75	106	1,04	1,47	26	37

Table 2: Scaling factors

from \ to	A0b	A0	A1	A2	A3	A4	A5	A6
A0b	1	0,92	0,65	0,46	0,32	0,23	0,16	0,11
A0	1,09	1	0,71	0,5	0,35	0,25	0,18	0,13
A1	1,54	1,41	1	0,71	0,5	0,35	0,25	0,18
A2	2,18	2	1,42	1	0,71	0,5	0,35	0,25
A3	3,08	2,83	2	1,41	1	0,71	0,5	0,35
A4	4,36	4	2,83	2	1,42	1	0,71	0,5
A5	6,16	5,65	4	2,83	2	1,41	1	0,71
A6	8,7	7,99	5,65	3,99	2,83	2	1,41	1

Table 3: LaTeX magnification factor

from \ to	A0b	A0	A1	A2	A3	A4	A5	A6
A0b	1000	913	634	440	335	233	162	112
A0	1095	1000	694	482	367	255	177	123
A1	1577	1440	1000	694	482	367	255	177
A2	2272	2074	1440	1000	694	482	367	255
A3	2986	2726	2074	1440	1000	694	482	367
A4	4300	3925	2726	2074	1440	1000	694	482
A5	6192	5652	3925	2726	2074	1440	1000	694
A6	8916	8139	5652	3925	2726	2074	1440	1000

2 Using text blocks with absolute positioning

When writing a poster with the *a0poster* class, one has to have in mind the final distribution of the different parts we have written and sometimes it may be difficult to obtain the text exactly where we want. In this section we would like to introduce another *philosophy* for composing the poster. Instead of composing the poster sequentially, we are going to write a poster by composing small pieces of text boxes that are pasted over the big poster sheet exactly where we want. This will be possible by using the *textpos* package.

Just write in the preamble of the document

Table 4: Offset correction

from \ to	A0b	A0	A1	A2	A3	A4	A5	A6
A0b	0	0	-7	-16	-61	-60	-60	-60
A0	1	0	-6	-16	-61	-60	-59	-61
A1	11	9	0	-11	-22	-57	-58	-59
A2	37	33	17	0	-14	-27	-53	-56
A3	183	167	46	20	0	-17	-30	-51
A4	255	233	155	55	24	0	-18	-32
A5	367	335	226	145	62	26	0	-19
A6	538	491	335	221	139	66	28	0

```
\usepackage[absolute,overlay]{textpos}
```

The *textpos* package will allow to place text boxes at a given *absolute* postition by creating a virtual grid over the page. The *absolute* option makes the origin of the grid on which text blocks are positioned the upper left hand corner.

overlay gives the text blocks opaque backgrounds. Without the *overlay* option, the background of the text blocks are transparent. You can change the background color of a given *textblock* by writing `\textblockcolour{color_name}`.

It is also useful to add the option *showboxes* when you are writing the poster. This will draw a rectangle around your text block making much easier to correctly place the block.

The package defines the environment *textblock* which is used as follows

```
\begin{textblock}{hsize}(hpos, vpos)
  Some text
\end{textblock}
```

The *hsize* and *hpos* arguments are given in units of a module `\TPHorizModule` and *vpos* is given in units of a module `\TPVertModule`. For example,

```
\begin{textblock}{20.5}(1.5, 2.5)
  Some text
\end{textblock}
```

will place a text box of width $20.5 \times TPHorizModule$ so that its left upper corner is placed $1.5 \times TPHorizModule$ to the right and $2.5 \times TPVertModule$ to the bottom from the left upper corner of the poster.

We are going to set these units to $1cm$ by placing the following code in the preamble

```
\setlength{\TPHorizModule}{1cm}
\setlength{\TPVertModule}{1cm}
```

A little trick that helps placing the text blocks is to define a grid over the page. You can do so by loading the package *eso-pic* [5]

```
\usepackage[colorgrid,texcoord]{eso-pic}
```

This will produce something similar to Figure 2.

