

DIPTYCHON

HANDBOOK

BJÖRN GOTTFRIED

MAY 31, 2016

Preface

This handbook describes the Diptychon software system for transcribing and analysing unrestricted handwritten documents. Diptychon is developed in the context of a DFG project with the reference numbers GO 2023/4-1, LA 3066/4-2, and LA 3007/1- $\{1,2\}$. The project has been initiated by

Dr. Mathias Lawo
Berlin-Brandenburgische Akademie der Wissenschaften
Monumenta Germaniae Historica

and by myself and is further accompanied by

Prof. Dr. Michael Menzel
Humboldt-Universität zu Berlin
Lehrstuhl für Mittelalterliche Geschichte und Landesgeschichte

and

Prof. Dr. Michael Lawo
Universität Bremen
Centre for Computing and Communication Technologies

I have developed the software together with Dipl.-Inf. Jan-Hendrik Worch and Dipl.-Inf. Marius Wegner. Jan and I have together designed the Diptychon system and Jan implemented a large part of the main functionality. Marius basically implemented the interactive fragmentation methods which help the user with post-editing the automatic suggestions for character segmentation.

A number of students have been helping in testing as well as implementing parts of the system. They include Timo Brose, Chokote Kekenou Armel Donald, Laurens Fischer, René Schröder, and Tobial Vahl. A number of Bachelor and Master theses have been written in the context of this project by the following students: Nicole Debowski, Christoph Eggert, Sophie Kohlberger, Benjamin Markowsky, Mascha Wagner, and Nils Wilshusen. Independent studies have been carried out by Benjamin Hemken and Marius Wegner.

We are thankful to Dr. Marianna Spano and Juliane Menzel, both Berlin-Brandenburgische Akademie der Wissenschaften, for testing the software and for giving us substantial feedback.

Terms of use

The Diptychon-System may be used for non-commercial research and teaching purposes only. Please make sure that the Diptychon-System is referenced in your bibliography and in any presentation by:

Björn Gottfried, Marius Wegner, Mathias Lawo (2015): Towards the interactive transcription of handwritings: anytime anywhere document analysis. International Journal on Document Analysis and Recognition (IJ DAR), 18 (1), 31-45, Springer.

Mai 2016,
Dr. Björn Gottfried

Contents

I	Introduction	1
1	Background	3
1.1	Processing documents	3
1.2	The status of Diptychon	4
1.3	Overview	4
2	Installation of the Diptychon system	5
2.1	Java runtime environment	5
2.2	The Diptychon system	6
2.3	Installing an update	6
3	A new project	7
3.1	Starting a new project	8
3.2	Saving a new project	10
3.3	Open an existing project	10
II	Document preparation	11
4	Enhancing document images	13
4.1	Zooming in and out	13
4.2	Displaying options	15
4.3	Change of panel size	15
4.4	Enhancing the contrast	15
4.5	Separation of text from the background	17
4.6	Text-background separation on the fly	19
4.7	Removing noise on the fly	19
5	Layout analysis	21
5.1	Text line detection	21
5.2	Text line corrections	23
5.3	Adjusting the height	24

III	Character segmentation and Transcription	25
6	The transcription process	27
6.1	Transcribing a text line	27
6.2	Entering the transcription first	28
6.3	Fragment editing	28
6.3.1	Connecting fragments	29
6.3.2	Cutting fragments into pieces	30
6.3.3	Fragmenting pieces by a rectangular tessellation	31
6.3.4	Fragmenting pieces by line segmentation	32
6.3.5	Reconciling the number of fragments and characters	33
6.3.6	Overlapping lines	33
6.4	Separating the glyphs first	34
6.5	Managing abbreviations	34
6.6	The word separation mode	35
6.7	Changing character assignments	36
6.8	The automatic transcription process	36
6.9	Correcting glyph-character assignments	37
IV	Working with transcriptions	39
7	Comparison methods	41
7.1	Find glyph images	41
7.2	The glyph gallery	42
7.3	Search for similar strings	43
8	Generating searchable PDFs	45
9	Export functions	47
10	Statistics	49
11	An example project	51

Part I

Introduction

Chapter 1

Background

This handbook describes the *Diptychon*¹ system and its employment. The main idea of Diptychon is the digital analysis of handwritten documents. One of the goals is the extraction of all glyph characters of a given document image and to assign to each glyph its corresponding character class.

In particular, Diptychon supports the transcription of handwritten manuscripts which are difficult to analyse automatically because of their complexity and variability. Complex handwritings make the collaboration between user and system necessary. The user has to spend some effort in *showing* the system how characters of a word are to be separated. Besides the separation of characters, the user has to interactively enhance the foreground-background separation, the detected text lines as well as single glyph images. While these interactions are quite cumbersome, the long-term objective of the system is to learn from the user how she transcribes the manuscript, so that the performance of the system grows during the transcription process of a document.

1.1 Processing documents

Diptychon is employed as follows. The image of a document page is to be loaded, enhanced by image pre-processing, and afterwards, the system tries to detect all text lines. While the document image is displayed on the left panel, the equally sized right panel provides empty text boxes for each text line found in the document image. The user has the opportunity to correct text lines, to delete them as well as to add new lines.

After that, the user transcribes the text, line by line, and the system learns how the glyph for each transcribed character looks like in the document image. For handwritten texts it is hardly possible to let the system automatically detect all the correspondences between the glyphs and the characters they are representing. For this reason, the system just proposes more or less accurate correspondences which the user needs to adjust. Those adjustments are learned

¹<http://diptychon.informatik.uni-bremen.de/>

by the system in order to automatically detect correct correspondences later in the document.

1.2 The status of Diptychon

Diptychon is work in progress. But the system can already be employed since methods have been implemented for all steps necessary. As the degree of automation has not been fully developed yet, there are still many user interactions necessary. As Diptychon is the result of a research project, the functionality presented in this handbook focuses on that part of the implementation which has already reached a certain maturity and which is part of the release that has been publicly made available on the Diptychon homepage¹ and the Digital Humanities-Dashboard².

1.3 Overview

The structure of this handbook is as follows. As it is the main purpose of Diptychon to support the separation of glyphs, which is also referred to as character segmentation, a substantial part of the software is devoted to this aspect. Character segmentation can be accomplished much better when enhancing the image first (chapter 4) and after having detected textlines (chapter 5) within a newly defined project (chapter 3).

Chapter 6 describes the core of the Diptychon-System, that is the possibilities to transcribe a given handwriting, together with the different possibilities to interactively enhance the character segmentation result. After the transcription a search routine as well as a gallery, showing the different glyph instances of a particular character class, are available, as shown in section 7. In chapter 8 a functionality is shown that enables to work with searchable documents even if the user has not installed the Diptychon system. Such searchable documents can be exported by Diptychon. Eventually, Chapter 10 shows how some fundamental document statistics can be computed and chapter 11 demonstrates an example project.

But first of all the installation of the Diptychon system is described in the following chapter.

²<http://dhdashboard.de/>

Chapter 2

Installation of the Diptychon system

First of all, in order to get Diptychon started, it is necessary to install the correct Java runtime environment, which is an older release that has been effective when the Diptychon project started. Newer releases generally also work, however, they entail a number of issues concerning the user interface. Therefore, it is advised to use java 7 in order to run Diptychon smoothly.

Note that it is possible to install more releases on your system in parallel. It is only necessary to address the correct release by considering the according folder hierarchy.

Diptychon has been developed platform independent. It has been tested on Mac OS, Windows, and different Unix systems. In the following the java releases of the Windows system are mentioned for an example installation.

2.1 Java runtime environment

At

<http://www.oracle.com/technetwork/java/javase/downloads/jre7-downloads-1880261.html>

several installation files can be found. The correct operating system is to be determined in order to select the correct installation file, such as

Windows x64 29.25 MB jre-7u40-windows-x64.exe

for 64-bit Windows systems, or

Windows x86 Offline 27.69 MB jre-7u40-windows-i586.exe

for 32-bit Windows systems. If you are not sure try initially the latter.

Figure 2.1 shows the web page that provides the JAVA installation files. The yellow arrow at the top left shows where to select a box in order to confirm the license agreement. At the bottom on the right hand side the red arrow points to the 32-bit installation file, while the yellow arrow shows the 64-bit installation file, both for windows (the screen shot is actually that one of the previous release). After having selected one of these installation files, the according file will be stored. It is to be executed afterwards. Thereby, it launches a dialogue box which guides the user through the installation of the JAVA runtime environment.

The screenshot shows the Oracle Java SE Runtime Environment 7 Downloads page. The main content area includes a license agreement section with two radio buttons: "Accept License Agreement" (selected) and "Decline License Agreement". Below this is a table of download links for various operating systems. The table has three columns: "Product / File Description", "File Size", and "Download".

Product / File Description	File Size	Download
_linux03	54.54 MB	rs-7u9-linux-i386.rpm
_linux03	45.77 MB	rs-7u9-linux-i386.tar.gz
_linux04	52.71 MB	rs-7u9-linux-x64.rpm
_linux04	44.52 MB	rs-7u9-linux-x64.tar.gz
Mac OS X	50.05 MB	rs-7u9-macosx-x64.dmg
Mac OS X	46.62 MB	rs-7u9-macosx-x64.pkg
Solaris x86	46.27 MB	rs-7u9-solaris-i386.pkg
Solaris SPARC	48.56 MB	rs-7u9-solaris-sparc.pkg
Solaris SPARC 64-bit	17.33 MB	rs-7u9-solaris-sparc64.tar.gz
Solaris x64	14.78 MB	rs-7u9-solaris-x64.tar.gz
Windows x86 C++ line	3.85 MB	rs-7u9-windows-i386-rtw.exe
Windows x86 C++ line	29.77 MB	rs-7u9-windows-i386.exe
Windows x86	38.47 MB	rs-7u9-windows-i386.tar.gz
Windows x64	31.18 MB	rs-7u9-windows-x64.exe
Windows x64	41.19 MB	rs-7u9-windows-x64.tar.gz

Figure 2.1: The Oracle web page provides the JAVA system.

2.2 The Diptychon system

A new folder called `C:\Diptychon` is to be created. The files `Diptychon.jar` and `start.bat` as well as the folder `libs` are to be copied into this folder.

2.3 Installing an update

In order to update the Diptychon system, it is sufficient to replace the file 'Diptychon.jar' with the new one.

