

Single-source publishing in multiple formats for different output devices*

Petr Sojka

Faculty of Informatics, Masaryk University
Botanická 68a, 60200 Brno, Czech Republic
sojka@fi.muni.cz
<http://www.fi.muni.cz/usr/sojka/>

Michal Růžička

Faculty of Informatics, Masaryk University
Botanická 68a, 60200 Brno, Czech Republic
xruzick7@fi.muni.cz

Abstract

TeX is traditionally used as an authoring tool for paper publishing of scientific texts and textbooks. Parallel electronic publications that are meant for on-screen viewing and web delivery are demanded by readers for many reasons today. The paper discusses ways to single-source author publishing from L^ATeX source file, and shows examples of several textbooks published by this approach. Special attention is given to the web document generation either to HTML or XHTML markup with the notation translated to MathML.

On-the-fly personalised document generation with JBIG2-compressed pictures for a digital library project DML-CZ is discussed as well.

1 Motivation

*Discover the outer logic of the typography
in the inner logic of the text.*
— Robert Bringhurst [5]

Documents conveying information have their *content* and *form*. Form (appearance) should reflect the *design*, it should use the graphical means consistently. Possibilities of a form of a document are constrained by an *output devices* (paper, LCD monitor, PDA).

It is well-known, but little respected fact, that design of a document has to be (re)done for every new output device. Many documents fine-tuned with our TeX-based systems for reading on a paper (microtypography etc.) are often proudly posted on the web without redoing the design phase for particular output device (LCD screen, PDA), for particular purpose and for specific readers' requests. This is in contrast with the goal Knuth had in mind e.g. when designing the fonts in METAFONT: even the tiny rasterization details affected by different printers should be fine-tuned by proper settings for particular printer in `modes.mf`.

The authors that use open-sourced TeX-based system have significant power and possibilities to in-

fluence every detailed aspect of the form when writing their papers and books. In the case authors write using logical markup only it is then possible to choose different typography that honor content independently to suit different output device qualities by changing the design mapping for logical entities in the text. Strict separation of content and form is possible almost always, only with rare exceptions as typesetting Christian Morgenstern's poems. However, author's discipline not to use visual typesetting commands as `\vskip` is necessary.

As the quality and size of display devices grow, even longer texts are going to be read directly on screen, or document snippets on the XHTML browsers in mobiles or PDAs. Authors naturally demand their content to be ready for the wide range of different output devices. Although the design is inherently harder as usually the whole class of output devices and browsers has to be thought of, it is what is demanded by the readers.

In this paper we comment on several publishing projects: two textbooks [2, 3] and database publishing in DML-CZ [10]. In all projects benefits from strict separation of form and content to produce different products are shown, using the single-source input created by author or generated from database on demand for different types of output.

* Support of Czech Academy of Sciences grants AV1ET-208050401 and AV1ET200190513 is acknowledged, as well as travel support of the Czechoslovak TeX Users Group.

2 Single-source publishing

*If the only tool we know is a word processor,
everything looks like a print document.*
— Peter Meyer [9]

Author wants to convey information, and the only thing she or he has to do is to mark logical entities in the text. Designer should enforce sameness: visual rendering of the same logical parts should be consistently same. This will allow publishing for different output devices just by switching between different designs. Maintenance of the text by the author will be much more easier and cheaper than maintaining forked versions for different purposes. It will also reduce errors, improve consistency and/or save translation costs: the term *single-source publishing* or *single-sourcing* is used for this type of document authoring (http://en.wikipedia.org/wiki/Single_source_publishing). The well-known system and DTD allowing single-source publishing is DocBook, with support for conversion to XHTML, DVI, PostScript or PDF, either with XSL, XSL-FO or L^AT_EX.

Many single-source publishing approaches start from XML, as the source of content [9]. For many authors, it is simply not convenient to write directly in XML, even with clever XML editors. Technical manuscripts full of mathematics will remain to be authored in some flavor of T_EX for the compactness and clarity of T_EX math notation. Author productivity raises significantly with author-centric system [8].

In academia, authors write textbooks for their courses, and want to publish them in the formats their students prefer. We have prepared two single-source textbooks [2, 3] from sources originally written in L^AT_EX. In the next sections we describe our experience with the project.