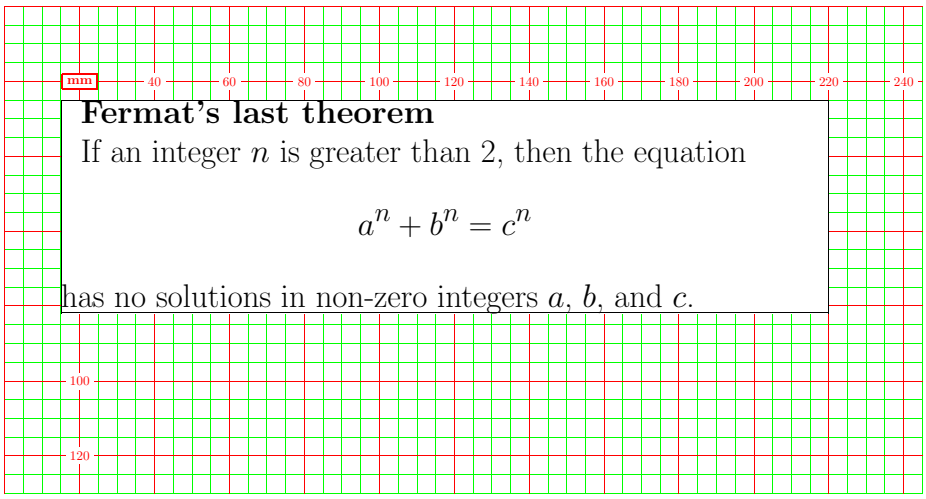

Figure 2: Zoom of poster page with a grid

3 Brian Amberg's poster template

Brian Amberg has created another class for posters in LaTeX. This class presents the advantage of creating the poster directly from small blocks with relative positioning. While it presents an easy way of creating posters, it leaves you less

choice to customise what your poster looks like, and sometimes you will have to edit the class file directly (always remembering to save it under a different name!) in order to make changes. This class has to be used with pdfLaTeX.

You can use it as usual

```
\documentclass[portrait,final]{baposter}

\begin{document}
\background{}
  \begin{poster}{
     <poster options>
   }
  Boxes definitions\ldots
  \end{poster}
\end{document}
```

The command `\background{}` has to be defined just after the `\begin{document}` and it lets you define a background for the poster. You can have a look at some of the templates provided by Brian Amberg to see some examples.

Then you have to write down the *poster* environment where you select the options for your poster. A typical example would be

```
   \begin{poster}{
  grid=no,
  colspacing=0.7em,
  color=orange,
  colortwo=white,
  textborder=roundedleft,
  eyecatcher=no,
  headerborder=none,
  headershape=roundedright,
  }
```

The possible options you can select are given in Table 5.

Once we have set the options for the poster, we begin writing the poster by composing different text blocks that are introduce with the command *headerbox*

Option	Values	Meaning
`grid`	yes | no	Shows a grid to help you place text boxes
`colspacing`	numeric	Defines the spacing between columns
`color`	color value	Color definition used as the main color of the poster
`colortwo`	color value	The other color for gradient based layouts
`textborder`	none, rectangle, rounded, ...	The style of the box around the text area
`headerborder`	none, closed, open	No extra border, full border around or border that is open below for box header
`headershape`	rectangle, rounded, roundedleft, roundedright	Shape of the box-header region

Table 5: Options for *poster* command

.

```
\headerbox{title}{name and position}{content}
```

This command takes three arguments. The first and the third are respectively the title and the content of the text block. In order to specify where the box will be placed, we shall use the second argument.

Each box has a name and can be placed absolutely or relatively. To specify the name we just write `name=custom_name` inside the second argument. This name has to be unique.

Then, if you want to place the box in an absolute position, you will have give the row and column number. By default, your poster will be divided into three columns (this option can be changed by editing the class file). Thus,

```
\headerbox{Contribution}{name=contribution,column=0,row=0}{
    content...
}
```

will place the text box named *contribution* in first column and first row. Remark that first row and first column correspond to 0.

But, you can also define a relative position. For example, to place a box just under the one named *contribution*, write

```
\headerbox{Model}{name=model,column=0,below=contribution}{
    some other text...
}
```

The possible choices for relative positioning are:

`below=`	name of other node
`above=`	name of other node
`aligned=`	name of other node
`bottomaligned=`	name of other node

You can even make the boxes to span several columns. For example

```
\headerbox{Results}{name=results,column=1,span=2,
           below=results neutralization,above=funding}{
    text...
}
```

will place a box named *results* that begins on second column and span to the third, placed under the box *results neutralization* and above the box *funding*. Remark that the referenced boxes have to be defined before.

By using this class, you can get pretty sofisticated posters like the one shown in Figure 3.

4 Making big posters from smaller pages

One can decompose a big paper size poster into smaller ones. For example, we could decompose one A3 poster into two A4 pages that may be printed normally and then stick them together.

