IVT BIBTEX / LATEX Workshop

Kirill Müller

Michael Balmer

Transport Planning and Systems **October 2014**

Institut für Verkehrsplanung und Transportsysteme
Institute for Transport Planning and Systems

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Contents

0 Introduction

This is a small guideline for using BibTeX and LaTeX environment at the IVT. LaTeX is a free typesetting system that allows producing documents from plain text files, conceptually similar to HTML. BibTeX is the integrated bibliography system that allows maintaining a list of literature references and using them in your documents. This document outlines how these two systems are implemented at the IVT; hence, it can also be of interest to you if you know LaTeX but are new at the IVT.

It starts with a very *brief introduction* that describes how to set up the vitally necessary software and to start working on a copy of the IVT LaTeX template. Some *workflows* are explained in the following. The main part of this document is devoted to *collaboration* – working with your colleagues on the same document or on the institute's BibTeX database. Some *other tasks* are described briefly. Appendices present even more *software* and a short introduction to *Subversion*.

We use the following formatting convention throughout this document:

Paths, URLs, and console commands are written in a `non_proportional/font`

File formats are written in non-proportional *`italics`*.

User interface commands are shown like ***this***.

Keyboard commands are shown like **this**.

Software packages are written in *italics*.

Subversion commands are written in SMALL CAPITALS.

The operating system used here is *Microsoft Windows*, but you can find the required software for all other major operating systems. For questions or suggestions please contact **kirill.mueller@ivt.baug.ethz.ch**. There is also a low-volume mailing list that so far has been used only for announcing new features but could also be used for questions and discussion. Visit `https://sympa.ethz.ch/sympa/info/ivt_latex` to sign up for this list.

1 First steps

The goal of this chapter is to enable the reader to use LaTeX in the IVT environment as quickly as possible, focusing on concision rather than on completeness. This section provides a minimal set of software required to compile the IVT LaTeX template, and instructions to derive your first document from this template. Software and instructions for collaboration are found in Chapter 3.

1.1 Software for typesetting LaTeX documents

There are several things to take care of before you can start using the environment. Please follow the instructions below. Note that under `U:\Install\allYouNeedForLaTeX` you find a collection of required and useful software. If you want to install the software from there, just copy the whole directory to a local directory on your computer, e.g., `My Documents\install`. Note that we will use example paths and names that will be consistent during the whole paper, e.g., the path to the install directory is now `[My Documents\install]`. The brackets denote that you could change that to another path.

The packages are numbered, this reflects the order as described in here. Some packages will be used in Section 3.1 and Appendix A. The software for this section and Section 3.1 has been downloaded on September 12th, 2012, you might want to browse the web for newer versions. Direct links to download pages are provided for your convenience.

Before installing, check if the software is already on your system. The IVT default installation already contains some of the software listed below, however please follow the instructions in Section 1.1.2.1 for updating MiKTeX in any case.

1.1.1 Adobe Reader

Adobe Reader (`http://get.adobe.com/reader/`) is the standard reader, and you should always check your documents if they are displayed properly in *Adobe Reader*.

1. Navigate to `[My Documents\install]\1.1.1_adobeAcroReader`
2. Run the installer executable

Like many other PDF viewers, *Adobe Reader* treats the PDF file as static and is not prepared for the file being replaced or modified in the background. Hence you should use another PDF viewer that watches for changes to the PDF file while writing your document, such as the one bundled with TeXStudio (cf. Section 1.1.4).

1.1.2 MiKTEX

MiKTEX is a free LATEX implementation for *Windows* (`http://miktex.org/2.9/setup`). It contains many hundreds of packages. All these together are the basis for compiling (i.e., creating PDF files out of) LATEX documents.

1. Disable the virus scanner. Installation of MikTEX fails with certain virus scanners. If you are unable to disable it, temporarily disable the virus scanner service:
   - Hit the **Start** button, type **Services**. Select the list entry called ***Services***.
   - Find the service(s) of your virus scanner in the list of services, click the ***Stop*** link to temporarily stop each of them.
2. Navigate to `[My Documents\install]\1.1.2_miktex`
3. Install `basic-miktex-2.9.????.exe`
4. Re-enable the virus scanner.

This installation process may take several minutes. When it is finished, additional command line commands are available, like `pdflatex` or `bibtex` that are necessary to produce PDF documents out of LATEX files. Missing LATEX packages will be installed as necessary upon user confirmation. (Another option is to get the full installation of MiKTEX, which takes more than 1 GB of hard drive space. Run `setup-2.9.????.exe` for this option.)

After the installation is completed, you can check the functionality by typing
`pdflatex -v`
into a *cmd* (command window). A copyright information should appear. If not, try restarting *Windows* first.

1.1.2.1 Keeping MiKTEX up to date

MiKTEX comes with its own *Package Manager*. It is advisable to upgrade the installed packages and binaries every once in a while, *especially before using LATEX for the first time on a standard IVT installation*. The update process has to be triggered manually.

1. Log in as local administrator to avoid entering the password multiple times.
2. Disable the virus scanner (cf. Section 1.1.2).
3. Hit the **Start** button, type **Update**. Select the list entry called ***Update (Admin)***.
4. Click through the wizard, select any mirror (the Swiss or German mirrors usually work well).
5. Re-enable the virus scanner.

In the list of packages to update, it might be that only one very few “base” (or “core”) package are selected. In this case, you will have to repeat the last two steps once more.

1.1.3 Active Perl

Perl is a scripting language. To run *Perl* scripts (files with ending `.pl`), you need to install its interpreter for *Windows*, provided by ActiveState (`http://www.activestate.com/activeperl/downloads`). (Linux/Unix and Mac already provide the interpreter by default). You need that to use the compilation script and the bibliography environment.

1. Navigate to `[My Documents\install]\1.1.3_activePerl`
2. Install `ActivePerl-?.??.?.????-MSWin32-x86-??????.msi` for a 32 bits system or `ActivePerl-?.??.?.????-MSWin32-x64-??????.msi` for a 64 bits system.
3. Define `C:\Perl` as the destination folder for the program.

After the installation is completed, you can check the functionality by typing

```
perl -v
```

into a *cmd* (command window). A copyright information should appear. If not, try restarting *Windows* first.

1.1.4 TEXStudio

TEXStudio is a free text editor specifically designed for editing LATEX documents (`http://texstudio.sourceforge.net/`, select "Download" from the menu at the left). It runs on all major operating systems, is easy to use and comes with a good PDF viewer with integrated support for SyncTEX – bidirectional synchronization between source and PDF files.

1. Navigate to `[My Documents\install]\1.1.4_TeXStudio`
2. Install `texstudio??_win32.exe`

Note: Former versions of the template used ISO-8859-1 as default text encoding. This is now unnecessary and not recommended anymore – UTF-8 should be the encoding of choice and is also selected by default in the most recent version of the IVT template.

1.2 Starting your first IVT LATEX paper

Everything you need to write a paper using the IVT environment is included in a publicly available template. It is available at `http://www.ivt.ethz.ch/education/index`; look for "Vorlage für LATEX". Download the template and extract it to any location, then execute (double-click) the `latex2pdf.bat` script. As a result, a directory `tmp` and the file `Template.pdf` should be created in the directory where you have extracted the template. Consult the file `tmp/Template.log` in case of errors.

LATEX is a descriptive way to write a paper: The resulting document is generated from plain text files, and commands define a section, a list of items, ... The source files usually have the extension *tex*. To view the sources for the template, open the file `Template.tex` in

TeXStudio and ***set it as master document*** by right-clicking the file in the pane at the left and choosing the corresponding menu item. There is also a brief `README.txt` file. To view this file on *Windows*, please use a text editor that understands Unix-style line endings.

The template can be used as starting point for your first report, paper, thesis etc. at the IVT. (Refer to the `README.txt` for instructions on renaming the main file.) However, it is **highly recommended** to switch to SVN storage once you are accustomed to the system. A paper started from the template can be moved seamlessly to the institute's Subversion repository, see Section 3.4.4 for details.

In the following, the structure of the template's main file `Template.tex` is presented.

1.2.1 Layout

At the top of the file `Template.tex` you are able to define in which language you want to write your paper. Also, the paper layout can be defined here. Several document layouts are predefined in the IVT environment. It is very easy to change a whole paper from one layout into another: Only one line of code has to be changed!

The following document types are provided already:

- Papers:
  - `ivt-wp`: IVT working paper layout for German & English language
  - `ivt-generic`: Generic IVT paper layout for different purposes in German & English language
  - `trb`: TRB conference paper for English language
  - many more, check the files in `_latexfiles/_layouts` that are not prefixed by an underscore
- Dissertations:
  - `ivt`: IVT working paper layout for German & English language
  - `eth`: ETH dissertation layout for German & English language (ETH Title page and a fancy document layout made by balmermi)
- CV:
  - a fancy CV layout made by balmermi (for German & English language)
- some miscellaneous reference list documents as already mentioned in the previous sections. (German & English)

Please contact **kirill.mueller@ivt.baug.ethz.ch** if there is no layout for your conference/journal/... yet.

