,

TECHNISCHE UNIVERSITÄT ILMENAU
Institut für Medientechnologie
Fakultät für Elektro-und Informationstechnik
Fachgebiet Elektronische Medientechnik

Masterarbeit

This is a very long title and complicated title for an absolutely incredible introduction to Typst - The LaTeX Killer

vorgelegt von

Max Mustermann
Matrikel: 123456

Betreuer:

Prof. Dr. Hans-Werner Unsinn
Dr. Oetker

Ilmenau, 20.04.2069

“What is reality?
Obviously, no one can say, because it isn’t words.
It isn’t material, that’s just an idea.
It isn’t spiritual, that’s also an idea;
a symbol…”
— Alan Watts.

Kurzfassung

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus.

Abstract

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus.

Acknowledgements

Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).
Thanks for everything (and nothing).

Contents

List of Figures

List of Tables

List of Listings (Code Blocks)

List of Abbreviations

DRR Direct-to-Reverberant Ration

LSP Language Server Protocoll

1 Introduction

This is a template for Bachelor and Master thsus or any other work that requires a nice looking template for showing your scientific in written form. You can use this template for your own work. If you need to collaborate with other people on this work you can have a look at **typst.app**. They provide an Online editor and hosting of your document as well as the possibility for collaborative writing, similar to **Overleaf** (LaTeX). Limited space per project and a stripped-down feature set is **free**.

If you migrate from LaTeX to Typst the **Guide for LaTeX users** might be helpful. Throughout this document you will find some basic Typst examples for typical thesis elements. Text elements with a small red circle in the top right corner are clickable and will lead you to the corresponding Typst documentation. The following chapters only a few selected examples taken from the official only resources[1,2].

1.1 Install Typst locally (offline)

As mentioned above, you can use the **typst.app** online editor for solo and collaborative writing.

If you want to use Typst locally you can find instructions to install Typst on your machine **HERE**.

[1], https://typst.app/docs

[2] https://github.com/typst/typst

Quickstart

For Windows users, install via Winget in the console:

```
winget install --id Typst.Typst
```

Mac users:

```
brew install typst
```

Linux: users:

View Typst on Repology°
or Snap°

Alternatively, you install Typst via the Rust toolchain if available on your on your system:

latest version:

```
cargo install --locked typst-cli
```

develop version:

```
cargo install --git https://github.com/typst/typst --locked typst-cli
```

1.2 Basic usage from command line window:

In your shell of choice (e.g. CMD or PowerShell on Windows) you can run the following commands.

Create `file.pdf ` in working directory

```
typst compile file.typ
```

Creates PDF file at desired path:

```
typst compile path/to/source.typ path/to/output.pdf
```

Watches source files and recompiles on changes:

```
typst watch file.typ
```

Lists available subcommands and options:

```
typst help
```

Print detailed info and usage of a subcommand (e.g. typst watch):

```
typst help watch
```

