Robert Koprowski, Zygmunt Wróbel

Image Processing in Optical Coherence Tomography
using Matlab

University of Silesia 2011

Series editor: Dr hab. inż. Porwik
Reviewer: Prof. dr hab. inż. Andrzej Dziech

Published in Poland by University of Silesia,
Institute of Computer Science,
Department of Computer Biomedical Systems

ISBN 978-83-62462-02-5

Cover design, title page, and technical editing:
Robert Koprowski robert.koprowski@us.edu.pl
Zygmunt Wróbel zygmunt.wrobel@us.edu.pl

Work funded by the Ministry of Science and Higher Education
in 2009-2011 – work number N518 427036

CONTENTS

PREFACE

Dear Readers, the book you have in your hands is a summary of research carried out at the Department of Computer Biomedical Systems, Institute of Computer Science, University of Silesia in Katowice in cooperation with the team of Prof. Edward Wylęgała, D.Sc., M.D. This cooperation resulted in the creation of methods for ophthalmologists support in OCT images automated analysis. These methods, like the application developed on their basis, are used during routine examinations carried out in hospital.

The monograph comprises proposals of new and also of known algorithms, modified by authors, for image analysis and processing, presented on the basis of example of Matlab environment with Image Processing tools. The results are not only obtained fully automatically, but also repeatable, providing doctors with quantitative information on the degree of pathology occurring in the patient. In this case the anterior and posterior eye segment is analysed, e.g. the measurement of the filtration angle or individual layers thickness.

To introduce the Readers to subtleties related to the implementation of selected fragments of algorithms, the notation of some of them in the Matlab environment has been given. The presented source code is shown only in the form of example of implementable selected algorithm. In no way we impose here the method of resolution on the Reader and we only provide the confirmation of a possibility of its practical implementation.

The book is addressed both to ophthalmologists willing to expand their knowledge in the field of automated eye measurements and also primarily to IT specialists, Ph.D. students and students involved in the development of applications designed for automation of measurements for the needs of medicine.

The authors thank cordially Prof. Edward Wylęgała, D.Sc., M.D. and his team for the provided images and valuable guidance and consultations.

1 INTRODUCTION

An optical tomography is a modern, non-invasive technique for a tissue section imaging, in this case of anterior and posterior eye segment, using the light scattered on individual layers of the examined tissue. The spectral tomography, as compared with the hitherto solutions (e.g. time tomography), features much higher resolution. The elimination of a moving mirror, necessary to scan deep into the object examined, allows also shortening the examination time (object scanning) approx. a hundred times. A short time of scan performance as well as its sequentiality and maintaining a constant shift allows obtaining 3D images [7]. Many instruments available now allow such imaging. Instruments, used to acquire images used in this book, have been selected from them, i.e.:

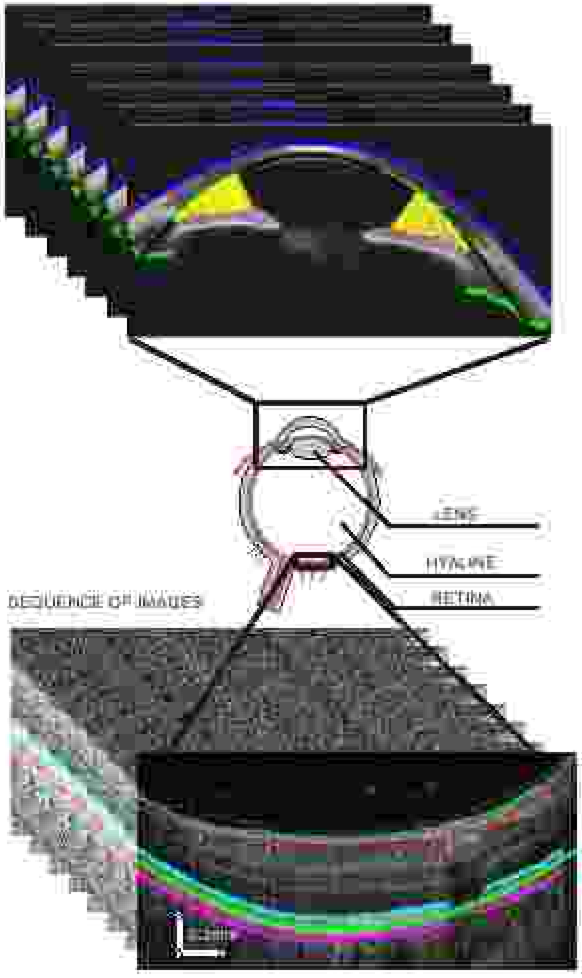
- SOCT Copernicus HR,
- Zeiss Cirrus HD-OCT,
- Zeiss Stratus OCT Zeiss Visante OCT.

Overall, the algorithms presented below have been tested on a group of more than 100,000 images of patients, both healthy and with a significant degree of eye pathology.

The most interesting fragments of algorithms presented have been recorded in the Matlab environment, version 7.2.0.232 (R2006a) and Image Processing Toolbox, version 5.2 (R2006a). Individual fragments of algorithms are separated only with text, comments and after integration create a whole allowing a full image analysis. Despite that, the authors assume that the Reader is familiar with basic functions and possibilities of the Matlab software, with special emphasis on the operations carried out on matrices. If it is not the case, the authors refer Readers to familiarise themselves with Matlab basics, e.g. references [41].

In terms of the imaged object, the description of OCT images analysis and processing methods has been divided into two parts: the anterior eye segment has been presented in the first part, while the posterior eye segment has been presented in the second part of this monograph, in accordance with Fig. 1-1.