1.2.2 Properties

In the first part of the document, you will find many different *properties* to define. Each of them is well-documented. The structure of a property definition is:

```
\newcommand{\propertyname}{value}
```

Please do NOT remove any of these definitions. If you do not want to specify a specific property, just leave the value empty. The following properties are available for any kind of paper layout:

- `\myfirstlang`
- `\mytitlefigure`
- `\mytitle`
- `\myinstitution{EN|DE}`
- `\mynumber`
- `\myyear`
- `\mymonth`
- `\myday`
- `\mywordcount`
- `\mykeywords{EN|DE}`
- `\my{first|second|third|fourth|fifth|sixth|...|twelfth}author`
- `\my{first|second|third|fourth|fifth|sixth|...|twelfth}authorREF`
- `\my{first|second|third|fourth|fifth|sixth}address`

The last property definition is a little bit more complex. Just insert the information according to the example:

```
\newcommand{\myfirstaddress}{
  \createcontact{\myfirstauthor}%
  {EDIT_ONLY_HERE}% address line 1
  {EDIT_ONLY_HERE}% address line 2
  {EDIT_ONLY_HERE}% address line 3
  {EDIT_ONLY_HERE}% phone
  {EDIT_ONLY_HERE}% fax
  {EDIT_ONLY_HERE}% email
}
```

It is recommended to create just one address entry for the authors from the same institution. The example in the template shows how to do this.

In addition, the following properties are required for dissertations:

- `\myfirstauthortitle`
- `\myfirstauthordoctortitle`
- `\myfirstauthorbirthday`
- `\myfirstauthorcity`

The second, third... authors specify the examiners.

1.2.3 Contents

The actual document starts with the command `\begin{document}`. It starts with the order of your document heading pages (title page, table of contents, abstracts, etc...). After that, you can add your sections, followed by the bibliography and an appendix. The order in which the different parts appear defines the order in which they will show up in the paper. If you do not want to use a specific part (e.g., a table of contents) just comment out the corresponding command.

If you want to split up parts into several files (for better organization, e.g., one file for each section), you can always use the `\input{FILENAME}` command. When compiling, LaTeX will replace this command with the contents of the given file.

2 Workflows

This chapter illustrates some common workflows such as finding a literature reference in the supplied BibTeX file, enabling *Word*-like continuous preview, and troubleshooting.

2.1 Literature references

The templates include the file `_latexfiles/bibs/all-eng.bib` which is a more or less current snapshot of the IVT BibTeX database. Usage examples are also provided. You can use all references in the supplied file right away, and also add own references to the file `my.bib`. Note the following:

- You do not need to care about the way a reference will be created. This will be automatically done via the LaTeX style definitions that are selected automatically when you select a paper layout.
- Entries from the IVT database will be automatically adapted to the German or English version. (E.g., in English references we have to write "Zurich" while in German references the town is witten as "Zürich").

Using a reference is as simple as writing `\citep{KEY}` or `\citet{KEY}` in your paper. To find out which `KEY` to use, open the file `_latexfiles/bibs/all-eng.bib` with your preferred text editor. Since the bibliography files are plain text files, you are able to find the references with just a normal search in your text editor. Therefore, you can search for authors, titles, etc...

Each bibliography entry looks like this:

```
@TYPE{KEY,
  ATTRIBUTE =    {VALUE},
  ATTRIBUTE =    {VALUE},
  ...
  ATTRIBUTE =    {VALUE},
}
```

The `KEY` is always defined at the first line of an entry. As an example, let us use the entry for Axhausen's Book "Moving Through Nets". So we search for "Axhausen", "moving" etc. and will sooner or later find the following entry:

```
@BOOK{Axhausen_2006,
  EDITOR =    Axhausen,
  TITLE =     {Moving Through Nets: The ...},
  PUBLISHER = Elsevier,
  YEAR =      {2006},
  address =   oxford,
}
```

The key for that entry is therefore `Axhausen_2006`. To cite this paper in a textual context, like

> As Axhausen (2006) has noted, ...

we write `\citet{Axhausen_2006}` (or `\Citet{Axhausen_2006}` at the beginning of a sentence). To obtain a reference in parentheses, like

> To execute her schedule, a person must interact with others in the networks and in activity opportunity places (e.g., shops, cinemas, other persons' homes, ...) (Axhausen, 2006).

we write `\citep{Axhausen_2006}`. The literature reference will appear automatically in the "References" section of the paper.

2.2 Continuous compilation and preview

If you are used to immediately see what you type, as in *Word*, you may find it strange to first "program" your text and then execute some command which produces the output. It is possible to configure your system so that the compilation is executed every time you save a file to disk. Together with a good PDF previewer that does not lock the file and understands which part of the document belongs to which part of the source code (SyncTEX), it is possible to achieve almost the same feeling.

1. Execute (double-click) the file `latex2pdfLoop.bat`. A console window should appear, indicating after some time that it is waiting for changed files.
2. Open *TeXStudio* with your master document (e.g., `Template.tex`). Hit **F7** to start the PDF viewer.
3. Now edit the file (e.g., add some text) and save it. After not more than one second, the console window should indicate that compilation is in progress.
4. The PDF viewer should update itself, displaying the text you have added.
5. In the text editor of *TeXStudio*, hit **F7** again with the caret positioned on the text you have just entered. The text is highlighted briefly in the PDF previewer.

6. Hold the **Ctrl** key and double-click any part of the PDF in the previewer. The text editor navigates to the approximate position of the text that you have clicked.
7. Close the console window to stop the compilation loop.

Make sure that you do not start two instances of this script simultaneously.

Preview is almost instantaneous, but it still takes several seconds until the document is ready. Some hints to hasten compilation:

- Make your virus scanner ignore the files that are used during compilation, at least your document and everything below the installation path of MiKTEX. (See the effect by temporarily turning off the virus scanner.)
- Split your document and compile only the parts you are currently working on.
- You can save about one more second by splitting your master file and pre-compiling those parts that rarely change. Ask **kirill.mueller@ivt.baug.ethz.ch** for details.

2.3 In case of trouble

If your code does not compile anymore, try the following:

- Kill the continuous compilation script (cf. Section 2.2).
- Execute `cleanLatex.bat` before attempting another compilation run.
- Revert to the last state where the document compiled correctly. (This requires version control, or at least a backup copy.)
- Consult the log file (extension: `log`).
- Ask a more experienced user for help.

3 Collaborating

This chapter shows how to work jointly on a paper or on the IVT BibTeX database. The BibTeX database is a precious resource that contains a large amount of classic and recent literature references; one central repository for literature references is a big time saver when the references are reused. Comparing to working on a copy of the publicly available template, this provides the following benefits:

- Your papers are under version control.
  - Backup. Revision history. Overview of most recent changes. Revert to clean state.
  - It is possible for two or more collaborators to work truly simultaneously on the same document.
  - You can easily refer others to your sources. Other authors will be able to look at the way you write your papers.
- You are always up to date.
  - References added by others can be used in your papers immediately, without having to manually copy files. You only need to be actually connected to the network when fetching updates.
  - Sometimes, also the templates and bibliography styles are updated to add new features or to include a template for a new journal or conference.
- You can give back: Other members of the institute can use the literature references created by you.
  - Each entry has to be added only once. All users get access to all entries without rewriting them over and over again.
  - Creating a list of papers published by a member of the institute (or even by the entire institute) is just a matter of collecting BibTeX keys once the references are added to the database. Useful for the IVT website, for your CV, for the yearly report (Jahresbericht), ...

The system gets more and more attractive when more and more people are using it.

3.1 Software for collaboration

Below you will find a compilation of software required to connect to the institute's SVN server and to execute the scripts in the BibTeX database.

3.1.1 TortoiseSVN

This is the SVN client (`http://tortoisesvn.net/downloads.html`). It is required to access the IVT Subversion server `repos.ivt.ethz.ch` (cf. Section 3.2). Appendix B is a very short introduction to the most important commands of this system.

1. Navigate to `[My Documents\install]\3.1.1_tortoiseSVN`
2. Install *one* of the following files:
   - `TortoiseSVN-?.?.?.?????-win32-svn-?.?.?.msi`, if you run 32-bit *Windows*
   - `TortoiseSVN-?.?.?.?????-x64-svn-?.?.?.msi`, if you run 64-bit *Windows*

   Hint: You run a 64-bit *Windows* if there is a folder named `Program Files (x86)` in the root of your system drive (usually, `C:\`).
3. Restart *Windows*

Now, you are able to CHECKOUT any part of the SVN repository to your computer. See Section 3.3 for instructions.