3 Mark up and conversion tools

*Data cannot be used at a finer grain
than it is marked up at.* — Rick Jelliffe

To allow single-source publishing, we had to clean the source files significantly, as they were not written from the beginning with the goal to publish in different formats. Even DEK writes very “low-level” code in his T_EXbook:

```
&\elevenit I\kern.7ptllustrations by\cr
&DU\kern-1ptANE BIBBY\cr
\noalign{\vfill}
&\setbox0=\hbox{\manual77}%
\setbox2=\hbox to\wd0{\hss\manual6\hss}%
\raise2.3mm\box2\kern-\wd0\box0\cr % A-W logo
&ADDISON\kern.1em--WESLEY\cr
%&PUBLISHING COMP\kern-.13emANY\kern-1.5mm\cr
```

He did not expect the code to be used for anything else than printing to phototypesetter, with given fonts, kerning for given size, etc.

For single source publishing, the main text has to be written without fine-tuning for single output device/printer, and low-level mark up has to be substituted by high-level one allowing multiple macro definitions for different outputs. Marks has to be written at the finest grain possible, expanding to the appropriate design setting for every type of output. The idea is not new, and it is used in the XML word as well (different CSS rendering for different devices or even browsers).

We have identified several types of output format our students have demanded. In addition to the standard version suitable for printing on paper, a searchable version optimised for LCD screen with 4:3 aspect ratio was requested. For some purposes (X)HTML version was needed for platforms and devices without PDF renderer. Finally, we prepared a XHTML+MathML version as well.

There are many tools and utilities to convert T_EX documents to different output formats—list if them is compiled at T_EX Users Group web page *TeX Resources on the Web* (<http://www.tug.org/interest.html>). PdfL^AT_EX with *hyperref* and *crop* package is suitable combination for print output. For on-screen version of document we have chosen *pdfscreen* package in combination with pdfT_EX and *hyperref* package.

4 PDF versions

*Make all visual distinctions as subtle as possible,
but still clear and effective.*
— Edward R. Tufte [13]

For every output version, design parameters and macros are, as usual, written in separate conditional branch. Simple source code example:

```
\newif\ifprint
\printfalse % Non-print version.
%\printtrue % Print version.

\ifprint
  \hypersetup{colorlinks=false,
              pdfborder={0 0 0}}
  % Center mirrored document pages on A4
  % paper and add crop marks.
  \usepackage[cam,a4,center,mirror]{crop}
\fi
```

4.1 PDF for printing

For print it is desirable to prepare grey-scale output. For this job is often used package *hyperref* with appropriate options (`colorlinks=false`, `pdfborder={0 0 0}`).

crop package (available on CTAN) is good choice to set up crop marks. *crop* package is also able to perform document transformation as page mirroring etc. The output is shown in Figure 1 on page 128.

4.2 PDF for on-screen viewing

Typical LCD screen is quite different output device. Comparing to current 1200 DPI printers, it has order of magnitude lower resolution. Usually, it has different aspect ratio, and allows colours of high depth. We can add interactive content like hypertext links, navigation tool-bar, etc.

For versions of textbooks designed for screen we started with package *pdfscreen* (available on CTAN). *pdfscreen* code together with definitions of environments and macros for screen are in conditional branch of textbook style file again.

To save to fine-tuning of line-breaking, we have made the line-breaking same in both print and screen versions. Page breaking will be different of course, as seen in Figure 2 on page 128.

5 Versions for the web delivery

A buzzword of today is XML—people spend most time browsing (X)HTML pages. For searchable and scalable math using outline fonts MathML is needed. There are several tools available: TeX2page (<http://www.ccs.neu.edu/home/dorai/tex2page/>), Tralics (<http://www-sop.inria.fr/apics/tralics/>), TeX4ht (<http://www.cse.ohio-state.edu/~gurari/TeX4ht/>), LaTeXML (<http://dlmf.nist.gov/LaTeXML/>) or LaTeX2HTML.

Every tool has some advantages and disadvantages; we do not want to discuss all of them. After testing some of the tools, we have chosen TeX4ht [4, 6]. TeX4ht uses native TeX compiler and process the document with special setup into standard DVI output with markup in `\specials`—there is not danger of omission of unsupported markup as e.g. with LaTeX2HTML. From enriched DVI HTML pages are extracted by TeX4ht scripts.

5.1 HTML

Most difficult part of the conversion setup process was web output. In the case of complex documents it is usually necessary to make some changes in the source code. These modifications and TeX4ht specific commands are, of course, in separate style file whenever possible.