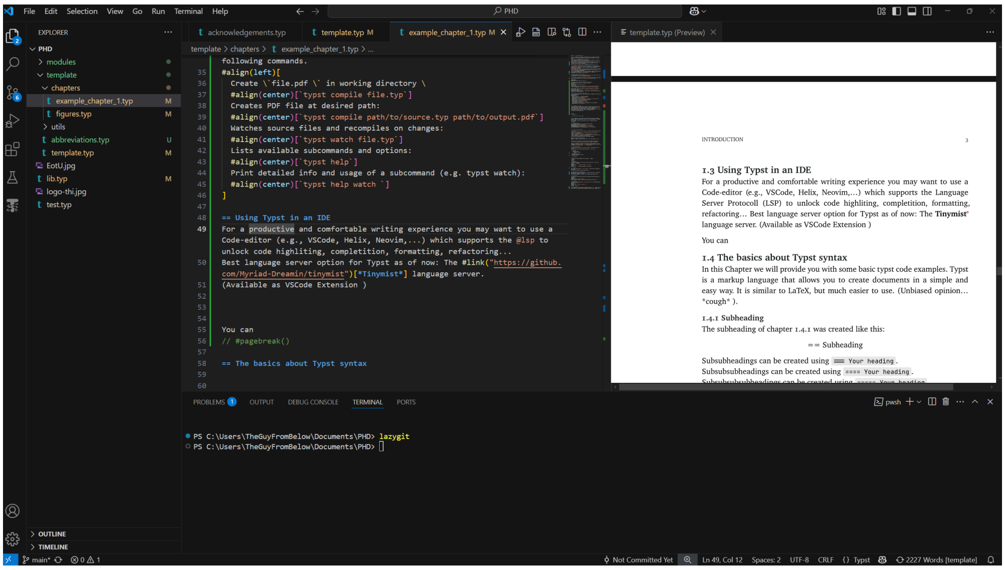
1.3 Using Typst in an IDE

For a productive and comfortable writing experience you may want to use a code editor which supports the Language Server Protocoll (LSP) to unlock code highliting, completition, formatting, refactoring…

Three recommended options:

- VSCode°
- Helix°
- Neovim° (if you dare)

Best language server option for Typst as of now: The **Tinymist**° language server. (Available as VSCode Extension)



```
following commands.
#align(left)[
  Create \`file.pdf \` in working directory \
  #align(center)[`typst compile file.typ`]
  Creates PDF file at desired path:
  #align(center)[`typst compile path/to/source.typ path/to/output.pdf`]
  Watches source files and recompiles on changes:
  #align(center)[`typst watch file.typ`]
  Lists available subcommands and options:
  #align(center)[`typst help`]
  Print detailed info and usage of a subcommand (e.g. typst watch):
  #align(center)[`typst help watch `]
]

== Using Typst in an IDE
For a productive and comfortable writing experience you may want to use a
Code-editor (e.g., VSCode, Helix, Neovim,...) which supports the @lsp to
unlock code highliting, completition, formatting, refactoring...
Best language server option for Typst as of now: The #link("https://github.
com/Myriad-Dreamin/tinymist")[*Tinymist*] language server.
(Available as VSCode Extension )



You can
// #pagebreak()

== The basics about Typst syntax


```

```
PS C:\Users\TheGuyFromBelow\Documents\PHD> lazygit
PS C:\Users\TheGuyFromBelow\Documents\PHD> 
```

INTRODUCTION 3

1.3 Using Typst in an IDE

For a productive and comfortable writing experience you may want to use a Code-editor (e.g., VSCode, Helix, Neovim,…) which supports the Language Server Protocoll (LSP) to unlock code highliting, completition, formatting, refactoring… Best language server option for Typst as of now: The **Tinymist**° language server. (Available as VSCode Extension)

You can

1.4 The basics about Typst syntax

In this Chapter we will provide you with some basic typst code examples. Typst is a markup language that allows you to create documents in a simple and easy way. It is similar to LaTeX, but much easier to use. (Unbiased opinion… *cough*).

1.4.1 Subheading

The subheading of chapter 1.4.1 was created like this:

== Subheading

Subsubheadings can be created using `=== Your heading`.
Subsubsubheadings can be created using `==== Your heading`.
Subsubsubsubheadings can be created using `===== Your heading`

Figure 1.1 – Using VSCode Editor for writing a Typst document. The Tinymist extension allows for syntax highliting as well as live-preview with hot-reload in keypress for instant viewing your changes.

1.4 The basics about Typst syntax

In this Chapter we will provide you with some basic typst code examples. Typst is a markup language that allows you to create documents in a simple and easy way. It is similar to LaTeX, but much easier to use. (Unbiased opinion… *cough*).

1.4.1 Subheading

The subheading of chapter 1.4.1 was created like this:

== Subheading

Subsubheadings can be created using `=== Your heading`.
Subsubsubheadings can be created using `==== Your heading`.
Subsubsubsubheadings can be created using `===== Your heading`.
…

1.4.2 Highliting text elements

Using markdown syntax you can **highlight** *things* or display them in plain `raw text`. You can also use `#text`° to highlight text. Example: `#text("This is a text")` yields: This is a text. `#text` provides a lot of options to customize the text. You can use it to create colored text, bold text, italic text, etc. You can also you Markdown syntax for styling text elements.

Example: To make a word bold you can use `*word*` which yields: **word**.