3.1.2 Python

Python is another scripting language. To run *Python* scripts (files with ending `.py`), you need to install its interpreter for *Windows* (`http://www.python.org/getit/`, choose version 2.7.x.). Linux/Unix and Mac already provide the interpreter by default. Scripts that add new entries to the bibliography are written in *Python*. (Note that these scripts are incompatible to *Python 3* at the time of writing.)

1. Navigate to `[My Documents\install]\3.1.2_python`
2. Install `python-2.7.?.msi`

After the installation is completed, you can check the functionality by typing

```
python -V
```

into a *cmd* (command window). A copyright information should appear. If not, try restarting *Windows*.

3.2 Getting access to the IVT Subversion server

To verify access to the environment, please open the following URL in your browser: `https://repos.ivt.ethz.ch/svn/ivt/doc/`. If you are outside the ETH network, make sure that your VPN connection is up and running. Please use your n.ethz credentials when asked for a user name and a password. A page titled **doc - Revision XXXX: /** should appear. If not, ask your system administrator for access.

As of September 2012, the BIBTEX / LATEX environment is located at `repos.ivt.ethz.ch`,

the institute's *Subversion* server. The SVN server can be reached through the HTTPS protocol, and no setup of SSH is needed anymore.

3.3 Getting the BibTeX / LaTeX environment

We now want to get the BibTeX / LaTeX environment with SVN. The environment contains all kinds of papers and presentations produced by IVT members. They do not have to be papers written in LaTeX, you will also find *Word* documents and *Power Point* presentations.

You can either get the complete environment (takes several GB of your hard drive) to see the whole structure, or you can only get the parts of the environment you are interested in, which consumes a lot less hard disk space. To get only parts of the environment and still have a good overview over the structure, you can do a sparse checkout (cf. Section 3.3.1). To check out the whole environment at once, there are two ways: by regular checkout (cf. Section 3.3.2), or by copying someone else's environment (cf. Section 3.3.3).

3.3.1 Sparse checkout

This is the recommended way if you are working on a laptop or on another device with limited disk space.

1. In *Explorer*, create a folder in which you want to check out parts of the repository (here: `[My computer\sandbox]`)
2. Right-click that folder and choose ***SVN Checkout...*** and enter the following URL: `https://repos.ivt.ethz.ch/svn/ivt/doc/trunk`
3. In the menu that appears, there is a section called *Checkout depth*. Select ***immediate children, including folders***
4. Click ***OK***

Now you checked out only first layer of the folder structure. All folders you see are still empty. You can explore the next layer of the folder structure by the following procedure:

1. Open the folder you want to explore (e.g. /papers)
2. Right-click in the empty folder and choose ***Update to revision...***.
3. In the menu that appears, there is a section called *Checkout depth*. Select ***immediate children, including folders***
4. Click ***OK***

Now you checked out the next layer of the folder structure in the chosen folder. By repeating this procedure you can explore the and check out the folder tree to the extent you need.

For the LaTeX environment to work, you need to fully checkout the folder `_latexfiles`. To do so you need to:

1. Open the folder `_latexfiles`
2. Right-click in the empty folder and choose ***Update to revision...***.

3. In the menu that appears, there is a section called *Chekout depth*. Select ***Fully recursive***
4. Click ***OK***

Fully recursive means that all files and folders including all versions in the selected folder are checked out. When you find a folder of which you want the entire content have checked out (e.g. `/papers/trb/2012` for all TRB Papers of the year 2012), you repeat the above procedure in the folder in question.

Sparse checkout is a bit more laborious, but you avoid to store GBs of data you don't need on your hard disk.

Thanks to Boris Jäggi for contributing this section.

3.3.2 Full checkout

This is the recommended way if you are connected with a cable to the ETH network (or to a network with a high-speed connection to the ETH). A checkout via WiFi (wireless LAN) or from your home network is also possible but will take several hours.

1. In *Explorer*, create a folder in which you want to check out parts of the repository (here: `[My computer\sandbox]`)
2. Right-click that folder and choose ***SVN Checkout...*** and enter the following URL: `https://repos.ivt.ethz.ch/svn/ivt/doc/trunk`
3. Click ***OK***

You are now getting the local copy of the complete folder tree of `ivt/doc` (that will take a while). If the process is interrupted, retry and ignore the warning that the checkout folder is not empty.

3.3.3 Copy someone else's environment

If you can get a full copy of someone else's SVN environment, you can simply copy it to your hard drive, *provided that you use the same or a later version of TortoiseSVN*. Please make sure that you copy all hidden directories. To ensure that the environment you obtained is "clean", i.e., completely in sync with the SVN repository, do the following.

1. Right-click that folder again and choose ***SVN Update***
2. Right-click that folder again and choose ***SVN Revert...***. If the list of files is empty, click ***Cancel***. Otherwise, check ***Select all*** and ***OK***.
3. Right-click that folder again and choose ***SVN Commit...***. Check ***Show unversioned files***. If the list of files is empty, click ***Cancel***. Otherwise, check ***Select all***, right-click any file and choose ***Delete***. Confirm.

Now your enviroment is ready to use, just like after a fresh checkout.

3.4 Writing papers in LATEX inside the IVT BIBTEX / LATEX environment

This section describes the organization of the institute's Subversion repository. It also shows how to import a paper that has been derived from the template.

3.4.1 Common files

All layouts, bibliography styles, logos, scripts etc. are stored in a central location: The `_latexfiles` directory in the root of the *Subversion* repository. This location must be made known to the build environment. There are two options to achieve this: (a) by creating a symbolic link to this location within the paper's directory, or (b) by using a relative path. The symbolic link is the recommended option, but this is not always possible (e.g., on a network share or on a *Windows XP* machine).

Section 3.4.3 describes how to set up either option when copying the template within *Subversion*. This applies to both starting a new paper and importing an existing paper (Sections 3.4.2 and 3.4.4). This sounds more difficult than it is, and has to be done only once per paper.

3.4.2 Starting a new paper

In `ivt/doc/papers` all papers should be stored at an appropriate place. We **highly recommend** the following folder structure:

```
ivt/doc/papers/[CONFERENCE_NAME]/[YEAR]/
[A_MEANINGFUL_NAME_OF_THE_PAPER]/[PAPERNAME].tex
```

Please do not use spaces in the file names.

Under `papers/workingpapers/recurring/template` you find an example paper which you can use as a template for your own paper. (By the way, this is the source of the template introduced in Section 1.2.) Assume we want to write a paper for *STRC 2013* about *social networks*. Then do the following:

1. In `papers`, create a folder `strc` (if not already exists).
2. In `papers/strc`, create a folder `2013` (if not already exists).
3. ADD and COMMIT these newly created folders to the SVN repository.
4. In *Explorer*, navigate to `papers\workingpapers\recurring`.
5. Use the BRANCH command to copy `template`: Right-click the `template` folder and select ***SVN→Branch/tag...***. Enter `https://repos.ivt.ethz.ch/svn/ivt/doc/trunk/papers/strc/2013/SocialNetworks` as target URL. Do not forget to provide a log message. (You can also perform this operation in the REPOSITORY BROWSER.)

6. UPDATE the folder `papers/strc/2013` to load the newly copied files into your working copy.
7. RENAME `Template.tex` to `SocialNetworks.tex` (using the ***SVN→Rename...*** command).
8. Edit the files `latex2pdf.bat`, `Makefile.in` and `SocialNetworks.tex` to replace all occurrences of `Template` by `SocialNetworks`.
9. COMMIT the changes to SVN.

In this state, the paper will not compile. As noted in Section 3.4.1, you have to declare where the common files live. This is detailed in the following section, so read on.

3.4.3 Defining the location of the common files

Decide if you want to create a symbolic link in your paper's directory, or if you want to refer to the location at the root using a relative path (cf. Section 3.4.1).

Symbolic link

1. Execute the script `createLatexfilesLink.bat`,
2. IGNORE the newly created link `_latexfiles`,
3. UPDATE and COMMIT the entire `SocialNetworks` folder.

Relative path

1. In `SocialNetworks.tex`, replace `\newcommand{\mypath}{...}` by `\newcommand{\mypath}{../../../}`,
2. COMMIT the file `SocialNetworks.tex`.

Compile by executing `latex2pdf.bat`. Check that the file `SocialNetworks.pdf` has been created in the `SocialNetworks` directory. Now, you can start editing your own LATEX paper. For this, open `SocialNetworks.tex` with your preferred editor.