Chapter 3

A new project

After the Diptychon system has been installed it can be launched by selecting the `start.bat` file. Although it is possible to directly open the jar-file, it is advised to use the `start.bat` file which allocates enough memory while starting the software. Figure 3.1 shows the user interface, after having launched the system for the first time.

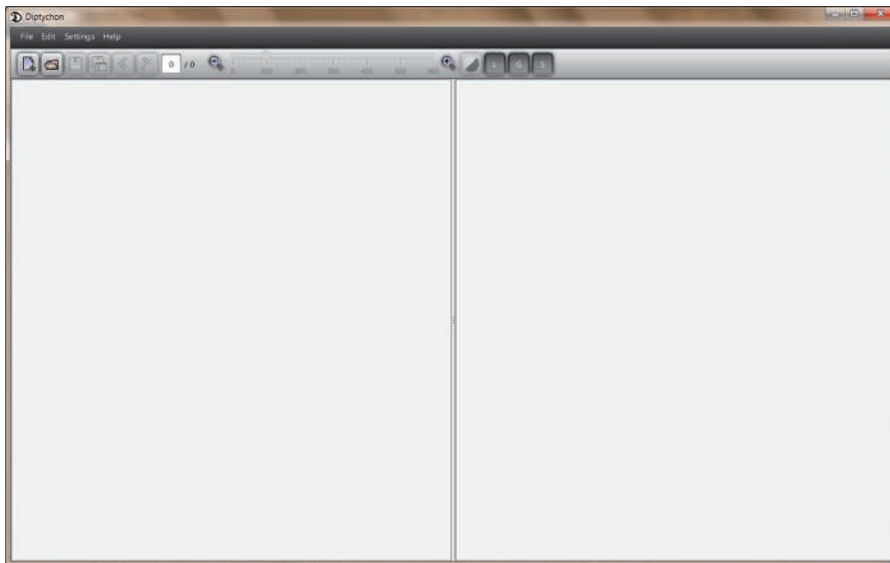



Figure 3.1: The Diptychon system when launching the system the first time. Your installation might look a little bit different, as the user interface is still updated, now and then.

3.1 Starting a new project

By File/New/Project or Ctrl+N or the left most icon  a file chooser dialogue is available. In the file directory a document image is to be selected, for example SBB-III A\Ms_Phill_1870_144_r_zugeschnitten.tif. This is shown in Figure 3.2 on page 8. If the image is not shown after a while it might be in a format that is not processable by Diptychon. In this case, the image can be converted with a programme like *Photoshop*, *Paint Shop Pro* or any other image processing programme. Diptychon can in particular read 'jpeg-' and 'tiff-files'.

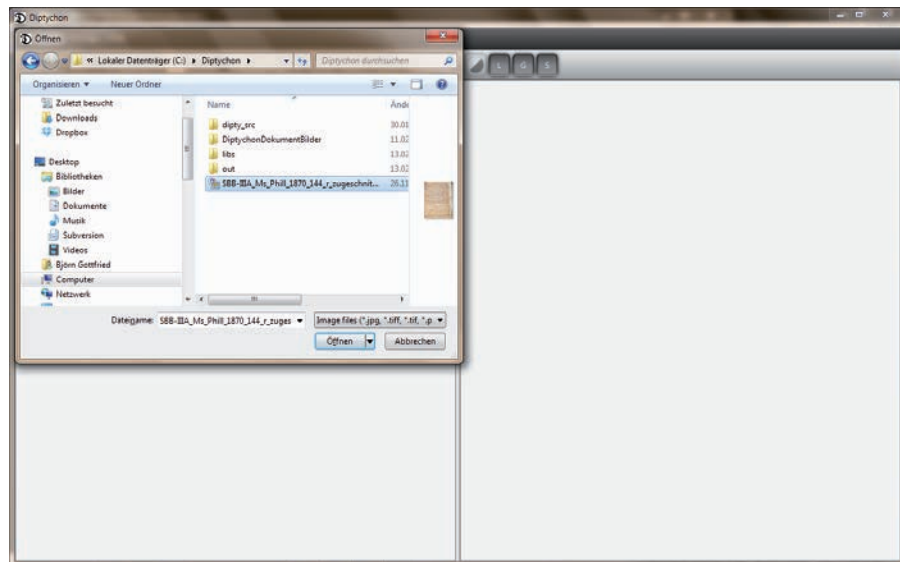


Figure 3.2: The dialogue for selecting a document image for a new project.

After a document image has been opened, the according image will be rendered in the left panel, shown in Figure 3.3.

The shown image will be analysed, as described in the following chapters. The empty space on the right hand side panel will be later used for the transcription.

The following Figure shows the title bar with the name of the complete folder of the document image.

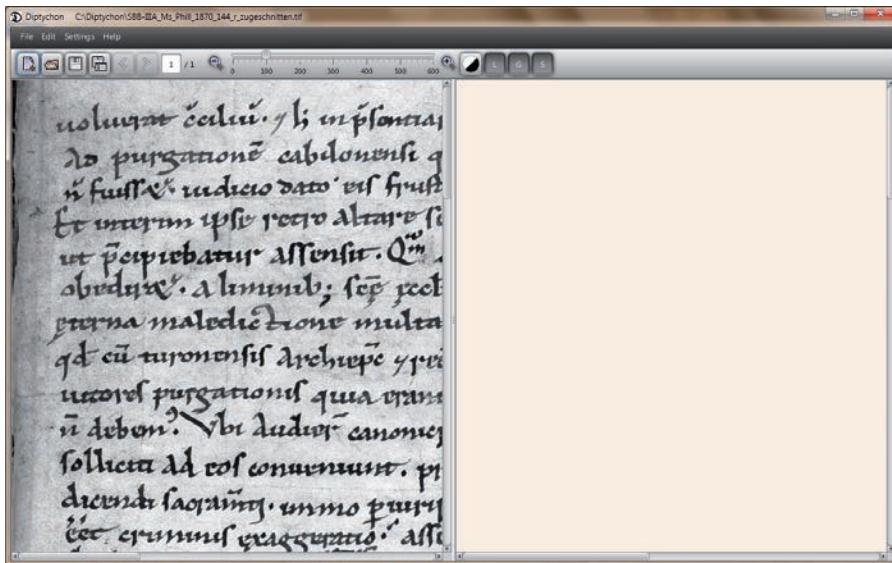
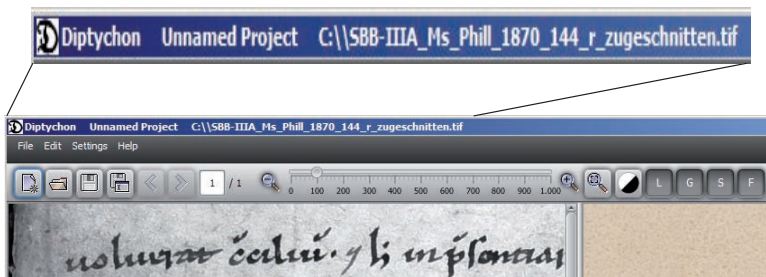


Figure 3.3: The opened document image for a new project.



3.2 Saving a new project

As long as the project has not been saved, the words **Unnamed Project** are found on the title bar. One should continuously save the current project. With

File/Save (as...)

the whole state of Diptychon is saved within a file which gets the file extension **.dsf** (**dsf** is short for *Diptychon save file*). With control-s (Strg-s, Ctrl-s) the status of the opened project can be easily saved. When the programme is launched again, one can continue to work wherever the work was interrupted.

It is advised to save all projects in a particular folder, for example:

c:\Diptychon\examples

The images which are used to launch a new project according to section 3.1 should also be placed in this folder.

Diptychon-projects can be easily exchanged and edited on different machines. One user can start a transcription, another one can continue it, just by sending her the image as well as *.dsf file. As the dsf-files might grow to some extent dependent on the complexity of the document, that is in terms of the number of characters, it is advised to use some file exchange system in the internet instead of eMail.

Certainly, subfolders for different projects can be used in the folder:

c:\Diptychon\examples.

3.3 Open an existing project

Whenever Diptychon is launched the last state is automatically recovered, as far as one has saved everything before leaving the programme. By means of

File/Open (recent)

a **.dsf** file can be chosen that completely saves the state of any project (**dsf** is short for *Diptychon save file*). A project includes a single image, the probably enhanced state of that image, and the current state of the transcription. With the aid of **.dsf** files different transcriptions of different documents can be processed in parallel or different editions of the same document can be dealt with in parallel by Diptychon.

Part II

Document preparation

Chapter 4

Enhancing document images

This chapter describes methods with which the document image can be inspected and enhanced. As Diptychon allows to load images which have already been processed by other image processing software, such as Paint Shop Pro or Photoshop, those applications can be deployed for enhancing the document images as good as possible. But Diptychon nevertheless provides a number of basic inspecting and image enhancement functions.

4.1 Zooming in and out

The zooming function enables the precise inspection of the document image. In order to get an overview of the whole image, one can zoom out of the image, as shown in Figure 4.1 or by the icon showing a magnifying glass.

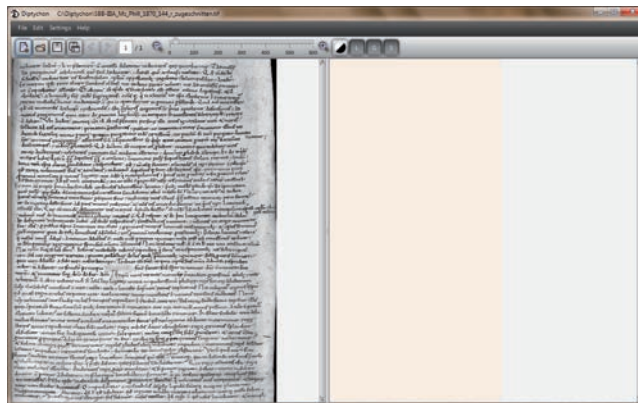


Figure 4.1: Zooming out.

By contrast, details can be made visible by zooming into the image, as shown in Figure 4.2.

