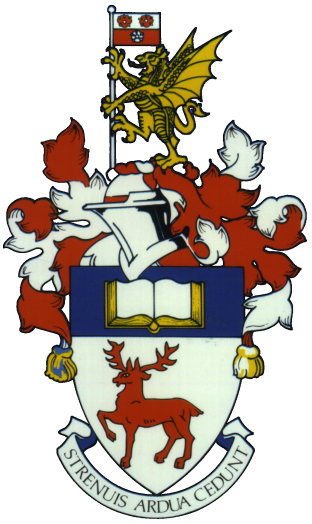
UNIVERSITY OF SOUTHAMPTON



Modular Data Management and Image Processing Approaches to Facilitate Medical Research

by

Lasse Wollatz

A thesis submitted for the degree of
Doctor of Philosophy

in the
Faculty of Engineering and Physical Sciences
Computational Engineering and Design

April 2019

Southampton, United Kingdom

doi:10.5258/SOTON/T0016

UNIVERSITY OF SOUTHAMPTON
FACULTY OF ENGINEERING AND PHYSICAL SCIENCES
AERONAUTICS, ASTRONAUTICS AND COMPUTATIONAL ENGINEERING

Doctor of Philosophy

MODULAR DATA MANAGEMENT AND IMAGE PROCESSING APPROACHES TO FACILITATE MEDICAL RESEARCH

by Lasse Wollatz

ABSTRACT

Digitalisation of medical research and diagnosis is becoming more important because it allows us to offer the best possible treatment to patients exploiting technological advances. New medical imaging methods and digital technologies, such as microfocus X-ray computed tomography (µCT) scanning, are opening up a new world for medical researchers. Due to continuous improvements in imaging techniques, the produced images have grown in size in proportion to improvements in computation. This means that, in order to use cutting-edge imaging techniques, equivalent cutting-edge computing hardware and software are required. Such hardware is expensive and the required software is not always available or incomplete with regards to features needed. This thesis will address the problem of extra-large images in three of the main areas of image-based research: management, processing and visualisation of images. The problems involved with each of these fields, arising because of the large image files, are identified and tackled at the example of a medical research project.

The management of images needs to provide an established and easy-to-use interface for both users and independent software utilized by the users. We suggest the use of a network file store for data access. We also suggest a folksonomy for metadata management including illustrative visual tools to search through data, metadata and dataset relations.

Processing of the stored data requires a lot of random access memory (RAM). We show that the splitting of image data into smaller blocks helps in the application of sophisticated image processing algorithms along two or three dimensions at the example of lung biopsy scans.

We also analyse methods to visualise three-dimensional (3D) objects to medical researchers. We provide a tool for fast visualisation, ideal for previewing of data, and also explore new technologies to enable presentation of 3D features including the use of 3D printing for visualisation.

Finally, these approaches are shown as part of an end-to-end workflow that handles the management, processing and visualisation of images. We present an exemplary scenario showing the system in use. The workflow is highly modular and uses standard protocols and programming interfaces, which makes it easier to extend than existing workflows.

Medical researchers require new approaches for handling their image workflow to further their studies. Adaptation of existing management techniques from other domains, targeted modification of the data structure, and the use of modern manufacturing and visualisation techniques can enable this without requiring new or modified software.

Table of Contents

List of Figures

List of Tables

Abbreviations

Symbols

µ-VIS Micro-scale volume imaging system. Pages 61, 77

µCT Microfocus X-ray computed tomography. Pages iii, ix, 1–3, 9, 59–61, 63, 66, 68–70, 72, 74, 76, 77, 87, 89, 92, 95, 103, 105–107, 110, *see* CT

2D Two-dimensional. Pages 1, 6, 25, 30–32, 38, 40, 43, 60, 63, 68, 73, 76, 77, 85, 86, 88–91, 95, 96, 98, 99, 104, 105, 108, 110, 118, 119, 123

3D Three-dimensional. Pages iii, 1–3, 6, 8, 9, 21, 23, 28–34, 40, 43, 60, 61, 63–65, 68, 70, 73, 75–77, 80, 81, 84–96, 98–100, 102–105, 108–110, 113, 120, 123

5D Five-dimensional. Pages 39, 40

A

ABS Acrylonitrile butadiene styrene. Pages 89, 92–94

ACID Atomicity, consistency, isolation and durability. Pages 17, 20

ACR American College of Radiology. Page 34

AJAX Asynchronous JavaScript and XML. Pages 76, 79, 81, *see* XML

API Application programming interface. Pages 38–40, 45, 48, 50, 58, 74, 76, 98, 104, 105, 112

AR Augmented reality. Pages 33, 43, 73, 90, 94–96, 102, 110, 113

B

BASE Basically available, soft state, eventually consistent. Page 17

BisQue The Bio-Image Semantic Query User Environment. Pages 38–41, 43, 48, 58, 98, 104, 112

BJ Binder jetting. Page 35

BMP Windows Bitmap. Pages 76–78

C

CAD Computer-assisted diagnosis. Pages 2, 37

CAH Cyanoacrylate with hydroquinone. Page 92

CAP Consistency, availability, partition-tolerance. Pages 16–18

CD Compact disc. Page 30

CNN Convolutional neural network. Page 28

COPD Chronic obstructive lung disease. Page 61

CPU Central processing unit. Pages 29, 69

CSS Cascading style sheet. Pages 96, 100

CT Computed tomography. Pages 1, 2, 6, 9, 19, 26, 27, 30, 31, 53, 61, 64, 66, 74, 76, 88, 91, 92, 94, 96, 98, 107

D

DBMS Database management system. Pages 14, 16, 17

DED Directed energy deposition. Page 35

DICOM Digital Imaging and Communications in Medicine. Pages 20, 21, 30, 34, 36, 37, 43, 47–50, 56, 75, 92, 93, 101, 106

DMLS Direct metal laser sintering. Page 35

DPS Digital pathology system. Page 37

E

EAA Extrinsic allergic alveolitis. Page 2

EBM Electron beam melting. Page 35

ECG Electrocardiogram. Page 21

EHR Electronic health record. Page 20

EMAP EMouse Atlas Project. Page 76

ENT Ear, nose and throat. Page 93

F

FDM Fused deposition modelling. Pages 34, 35

Fiji Fiji is just ImageJ. Pages 62, 64, 90, 101, 123, 133, *see* Fiji

fps Frames per second. Pages 82, 84

G

GIF Graphics Interchange Format. Page 76

GPU Graphical processing unit. Pages 29, 90

GSDF DICOM grayscale standard display function. Pages 36, *see* DICOM

GUI Graphical user interface. Pages 63, 116

H

HDC The Heterogeneous Data Centre. Pages 41–43, 50, 51, 55, 56, 58, 84–87, 97, 98, 101, 106, 107

HDD Hard disk drive. Page 69

HPC High-performance computing. Page 6

HRCT High-resolution computed tomography. Pages 2, 92, 93, 95, *see* CT

HTML Hypertext markup language. Pages 88, 96

HTTP Hypertext transfer protocol. Page 86

HU Hounsfield unit. Page 81

I

ID Identification. Page 86

IIP3D Three-dimensional internet imaging protocol. Page 76

IIS Microsoft Internet Information Service. Pages 46, 55, 58, 84, 98

IPF Idiopathic pulmonary fibrosis. Page 2

IT Information technology. Pages 7, 19, 20, 105

J

JPEG Joint Photographic Experts Group. Pages 75–79, 82, 84, 103

JSON JavaScript object notation. Pages 15, 85, 86

L

LIMS Laboratory information management system. Pages 5, 37

LIS Laboratory information system. Pages 37, *see* LIMS

LOM Laminated object manufacturing. Page 35

LUT Look-up table. Page 36

M

MCTV Multiresolution Computed Tomography Viewer. Pages 9, 74, 81, 85–88, 96, 98–102, 105, 107, 109, 110, *see* CT

MDC Materials Data Centre. Page 41

MIV Multiresolution Image Viewer. Page 76

MJ Material jetting. Page 35

MJF Multi jet fusion. Page 35

MRI Magnetic resonance imaging. Pages 1, 21, 31, 36, 37

N

NEMA National Electrical Manufacturers Association. Page 34

NoSQL Not only SQL. Pages 13–17, 19–21, 42, 43, 55, 104, *see* SQL

NSIP Non-specific interstitial pneumonia. Page 2

O

OBJ Object. Pages 86–88

OECD Organisation for Economic Co-operation and Development. Page 103

OME Open Microscopy Environment. Page 38

OMERO The Open Microscopy Environment Remote Object platform. Pages 38–41, 43, 48, 58, 98, 104, 112, *see* OME

P

PACS Picture archiving and communication system. Pages 37, 38, 74, 75

PDE Partial differential equation. Page 26

PEEK Polyetheretherketone. Page 92

PHP PHP: Hypertext Preprocessor. Pages 8, 50, 51, 55, 56, 58, 87, 98, 101, 102, *see* PHP

PNG Portable Network Graphics. Pages 74, 76–79, 82, 84, 103

R

RAM Random access memory. Pages iii, 5, 29, 63, 64, 69, 84, 104, 107, 121

RDBMS Relational database management system. Pages 16, 19, 20, *see* DBMS

RDF Resource description framework. Page 15

RGB Red, green, blue. Page 77

RLE Run-length encoding. Page 78

rms Root mean square. Page 78

S

SCM Storage commitment. Pages 36, 47

SCP Service class provider. Pages 36, 47

SCU Service class user. Pages 36, 47

SHS Selective heat sintering. Page 35

SLA Stereolithography apparatus. Pages 34, 35

SLM Selective laser melting. Page 35

SLS Selective laser sintering. Page 35

SPECT Single photon emission computed tomography. Pages 53, *see* CT

SQL Structured query language. Pages 14–17, 20, 21, 41–43, 48, 50, 51, 55, 85, 101, 102

SSL Secure sockets layer. Page 39

SSO Single sign-on. Pages 55, 56, 58, 98

STL Stereo-lithography. Pages 86–88, 90, 93, 96

T

TIF Tagged Image File. Pages 63, 76, 78, 79, 87

TIFF Tagged Image File Format. Pages 76, 77, *see* TIF

TTD Time to display. Pages 81, 82, 84

TTR Time to respond. Pages 81, 82, 84

TWS Trainable WEKA Segmentation. Pages 29, 64–66, 70, 92, 123, 133–135, 141, *see* WEKA

U

UAM Ultrasonic additive manufacturing. Page 35

UK United Kingdom. Page 61

US Ultrasonography. Page 1

USA United States of America. Pages 19, 92

USB Universal serial bus. Page 19

UV Ultraviolet. Page 35

V

VASARI Visual arts systems for archiving and retrieval of images.

VIPS VASARI Image Processing System. Pages 30, 62, *see* VASARI

VPN Virtual private network. Pages 55, 84

VR Virtual reality. Pages 32, 33, 43, 73

W

WEKA Waikato Environment for Knowledge Analysis.

X

XML Extensible Markup Language. Pages 18, 19, 40, 41, 58, 76, 98

Y

YCSB Yahoo Cloud serving benchmark. Page 18

Declaration of Authorship

I, Lasse Wollatz, declare that this thesis entitled 'Modular Data Management and Image Processing Approaches to Facilitate Medical Research' and the work presented in the thesis are my own, and have been generated by me as the result of my own original research.

I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University;
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- Where I have consulted the published work of others, this is always clearly attributed;
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- Parts of this work have been published:

  Wollatz, L., Scott, M., Johnston, S. J., Lackie, P. M., Cox, S. J., Oct. 2018. 'Curation of image data for medical research'. *In: Proc. 14th Int. Conf. eScience (eScience 2018).* IEEE, Amsterdam, Netherlands, pp. 105–113. doi:10.1109/eScience.2018.00026

  Wollatz, L., Johnston, S. J., Lackie, P. M., Cox, S. J., Dec. 2017. '3D histopathology—A lung tissue segmentation workflow for microfocus X-ray-computed tomography scans'. *J. Digit. Imaging* 30 (6), 772–781. doi:10.1007/s10278-017-9966-5

  Wollatz, L., Konieczny, K., Vandervelde, C., Mitchell, T., Cox, S., Burgess, A., Ismail-Koch, H., Sep. 2016. 'The use of 3D-printed paediatric temporal bones as a training tool'. *In: BAPO Annu. Sci. Meet. (BAPO 2016).* British Association for Paediatric Otolaryngology, Liverpool, United Kingdom, p. 24

Wollatz, L., Cox, S. J., Johnston, S. J., Sep. 2015. 'Web-based manipulation of multiresolution micro-CT images'. *In: Proc. 11th Int. Conf. eScience (eScience 2015)*. IEEE, Munich, Germany, pp. 308–311. doi:10.1109/eScience.2015.42

- Additionally, parts of this work have been presented:

Wollatz, L., Frampton, S., Konieczny, K., Mitchell, T., Johnston, S. J., Cox, S. J., Burgess, A. and Ismail-Koch, H., Jun. 2018. '3D-printed paediatric temporal bones as an alternative tool for otological training'. In: *14th Congr. Eur. Soc. Pediatr. Otorhinolaryngol.* European Society of Pediatric Otorhinolaryngology, Stockholm, Sweden

Wollatz, L., Konieczny, K., Vandervelde, C., Mitchell, T., Johnston, S. J., Burgess, A., Cox, S. J. and Ismail-Koch, H., Mar. 2017. 'The use of 3D-printed paediatric temporal bones as a training tool'. In: *Surgeon Educators Day (SED 2017)*. Royal College of Surgeons, London, United Kingdom

Wollatz, L., Jul. 2015. 'Blood Vessel'. In: *CII Art Exhibition*. University of Southampton, Southampton, United Kingdom. URL: http://imaging-usrg.wixsite.com/southampton/art-gallery-2015?lightbox=i161zos

Wollatz L., Katsamenis, O. L., Lackie, P. M., Schneider, P., Warner, J., Sinclair, I., Cairns, J. H. R., Robinson, S., Scott M., Johnston, S. J., Cox, S. J., Jun. 2015. '"LungJ" plugin for ImageJ/Fiji: Implementing machine-learning in pathology'. In: *IfLS Human Nexus Poster Session*. University of Southampton, Southampton, United Kingdom

Wollatz, L., Cox, S. J., May 2015. 'Organisation and Analysis of 3D Tissue Samples'. In: *EPSRC Healthcare Technology Visit*. University of Southampton, Southampton, United Kingdom

Signed:

Date:

Acknowledgements

This project was supervised by Prof. Simon J. Cox and Dr. Steven J. Johnston, whose guidance and support throughout this Ph.D. have been very valuable. I am very grateful to Dr. Peter M. Lackie, who acted as a third supervisor and provided me detailed feedback on medical questions. Dr. Mark Scott, Dr. Gereon Kaiping, Nana O. K. Abankwa, Mihaela Apetroaie-Cristea, Andrew J. Poulter, Dr. Philip J. Basford, and Florentin Bulot all participated in fruitful discussions during our team meetings and beyond. Hasnaa Ismail-Koch and Andrea Burgess contacted me with their idea of an additive manufacturing project and gave me great insight into their work as paediatric ENT surgeons.

Several people helped me test my software, and besides Peter Lackie and Mark Scott, I would also like to mention Paige E. Bennett-Rise, Stephanie Robinson and James Cairns. For this research project, I made use of the high-performance workstations at the µ-VIS X-Ray Imaging Centre as well as images produced by their µ-CT machines. This was made possible thanks to the academic team leader Prof. Ian Sinclair and the core team member Dr. Orestis Katsamenis. I also benefited from the university's 3D-printing service and would like to thank Prof. Shoufeng Yang and Dr. Andrew Hamilton for providing me access to the facility as well as Patrick L. Fenou Kengne, who helped with some of the 3D printing. Rob Penn from iSolutions helped load my files onto a HoloLens and recorded videos of the AR temporal bone for me. I would also like to thank the Biomedical Imaging Unit, specifically Mark Jones and Anna Scott, whose image data served as examples for figures involving lung tissue in this thesis.

Finally, I thank my family, who always believed in me, and all my friends and my wonderful girlfriend, who accompanied me on this journey.

The Ph.D. was funded in part by a grant from the Engineering and Physical Sciences Research Council (EPSRC reference number 1511465).

Chapter 1

Introduction

The first computed tomography (CT) scan was produced in 1971 using a combination of an X-ray machine and a computer, developed by Godfrey Hounsfield and Allan Cormack (Hounsfield, 1973). It was a single-slice brain scan with a size of 80×80 pixels and took 5 minutes to complete (Hounsfield, 1973; New et al., 1974). The technology has, however, evolved rapidly and current medical CT scanners produce 3D images of $1024 \times 1024 \times 64$ pixels in less than half a second (GE Healthcare, 2017; Siemens Healthcare GmbH, 2016; Toshiba Medical, 2016). Compared to that of the first CT scan, the scan time has been reduced by a factor of 1000, and the image size has increased from 6 kB to 128 MB. Kemerink et al. (2011) compared an X-ray system from 1896 with one from 2011 and came to a similar conclusion, citing a reduction in the exposure time from 90 min to 21 ms.

Microfocus X-ray computed tomography (µCT) scans and 3D microscopy produce much larger images than what conventional 2D microscopes can achieve. A $2000 \times 2000 \times 2000$ pixel µCT scan at 32-bit is approximately 30 GB in size. 3D microscopy for 100 slices at 20× magnification creates approximately 90 GB of uncompressed data. In comparison, a 40× objective 2D microscopy scan produces 4 GB of uncompressed image data.

In this thesis, we will often talk about '*large images*' when referring to images being too large to process or visualise them efficiently on a '*low-performance computer*'. We classify computers as 'high-performance' ones, if their processing power is significantly higher than the average computer that can be bought on the market. This includes supercomputers and grid computers, but also (expensive) targeted or custom-built machines. All other computers we classify as 'low-performance'. For the topic of this thesis, we will exclude devices that have computational power but are not commonly used for general computational tasks, such as microcontrollers or mobile phones, when talking about computers. We do not attach any specific figures to the performance of low- or high-performance machines since their performance is continuously improving. Instead, we assume that there will be a proportional increase in both image size as well as computing power over time. This trend is shown in Figure 1.1.

Computers have been of great importance since digitalisation of images took over. The first X-ray images were produced on a film (Röntgen, 1898). Recently, with the introduction of CT, images have begun to be stored digitally, and many digital imaging techniques have been developed over the last century, such as magnetic resonance imaging (MRI) and ultrasonography (US) (Laal, 2013). Medical research relies on top-end imaging devices, which produce the largest images. An example for medical research relying on large images is given in the next section.

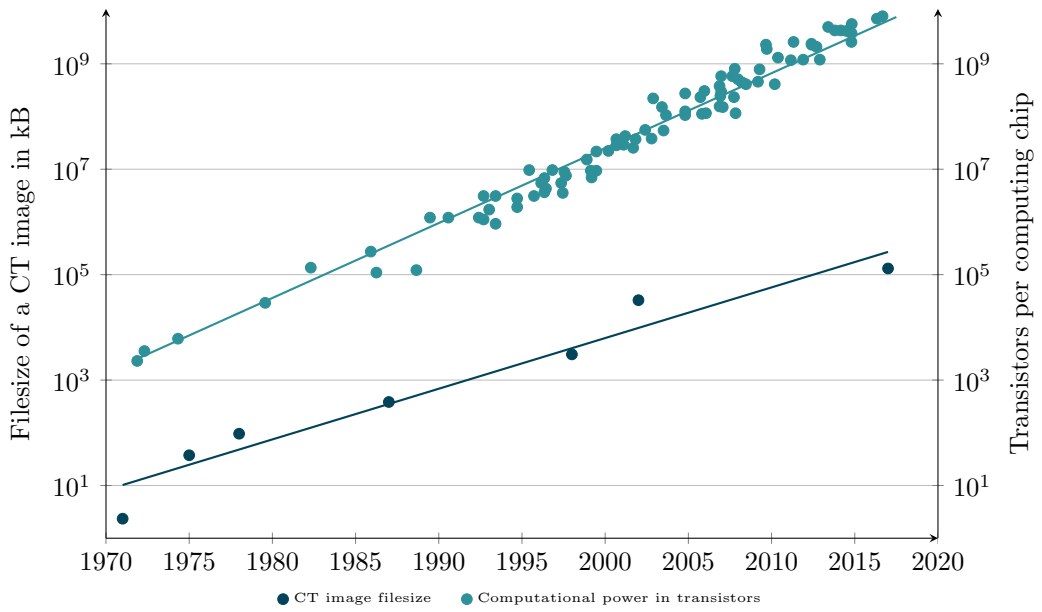
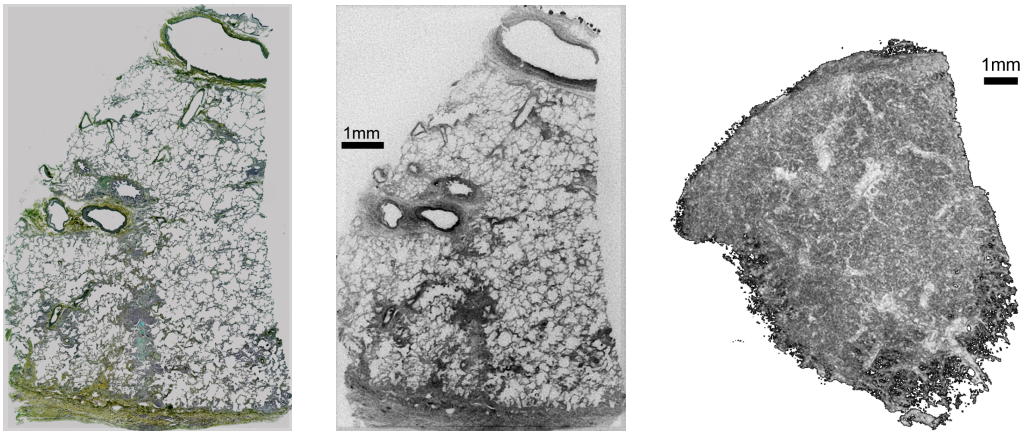


FIGURE 1.1: Comparison of CT image size and computational power over time using the number of output voxels for CT devices (GE Healthcare, 2017; New et al., 1974; Suryawati, 2014; Wong and Lou, 2000) and the number of transistors per computer (Rupp, 2018) as suggested by Moore (1965) (Golio, 2015).

