FAKULTÄT
FÜR !NFORMATIK

Faculty of Informatics

Enhancement of Footwear Impressions

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Visual Computing

eingereicht von

Rebeka Koszticsak, Bsc
Matrikelnummer 01325492

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Robert Sablatnig
Mitwirkung: Dipl.-Ing. Manuel Keglevic

Wien, 4. Jänner 2020

Rebeka Koszticsak

Robert Sablatnig

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Enhancement of Footwear Impressions

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Visual Computing

by

Rebeka Koszticsak, Bsc
Registration Number 01325492

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Robert Sablatnig
Assistance: Dipl.-Ing. Manuel Keglevic

Vienna, 4th January, 2020

Rebeka Koszticsak

Robert Sablatnig

Erklärung zur Verfassung der Arbeit

Rebeka Koszticsak, Bsc
Address

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 4. Jänner 2020

Rebeka Koszticsak

Danksagung

Ihr Text hier.

Acknowledgements

Enter your text here.

Kurzfassung

Ihr Text hier.

Abstract

Shoeprint images are one of the most commonly secured evidences on crimescenes. Even though automatic shoeprint processing is a highly researched topic, the final identifacion is usually done by human forensic experts. The two main steps of shoeprint identification are enhancement and matching.

In this thesis the possibilities for enhancement of shoeprint samples from a real-life dataset are investigated. The main challange of this task is to correctly filter the pattern regardless the versitile, possibly heavily structured and clutterd noise on the samples. Two main approaches are examined, pattern enhancement and noise suppression. Among fully automated methods, a semi-automated technique is also tested, where user input is required for noise separation.

The main goal of this work is to find a universal approach which is able to filter and enhance the shoeprint data even in the presence of noise and the possible low image quality. Based on the experiences acquired while investigating the possible techniques a new noise-supression pipeline for shoeprint images is introduced. The noisy pixels are identified based on the Fourier-Mellin features of their multi-sized neighborhood. In the same time a model is built about the average appearance of noise, to eliminate that structure from the foreground as well. Additionally a gradient based line detector is also applied and the edge structures of the shoeprint are clustered to distinguish between pattern and noise edges. The experimental results show that the processed images are clearer, the pattern is sharper whereas the noise is either completely eliminated in the background or suppressed in the foreground. Furthermore based on the results of three differenet basic image descriptor features, the enhanced shoeprints have higher matching rate to their ground-thruth samples than the original images.

Contents

CHAPTER 1

Introduction

Shoeprints found on crimescenes can be important hints or evidences in a criminal investigation [KYZ14]. Event though on one thrid [Ale96] of crimescenes usable shoepatterns can be secured, there is no fully automatized algorithm available yet, which is able to identify and match those prints with the original shoe sole. Because of that human power is needed [WSYZ14] to recognize and analyze the found patterns. The work of forensic experts is not only time consuming and expensive, there is no guarantee about the objectivness of the final outcome[GBCN08], furthermore the stages of the human matching process are unclear and not necessarily reproducible.

There is an excessive amount of research already done [RBCP19] in order to help or repleace the work of forensic experst. There is however no algorithm published yet, which can be relaiably used in varying conditions and sample quality. One reason for that are the already mentioned versitile conditions, the features and properties of the pattern on the shoe, like age, material, etc., the characteristics of the ground where the shoeprint is left and enviromental conditions like for example the weather highly influence the overall quality of the acquired sample. Those high amount of factors result in changing appearance of the prints of the same shoe causing high intra class variance while clustering. Additionally there is a lack of universal, wide ranged database [RBCP19] which correctly depicts the common scenarios occuring on real-life crime scenes.

In 2014 a new database, called FID-300 [KAV14] was released which aims to solve the database problem described above. It contains over 1000 reference shoeprint patterns acquired in a laboratory. Moreover the database introduces 300 new shoeprint samples collected by the police providing an insight on images forensic experts are working on the daily basis.

1.1 Problem Definition

Instead of automatizing the entire shoeprint matching pipeline this work only concentrates on the possible ways of icreasing the sample quality. Because of the mentioned absence of general, appropiate database it is difficult to compare the already available methods. Furthermore it is also challanging to estimate which one is applicable in a real-life scenario. In this thesis

1.2 Challanges

1.3 Contribution

1.4 Structure of the Work

CHAPTER 2

Additional Chapter

Enter your text here.

CHAPTER 3

Introduction to LaTeX

Since LaTeX is widely used in academia and industry, there exists a plethora of freely accessible introductions to the language. Reading through the guide at `https://en.wikibooks.org/wiki/LaTeX` serves as a comprehensive overview for most of the functionality and is highly recommended before starting with a thesis in LaTeX.