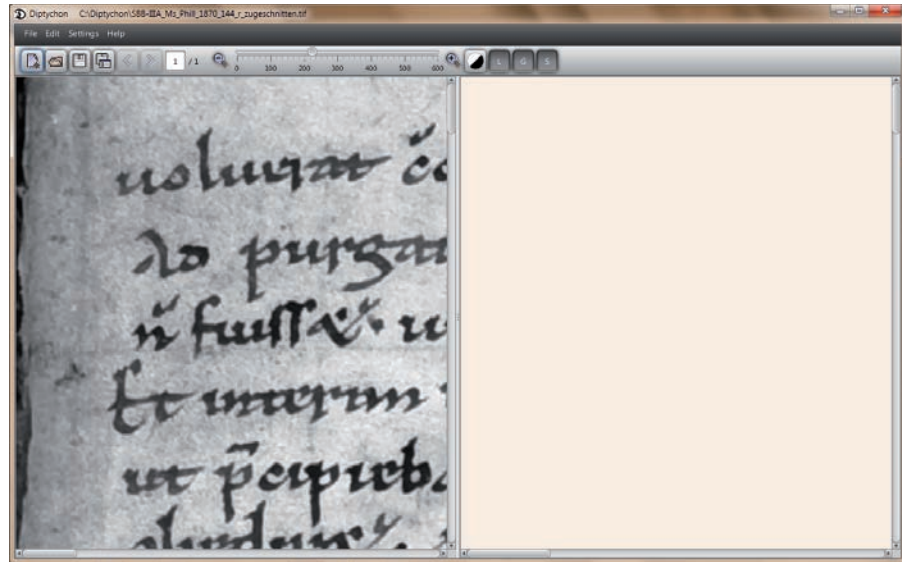


Figure 4.2: Zooming in.

Zooming is performed at the zooming bar, shown in Figure 4.3. The left hand side end of the zooming bar shows a magnifier with a minus sign contained in it. Pressing this magnifier by the mouse enables a stepwise zooming out function. By contrast, the right hand side magnifier allows the stepwise zooming in. The zooming bar itself can be employed stepless by selecting the ring on the bar by pressing the left mouse bottom and moving the mouse towards the left or right, for zooming in and out, respectively.



Figure 4.3: Zooming bar.

Right of the zooming bar, there is a second, differently looking zooming icon. Selecting it, the document image will be resized, so as to fit the whole page into the visible area of the screen. By contrast, when pressing the middle mouse button twice, the programme will zoom into the respective position within the image by 250%. This is frequently useful in order to efficiently zoom in at a specific location in the image.

4.2 Displaying options

Figure 4.3 shows four buttons which change the displaying modalities:

- **L**: enables the display of rectangles around text lines. This enables the user to see where text lines have been detected. Positioning the mouse cursor within such a rectangle a tool-tip shows the unique number of the text line. Additionally, a context menu is available which relates to single text lines, while the context menu is different when the text lines are not visible. Then, the menu relates to the image as a whole.
- **G**: is to display all glyph images which have already been extracted. This might help to get a better overview.
- **S**: synchronises the sliders of both panel windows, the document image and the transcription. Depending on the magnification it sometimes becomes necessary to desynchronise both panels: While the left hand side shows the glyph in an appropriate size, the transcribed text might not be visible to the extent required.
- **F**: just focuses the currently selected line, that is only glyphs extracted in that line will be displayed.

The latter option is significant to improve the system performance, at least as soon as a critical amount of glyphs have been extracted. The performance of the system has been tested with a document page that contains almost 5000 glyphs, a document page closely covered with writing.

4.3 Change of panel size

It is possible to change the size of the two panels, for example when more space is needed for inspecting the original document image. For this purpose, the mouse cursor needs to be placed mid of the screen at the vertical bar that divides the two panels. See the orange arrow in Figure 4.4. It is accordingly possible to enlarge the right hand side panel, for example when the user wants to get an overview of the text already transcribed.

4.4 Enhancing the contrast

The contrast can be enhanced by the context menu when placing the mouse on the image. A dialogue pops up which is shown in Figure 4.5 on page 16. A high contrast will generally improve the text-background separation which is described in section 4.5.

The histogram of grey values is shown when opening the dialogue window. The whole range of grey values should be used in order to get a high contrast. Dark values are on the left of the histogram, while bright pixels are found on

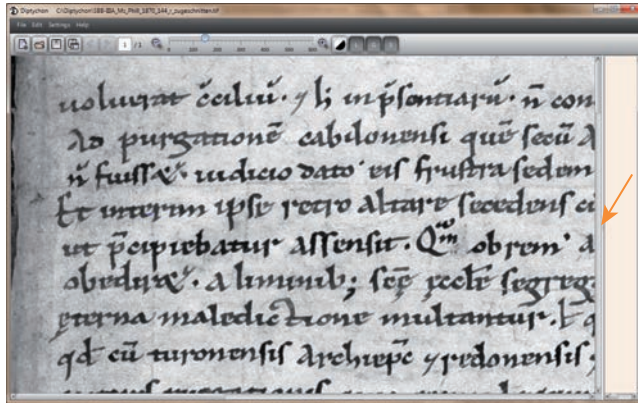


Figure 4.4: Changing the size of the panels.

the right hand side. The narrower the area which shows the grey values of the image, the weaker the contrast of that image.

When entering the dialogue the programme will automatically adjust the contrast as good as possible, so that the sliders are already at the most left and most right positions.

Employing the rubber band with the left mouse button, a rectangular area can be defined for which the contrast can be separately changed.

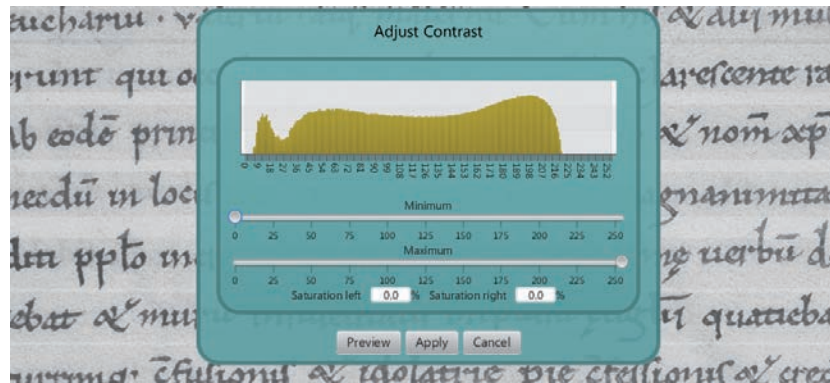


Figure 4.5: The contrast can be changed by two sliders. The histogram of grey values shows how well the contrast is. When entering the dialogue the programme will automatically adjust the contrast as good as possible.

4.5 Separation of text from the background

There is a little toggle button (orange arrow in Figure 4.6 on page 17). Pressing it the text is separated from the background. As a consequence, the grey value image changes into a black and white image.

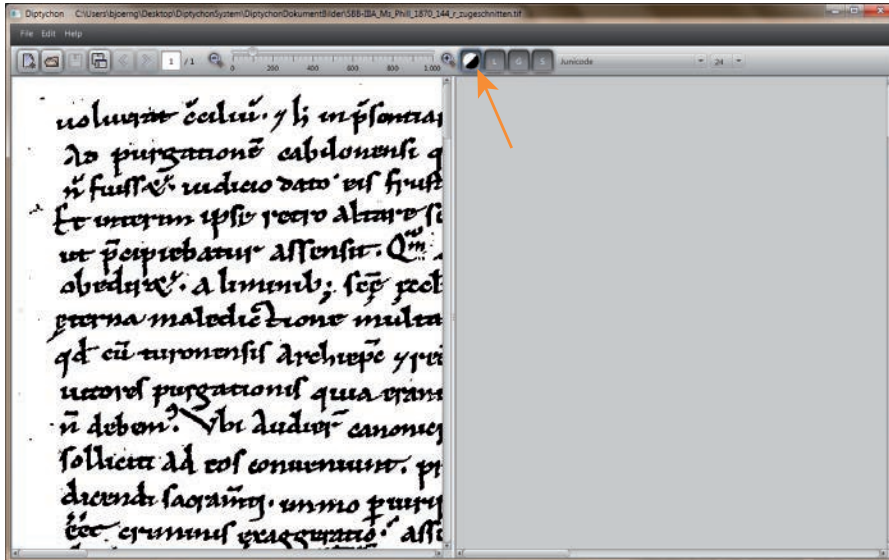
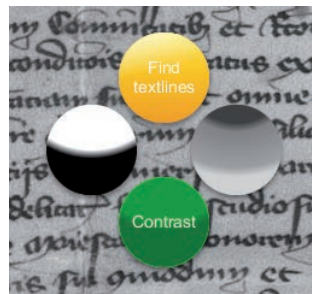


Figure 4.6: Separating the text from the background.

If the resulting black and white image is not satisfactory, other text separations can be tried out. For this purpose, the mouse is to be placed on the document image and by the right mouse button a context menu appears:



Selecting the item "Binarize Image", which is shown by the black-and-white ball, the dialogue shown in Figure 4.7 on page 18 pops up. The user should try out the two sliding controllers. Pressing "Preview" the changes are directly applied to the image. By means of "Apply" the complete image is changed accordingly.

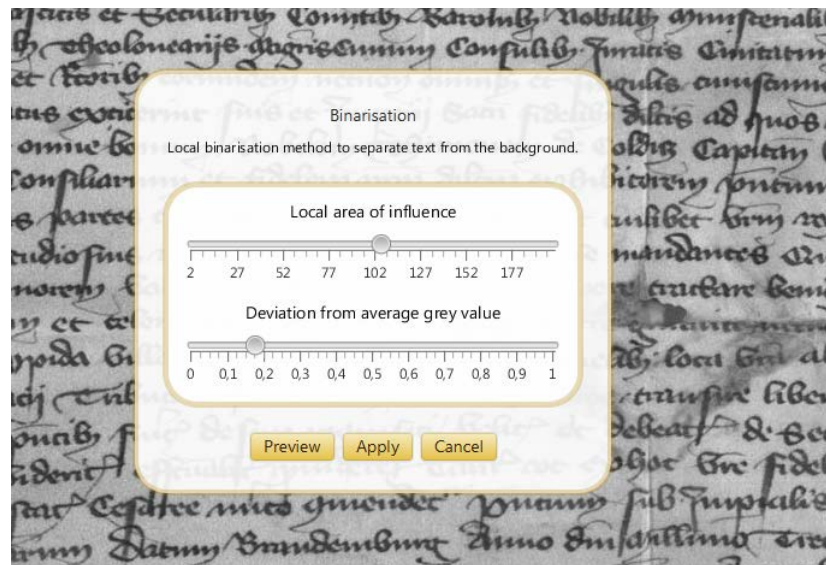


Figure 4.7: The text separation can be changed by two sliding controllers.