1.1 New Imaging Systems in Histopathology

Histopathology provides structural details of tissue samples on a cellular level, allowing disease-associated changes to be identified. It offers a key diagnostic tool for fibrotic lung diseases, particularly those that cannot be clearly identified on the basis of patient CT or high resolution computed tomography (HRCT) (Brown, 2006; Dawson et al., 2002; Ko et al., 2015; Webb et al., 2014). Histopathology is often regarded as the gold standard (He et al., 2011; Raghu et al., 2015). For routine histopathology, surgical biopsies are taken from a patient, providing 3D tissue samples that are chemically fixed to preserve the tissue structure and embedded in wax to allow for histological sectioning. For diagnostic purposes, sections are stained to identify the overall tissue structure and highlight specific tissue components before analysis under a microscope by a trained histopathologist. These microscopy images are being increasingly obtained through digital scans of tissue slices. This allows analysis based on the digital image (virtual microscopy). Systems used for computer-assisted diagnosis (CAD), to highlight relevant features or suggest a possible diagnosis, are also becoming available (Raghu et al., 2015; Ross and Pawlina, 2011).

The established methods in histopathology have proven to be critical in the diagnosis of many lung diseases. Diagnosis of some diseases such as idiopathic pulmonary fibrosis (IPF), non-specific interstitial pneumonia (NSIP) and extrinsic allergic alveolitis (EAA) is difficult and agreement between histopathologists is generally only fair to moderate (kappa agreement coefficients ranging from 0.2 to 0.7) (Nicholson et al., 2002). Medical researchers at the University of Southampton have developed a new workflow that involves 3D images of biopsies, to improve the diagnosis of lung diseases. Generation of 3D scans is achieved through the use of µCT scanners. Part of their project involves the development of a custom µCT scanner. Prototype scans have reached



(a) Traditional microscopy image (Scott et al., 2015)

(b) Section of a µCT scan showing the same area as (a) (Scott et al., 2015)

(c) 3D visualization of the whole image block

FIGURE 1.2: Current and proposed histology workflow in images.

a resolution of up to $2000 \times 2000 \times 2000$ voxel and a grayscale depth of up to 4 B. The raw image file size therefore was 29.7 GB.

By producing a three-dimensional scan of the sample, additional information about the tissue will be gained (Hsia et al., 2010b). Instead of single slices, 100 to 1000 times the number of slices can be viewed at various slice orientations. This aids the identification of rarer structural changes and reveals the degree of heterogeneity in the tissue structure. Further benefits of 3D scans include analysis of 3D structures (see Figure 1.2), specifically 3D networks. These reveal tissue functions as well as interrelationships between objects (Hsia et al., 2010b; Mitzner and Weibel, 2010), enabling the application of established stereological methods (Hsia et al., 2010a).

The project has already produced several µCT scans that have been used for testing purposes in the research described in this thesis. Preferable µCT settings and a suitable reconstruction method are still being tested. The images provided vary in size and bit depth. The hospital holds around 60 000 lung biopsies and more than 2500 samples are added every year (Peter M. Lackie, personal communication, 2018). The project is intended to encompass data of this scale in the future. Currently, images are segmented manually for the detection of network structures. This procedure takes around 3 months per image. For evaluation purposes, the segmented images need to be compared with traditional microscopy images by experts in the field. Further research is expected to find new ways of understanding and diagnosing lung diseases.

The example of µCT scans in histopathology and the trend shown in Figure 1.1 imply that researchers require high-performance computers to accomplish their various tasks. The next section will introduce these tasks as parts of the image life-cycle.

1.2 The Image Life-Cycle

The use of images in medical research is complex. It involves many people with different jobs and backgrounds. The use also varies drastically between different research projects. In this

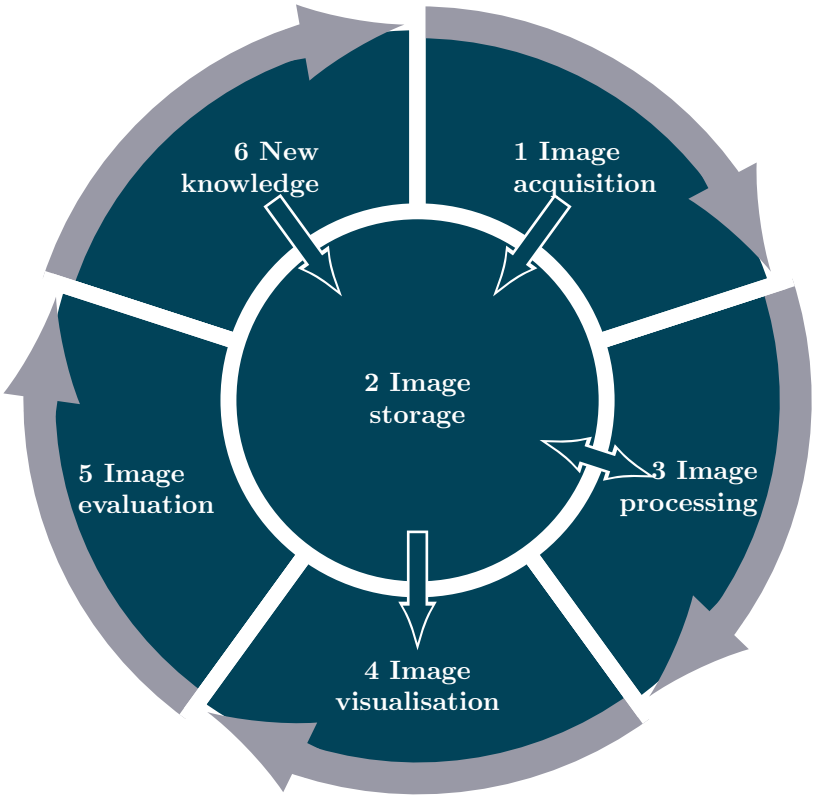


FIGURE 1.3: The image life-cycle in medical research.

section, we will illustrate the use of images in medical research by introducing the 'image life-cycle', shown in Figure 1.3. This life-cycle shows the individual steps of the workflow in using images in medical research.

As shown in Figure 1.3, the first step is to *acquire* digital images. Images are generated as part of tests from regular patient diagnosis or specifically for research projects. They can be generated directly from a patient through non-invasive image acquisition techniques or by imaging of a biopsy. These two fields of imaging are commonly referred to as radiology and pathology, respectively (Hipp et al., 2011; Sorace et al., 2012).

Then, these images need to be stored for the duration of the research project and, if appropriate, beyond the end of the project, to guarantee researchers access to the data. Some fields of research and some funding organisations require data to be stored for decades after a project has ended (BBSRC, 2017; EPSRC, 2004; Montagnat et al., 2004). Metadata and access permissions of anonymised but often-sensitive data need to be managed, as discussed in Sections 2.1.1 and 2.1.2. This means that *image storage* (see Section 2.5) fulfils a central role in the whole research cycle (Castiglione et al., 2015; Higgins, 2008).

Once the images are acquired and organized by a storage system, they need to be processed to extract the information of interest (Robb, 2000, Chapter 5.1). *Image processing* involves image enhancement and segmentation and can extend to feature detection, which generates new image data and metadata, which in turn needs to be managed. It also includes the improvement of

digital images for human interpretation (Kumar and Shaik, 2016). A more detailed review of image processing is presented in Section 2.2.

Data then needs to be *visualised* for easy analysis by a human (Robb, 2000, Chapter 4.1) (see Section 2.3). Previewing of images helps researchers identify data of interest; it also allows humans to interact with the data more naturally.

Images can be *evaluated* and interpreted by expert researchers after visualisation, which eventually leads to the generation of *new knowledge*. This new knowledge can be added to the stored data in the form of reports and metadata. Based on the new knowledge, new research questions arise, which restarts the cycle. Data from old research projects can be reused for later cycles with help from the image storage system.

We have summarized the main steps of the image life-cycle as image acquisition, image storage, image processing, image visualisation and image evaluation. This aligns with the example of digital pathology as explained by Griffin and Treanor (2017). The field of pathology is moving towards the utilisation of digital technologies. According to them, digital pathology includes a laboratory information management system (LIMS), digital dictation, dashboards, workflow management, digital image analysis, specimen labeling, tracking and synoptic reporting tools—in other words, the storage of images and related data, visualization tools, and image processing. Allan et al. (2012) names the 'collection, management and analysis' of datasets as the most important aspects of biological research. The analysis undertaken in their work corresponds to the processing, visualisation and evaluation of the life-cycle shown in Figure 1.3.

In this section, we have looked at the use of images in medical research. We introduced the medical image life-cycle, which governs the workflow for images in medical research. The next section will show how the increase in image size discussed in the beginning of this chapter affects this image life-cycle.

1.3 Associated Problems

Images used in medicine are increasing not only in number but also in size. Castiglione et al. (2015) described this phenomenon as an 'atypical big data problem'. The current medical image workflow is unable to cope with this load at the time of the publication of this thesis.

Image management systems target the much smaller images currently used in hospitals by radiologists. For instance, for the viewing of an image, most software would load the entire image file into the random access memory (RAM). For small images, this is beneficial, as it avoids constant file access. But large images can fill up the RAM of a computer very quickly. A 90 GB image would require 90 GB of RAM to load the image and will create a second copy requiring another 90 GB to run a filter that is not 'in place', for example, a simple Gaussian blur (see Section 2.2.6.1). At the time of writing this thesis, only high-performance computers can process these kinds of images and future improvement of computing power is expected to be accompanied by increase in the image size produced by newer imaging devices. This means that the software involved in the medical image life-cycle needs to be able to cope with lower hardware performance than ideally required. Otherwise, research institutions and health services will face

regular, significant investments in the latest high-specification computers to enable research and implementation of the best patient treatment possible using current technological standards.

Many people working at different locations were involved in the project introduced in Section 1.1. Buying a central high-performance computing (HPC) station is affordable in such scenarios, but buying several HPC stations to support all researchers involved is not desirable. Researchers want their experiments to be as facile as possible, which means that image viewing and processing should take place on their office computer where possible. The workflow should allow easy introduction of different software to enable their testing in the early research stage.

1.4 Requirements for a New Image Management Workflow

Many people are involved in the medical image workflow. These people include the professionals operating the CT scanners, researchers in the laboratories working on the biopsy preparation, and doctors and specialists who interact with different parts of the image workflow. A majority of these positions require a high level of experience, indicating that changes require people to adapt suitably. For new images and new workflows to be efficient, it is necessary to make them as user-friendly as possible (Hipp et al., 2011). For example, one needs to consider how to display 3D grey-scale CT images to an audience accustomed to 2D microscopy images, where different cells are stained in different colours. At the same time, advantages of 3D images in comparison to 2D images need to be emphasised to encourage the use of new image visualisation techniques. This example emphasizes the importance of user-friendly modifications which influence the choice and development of software as well as the visualisation of images.

The improvement of the workflow, while considering demands from both the hardware and the end-user, forms a major challenge. When trying to solve this challenge, previous research identified specific areas of importance affecting all areas of the image workflow (Allan et al., 2012; Amat et al., 2015; Griffin and Treanor, 2017; Kvilekval et al., 2010; Montagnat et al., 2004; Müller et al., 2004):

- support for large-sized images
- usability
- system modularity
- metadata support
- data traceability
- fast response & visualisation
- data security

'*System modularity*' refers to the ability to separate the system into several independent parts which communicate with each other through standard protocols. '*Data traceability*' refers to the ability to determine the origin of a data record. It can be extended to data provenance, wherein

explanations of the transformations and reasons for the transformations are recorded in addition to the origin (Gupta, 2009).

Parallel (high-performance) computing is also considered an essential aspect of an image management system (Montagnat et al., 2004), but it applies to computation-intensive image processing algorithms, which are out of the scope of this thesis. So far, the medical research project in question had to decide on the image processing algorithms to use and speed has not been an issue. We will therefore not consider parallelisation further for our specific image management workflow but will briefly discuss parallel processing in the context of the end-to-end workflow in Section 6.2. Most solutions focus on the functionality but ignore the need for a user-friendly system though they will be used by medical experts rather than information technology (IT) professionals. Ever-increasing image file sizes lead to the requirement of a larger total storage space—a problem not solved by most existing workflows which only consider the number of images (Kvilekval et al., 2010).

1.5 Research Question and Objectives

In this thesis, we will demonstrate approaches that apply to the life-cycle of medical research images. In Section 1.3, we identified the main parts of this image life-cycle. This thesis will focus on the improvement of the image storage, image processing and image visualisation. We will not deal with image acquisition since we see imaging as the cause of the problems but explicitly do not want to impose restrictions on the medical imaging process and the size of the data constructed. The analysis/evaluation of data is also considered as out of the scope of this thesis since it is highly reliant on specific medical problems. Instead, we will keep the approaches as general as possible, and only use specific examples and use cases to demonstrate the application of the approaches.

1.5.1 Overall Research Question

What approaches support large images in medical research without significantly changing the established workflow?

1.5.2 Specific Objectives

In order to address the overall research question, we aim:

1. To provide a framework for image storage that can be utilised by medical researchers and by the software that is commonly used by the researchers.
2. To develop a method that enables the processing of very large images using existing software and processing algorithms.
3. To implement the processing method from objective 2 as a software or plug-in and show that it can be applied to an existing image processing algorithm.

4. To develop visualisation methods for medical researchers to view large 3D images in a way familiar to them.

5. To explore the possible use of new and sophisticated visualisation methods in medicine.

6. To show that all the developed tools are coherent with the requirements for a medical research image workflow.

1.6 Novel Contributions

This section presents the details of the novel approaches and their contributions to the image life-cycle in medical research that have resulted from this research. This thesis has led to the following novel contributions.

1.6.1 An Image Management Framework

This work involves the design and implementation of a file store-based image management system, Mata[1] (a combination of medicine and data). It addresses the first objective. The main contributions of Mata are as follows:

- It improves accessibility over existing medical management systems, by giving users direct access to a file store. This makes it more intuitive to use and easier to combine with existing image processing software in comparison to other image management systems.
- It provides an improved interface in terms of usability for the metadata management, metadata visualisation and image visualisation.
- It is easier to maintain for the system administrators since it uses a lightweight PHP (PHP: Hypertext Preprocessor) website.
- It is also designed to be modular, making it extensible. Pieces, such as plug-ins or image-viewers, can be added easily. Moreover, parts such as the front-end or the database can be exchanged if required.
- It is scalable, since it grows with the file store capabilities. It also provides the necessary security features for medical research data.

1.6.2 A Method for Processing Large Images

This work involves the design, implementation and evaluation of an approach for the processing of extra large images without the need of high-performance computers. It addresses the second and third objective. The main contributions of this approach are:

- It provides a novel method to process large images by splitting them into manageable chunks.

[1] Source code and documentation published as doi:10.5258/SOTON/D0430 (Apache 2.0 License)

- It has been implemented as a plug-in, LungJ[2], by extending the popular open source image processing software ImageJ.
- It is the only tool known to the author that works for custom processing algorithms and any size image without the need for a high-performance computer. Other tools tested failed in this context as they were not able to reduce the memory requirement for low-performance hardware.
- It has been designed to be modular, so individual parts can be altered and improved.
- It has been successfully tested to work for threshold filter application with 4 different CT images. It was also used to enable the application of a machine-learning algorithm to 5 different µCT images.

1.6.3 A Web-Viewer for Large Volumetric Images

This work involves the creation of a tiled image viewer for 3D volumetric images, Multiresolution Computed Tomography Viewer (MCTV)[3]. It addresses the fourth objective. The main contributions of the viewer are as follows:

- It allows fast previewing on client devices by limiting the overall data sent to the client.
- It enables basic image modification such as changing the value display range and measuring distances on the displayed image.
- It does not require any extra software and runs in any of the major modern browsers using the native JavaScript.

1.6.4 A Workflow for Creating 3D Printed Temporal Bones

This work involves the development and testing of a workflow to create 3D-printed paediatric temporal bones as a new visualisation method for surgeons' training. It addresses the fifth objective. The main contributions of the workflow are as follows:

- It can create models with a much more accurate anatomy than that of existing training models.
- It manages to produce 3D-printed temporal bones at a much lower cost than other methods in this research area due to our choice of the 3D-printing technique and cheap material.

1.7 Thesis Structure

Medical workflows for digital images are not capable of dealing with the increasing size of images. To tackle this issue, this thesis will look at different steps of the workflow and provide solutions

[2]Source code and documentation published as doi:10.5258/SOTON/401280 (Apache 2.0 License)
[3]Source code and documentation published as doi:10.5258/SOTON/400332 (Apache 2.0 License)

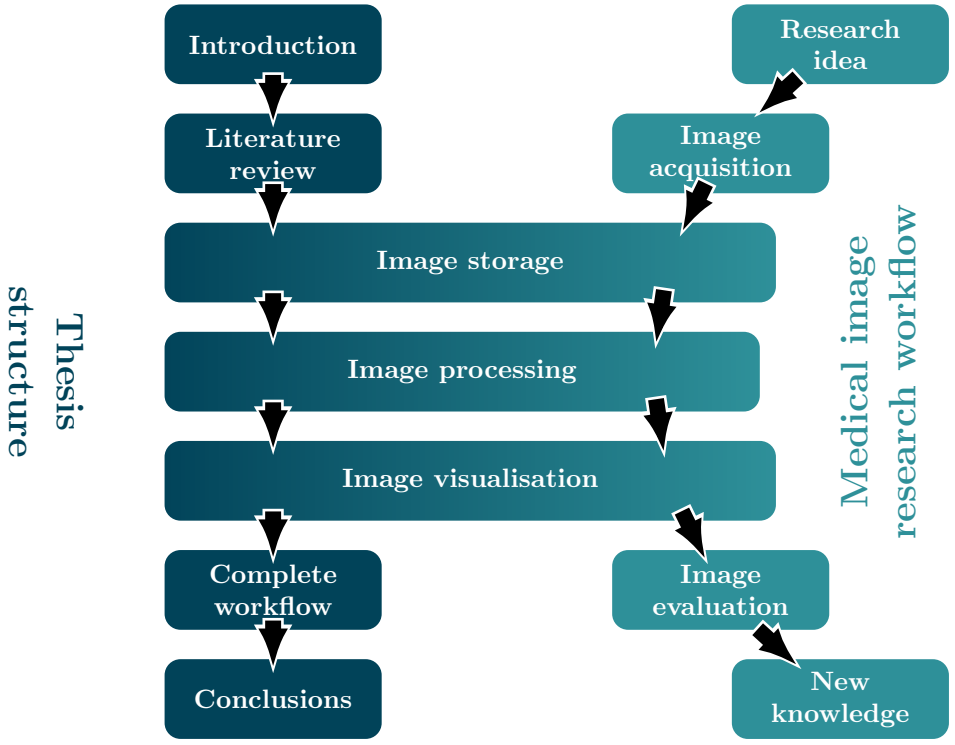


FIGURE 1.4: Structure of the thesis and its relation to the workflow of images in medical research. Image acquisition, as well as the evaluation of images and generation of new knowledge in the medical domain is very specific to the medical research project and therefore will not be considered as part of this thesis.

for each of them. The outline is shown in Figure 1.4. The current chapter discussed a general overview of the work. Literature providing background for the different workflow steps will be introduced in Chapter 2. Chapters 3, 4 and 5 will each look at one step of the workflow and discuss the developments of new and improved approaches to the step in question.