3.1 Installation

A full LaTeX distribution consists of not only of the binaries that convert the source files to the typeset documents, but also of a wide range of packages and their documentation. Depending on the operating system, different implementations are available as shown in Table 3.1. **Due to the large amount of packages that are in everyday use and due to their high interdependence, it is paramount to keep the installed distribution up to date.** Otherwise, obscure errors and tedious debugging ensue.

3.2 Editors

A multitude of TeX editors are available differing in their editing models, their supported operating systems and their feature sets. A comprehensive overview of editors can

Distribution	Unix	Windows	MacOS
TeX Live	**yes**	yes	(yes)
MacTeX	no	no	**yes**
MikTeX	no	**yes**	no

Table 3.1: TeX/LaTeX distributions for different operating systems. Recomended choice in **bold**.

	Description
1	Scan for refs, toc/lof/lot/loa items and cites
2	Build the bibliography
3	Link refs and build the toc/lof/lot/loa
4	Link the bibliography
5	Build the glossary
6	Build the acronyms
7	Build the index
8	Link the glossary, acronyms, and the index
9	Link the bookmarks

	Command
1	`pdflatex.exe  example`
2	`bibtex.exe    example`
3	`pdflatex.exe  example`
4	`pdflatex.exe  example`
5	`makeindex.exe -t example.glg -s example.ist -o example.gls example.glo`
6	`makeindex.exe -t example.alg -s example.ist -o example.acr example.acn`
7	`makeindex.exe -t example.ilg -o example.ind example.idx`
8	`pdflatex.exe  example`
9	`pdflatex.exe  example`

Table 3.2: Compilation steps for this document. The following abbreviations were used: table of contents (toc), list of figures (lof), list of tables (lot), list of algorithms (loa).

be found at the Wikipedia page `https://en.wikipedia.org/wiki/Comparison_of_TeX_editors`. TeXstudio (`http://texstudio.sourceforge.net/`) is recommended. Most editors support the scrolling the typeset preview document to a location in the source document by `Ctrl` clicking the location in the source document.

3.3 Compilation

Modern editors usually provide the compilation programs to generate Portable Document Format (PDF) documents and for most LaTeX source files, this is sufficient. More advanced LaTeX functionality, such as glossaries and bibliographies, needs additional compilation steps, however. It is also possible that errors in the compilation process invalidate intermediate files and force subsequent compilation runs to fail. It is advisable to delete intermediate files (`.aux`, `.bbl`, etc.), if errors occur and persist. All files that are not generated by the user are automatically regenerated. To compile the current document, the steps as shown in Table 3.2 have to be taken.

3.4 Basic Functionality

In this section, various examples are given of the fundamental building blocks used in a thesis. Many LATEX commands have a rich set of options that can be supplied as optional arguments. The documentation of each command should be consulted to get an impression of the full spectrum of its functionality.

3.4.1 Floats

Two main categories of page elements can be differentiated in the usual LATEX workflow: *(i)* the main stream of text and *(ii)* floating containers that are positioned at convenient positions throughout the document. In most cases, tables, plots, and images are put into such containers since they are usually positioned at the top or bottom of pages. These are realized by the two environments `figure` and `table`, which also provide functionality for cross-referencing (see Table 3.3 and Figure 3.1) and the generation of corresponding entries in the list of figures and the list of tables. Note that these environments solely act as containers and can be assigned arbitrary content.

3.4.2 Tables

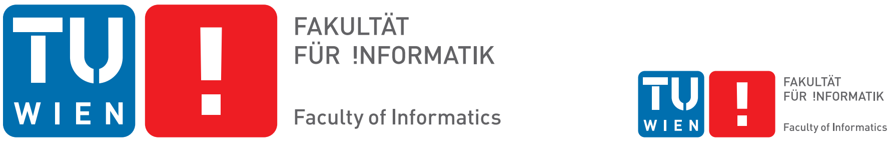
A table in LATEX is created by using a `tabular` environment or any of its extensions, e.g., `tabularx`. The commands `\multirow` and `\multicolumn` allow table elements to span multiple rows and columns.

Position		
Group	Abbrev	Name
Goalkeeper	GK	Paul Robinson
Defenders	LB	Lucus Radebe
	DC	Michael Duburry
	DC	Dominic Matteo
	RB	Didier Domi
Midfielders	MC	David Batty
	MC	Eirik Bakke
	MC	Jody Morris
Forward	FW	Jamie McMaster
Strikers	ST	Alan Smith
	ST	Mark Viduka

Table 3.3: Adapted example from the LATEXguide at `https://en.wikibooks.org/wiki/LaTeX/Tables`. This example uses rules specific to the `booktabs` package and employs the multi-row functionality of the `multirow` package.