It is usually required to try out a number of different parameter configurations. This is necessary because it is a common problem in image enhancement that it is not possible to always find automatically a satisfactory parameter adjustment for separating the text from the background. The separation method works by looking at each single pixel, one after the other, and to decide for it individually whether dyeing it black or white.

The dialogue shown in Fig. 4.7 comprises two parameters: **Local area of influence** determines how local the decisions are in order to dye a pixel black or white; the left side of the slider corresponds to lower values and the lower the chosen value the fewer pixels around the pixel under consideration are taken into account. Conversely, the more the slider on the right the larger the area of influence and the more pixel values are taken into account for deciding for a single pixel whether to dye it black or white.

The other parameter is **Deviation from average grey value** and states how far the pixel under consideration is allowed to differ from the mean grey value in order to dye the pixel black. This second parameter is influenced by the first one insofar the first parameter determines the area of influence in order to compute an average grey value.

Concerning the first parameter **Local area of influence** one has to be careful not to choose a value which is too large. Otherwise it might happen that text near the border of the document image could be cut off from the image, that is those areas would be completely assigned to the background and text will disappear.

As Diptychon is able to load images which have been preprocessed with

other programmes, the user can take another such programme she is familiar with and can even delete noise from the image by appropriate rubber tools.

4.6 Text-background separation on the fly

While transcribing the text the user might detect parts of glyphs which are not separated from the background very well. Such parts can be corrected on the fly.

Using the toggle button it is possible to switch back and forth between the original grey level image and the black & white image. The latter shows whether it might be possible to improve the text-background separation, while the former shows the current status of the separation. As far as structures can be seen on the original grey level image, there is a chance that the black and white image can be improved for that local image area in order to reflect those structures.

Displaying the grey level image and deactivating the 'L'-button, a rectangular area can be chosen. From the context-menu the black-and-white option starts the binarisation dialogue, which is only applied to the rectangular area (Fig. 4.8). When applying the changes and saving the project, the system will save such local binarisation corrections in the '*.dsf' project file.

4.7 Removing noise on the fly

As for binarisation corrections, a rectangular area can be defined for any area with noise. Choosing the binarisation dialogue, the two parameters can be changed until the noise is removed.

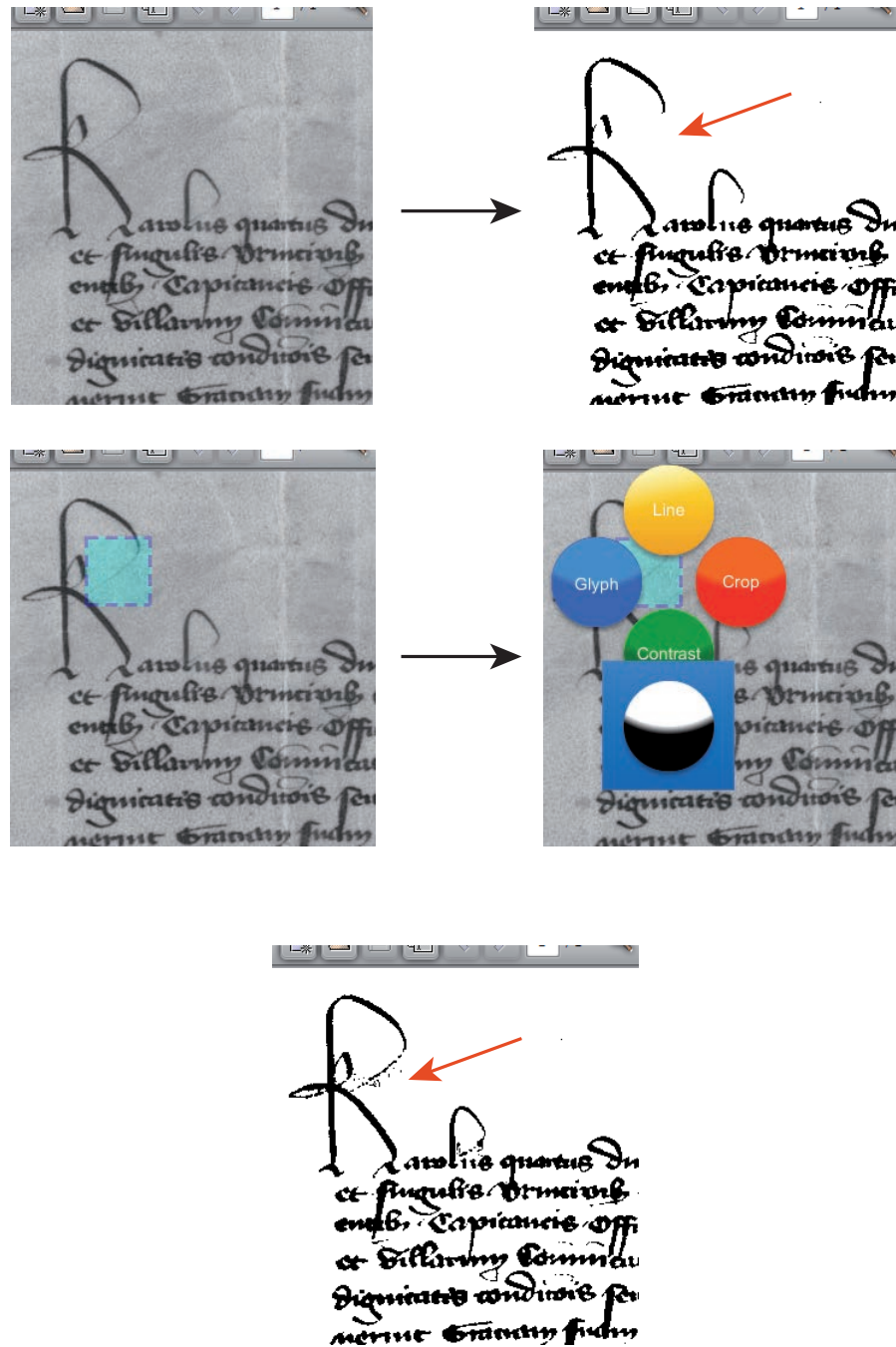


Figure 4.8: For a rectangular area the text-background separation can be adjusted independently of the rest of the image.

Chapter 5

Layout analysis

Input fields are required before starting the transcription. This chapter describes how it is possible to let the system automatically search for text lines and how the user can adjust them if they are not correctly identified. Additionally, it is possible that the user can herself mark regions to define new text lines or to delete them if they are poorly detected. Instead of letting the system automatically detect all text lines, the user might just be interested in transcribing some parts of a large document page.

5.1 Text line detection

The system tries to detect text lines in the document image and automatically places according input fields on the transcription panel on the right-hand side. In this way, there is a one-to-one correspondence between document image and transcription panel. Additionally, the input fields on the right hand side are approximately placed at the same position like the corresponding text lines in the image.

To let the system detect the text lines, one has to open the context menu on the document image with the right mouse button in order to select the method "Find text lines", which is the yellow circle at the top of the context menu:



As a result of applying this method, the input fields for all of the detected text lines are shown on the right-hand side panel, as shown in Figure 5.1. As the input fields are distributed in accordance to the text lines in the document image, they are not regularly arranged on the right hand side panel. This is the reason why some text lines start earlier on the left-hand side than others and why their vertical distance to each other differs. While the user needs to get used to it, she will sooner or later take advantage of it, namely when looking for specific correspondences between both panels.

When zooming out of the document image, all detected text lines can be seen at a glance (Figure 5.2).

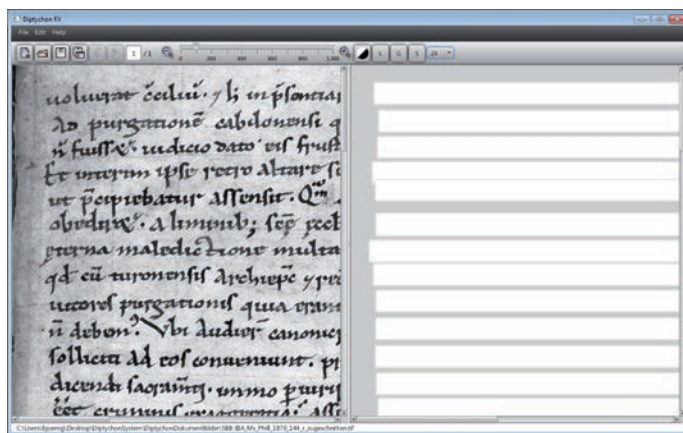


Figure 5.1: To each of the detected text lines a separate input field is assigned.

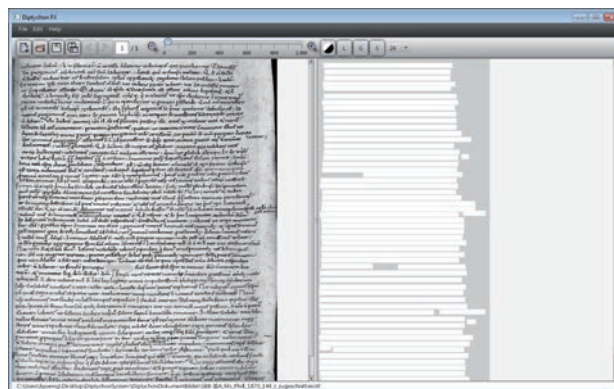


Figure 5.2: When zooming out, all detected text lines can be seen at a glance.

5.2 Text line corrections

In Figure 5.3 the purple rectangle visualises how accurate the text line has been detected.

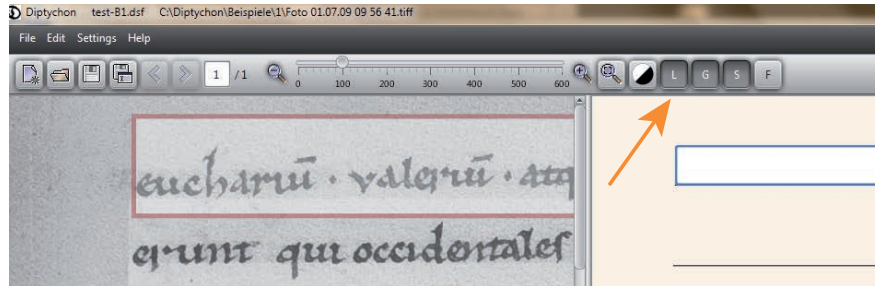


Figure 5.3: In order to turn the purple rectangle on, the toggle button with an **L** on it needs to be selected (pressed down).

The rectangles can be changed horizontally as well as vertically in order to enclose the text lines more accurately. As Figure 5.5 shows, when changing the horizontal size of a text line the corresponding input field is changed accordingly.

