TUG 2016 abstracts

Editor's note: Slides and other related information for many of the talks are posted at `http://tug.org/tug2016/program.html`.

– – ∗ – –

Kaveh Bazargan

*A graphical user interface for Ti*k*Z*

Ti*k*Z is a powerful vector graphics program. The capabilities are far higher than those of any interactive graphics program, e.g. Adobe Illustrator. The fact that Ti*k*Z can be coded logically, keeping the structure of a graphic intact has many advantages, including the generation of "semantic" SVG output, allowing for better accessibility of graphics, e.g. for blind and visually impaired readers. The main barrier against popular usage of Ti*k*Z outside the TEX community is that it is purely code driven, and has a steep learning curve.

At River Valley we have a long term program to produce rich, semantic illustrations which are accessible to readers who are blind or visually impaired. We have chosen Ti*k*Z as the engine to generate these.

To make easier to create the diagrams in the first place, we have been working on a graphical user interface (GUI) to generate Ti*k*Z code and to show its output in near real-time. The first implementation addresses the relabelling of existing illustrations (vector or bitmap) using Ti*k*Z. The system has the following advantages: relabelling can be done as fast or faster than conventional methods; labels do not change size as figures are resized (similarly to Pinlabel); labels are saved as text and can be spell-checked along with the main text of a TEX file.

I will give a live demo of the software.

Charles Bigelow

Looking for legibility

Scientific studies of legibility and their contributions to typography and type design from the 19th century to the present. With amusing anecdotes, ingenious contraptions, clueless assumptions, cooked results, Herculean efforts, and occasional progress toward understanding one of our most puzzling cultural activities.

The following bibliography covers most of the content of the talk on legibility studies of the 20th century, especially the first half of the century, but sadly omits the images and the jokes in the talk.

• *From 1900 to 1950*

Émile Javal (1905). *Physiologie de la lecture et de l'écriture.* Félix Alcan, Éditeur, Paris.

Edmund Burke Huey (1908). *The Psychology and Pedagogy of Reading.* Macmillan, New York.

Barbara Roethlein (1912). "The Relative Legibility of Different Faces of Printing Types", *The American Journal of Psychology*, Vol. 23, No. 1, Jan. 1912, pp. 1–36.

British Association for the Advancement of Science (1913). *Report on the Influence of School-Books upon Eyesight.* Offices of the Association, London.

Lucien Alphonse Legros and John Cameron Grant (1916). *Typographical Printing Surfaces.* Longmans, Green, and Co., London.

Lucien Alphonse Legros (1922). "A Note on the Legibility of Printed Matter: prepared for the information of the Committee on Type Faces." H. M. Stationery Office, London.

L. Richard Pyke (1926). "Report on the legibility of print." Medical Research Council, Special report series, no. 110. His Majesty's Stationery Office, London.

Gerrit Willem Ovink (1938). *Legibility, Atmosphere Value and Forms of Printing Types.* A.W. Sijthoff's Uitgeversmaatschappij N.V., Leiden.

Donald G. Paterson and Miles A. Tinker (1940). *How to Make Type Readable.* Harper & Brothers.

Mergenthaler Linotype Company (1941). The Readability of Type. Brooklyn, N.Y.

Matthew Luckiesh and Frank Kendall Moss (1944). *Reading as a visual task.* D. Van Nostrand.

• *After 1950*

Cyril Burt (1959). *A Psychological Study of Typography.* Foreword by Stanley Morison. Cambridge University Press, Cambridge.

Miles Tinker (1963). *Legibility of Print.* Iowa State University Press, Ames.

E. C. Poulton (1965). "Letter differentiation and rate of comprehension in reading," *Journal of Applied Psychology*, Vol. 49, No. 5.

Bror Zachrisson (1965). *Legibility of Printed Text.* Almqvist & Wiksell.

François Richaudeau (1984). *Recherches actuelles sur la lisibilité.* Retz.

Herbert Spencer (1969). *The visible word: problems of legibility.* Lund Humphries/Royal College of Art, London.

Erich Schulz-Anker (1970). "Syntax-Antiqua, a Sans Serif on a New Basis; Le Syntax Romain, un Linéale sur une base nouvelle; Syntax-Antiqua, eine serifenlose Linearschrift auf neuer Basis." Gebrauchsgraphik 7/1970. Reprinted by D. Stempel AG, n.d.

Rolf F. Rehe (1974). *Typography: how to make it most legible.* Design Research International, Carmel, Idaho.

Keith Rayner and Alexander Pollatsek (1989). *Psychology of Reading.* Prentice Hall.

Dirk Wendt (1994). "What do we mean by legibility", in *Font Technology*, by Peter Karow, Springer.