The blue box was created in the last line was created using the `#box`° function.

Labels for headings and figures or else can be created using the `#label` command and used with `@label[]` to reference them like this one here: ClickMe 1.4.1 (`== Subheading<sub>  ....  @sub[ClickMe]`).

1.4.3 Lists

Let's look at some lists°.
A simple bullet list can be created like this `- your bullet`.

- First item
    - Sub-first item
        - Subsub-first item
            - Direct-to-Reverberant Ration (DRR)
- Not the first item
- Definetly not the first item
- Last item (DRR)

You can also create numbered lists using the `+` symbol.

1. Second.
2. Third.
3. First.
4. Stupid list.

If you need to define a term a term you can you use the `/ TERM: *your definition ` command. Example:

`/ TERM: Description of the term`

yields:

TERM Description of the term

1.4.4 Comments

Just write them like this `/* Your Comment */` and they will be hidden in the output. You can also comment out whole lines using `//`.

1.4.5 Equations & Math-related stuff

Math equations are constructed similar to LaTeX. Math mode is created by wrapping the formula in `$`-signs. You can insert equation inlined or as a numbered stand-alone block.

INLINE MATH

You can write inline math like this by enclosing the math formula with `$`(NO SPACE). Example:

`$alpha < Theta^2$`

yields: $\alpha < \Theta^2$ which nicely integrates into your text.

BLOCK MATH

Or generate block math using `$ *your equation here* $` (With space before and after the `$`). Example:

`$ \alpha < \Theta^2 $`

yields:

$$\alpha < \Theta^2 \tag{1.1}$$

Block equations are consecutivley numbered like this `2.13`, where 2 and 13 would denote the chapter-number and global equation-count, respectively.

2
Figures and tables

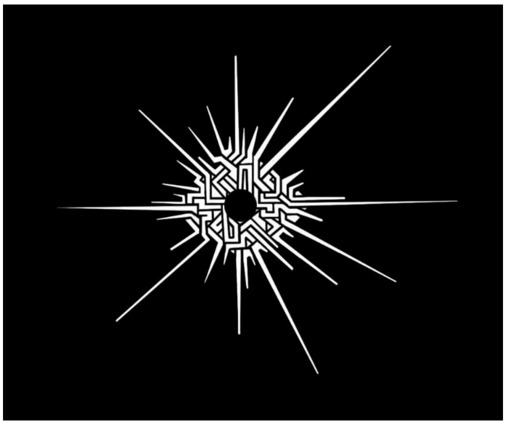


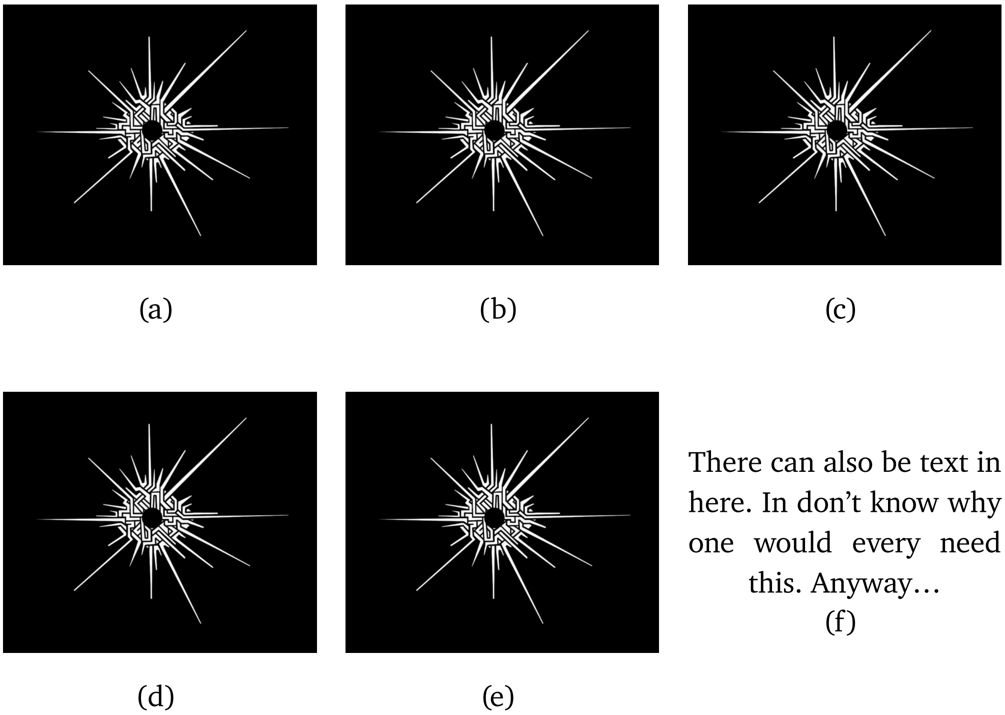
Figure 2.2 – This symbol was found at one the ruins on the Attle Rock. According to script fragments found near the side, it represents what the Nomai refer to as the *Eye of the Universe*.