PART I



PART II

Fig. 1-1 Cross-section of the front and back of the eye with a marked characteristic of the location areas

2 ACQUISITION OF IMAGE DATA

Difficulties with reading and appropriate interpretation of data recorded for individual patients in OCT equipment result primarily from manufacturers' fears of developing own competitive software. Frequently the information is a company secret. Fortunately the OCT equipment software produced nowadays more and more often records the data acquired in a DICOM or similar format. The Optopol OCT is an exception here, recording the acquisition data in one compressed file.

On the other hand DICOM images may be read in Matlab using the dicomread function available in the Image Processing package. Unfortunately, the usability of this function, in version 5.2 of the Image Processing package possessed now by the authors to read DICOM images originating from reputed OCT manufacturers, is small. Missing header tags and frequently a specific record of the image (JPEG2000) are the reason for which the reading of files is difficult. Such files cannot be read also by majority of freeware available in the Internet and designed for viewing typical DICOM images.

Let us look at the header read from the track

```
path_name='D:/source/1.DICOM'
```

as follows:

```
fid = fopen(path_name, 'r');
dataa = fread(fid,'uint8');
fclose(fid);
```

then we obtain the result directly from OCT Carl Zeiss Meditec file for example for the first thousand of characters

```
char(dataa(1:1000)')
```

we obtain the result:

```
ans =

DIC C ORIGINA  UI  1.2.826.0.1.3680043.2.139.3.1.1 UI:
1.2.826.0.1.3680043.2.139.3.1.1001.1017.20070828114546359
D 2007082 1 D 2007082 " D 2007082 0 TM11435 1 TM11452 2
TM11452 P SH  p LO Carl Zeiss Meditec Inc.  LO4 Uniwersytet
Slaski w Katowicac SH  OLO  >LO pPN
```

```
 test^test^^
 LO
```

```
1000 PN Koprowski^Robert^^^ L KOWALSKI | LO  0 D 20060115M
CS E   @LT
LO
```

```
1C54  L 1.2.0.1 SL Chamber
 UI:
1.2.826.0.1.3680043.2.139.3.1.1001.1017.20070928114359062
UI:
1.2.826.0.1.3680043.2.139.3.1.1001.1017.20070928114546312
IS 0   IS 0   IS0   A UI:
1.2.826.0.1.3680043.2.139.3.1.1001.1017.20070928114546312
  CS OU   @LT  | US  |
```

```
CS MONOCHROME2 | US  | US
```

```
|  US  | US  | US  |                              US  @ SQ
SH
 ALL_SCANS SH 3BCZM
```

The information available in the Internet specify clearly the place, where the data is located, e.g.

- 0010,0010 PatientName N
- 0020,0013 InstanceNumber N
- 0002,0010 TransferSyntaxUID N
- 0028,0100 BitsAllocated N
- 0028,0111 LargestImagePixelValueInPlane N

This means that patient's Name and Surname given in hexadecimal notation in appropriate sequence starting from LSB and ending at MSB will be preceded with values read from the file, i.e.: `16 0 16 0`. In the example file presented these are the values comprised by elements from 480 to 506 range, i.e.: `char(datas(480:506)')`, `datas(480:506)'` as a result we obtain:

```
ans =
 PN Koprowski^Robert

ans =

  Columns 1 through 23

     16      0     16      0     80     78     20      0     75
111    112    114    111    118    115    107    105     94     82
111     98    101    114
```

```
  Columns 24 through 27

   116     94     94     94
```

The reading of the remaining information comes down only to finding appropriate tag and then the record content. The skeleton of example function `OCT_head_read`, returning the information on the header `header_dicom` and the matrix `Lx` of image, designed to read the data originating from OCT Visante, are presented below:

```
function [header_dicom,Lx]=OCT_head_read(dataa)
flagi=zeros([1 100]);
iu=1;
header_dicom=[];
for  i=1:30000
     te=dataa(i:(i+3));
     if  (sum((te'==[16 0 16 0]))==4)&(flagi(1)==0)
          Patinet_Name=char(dataa(i+8:(i+8+dataa(i+6)-1))');
          header_dicom.Patinet_Name(iu)=Patinet_Name;
           flagi(1)=1;
     end
     if  (sum((te'==[32 0 19 0]))==4)&(flagi(2)==0)
                Instance_Number=char(dataa(i+8));
          header_dicom.Instance_Number(iu)=Instance_Number;
           flagi(2)=1;
     end
     if  (sum((te'==[2 0 16 0]))==4)&(flagi(3)==0)
          UID=dataa(i:i+30);
          header_dicom.UID(iu)=char(UID)';
           flagi(3)=1;
     end
     if  (sum((te'==[40 0 0 1]))==4)&(flagi(4)==0)
          Bp=dataa(i+8);
          header_dicom.bits_per_pixel(iu)=dataa(i+8);
           flagi(4)=1;
     end
     if  (sum((te'==[40 0 17 0]))==4)&(flagi(5)==0)
          M=dataa(i+9)*256+dataa(i+8);
          header_dicom.Mlumn(iu)=M;
          flagi(5)=1;
     end
     if  (sum((te'==[224 127 0 0]))==4)&(flagi(6)==0)

header_dicom.length_pixel_data(iu)=dataa(i+10)*256*256+data
a(i+9)*256+dataa(i+8) ;
           flagi(6)=1;
     end
     if  (sum((te'==[40 0 0 1]))==4)&(flagi(7)==0)
```