Paul Kolers, Merald Wrolstad, and Herman Bouma, editors (1979). *Processing of Visible Language, Volume 1.* Plenum Press.
Paul Kolers, Merald Wrolstad, and Herman Bouma, editors (1980). *Processing of Visible Language, Volume 2.* Plenum Press.
Ole Lund (1999). "Knowledge construction in typography: the case of legibility research and the legibility of sans serif typeface". Ph.D. thesis, Department of Typography and Graphic Communication, University of Reading.
Gordon E. Legge (2007). *Psychophysics of Reading in Normal and Low Vision.* Lawrence Erlbaum Associates, Mahwah N.J.
Sofie Beier (2009). "Typeface Legibility: Towards defining familiarity". Ph.D. thesis, Royal College of Art.

Robert Bringhurst
The evolution of the Palatino tribe
Palatino is not just a single typeface but a large and varied group of faces: a taxonomic tribe produced over more than half a century. The members include Aldus, Enge Aldus, Aldus Nova, Heraklit, Michelangelo, Phidias, Sistina, and Zapf Renaissance, as well as foundry Palatino, Linotype Palatino, American Export Palatino, Linofilm Palatino, PostScript Palatino, Palatino Nova, and Palatino Sans.

This constellation of type designs was Hermann Zapf's most ambitious and enduring design project. It began with the "Medici" sketches of 1948 — which led to the first trial cutting of foundry Palatino roman by August Rosenberger at the Stempel Foundry, Frankfurt, in 1949 — and it continued through 2006, when the last authentic members of the group were drawn on screen under Zapf's direction by Akira Kobayashi in Bad Homburg. In between these dates, the underlying designs adapted again and again to changing conditions, represented by the Linotype machine, Linofilm and other phototype machines, and a variety of pre-PostScript digital systems.

Zapf was not the only type designer whose career spanned the tumultuous transitions from metal type to phototype to digital type, but Palatino and its relatives appear to be unique in the complexity of their evolution and the multiplicity of their successive adaptations, under the hand of the original designer, to repeatedly changing methods of typesetting and printing.

Robert Bringhurst has argued for many years that the most promising system of typeface classification is based on botanical and zoological taxonomies. His new book, *Palatino: The Natural History of a Typeface*, published in a limited edition by the Book Club of California, with a trade edition from David R. Godine coming this fall, is an extended test of this thesis. Over many years of research, he has also accumulated hundreds of illustrations documenting the artistry and care, and the industrial advances and collapses, involved in creating these components of our typographic heritage.

Jennifer Claudio
The case for justified text
Documents must be aesthetically pleasing without appearing deliberately designed. One of the basic functions built into most word processing software is text justification. The algorithms behind this function, however, vary based on the software and the preset rules within the software. Some criticisms include compromised readability. Defenders of justified text argue that as long as the typeface is appropriately sized and kerned, justification does not detract from readability. This presentation succinctly demonstrates the behavioral differences visible in WordPerfect, Word, InDesign, and LaTeX, and examines the ability for people who read common printed media to notice the differences.

Jennifer Claudio
A brief reflection on TeX and end-user needs
TeX attracts users who seek a robust method of creating precise typographic products. However, beginning with the instinct to disambiguate the pronunciation of TeX and LaTeX, new users often feel daunted by the so-called learning curve of TeX and its relatives. Given time to learn the syntax and experience in troubleshooting errors, many fare well; however, a population exists that would benefit from the use of TeX who have insufficient time or comfort in the field.

This presentation describes the following: 1) personal use of products that have been shared by other members of this TeX user group community; 2) the quest to recruit new users, abusers, and developers into the TeX community; and 3) requests for specific end-user products.

Tim Inkster
The beginning of my career
Tim Inkster was one of any number of English undergraduates at the University of Toronto in the late 1960s who were entranced by the lure of Stan Bevington's shop in the alley behind 401 Huron Street.

Inkster applied for an entry-level position at Coach House Press, the first time, in 1969, and then a second time a couple of years later.

Unable to secure gainful employment, Inkster felt he had little choice but to start his own small press, the Porcupine's Quill (1974), which has led to bronze and silver medals at Leipzig, a Citation from

the Art Directors' Club of New York, and the Order of Canada (2009) for both Tim and his wife, Elke, "For their distinctive contributions to publishing in Canada and for their promotion of new authors, as co-founders of the Porcupine's Quill, a small press known for the award-winning beauty and quality of its books."

Stefan Kottwitz
TEX in industry I — programming Cisco switches using TEX
I report on the pure text-based, macro-based, TEX-programmed switch configurations which I use at my work at Lufthansa in cruise ship projects. A brief description is available at `http://tex.tips/programming-network-switches`. A demonstration of a non-standard use of TEX in industry.

Stefan Kottwitz
TEX in industry II — designing converged network solutions
Pure graphics, with efficient use of TEX and Ti*k*Z for programming drawings of network architectures (LAN, wi-fi, VoIP, ...). "Efficient" in the sense of my note on good practices at `tex.stackexchange.com/a/297029/213`. Sample notes are at `tex.tips/tag/industry`, and example drawings at `tex.tips/LAN-1-2.pdf`.

Presenting a showcase of what can be done with TEX (instead of Visio, PowerPoint or CAD) and how, with a view toward efficiency (time pressure on projects).