Chapter 3 addresses the storage of images, including the storage of image files produced by an imaging device as well as the creation and attachment of metadata and finding and retrieval of relevant images stored. In Chapter 4, the extraction of information from images through digital processing is discussed. It develops an approach for processing large images on a low-performance computer. The last step of the workflow analysed is the visualisation of images to the specialists. The approaches in Chapter 5 help these specialists make an informed decision as to what the image contains, possible diagnosis, and possible treatment methods. Chapter 6 will summarise all the proposed and evaluated solutions and explain how to combine them into one workflow. It will also address limitations of the workflow.

Finally, Chapter 7 will conclude this thesis by revisiting the objectives and discussing how each of them has been addressed throughout the thesis. A final outlook will be provided in the form of future work.

Chapter 2

Literature Review

The workflow for medical research images involves an extensive number of research areas. In the previous chapter, we introduced this field and laid out how we will address the various steps of the workflow. We identified image management, processing and visualisation as the key components of the image life-cycle to be addressed in this thesis. Before we develop new methods and approaches to solve the problems named in Section 1.5, this chapter will introduce basic concepts and give an overview of previous research.

Image management, the first key component, involves the storage of images and storage and editing of related metadata. Therefore, we will review the concept of metadata and databases in Section 2.1. The second key component, image processing, is addressed in Section 2.2. Section 2.3 addresses image visualisation as the third component. In medicine, the standard for digital images unifies image storage, transfer and visualisation for different vendors. Since this standard is very important in medicine, Section 2.4 contains a brief review of it. The combination of metadata storage, data storage and image visualisation and processing will be discussed in Section 2.5 again. A summary of this chapter is provided in Section 2.6.

2.1 Image Storage

The storage of images has been identified as the central piece of the image life-cycle in Section 1.2. Images need to be managed as soon as they are created. Image storage involves storage of the actual image data. Besides raw image data, related metadata needs to be managed. We will introduce the concept of metadata in Section 2.1.1. The metadata is stored in a database to facilitate answering queries about the data. We therefore review database systems in Section 2.1.2.

2.1.1 Metadata

Metadata is information related to but not contained within the actual data (Hansen and Johnson, 2004). Metadata is important for correct interpretation of the data (Linkert et al., 2010) and appears in the form of single values assigned to a dataset or key-value pairs (Scott et al., 2014). These values or key-value pairs are often referred to as tags (Ghabayen and Noah, 2014; Golder and Huberman, 2006). Different ways can be applied to organise metadata, and this section will introduce the two extreme schemes: '*taxonomy*' and '*folksonomy*'.

'A taxonomy is a scheme for organising and classifying information, objects, life forms, and other items' in a structured hierarchical way (Karch, 2016). Every item falls into exactly one category of a taxonomy scheme, which can itself be a subcategory leading to groups of categories and a hierarchical structure. Categorising items in such a strict way requires the categories to be pre-defined by specialists. A taxonomy system, therefore, has limited scalability (Mathes, 2004).

In a folksonomy system, users create tags for the data, but there is no strict categorisation of the tags (Hammond et al., 2005). Data in a folksonomy can have multiple tags. In this system, tags are created for datasets by each user. A folksonomy follows how people structure the data in their minds more closely. This makes tags in a folksonomy less comparable as different users have different ways of structuring data (Mathes, 2004). This leads to the downside of using a folksonomy, that is, the ambiguity of tags. Different users may use the same tag to describe different items, and even one user may change his way of describing and categorising items over time or use the same tag for two different items (Mathes, 2004). For example, 'Cloud' may refer to a cloud in the sky or to an online data service. Using one term with different meanings is only one aspect, the other being different terms describing the same thing. Users tend to use synonyms to describe the same item (Mathes, 2004). For instance, 'machine learning' and 'neural network' are often used interchangeably even though strictly they do have different meanings.

In a folksonomy, a hierarchy of tags is not given, but the similarity of tags is. By applying computational power to a folksonomy, it is possible to identify similar structures between users and accordingly pre-determine the tags a user would give to an item (Ghabayen and Noah, 2014).

In a taxonomy, it is difficult for a user to find the category describing the items they are interested in but, once found, every related document can be presented. A folksonomy represents users' terminology. This makes it easier for lay users to search through data. Having users create tags distributes the load of tagging from a few experts to the whole community. Regarding the storage of metadata, one has to take the advantages and disadvantages of these different representation methods into consideration. An important aspect is the existence of an applicable taxonomy for the field or the feasibility of developing one.

2.1.2 Databases

Databases store related information in a systematic way for retrieval on demand (Elmasri and Navathe, 2011, Chapter 1). In Section 2.5, we will investigate different data management systems. All of these use one or more databases to store metadata about the managed data, user access permissions and data relationships.

There are different types of databases and different architectural designs to make information from databases available. The architecture has a major impact on the scalability of the proposed application regarding multiple images and users. This section will present the different architectures and database types and compare their performance and security.

2.1.2.1 Database Architectures

Database servers and web servers are commonly kept on separate machines to improve performance. Each of them (database and web application) can be scaled up to multiple servers to

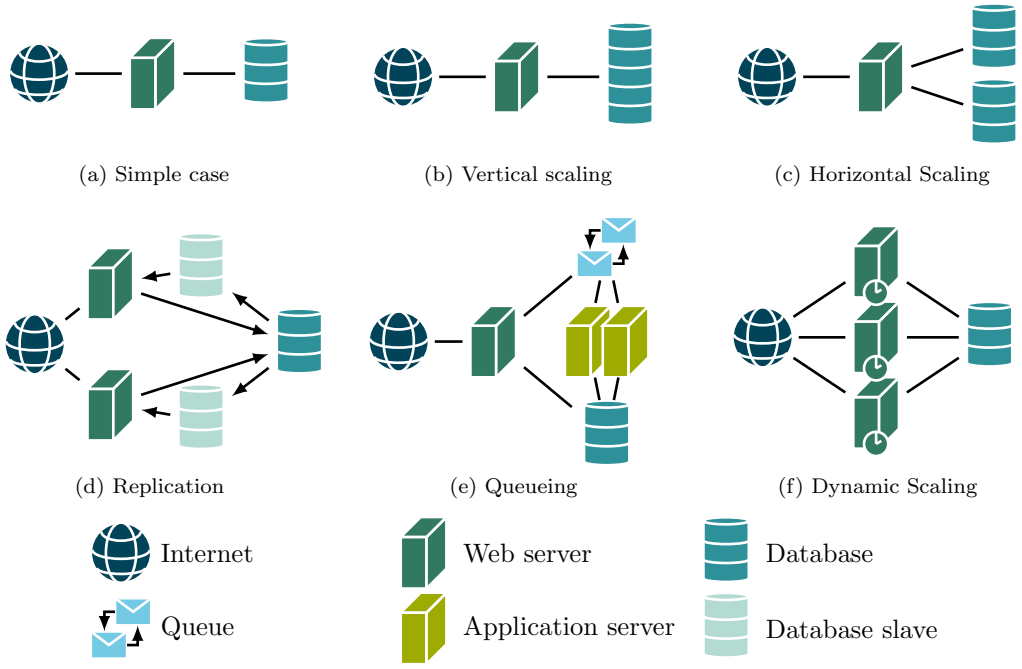


FIGURE 2.1: Different database architectures: (a) In the simple case, a web server handles requests and consults a database. (b) Vertical scaling increases the performance of a database server to handle more data, while (c) horizontal scaling splits the data over multiple databases. (d) Replication allows efficient load balancing for read-intensive environments by providing additional copies of the database. (e) Application servers can be added to perform tasks based on a queue. (f) The addition of servers can occur dynamically based on current load, in which case it is called dynamic scaling.

reduce the load and increase the performance of either side. In the simple case, a webserver is connected to a database. It stores data in the database and requests information from it (see Figure 2.1a). Improving the database performance involves replication and vertical or horizontal scaling, while the application side can be improved through queues and caching. Both sides may allow dynamic scaling. An overview of the different architectures is given in Figure 2.1.

If the database size increases, the database server needs to be scaled. This can happen as vertical scaling or horizontal scaling. *Vertical scaling* (see Figure 2.1b) describes the process of increasing the performance of a single machine (i.e. by upgrading to a more powerful server) while *horizontal scaling* describes the partitioning of the data across multiple servers (see Figure 2.1c) (Mullins, 2017). This partitioning is commonly done through sharding. Sharding uses identically structured databases. Each database server receives an equal amount of database entries. For sharding to work, knowledge of where to look for which piece of information has to be retained. Therefore, alternating splitting of information is of advantage (Kubacki, 2010). Horizontal scaling is easier to implement for not only structured query language (NoSQL) databases. This will become important when choosing a database type in the next section. If most of the database operations involve reading information from the database, extracts of the database can be copied to slaves (see Figure 2.1d). *Replication* does not handle increases in write load as all writing operations are handled by the master (e.g. Mullins, 2017). These methods help improving the database servers' performance. Sometimes it is necessary to improve the performance of the web server.

If specific requests are very common, their result can be *cached* to improve performance. For example, a website showing a medical image may display related images based on a complicated

algorithm. Every visitor of the page will request the latest computation of related images, but new images might come in only every other hour or day. In that case, the result can be stored in memory together with a timestamp. The next visitor gets to see the same related images if the result has not expired (e.g. Grigorik, 2018). *Queues* are a means of splitting the workload according to the difficulty of the tasks (see Figure 2.1e). Tasks are sent to a queue which acts as a to-do list. If a server is idle, it checks the next task in the queue and works through it. Once completed, it can take up the next task. Similar to parallel computing methods, queues are useful for unbalanced tasks. If each chunk of work amounts to the same processing time, it is more efficient to pre-allocate the tasks to the servers (e.g. Wasson, 2017). We will make use of queues later in Section 5.1.4.

Dynamic scaling, shown in Figure 2.1f, allows for great reliability while reducing cost. Cloud providers offer servers charging by the minute. If the system can expand the number of web servers or database servers according to the current requirements, none of the servers paid for are being idle. Scaling for database servers can be difficult to implement, as the location of the information needs to be known (Kubacki, 2010). Database servers are more likely to only be scaled up and therefore are less likely to profit from dynamic scaling.

To a certain limit, it is possible to scale up the performance of a single machine vertically. Afterwards, horizontal scaling becomes necessary. In a read-heavy environment, replication can help reduce the load on a master server. For applications, load balancing is important and may be achieved through queues while duplicate work can be avoided through caching. Not all architectures can be implemented equally well for different databases. Therefore, the choice of database type will affect the possible ways of scaling the database performance. When designing a storage system as discussed in Chapter 3, we will need to consider scalability for potential increase in the amount of data during the medical research project described in Section 1.1. Certain database types provide better scalability than others. The next section will look at the different database types available for storing metadata.

2.1.2.2 Database Types

There are five different types of database management system (DBMS) commonly distinguished. These are relational databases, key-value stores, document databases, column databases and graph databases (e.g. Vaish, 2013, Chapter 3). Often, they are referred to as 'structured query language (SQL)' databases (relational databases) and 'NoSQL' databases (the other four), as shown in Figure 2.2.

Relational databases store entries in tables (see Figure 2.3a) which have predefined columns for each piece of information associated with an entry. Entries can be added as rows. Information from relational databases can be extracted using the SQL (Elmasri and Navathe, 2011, Chapter 3).

Key-value stores are the simplest databases and store information as keys and corresponding values, as shown in Figure 2.3b. They provide very good write availability but are limited in terms of data structure representation and querying (Strauch, 2011).

Document databases store entries in documents, wherein the information within each document can vary (see Figure 2.3c). Adding schemas to a document store allows it to behave almost

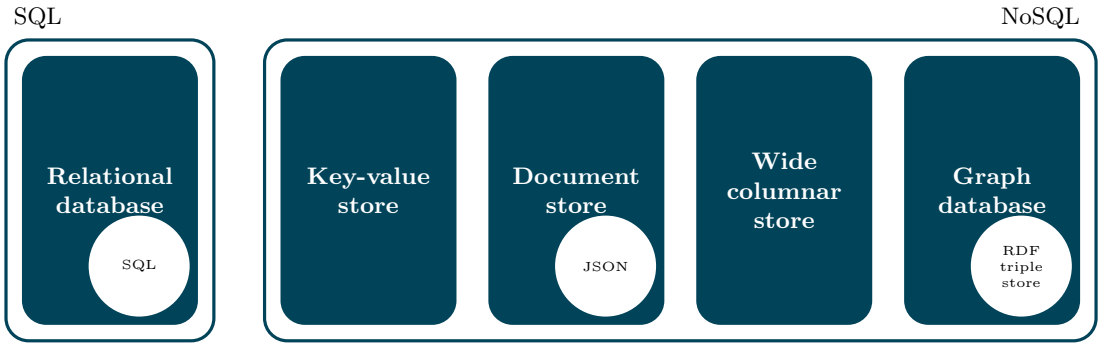
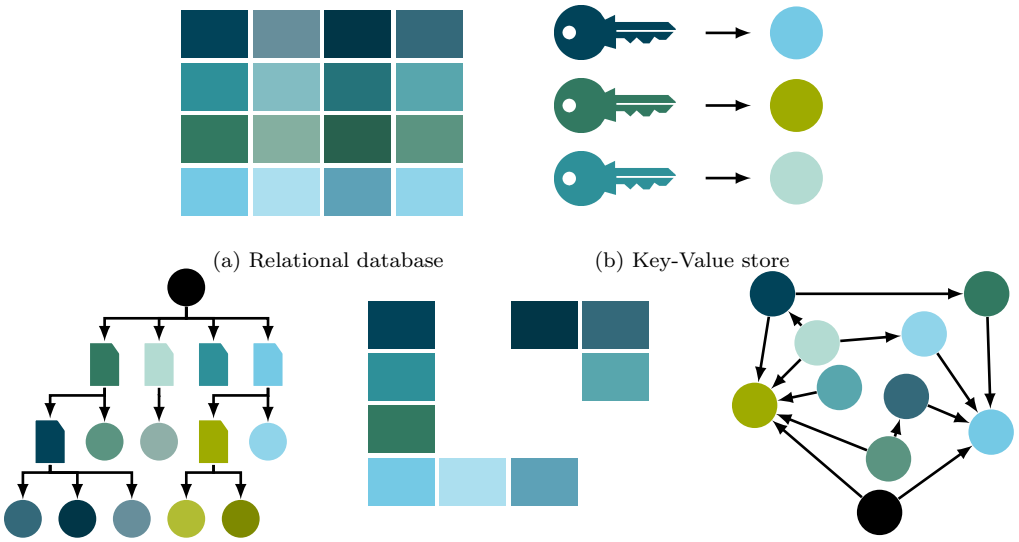


FIGURE 2.2: Database types: Databases come in a great variety but are often grouped into SQL and NoSQL. Some database types have established standards.



(a) Relational database (b) Key-Value store

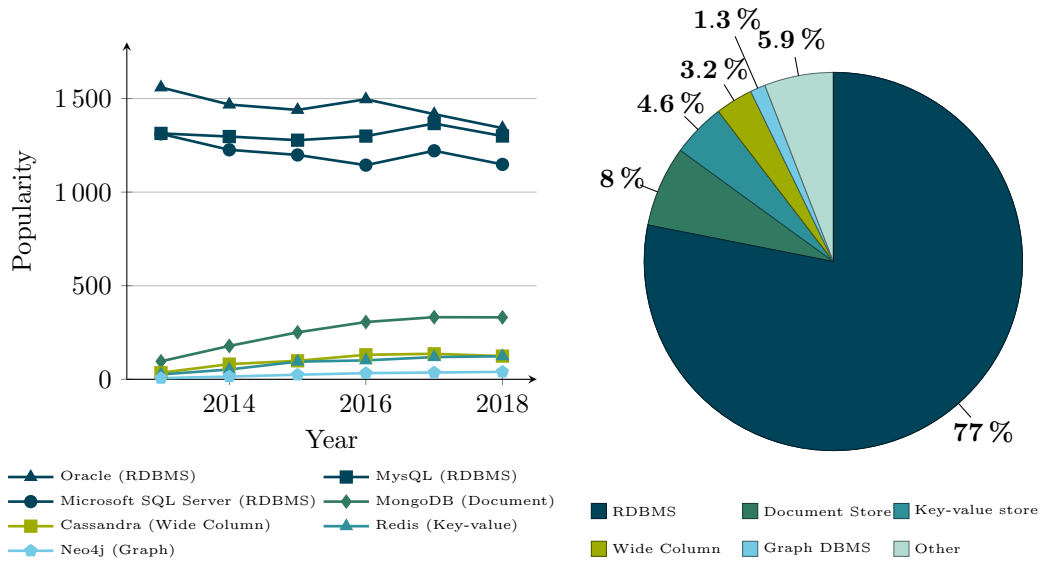
(c) Document store (d) Wide columnar store (e) Graph database

FIGURE 2.3: Database type illustrations. A relational database has pre-defined columns and stores values as rows (a). A key-value store simply stores pairs of keys and values for the key (b). Document stores can have several values grouped in a document and even documents inside a document (c). A wide column database has tables similar to the ones a relational database has, but only has a fixed primary key while the other values are not required or are not strictly defined (d). In a graph database, values and relations between values are defined (e).

identically to an SQL database. One of the standards established for storing document databases is JavaScript object notation (JSON) (e.g. Adamanskiy and Denisov, 2013).

Wide columnar databases (see Figure 2.3d) look similar to SQL databases but only allow querying by a primary key. Their integrated grouping of columns allows for horizontal sharding (Bakshi, 2012; Eini, 2010; Han et al., 2011).

In *graph databases*, each entry forms a node of a graph and nodes are connected in a relational way as shown in Figure 2.3e. This simplifies cross-references and enables topological queries for large-scale topological data (Liu et al., 2017). Among the various database storage models, the resource description framework (RDF), commonly known as triple store, will be highlighted due to its



(a) Popularity trend

(b) Popularity August 2018

FIGURE 2.4: Database popularity according to DB-Engines (DB-Engines, 2018).

increasing popularity. Based on the binary-relational model (Abrial, 1974), it stores relationships as triples of the form 'subject, relation, object'. Accordingly, all subjects with a certain object can be queried, as well as the object value of a relationship for a certain subject (Mariani, 1992).

A number of other database models, as well as hybrid models, exist. Triple stores, for example, do not have to be implemented as graph databases. There are attempts to implement them in document stores or relational database management system (RDBMS) as well (Tomaszuk, 2010).

NoSQL databases are still comparatively new. According to the database ranking by DB-Engines (2018) (see Figure 2.4), Oracle, MySQL, and Microsoft Server SQL are by far the most popular database systems. This is also driven by the fact that more expert knowledge is available on SQL databases and most companies rely on and hire people for RDBMS. In 2013, there was only a single NoSQL database vendor in the top 10. However, as illustrated in Figure 2.4 and Table 2.1, there has been an increasing interest in NoSQL, and in August 2018, 4 NoSQL databases made it into the top 10 (DB-Engines, 2018).

2.1.2.3 Comparison

The choice of database largely depends on the aim of the application. DBMSs implement different query possibilities and different ways of presenting the data, which may be more or less natural to the posed problem. The speed of databases varies depending on the posed application. Contrary to most software solutions, a comparison in terms of speed and cost is difficult and meaningless for databases. Benchmarks are inconsequential out of the context of a specific application. The 'CAP (consistency, availability, partition-tolerance) Theorem' and speed comparisons will be introduced here as two ways of how to compare DBMS apart from their data representation.

TABLE 2.1: Popularity of database providers: Ranking by DB-Engines, as of August 2018, for the top three providers of each database type (DB-Engines, 2018).