3.4.4 Importing a paper that has been derived from the template

If you already have started writing a paper using the template presented in Section 1.2, you can import it into the SVN repository. We assume the same setting as in Section 3.4.2; also, we assume that the paper has not been under version control before. (Hint: Check for hidden directories named `.svn` or `SVN`.)

1. In `papers`, create a folder `strc` (if not already exists).
2. In `papers/strc`, create a folder `2013` (if not already exists).
3. ADD and COMMIT these newly created folders to the SVN repository.
4. In *Explorer*, paste your paper directory into `papers\strc\2013`. Rename the newly pasted directory to `SocialNetworks`.
5. Execute (double-click) the script `cleanLaTeX.bat`.
6. Delete the subdirectory `_latexfiles`.

7. ADD the folder `SocialNetworks`.
8. UPDATE and COMMIT the directory `strc\2013` to SVN.
9. Proceed as in Section 3.4.3 to specify the location of the common files.
10. IGNORE the `tmp` directory and all generated files (by extension), notably `.synctex.gz` and `.pdf`.
11. UPDATE and COMMIT the entire `SocialNetworks` folder.

3.4.5 Check in autogenerated files?

When using version control, it is common practice to check in only those files that are needed to generate the output. This means that the resulting *PDF* should *not* be COMMITted to the repository, and this is the reason why it was IGNOREd in the previous sections. However, it makes sense to keep "stable" versions of a document, e.g., the state in which it was sent to review or in which it was finally accepted.

To achieve this, we suggest creating a directory named `out` to store the output, copying the document you want to keep to this directory and ADDing and COMMITting it. One good thing about version control is that we do not have to store several versions of the file: Once we COMMIT a file it will be available indefinitely (e.g., through the LOG command), even if we later overwrite it. This means that subsequent versions can be stored using the same name.

3.4.6 Dissertations

The use of the dissertation layout is similar to the use of the paper layouts. All dissertations should be stored at an appropriate place under `ivt/doc/dissertations`. We highly recommend the following folder structure:

```
dissertations/[YOUR_ETH_USERNAME]/[DISS_NAME].tex
```

Please do not use spaces in the names.

Under `ivt/doc/dissertations/balmermi` you find an example dissertation (well, actually it's balmermi's dissertation).

3.5 Adding BIBTEX entries

Here the structure of the BIBTEX environment is shown. We explain how to add new bibliography entries to the BIBTEX environment.

3.5.1 Files

The following files are used to define Bibliography Entries:

ivt/doc/_latexfiles/bibs/translations.txt Language specific words, like:

- The months (e.g., March vs. März)
- Addresses (e.g., Zurich vs. Zürich)
- Organizations (e.g., Swiss Federal Roads Authority vs. Bundesamt für Strassen)
- Universities and schools (e.g., ETH Zurich vs. ETH Zürich)
- Departments, institutes and groups (e.g., Institute for Economic Research (IRE) vs. Istituto Ricerche Economiche (IRE))
- Types for technical reports (e.g., Working paper vs. Arbeitsbericht)
- Types for research reports (e.g., Research Report vs. Forschungsbericht)
- Types for unpublished references (e.g., internal presentation vs. Interner Vortrag)
- Miscellaneous publications like webpages or software
- the word "forthcoming" vs. "im Druck"

ivt/doc/_latexfiles/bibs/author.txt Authors.

ivt/doc/_latexfiles/bibs/journal.txt Journals.

ivt/doc/_latexfiles/bibs/publisher.txt Publishers.

ivt/doc/_latexfiles/bibs/special.txt Internal definitions.

ivt/doc/_latexfiles/bibs/{bib-type}.bib The actual bibliography entries. The following types are available:

- `_latexfiles/bibs/article.bib`
- `_latexfiles/bibs/book.bib`
- `_latexfiles/bibs/incollection.bib`
- `_latexfiles/bibs/inproceedings.bib`
- `_latexfiles/bibs/manual.bib`
- `_latexfiles/bibs/mastersthesis.bib`
- `_latexfiles/bibs/misc.bib`
- `_latexfiles/bibs/phdthesis.bib`
- `_latexfiles/bibs/proceedings.bib`
- `_latexfiles/bibs/researchreport.bib`
- `_latexfiles/bibs/techreport.bib`
- `_latexfiles/bibs/unpublished.bib`

Each of the above mentioned files are plain text files. Edit them with *TeXStudio* or your preferred editor.

3.5.2 Compilation and verification

The files `ivt/doc/_latexfiles/bibs/all-eng.bib` and `.../all-ger.bib` should not be edited. They will be created from the files shown in the previous section by the script `mergeAll.pl` in the `_latexfiles` directory. This script is executed automatically when you run `latex2pdf.bat` (cf. Section 1.2), but not during continuous preview (cf. Section 2.2). To run that script just double-click it.

There is also a script `check.cmd` in the `_latexfiles` directory that checks the consistency of the BIBTEX database. It tries to compile every reference and stops if there are errors. Please execute this script every time after changing something in the BIBTEX database before committing to SVN. As a side effect, the file `_latexfiles/unsortbibs/example.pdf` is created; it contains all references properly written out, just as they would appear in a paper.

3.5.3 File partitioning

Since the above mentioned files are plain text files documents, it makes sense to define a way how the files should be partitioned such that entries are easier to find.

3.5.3.1 translations.txt

The `translations.txt` file is a simple three-column table separated with **tabs**. It has the following form:

```
KEY[tab]ENGLISH_VALUE[tab]GERMAN_VALUE[return]
```

A line starting with the **%** character is a comment line. This line will be ignored by BIBTEX. It is an appropriate way to post comments or suggestions for the other users.

Use an appropriate **KEY**. The rules are:

- only characters small characters and dashes (-). No special characters like umlauts
- add a new triplet such that the keys of each partition are ordered lexicographically.
- **addresses** as simple but unambiguous words
- **organizations** starts with *org-*
- **universities and schools** starts with *uni-* or *school*
- **departments, institutes and groups** starts with the key of the organization, university or school as explained above, followed by a dash and the abbreviation of the group
- **types of technical reports and research reports** starts with *rep-*
- **types of unpublished papers** starts with *unpub-*
- **types of misc** starts with *misc-*

For all the above mentioned groups int the translation file, please sort them **lexicographically** according to the **key**!

3.5.3.2 author.txt, journal.txt, publisher.txt

These files are simple two-column table separated with **tabs**. They have the following form:

```
KEY[tab]VALUE[return]
```

Just like in `translations.txt`, a line starting with the **%** character is a comment line.

The *KEY* of an author is always written as:

```
SURNAME, FIRSTNAME SECONDNAME ETC
```

Please write down the complete name. If you know only the initials of the *FIRSTNAME*, etc. write them with appending dot (.). There are already many examples in `author.txt`, so just keep to that syntax. Please order your entries **lexicographically** according to the **key**!

Adding new authors, journals and publishers has been automated. To add an author, execute the script `add_author.py` in `_latexfiles`. (This script requires Python 2.6, 2.7, 3.3 or later.) A console window with instructions appears. Enter the author's names, one per line. Finish by pressing **Ctrl+Z** followed by **return**. The authors are added to the corresponding file, in accordance to above rules. The key is created automatically. To verify the script's operation, execute DIFF on the `bibs` directory. Do not forget to COMMIT these modifications to the SVN repository. – Adding journals and publishers works in an analogous way, only that here you need to specify a key yourself. Please contact **kirill.mueller@ivt.baug.ethz.ch** for comments and suggestions.

3.5.3.3 The Entry files (ENTRYTYPE.bib)

The most tricky part for adding a new bibliography entry is to find out what kind of entry it is. Here is a short overview of the entry types:

article.bib An article from a journal or magazine. Therefore, there MUST be a journal or a magazine in `journal.txt`, e.g., Transportation; Transportation Research Record; Zeitschrift für Verkehrswissenschaft; etc...

mastersthesis.bib A master thesis (or "Diplomarbeit").

phdthesis.bib A PhD thesis (or "Doktorarbeit", Dissertation).

techreport.bib A report published by a school or other institution, organizations or groups, usually numbered within a series. Also semester projects but no master / diploma thesis and no dissertations. Also, do not add research report here!

researchreport.bib A report published by a school or other institution, organizations or groups. Compared to techreport.bib, research reports **MUST** have a **client**.

manual.bib A technical documentation. Typically written by an institution, sometime there is an author. It is NOT a TechReport!

unpublished.bib Everything which is NOT going to be published. Typical examples are presentations, internals (presentations, seminars, org-meetings, tech-meetings, etc...), personal discussions, etc. It is NOT a paper which is going to be published somewhere, like PhD-Thesis in progress, Tech-Reports, submitted papers of a conference or a journal, etc. Usually, when you write a paper, you do not refer to something that is unpublished. But sometimes, you want to add stuff that you have done (especially when it comes to the "Jahresbericht").

book.bib A book with an explicit publisher. NO Journals like TRR! E.g., Arentze and Timmermans: "ALBATROSS: A Learning-Based Transportation Oriented Simulation"; Bieger,

Laesser and Maggi: Jahrbuch 2002/2003 Schweizerische Verkehrswirtschaft; etc.