Once we have source code in L^AT_EX format with TeX4ht styles loaded we can try to convert document. The first trial is to run command:

```
htlatex filename.tex 'html'
```

When TeX4ht successfully completes its work we will get HTML document with complicated formulae rendered into PNG images, as seen in Figure 3 on page 129. This way is simple and safe for rendering in all, even old, web browsers.

Whole document is implicitly generated into one HTML file. TeX4ht is able to split longer document into tree of web pages automatically. Command `htlatex filename.tex 'html,2'` will generate document that has each chapter in separate file. Navigation between chapters is realised by tool-bar generated on the top and bottom of every page.

When generating HTML output it may be useful to insert some HTML code with command `\HCode{Some HTML code.}`. This command can be used for CSS code insertion, too. TeX4ht offers both HTML or XHTML output generation.

We existing L^AT_EX logical markup as much as possible. For CSS code definition we use command `\Css{CSS definition}`. CSS attributes are mapped onto documents elements through appropriate `\HCode` commands. Command `\ConfigureEnv` makes it possible to add own code before and after environment content in output document.

As a little bit more complex example of CSS definition could serve theorem environment:

```
\newtheorem{theorem}{Theorem}[chapter]
...
\ifweb % In case of web format output...
  \Css{% CSS code definition block
    .theorem {
      background-color: \#FFFFFF;
      border: 1px solid;
      border-color: \#0000FF;
    }
  }
  % In the resulting HTML document mark
  % "<div class="theorem">" is placed
  % before each "theorem" environment
  % and mark "</div>" after that.
  \ConfigureEnv{theorem}
    {\HCode{<div class="theorem">}}
    {\HCode{</div>}}
    {}{}
\fi
...
% In document, same LaTeX markup is used
% for all versions: "theorem" environment
\begin{theorem}
  Function~$f$ have only one limit
  in point~$[x_{0},y_{0}]$.
\end{theorem}
```

Řešení. Výraz pod odmocninou musí být nezáporný, tj. musí být splněna podmínka

$$\left(\frac{(y-2)^2}{4} + x^2 - 1\right)(x^2 + y^2 - 6x) \geq 0.$$

To nastane, právě když

$$\frac{(y-2)^2}{4} + x^2 - 1 \geq 0 \quad \text{a} \quad (x^2 + y^2 - 6x) \geq 0$$

nebo

$$\frac{(y-2)^2}{4} + x^2 - 1 \leq 0 \quad \text{a} \quad (x^2 + y^2 - 6x) \leq 0.$$

Rovnice $\frac{(y-2)^2}{4} + x^2 = 1$ je rovnicí elipsy se středem v bodě $[0, 2]$ a poloosami délek $a = 1$ a $b = 2$, rovnice $x^2 + y^2 - 6x = 0$ je rovnicí kružnice se středem v bodě $[3, 0]$ a poloměrem $r = 3$, neboť tuto rovnici lze převést na tvar $(x-3)^2 + y^2 = 9$. Množina všech bodů $[x, y] \in \mathbb{R}^2$ splňující výše uvedené nerovnosti, tj. definiční obor funkce f , je znázorněna na vedlejším obrázku. Je to uzavřená množina v \mathbb{R}^2 .

ii) Zobrazte v rovině definiční obor funkce

$$f(x, y) = \arccos(x^2 + y^2 - 1) + \sqrt{|x| + |y| - \sqrt{2}}.$$

Řešení. Definičním oborem funkce \arccos je interval $[-1, 1]$, první sčítanec je tedy definován pro $[x, y]$ splňující nerovnosti

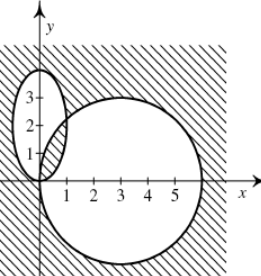
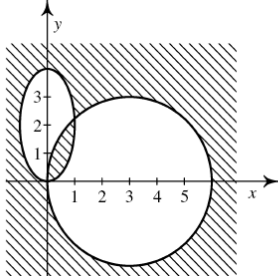
$$-1 \leq x^2 + y^2 - 1 \leq 1,$$


Figure 1: Print output (without page mirroring)