Database Type	**Provider**	**Popularity Rank**
Relational DBMS	Oracle	1
	MySQL	2
	Microsoft SQL Server	3
Key-value stores	Redis	7
	Amazon DynamoDB (Key-value and document)	21
	Memcached	24
Document stores	MongoDB	5
	Amazon DynamoDB (Key-value and document)	21
	Couchbase	23
Wide columnar stores	Cassandra	10
	Microsoft Azure Cosmos DB (Key-value, document, wide columnar and graph)	29
	Datastax Enterprise (Wide columnar and graph)	43
Graph DBMS	Neo4j	22
	Microsoft Azure Cosmos DB (Key-value, document, wide columnar and graph)	29
	Datastax Enterprise (Wide columnar and graph)	43

The 'CAP Theorem' (Fox and Brewer, 1999) states that a database system can achieve only two out the following three properties:

- 'Consistency', meaning that the data accessed from outside is always up to date;
- 'Availability', referring to the availability of the data, meaning that a request to the database will always lead to a response within reasonable time; and
- 'Partition Tolerance', describing the ability of a database to be spread across multiple instances without operation being affected by loss of communication between nodes.

As reviewed by Brewer (2012), the CAP theorem is a fluent model and database solutions can lie anywhere within the triangle shown in Figure 2.5.

Various database models and implementations are spread across the triangle shown in Figure 2.5. Many NoSQL systems are BASE (Basically Available, Soft state, Eventually consistent), whereas MySQL and other relational database systems fully meet the ACID (Atomicity, Consistency, Isolation and Durability) criteria. While relational databases do not have a high partition tolerance, NoSQL databases usually show a great partition tolerance and spread between consistency and availability. This means that NoSQL provides better scalability than SQL, but the choice of a NoSQL architecture also implies a choice between availability and consistency.

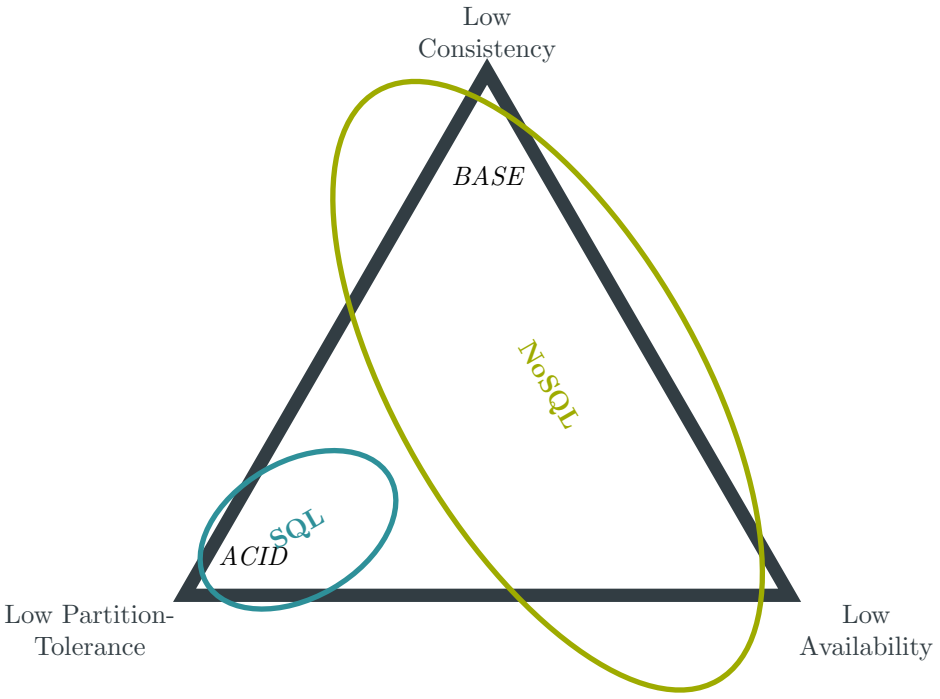


FIGURE 2.5: Databases in CAP theorem: Databases represented in a negative CAP triangle. As CAP states that only two out of three options can be achieved.

Speed comparisons between different database types are not very well established. They might be misleading, as different database types target different data structures and therefore some are expected to outperform others in the scenario they were designed for. Some benchmarks such as the Yahoo Cloud serving benchmark (YCSB) exist, and further ideas such as LinkBench have been proposed (Armstrong et al., 2013; Kubacki, 2010; Ray et al., 2011). Several comparisons have been conducted and will be presented here.

Tudorica and Bucur (2011) compared Cassandra, HBase and MySQL—two wide column databases and one relational database. The main difference was seen in the consistency, as wide column store implementations were eventually consistent. Moreover, MySQL was limited to about 1 million average-sized database entries, while wide column stores were documented to take several tens of millions of entries (Tudorica and Bucur, 2011). A comparison of the three systems for read- and write-operations in read- or write-intensive environments has been conducted by Cooper et al. (2010) using the YCSB. It shows that, in write-intensive environments, MySQL does not scale well with high throughput. For read-intensive environments, the wide column stores showed better writing performance but were not as good when it came to read-latency. However, all three implementations scaled equally well for read-intensive environments (Cooper et al., 2010).

Lee et al. (2013) evaluated the usefulness of XML and key-value pair databases for medical data. By testing a key-value store and two XML databases, they showed that key-value pairs are much faster and scale better than XML databases. While their performance was almost equal for a

low number of records (1000), key-value store queries ran three to eight times faster than native XML queries for a large number of records (50 000). This was only tested for a thousand to 50 thousand records and no direct numeric comparison to relational databases was made. The recorded times were said to be much faster than on the commonly used RDBMS (Lee et al., 2013).

Hadjigeorgiou (2013) compared a MongoDB document database with a MySQL RDBMS. The speed of simple and '`INNER JOIN`' queries was compared using varying numbers of threads and queries. The comparison showed that, for simple queries, the performance was very similar for the two database systems, with MySQL performing slightly better on average. For '`INNER JOIN`' queries, MongoDB performed a lot better than MySQL. Scaling of the systems over multiple threads could not be observed, as the system seemed overhead dominant and scaled negatively with increasing number of threads (Hadjigeorgiou, 2013).

Based on the literature reviewed, NoSQL databases scale better over multiple threads and therefore perform faster on clusters. This agrees with the observation that these types of database systems are more focused on partition tolerance. The performance of partitioned NoSQL compared to RDBMS running on a high-performance machine and implied costs remain untested. This review also highlights that we have to choose the database type for managing image metadata depending on the required structure of the datasets and choose between different vendors depending on speed and security.

2.1.2.4 Security

The vulnerabilities of hospitals IT and their impact have received much awareness during the recent ransomware attack in May 2017 (Cellan-Jones, 2017; Fox-Brewster, 2017). CT scanners could not be used anymore, and operations had to be cancelled due to the encryption of patient data and of computers forming part of hospital workflows. This highlights the need for IT security in medicine as one of the primary concerns (Montagnat et al., 2004). According to a report by IS Decisions, 22 % of hospitals in the USA allow people to access their network without any login credentials (White, 2015). Independent Security Evaluators (2016) confirmed this in a two-year-long study. The team used infected universal serial bus (USB) sticks and external servers to gain access to the hospital's networks. They were able to demonstrate practical steps that could lead to compromised medical devices and alteration of patient information that could lead to mistreatment and death.

In this thesis we consider details of security implementations out of scope. We do however acknowledge the importance of security for medical applications and therefore will make use of underlying security features of systems we use. Therefore the possibility of implementing security for a specific type of database is important to us.

Database security aims at avoiding confidential and sensitive data to be accessed by unauthorised people. This can happen on purpose or accidentally (e.g. due to permissions set wrong). The main threats to database security are loss of integrity, availability and confidentiality (Basharat et al., 2012; Bertino and Sandhu, 2005). Furthermore, Bertino and Sandhu (2005) considers privacy as a requirement for a secure database. This exceeds confidentiality since it includes

means of obtaining user consents as well as ensuring data usage only for its indicated purpose even after disclosure.

Top threats to IT security according to IMPERVA (2015) and Malik and Patel (2016) are:

1. Excessive and unused privileges
2. Privilege abuse
3. Input injection/SQL injection
4. Malware
5. Weak audit trail
6. Storage media exposure
7. Weak authentication
8. Exploitation of vulnerabilities and misconfigured databases
9. Unmanaged sensitive data
10. Denial of service
11. Limited security expertise and education

Many common security issues are related to poor implementation and maintenance. As highlighted by Osborne (2013), deployment failures, failure to install fixes and unencrypted backends are the most common security vulnerabilities. They are therefore independent of the choice of database. However, while security of SQL databases is well established, security for NoSQL databases is not, as Okman et al. (2011) showed for Cassandra and MongoDB. Overall, SQL databases are still considered the better choice for applications where data security is critical (Chahal et al., 2017; Ron et al., 2016).

2.1.2.5 Databases in Medicine

In medicine, SQL databases have been popular for a long time as the metadata for diagnostic data is very strictly defined (Das and Musen, 1994; Huff et al., 1991; Korenblum et al., 2011; Traina Jr. et al., 2005). This is mainly caused by the strict implementation of the Digital Imaging and Communications in Medicine (DICOM) standard that will be discussed in Section 2.4. For medical data, security is of high importance (Castiglione et al., 2015). Furthermore, medical data is very sensitive and requires consistency and availability. The ACID properties of RDBMS thus make SQL databases the primary choice for medical applications (Castiglione et al., 2015; Wu et al., 2017). Due to their better scalability, NoSQL databases are more suitable for large amounts of data. Celesti et al. (2017) implemented a document database for medical storage based on Cassandra. Ercan and Lane (2014) discussed the use of NoSQL databases for the electronic health record (EHR) of patients. They were not able to find a sufficient argument for or against the use of NoSQL databases and plan to have a pilot study. Jagadish and Olken (2004) went a step further and stated that biological applications need something entirely different

from SQL and NoSQL specifically customized to requirements in life sciences. One of the major requirements they see is the need for similarity queries. Bueno et al. (2002) and Traina Jr. et al. (2005) demonstrated an image storage system that determines the similarity of DICOM images stored. This shows that other database types apart from SQL will be used for medical data in the future.

One common solution as mentioned by Nance et al. (2013) and Wu et al. (2017) is the use of both NoSQL and SQL, where the NoSQL databases store only non-sensitive information. However, security still remains a major concern, independent of the implementation chosen.

Clouds offer an alternative to the set-up of a local server. Storing medical data in a cloud is accompanied by security concerns. To tackle these concerns, one option is to separate confidential and public information over two clouds (Wang et al., 2015). Fabian et al. (2015) went a bit further and considered cloud-providers to be only semi-trustworthy. This approach secures data against possible data collection by the cloud provider through encryption.

For a medical datastore, security is a high priority and NoSQL database types only enable coping with large amounts of datasets, not simply large data. The use of SQL is therefore preferred when designing a storage system for medical images in Chapter 3.

In this section, we reviewed different types of databases for metadata storage. We have discussed different ways of looking at metadata and different database types for storing metadata.

As discussed in Section 1.2, the medical image workflow is not only about the storage but also about the processing of image data. The next section will look at how to process image data, which is managed by the data management system.

2.2 Image Processing

An image is a set of numbers arranged in a two- or higher-dimensional matrix or a discrete multi-dimensional function (Gonzalez and Woods, 2002). Each number represents one measurement or a set of measurements. A medical imaging device such as an MRI records properties of a patient, which can then be represented as numbers in an image. Images can have up to five dimensions. The first three dimensions are commonly the spatial dimensions. The fourth and the fifth dimensions in medical images represent time and colour. Usually, one cell of the image matrix is referred to as a pixel or in 3D images as a voxel. In this thesis, we will use the more common word, pixel, unless we want to clarify that we are explicitly talking about a 3D image. Image processing is a sub-field of signal processing (Gonzalez and Woods, 2002, Chapter 1). Most signals analysed in science are one-dimensional over time. Examples are measurements of acoustics, vibrations, or electrical activity, such as the heartbeat of a patient over time measured by an electrocardiogram (ECG). In such cases, sensors are placed at different locations to collect measurements at a certain rate over an assumed infinite time-frame. Many basic signal-processing algorithms can be extended to higher dimensions to be applied to images. Images are finite and thus assumptions need to be made about infinite continuation of the image.

In this section, we will introduce image processing and survey some of the different aims of image processing as well as algorithms available. Image processing is the procedure of analysing

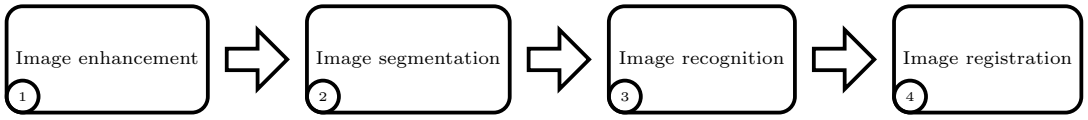


FIGURE 2.6: Image processing workflow: The workflow is commonly split into four parts. Not all of these parts are required in every scenario

or modifying image data. The aim of image processing can be image enhancement, image segmentation, or the extraction of information from the image through image recognition or image registration. These aims are often reliant on each other as shown in Figure 2.6.

2.2.1 Image Enhancement

Image enhancement refers to the improvement of the image pixels to result in an image that is easier to process further or just visually more 'natural' to the viewer. A common example is a histogram equalisation filter, which balances the contrast and brightness of an image in such a way that it 'looks better' to the human eye (Nixon and Aguado, 2012, Section 3.3.3). Image enhancement filters can help improve the signal-to-noise ratio or be used to remove image blurring (Sheppard et al., 2004). Most importantly, they assist in normalising images for further processing steps.

2.2.2 Image Segmentation

Once the images are normalised, they can be segmented. Image segmentation assigns labels to each pixel so that features or structures of interest can be highlighted and analysed. There are several algorithms for the detection and classification of pixels of interest. Section 2.2.6 will give an overview of algorithms that are widely used for image segmentation. Image segmentation results in one or more masks. A mask is a binary image where all the pixels that belong to the same segmentation label will have the same value.

In medicine, image segmentation finds application in organ detection (Cuingnet et al., 2012) as well as detection of all kinds of generic or specific structures of interest (e.g. Bae et al., 2005; Sato et al., 1998).

2.2.3 Image Recognition

The masks created by image segmentation as well as the pixel data of a raw or normalised image can be used to recognise quantitative information inside an image. Such information describes features, such as metadata, or shapes included in an image (Nixon and Aguado, 2012, Chapter 7). Image recognition includes shape recognition and object recognition (Hong, 1991). For example, one can detect a circle in an image and measure its diameter using shape recognition (Nixon and Aguado, 2012, Chapter 7). This has for example been done by Bloem et al. (1995) for automatic cell detection and volume measurement.

Image recognition requires image segmentation to highlight areas of potential interest. It then uses algorithms to determine the properties of these areas. Image segmentation will return a binary mask which requires much less memory than a 32 bit image. Examples of image recognition are machine learning algorithms (see Section 2.2.6.3) that combine image segmentation and recognition.

2.2.4 Image Registration

Image registration enables the alignment of two or more images from the same object or scene (Zitová and Flusser, 2003). It needs to take into account translation, rotation, scaling and shear as well as projective and curved transformations between two images on a global or local scale (Brown, 1992; Maintz and Viergever, 1998). Examples are the creation of panorama photos (Deng and Zhang, 2003; Shum and Szeliski, 1998) or 3D scanning of objects (Levoy et al., 2000; Tong et al., 2012). In medicine, image registration is used to align images from different imaging modalities to combine their information (Brown, 1992; Maintz and Viergever, 1998) or to map image data to a model (Maintz and Viergever, 1998). Maintz and Viergever (1998) and Oliveira and Tavares (2014) give many examples of different medical use cases and related research projects. An example is patient-to-modality registration, in which a probe is used to create a fixed reference between the patient and the image of the patient. The reference later aids the surgeon during an operation by allowing a comparison between the actual patient and the location of organs and features of interest from the images taken (Maintz and Viergever, 1998). Image registration often requires manually or automatically segmented markers. The challenges of aligning large medical images will not be tackled as part of this thesis since the other aims (image enhancement and pixel classification) are deemed more important, as they form the foundation of image processing, enabling further research in the future.

2.2.5 The Image Processing Workflow

In many cases, several of the aims of image processing presented in Sections 2.2.1 to 2.2.4 are combined. In the case of facial recognition (Bledsoe, 1964), image enhancement ensures that photos taken in different environments and different light levels are balanced out. Afterwards, the face is detected and different parts of the face are labelled through image segmentation. Based on these labels, different features of the face can be recognised and measured, such as the roundness of the head or the relation between the width of the nose and mouth.

The next section will look more specifically at image processing algorithms, which are used to achieve enhancement or segmentation. Image enhancement and segmentation will take an image and produce one or more resulting images. Image recognition and image registration use similar techniques but follow a different procedure. Image recognition will produce non-image data, and image registration takes more than one image as the input.

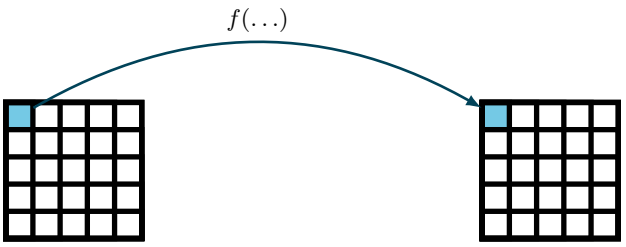


FIGURE 2.7: Simplified illustration of an in-place filter. The value of a pixel is modified according to a given function. New pixel values are only affected by the old value of that pixel. They can therefore be computed in one place in memory, overwriting the old values as the new ones are being computed.

2.2.6 Image Processing Algorithms

Many processing algorithms have been created to tackle the challenges described in Sections 2.2.1 to 2.2.4. Most of these are targeted at image enhancement or segmentation. Various options are available to differentiate between image processing algorithms (Gonzalez and Woods, 2002; Kalinić, 2009; Pham et al., 2000; Robb, 2000) and here we will classify them into three categories: filter-based algorithms, iterative algorithms and machine learning algorithms. Filter-based algorithms use the value of a pixel and its neighbours to classify pixels into segmentation regions. Filters range from histogram operations involving only one pixel to frequency filters that apply a domain transform before processing the pixel data. Iterative algorithms grow, shrink or adapt regions or seeds that have been obtained automatically or manually to separate different regions in an image. A region growing from manual markers is one such algorithm. Active contour models are a more elaborate version of iterative algorithms. Machine learning is a separate approach to the programming of the image processing. It usually relies on filter-based algorithms, but adds another layer of interpretation to combine results of different image processing algorithms.

2.2.6.1 Filter-Based Algorithms

Image processing filters have in common the fact that they require a fixed number of steps and computations to complete. The number of computations required scales with the size of the image and is not related to the image content. We categorize them, loosely following (Robb, 2000, Chapter 5), into in-place filters, spatial filters, and frequency filters.

- In-place image filters, such as thresholding and histogram operations, are the most simplistic image processing filters. Each pixel is transformed individually based on its value and, in the case of a histogram operation, based on global properties of the image as shown in Figure 2.7 (Pham et al., 2000; Robb, 2000). The histogram is a count of pixels with a specific value, where groups of values are commonly formed for images with more than 8 bit to create 256 equally spaced value groups (Nixon and Aguado, 2012, Chapter 3).
- Spatial filters are based on group operations. They involve a large group of different filters. Such filters can be implemented as a template that is convoluted with the image matrix (Nixon and Aguado, 2012, Chapter 3). Such a convolution can also be implemented in the frequency domain as explained in the next paragraph. Figure 2.8 illustrates how a template is used to filter an image. The template is applied to every pixel but stretches over that

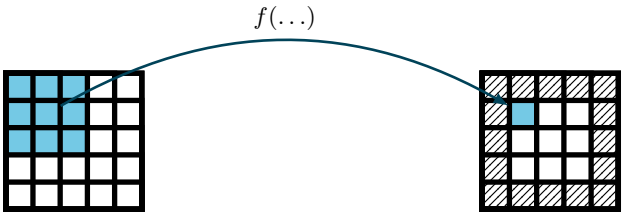


FIGURE 2.8: Simplified illustration of a spatial filter. Values from a pixel and its neighbour pixels affect the value of the pixel at that position in the output. Since pixels at the border do not have sufficient neighbours, the values at the border (shaded) cannot be computed.

pixel and its neighbouring pixels. The template can be larger, thus including neighbours that are further away. The values of the pixel and its neighbours are multiplied by the corresponding values in the template. The result is added up, and this sum is the new value of that pixel in the output image. Since the filter involves neighbouring pixel values, an output image is needed, and pixels cannot be calculated in-place. Further, pixel values at the outer edge of the image cannot be calculated since they do not have neighbours. They are either approximated by assuming a repeating image or pixel values are set to fixed values around the edges. This means that the larger the template is, or the higher the number of template convolutions calculated is, the smaller is the size of the output image. This reduction in image size will become important later in Section 4.2.2. The example from Figure 2.8 shows the implementation of a spatial filter. Common spatial filters involve edge-detection filters, such as the first-order Sobel filter or the second-order Laplacian filter.