This chapter shows some examples on how figures and tables work in Typst. Let's start with the basics. Figures are created with the `#figure` function. This creates a specific in which an `#image` can be placed (see Figure 2.2). You could also use the `#image` function directly, but this would not allow you to add a caption or a label. The label is used to reference the figure in the text. You can use the `@` symbol followed by the label name to reference it. For example, `@dd` will produce Figure 2.2.

2.1 Subfigures - Even numbers

Figure 3 shows an example on how to deal with subfigures. This template uses the **subpar**° package for creating subfigures. If you want to dig deeper into the

Typst language and create your own custom functions this package have a look at the respository. It is a nice example on how to make custom overrides.



(a) (b) (c)

There can also be text in here. In don't know why one would every need this. Anyway…
(f)

(d) (e)

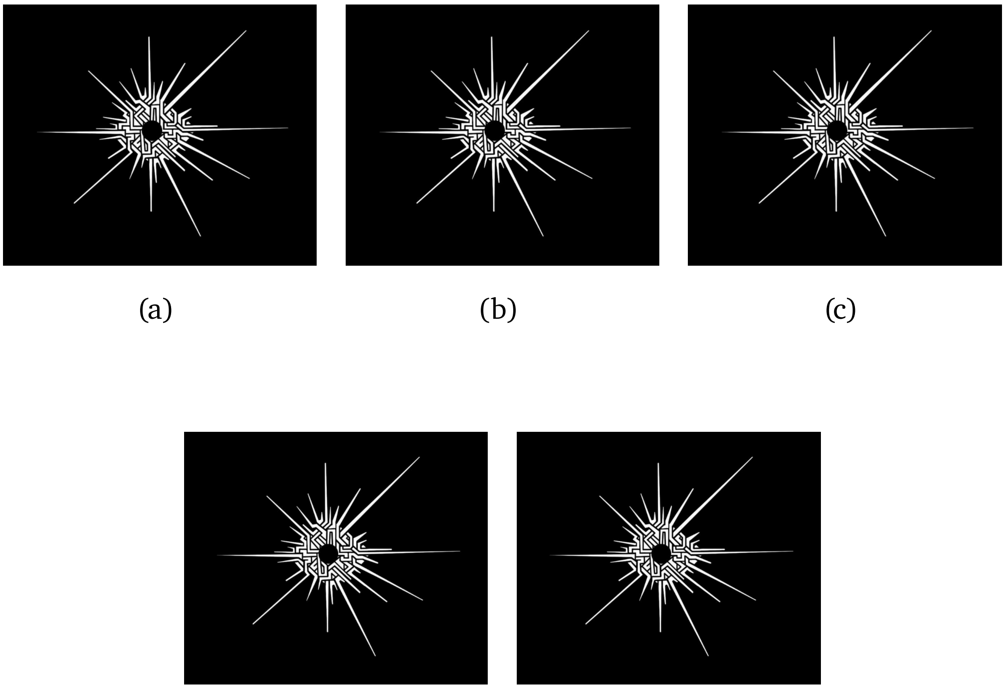
Figure 3 – This is an example of how to place subfigure. You can also reference, e.g. `@a` which produces Figure 3a or Figure 3d

Subfigures can be used to arrange multiple images / graphs in a single figure. The easiest way to do this is to use the `#subpar.grid` function. Figure 3 shows an example of how to use this function. The `columns` parameter specifies the number of columns in the grid. In this case, we have 3 columns. In the case of an even number of subfigures, the `#subpar.grid` function will automatically arrange the subfigures in a grid.

