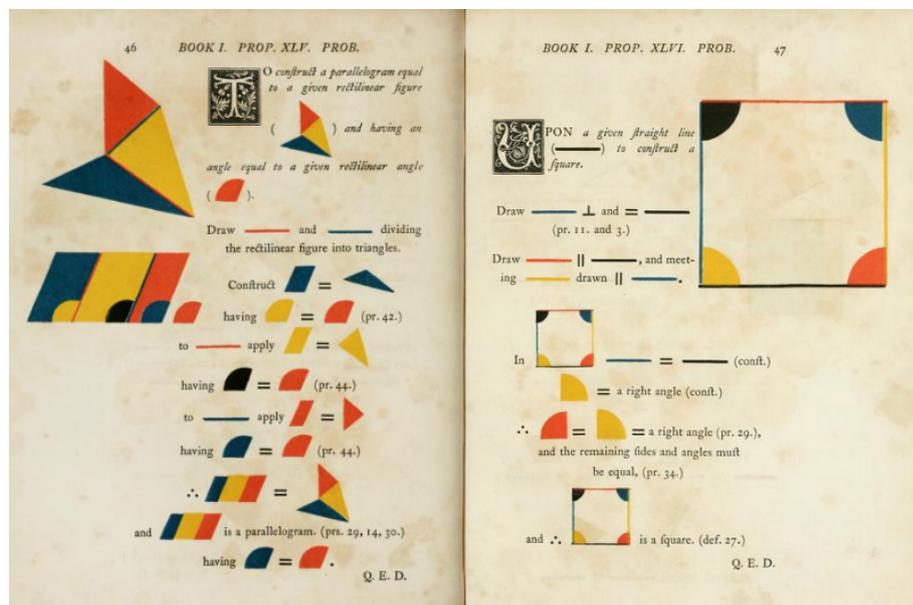


# Oliver Byrne's "The first six books of the Elements of Euclid" in Con $\TeX$ t and MetaPost

The most famous book of an Irish mathematician Oliver Byrne is his 1847 rendition of the first books of Euclid's "Elements." What makes this book stand out is that instead of ordinary letter designations such as "triangle ABC," miniature pictures directly in

the text are used, like this . As fancy as it looks, it's actually quite a nice way to deal with referencing diagram features without overwhelming the reader with a lot of mediatory letters. On the writer side, however, this approach presents more challenges than the traditional one, since typesetting tools tend not to be suited to be used in such a manner.



GUI tools aren't especially easy to extend, but  $\TeX$  and its extensions such as  $\LaTeX$  or Con $\TeX$ t combined with MetaPost or a similar software provide an excellent environment to add the tools needed to recreate such a book. To prove that this is indeed the case I chose Con $\TeX$ t and MetaPost combination to recreate Byrne's book and to make appropriate tools to potentially apply Byrne's approach to different situations.

## The overall structure

Byrne has made only 6 of 13 books of the "Elements." Each book mostly consists of "propositions"—theorems and problems. Most propositions have diagrams (usually one per proposition) which are referenced in proposition texts.

For diagrams, I made a ConTeXt macro that creates a new MetaPost instance. And in MetaPost, some functions to create these constructions. Their use looks somewhat like this:

```
\defineNewPicture{ % Inside this thing you put the construction
  pair A, B, C, D; % MetaPost has a special type of variables
                  % for coordinates

  numeric d;
  d := 2u;
  A := (0, 0);
  B := A shifted (d, 0); % Point coordinates are set here
  C := A shifted (0, -d);
  D := A shifted (d, -d);
  byAngleDefine(B, A, C, byblack, 0); % These define angles:
  byAngleDefine(D, B, A, byblue, 0); % firstangle points,
  byAngleDefine(C, D, B, byred, 0); % then color,
  byAngleDefine(A, C, D, byyellow, 0); % then style.
  draw byNamedAngleResized(); % This thing draws all the angles.
  byLineDefine(A, B, byred, 0, 0); % These define line segments:
  byLineDefine(B, D, byyellow, 0, 0); % firstends,
  byLineDefine(D, C, byblack, 0, 0); % then color and style,
  byLineDefine(C, A, byblue, 0, 0); % then thickness.
  draw byNamedLineSeq(0)(AB,BD,DC,CA); % This thing draws a sequence
                                      % of lines.
}
\drawCurrentPicture % And this thing draws the whole picture.
```



For proposition texts, I made a series of macros that draw pictures in the same MetaPost instance and based on the existing diagram. In general, they execute arbitrary MetaPost code, but most of the time, they take object names as arguments. Like this:

```
% Names are automatically assigned to line segments but can be assigned
% manually.
Draw $\drawUnitLine{CA} \perp \mbox{ and } = \drawUnitLine{DC}$.\\
Draw $\drawUnitLine{AB} \parallel \drawUnitLine{DC}$.\\
and meeting \drawUnitLine{BD} drawn $\parallel \drawUnitLine{CA}$.

Draw   $\perp$  and  $=$  .
Draw   $\parallel$  ,
and meeting  drawn  $\parallel$  .
```