- Frequency filtering involves the transformation of an image into a frequency domain using Fourier transform or other methods such as the cosine transform, Hartley transform or wavelets (Nixon and Aguado, 2012, Chapter 2.7). As mentioned earlier, an image is just a matrix of numbers and as such can be viewed as a discrete 2D function. Since an image is finite in size, but frequency transforms operate on infinite data, an assumption has to be made about the extension of the image into infinity. To achieve this, the image is usually mirrored at its sides to form an infinite pattern. The 2D discrete Fourier transform is separable and symmetric. This allows the function to be applied to single rows and columns of an image sequentially (Nixon and Aguado, 2012, Chapter 2). Figure 2.9 illustrates how a frequency filter works. The frequency domain can also be used to apply spatial convolution operators by multiplication of the Fourier transform of the image and the filter. For kernel sizes of 60 or larger, this is considerably faster than convolution in the spatial domain (Fialka and Čadík, 2006; Smith, 1997, Chapter 18).

2.2.6.2 Iterative Algorithms

Iterative approaches are based on an initial estimate, which is improved over a fixed number of iterations or until an equilibrium is found. Some of these approaches, such as the anisotropic diffusion filter, are used for image enhancement, but most of the ones presented here target the segmentation of an image into separate regions. The basic implementation of an iterative algorithm is illustrated in Figure 2.10. The following points discuss widely applied iterative approaches.

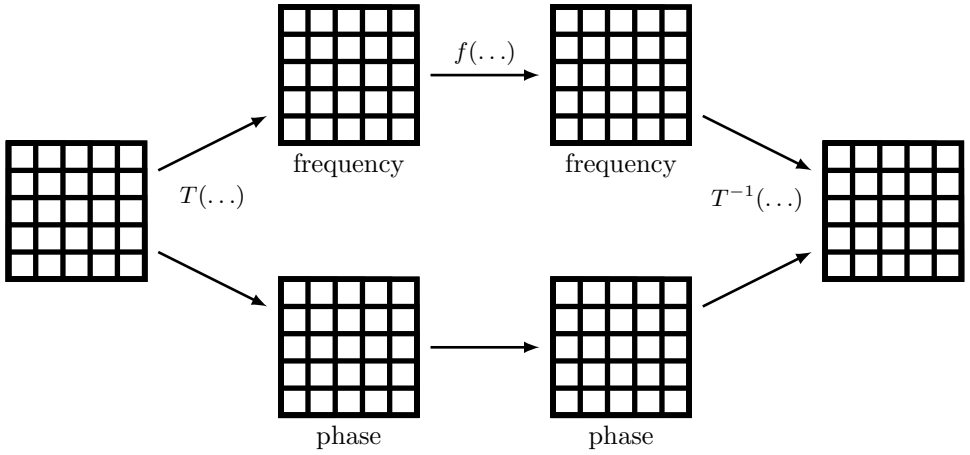


FIGURE 2.9: Simplified illustration of a frequency filter. A discrete transform, T, is applied to convert the image into the frequency domain. The frequency and potentially the phase are changed according to the filter's rules. The result is then converted back into the spacial domain using the inverse transform.

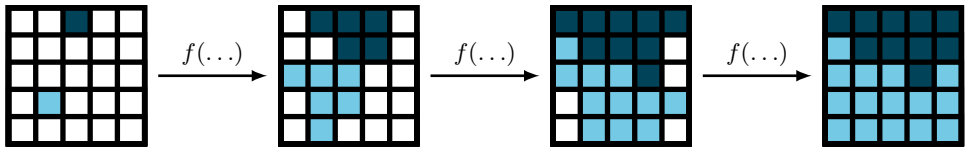


FIGURE 2.10: Simplified illustration of a region growing algorithm. Seeds are determined automatically or introduced by manual input. The algorithm will then apply the same rules to converge from the seeds to a final segmentation or image manipulation over several iterations. This representation is illustrating region-growing or watershed algorithms. Deformable model algorithms work slightly differently, as shown in Figure 2.11.

- Anisotropic diffusion uses the image gradient to compute a diffusion tensor, which drives the diffusion of pixel values over several iterations (see Appendix C.11). The idea is to smooth everything but the edges (Weickert, 1998). The diffusion results in a loss of sharp edge information. To counteract this, a higher number of iterations requires an adaptation of the gradient threshold (e.g. Tsiotsios and Petrou, 2012). Anisotropic diffusion is beneficial for de-noising CT images (Sheppard et al., 2004). Several filters are similar to anisotropic diffusion. They are also referred to as partial differential equation (PDE) filters (Weickert, 1998).

- The connected-regions algorithm checks neighbours of a pixel and, if they have the same value, assigns them to the same group. It continues to group pixels in this way till the whole image is grouped. If the image is binary, this image can automatically detect all white regions. Otherwise, it requires a marker to start from and selects all pixels in the surrounding with the same or a similar value.

- Region-growing algorithms start from a seed and then grow in all directions based on the difference between the neighbouring pixel and the original seed or the boundary (gradient) (Pham et al., 2000). This is also possible for a shrinking algorithm, where an area is selected and reduced until a boundary is hit. In such a case, it is also referred to as lasso or region shrinking.

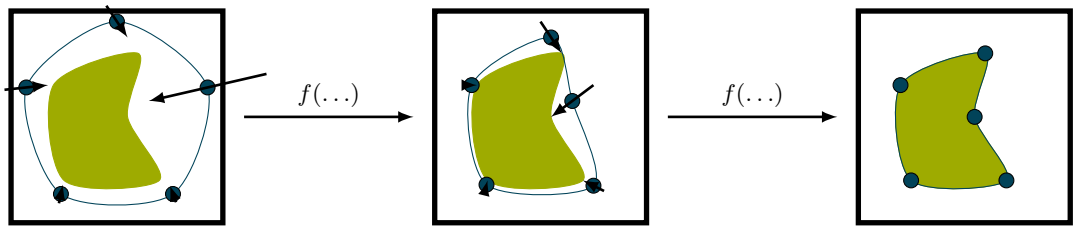


FIGURE 2.11: Simplified illustration of a deformable model algorithm. The dark curve is fitted to the light rectangular shape in the image over several iterations.

- The watershed filter is a special case involving region growing. In the first step, it finds the local minima in the image matrix. Alternatively, it can use pre-defined markers to reduce noise (Vincent and Soille, 1991). It then expands regions from the markers, taking the gradient between pixels into account. When two areas meet at a point, a boundary is created at that point. After filling the whole image, boundaries become the segmentation borders. The watershed algorithm is a very widely used image segmentation method. It has been applied to lung segmentation (Shojaii et al., 2005) and liver segmentation (Lim et al., 2004) from CT scans. In general, a marker-based version is used to filter out whole organs rather than small structures. The placement of user-defined markers itself is not an issue since it is not very error-prone as demonstrated by Adams and Bischof (1994).

- Deformable models, also referred to as active contour models (Albrecht et al., 2009), use one or several curves as a starting point and calculate velocities to adjust that curve to match a shape in an image. The curve is varied (see Figure 2.11) until the model fits the shape in the image with the least amount of 'energy' when modelling the image as a force field (Albrecht et al., 2009; Kalinić, 2009). The energy minimization relies on the variation of neighbouring pixels. The most popular approach are snakes, a local model for finding objects (Kass et al., 1988; Wittman, 2014). Active contour models have found application in segmentation of images of organs obtained by CT (e.g. He et al., 2017; Rebouças Filho et al., 2017).

2.2.6.3 Machine Learning Algorithms

Machine learning-based approaches are known by many different names. 'Machine learning' is the more generic term, now often replaced by the term 'artificial neural networks' (Torbati et al., 2014). In the field of image processing, these algorithms are often referred to as map-based or atlas-based approaches (Kalinić, 2009) or sometimes simply classifier-based approaches (Pham et al., 2000). Although all of these terms are different, most authors do not make a clear distinction between them. Since machine learning is the area of study that includes approaches such as neural networks and atlas- or classifier-based approaches, we will refer to them as 'machine learning' in this thesis. This section will give a brief overview of different types of machine learning with a focus on the random forest algorithm that will be used later in this thesis in Section 4.2.3.

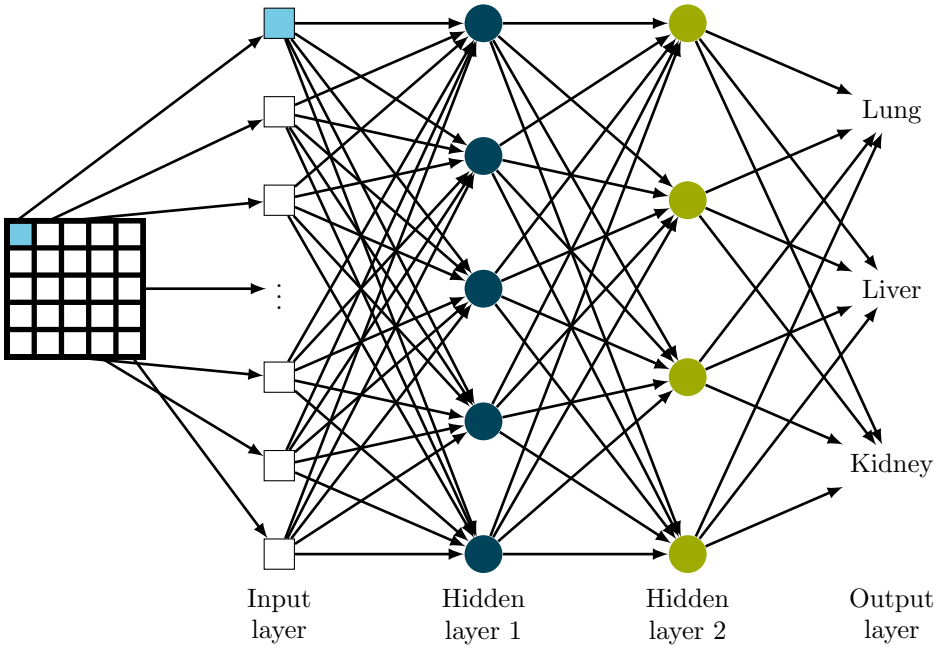


FIGURE 2.12: Simplified illustration of a neural network machine learning algorithm. Each pixel of the image forms part of the input layer. Weights are applied to calculate nodes in one or more hidden layers. Finally the nodes are used to calculate the output layer. In this scenario each node of the output layer corresponds to a different organ that could be shown in the image.

There are three types of machine learning: supervised, unsupervised and reinforcement machine learning. Supervised machine learning requires training data to tune the algorithm. Unsupervised machine learning finds patterns in datasets and can be used for clustering. Reinforcement machine learning is trained based on a reward system (Fumo, 2017; Patel, 2018).

The most well-known machine learning algorithms at the moment are neural networks, shown in Figure 2.12. For image processing, there are two common machine learning techniques: convolutional neural networks (CNNs) and the random forest algorithm (Brownlee, 2018; Litjens et al., 2017, e.g.).

CNNs differ from regular neural networks (see Figure 2.12) in that the neurons from one layer in the CNN are connected to only part of the neurons in the next layer (Cornelisse, 2018). In a CNN, several convolutions are performed on the input image as done for spatial filters (see Section 2.2.6.1). The results are assembled as a 3D matrix. After this convolution layer, a pooling or sub-sampling layer is added to reduce the number of parameters. Several convolution and pooling layers can be added until the final output is a single vector of probability scores (Cornelisse, 2018; Lawrence et al., 1997).

A random forest classifier consists of many decision trees as shown in Figure 2.13. Each decision tree splits on a random feature at each node. In image segmentation, different filters, as presented in the previous sections, can be applied to generate different features for the decision trees. For classification, the outcome of each tree is taken and the most common outcome from all the trees is chosen as the outcome of the forest. With increasing number of trees, the accuracy increases and converges. Strong individual trees as well as a low correlation between the trees results in a small generalisation error of a random forest (Breiman, 2001).

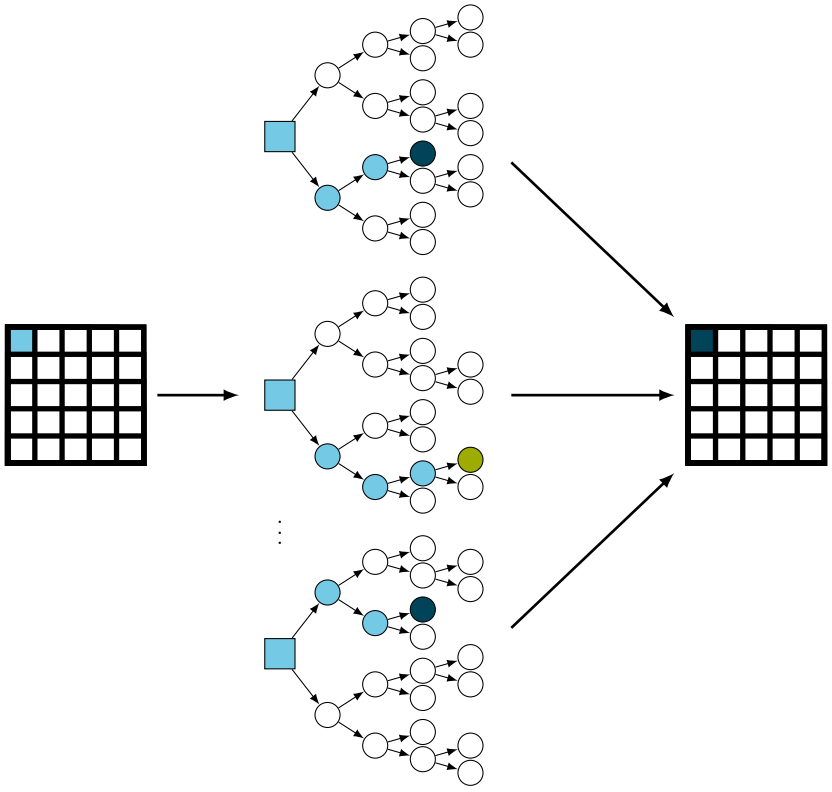


FIGURE 2.13: Simplified illustration of a random forest machine learning algorithm as implemented by the Trainable WEKA Segmentation (TWS). Each pixel is tested against a set of decision trees. The result of all decision trees is compared and the most common outcome is chosen.

If supplied with known segmentation results as training data, the machine learning algorithm creates a map (or atlas or decision tree) that tells it how to classify a newly presented pixel. Some approaches are untrained and thus try to create clusters of pixels that seem to be similar based on the applied processing methods (Pham et al., 2000).

2.2.7 Approaches to Process Large Images

In order to process an image using one of the processing methods from Section 2.2.6, most software will load the whole image into the computer memory and then apply the method to generate an output image. For large images, this requires a lot of RAM and several approaches have been developed to avoid having to use more powerful RAM, central processing unit (CPU) or graphical processing unit (GPU).

One approach is the outsourcing of work to the cloud as done by cytomine[1]. Renting a powerful computer for the time period it is needed can be more cost-efficient than buying a powerful computer that sits idle most of the time. ImageJ (Rueden et al., 2017) and QuPath (Bankhead et al., 2017) enable loading a single image slice at a time and processing them sequentially, greatly reducing the memory requirements for images that are large in the 3rd dimension. This method does not provide any benefits to images that are large in the 2nd dimension and it prevents the application of a 3D image filter. A solution to dealing with images large in the 2nd dimension

[1]Official site: https://cytomine.coop/ (last accessed 23/11/2018)

is the use of 2D tiles as done by Graphics Mill[2]. Matlab[3] takes this approach even further by loading a low-resolution image first to enable the preview of complicated image processing filters before applying the filter to the whole image. Both applications are restricted in the use of 3D filters. The last approach we found in the literature is the use of custom functions that enable processing methods to be automatically pipelined. This has been successfully implemented in the VASARI Image Processing System (VIPS) (Cupitt and Martinez, 1996).

The approaches found are limited in their application and cannot be scaled to large 3D images or require custom processing algorithms to be programmed. In Chapter 4, we will propose a new approach to processing large multi-dimensional images.

As stated at the beginning of this section, digital images are a collection of numbers, making them ideal for processing with a computer. Without visualisation, an image just remains a collection of numbers and is difficult to interpret by humans. When we look at an image, we expect to see something that represents the object or scene the image is of. The next section will look at the visualisation of images, converting them from numbers on a hard-drive into light received by our eyes.

2.3 Established and Upcoming Visualisation Techniques

Image visualisation has come a long way, and while images had to be drawn by hand in the past, computers and digital monitors are now widely used (Blakstad, 2011). With computers improving in calculation power and the development of new manufacturing and visualisation devices, the presentation of images has developed into a multitude of new areas, which we will cover in this section. While the first two sections will focus on more traditional display using a monitor, the other sections will focus on new technologies which are only beginning to make their way into medicine.

2.3.1 2D Sections

The most common way to look at images created by a CT scanner is through 2D sections. Robb (2000) gives several examples of different 2D visualisation methods, the most basic one being multi-planar reformatting. *Multi-planar reformatting* shows slices through the 3D image for two to three different orthogonal orientations. This is still the most important way of visualising CT data and vendors such as General Electric Healthcare include a multi-planar viewer when DICOM images are burned from their systems to a compact disc (CD). Alternative 2D visualisations, such as oblique sectioning and curved sectioning have been created and are commonly used (Robb, 2000). *Oblique sectioning* essentially allows on-the-fly reformatting at any angle, making rotation around a point in the dataset possible. *Curved sectioning* goes a bit further and slices the image not along a straight line but along a pre-defined path. This can be useful for investigating the spinal canal or the intestine (Pickhardt et al., 2003; Robb, 2000). The reason that 2D image viewing is so widely applied and plays an important role in medicine is that doctors have used these images for decades. Doctors have thus become experts in 2D image viewing. The

[2]Official site: https://www.graphicsmill.com/ (last accessed 23/11/2018)
[3]Official site: https://uk.mathworks.com/products/matlab.html (last accessed 23/11/2018)

(a) 2D section display (b) 3D volume rendering (c) 3D surface rendering

FIGURE 2.14: Comparison of established visualisation methods on a computer monitor.

importance of 2D sections is reflected in the widespread use of the imaging method known as CT or 'computed tomography'. A tomography is a 2D graphical cut (Robb, 2000, Chapter 2).

2.3.2 3D Rendering

The downside of 2D visualisation is that real-life data is 3D and therefore only part of the data can be shown at a time. 3D visualisation gives a better overview of an image, allowing to estimate distances and relations between objects in 3D space (Hansen and Johnson, 2004). The problem with 3D views is that we use 2D displays to look at them. This means that we can only look at a shadow of a 3D object and try to create an illusion tricking our brain into thinking it is 3D (Hansen and Johnson, 2004). We will, therefore, refer to this visualisation technique as 3D rendering or projected 3D in this thesis.

The two primary methods used for 3D rendering are volume rendering and surface rendering, as shown in Figure 2.14. *Volume rendering* takes the 3D voxels as they are and renders them in a virtual 3D space, assigning them different transparency based on their values. For example, denser voxels are assigned a lower transparency, to make dense objects become visible in 3D. The use of surface rendering requires the definition of surfaces in an image. Imaging devices such as CT or MRI look at slices through the body. This means that information exists about every single point between the source and the detector. This information is usually reconstructed into Cartesian coordinates but remains as a block of values. A surface, however, is a collection of points in space which are connected through edges and faces. Creating a *surface render* requires defining which points in the 3D image block are part of the surface. This can be achieved through image segmentation, for example, as described in Section 2.2.2. Surface views allow clearer representations of objects of interest. Example uses of 3D rendering in medicine are in virtual colonoscopy (e.g. Pickhardt et al., 2003) and surgery planning (e.g. Ribeiro et al., 2015).