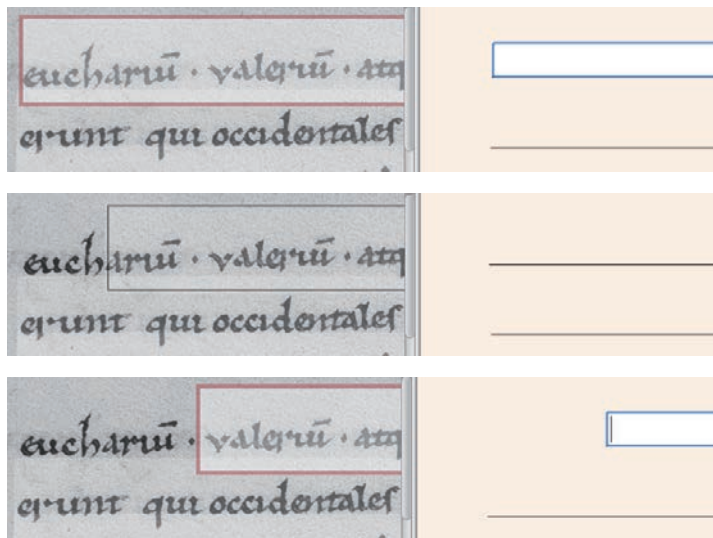


Figure 5.4: Rectangles can be resized horizontally.

The vertical change of text lines (concerning their height) is more often necessary. One of the reasons are ascenders and descenders which are frequently not completely overlapped with the detected text lines. There are other reasons which are discussed next, together with a setting that is to be applied before letting the system detect text lines automatically.

5.3 Adjusting the height

It is sometimes difficult to let the system automatically detect text lines appropriately due to ascenders, descenders, nearby text lines, annotations between text lines, or due to noise in the background. As a consequence, the height is frequently underestimated by the underlying algorithms.

There is a way to prevent the system to detect text lines which are too narrow. The user can let the system know how much it should deliberately overestimate both the top boundary of a text line and its bottom boundary. For this purpose, `Ctrl.-L` (`Strg-L`) or the menu item `Settings/Line Detection...` opens a dialogue window. The top slider and the bottom slider in Figure 5.5 determine the percentaged amount how the automatically detected height are to be extended towards the top and bottom, respectively.

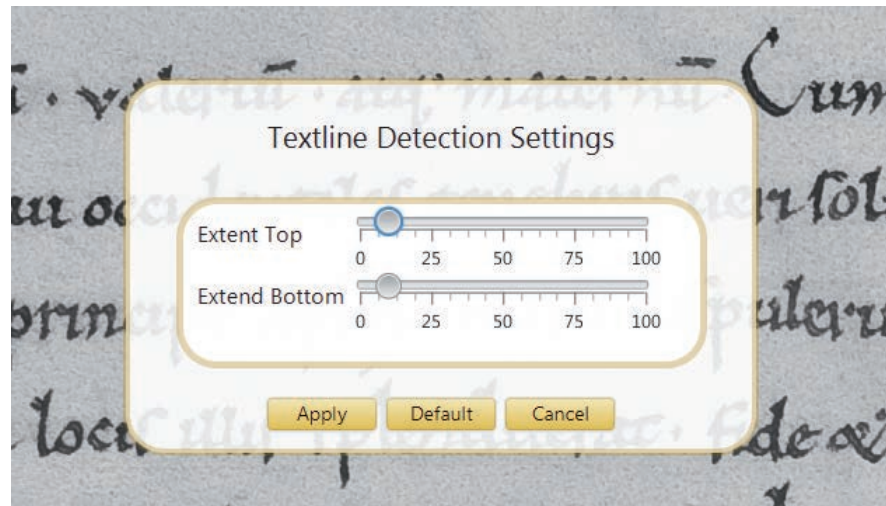


Figure 5.5: The height of a text line can be adjusted.

This automatic adjustment prevents the user from adjusting each rectangle by hand, as it is described in section 5.2. At least, fewer adjustments will be necessary.

Part III

Character segmentation and Transcription

Chapter 6

The transcription process

The main purpose of Diptychon is the extraction of each single glyph in a document image and its assignment to the correct character class. For this purpose, a workflow is realised by Diptychon which the user has to follow. As a prerequisite the lines of text must have been detected as described in chapter 5.

For each separate text line the workflow includes two main steps: In the first one, the system suggests the user how it would separate the glyphs in that line and the user has to edit that suggestion. The second step consists in entering the text of that line which the system will automatically assign to the glyphs. This assignment is determined by the linear order of the glyphs as they occur in a text line, which corresponds to the very same linear order of characters in the according text field.

Both steps can be applied in an arbitrary order. This depends on the user's intentions and background knowledge of the given writing. Sometimes it might be relevant to analyse the complex glyph images first before being able to transcribe those glyphs. In other cases, the transcription is already available and the user is only interested in separating the glyphs according to the known text.

6.1 Transcribing a text line

The text is to be transcribed line by line. In principal, the user can start with any of the text lines. The order of lines, which the user is going to transcribe, can be chosen as she wishes. The user can choose whether she will first enter the text of a line into the according text field or whether she first wishes to separate the glyphs in the document image. The former mode makes sense when a transcription is either already available or the user has no difficulties to read the text. In this case Diptychon will mainly be applied to separate all glyphs. The latter mode will be of interest, if the user first of all has to analyse the text carefully by separating all glyph images before being able to recognise the underlying characters. In this case, Diptychon helps the user to tell apart the glyphs. Both modes of transcription will be explained in turn.

6.2 Entering the transcription first

In order to transcribe a line, the text can be written into the correct text field, which is highlighted when the according line in the document image is selected by the left mouse button (Figure 6.1). The text field can be used like a usual text processing system with all editing functions. In particular, it is possible to copy a line of text via the windows clipboard into the text field, if there is already a transcription available for this document. But then, one should be careful with control characters which might be used in a windows application.

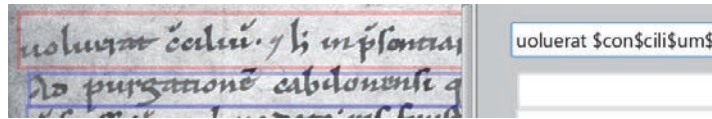


Figure 6.1: The selected line is highlighted by a bright red rectangle in the programme. The text field on the right has a turquoise border.

After the text has been typed or copied into the text field, the **Enter** button enables the user to continue the workflow with the next step. Diptychon analyses the text line in the image with the aid of the text which the user wrote or copied into the text field. This analysis may last a few seconds (on a slow machine or when other programmes are running concurrently in the background, this step might even last one or two minutes).

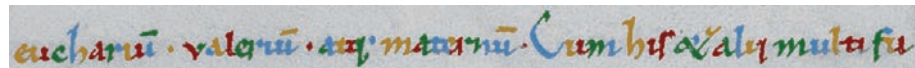
Diptychon makes use of the number of characters and the lengths of the words in order to make a suggestion for how the glyphs separate. In most cases that suggestion needs to be edited by the user in order to adjust the glyph separations. The glyph images can be conceived of as *fragments* the user either has to further separate or melt together. The next section describes what is called *fragment editing* within Diptychon, that enables the user to revise Diptychon's suggestions.

6.3 Fragment editing

Provided that the user enters the following transcription:

euchari\$um\$, valeri\$um\$ at\$que\$ matern\$um\$. Cum his \$et\$ alii multi fu

After having pressed the **Enter** button, the following suggestion with 63 fragments is provided:



Going from the left to the right, it is shown that a mix of four colours is used for visualising the fragments: orange, red, green, and blue.

These fragments are determined by the following algorithm:

The number of characters is taken, excluding the space characters; additionally, everything between the \$-signs counts as one single sign, including the \$-signs themselves¹. This amounts to the number of 45 characters for the given example.

As the space characters show where one word ends and the next one starts, it is possible to determine the number of characters for each word. By means of the number of characters in each word and under the assumption that, in general, the space characters are longer between two words than between characters within a word, it can roughly be estimated how long a single glyph in a word is.

This length is somewhat reduced in order to obtain a fragmentation with too many fragments. The idea behind this approach is that it is easier for the user to melt together fragments than to cut a single fragment into a number of different pieces with appropriate borders.

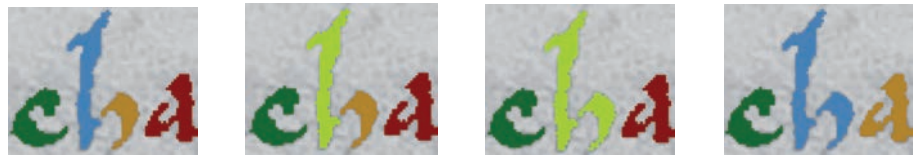
6.3.1 Connecting fragments

Provided that two fragments are to be connected to form a single piece. For this purpose, one of the two fragments is to be chosen by pressing and holding the left mouse button. As a consequence, the colour of that fragment changes to bright green. Keeping the left mouse button pressed, the mouse cursor can be moved over other fragments which will all be added to the former fragment. This is visualised by inking all fragments bright green, which have been touched with the mouse. As soon as all fragments which are to be connected are coloured bright green, the left mouse button can be released.

In the end, every fragment which has been touched by the mouse will be connected to all other touched fragments. To visualise their conjunction as a single fragment, they get the same colour. This also requires to change the colours of all successive fragments, so that again all neighbouring fragments are differently coloured, again in the order orange, red, green, and blue.

¹For the \$-signs and their employment for dealing with abbreviations see section 6.5.

For instance, the blue and orange fragments in the example below are to be connected. Then, one of both fragments needs to be selected and the mouse needs to be moved over the other fragment while the left mouse button is pressed. Releasing the mouse button, the right most image shows how Diptychon redistributes the colours, the 'h' being coloured blue in the end, while the following 'a' changes to orange. This example simultaneously shows how even disconnected fragments can be joined to a singly fragment, since the letter 'h' in this glyph-example consists of two disconnected components.



6.3.2 Cutting fragments into pieces

The left most image below shows a mistaken fragmentation for the glyph images of 'eu'. Selecting the orange fragment by the left mouse button, which is to be released again immediately after it has been pressed, the fragmentation in the second image is obtained, that correctly separates the 'e' from the 'u'. Though both the 'e' and the 'u' are now themselves fragmented into two pieces. But as the following images show, these fragments can easily be connected as described in the previous section. The last image eventually shows the correct separation between both characters.