There are **two types of books**. One has an explicit **author** which actually wrote the whole book (like Ferber's Book "Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence"). The other has an **editor** which collects different papers from other authors. A known example is Axhausen's book "Moving Through Nets: The Physical and Social Dimensions of Travel".

incollection.bib A part of a book with its own title. Therefore, the book MUST exist in the *book.bib* as a book with an explicit **editor**! NO Article in a Journal or magazine! E.g., Axhausen in Jones (1990); Balmer et al. in Timmermans (2005); Balmer et al. in Bieger et al. (2005); etc.

For example: Axhausen has written the introduction for the book "Moving Through Nets". If you want to refer to his introduction it is an *incollection* entry. The book itself is part of *book.bib*.

proceedings.bib The proceedings of a conference. NO Journals like TRR! The papers can be published as CD-ROM, on the web or similar. If the paper of a conference is published in a book DO NOT put it here. It MUST be a BOOK entry with the paper as an INCOLLECTION entry. Proceedings examples are: STRC (web), IATBR (CD-ROM), TRB annual meeting CD-ROM

inproceedings.bib A paper published in a PROCEEDINGS. Therefore, the conference proceedings MUST exist in the proceedings.bib! NO Article in a Journal or magazine! NO published book from a conference! E.g., Axhausen and Goodwin in DRIVE (1991); Balmer et al. in AAMAS (2004)

misc.bib Use that only when nothing else fits. Typical examples are webpages and software.

At the beginning of each of the bib-entry files, a precise description is given, how to add a specific entry. More precisely, each file is separated into several parts:

1. Bib-Entry type description
2. Description of the bib-entry **KEY**
3. Description of the available attributes
4. Example entries
5. TODO entries
6. Actual bib-entries

Please strictly follow the description given for each BIB file.

3.5.4 Hints for adding entries

Here are some hints which probably helps to create an entry easily. It is a kind of a "cookbook" which you may follow.

1. Make sure `_latexfiles` is "clean", i.e. there are no uncommitted changes. Check this by trying a COMMIT and looking at the list of files to commit, it should be empty. (If

you follow this cookbook, you should always have a "clean" state here. If not, you will have to do a mix of UPDATE, COMMIT and REVERT commands interspersed with calls of the `check.cmd` script. You have been warned.)

2. Run an UPDATE on `_latexfiles` (see Appendix B) to get the newest version of the BIB files.
3. Run the `check.cmd` script. If errors are reported, they are most likely someone else's – fix them or contact the one who added the entry. (Again, if everybody runs `check.cmd` before COMMITting, this should never happen.)
4. Before you actually start adding a new entry, be sure that it does not already exists. One way is to open the file `_latexfiles/bibs/all-eng.bib` with an appropriate text editor and search in there. If you found it, your work is done! Just use it as described in Section 2.1. If it is not there yet, proceed.
5. Find out what kind of bib-entry your reference is. Sometimes it is simple (e.g., STRC or TRR paper), sometimes it is quite tricky. The reason for that is typically that you already have the reference from another paper or similar. Unfortunately, it is not guaranteed that this reference is "correct" (whatever that means...).
   We suggest to search via *Google*, *Scholar Google*, *e-collection*, *NEBIS*, etc. for keywords. Usually, you will find several different ways of writing of the same reference. With that information—normally—you are able to find out if it is a paper from a book, a paper in a journal, and so on.
   HINT: Take some time to find out what it is. You typically spend twice the time as if you are not a careful searcher. But, at the end, it will pay off!
   Therefore, when you have found out what kind of reference that is, open the corresponding bib-entry file (cf. Section 3.5.3.3).
6. Copy the first given example entry (it contains all attributes available) into the TODO part and start replacing the values of the attributes by the information of your reference.
   Note, that attributes written in *capitals* are *REQUIRED* attributes, while the others are optional. If possible, always try to use all attributes.
   Note, that it is **HIGHLY RECOMMENDED** to use the formats as given by the examples at the top of each bib file. Unfortunately, if you miss one of the brackets, or a comma, or something else, that looks unimportant, there will be strange error messages while compiling.
7. Use the *keywords* for authors, publishers, organizations, schools, types, month, address, etc. instead of the names. If an author, publisher, journal, address, etc. is not yet present in the `author.txt`, `publisher.txt`, `journal.txt` or `translations.txt` file, resp., add them there (cf. Sections 3.5.3.1 and 3.5.3.2).
8. At last, add the **correct key** to that entry and save the file(s).
9. Run the `check.cmd` script. Fix the newly added reference if the script reports errors.

(By running `check.cmd` before changing anything, we know that the error is ours now.)

10. Compile a document that uses this reference. (You are not adding the reference just for fun, right?) Edit the reference, `check.cmd` and recompile the document until it looks alright.
11. Move your bib-entry in the BIB file from the TODO items to the place where it belongs to (**keys** are sorted **lexicographically**!).
12. Save the file(s) and run the `check.cmd` script one last time.
13. Do not forget to provide your new entry to the other users. For that, use the COMMIT command on the folder `_latexfiles` (see Appendix B). Be sure that the only files which will be committed to the repository are `*.txt` and/or `ENTRYTYPE.bib`. REVERT all other files. Add an appropriate log message and click ***ok***. Now, all other users can use this new bib-entry, and you have a "clean" BIBTEX database again.
    If you skip this step, your entry might get corrupted during subsequent UPDATE runs.
14. Now, you can use it as described in Section 2.1.

Surely, this process sounds quite time consuming. But just think about how much time you spend for writing a single reference entry for a working paper. Then add the time you spend for reformatting the entry, because somebody complains about the formatting. Then add the time you spend for reformatting the entry again, because you want to send the paper to a conference with its own reference style. Multiply that time with the number of times you write the same reference again for another paper. Multiply that with the number of persons at the IVT doing the same thing as you for the same reference:

$$T_{\text{BibTeX}} < T_{\text{no BibTeX}} = (t_{\text{add}} + t_{\text{reformat}} + t_{\text{rereformat}}) \cdot p \cdot n$$

We are pretty sure that the above cookbook for creating an entry is a little bit faster. By the way, LATEX is quite good at typesetting equations, too.

4 Other tasks

This chapter outlines solutions to more advanced tasks like creating a table, counting words (!), creating reference lists and converting to word processor formats.

4.1 Excel2LaTeX

Since it is cumbersome to write and edit a table in LaTeX, a nice tool can be used to convert Excel tables to LaTeX. You can download it from `http://www.ctan.org/tex-archive/support/excel2latex/`. It will be introduced in the workshop.

4.2 Counting words

Due to the nature of LaTeX, counting words in the source file can only serve as a rough estimation: Unless you process the whole document, it cannot be determined if a sequence of characters is a word that will appear in the final document or if it is, say, a variable in an equation, a cell in a table, or a parameter to a macro. To obtain a more or less reliable word count in a LaTeX document, it has to be processed in a special way.

There is a script that automates this process and even creates a *tex* file for inclusion into your document. It requires *Python*, version 2.6 or later.

- Compile your document as shown in Section 1.2.
- From your working directory, call the script `wordcount.py` in the `_latexfiles` directory. For `papers/workingpapers/recurring/template`, you would run `python ..\..\..\..\_latexfiles\wordcount.py Template.tex` from a *cmd* window. (Omit the `..\..\..\..\` if `_latexfiles` is in your paper's directory.)
- This generates a file named `mywordcount.tex` in your working directory. You can add this to SVN and `\input` it from your LaTeX document. (The `trb.tex` template already does this for you – see below.)

For convenience, you can also create a *cmd* or *bat* file or a Windows shortcut (*lnk*).

TRB has a limit on the number of words. In addition, each figure and table counts as 250 words. There is some logic in the new `trb.tex` template (the one in `_latexfiles/_layouts`) that saves you from computing the "word equivalents". For existing papers, just remove any

definition of `\mywordcount` from your document; then, the one provided by the template is used.

There are two ways to tweak this:

Exclude parts of text The LaTeX macro `\ifwc` and `\ifnwc` allow including or excluding parts of the document if in "word count" mode. In case you want to exclude, say, the abstract and the bibliography, from the word count, use the following construct in your document:

```
\ifnwc{%
  \renewcommand{\createabstract}[1]{}%
  \renewcommand{\bibliography}[1]{}%
}%
```

Turn off the feature Redefine the command `\mywordcount`:

```
\renewcommand{\mywordcount}{X words + Y figures + Z tables}
```

Note that this has to occur in your main file just after `\begin{document}` but before `\createtitlepage`.