3.4.3 Images

An image is added to a document via the `\includegraphics` command as shown in Figure 3.1. The `\subcaption` command can be used to reference subfigures, such as Figure 3.1a and 3.1b.



(a) The header logo at text width. (b) The header logo at half the text width.

Figure 3.1: The header logo at different sizes.

3.4.4 Mathematical Expressions

One of the original motivation to create the TeX system was the need for mathematical typesetting. To this day, LaTeX is the preferred system to write math-heavy documents and a wide variety of functions aids the author in this task. A mathematical expression can be inserted inline as $\sum_{n=1}^{\infty}\frac{1}{n^2}=\frac{\pi^2}{6}$ outside of the text stream as

$$\sum_{n=1}^{\infty}\frac{1}{n^2}=\frac{\pi^2}{6}$$

or as numbered equation with

$$\sum_{n=1}^{\infty}\frac{1}{n^2}=\frac{\pi^2}{6}. \tag{3.1}$$

3.4.5 Pseudo Code

The presentation of algorithms can be achieved with various packages; the most popular are `algorithmic`, `algorithm2e`, `algorithmicx`, or `algpseudocode`. An overview is given at `https://tex.stackexchange.com/questions/229355`. An example of the use of the `alogrithm2e` package is given with Algorithm 3.1.

3.5 Bibliography

The referencing of prior work is a fundamental requirement of academic writing and well supported by LaTeX. The BibTeX reference management software is the most commonly used system for this purpose. Using the `\cite` command, it is possible to reference entries in a `.bib` file out of the text stream, e.g., as [Tur36]. The generation of the formatted bibliography needs a separate execution of `bibtex.exe` (see Table 3.2).

Algorithm 3.1: Gauss-Seidel

```
Input: A scalar $\epsilon$, a matrix $\mathbf{A} = (a_{ij})$, a vector $\vec{b}$, and an initial vector $\vec{x}^{(0)}$
Output: $\vec{x}^{(n)}$ with $\mathbf{A}\vec{x}^{(n)} \approx \vec{b}$
for $k \leftarrow 1$ to maximum iterations do
    for $i \leftarrow 1$ to $n$ do
        $x_i^{(k)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j<i} a_{ij} x_j^{(k)} - \sum_{j>i} a_{ij} x_j^{(k-1)} \right)$;
    end
    if $|\vec{x}^{(k)} - \vec{x}^{(k-1)}| < \epsilon$ then
        break for;
    end
end
return $\vec{x}^{(k)}$;
```

3.6 Table of Contents

The table of contents is automatically built by successive runs of the compilation, e.g., of `pdflatex.exe`. The command `\setsecnumdepth` allows the specification of the depth of the table of contents and additional entries can be added to the table of contents using `\addcontentsline`. The starred versions of the sectioning commands, i.e., `\chapter*`, `\section*`, etc., remove the corresponding entry from the table of contents.

3.7 Acronyms / Glossary / Index

The list of acronyms, the glossary, and the index need to be built with a separate execution of `makeindex` (see Table 3.2). Acronyms have to be specified with `\newacronym` while glossary entries use `\newglossaryentry`. Both are then used in the document content with one of the variants of `\gls`, such as `\Gls`, `\glspl`, or `\Glspl`. Index items are simply generated by placing `\index{`⟨*entry*⟩`}` next to all the words that correspond to the index entry ⟨*entry*⟩. Note that many enhancements exist for these functionalities and the documentation of the `makeindex` and the `glossaries` packages should be consulted.

3.8 Tips

Since TeX and its successors do not employ a What You See Is What You Get (WYSIWYG) editing scheme, several guidelines improve the readability of the source content:

- Each sentence in the source text should start with a new line. This helps not only the user navigation through the text, but also enables revision control systems (e.g. Subversion (SVN), Git) to show the exact changes authored by different users. Paragraphs are separated by one (or more) empty lines.

- Environments, which are defined by a matching pair of `\begin{name}` and `\end{name}`, can be indented by whitespace to show their hierarchical structure.
- In most cases, the explicit use of whitespace (e.g. by adding `\hspace{4em}` or `\vspace{1.5cm}`) violates typographic guidelines and rules. Explicit formatting should only be employed as a last resort and, most likely, better ways to achieve the desired layout can be found by a quick web search.
- The use of bold or italic text is generally not supported by typographic considerations and the semantically meaningful `\emph{...}` should be used.

The predominant application of the LaTeX system is the generation of PDF files via the PdfLaTeX binaries. In the current version of PdfLaTeX, it is possible that absolute file paths and user account names are embedded in the final PDF document. While this poses only a minor security issue for all documents, it is highly problematic for double blind reviews. The process shown in Table 3.4 can be employed to strip all private information from the final PDF document.