This algorithm for separating a single fragment into sub-fragments usually finds an appropriate border where to break down a fragment into two or more fragments. It looks for narrow sections within the fragments under the assumption that they frequently indicate borders between adjacent glyphs.

There are two parameters which can be changed in order to adjust the behaviour of this method.

- *Minimum fragment size*: it will be guaranteed that the resulting fragments will be greater or equal to this size;
- *Fragmentation strength*: a factor that is multiplied with the number of characters in a word or in a line of text. That product indicates the number of fragments the algorithm will initially try to create. The larger this number the higher the degree of over-segmentation.

6.3.3 Fragmenting pieces by a rectangular tessellation

There are several cases in which the former fragmentation method fails:

- there are more narrow sections which are directly adjacent to each other and the algorithm could choose the wrong one;
- two glyphs meet at a border which is not particularly narrow;
- the border has been detected almost perfectly, however parts of the border should be chosen somewhat differently.

In all those cases another fragmentation method can be employed that tessellates a region into small squares. The length of the side of a single square can be adjusted by a parameter called the *burst size*. As fragments do hardly form themselves rectangular areas or even squares, the border areas of fragments will in general be filled with different shapes, which are the remaining parts of those squares that are too large in order to completely fit into the border areas. Figure 6.2 shows an example and Figure 6.3 a blow-up of the critical section.

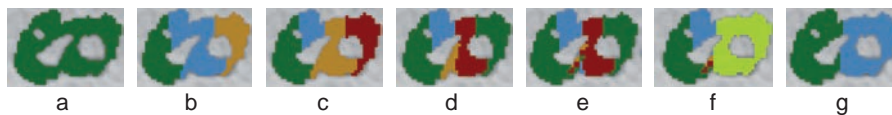


Figure 6.2: Different fragmentation steps: (a) a single fragment, (b) after having clicked with the left mouse button into the green area, (c) after having clicked into the blue area, (d) after having clicked into the orange area, (e) the orange part has been tessellated into small squares by choosing 'Burst Fragment' from the context menu, (f) all fragments belonging to the letter 'o' have been selected, (g) the remaining fragments have been connected to form the letter 'e'.

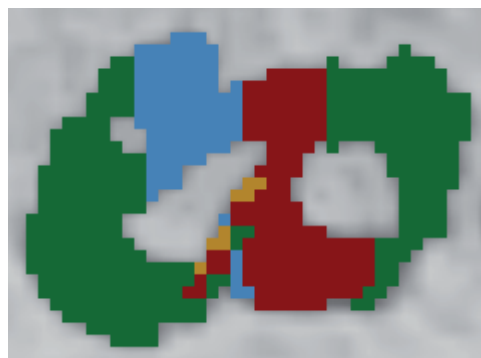


Figure 6.3: The enlargement of image (e) from Figure 6.2.

It should be noted that it might happen that adjacent fragments could get the same colour when partitioning a fragment into many small squares. Later when having connected fragments to form the correct glyphs the redistribution of colours will again guarantee that adjacent fragments get different colours.

6.3.4 Fragmenting pieces by line segmentation

There are a number of cases which are particularly challenging, for example, when glyphs are very close by each other, so that there is no clear transition between adjacent glyphs, when ligatures are to be separated into their constituent parts, or when there are imperfections, such as blobs or holes in the paper, smearing the boundaries of the glyphs. Yet another common problem are nearby text lines. In this case, descenders are overlapping with ascenders of the next text line. The previously described methods are generally not very successful in such cases.

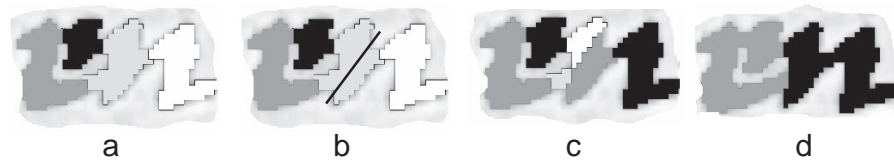


Figure 6.4: Corrections by the line segmentation method.

The line segmentation method allows the user to draw an arbitrary line along which a region is supposed to be separated into two pieces. In this way, even oblique transitions between glyphs can be dealt with. The line might even cross a couple of adjacent regions all at the same time. As a consequence, they are simultaneously divided into twice as many pieces as there have been regions before. See Fig. 6.4 for an example.

A segmentation line can be drawn wherever desired in order to divide regions. The sole exception is that such a line needs to start in the background and that it ends somewhere else in the background. This is to ensure that such a line does not end up in the midst of a region; in this case, not all boundaries of the intended regions would have been clearly determined. Provided that the user does not care about this restriction the triggered method will simply do nothing.

The line segmentation method could in principal be always applied, but needs some effort and carefulness by the user. By contrast, the simple segmentation method described in section 6.3.2 is much easier to employ, however, fails at some complex regions. But just in such a case, the employment of the line segmentation method resolves what the automatic algorithm is not able to handle.

6.3.5 Reconciling the number of fragments and characters

The number of fragments and the number of characters have to be equal. Diptychon permanently monitors the number of fragments while the user is editing them. It increments and decrements that number while connecting and separating fragments, respectively. Diptychon compares the number of fragments with the number of characters in the corresponding input field.

The user does not need to take care of those numbers and that they have to be equal. Instead, Diptychon provides a modus during which the user edits fragments. This modus cannot be left by the user until Diptychon counts the same number of glyphs and characters or until the user decided to discard the whole line by selecting the according button. Figure 6.5 explains this modus.



Figure 6.5: The current number of glyphs (52 in this example) and the number of characters (53) are permanently updated and shown to the user. This modus can abruptly be left by selecting *Discard*. Then, however the whole text line would be removed and the user would have to start from the beginning regarding this line. As soon as both numbers are equal the apply button can be selected, finalising that dialogue, that is leaving the fragment editing modus.

6.3.6 Overlapping lines

Frequently adjacent lines of text will overlap. Descenders of the previous row as well as ascenders of the next row might overlap with the row which is currently edited. Diptychon provides a general way for dealing with overlapping lines of text.

When determining a text line (as described in section 5.1 on page 21), the user has to be careful to include all ascenders and descenders of the current

line, irrespectively of any overlaps with the previous or next line. After having entered the fragmentation mode, the user simply has to remove those fragments by 'Remove Fragment' of the context menu, which are part of glyphs of the previous or next line.

In order to better perceive those fragments, the user should deactivate the G button of the button-row right of the magnifying bar. As a consequence, all glyph images of all rows but the current one will be faded out; only the glyphs of the current row keep visible.

6.4 Separating the glyphs first

The user might have difficulties in recognising the characters of a text line. In this case, she should initially analyse the handwriting by separating all glyph images. The context menu item 'Fragmentate' will generate a suggestion for how to fragment the glyph images. The user can edit that suggestion as described in the previous section.

After 'fragment editing', the characters of the glyphs can be typed into the corresponding input field. As soon as the number of characters and glyphs are equal, the user can proceed by pressing 'Apply', finalising the line transcription.

Each line, which has previously been transcribed can be re-entered and again edited by choosing the context menu item 'Fragmentate'. In the case that this text line was finished before by 'Apply', the current glyph separation is adopted. This allows to improve a text line without starting from the beginning. One can even trick the system by changing the text in the input field arbitrarily so that 'Apply' can be chosen, in order to continue later editing that line. This might be useful if dealing with a long and complex line of text.

6.5 Managing abbreviations

Abbreviations can be dealt with by using the \$-sign. As soon as an abbreviation needs to be dissolved, the complete string of characters is to be enclosed by two \$-signs, for instance, \check{c} translates to $\$pre\$$. An example with two abbreviations in a word is:

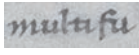
$$\check{c}cili\ddot{u} \rightarrow \$con\$cili\$um\$$$

which translates to

$$c(on)ciliu(m)$$

In principal, any characters or signs which are not seen in the original document image can be added to an existent character in order to include that sign to the transcription. An example are commas or separators which have not been written by the scribe, but which the editor wishes to include into the transcription.

An example for a missing separator is given for the word *fu*erunt for which the first two characters, *fu*, are written at the end of a row, however without the separator 'u'

... 

while the rest of that word, *erunt*, is written at the beginning of the next row. Employing the \$-sign it is possible to assign 'u-' to the glyph 'u' in the example:

fu- → f\$u-\$

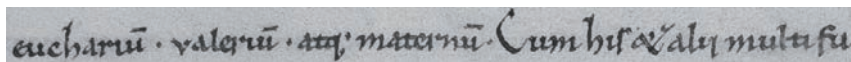
which translates to

fu(-)

While it is possible to include any characters into the transcription for which the scribe did not write any glyph, this brings in a disadvantage. In the example the glyph image for the 'u' is assigned the string 'u-'. As a consequence, whenever Diptychon finds the glyph image for 'u' it will accidentally assume that 'u-' is a valid transcription for such a glyph image. As there will be other occurrences of that glyph Diptychon will more precisely *learn* that such a glyph image might translate either to 'u-' or simply to 'u'. But Diptychon has no rule which transcription is correct for a given context. Consequently, the editor should either omit this feature for unwritten glyphs or should manage them with carefulness.

6.6 The word separation mode

Sometimes a better fragmentation suggestion can be made by trying to detect space characters between words. As a consequence, the number of characters can be assigned to each word and Diptychon can create other fragments than when ignoring space characters. This mode is entered via the menu *Settings/Fragmentation/Manual Space Extraction*. An example is given as follows. The text line:

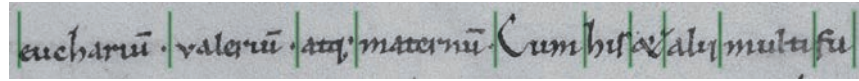


is to be transcribed to:

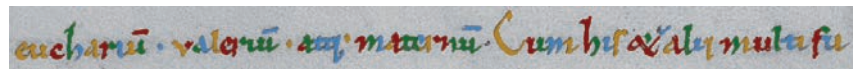


After a text line has been completely transcribed it is necessary to press enter in order to let the system know that the text line is completed. Then, Diptychon is aware of the number of characters and positions of space characters.

It makes a suggestion regarding where words do end and start. This is indicated by green vertical bars, which the user can adjust to their correct positions:



When finished the known fragment editing mode is entered in order to correct the resulting fragment distribution:



The final steps are as usual (see section 6.3 on page 28).