But how to nicely center subfigures in case of having an an odd number of subfigures?

2.2 Subfigures - Odd numbers

So here we have an example with an odd number of subfigures. The `#subpar.grid` function will automatically arrange the subfigures in a grid. However, we can not directly center n subfigures if we m columns and $m > n$. A workaround is to use the `columns` parameter in the `#grid` function. In example Figure 4 we have 5 subfigures. The first row specifies 3 columns with `columns(1fr, 1fr, 1fr)`. The second row specifies 4 columns with `columns(1fr, 2fr, 2fr, 1fr)`. This way you can insert empty (`[]`) block which centers the remaining 2 subfigures (Figure 4d, Figure 4e).



(a) (b) (c)

(d) (e)

Figure 4 – This example shows on how to nicely center subfigures in case of having an an odd number Using the `columns`-Parameter in the `#grid` function. The first colum is specified using same width factions for all three subfigures `colums(1fr, 1fr, 1fr)`. The second row specifies 4 columns with `columns(1fr, 2fr, 2fr, 1fr)`. This way you can insert empty (`[]`) block which centers the remaining 2 subfigures (Figure 4d, Figure 4e)

2.3 Tables

Let's look at some tables°.

Table 2.1 was created with the default styling of this template. This should be convenient for most cases. The caption for tables is automatically placed on top which is common for scientific literature. By putting tables inside a `#figure` you can also add a label to the table and reference it in the text. This way tables will be automatically numbered and added to the list of tables.

Table 2.1 – This is an example of a table. You can also reference, e.g. `@tab:Table1`. This table uses the template's default styling.

SUBSTANCE	EFFECTIVE DOSAGE	MISC
Nutella	1 table spoon	hot take: better with butter
Coffee	1 cup	10 cups (ask Lukas)
Spice	2 table spoons	$>$ 2 table spoons
Substance A	1.0 μ	10.2 Mμ
Substance A	1.0 μ	10.2 Mμ
Substance A	1.0 μ	10.2 Mμ

$$a = b \tag{2.2}$$

However, if you need to customize the table you can override the default styling of the table, look at the source code of Table 2.3. It was created in scope `#[ ]`. Thos allows to override the default styling of the table within this scope. You can also include images and math equations in the table. The table will automatically adjust the size of the columns and rows to fit the content.

	ITEM A	ITEM B
Lukas	5	7
Georg	5	7

Ignore this small table. This ist just for testing purposes. AAAAAAAAA

Table 2.3 – This is an example of a complex table. You can also reference, e.g. `@tab:Table2`. This table uses custom styling in a seperate scope. This table features text, images and math equations and even other tables.

<table>
<tr><th>Substance</th><th>Effective Dosage</th><th>Misc</th></tr>
<tr><td>Nutella</td><td>1 table spoon</td><td>hot take:
better with butter</td></tr>
<tr><td>Coffee</td><td>1 cup</td><td>10 cups (ask Lukas)</td></tr>
<tr><td></td><td colspan="2"><b>Shai-Hulud</b> recommends</td></tr>
<tr><td>Spice</td><td>1 sniff</td><td>10 sniffs</td></tr>
<tr><td>th</td><td>10 courses</td><td>$$\int_0^\infty \frac{x}{\left(\frac{\sin(x)}{d}\right)+2}dx$$</td></tr>
<tr><td>McRib</td><td>1 McRib</td><td>1 McRib</td></tr>
<tr><td>Table</td><td><table><tr><th></th><th>Item A</th><th>Item B</th></tr><tr><td>Lukas</td><td>5</td><td>7</td></tr><tr><td>Georg</td><td>5</td><td>7</td></tr></table></td><td><table><tr><th></th><th>Item A</th><th>Item B</th></tr><tr><td>Lukas</td><td>5</td><td>7</td></tr><tr><td>Georg</td><td>5</td><td>7</td></tr></table></td></tr>
</table>

On the next page we are going to check out some Code listings!