Here are some details on the counting algorithm:

- Hyphenated words like "within-day" count as one (just like *Word* counts them)
- Formulae (both inline and separate) count as one word each
- Literature references count as one word
- Text inside figures and tables is not counted
- The title page and header/footer are not counted

The algorithm aims at replicating *Word*'s behavior or at least underestimating the word count.

4.3 Create a PDF file with all Bib-Entries

As noted in Section 3.5.2, a list of references is created as a byproduct of checking the BibTeX database. Run the `check.cmd` script in the `_latexfiles` directory and examine the output in `_latexfiles/unsortbibs/example.pdf`.

If you want to create the references for German papers, open the file `example.tex` in the `_latexfiles/unsortbibs/` directory and change the definition of `\myfirstlang` from

`\newcommand{\myfirstlang}{english}`

to

`\newcommand{\myfirstlang}{german}`.

Save it and compile the file again with `latex2pdf.bat`.

4.4 Create your own reference list

The BIBTEX environment can also be used to create your own reference list (e.g., for the website, for a yearly report, or to show off). An example is shown in `misc/reflist/muelleki`. This LATEX document creates all references in a predefined order. If you now want to have your own reference list created, we suggest using this as a template. Follow the instructions in Section 3.4.2 to create a copy of muelleki's reference list, also rename the file `muelleki.tex` accordingly. Edit the renamed file to replace the `\bibentry{...}` commands with your own references. Section 2.1 shows how to find the keys for existing references in the BIBTEX database to be used inside the `\bibentry{...}` command. In Section 3.5 the process of adding new entries to the database is described.

4.5 Conversion to Word format

Several journals (e.g., TRR) and even conferences (e.g., CUPUM) require you to submit in Microsoft Word format. Unfortunately, there is no "perfect" way to obtain an exact copy of a LATEX document in Word format. If your content is already in LATEX and you need to convert it to Word for a submission, you have the following options:

- Copy-paste your text into Word
- In Adobe Acrobat Pro, save the PDF as a Word document and edit from there
- Follow the instructions in this section

The first two options are straightforward but require substantial editing. The workflow proposed below provides the following benefits:

- Formatting of characters is preserved
- Sectioning, cross references and citations just work
- All graphic formats are automatically embedded, *PDF* files are converted to *PNG* files at a resolution of 300 dpi
- Tables are embedded in their LATEX representation as *PNG* files
- Equations are converted to the MathML-based Word 2007 equation editor format

As a result, you can still write your document in the LATEX environment and convert to Word just before submission

The conversion is unidirectional and requires some preparation and postprocessing. It has several limitations, some of which will be covered below, The *Open Document Format* is used as intermediate format.

4.5.1 Preparation

Install required software

1. *LibreOffice* (tested with version 3.6)

2. *Microsoft Word 2007* or later (tested with version 2010)
3. *ImageMagick* with command-line utilities (e.g., *convert*) in the `PATH`
4. *Java*

Make sure that your document compiles without error The conversion is just another way of compiling your document, and might fail or produce strange results if the document syntax is incorrect. For example, using `\\` to start a new paragraph is not a syntax error but will produce strange results – a new paragraph is started after each empty line in your source file, or using the `\par` command.

Convert subfigures to regular figures Unfortunately, the conversion to *Word* does not support the `\createsubfigure` and `\createsubtable` constructs. A possible workaround is:

1. Compile your paper to PDF.
2. Extract graphical representations for the figures that are made of subfigures, e.g. using *Adobe Acrobat*. Save them as individual PDF files.
3. Change the code that creates the subfigures. The `\ifelseht` command comes handy here:

```
\ifelseht{
  \includegraphics[...]{combined-subfigs.pdf}
}{
  \createsubfigure{...}{...}{...}{...}
  \createsubfigure{...}{...}{...}{...}
}
```

Change the document layout The `ivt-generic` layout (cf. Section 1.2.1) is not particularly well suited for conversion to Word. At the time of writing, there are two layouts that work well:

trr if you are writing for the Transportation Research Record

word for everything else

Set up a consistent resolution for auto-generated images (optional) This step depends on the local computing environment and unfortunately cannot be generalized.

1. Locate the file `tex4ht.env` on your system and copy it to your paper's main directory.
2. Open the copied file with a text editor.
3. Substitute the line

   ```
   Gdvipng -T tight -x 1400 -D 72 -bg Transparent -pp %%2:%%2 %%1 -o %%3
   ```

   just after `G.png` by the following two lines:

   ```
   Gdvips -E -D 300 -pp %%2:%%2 %%1 -o %%3.ps
   ```

```
Gconvert -density 300 %%3.ps %%3
```

If you omit this step, images that are automatically generated from your LATEX document (such as tables) will be included at an inferior resolution not suitable for printing.

4.5.2 Compilation and inspection

The compilation process can take several minutes even on a fast machine.

Windows Execute the scripts `latex2pdf.bat` and `latex2word.bat` in your paper's directory. (Copy the scripts from `https://repos.ivt.ethz.ch/svn/ivt/doc/trunk/papers/workingpapers/recurring/template/` if you do not have them.)

Linux/Mac OS X Execute `make <main>.odt` in your paper's directory, substitute `<main>` by the name of your master document file without extension.

In any case, a file `<main>.odt` should be created.

Equations Open the `ODT` file in *LibreOffice*. Check all equations, they might have been converted incorrectly. (Examples are `km$^2$` and `$\hat{\alpha}_{bc}$` which are correctly typeset in LATEX (km^2, $\hat{\alpha}_{bc}$) but incorrectly converted. Consider also editing the source file to use a friendlier expression, and recompiling. (For example, `$\text{km}^2$` is typeset and converted correctly.) Also, make sure to prefer `\num` or `\SI` from the `siunitx` package over putting standalone numbers into equations: This avoids creating an equation object for every number you have in your document.

Badly converted parts Sometimes LATEX code (e.g., listings) cannot be translated properly. These parts can be embedded as pictures just the way they are produced by LATEX. To achieve this, edit the source and put the problematic code into an `\AsPicture{}` command. Example: `\AsPicture{\rule{3em}{1ex}}` will produce a horizontal rule that is 3 em wide and 1 ex thick (▬) and automatically embed its pictorial representation as a *PNG* file. (This is how all tables are embedded.) This command works also when compiling to *PDF*, so you can just keep it in the code. Recompile as necessary.

4.5.3 Postprocessing

Save the file in LibreOffice Open the `ODT` file and save it as `DOCX` even if you do not have any equations. Otherwise opening it later in Word might fail.

Convert to Word Open the `DOCX` file in *Microsoft Word* or later. Trust the document if *Word* complains that it cannot open it. Save the document again in the `DOCX` format.

Resize graphics Graphics are converted at 300 dpi and usually appear oversized. Rescale them as required, keeping the aspect ratio. Use 32 % of the original size as a starting point.

Layout Apply the *Word* layout required by the journal or conference. Define page breaks, header and footer lines. Set document properties (title, authors, . . .). Check the placement of figures. Reformat the tables that surround display equations.

A Unsupported Software

This appendix lists popular software for which support cannot be provided.

A.1 WinMerge

WinMerge (`http://winmerge.org/`) is a very nice tool to compare two different text files. It shows the difference between them line by line. The tool does not work for binary files, e.g., *zip* files.

1. Navigate to `[My Documents\install]\A.1_winMerge`
2. install `WinMerge-2.10.2-Setup.exe`
3. choose ***typical installation***, add ***Plugins***, leave ***Integrate into TortoiseSVN*** checked

If you now right-click on a file, you will find ***WinMerge*** in the context menu.

(Installation of this tool is optional, since *TortoiseSVN* comes with its own file comparison tool.)

A.2 TEX2html

The program *TEX2html* creates a HTML page out of a LATEX document (`http://hutchinson.belmont.ma.us/tth`). It knows only very basic functionalities and therefore, the resulting `HTML` file looks a little bit crappy. But it is still useful for a simple conversion.

1. Navigate to `[My Documents\install]\A.2_tth` and unzip `tth_exe.zip`
2. create the directory `[C:\tth]` (Note: It is highly recommended to install it directly under `C:` and to use a folder name with no special characters in it, e.g., spaces).
3. copy all unzipped files into `[C:\tth]`
4. ***Start→Control Panel*** (if you are working on Windows Vista or Windows 7 ensure that you have a classic view of the control panel)
5. Double-click ***System***
6. ***Advanced→Environment Variables***
7. In ***System variables*** double-click on ***Path***
8. In ***Variable value*** add (**do not overwrite!**) `[C:\tth];` at the **very beginning** of the line. →***OK***

The last few items adds `[C:\tth]` to the execution path of *Windows*. With that, you can run those executables in any directory of your machine. To check if the settings are correct, just open a *cmd* (command window) and type in *tth -help*. A copyright Information should appear.