This is how these parts work together:



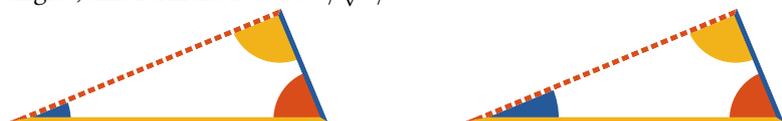
### Some features

Pictures in the book look simplistic, but not to the point when you can get away with just using the default drawing commands.

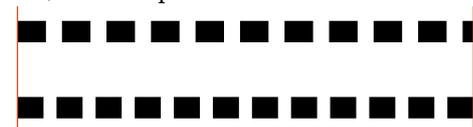
The touching points of line segments for example need some nice representation. For now, connection of only two lines is supported, but other lines can simply be put beneath such a connection.



To depict angles Byrne mostly uses circular sectors. If an angle is small enough, the sector with the same radius may look tiny so it makes sense to enlarge it. Currently, the radius stays the same for angles above 60 degrees, and for smaller angles, this formula is used:  $r/\sqrt{a/60}$

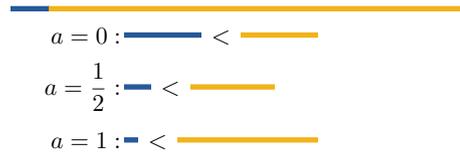


Byrne uses dashes and line thickness along with the colors to differentiate the lines. Dashed lines to look nice should begin and end with full dashes. To achieve this, the dash pattern is scaled a bit to fit a line length.

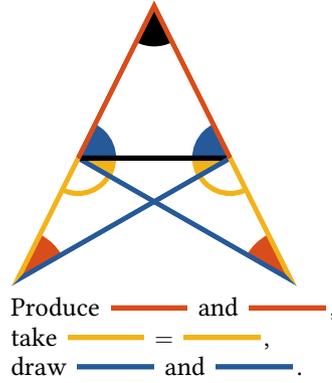


When Byrne references a line segment in text, he draws it always horizontally retaining the color (or colors) and style of the original line on a diagram. With respect to the length he either makes all the lines in the text the same length, or tries to keep relative lengths to some degree. To achieve this I use this formula:  $L_{text} = (L_{diagram}S)^a L_{desired}^{1-a}$  where  $0 \leq a \leq 1$  and  $S$  is the scale factor for in-text graphics.

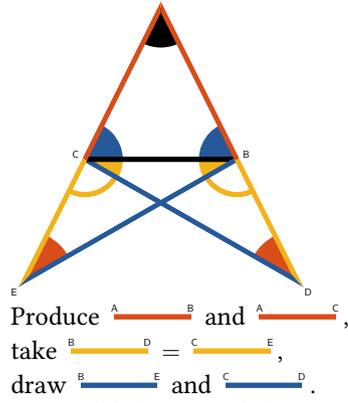
When  $a = 1$  the length of the segment in text  $L_{text}$  is equal to that on the diagram  $L_{diagram}$  times  $S$ . When  $a = 0$ ,  $L_{text}$  is always equal to some desired length  $L_{desired}$ . And for values of  $a$  in between  $L_{text}$  retains the relative sizes of the lines on the diagram, but makes shorter than  $L_{desired}$  lines a bit longer and vice versa. You may want this if you are about to retain relative line lengths while avoiding 2 mm lines next to 2 cm ones.



Byrne didn't use letter designations for points except for in the Introduction. But even in his own book he couldn't avoid using the same colors and styles for different lines causing potential confusion. Take this, for example:



Edward Tufte in his "Envisioning Information" has pointed out that letters wouldn't do any harm to Byrne's method and on the contrary would be very useful. To check if this is true, I added optional text labels, which you can turn on by setting `textLabels := true`.



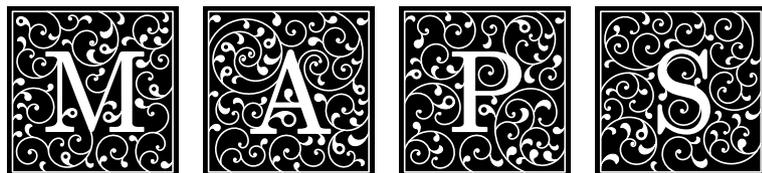
Text labels aren't fully automatic on the main diagram and you have to specify their placement like so:

```
% labels will be placed at the vertices of ECABDCB polygon
% note that labels on C and B are placed only once
draw byLabelsOnPolygon(E, C, A, B, D, C, B)(0, 0);
```

On the inline pictures labels are placed automatically in most cases.