	Command
1	Rename the PDF document `final.pdf` to `final.ps`.

2	Execute the following command:
	`ps2pdf -dPDFSETTINGS#/prepress ^`
	` -dCompatibilityLevel#1.4 ^`
	` -dAutoFilterColorImages#false ^`
	` -dAutoFilterGrayImages#false ^`
	` -dColorImageFilter#/FlateEncode ^`
	` -dGrayImageFilter#/FlateEncode ^`
	` -dMonoImageFilter#/FlateEncode ^`
	` -dDownsampleColorImages#false ^`
	` -dDownsampleGrayImages#false ^`
	` final.ps final.pdf`

On Unix-based systems, replace # with = and ^ with \.

Table 3.4: Anonymization of PDF documents.

3.9 Resources

3.9.1 Useful Links

In the following, a listing of useful web resources is given.

https://en.wikibooks.org/wiki/LaTeX An extensive wiki-based guide to LaTeX.

http://www.tex.ac.uk/faq A (huge) set of Frequently Asked Questions (FAQ) about TeX and LaTeX.

https://tex.stackexchange.com/ The definitive user forum for non-trivial LaTeX-related questions and answers.

3.9.2 Comprehensive TeX Archive Network (CTAN)

The CTAN is the official repository for all TeX related material. It can be accessed via `https://www.ctan.org/` and hosts (among other things) a huge variety of packages that provide extended functionality for TeX and its successors. Note that most packages contain PDF documentation that can be directly accessed via CTAN.

In the following, a short, non-exhaustive list of relevant CTAN-hosted packages is given together with their relative path.

algorithm2e Functionality for writing pseudo code.

amsmath Enhanced functionality for typesetting mathematical expressions.

amssymb Provides a multitude of mathematical symbols.

booktabs Improved typesetting of tables.

enumitem Control over the layout of lists (`itemize`, `enumerate`, `description`).

fontenc Determines font encoding of the output.

glossaries Create glossaries and list of acronyms.

graphicx Insert images into the document.

inputenc Determines encoding of the input.

l2tabu A description of bad practices when using LaTeX.

mathtools Further extension of mathematical typesetting.

memoir The document class on upon which the `vutinfth` document class is based.

multirow Allows table elements to span several rows.

pgfplots Function plot drawings.

pgf/TikZ Creating graphics inside LaTeX documents.

subcaption Allows the use of subfigures and enables their referencing.

symbols/comprehensive A listing of around 5000 symbols that can be used with LaTeX.

voss-mathmode A comprehensive overview of typesetting mathematics in LaTeX.

xcolor Allows the definition and use of colors.

List of Figures

List of Tables

List of Algorithms

Index

Glossary

editor A text editor is a type of program used for editing plain text files.. 5

Acronyms

CTAN Comprehensive TeX Archive Network. 11

FAQ Frequently Asked Questions. 11

PDF Portable Document Format. 6, 10, 11

SVN Subversion. 10

WYSIWYG What You See Is What You Get. 9

Bibliography

[Ale96] Girod Alexandre. Computerized classification of the shoeprints of burglars' soles. *Forensic Science International*, 82(1):59–65, 1996.

[GBCN08] Mourad Gueham, Ahmed Bouridane, Danny Crookes, and Omar Nibouche. Automatic recognition of shoeprints using fourier-mellin transform. In *2008 NASA/ESA Conference on Adaptive Hardware and Systems*, pages 487–491. IEEE, 2008.

[KAV14] Adam Kortylewski, Thomas Albrecht, and Thomas Vetter. Unsupervised footwear impression analysis and retrieval from crime scene data. In *Asian conference on computer vision*, pages 644–658. Springer, 2014.

[KYZ14] Xiangbin Kong, Chunyu Yang, and Fengde Zheng. A novel method for shoeprint recognition in crime scenes. In *Chinese Conference on Biometric Recognition*, pages 498–505. Springer, 2014.

[RBCP19] Imad Rida, Sambit Bakshi, Xiaojun Chang, and Hugo Proenca. Forensic shoe-print identification: A brief survey. *arXiv preprint arXiv:1901.01431*, 2019.

[Tur36] Alan Mathison Turing. On computable numbers, with an application to the entscheidungsproblem. *J. of Math*, 58:345–363, 1936.

[WSYZ14] Xinnian Wang, Huihui Sun, Qing Yu, and Chi Zhang. Automatic shoeprint retrieval algorithm for real crime scenes. In *Asian Conference on Computer Vision*, pages 399–413. Springer, 2014.