6.7 Changing character assignments

Provided that the user suddenly recognises that she accidentally wrote a wrong character into the text field on the right hand side, that character can be changed by entering the text field. After having edited the contents of the text field, the user has to leave it with the **Enter** button, so that Diptychon gets the information that it has to change the glyph-character assignments.

The user should be careful not to accidentally delete a character. In that case, there would be more glyphs than characters. As a consequence Diptychon would restart the fragmentation modus. In this case, however the user knows that she has to carefully check whether there are both enough glyph images and enough characters. The fragmentation modus can be finished as usual.

If accidentally entering the fragmentation modus in this way, it will often happen that there are too many fragments. The reason will be that there are overlapping components of the previous line or the next one. Those components will be taken into account by Diptychon, as the system itself does not know whether those components are relevant or not. The user has to remove them by means of 'remove glyph' in the context menu.

6.8 The automatic transcription process

After having transcribed at least one line of text, Diptychon is aware of a couple of different glyph images and their corresponding characters. By means of the context menu the user can check how well Diptychon is able to automatically identify the glyphs which have already been described. As this requires a huge amount of computation, an hourglass will be placed onto the top right of the Diptychon screen. In this way, the user gets a rough idea about the progress of the automatic transcription mode.

The contents of the text-line can be changed later by means of the usual text editing functions of text processing systems. But after each change the

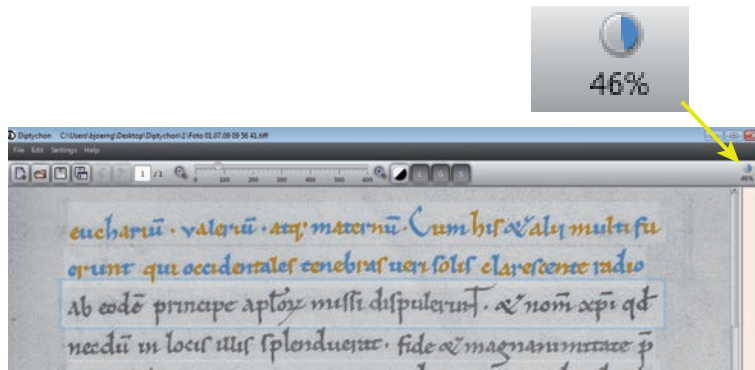


Figure 6.6: During the automatic transcription process an hourglass shows the progress on the top right of the Diptychon screen.

enter button needs to be pressed. Otherwise the system will not be aware of the applied changes. This will launch the fragment-editing modus, however all fragments which the user already edited before, will be taken.

6.9 Correcting glyph-character assignments

It is also possible to carry out corrections without entering the fragment-editing modus. It might be sufficient to launch the glyph editor in order to apply corrections to a single glyph:



In other cases possible it might become necessary to swap the correspon-

dences between two characters and their glyph assignments. For example, this happens when a glyph has a long horizontal part that overlaps with adjacent glyphs. In such cases, the automatic glyph-character assignment might be wrong due to the ambiguous order of glyph-positions.



An example is the word `a$posto$lorum`: because the letter 'l' extends towards the right on top of the letter 'o'. Another reason is, that the user, being in the fragment-editing modus, accidentally made a mistake while typing in the transcription quickly. Provided that this causes the assignments of adjacent glyphs to be interchanged in the transcription, these assignments can be interactively corrected. In order to exchange two adjacent assignments the following menu is to be used:



If the 'l' is at the location of the 'o' in the transcription, 'Move left' is to be chosen. As a result the order of letters in the transcription will change accordingly. An arbitrary number of letters can be interchanged in a text-line in this way. Whenever dealing with text-lines which are close by each other or when having annotations being in between text-lines, such problems might occur more often.

Part IV

Working with transcriptions

Chapter 7

Comparison methods

Diptychon supports the analysis of the writing style in a document image by means of a number of methods.

7.1 Find glyph images

In order to compare the different glyph instances of a character it is possible to find successively all instances. With **Ctrl-f** a dialogue box pops up. Since this feature aims at the search for specific glyphs in the document image panel it is suggested to enlarge that panel first, as shown in Figure 4.4 on page 16.

In order to find glyph images for a specific character that character needs to be written into the input field, as shown in Figure 7.1 for the letter *b*. All transcribed glyph images will be hidden in order to enable a better focus towards the glyphs which the user intends to look for.



Figure 7.1: Searching for glyph images.

With the **Next** and **Previous** buttons it is possible to iterate step by step forward and backward from one instance to the next one. Selecting **Close** ends the search dialogue and transcribed glyph images are shown again.

The option can be selected, whether one is interested in lower case and capital letters. If the option box is not selected both cases will be found. Additionally, it is shown how many instances Diptychon found, each one being enumerated.

Instead of single characters it is also possible to search for strings. For example, how often the string 'eu' occurs in the document and how those occurrences look like. The left hand side of Figure 7.3 shows the search for the word 'solis'.

As it might be of interest to analyse how a particular character looks like at the end of a word, it is possible to enter a space character after that particular character. As a consequence, only those occurrences will be found which show that character at the end of a word. The same is possible if being interested in finding just those occurrences, that start a word, though those instances will be left out which occur at the beginning of a row, as the very first character of a line text.

Similarly, space characters can also be included into strings. For instance, one might be interested in learning how a scribe wrote an 'a' at the beginning of a word when the previous word ended by an 'e'. The right hand side of Figure 7.3 shows this example.

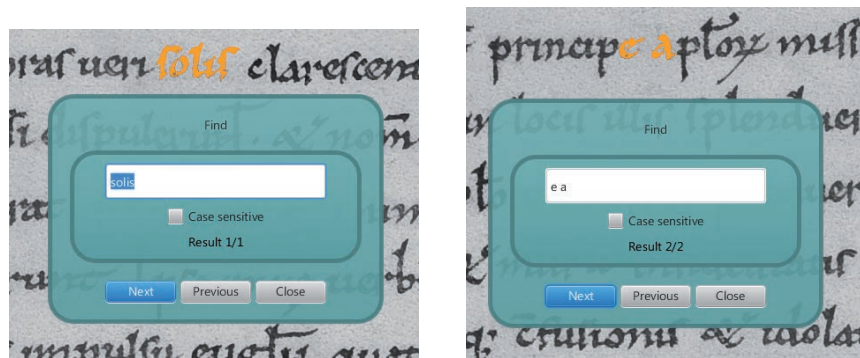


Figure 7.2: Searching for the string 'solis' and for strings that include a space character. This enables the search for pairs of words which end with a specific character and start with another specific one, in this example 'e a'.

If a single character or a string cannot be found, there are no transcriptions for them available yet. Figure 7.4 shows the dialogue box in this case.

7.2 The glyph gallery

The glyph gallery shows for a chosen character class all instances of the document which have already been described. By means of the left mouse button and the right mouse button one of the glyphs can be displayed in the left separate window and the right one, respectively. Their size can be changed with the left mouse button to increase their size and with the right mouse button in order to scale them down.

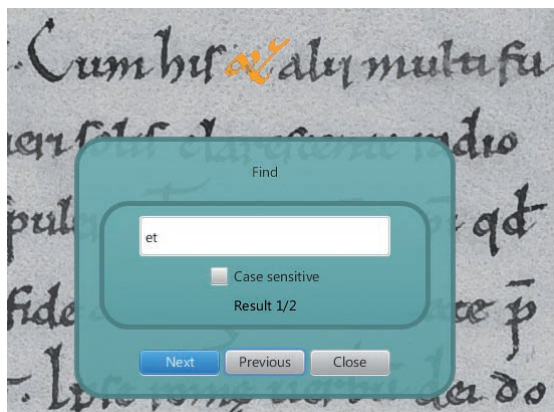


Figure 7.3: Abbreviations which are defined by the \$-sign can also be found.

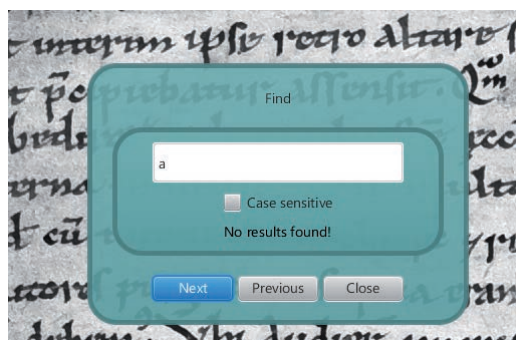


Figure 7.4: Searching for a glyph which has not been found or transcribed yet.

7.3 Search for similar strings

As an example for an application scenario two different texts are given which are to be compared in order to determine the similarity of the scribes. Conventional systems would be confined to extract features of the two documents pages independent of their content. With Diptychon the following strategy becomes possible:

All glyphs can be compared directly with glyphs of the other document, however, of the same character class. As the appearance of glyphs depends on the context in which they appear, it is sometimes crucial to distinguish whether they are found either at the beginning of a word, between other glyphs within a word, or at the end of a word. Therefore, the largest character context is considered in which the glyphs appear. In order to find this largest context, first of all it is examined whether the same words or even phrases can be found in both texts. If this is not possible, the strings are gradually shortened until a match can be found. Then, the glyphs of these words can be directly juxtaposed

in order to compare them.

For their juxtaposition the glyphs need to be exported via the menu

File/export/Glyphs

The user has the option to either extract the binary images of all glyphs or the greyscale images as excerpts of the original document image. A zip-folder is generated with all the hundreds or thousands of tiny images inside. The filenames are equal to the information given through the tool-tips in Diptychon, that is when moving the mouse over the according glyph its filename is shown. It include the line number and column in that line.

This approach has been tested by putting side-by-side equal strings of different documents which have been written presumably by Charles IV (see Fig. 7.5).

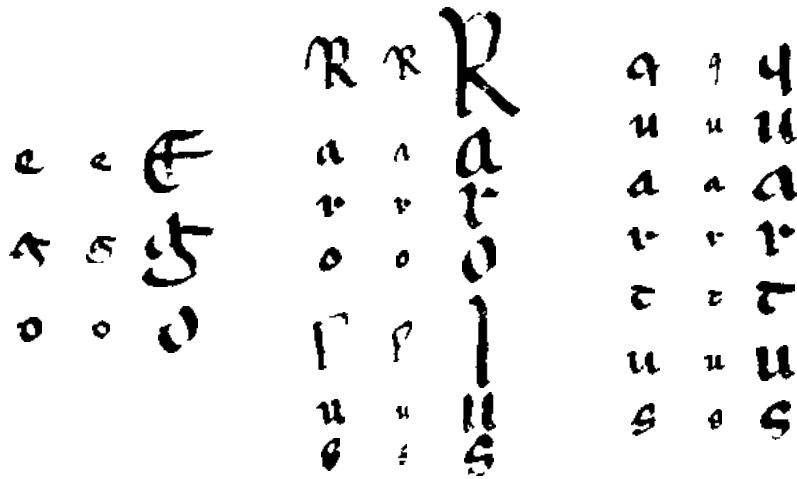


Figure 7.5: The same strings extracted from different documents.