The idea is to use the package *pdfpages* and pdflatex. As an example, say that we have one A3 poster of size 842×1190 in *bp*, and we would like to print it in two pages A4 of size 596×842. See Figure 4.

To do so, we first save the A3 poster into the file `input.pdf` and then we write into a file named, for example, `make-a4-from-a3.tex` the following

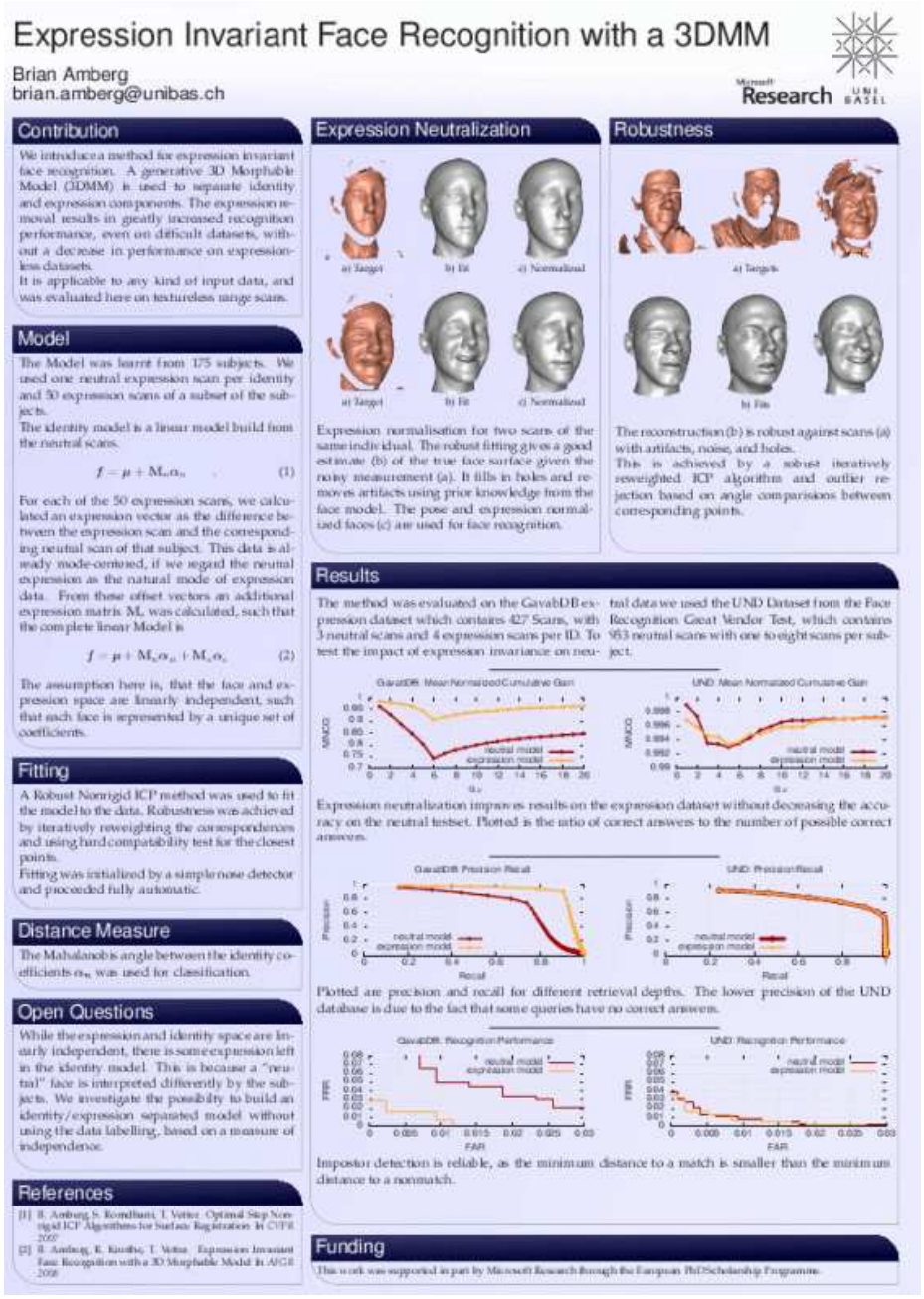

Figure 3: A poster created by Brian Amberg

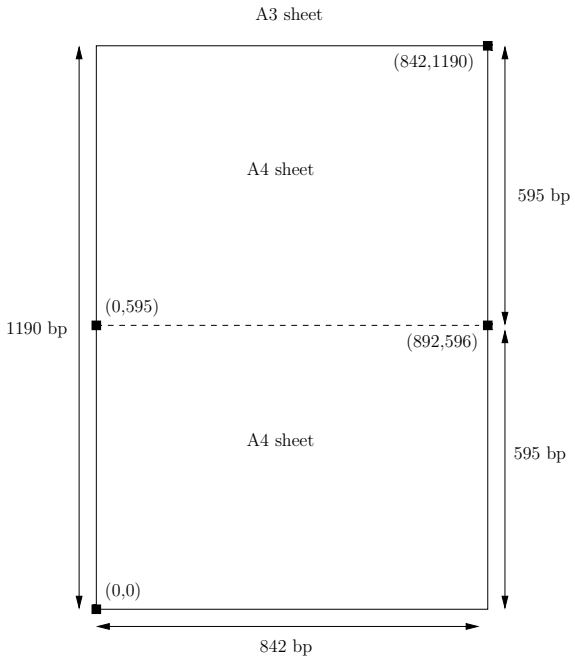

Figure 4: Dividing one A3 into two A4

```
\documentclass[landscape,a4paper]{article}

\usepackage{pdfpages}

\begin{document}

\includepdf[viewport=  0 595 842 1190]{input.pdf}
\includepdf[viewport=  0 0 842 595]{input.pdf}

\end{document}
```

We run pdfLaTeX and we obtain the desired result in the file `make-a4-from-a3.pdf`

In a similar way, we can decompose one sheet of A0 into 16 sheets of A4, using a file named, for example, make-a4-from-a0.tex, containing the following code:

```
\documentclass[a4paper]{article}
```

```
\usepackage{pdfpages}
\begin{document}

% top row, left to right
\includepdf[viewport=   0 2526  596 3368]{input.pdf}
\includepdf[viewport= 595 2526 1192 3368]{input.pdf}
\includepdf[viewport=1190 2526 1788 3368]{input.pdf}
\includepdf[viewport=1785 2526 2384 3368]{input.pdf}

% 2nd row, left to right
\includepdf[viewport=   0 1684  596 2526]{input.pdf}
\includepdf[viewport= 595 1684 1192 2526]{input.pdf}
\includepdf[viewport=1190 1684 1788 2526]{input.pdf}
\includepdf[viewport=1785 1684 2384 2526]{input.pdf}

% 3rd row, left to right
\includepdf[viewport=   0  842  596 1684]{input.pdf}
\includepdf[viewport= 595  842 1192 1684]{input.pdf}