To nastane, právě když

$$\frac{(y-2)^2}{4} + x^2 - 1 \geq 0 \quad \text{a} \quad (x^2 + y^2 - 6x) \geq 0$$

nebo

$$\frac{(y-2)^2}{4} + x^2 - 1 \leq 0 \quad \text{a} \quad (x^2 + y^2 - 6x) \leq 0.$$


obr. 1.1 Definiční obor funkce f

Rovnice $\frac{(y-2)^2}{4} + x^2 = 1$ je rovnicí elipsy se středem v bodě $[0, 2]$ a poloosami délek $a = 1$ a $b = 2$, rovnice $x^2 + y^2 - 6x = 0$ je rovnicí kružnice se středem v bodě $[3, 0]$ a poloměrem $r = 3$, neboť tuto rovnici lze převést na tvar $(x-3)^2 + y^2 = 9$. Množina všech bodů $[x, y] \in \mathbb{R}^2$ splňující výše uvedené nerovnosti, tj. definiční obor funkce f , je znázorněna na obrázku 1.1. Je to uzavřená množina v \mathbb{R}^2 .

Titulní strana

Obsah

Výsledky cvičení

Rejstřík

◀◀ ▶▶

◀ ▶

Strana 21 z 491

Zpět

Vpřed

Zavřít

Konec

Figure 2: Screen output

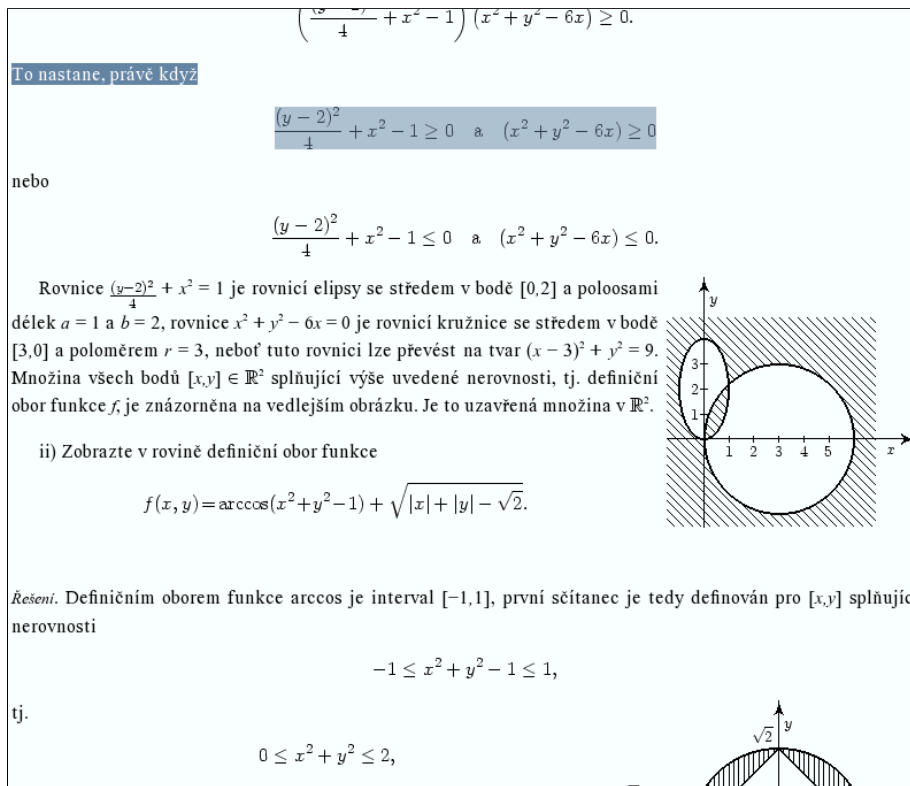


Figure 3: HTML output

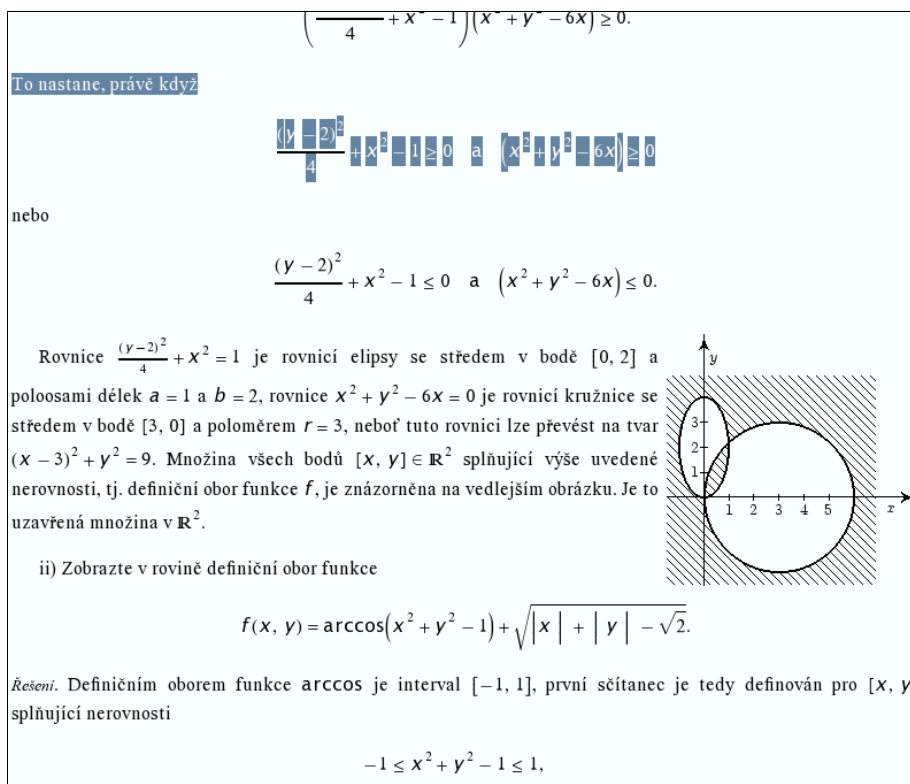


Figure 4: MathML output

5.2 XHTML + MathML

\TeX is very often used for typesetting of scientific texts that make heavy use of mathematical notation. Much more interesting alternative to the HTML with math as pictures is XML language MathML. The most complicated conversion process we tried was the conversion to XHTML + MathML output, especially for our case of mathematical texts with heavy use of mathematical notation.

There are yet some complication for both author and user when using MathML on the web. Firstly, there are different MathML implementations in web browsers, leading to different results. Best results we got with TeX4ht `mozilla` compile option in combination with Mozilla Firefox web browser (or another Gecko based browser). Secondly, user has to have appropriate mathematical fonts installed. Information about necessary fonts, download links and installation instructions for Mozilla Firefox users are available on the Mozilla MathML Project web pages (<http://www.mozilla.org/projects/mathml/fonts/>).