A.3 LaTeX2rtf

The program *LaTeX2rtf* (`http://latex2rtf.sourceforge.net/`) creates an RTF document out of a LaTeX document. It knows only very basic functionalities and therefore, the resulting `RTF` file looks a little bit crappy. But it is still useful for a simple conversion.

1. Navigate to `[My Documents\install]\A.3_latex2rtf`
2. install `latex2rtf-1.9.19a_win.exe`

Finally, you have installed everything you need to gather, compile, write and commit LaTeX papers inside the IVT BibTeX / LaTeX environment.

A.4 WinEdt

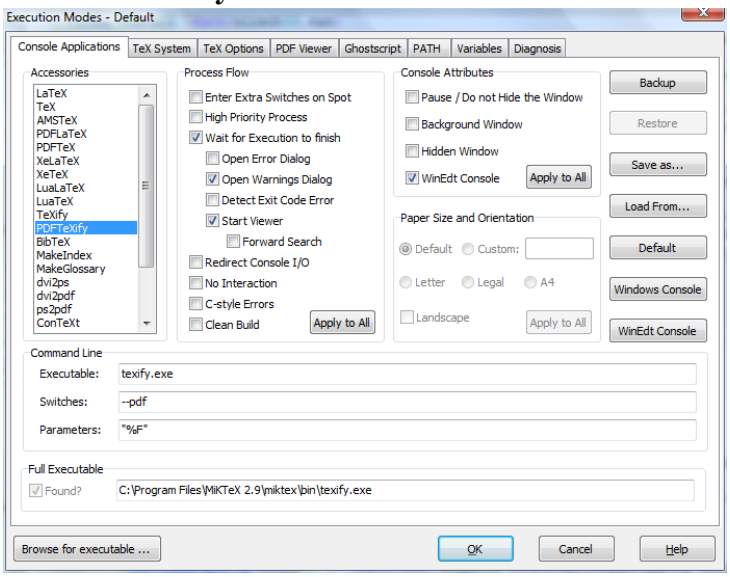
WinEdt is a very useful program to write LaTeX documents (`http://www.winedt.com/`), but it is not necessary. Most colleagues of the IVT that are already work with the environment are using this program already.

A.4.1 Installation and Setup

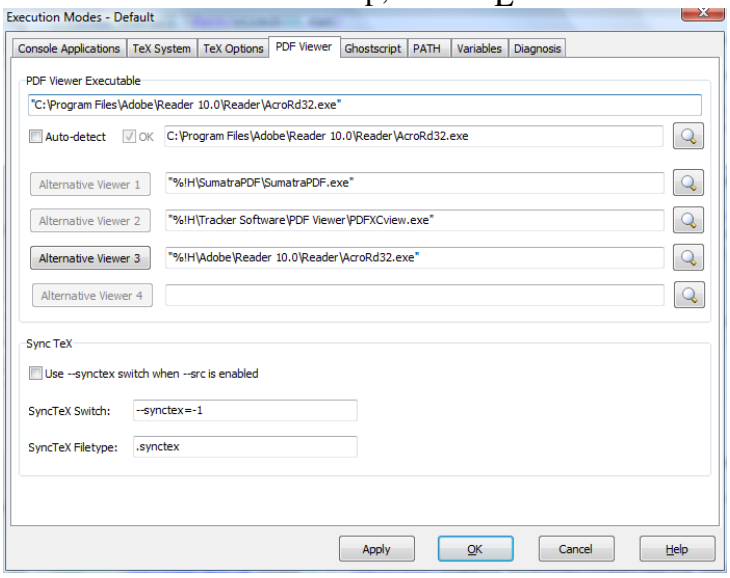
WinEdt offers a huge amount of functionalities and different configuration possibilities. The configuration shown here is just a simple one, but it's good enough to work efficiently with *WinEdt* in combination with the IVT BibTeX / LaTeX environment. If a configuration is not shown, it is kept to the default.

1. Navigate to `[My Documents\install]\A.4.1_winEdt`
2. Install `winedt60.exe`
3. Start *WinEdt*. It shows the *Configuration Wizard* please close it
4. ***Options→Options Interface...***
5. In the left menu ***Formatting: Wrapping,Environments...→Wrapping***
6. Overwrite the current file content with the `[My Documents\install]\14a_winEdt\Wrapping.ini` file content..
7. Save the file
8. In the left menu ***Backup, Auto Saving, File Status...→Backup***
9. Overwrite the current file content with the `[My Documents\install]\14a_winEdt\Backup.ini` file content..
10. Save the file
11. In the left menu ***Backup, Auto Saving, File Status...→Auto Saving***

12. Overwrite the current file content with the
    `[My Documents\install]\14a_winEdt\AutoSave.ini` file content..
13. Save the file
14. Click on the ***Load Current Script*** button (in the up-left corner of the options interface)
15. open the *Execution Modes* via ***Options→Execution Modes...***. The only LATEX compile mode we will use is *PDF TeXify*. Therefore, we configure only this one.
    a) **→*Console Applications***
    b) select ***PDFTeXify***:



    c) select ***Acrobat*** in the tab strip, to use TEXWorks as PDF viewer:



A.4.2 Dictionaries

That's already enough to work with *WinEdt* nicely. *WinEdt* also provides dictionaries to correct typos. If you want to add and or replace the preset English dictionary with a German or a Swiss-German one, do the following:

1. Unzip the dictionary you like (e.g. `de_neu.dic`). It must be on the
   `[My Documents\install]\A.4.2_Dictionaries`
   folder
2. ***Options→Options Interface...***
3. In the left menu ***Dictionary Manager: Word Lists, Spelling...→Word Lists (Dictionaries)***
4. Add the next text to the end of the `dictionaries.ini` file (ensure to replace the `[My Documents\install]` folder for your working folder)

```
DICTIONARY="de_new"
FILE="[My Documents\install]\09_Dictionaries\de_neu.dic"
ENABLED=1
MODE_FILTER=""
LOAD_ON_START=0
SAVE_ON_EXIT=0
ADD_NEW_WORDS=0
USE_FOR_COMPLETION=0
ALLOW_COMPOUNDED_WORDS=0
```

5. Save the file

A.4.3 mergeAll.pl

Anytime you update/change/add bibliography entries in the environment, you need to run a special *Perl* script that can be found in the SVN at `ivt\doc\trunk\_latexfiles\mergeAll.pl`. Navigate to the folder tree `C:\[pathToYourSandbox]\_latexfiles` and run that script via a double-click. It creates the file `C:\[pathToYourSandbox]\_latexfiles\bibs\all.bi` (and others). To run that script from any folder (and therefore, from any program) on your machine, you need to create with a text editor a *bat* file containing the following lines:

```
  C:[1]
cd "C:\[pathToYourSandbox]\_latexfiles"[2]
mergeAll.pl[3]
```

Store that file as `[C:\putty\mergeAll.bat]`. You have already added `[C:\putty]` to the system path. So, you do not need to add another path to the environment variables.

1. ***Start→Run...*** and type in `cmd`
2. in the console, type in `mergeAll` and hit **return**

If many logging information appear ending with:

```
...
```

[1] change to that hard drive partition where your keep your *SVN* sandbox
[2] go to folder `_latexfiles`
[3] run the *Perl* script

```
done.
setting language specific header to ...
done.
done.
```

then the *Perl* script "`mergeAll`" can be run everywhere on your folder tree.

Now, if you want to call that program from inside of *WinEdt* you can add another execution button:

1. ***Options→Options Interface...***
2. In the left menu ***Menus and Toolbar→Main Menu***
3. Overwrite the current file content with the `[My Documents\install]\08_winEdt\MainMenu.ini` file content.
4. Save the file
5. Click on the ***Load Current Script*** button (in the up-left corner of the options interface)

Now, the mergeAll program can be executed inside *WinEdt* via ***Accessories→mergeAll***. A command window should appear and producing the logging output as you know from above. Then the window will disappear again.

A.4.4 Producing a PDF from an Existing Paper in the IVT SVN

To test, if everything is working fine, we will compile this working paper which is located in SVN at `C:\[pathToYourSandbox]\papers\workingpapers\2011\bibtex-latex-workshop`.