## Initials

One thing Byrne's book shares with many other books of the same period is the extensive use of initials. Not being an overly important feature, initials are still fun and I decided to keep them. There are plenty of fonts with initials out there including the one which comes with EB Garamond used in this book, but not many of them contain letters outside the English alphabet, which is an issue, because the Russian translation requires Cyrillic letters. So instead of looking for an appropriate font or drawing one myself I chose to try and make some kind of procedurally generated initials substitute in a way that would allow me to get any letter without drawing it manually and would make all the letters decorated with slightly different ornaments.



The algorithm is pretty straightforward: curls are placed on the parts of a letter and on the frame, as large as they can grow. This is done several times. All the curls are included in the subsequent iterations. After that, some "leaves" are grown in the same manner, filling the remaining gaps. It's also quite slow, so initials generator is located in a separate MetaPost file and it processes a text file with the list of all the initials and vignettes created when the book is processed.

The results aren't fully satisfying, so it's possible to put PDF or SVG files with any initials instead of generating ones.

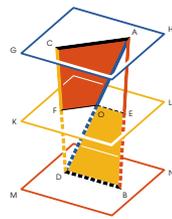
## Translation

Having in mind the flexibility of both the diagrams and the initials, translating this book to Russian was not more difficult than any other book would be. At first, I used the well-known Murduhai-Boltovskoi's translation as a reference but soon found that Byrne changed too much in Euclid's theorems for a reference translation to be very useful. The process of translation helped catching bugs, both mine and Byrne's. For instance, the diagram for proposition 9 in the sixth book doesn't match the text, making the proof incorrect, so I had to change it.

Marcin Ciura from Poland started to make a Polish translation of the book. He have already found even more typos and bugs and made many extremely valuable contributions on the way (scaling angles, for instance, was his idea).

## Plans

Byrne's book doesn't contain any solid geometry (save an image of a parallelepiped in the Introduction). Neither did I make any tools for it, but at some point, I decided to add some, so I started to "byrnify" books 11–13 to have something to use the new tools on. For now, only a bit more than half of the 11th book is roughly made, along with some functions to make solid constructions and project them onto the screen plane. Solid constructions, though, on the average are substantially more complex than plane ones, and I'm not yet sure if Byrne's approach would fit them well and if MetaPost is a reasonably convenient tool for the job.



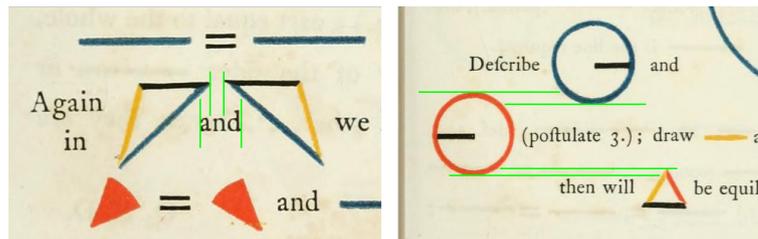
*If two straight lines be cut by parallel planes, they will be cut in the same ratios.*

For let the two straight lines  $^A B$ ,  $^C D$  be cut by the parallel planes  $^\alpha$ ,  $^\kappa$ ,  $^\lambda$ ,  $^\mu$  at the points  $^A$ ,  $^E$ ,  $^B$  and  $^C$ ,  $^F$ ,  $^D$ .

I say that,  $^A E : ^E B :: ^C F : ^F D$ .

For let  $^A C$ ,  $^B D$ ,  $^A D$  be joined, let  $^A D$  meet the  $^\kappa$  at the point  $^O$ , and let  $^E O$ ,  $^O F$  be joined.

Pictures in text oftentimes need some kerning. Similarly, yet trickier, taller pictures on adjacent lines spread the lines very wide, which is appropriate only if these pictures collide. I don't yet know how and if I can make an automated solution for these issues, but I definitely want to because fixing them by hand is really tedious.



MetaPost can also be used from within L<sup>A</sup>T<sub>E</sub>X and as a standalone program. In the future, I plan to make L<sup>A</sup>T<sub>E</sub>X macros similar to ConT<sub>E</sub>Xt ones, to allow using them there too.

The code as of now is poorly documented and this issue is high on my priority list.

Last but not least, I want to apply the tools to something more modern and practical than “the Elements.”

The code is licensed under GNU GPLv3 or later and the book and its translation are CC-BY-SA 4.0. Both can be found here: [github.com/jemmybutton/byrne-euclid](https://github.com/jemmybutton/byrne-euclid)

Sergey Slyusarev