As 3D rendering only shows a 2D render of a 3D representation, it is important how the user interacts with the 3D image to ensure correct interpretation. The most simplistic 3D render can be a static image. Interactive 3D viewers are available: they allow the user to use a mouse and keyboard to rotate and zoom in and out of the object. 3D rendering can also be combined with techniques involving 3D glasses. The idea is to create two images of the 3D object as they would look like from our right and left eyes. Then, different filtered glasses can be used to ensure

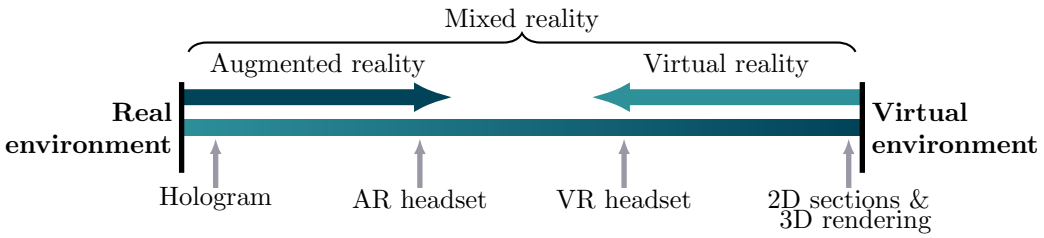


FIGURE 2.15: Mixed reality: Augmented reality and holograms place a digital object into the physical reality; virtual reality replicates physical behaviour in a digital reality (Bray and Zeller, 2018).

that the left eye only sees the projection for the left eye and the right eye its corresponding one. 3D-rendered images can also be used in combination with haptic devices (e.g. Wiet et al., 2002). Several companies propose the use of haptic devices for surgical training based on 3D images, often modelled and combined with material properties for haptic feedback.

Sections 2.3.3 and 2.3.4 will discuss different means of interaction which go beyond viewing 3D images on a 2D monitor. These modern visualisation techniques involve holograms, virtual reality (VR) and 3D printing.

2.3.3 Mixed Reality

The term mixed reality has been created by Milgram and Kishino (1994). It includes devices that render digital objects into the physical world, as well as those that add input or data from the physical world into a virtual environment as shown in Figure 2.15 (Bray and Zeller, 2018). We will discuss the different types of mixed reality in the following sections. We will differentiate between augmented reality, virtual reality and holograms.

2.3.3.1 Holograms

Holograms are projected images that trick the eye into thinking they are real 3D objects. There are different types of holograms: Işık (2014) classified them as transmission holograms and reflection holograms. *Transmission holograms* rely on a light source passing through the recording medium, similar to a diascope/slide projector. *Reflection holograms* can be viewed by shining a light source from the same side from which the hologram is viewed, similar to a traditional photograph. Holograms range in their complexity. Some holograms require a specific type of light source to be viewed. Other holograms can be viewed from any angle and present a true colour image or even a video (Işık, 2014; Tay et al., 2008). The advantage of holograms is the possibility of representing objects in 3D without the need for the viewer to wear special equipment. This gives holograms an advantage over 3D glasses (Section 2.3.2) and other techniques (Sections 2.3.3.2 and 2.3.3.3). The latest research on free-space volumetric displays enables the viewing of 3D images from any angle. Although the output is similar to that of holograms, the underlying technology is different (Smalley et al., 2018).

Companies such as Echopixel[4] develop medical holographic images, that can be particularly useful in teaching environments. Moreover, research sees a high value in the use of holograms as a tool for teaching medical students (Perozo et al., 2016).

2.3.3.2 Augmented Reality

Projecting virtual objects in a real environment, referred to as augmented reality (AR), has opened up a lot of possibilities in medicine. Fuchs et al. (1998) experimented with the use of AR for laparoscopic surgeries almost two decades ago. Around the same time, Weghorst (1997) investigated the use of AR to help people with Parkinson's disease walk across open plan areas by placing virtual objects along the way for orientation and Azuma (1997) proposed the use of AR to aid surgeons during an operation by overlaying scan data with the patient. Still, no common practical application of AR exists in medicine yet as the market for AR is still relatively young. Google Glass[5] and Microsoft HoloLens[6] are recent examples of commercial AR devices. In this thesis we will use the term '*AR*' meaning augmented reality based on headsets or mobile devices. Holograms are a type of AR but significantly different in their technology and we therefore define it as a separate category.

2.3.3.3 Virtual Reality

The day of birth of virtual reality (VR), 6th of June 1989, is termed the 'VR Day' (Bricken, 1990). VR soon became the key topic of the computer graphics conference SIGGRAPH (Haggerty, 1991). Jaron Lanier, was among the famous personalities at that time who made the term 'virtual reality' popular. Even though the term originates from work carried out by Antonin Artaud, Lanier was the one to connect it to a technology (Karaliotas, 2011; Steuer, 1992). The idea of virtual reality was often combined with that of augmented reality, introduced in Section 2.3.3.2. Whilst augmented reality combines virtual objects and the real world, VR immerses the user into a purely virtual world.

Very soon after VR became popular, potential applications for it in medicine were found (Rosen et al., 1996). Today, applications range from the training of surgeons (Thomsen et al., 2017) to the treatment of mental health problems (Freeman et al., 2017). This new technology is limited in its application and does not always improve the performance of surgeons (Phé et al., 2017). VR has reached the gaming industry with several head-wear products such as PlayStation VR[7] or Oculus Rift[8], which will likely push the technological advances in this area. VR remains an important part of visualisation and will find more applications in the future.

2.3.4 3D Printing

Additive manufacturing, more commonly known as 3D printing, allows the production of complicated shapes by creating them layer-by-layer. Since it does not require the creation of moulds

[4]Official Site: http://www.echopixeltech.com/ (last accessed 22/06/2017)
[5]Official Site: https://www.google.com/glass/start/ (last accessed 13/02/2018)
[6]Official Site: https://www.microsoft.com/en-gb/hololens (last accessed 13/02/2018)
[7]Official Site: https://www.playstation.com/playstation-vr/ (last accessed 13/02/2018)
[8]Official Site: https://www.oculus.com (last accessed 13/02/2018)

or other factors that increase the set-up cost and time, it is also referred to as rapid prototyping (Huang et al., 2013). While commonly viewed as a manufacturing method, additive manufacturing can also be used for visualisation. We will explore the application of 3D printing as a visualisation method further in Section 5.2.3 and Section 5.3.

A large variety of techniques for additive manufacturing exist, an overview for which is given in Table 2.2 (Bikas et al., 2016; Gibson et al., 2010; Gu, 2015; Huang et al., 2013). Most techniques use an energy source, such as a heater or a laser, to change the aggregate of the material to print and cause 'sticking' of the material in desired locations. Different additive manufacturing techniques are usually named after the type of energy they use and the way they achieve the binding of the material. In this thesis, we will refer to the whole of additive manufacturing as 3D printing for simplicity.

Creating a physical replica using 3D printing is becoming more popular in medicine (Adams et al., 2017; Suzuki et al., 2007). With more advanced materials becoming available, even whole 3D-printed human replicas have been created (BBC, 2016; Nottingham Trent University, 2016). The main advantage of 3D printing over other manufacturing and visualisation methods is the high model complexity that can be produced. However, 3D printing comes with a high cost per model and slow speed for generating these models (Rengier et al., 2010). FDM is the most common 3D printing method at the moment due to its low cost. Other methods such as SLA start to become more affordable for low-cost printing. The main challenges of rapid prototyping in medicine are the application to clinical routines and the proof that they have a real benefit over other visualisation methods, as well as adequate resemblance to real tissue (Rengier et al., 2010). A possible application proposed by Rengier et al. (2010) is the use of 3D printed models in a teaching environment.

The last section of this chapter will look at digital standards in medicine. These define a variety of aspects important to the implementation of the image management, processing and visualisation in medicine.

2.4 DICOM—The Medical Standard

In 1985, the American College of Radiology (ACR) and the National Electrical Manufacturers Association (NEMA) jointly published a standard for the transfer of medical images and associated metadata. This standard evolved to the now well-known DICOM standard (NEMA, 2014a,d). The standard (NEMA, 2014a) with over 5500 pages ensures compatibility between various hardware and software vendors across different hospitals. DICOM regulates file storage, file transfer and image display. The recently added DICOMweb standard also regulates web-based systems (Genereaux et al., 2018).

2.4.1 DICOM File Format

DICOM files function almost like a wrapper around the commonly used image files, adding information relevant to the medical community. As such, each file must explicitly define a patient by unique properties, such as their name, identification number, and date of birth. In

TABLE 2.2: Overview of various additive manufacturing techniques (Bikas et al., 2016; Gibson et al., 2010; Gu, 2015; Huang et al., 2013).

Method				**Name**	**Material**	**First Publication**
Liquid	hardened	by ultraviolet (UV) light	(liquid in a vat hardened at required positions)	Vat photopolymerisation	thermoset photopolymers	Hull (1986)
			(liquid positioned as drops at required positions)	Material jetting (MJ)	thermoset photopolymers	Thomas et al. (1997)
		by laser		Stereolithography apparatus (SLA)	thermoset polymers	Kodama (1981)
Filament spool	melted	by heater		Fused deposition modelling (FDM)	thermoplastics polymers	Crump (1992)
Powder	fused	by binding agent		Binder jetting (BJ)	metals, sand, ceramics	Sachs et al. (1992)
		by fusing agent and infrared		Multi jet fusion (MJF)	thermoplastic polymers	Nielsen and Kasperchik (2004)
	sintered	by heater		Selective heat sintering (SHS)	thermoplastic polymers	Hartmann and Tjellesen (2012)
		by laser		Selective laser sintering (SLS)	thermoplastic polymers	Beaman and Deckard (1990)
				Direct metal laser sintering (DMLS)	metal alloys	Deckard (1988)
	melted	by laser		Selective laser melting (SLM)	metals	Greulich et al. (1995) and Agarwala et al. (1995)
		by electron beam		Electron beam melting (EBM)	metals	Larson (1998)
		by laser or electron beam	(on existing structure)	Directed energy deposition (DED)	metals	Voelkel and Mazumder (1990)
Laminate	fused	by heater	(cut to shape by laser)	Laminated object manufacturing (LOM)	paper, plastic or metal laminates	Feygin (1987)
	welded	by ultrasound	(cut to shape by drill)	Ultrasonic additive manufacturing (UAM)	metals	White (2003)

this way, it is always possible to map the images back to the patients to ensure that doctors have access to the relevant images for their diagnosis.

2.4.2 DICOM File Transfer Protocol

The DICOM file transfer protocol enables communication between various medical equipment and software. For a storage commitment (SCM) an image is sent from a service class user (SCU) to a service class provider (SCP), as described in (NEMA, 2014e, Chapter 9.2). We will make use of this protocol in Section 3.1.2.

2.4.3 DICOM Image Display

Chapter 14 of the DICOM standard describes 'a standardised display function for display of grayscale images' (NEMA, 2014b). As discussed in Section 2.2, a digital image is a set of numbers and needs to be converted into luminance levels to become a viewable image. In medicine, it is considered very important to ensure that the same image always looks the same, independent of the display device or environment. In this way, information necessary for diagnosis will always look the same, and there is less room for errors due to wrong interpretation of the image (Barten, 1999; NEMA, 2014b).

The defined DICOM grayscale standard display function (GSDF) allows creating a look-up table (LUT) for individual images, as reported by Jones (2006). It can also be used by vendors to record the output of a monitor with a camera and compare it with the expected image output. The comparison is used to calibrate the intensity of individual pixels of the image based on current light conditions (Compton and Oosterwijk, 2012).

2.4.4 DICOM Data Model

The previous three areas of the DICOM standard show the importance of complying with DICOM when designing a medical workflow. By ensuring consistent data storage and transfer, hospitals can purchase equipment from different vendors and rely on them working together. And if every medical display shows the images in the same way, then doctors do not have to adapt and the chance of misdiagnosis is reduced. DICOM is designed for clinical/diagnostic purposes, which becomes apparent when looking at the 'DICOM data model'.

According to DICOM, data is structured as shown in Figure 2.16 (NEMA, 2014c). The most important information for any data is the patient. From a diagnostic point of view, this makes sense. The patient is the subject to be treated and information related to the patient needs to be known. A doctor can conduct a study for a patient (for example an MRI scan) that involves one or more series of images.

From a research perspective, this approach is not beneficial. First of all, in research, it is important that the patient is not identified by the research data. There must not be any identifier for the patient. Instead, it is often of interest to compare studies from different patients and find things that relate them. For example, this allows predicting risks for groups of people, such as

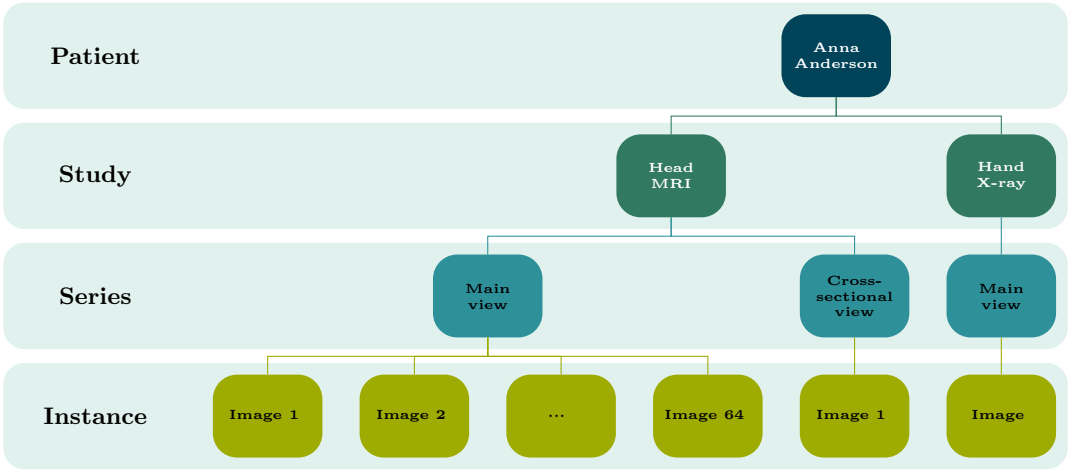


FIGURE 2.16: The DICOM model: the labels on the left shows the structure according to the standard, the tree diagram on the right shows an example of how image data fits into this model.

smokers or men above the age of 60. The anonymisation of data can be done through encryption or removal of personal data. Usually the identifying data from DICOM files is removed which makes the DICOM model useless in a research environment.

2.5 Image Management Systems

The use of databases, as introduced in Section 2.1.2, to store and retrieve data is vital to biological and medical sciences in dealing with the increasing flood of data (Jagadish and Olken, 2004). Systems for managing image data and related metadata in medicine are commonly known as picture archiving and communication systems (PACSs) in a radiology environment (Choplin et al., 1992). Within a digital pathology system (DPS), they are also referred to as LIMS or just laboratory information system (LIS) (Chlipala et al., 2010; Pantanowitz, 2010). We will use '*image management system*' as a more generic term throughout this thesis.

Griffin and Treanor (2017) encouraged the application of a digital pathology workflow since it allows quick sharing of data for assessment by multiple experts. The multiple assessment results in fewer errors. Moreover, the use of automated image analysis aids the experts in their diagnosis as commonly done through CAD. In a cost-benefit analysis, Griffin and Treanor (2017) showed that a 10 % productivity increase would result in breaking even after just two years for a large hospital. This shows the feasibility of digital workflows. Their article also lists several downsides of a digital pathology workflow, such as the unfamiliarity of pathologists with digital slides, network latency and the related perceived inefficiency. Possible reasons for this inefficiency are a small field of view, software design issues and unfamiliarity. This shows that usability of medical software and viewing options are important to the success of a digital pathology system (Griffin and Treanor, 2017; Treanor and Quirke, 2007).

The use of image management systems based on the European DataGrid Testbed has been previously investigated (Montagnat et al., 2004). The authors define the image management

system to include image processing. In this thesis, we clearly distinguish between image management and image processing (see Section 1.2). For these two systems, they identify relevant specific areas of importance, including data security, data semantics, traceability, computation pipelining, parallel computations and interactive applications with a focus on responsiveness and visualisation. They further envision ease of access as a future aim (Montagnat et al., 2004).

PACSs store images on a server and make them available to radiologists, allowing them to search through the data and view the images in the browser. In Sections 2.5.1 and 2.5.2, we will review the most popular image management systems in medical research. The Open Microscopy Environment Remote Object platform (OMERO) and the Bio-Image Semantic Query User Environment (BisQue) are free web-based image-servers that are providing solutions for the increasing image size. More of such systems exist (eg. Linkert et al., 2010; Marcus et al., 2007; Norcen et al., 2003) but none of them has the popularity and impact OMERO and BisQue have. We will also introduce a system that presents an alternative way of managing data to existing medical systems in Section 2.5.3.

2.5.1 OMERO

The Open Microscopy Environment Remote Object platform (OMERO)[9] (originally OME-Server) is an image repository dating back to 2000 that is used for medical research data (Allan et al., 2012; University of Dundee, 2014). It is maintained by the University of Dundee and the Open Microscopy Environment. An overview of the structure of OMERO is given in Figure 2.17. As seen in the figure, OMERO is more than an image storage system. As a PACS alternative, it combines image management, processing and visualisation in one system. We will therefore have a look at all the requirements for an image workflow described in Section 1.4 when reviewing OMERO in this section.

OMERO was designed for small 2D images (Kvilekval et al., 2010) but now also supports the storage of large multi-dimensional images. Metadata is stored in a relational database. Most of its services are supported by a browser-external software client (OMERO.insight), which connects to the OMERO server. The server holds the data and metadata, which the client can access remotely (Burel et al., 2015). Images can also be stored and manipulated through customized codes written by users. Such codes need to be compatible with the OMERO application programming interface (API) (Allan et al., 2012). This feature allows for great usability by programmers. Users with a medical focus can rely on a multitude of plug-ins for popular software, such as the OMERO plug-in[10] for ImageJ and the OMERO.matlab toolbox[11]. The support for new software requires a programmer to create a new plug-in for the medical user. The use of a custom server limits the modularity of the system. Since all parts need to be compatible with the OMERO API, the integration of third-party applications requires adjustment by a programmer familiar with OMERO. Data traceability largely relies on the user, but inbuilt 'OMERO Analytics' can help ensure not only traceability of the origin of the data but also trusted provenance (The Open Microscopy Environment, 2017). By running OMERO on a powerful server, fast response can be achieved. In version 4.3, a tiled image viewer for large web-based 2D images

[9] Official site: https://www.openmicroscopy.org/omero/ (last accessed 02/08/2018)
[10] Available at: https://help.openmicroscopy.org/workflows-fiji.html (last accessed: 12/12/2018)
[11] Available at: https://docs.openmicroscopy.org/omero/5.4.9/developers/Matlab.html (last accessed: 12/12/2018)

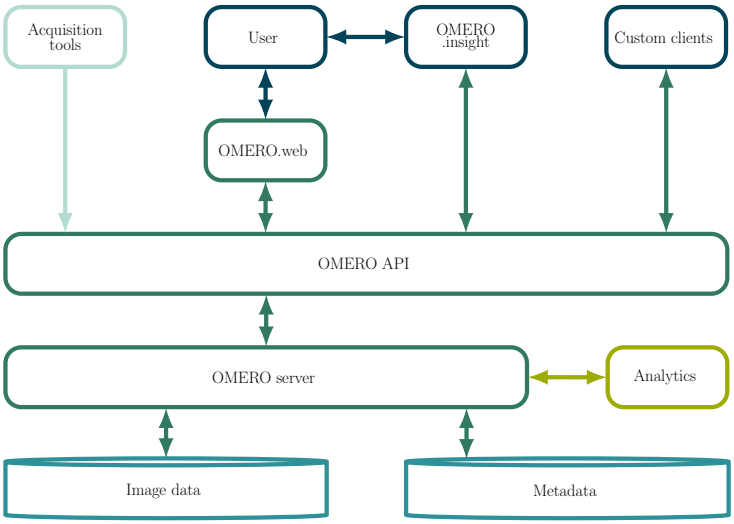


FIGURE 2.17: Architecture of OMERO: All requests are handled by the OMERO API. Matching colours indicate parts which fulfil a similar role within the architecture.

was added (University of Dundee, 2014). OMERO was scalable to multiple terabytes of datasets as of 2012 (Allan et al., 2012) and it is assumed that this capability has increased since. A connection pool increases the number of possible database connections to 500 and the server allows an equal number of simultaneous invocations (The Open Microscopy Environment, 2018). Clustering OMERO using multiple servers is possible using OMERO.grid. Burel et al. (2015) described the steps taken towards data security. The API uses secure sockets layer (SSL) encryption by default. Data access can be granted through a privilege system using user groups (Burel et al., 2015).

OMERO has been around for many years and provides reliable image management mechanisms. Although the API is not user friendly to medical users, it enables programmers to develop useful tools for medical researchers and integrate new software. It also provides an option for data traceability and provenance. Over the years the importance of larger images has been acknowledged by the OMERO developers. Scalability and data security has been addressed when implementing custom servers and tools. The use of many tailored software solutions reduces the modularity of OMERO.

2.5.2 BisQue

The Bio-Image Semantic Query User Environment (BisQue)[12] is a comparatively new alternative to OMERO. Dating back to 2007, it was established with the aim of making 5D medical images of several hundreds of gigabytes in size accessible over a web server (Kvilekval, 2007; Kvilekval et al., 2010). It is maintained by the University of California. Similar to OMERO, BisQue provides a full solution for all parts of the image life cycle introduced in Section 1.2. We will look at all the requirements for an image workflow described in Section 1.4 when reviewing BisQue in this section.