1. ***Project→Open Project...***
2. Navigate to the directory `var\lib\svn\ivt\doc\trunk\papers\workingpapers\2011\bibtex-latex-workshop`
3. open `bibtex-latex-workshop.prj`
4. click on ***Erase Working Files*** (recycle bin button)
5. run the `mergeAll` script with ***Accessories→mergeAll***
6. create the PDF by clicking ***PDF Texify*** (button with the teddy bear in front of the Acrobat icon)

Then, the PDF will be created and *Acrobat Reader* will show the resulting PDF.

Those three clicks should always be done to create the pdf from a LaTeX project. You can define keyboard shortcuts for the three clicks as seen above to make the use more comfortable.

B A short introduction to Subversion

SVN (or Subversion) is a version control system – a data storage with a version management. In simple words, several versions of the same file are stored at the same location with the same name. There are features that allow one to get any of those versions of the file. That means it (i) tells you the full history of a file, (ii) it (more or less) guarantees a specific location of that file and (iii) it is also an backup system for your files.

The following gives you a basic introduction how to use SVN. But first you need to know some important rules:

- A CHECKOUT produces a **local copy** (called **sandbox**) of the files on your machine. You can work with them even you are not connected. Modifying or deleting files or folders on your machine does not change anything in the repository. You can always get a fresh copy again.
- Differentiate between your sandbox and other files on your machine (which are not part of the repository). We highly suggest to create a special folder in which you check out files from the repository. Here, we are using `[My Documents\sandbox]` as the base path to the sandbox.
- **ALWAYS** do an CHECKOUT directly **in** the `sandbox` folder. **NEVER** check out something inside a subdirectory of your sandbox.
- **NEVER** rename files or folders that are part of the local copy of the repositories using *Explorer*. Only rename files or folders via SVN. However, you can safely move or rename those folders that are at the top level of the sandbox.
- **NEVER** create two files or folders with the same name but different case in the same folder hierarchy level (e.g., `ivt/doc/papers/STRC` and `ivt/doc/papers/strc`). The (unfortunate) reason for that is that *Windows* is not case-insensitive, but SVN and most other file systems are. If that happens, some weird conflicts occur (only) on *Windows* machines.

Because of the above mentioned rules, it makes sense to keep the repository well organized, meaning that folder hierarchies should make sense. (Actually, that always makes sense, even one does not use SVN.)

Below is a description of the most frequently used commands, and a note on conflicts after concurrent changes.

B.1 checkout

Location In a NON-SVN folder use right click ***SVN Checkout...***

Purpose Gets the current version of a folder tree from the SVN repository

NOTE For the IVT SVN repository, use the URL `https://repos.ivt.ethz.ch/svn/ivt/doc/trunk` as base. You can browse the SVN repository by pushing the **...** button in the checkout dialog.

NOTE As long as you only want to use one local sandbox, you need to use CHECKOUT only once.

B.2 update

Location In a SVN folder use right click ***SVN Update***

Purpose Gets all the updates of the folder tree below the folder on which you used this command.

NOTE You can use this command any time you want. It does NOT destroy anything in your sandbox, it just gives you everything new. (However, see Appendix B.13 for a small amendment to above statement.)

NOTE Use is as often as needed.

B.3 add

Location In a SVN folder use right click ***SVN Add...***

Purpose Adds new files and folders to the SVN repository. (It is recursive).

NOTE Add new files and folders which you want to store in the SVN repository. Do not add unnecessary files (like temporary files).

NOTE Before you add files and folders to the repository, be sure that they are at an appropriate position and have appropriate names.

NOTE If you are working on a file, add it to the repository as soon as possible.

NOTE To finally put the file into the repository use COMMIT

B.4 commit

Location In a file in SVN use right click ***SVN Commit...***

Purpose Puts the changes of the file into the repository.

NOTE To check what has been changed in the files, use ***SVN Diff***. You can edit the files directly in the ***Diff*** view.

NOTE It is advisable to supply a short log message that describes the nature of the modifications, like “added”, “edited ACME section”, “reorganized paper”, “substituted Creator by

Generator", etc.

NOTE If you want to keep all changes, click ***OK***.

NOTE If you want to restore the file to the previous state, right-click this file (in the ***commit*** dialog or in *Explorer*) and select ***SVN Revert...***

NOTE If the file has been modified in the repository since the last update, the commit will fail. In this case, you will have to use the UPDATE command first. This may introduce conflicts if the changes overlap in the file, but this will not happen if you are the only one who edits your files.

NOTE Try to commit as often as possible, ideally after each "stable" state. By that, you can always easily revert to the last stable state in case your work moves into a wrong direction.

B.5 diff

Location In a file in SVN which is modified use right click ***SVN Diff***

Purpose Shows the changes of your file compared to the version in the SVN repository.

NOTE It does only work properly with text files. There is limited support for diffing `.doc` and `.xls` files; diffing other kinds of binaries does not work.

NOTE It makes sense to check your modifications first before you commit files to the repository.

NOTE The difference view is editable.

B.6 remove

Location In a file or folder in SVN use right click ***SVN→remove***

Purpose Removes a file from the repository so that it does not appear anymore when you do a CHECKOUT or an UPDATE.

NOTE After you have used the REMOVE command, you have to use a COMMIT to actually delete the file or folder from the SVN repository.

NOTE In fact, the file is not completely removed from the repository. There is always a possibility to get older versions from that file.

B.7 rename

Location In a file or folder in SVN use right click ***SVN→Rename...***

Purpose Moves a file or folder in the repository to a different location, retaining the full history. You can always determine the file's origin by looking at the change log.

NOTE After you have used the RENAME command, you have to use a COMMIT to actually move the file or folder in the SVN repository.

NOTE The RENAME command is implemented as a COPY and DELETE command.

NOTE To move a file or folder to a different folder, specify the relative path of the new location, e.g., `../destination_folder/new_file_name`.

NOTE Moving a big directory tree is faster when using BROWSE (see below)

B.8 copy

Location In a file or folder in SVN use right click ***SVN→Branch/Tag...***

Purpose Copies a file or folder from the repository to a different location, retaining the full history. You can always determine the file's origin by looking at the change log.

NOTE The target is a SVN URL, not a working copy path.

NOTE After you have used the COPY command, you have to use a UPDATE command to fetch the copied files from the repository.

B.9 log

Location In a file or folder in SVN use right click ***SVN Log...***

Purpose Lists all committed changes to that file or to all objects in that folder.

B.10 browse

Location In a file or folder in SVN use right click ***SVN→Repo-browser***

Purpose Allows display and modification directly in the repository.

NOTE All commands that change objects in the repository ask for a log message.

B.11 ignore

Location In a file or folder in SVN use right click ***SVN→Add to ignore list→*.extension***

Purpose Stops SVN from offering files for addition.

NOTE Files in your sandbox should be (a) added to the repository, (b) ignored, or (c) just not there. This simplifies the definition of a *clean copy*: The sandbox is clean if and only if the commit list is empty (with the ***Show unversioned files*** checkbox active).

B.12 revert

Location In a file or folder in SVN use right click ***SVN→Revert...***

Purpose Undoes all changes since the last commit.

NOTE This command helps you quickly return to a stable state if it turns out that your work went into a wrong direction since the last commit. Hence, it is advisable to *commit early and often*.

NOTE This also removes conflict markers (see below).

B.13 Conflict resolution

While no content is destroyed during the UPDATE command, an update might generate a *conflict* under the following conditions:

- You have uncommitted changes in your sandbox.
- Somebody else ("they") has changed a file that you have not committed yet.
- "Your" changes and "their" changes overlap.

This happens every once in a while after concurrent edits of the BIBTEX database. The reasons for this behavior are that Subversion is unable to decide how to merge two overlapping changes and leaves the decision to the user. Here are some suggestions to handle this situation.

B.13.1 The safe way

- When the UPDATE command notifies you about a conflict, choose ***Postpone conflict resolution***.
- Note the files with a *conflict marker* in *Explorer*.
- Use the DIFF command to record your pending changes on these files.
- Use the REVERT command on these files to return them to a stable state. This also removes the conflict markers.
- Manually redo your pending changes to the previously conflicted files.

B.13.2 The more challenging safe way

- When the UPDATE command notifies you about a conflict, choose ***Postpone conflict resolution***.
- Note the files with a *conflict marker* in *Explorer*.
- Use the DIFF command to edit your pending changes on these files. Manually remove the lines with the *conflict location indicators* <<<<<<<, ======= and >>>>>>> so that only "your" *truly intended changes* are shown in the difference view.
- Clean the conflict marker using the RESOLVE command (not described above).

B.13.3 A risky shortcut

- Choose the ***Accept mine-conflict*** option.

- Redo "their" changes with the help of the DIFF command so that only "your" *truly intended changes* are shown in the difference view.
- Clean the conflict marker using the RESOLVE command (not described above).

However this requires some discipline, as there is a risk to earn resentment from "them" if you accidentally undo "their" changes.