2.4 Code Listings

Let's make a small listing. Code blocks are also wrapped in a caption.
Like this:

~~~
#figure()[
  ```rust
    println!("yeah");
  ```]
~~~

Code highliting is supported for a a lot of languages. This is what Rust Code (Listing 2.1) looks like in the Code Block. Adding a caption registers the code bock to the List of Listings - Outline in the front matter section.

```rust
src/main.rs
// This main function doesn't do shnit
pub fn main() {
  let g: i32 = 0;
  let mut aah: String = String::new();

  aah.push("A");
  for i in 0..10 {
    aah.push("a");
  }
  aah.push("h!")
  println!("Stephan yells: {}", &aah);
}
```

🦀 Rust

Listing 2.1 – This function's only purpose is to print out the agony of Stephan after seeing with another "Underfull Box (Badness: 10000)" Warning

3 Bibliography

It is best to organize your literature in a bibtex file (.bib). Using ref-code in your bib-entry you can reference the entry with the `@`-smybol like this [1], [2], [2], [3], [4] ggdg [5], [6], [7], [8] gddgd [9], [10], [11], [12]

Bibliography

[1] Z. Meng, F. Zhao, and M. He, “The just noticeable difference of noise length and reverberation perception,” in *2006 International Symposium on Communications and Information Technologies*, 2006, pp. 418–421.

[2] J. S. Abel and P. Huang, “A Simple, Robust Measure of Reverberation Echo Density,” in *Audio Engineering Society Convention 121*, Oct. 2006. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=13819°

[3] P. Zahorik and E. Brandewie, “Perceptual Adaptation to Room Acoustics and Effects on Speech Intelligibility in Hearing-Impaired Populations,” in *Proceedings of Forum Acusticum 2011 Jun 27:2167-2172.*, Jun. 2011.

[4] S. A. Arboleda *et al.*, “ Beyond Looks: A Study on Agent Movement and Audiovisual Spatial Coherence in Augmented Reality ,” in *2024 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*, Los Alamitos, CA, USA: IEEE Computer Society, Mar. 2024, pp. 502–512. doi: 10.1109/VR58804.2024.00071°.

[5] S. Werner, F. Klein, T. Mayenfels, and K. Brandenburg, “A summary on acoustic room divergence and its effect on externalization of auditory events,” in *2016 8th Int. Conf. on Quality of Multimedia Experience (QoMEX)*, 2016, pp. 1–6. doi: 10.1109/QoMEX.2016.7498973°.

[6] S. J. Schlecht and E. A. Habets, “Accurate Reverberation Time Control in Feedback Delay Networks,” in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, Edinburgh, UK, Sep. 2017, pp. 337–344.

[7] P. D. L. G. Pestana, “Automatic Mixing Systems Using Adaptive Digital Audio Effects,” 2013.

[8] C. Schneiderwind, M. Richter, N. Merten, and A. Neidhardt., “Effects of Modified Late Reverberation on Audio-Visual Plausibility and Externalization in AR,” in *2023 Immersive and 3D Audio: from Architecture to Automotive (I3DA)*, 2023, pp. 1–9. doi: 10.1109/I3DA57090.2023.10289186°.

[9] A. Neidhardt, “Data set: BRIRs for position-dynamic binaural synthesis measured in two rooms,” in *5th Int. Conference on Spatial Audio, Ilmenau, Germany*, 2019. doi: 10.22032/dbt.39972°.

[10] A. Neidhardt and C. Schneiderwind, “The influence of the DRR on audiovisual coherence of a real loudspeaker playing virtually over headphones,” in *47th Annual Conference on Acoustics (DAGA), Vienna, Austria.*, 2021.

[11] A. Neidhardt, “Effect of impaired early reflection patterns on plausibility and similarity of position-dynamic binaural AR audio,” *Submitted*, 2025.

[12] A. Neidhardt and A. Zerlik, “The availability of a real hidden reference affects the plausibility of position-dynamic auditory AR,” *Frontiers in VR*, 2021, doi: 10.3389/frvir.2021.678875°.