[12]Official site: http://bioimage.ucsb.edu/bisque (last accessed 02/08/2018)

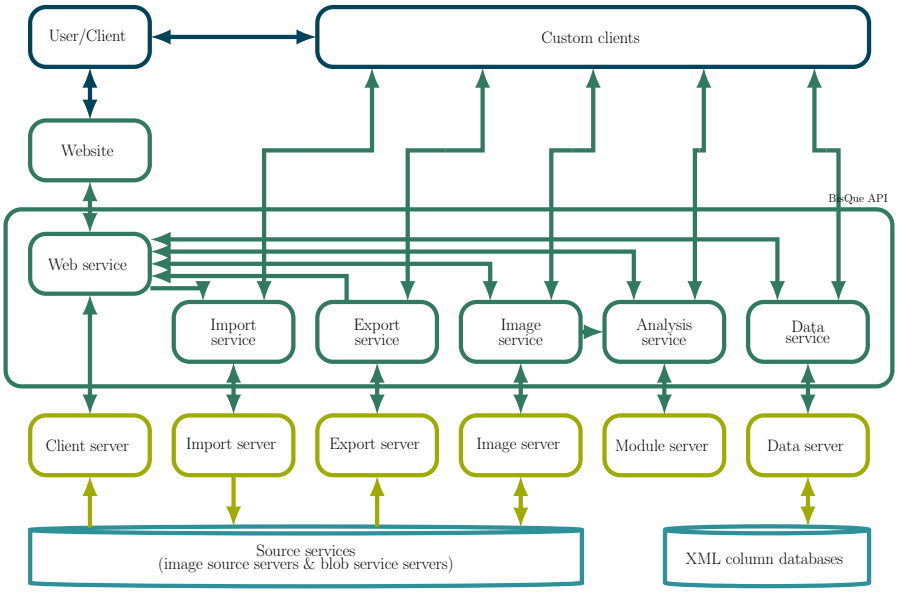


FIGURE 2.18: Architecture of BisQue: Every task has its own service. An API helps establish communication with these services. Matching colours indicate parts which fulfil a similar role within the architecture.

Besides large 5D images, BisQue supports a variety of image formats and is compatible with several image processing tools. Through the use of cloud computing, several image processing algorithms are provided server-side. Custom codes can be programmed and executed in the cloud (Kvilekval et al., 2010). The BisQue API allows custom plug-ins to be created. As mentioned in the previous section, an API is very useful for programmers but not user friendly for medical researchers. BisQue is built as many interacting servers, each running a service as shown in Figure 2.18. This mechanism makes BisQue very scalable, as multiple servers for the same service can be deployed to improve the performance of that service. Some services rely on other services, but all are accessible through the BisQue API. Data is split over 'Image Sources' and 'Blob Services'. The first ones are responsible for storing small files and images for direct access, while the latter ones are specifically for large files and complex data formats. Metadata is stored as documents in XML (Kvilekval et al., 2010). BisQue supports trusted provenance. Any alteration of data has to be done through BisQue and is recorded by the metadata server. Datasets stored cannot be altered (Kvilekval et al., 2010). BisQue's inbuilt image viewers allow viewing 3D to 5D images as a volume and 2D to 5D images as single slices. The slice viewer allows zooming a tiled image as well as moving through the planes of it with a slider. Data security is ensured through server-based authentication and secure protocols. There are three access levels: owner, public and private access. The creator of a dataset is automatically assigned to be the owner (Kvilekval et al., 2010).

While BisQue and OMERO both have several modules, they are not modular based on our definition in Section 1.4, since their modules are connected through their own APIs rather than established standards.

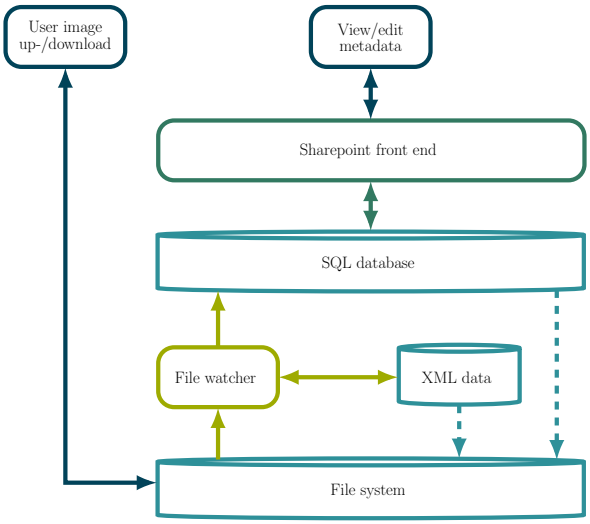


FIGURE 2.19: Architecture of HDC: Users have direct access to their data through a network file share. A file watcher synchronizes between the file store and a database containing metadata. Matching colours indicate parts which fulfil a similar role within the architecture.

2.5.3 Heterogeneous Data Centre

The Heterogeneous Data Centre (HDC) by Scott (2014) is based on the Materials Data Centre (MDC) and aims to manage experiment data over its entire life-cycle. Whilst OMERO and BisQue consider the data to be acquired, processed and then stored for the purpose of archiving and sharing, the HDC is built around the idea of managing data through its full life-cycle, starting at the data acquisition. In order to achieve data management this early on, HDC is designed not to interfere with the end-user's data usage. The main difference among HDC, BisQue and OMERO is that, unlike the others, HDC provides direct file store access that allows users to manage experiments without the need to install specific software (see Figure 2.19).

Using a file store to store large datasets is common practice since databases are limited in size. If the data is stored as separate files, the size is only limited by the file store. As Sears et al. (2006) showed, SQL databases are much more efficient than file systems for large amounts of small files but less suitable for files larger than 1 MB. Metadata of the data is stored in a database, as explained in Section 2.1.2. In HDC, a link to the data is stored in the database so that any application connecting to the database knows where the actual data resides on the disk. Giving the user direct access to the file store introduces issues: for instance, the content of the data might be changed and the data on the disk and metadata in the database might get out of sync. Therefore, a tool is required to monitor both the file store and database and resolve disagreements through synchronisation.

To enable synchronisation, HDC has a file-system monitor, which will be referred to as the '*file watcher*' for the rest of this thesis. The file watcher compares the state of the file system storing the data to that of the database storing the metadata and modifies each of them to attain consistency. The file watcher is triggered by file-system events to analyse a particular folder. It also checks every folder in the system at regular intervals in case system events were missed. Information about the datasets' files is stored in hidden sub-directories of the data folder. With

this information, HDC determines and differentiates between additions, removals, renaming, and copies on a folder and file level (Scott, 2014; Scott et al., 2014). If the database differs from the file store, then:

- New files/folders are added to the database,
- renamed files/folders are renamed in the database,
- copied files/folders are added to the database,
- removed files/folders are marked as deleted in the database and
- new permissions are copied from the database to the file system.

Another functionality provided by the file watcher is automatic execution of plug-ins if specific file-types are added. This allows extending the capabilities of the file watcher further by interfacing with other external software (Scott et al., 2014). We will make use of this functionality later in Section 5.1.4.

While the file watcher synchronises the file store and database, users get access to the file store via a network file share and can edit, add and remove data. The dataset owner can provide other users read or read and write access through the website or the file store. The permissions are synchronised between the website and the file store. Moreover, a web front-end based on Microsoft SharePoint[13] allows editing of database entries such as the metadata of datasets, relations and access permissions. SharePoint tools were created to allow metadata to be edited or imported from templates. In HDC it is possible to search for datasets and to execute plug-ins on them. Figure 2.19 gives an overview of the architecture of HDC.

2.6 Summary

In this chapter, we have reviewed technologies and theories related to the life-cycle of images in medical research. First, we addressed the management of images. We introduced approaches to managing metadata, and looked at different database systems and their advantages and disadvantages. We then combined the knowledge acquired about metadata and databases when reviewing existing image management systems used in medicine. When managing image data, one has to store not only the numbers defining the image, but also relevant metadata. Metadata can be structured in a strict hierarchical taxonomy or in a user driven folksonomy (see Section 2.1.1). To store this metadata, five common types of databases are available, usually grouped into SQL and NoSQL databases: we analysed them in Section 2.1.2. NoSQL databases offer more flexibility when it comes to structuring data and also enable horizontal scaling of databases over several servers. However, when using SQL databases, usually vertical scaling is the only way to increase performance. The limit in database size is a major disadvantage of relational databases when it comes to datasets with millions of entries, and other solutions may offer better performance in write-intensive applications or complicated queries. However, SQL databases outperform NoSQL databases for simple, read-intensive applications and have much

[13] Official site: www.sharepoint.com/ (last accessed: 02/08/2018)

better security implementation than the less well-established NoSQL alternatives. For a medical datastore, security is a priority and new database types only enable coping with large amounts of datasets, not data which is large in itself. The use of SQL was therefore preferred when designing a management system for medical images in Chapter 3. In Section 2.5, we introduced the concept of an image management system which manages image data by using a database to store related metadata. The underlying part of any image management system is the storage of data. We reviewed existing image management systems for medical research, such as OMERO and BisQue, and finally investigated HDC as an alternative approach to data management. HDC provides users direct access to their data through a network file share.

The medical image workflow requires not only the management of but also the processing of image data to extract useful information from the images. Different steps of image processing have been introduced. We categorized processing algorithms into various groups and explained each of them in Section 2.2. Image enhancement and image segmentation are affected the most by the increase in image size described in the beginning of Chapter 1. Section 2.2.6 described the different processing algorithms. While thresholding, region-growing, active contour models and machine learning algorithms produce distinct regions/segments, most filters are used for image enhancement and can be used for segmentation only in combination with other methods.

Digital images are a set of numbers and therefore can be interpreted by a machine. However, for humans to understand digital images, visualisation is needed. In Section 2.3, we looked at traditional visualisation methods in 2D and 3D, namely, variations of 2D sectioning and 3D volume and surface rendering. We also looked at Holograms, VR and AR, which give the eye a better perception of 3D objects retrieved from 3D scans. We talked about 3D printing as a new fast manufacturing technique and how it can be used to the advantage of doctors in medicine.

Finally, we had a look at the DICOM standard as the main medical standard for software engineering, comprising both a data storage standard, as well as a data transfer protocol.

We will build upon the information provided in this chapter when developing new and improved approaches to handling the steps of the image life-cycle in the following chapters.

Chapter 3

Image Storage

The work described in this chapter has been accepted for publication as:

Wollatz, L., Scott, M., Johnston, S. J., Lackie, P. M., Cox, S. J., Oct. 2018. 'Curation of image data for medical research'. *In: Proc. 14th Int. Conf. eScience (eScience 2018)*. IEEE, Amsterdam, Netherlands, pp. 105–113. doi:10.1109/eScience.2018.00026

In Chapter 2, we reviewed literature related to the image life-cycle defined in Section 1.2. We established different groups of image processing and visualisation and provided a detailed review of existing image management systems and the underlying concepts of databases and metadata. A more general review of database systems, which form the basis of every data storage system, and their suitability for medical image data were discussed in Section 2.1.2. The most popular existing research systems for managing medical images have been reviewed in Section 2.5. Figure 3.1 gives an overview of how these systems work. The connection to other software is restrictive and not user-friendly to a non-technical audience, such as medical researchers.

In this chapter, we will look at improving the storage of medical images. Users require a way to store, find and share relevant information. The storage of data is important since its value may not be fully obvious at the time of its creation, but future research can benefit from the data.

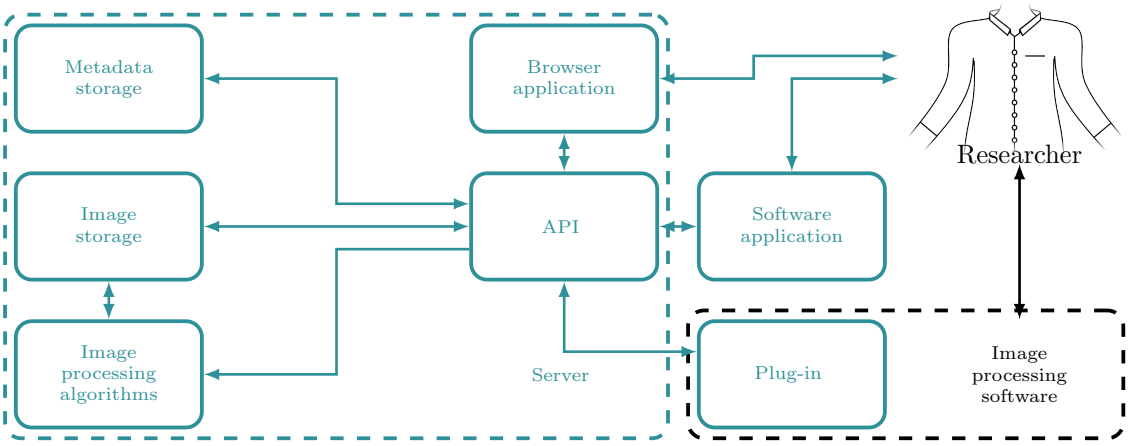


FIGURE 3.1: Research image management system as used today. Light parts are system specific in their implementation.

To make sure the data becomes and stays available, management of data has to start at a very early stage of the life-cycle of the data. It ensures that data can be shared within a research project. Several people may create data and the same or others may process or interpret the data, which refers especially to images in our case.

A storage system needs to implement data-storage and access, handle user permissions and help users find the data they require. Researchers also have a particular interest in processing the data. For smooth data processing, the interaction between processing software and the storage system is of interest.

Based on the outcome of the review in Section 2.5, a web-based image storage system will be developed in Section 3.1. The downfalls of a purely website-based system will be discussed and an alternative approach, which gives users direct access to the data, will be taken and discussed in Section 3.2. A summary of the approaches and how they improve the existing storage systems is given in Section 3.3.

3.1 A Website for Medical Data

Databases, as discussed in Section 2.1.2, require being linked to an interface in order for data to be accessible. For an initial system, it was decided to create a website-based interface. Websites have the advantage that they can be made widely available without the requirement of specialised software on the end user's machine. For this project, Microsoft WebMatrix 3[1] was chosen as the working environment, using Microsoft SQL and Microsoft Internet Information Service (IIS). WebMatrix 3 provided a gallery template, which made it easy to prototype a website with the intent of testing interacting software. WebMatrix 3 is no longer supported and we will address this in the discussion of this section. The simple website is able to store information about images as defined by users inside the database, while keeping a link to the image data files. Image files can be sorted by tags, galleries or users. We call this system Mata[2].

The main target was to test users' access to the image data. While the download of images to the user's computer was achieved through direct links to the corresponding files, uploading files is more complicated. Either users or medical imaging devices can undertake uploads. Sections 3.1.1 and 3.1.2 will discuss how each of these scenarios can be achieved. In Section 3.1.3, we will discuss the results and argue why it is not ideal to provide users with web access only.

3.1.1 File Upload

One of the key challenges faced when building the website was to enable users to upload large image files over the internet to the server. IIS has an inbuilt security layer that limits the maximum data size that can be sent within one request. Files larger than the limit will be denied. The limit can be increased, but browsers are likely to run out of memory as they prepare to send GB of data or restrict the maximum file size (Lawrence, 2011).

[1] Official site: https://www.microsoft.com/web/webmatrix/ (last accessed: 31/01/2018)
[2] Source code and documentation published as doi:10.5258/SOTON/D0430 (Apache 2.0 License)

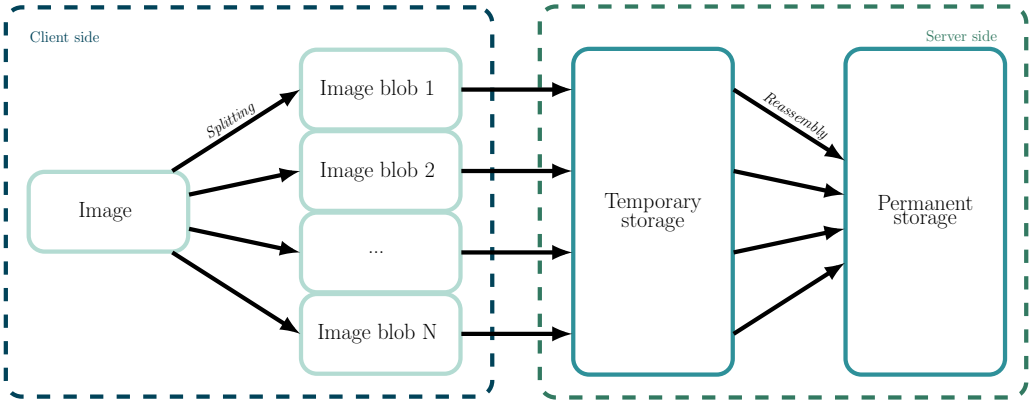


FIGURE 3.2: Schema of Plupload: files are split client side and sent in small blobs to the server.

To overcome this limitation, the default image upload process was extended using Plupload[3]. This library allowed files to be split on the client side, sent as separate file blobs and reassembled on the server side when received (see Figure 3.2).

By uploading the data as small blobs, file upload limits can be bypassed by simply setting the blob size to that limit. Moreover, browsers can release memory after every blob sent. This way, the upload process also becomes more responsive.

In spite of Plupload overcoming the file-size limitations, it did not overcome issues regarding the speed of uploading files. The university network speed for example was measured to provide a maximum of 260 Mbit/s upload speed. This was never reached in practical scenarios due to bottlenecks in other parts of the transmission. Uploading images tens of gigabyte in size will still take up to several hours and users need to remain on the webpage during that time. This lengthy process is connected to errors in data transfer becoming more likely with increasing image size. Additional solutions like resumable.js[4] are required to overcome this limitation.

3.1.2 DICOM Receiver

In order to support medical imaging and display services, it is important to support the DICOM standard. For this, a DICOM receiver was built, based on miniPACS[5]. Section 2.4 demonstrates the procedures of a file transfer for an SCM when an image is sent from an SCU to an SCP.

The receiver created uses DCM4CHE[6] to receive DICOM files according to the transfer protocol, as defined by NEMA (2014e, p. 39–43). It accepts DICOM files sent to it and places them into a pre-defined directory. A Python script was written following miniPACS, which monitors that directory and copies the incoming image files into the website directory. We tested this set-up using DCM4CHE to send DICOM files from another computer. They were successfully received and stored by our system.

[3]Official site: http://www.plupload.com/ (last accessed 03/08/2018)
[4]Official site: http://www.resumablejs.com/ (last accessed 14/01/2019)
[5]Available at: http://www.free-dicom-pacs.com (last accessed 13/02/2018)
[6]Official site: http://www.dcm4che.org (last accessed 03/08/2018)

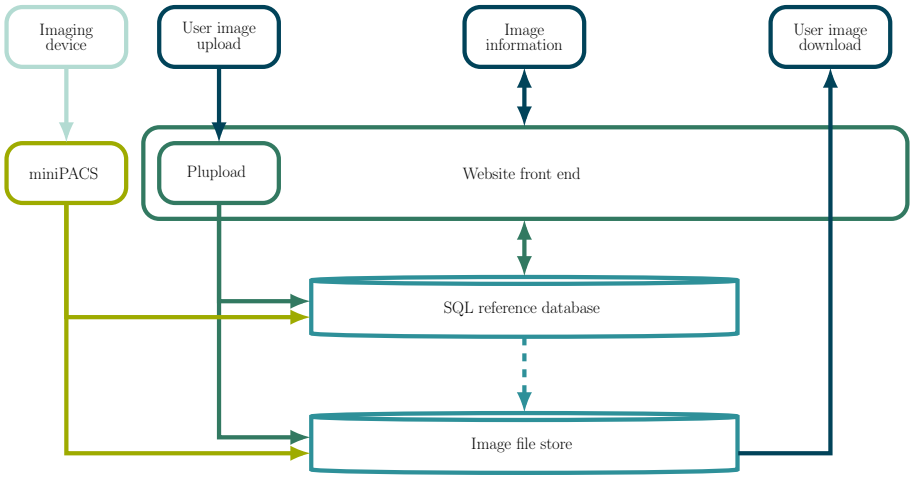


FIGURE 3.3: Architecture of the website with the addition of miniPACS and Plupload. Matching colours indicate parts which fulfil a similar role within the architecture.