Big TeX4ht advantage is MathML output possibility which is very useful in case of mathematical texts — XHTML + MathML generation is similar to the HTML one:

```
htlatex filename.tex 'xhtml,mozilla'
```

When all goes well we get XML file contains XHTML code that uses MathML vocabulary for expressing mathematical formulae. You can see the result, fully scalable in Mozilla Firefox window, in Figure 4 on page 129.

TeX4ht is very sensitive to the clean mathematical notation for MathML generation. For example expression

```
$M=\{x|x$ is odd $\}$
```

is correct \TeX code. But TeX4ht does not have information about curly brackets pair. One needs to use the right expression:

```
$M=\{x|x \mbox{is odd}\}$
```

In complex situations can be the math reformulation much more complicated or even impossible. In these rare cases TeX4ht offer emergency solution — one can use pictorial object representation:

```
\ifweb
  \Picture*{}
\fi
$M=\{x|x$ is odd $\}$
\ifweb
  \EndPicture
\fi
```

This solution is not limited by the mathematical notation and may be used for any object with problematic rendering in MathML.

6 On-the-fly document generation

Second project where single source approach is being used is the DML-CZ project [10, 1]. The project goal is not only to digitise a quarter million of Czech and Slovak mathematical journal pages, but also provide for parallel print and web-optimised version generation for digital-born data.

The digitised versions are going to be generated from scanned and preprocessed bi-tonal TIFF 600 DPI images. OCR process is being performed by several stage process using FineReader (text OCR) and InftyReader (mathematics OCR) software [11]. InftyReader [12] is able to export the text in \LaTeX or MathML, together with positioning information. This allow to typeset the OCRed \LaTeX or MathML (or both) data as text under image: PDF \LaTeX allows this layered typesetting e.g. using standard `picture` environment. To make the text searchable `cmap` package has to be used for texts in different languages.