Kevin Larson
Reading between the lines: Improving comprehension for students
While reading is arguably a student's most important skill, the technology of reading is relatively unchanged. Can the power of computing improve a student's reading comprehension? We will discuss what has been learned about typography in the last 500 years, about reading psychology in the last 100 years, and what technology can be invented right now.

Bio: Kevin Larson works for Microsoft's Advanced Reading Technologies team. He collaborates with type designers, reading psychologists, and engineers on improving the onscreen reading experience. Kevin received his PhD for studies of reading acquisition.

Frank Mittelbach
Alice in Wonderland — The tale of the long tail in globally optimized page breaking (part 2)
The story of Alice's Adventures In Wonderland by Lewis Carroll contains a long tale about the tail of the mouse, which starts with

```
      'Fury said to a
     mouse, That he
    met in the
   house,
  "Let us
   both go to
    law:  I will
     prosecute
      YOU.---Come,
        I'll take no
        denial; We
       must have a
      trial:  For
   really this
  morning I've
 nothing
 to do."
  Said the
   mouse to the
    cur, "Such
     a trial,
      dear Sir,
     ...
    many more
   lines ...
  ...
```

Obviously a tail like this should be typeset in full beauty and should not be laid out in knots or cut between pages. Unfortunately, that is more easily said than done, given that all typesetting systems use a greedy page algorithm that cuts page by page. Thus chances are high that disaster strikes and we have to manually adjust earlier page breaks to prevent it.

Using global optimization in pagination has been envisioned already more than 30 years ago by Michael Plass in his PhD thesis and throughout the years other people worked on specific aspects of global optimized pagination, but until today all typesetting engines have taken the "easy" way out and leave the problem essentially to the user.

This is in fact not that surprising, as the problem can get easily out of hand: A naive approach will immediately result in exponential complexity (without introducing float objects into the mix which makes it worse). But even with careful restrictions and specialized algorithms we will soon be reaching the limits of modern hardware with any real live document.

However, computers are getting faster and thus get us closer to make globally optimized pagination a reality. So this year I started to implement a framework that assists in this task, in the hope that for my next book I do not have to hand-adjust half of the page breaks manually — first results are promising!

At Bachotek 2016 I gave some theoretical background to the problem and discussed the basic ap-

proach taken by the framework, including two already implemented optimization strategies: The automatic change of page length on double spreads to add flexibility and the use of automatic variation in paragraph breaking (think `\looseness`) to gain further flexibility.

This time around I like to demonstrate new results where Alice goes "floating" and all the beautiful pictures of the orginal magically appear in their appropriate place.

So slowly the work evolves toward a usable solution. Sit back, relax, and enjoy.

Norbert Preining
Security improvements in the TEX Live Manager and installer

Since the switch to the current distribution method and the introduction of network installs and updates, some years ago, many things have changed in the TEX (Live) world. But one thing has not kept up with the new distribution methods: security.

Until now, there has been almost no verification of a package as downloaded from the CTAN mirrors compared to the original package created in TL. Although we have been shipping MD5 checksums and sizes in the accompanying information, these were used only in rare instances (namely, when restarting a failed installation).

We report about the recent improvements and consistent confirmation of checksums and sizes of the downloaded packages, as well as improvements regarding strong cryptographic signatures of the package information.

Arthur Reutenauer, Mojca Miklavec
The TEX Live M sub-project

TEX Live is the most versatile of TEX distributions, available on a variety of platforms, and very actively developed. It is the basis for MacTEX on Mac OS X, and is bundled by many package managers in Unix distributions. It has, however, a major drawback: its titanic size. This talk will discuss a sub-project to address that, with time for general discussion on wishes and ideas for TEX Live's future.

At its inception in 1996, it was contained in a CD and started growing immediately. Packages are rarely removed, due to compatibility considerations, and only technical considerations are taken into account when considering new packages: if a package fits the requirements, it is added. Today, the `texlive-full` installation scheme includes over 140,000 files and has an installed size of over 4.5 GB.

This situation is a problem for many downstream package developers and also affects the TEX community as a whole. We have started a conversation to see how we could help users find packages. We would like to offer an option to have a more controlled set of packages, probably by creating a new TEX Live "scheme" in the existing distribution by selecting among the 3200+ (to date) packages. We could define strict dependencies between packages, and also strive to do some measure of quality control, in order to create a distribution that's truly useful for newcomers and long-time users alike. The selection has to be community-driven, but there has to be a selection.

In another area, we also want to improve how the binaries are built: at the moment, they're compiled once per year by a number of volunteers who work on one or more of the twenty or so different platforms, and never get updated during the year. While this strikes a good balance between stability, the demand for reasonably recent binaries, and the workload of volunteer builders and packagers, we thought we could do better.

We have recently set up a build infrastructure that can automatically build TEX binaries after every source change for a number of platforms, send emails when builds break, show reports, and make the binaries available to users. This approach takes a lot of burden off the shoulders of people previously responsible for building TEX binaries, while at the same time giving us freedom to run the builds a lot more frequently, getting binaries to users much faster and providing earlier feedback about problems to developers. This part is almost ready and we will give some technical details of how it works.