Chapter 8

Generating searchable PDFs

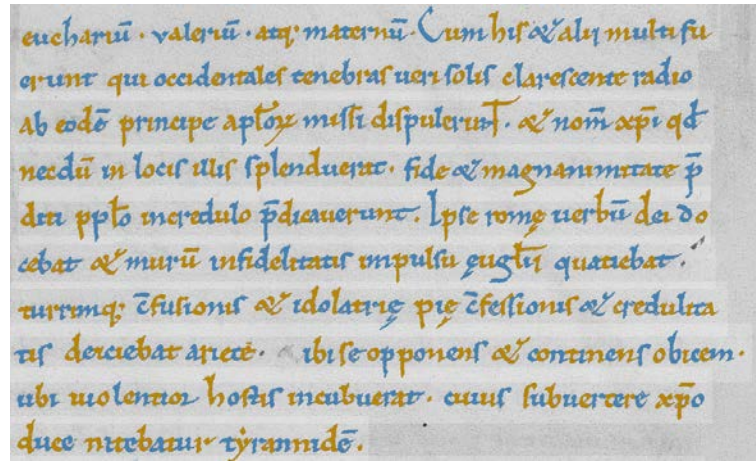
While it is possible to search for strings within Diptychon, there is an export function which generates a searchable PDF-document that can be passed to other users who have not installed the Diptychon system. The PDF document can be used and exchanged like other usual PDF documents, requiring no specific software in order to deal with them. In Diptychon it is generated by the following menu entry:

`File/Export.../Images.../PDF...`

The user has the choice whether putting abbreviations within parentheses or not. Those parentheses are to be taken into account when searching for strings that include abbreviations.

For the recipient of the PDF only the original document image is shown. However, the PDF contains both, the original document image as well as a transparent layer with the transcription itself. For the latter, the location of each single word is related to the document image, and for the user, the impression arises that the original handwriting is searched. This enables the palaeographer to find each string within the original handwriting and to easily compare single letters, whole words, and other arbitrary strings in order to analyse their appearance, for example for scribe comparison.

Fig. 8.1 shows an example. The image on the top shows the document within the Diptychon system, while the image on the bottom is generated when the PDF is made. Any PDF reader resorts to the latter in order to access the text, while the user is just seeing the image.



euchariu(m), valeriu(m) atq(ue) maternu(m). Cum his e(t)alii multi fu
erunt qui occidentales tenebras ueri solis clarescente radio
ab eode(m) principe ap(osto)lorum missi dispuleruNT, e(t) nom(en) C(hrist)i q(uo)d
neccu(m) in locis illis splenduerat, fide e(t) magnanimitate p(re)
diti p(ro)p(ri)o incredulo p(re)dicaverunt. Ipse rom? uerbu(m) dei do
cebat e(t) muru(m) infidelitatis impulsu ?u(an)g(e)lii quatiebat
turrimq(ue) c(on)fusionis e(t) idolatri? pi? c(on)fessionis e(t) credulita
tis deiciebat ariete ibi se opponens e(t) continens obicem,
ubi uiolentior hostis incubuerat, cuius subuercere C(hrist)o
duce nitebatur tyrannide(m).

Figure 8.1: An excerpt of a document page with a handwriting of the 11th century (Philipps. fol. 11v, 1870. Staatsbibliothek zu Berlin, Preußischer Kulturbesitz). The top image shows the original document as a greyscale image with blue and orange dyed glyphs showing the segmentation results. The bottom image shows the location preserved transcription. Questionmarks indicate editors special characters which have not been correctly adopted yet. Braces include long forms of abbreviation signs.

Chapter 9

Export functions

Whenever Diptychon is launched the last state is automatically recovered, as far as one has saved everything before leaving the programm. By means of

`File/Open (recent)`

a `.dsf` file can be chosen that completely saves the state of any project (`dsf` is short for Diptychon saved file). The folder is opened that has been used lately.

A project includes a single image or an image sequence of document pages, the probably enhanced states of those images, and the current state of transcription. With the aid of `.dsf` files different transcriptions of different documents can be processed in parallel or different editions of the same document can be dealt with in parallel, for example by different editors working together or on alternative transcriptions.

With

`File/Save (as...)`

the whole state of Diptychon is saved. When the programm is launched again, one can continue to work wherever the work was interrupted.

Besides the storage of the Diptychon working state, a number of other file types can be chosen in order to export the current state of transcription. The transcript can either be saved in the rich text format:

`File/Export.../Transcript.../Rich Text Format (*.rtf)`

which can be loaded in Word as a formatted text, or as a plain text:

`File/Export.../Transcript.../Plain Text (*.txt)`

which can be loaded in any arbitrary text processing system, including Word or WordPad.

Moreover, the result of the transcription can be formatted and exported according to the TEI standard¹. A generic xml-format is chosen which can be

¹<http://www.tei-c.org/index.xml>

adapted for particular purposes according to the specificities of a genre and the kinds of documents at hand, e.g. charters as opposed to chronicles:

`File/Export.../Transcript.../Text Encoding Initiative (*.xml)`

The preprocessed images can also be saved separately; either the greyscale image or the black and white one:

`File/Export.../Images.../...`

Finally, the extracted glyphs of the document can be exported in single images, either as greyscale images or black and white images:

`File/Export.../Glyphs...`

As a single document page might include thousands of glyphs, the images are stored within a zip-Folder. They are stored in the png-format and have a size of about one kilobyte.

Chapter 10

Statistics

The following menu item exports a text file with some document statistics:

`File/Export.../Statistics`

First of all, a number of layout related data is shown, including the number of textlines, spacing between text lines, widths, heights, the slant of text lines, etc.. Some information about the abbreviations are given and for each character class the number of glyphs pertaining to that class is shown, together with some information about the sizes of glyphs. All these information is given in plain text.

Through the export of glyph images by

`File/Export.../Glyphs...`

a file called **Glyph Properties** is generated that includes for each single glyph a number of extracted features. The glyphs are related to this statistics through their file name found in the first column, for example `P_0_L_1_COL_1` points to the png-file that includes the glyph `P_0` in line number 1 and column 1. The information in this file is also given in plain text.

Chapter 11

An example project

There are two example files:

```
SBB-IIIA_Ms_Phill_1870_043_r_EinAusschnitt.tif
```

and

```
SBB-IIIA_Ms_Phill_1870_043_r_EinAusschnitt.dsf
```

Both files are to be copied into the following folder:

```
C:\Diptychon\Examples
```

After that Diptychon can be launched. By File/open (Ctrl-O) the

```
SBB-IIIA_Ms_Phill_1870_043_r_EinAusschnitt.dsf
```

file needs to be selected. An excerpt of a document with five rows will be opened. Pressing the F-button on the right of the menu bar, all extracted glyphs are depicted alternating in yellow and blue as shown in Figure 11.1.

With this document the Ctrl-f function can be tested. This shows that all occurrences of a particular character can be found, including its occurrence within abbreviations. For example, searching for the letter 's' both occurrences of 's' within the word 'Virdunensis' are shown. However, concerning the third case, what is found is the glyph \bar{p} which is transcribed as 'piscop' that contains an 's'. The same holds for the next case, where a 't' is found, which however, has been transcribed as 'tus'.

The latter example additionally shows a mistake concerning the extracted glyph. The number '9' on the top right of the letter 't' indicates the abbreviation 'us' and should therefore be extracted like the other letters. A similar mistake has been made in the second row with the word 'partiretur' and the abbreviation for 'ur' that is indicated by the number '2' on the top right of the 't'. The third row shows another occurrence of that kind of abbreviation. In this case the glyph-image of the number '2' is assigned to the glyph of the letter 't'. This is how abbreviations currently can be dealt with in Diptychon.

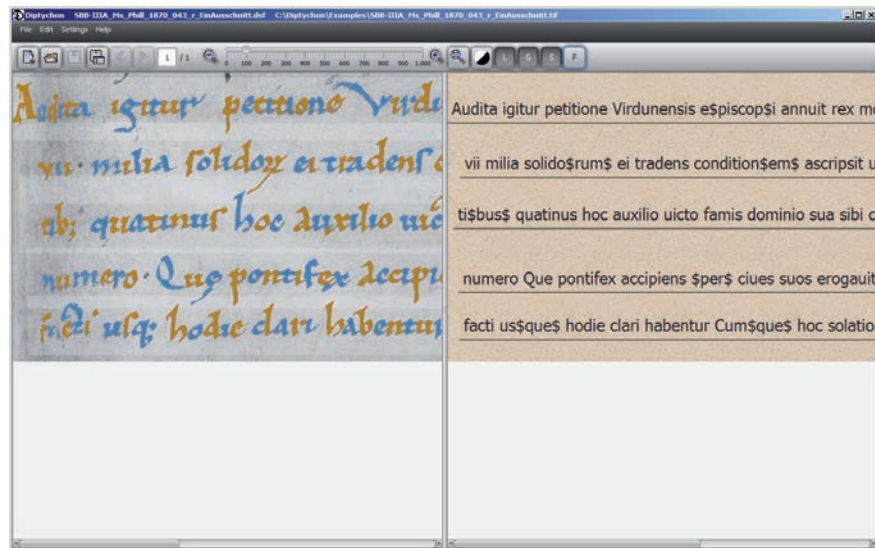



Figure 11.1: An example document.

For the future, other options are planned to be considered by Diptychon in order to separate the glyphs of the 't' and the '2' by simultaneously letting the user indicate their relationship within the transcription. Moreover, other abbreviation modalities are widespread, for instance, that a number of unseen characters are to be put ahead of a specific glyph with an abbreviation sign, instead of appending the string to it.



Another problem is illustrated by the very first glyph: . It is not a simple connected region, but consists of two components. However, with Diptychon disconnected parts can be assigned to the same character in the transcription in order to deal with defected images that cannot be appropriately enhanced. There are other examples for this problem as on the left hand side on the last row. It shows that the word 'facti' is quite destructed.