Important is the size of the generated files — for print, resolution of printing device is usually demanded, for on screen viewing lower resolution is sufficient. PDF allows different kinds of compression filters — since PDF 1.4 JBIG2 bi-level image compression is supported. PDF \TeX supports JBIG2 encoded figure inclusion in its recent version as well. In addition, Adam Langley has written open source `jbig2enc` converter, as work supported by Google [7].

JBIG2 allows both lossless and lossy compression. For scanned content, the best approach is to get best compression by accepting some loss of image data by using symbol encoding where variation comes from printing errors [7]. JBIG2 compression ratio depends on context size — it gives better results when compressing page ranges instead of single page only. Using slightly lossy compression and high context sizes, we are able to generate PDF images that have about 10–20 % of size of CCITT encoded and LZW compressed images. The parameters of `jbig2enc` allow for fine-tuned picture regeneration, even in the on-the-fly generation scenario.

Conclusion

Using examples of several projects, we argue that \TeX -based authoring and single-source publishing is very natural and effective way of preparation of personalised documents for multiple output devices. TeX4ht is very customisable tool for web publishing from \TeX sources, JBIG2 format for bi-tonal pictures saves space when generating lots of pictures (as in digitisation projects as DML-CZ or Google Scholar).

References

- [1] Miroslav Bartošek, Martin Lhoták, Jiří Rákosník, Petr Sojka, and Martin Šárky. DML-CZ: The Objectives and the First Steps, September 2006. accepted for publication as book chapter CMDE 2006 (A.K. Peters).
- [2] Zuzana Došlá and Ondřej Došlý. *Metric Spaces: Theory and Examples (in Czech)*. Masaryk University in Brno, second edition, 2000.
- [3] Zuzana Došlá, Roman Plch, and Petr Sojka. Matematická analýza s programem Maple: 1. Diferenciální počet funkcí více proměnných (Mathematical Analysis with Program Maple: 1. Differential Calculus). CD-ROM, <http://www.math.muni.cz/~plch/mapm/>, December 1999.
- [4] Michel Goossens, Sebastian Rahtz, Ross Moore, and Bob Sutor. *The L^AT_EX Web Companion*. Addison-Wesley, Reading, MA, 1999.
- [5] Philip Babcock Gove and Merriam Webster. *Webster's Third New International Dictionary of the English language Unabridged*. Merriam-Webster Inc., Springfield, Massachusetts, U.S.A, January 2002.
- [6] Eitan M. Gurari. TeX4ht: L^AT_EX and T_EX for Hypertext. <http://www.cse.ohio-state.edu/~gurari/TeX4ht/>, February 2005.
- [7] Adam Langley and Dan S. Bloomberg. Google Books: Making the public domain universally accessible. In *Proceedings of SPIE — Volume 6500, Document Recognition and Retrieval XIV*, pages 1–10, San Jose, CA, January 2007. The International Society of Optical Engineering. <http://www.imperialviolet.org/binary/google-books-pdf.pdf>.
- [8] Peter Meyer. Planning a single source publishing application for business documents, 2005. http://www.elkera.com/cms/articles/seminars_and_presentations/planning_a_single_source_publishing_application_for_business_documents/.
- [9] Peter Meyer. Introduction to single source publishing, 2006. http://www.elkera.com/cms/articles/technical_papers/introduction_to_single_source_publishing/.
- [10] Petr Sojka. From Scanned Image to Knowledge Sharing. In Klaus Tochtermann and Hermann Maurer, editors, *Proceedings of I-KNOW '05: Fifth International Conference on Knowledge Management*, pages 664–672, Graz, Austria, June 2005. Know-Center in coop. with Graz Uni, Joanneum Research and Springer Pub. Co.
- [11] Petr Sojka, Radovan Panák, and Tomáš Mudrák. Optical Character Recognition of Mathematical Texts in the DML-CZ Project, September 2006. accepted for publication as book chapter CMDE 2006 (A.K. Peters).
- [12] Masakazu Suzuki, Fumikazu Tamari, Ryoji Fukuda, Seiichi Uchida, and Toshihiro Kanahori. INF_{TY} — An integrated OCR system for mathematical documents. In C. Vanoirbeek, C. Roisin, and E. Munson, editors, *Proceedings of ACM Symposium on Document Engineering 2003*, pages 95–104, Grenoble, France, 2003. ACM.
- [13] Edward R. Tufte. *Visual Explanations*. Graphics Press LLC, 1997.