\includepdf[viewport=1190  842 1788 1684]{input.pdf}
\includepdf[viewport=1785  842 2384 1684]{input.pdf}

% bottom row, left to right
\includepdf[viewport=   0    0  596  842]{input.pdf}
\includepdf[viewport= 595    0 1192  842]{input.pdf}
\includepdf[viewport=1190    0 1788  842]{input.pdf}
\includepdf[viewport=1785    0 2384  842]{input.pdf}

\end{document}
```

5 Conclusion

We have now seen some of the many options available to write posters in LaTeX. It is now up to you to decide which ones adjusts better to your needs. My advice would be would be to play a bit with the *a0poster* class first as this is the simplest

one and nearest to other classes like *article*. This class is especially suitable when you have a rather fixed distribution of your poster in mind, combined or not with the *textpos* package. Brian Amberg's poster class is suitable when you are not sure of the final distribution or when you want to easily modify the position of the text boxes. But of course, the final choice is up to you.

References

[1] *The a0poster class and manual*.
http://www.ctan.org/tex-archive/macros/latex/contrib/a0poster/

[2] *The PStricks package*.
http://www.ctan.org/tex-archive/graphics/pstricks/

[3] Pierre Poulain. *How to make a LaTeX A0/A1 poster*
http://pierrepoulain.free.fr/poster/poster.php

[4] *The textpos package*
http://www.ctan.org/tex-archive/macros/latex/contrib/textpos/

[5] *The eso-pic package*.
http://www.ctan.org/tex-archive/macros/latex/contrib/eso-pic/

[6] Brian Amberg. *LaTeXposter template*.
http://www.brian-amberg.de/uni/poster/

[7] *The pdfpages package*
http://www.ctan.org/tex-archive/macros/latex/contrib/pdfpages/