The main issue found was that DICOM access security is based on patient information (also see Section 2.4.4). A doctor is granted access only to their patients' data. The information identifying the patient is found in the file metadata. For research-related files, it is of high importance to anonymise files for ethical reasons. Accordingly, any patient-related information is removed from DICOM files before being used in research. Therefore, no link that defines users who should be granted access to a received file exists. This means that any received DICOM data needs a (human) moderator, who assigns permissions to the dataset.

With a moderator in place, an open source DICOM receiver can be used to allow medical imaging devices to send images to the storage system.

3.1.3 Discussion

Using WebMatrix 3, a website that fulfilled the basic requirements of a storage system as outlined in Figure 3.3 was created, wherein users have two options to upload their data. The first option is to upload data from a medical imaging device using the DICOM transfer protocol to connect to the miniPACS DICOM receiver. The receiver in turn stores the received files and adds them to an SQL database. Alternatively, Plupload can be used to upload files of any size directly through the browser. Additionally, support for previewing files was added, as will be discussed in detail in Section 5.1.

This design is much simpler than the common one, as can be seen on comparing Figure 3.1 with 3.4. However, this comes with the downside that the external software cannot be directly supported anymore. Without an API, users need to download data first before importing it into the software they are using. Moreover, the software application used by systems such as BisQue and OMERO allow more reliable image uploads than our system. Software applications can handle uploads in the background while the user continues navigating the storage system and can even pause the file transfer. The latter allows systems to deal with network issues since

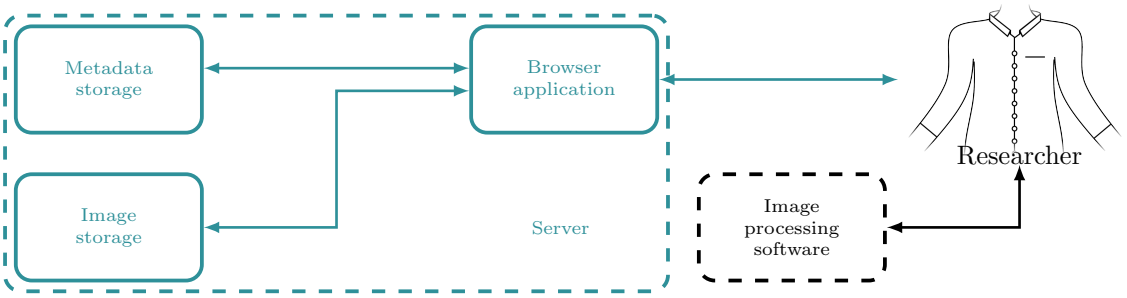


FIGURE 3.4: Research image storage system implemented by the first iteration of Mata compared to the common implementation used by existing systems in Figure 3.1 on page 45. Light parts are system specific in their implementation.

started file transfers can be continued after the connection is re-established. These functions can be implemented in a browser using existing JavaScript libraries. The downside of both browser internal and external upload solutions is the lack of integration with existing software.

By deploying the web-based data storage system, it was shown that it is possible to provide users with the essential functionality of file upload and download and editing of the metadata. Further, it was shown how to implement tools to enable DICOM-based medical devices to upload data to a custom server.

This attempt also revealed certain issues, such as that of ownership of images coming from a device without user authentication as well as the bottle-neck of uploading very large datasets over the web. The use of Plupload enables the upload of large files. It does not overcome issues regarding the speed and reliability of uploading files. The WebMatrix 3 framework is no longer supported, which will affect security updates and long-term sustainability if used in a real system. The use of a framework should be considered carefully when trying to provide a long-term solution. Instead, one can use self-sustained frameworks or provide a solution without the use of frameworks. This allows web developers to add their preferred framework, which they can replace if needed.

This first iteration of Mata is replaced by a new system presented in the following section, which will overcome these disadvantages.

3.2 File Store-Based Version of Mata

In Section 3.1, we tested a website-based system for image storage. The upload of several tens of gigabytes of data through a web browser is slow and unreliable due to browser limitations (Ephox, 2017; Lawrence, 2011). The upload speed is mainly dependent on the internet performance, which cannot be controlled by the system. There are two common solutions to implement the upload of large files. Either a web browser application is used, or a separate software connects to both the local file store and the remote one. We used Plupload in Section 3.1.1 to overcome the issue of file size when uploading large datasets, but it was not able to increase reliability compared to that offered by software-based file upload as used by other storage systems (Allan et al., 2012;

Kvilekval et al., 2010; Marcus et al., 2007). The alternative approach, as used by Allan et al. (2012), Kvilekval et al. (2010) and Marcus et al. (2007), is a software running on the researcher's computer that interacts with an API on the storage server. The downside of a system-specific software for image upload and download is that it reduces user-friendliness since it requires users to learn the operation of the image management software. Providing access to data only via an API introduces an additional step to connect other software. Any new software will require a computer scientist to integrate it with the storage system via an API. The APIs used by existing management systems are not standardized. The existing clinical standard for data storage and transfer, DICOM, relies on detailed patient information as explained in Section 2.4.4 and cannot be used in a research environment, where the patient needs to be anonymous (NEMA, 2014a; Warnock et al., 2007).

To overcome concerns regarding reliability and user-friendliness of data transfer, we chose to create a system based on HDC, as introduced in Section 2.5.3 on page 41, as a back-end. HDC gives users direct access to the server file store. Using a network file share gives users a familiar environment for accessing their data while ensuring robust file transfer and storage methods that evolve with the file store.

In this section, we will explain how we developed a functional, user-friendly system for managing medical image data by modifying HDC.

3.2.1 Adapting HDC

HDC implements a variety of features for the storage of data, specifically the synchronisation of a file-store and a database. Most features were deemed useful for medical research. For Mata, we retained the Windows server, to integrate with companies' Windows networks, as well as the SQL database used by HDC for metadata storage. The layout of the database was copied since it already exhibited the versatility required for this project.

Other parts of HDC were considered unnecessary for this particular implementation. Specifically, the SharePoint interface was removed and replaced with a more lightweight PHP website. SharePoint integrates well with Microsoft environments since it is part of the Microsoft Office suite. The number of third-party applications for SharePoint is limited and integration of custom plug-ins, though possible, requires programming them in .NET or finding another way of implementing them. The original interface was built with SharePoint 2010 and .NET 3.5. That version of SharePoint is out of date, and we found it to be slow. We did not use many of the SharePoint features that were implemented with the HDC front-end, either. Instead of reimplementing the SharePoint front-end in a more recent release, we created a PHP site for Mata that was easier to prototype and is fully open source. The use of PHP also allowed easy reimplementation of essential features, such as the editing of user permissions and dataset metadata. The architecture of this new version of Mata is shown in Figure 3.5.

The data was structured as follows. Each user has a folder in the top level of the shared file store. Inside that folder, they can create more folders, each of which is considered a dataset. Each image is a dataset of its own. If an image is processed, a new folder is created for the processed version of the image. New folders are also created for intermediate results of image processing or images that show an extracted region of another image. The related datasets can

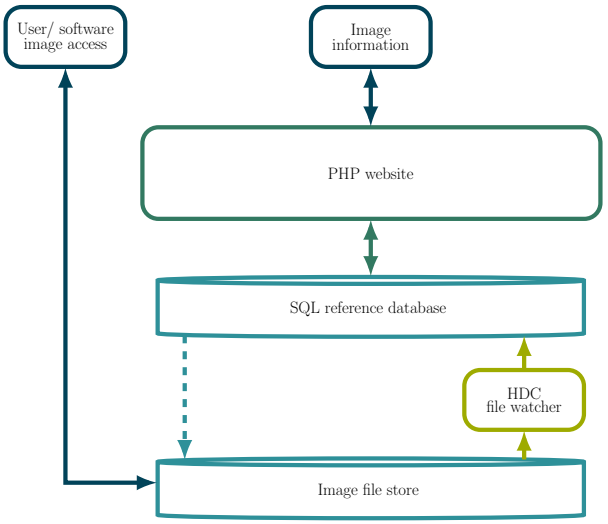


FIGURE 3.5: Architecture of Mata 2. Compared to the architecture of HDC illustrated in Figure 2.19 on page 41, the website has been changed from SharePoint to PHP. Matching colours indicate parts which fulfil a similar role within the architecture.

be linked through the database. As with any storage system, we rely on the user to only upload and store sensible data.

The rest of this section will address various parts of the storage system and the corresponding solutions we have developed. We will briefly discuss feature re-implementations of HDC and then introduce any additions made to the system for each part. We distinguish between the following modules of the storage system: editing of metadata, visualisation and searching of metadata, and system security. We will also discuss the visualisation of datasets later in Section 5.1.

3.2.2 Editing of Metadata

Metadata here refers to key-value pairs (Scott et al., 2014), which we refer to as tags in order to match users' expectations (see Section 2.1.1). Tags allow users to assign metadata to datasets. Tags can help maintain information about data that is important to medical researchers but cannot be stored as part of the image file. For example, a tag can describe the species seen in the image, which may be a human, a sheep, or a zebrafish. In this case 'species' would be the key and 'human' the value. As with HDC, editors of the dataset are allowed to sort tags in a particular order and create hierarchies for them. These hierarchies allow users to structure the tags according to their needs.

We chose a folksonomy over a taxonomy (see Section 2.1.1), as there was no suitable taxonomy developed for the specific field of pathology. A challenge faced when allowing users to edit the metadata is the lack of consistency, which makes comparison of the tags more challenging (Golder and Huberman, 2006). We used two solutions to overcome this problem.

One solution we adapted from HDC to avoid users using different terms to describe the same thing involved the ability to import tags from parents. It assumes that users will tend to use this

functionality to save the effort of copying common tags. When importing tags from a parent, all tags from that parent, where the key has not yet been assigned any value for the dataset, are copied over to the dataset. This approach ensures consistency among tags between different datasets, users, and sites. Not copying values of keys which have already been defined avoids overwriting data or having duplicate keys. We also preserved the hierarchy of tags to ensure equality of tags, as defined in Section 3.2.3.3, between different datasets.

In addition, Mata implements a dictionary to guide users into using the same tags for describing the same features (Noruzi, 2007). The advantage is that synonyms can be avoided. The dictionary also provides guidance, to avoid typographical errors and reduce the number of grammatical variations of the same term. The system suggests existing tags to users based on the user's input. Tags that have been used more often have been ranked higher amongst the suggested tags.

3.2.3 Visualisation and Searching of Metadata

One of the main requirements of a data storage system is the ability to look up data in an efficient manner. Our solution provides three different options for finding datasets: a basic search, tag clouds, and relation networks.

3.2.3.1 Basic Search

One functionality re-implemented is a basic search function which looks for the exact phrase in the datasets title, description and tags. It returns any matches found in the order of the number of matches of the search term found in the database. This search function can be expanded to be more sophisticated if needed (Brin and Page, 2012; Dessloch and Mattos, 1997; Kießling and Köstler, 2002; Tsuruoka et al., 2008) and to use a medical ontology, as done by Díaz-Galiano et al. (2009). Due to the limited amount of metadata stored for the first samples of the research project introduced in Section 1.1, such extension of the search function was not considered necessary at this point.

3.2.3.2 Tag Cloud

A page with all of the different tags was created, to enable users to browse the tags. This was a request from medical researchers to help them visualise the metadata. Tag clouds also make the metadata more prominent and encourage users to add tags (Willoughby et al., 2014). Tags were arranged in an elementary rectangular tag cloud as shown in Figure 3.6, to aid users in finding important tags quicker. The implementation of more sophisticated tag clouds is possible and has been discussed in many papers (Halvey and Keane, 2007; Kuo et al., 2007; Seifert et al., 2008). We implemented our own tag cloud here and will outline it in the following paragraphs.

In order to prioritize tags in a tag cloud, they are represented in different font sizes, colours, or layouts. Tags that appear more often are larger, such that specific, unusual tags or typographical errors that only appear for a few datasets are smaller and less noticeable.

We assume that more frequently used tags are more likely to be of interest to a common user and therefore should be larger and easier to find (Sinclair and Cardew-Hall, 2008). Meanwhile,

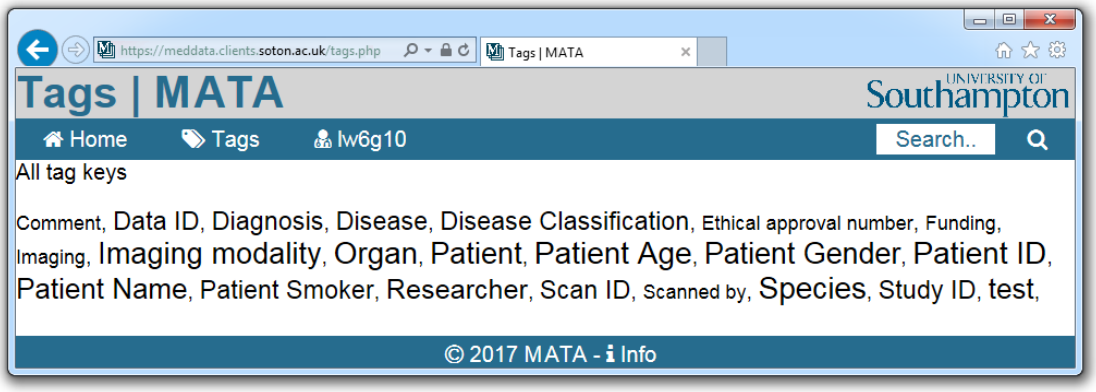


FIGURE 3.6: Screenshot showing a rectangular tag cloud for a small number of tags. Entries are in alphabetical order. More common tags have a larger font size than less common tags.

to help users searching for specific tags, we minimized the variation of font size between tags. We used Zipf's Law (Zipf, 1949) to map the occurrence of specific tags to a linear distribution. We also maintained an alphabetical ordering of tags to help users find specific tags (Halvey and Keane, 2007). Details about the implementation of the tag cloud can be found in Appendix A.

3.2.3.3 Relation Networks

When editing a dataset, users can define parent datasets, i.e. datasets which the current dataset was derived from (Scott et al., 2014). An example is a segmentation of a scan which can be linked to the original image dataset. By default, a dataset is expected to have precisely one parent unless it is an original scan, in which case it does not have any parent. In some cases, several images may be combined into one. An example is the correlation of a single photon emission computed tomography (SPECT) image and a CT image. Areas of abnormality found by a SPECT can be located relative to internal organs, detected by the CT (Soo and Cain, 2017). In such cases, more than one parent may exist. Having the origin of an image defined ensures that they can be traced back.

A network graph (see Figure 3.7) is generated, using recursion to search for all related datasets in the database and vis.js[7] for display. In this graph, each node represents a dataset and arrows connecting the nodes indicate their relation, as seen in Figure 3.7. This graph helps users in finding other datasets derived from the same scan with just a few clicks. Similar to the tag cloud, it also provides users with a prominent visualisation of metadata and, therefore, encourages the use of metadata.

This view has also been combined with the tags explained in the previous section. As a result, it can display a hierarchy of tags for a folksonomy (see Section 2.1.1) by assuming that two tags are equal, if their keys, their values, and child tags are all the same. The order of the children is disregarded.

[7]Official site: http://visjs.org (last accessed 03/08/2018)

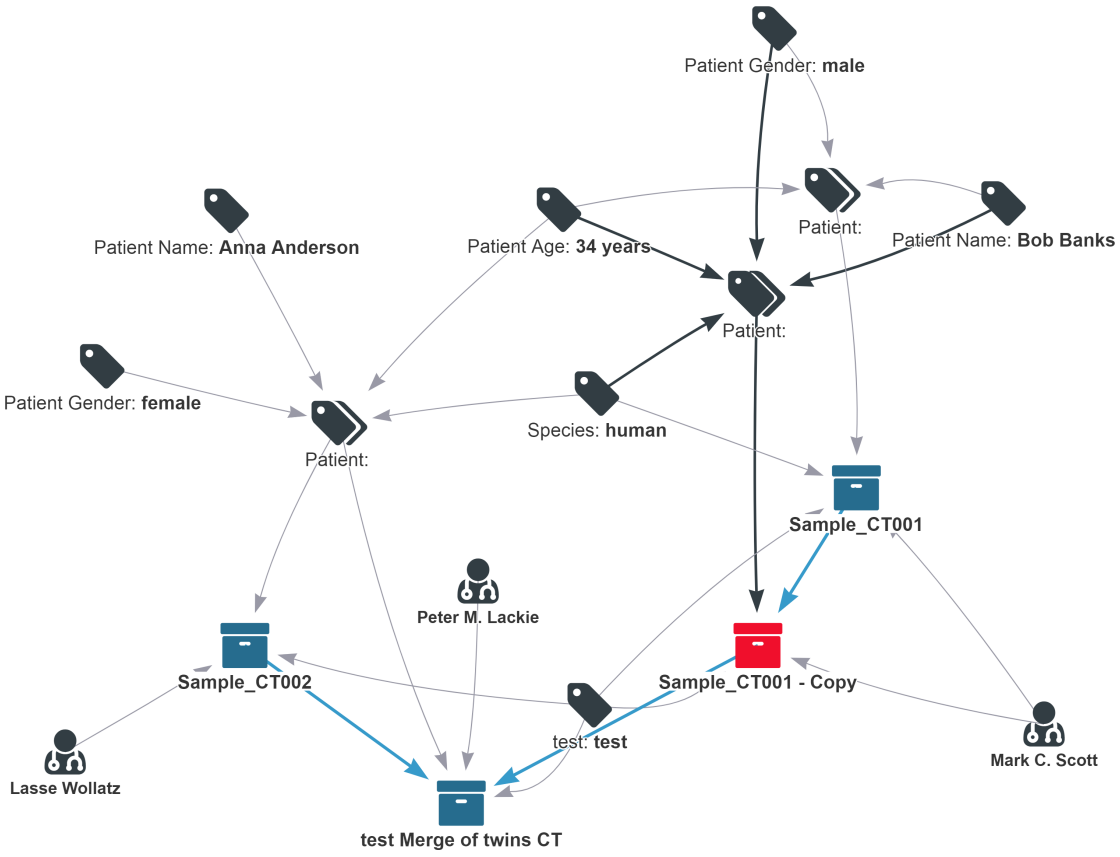


FIGURE 3.7: Screenshot of a dataset relation network graph using fictional datasets for demonstration purposes. The red dataset is the one the graph was created for. Related datasets are shown as dark blue boxes and related tags and users with according symbols. Normally, a research dataset would not contain the patient name, but we use made up names here for demonstration purposes.

As seen in Figure 3.7, the datasets '*Sample_CT002*' and '*testMerge of twins CT*' are tagged with the same patient, as all the child tags related to '*Patient*' are the same. However, they are different from the patient of '*Sample_CT001*', as they only share a single attribute.

Figure 3.7 also illustrates the limitation of this strict definition of equality, since the '*Patient*' in '*Sample_CT001*' and '*Sample_CT001-Copy*' are considered different even though they describe the same person. Not all child tags of '*Patient*' are the same, because the researcher in one dataset lists the species as part of the '*Patient*' information. The gravitational physics model behind vis.js solves this issue by ensuring that such tags, which are almost equal, will remain close together since they share many children.

The rather strict definition avoids false positives. In medicine, critical metadata, such as the patient, have multiple identifiers. This reduces the chance of tags falsely being identified as equal.

The comparison of tags and their children, as mentioned before, occurs over several levels. As a result, the tag 'Imaging' may include subcategories, for example, 'Scan settings', containing a list of settings such as the exposure rate and projections. In the web interface, every tag links to a page that shows all datasets containing that tag.

3.2.4 System Security

The importance of security for medical databases has been highlighted through recent incidents (Fox-Brewster, 2017; TrapX Research Labs, 2016). For the security of a service, various functional components have to be considered and evaluated. This is particularly important when dealing with sensitive medical data. Components that could be vulnerable include the user authentication and the database. Most of these components have been discussed by Scott (2014) and Scott et al. (2014). The specific permissions in Mata were kept the same as in the case of HDC. Everyone can list folder content but access to file content is limited to certain users. The owner (the creator) of the dataset can decide whom to give access to, and people with access can preview and edit the data in a browser. Permission can only be modified by the owner or a system administrator. Content is moderated by trusted administrators. This can help overcome security issues with external imaging devices, as encountered in Section 3.1.2.

HDC also forwards the permissions as read and write permissions to the server file store and the connected network file share. The file access on the network file share is secured through Windows authentication. Applications, such as the PHP interpreter, running on the server responsible for the website need elevated permissions in order to access the database and other data. Further, the website authentication has to be re-implemented with PHP.

When webpage content is generated, it is essential that confidentiality of the data is maintained. Accordingly, when rendering the content of a webpage, PHP needs to know who is allowed to view or edit individual content and who is accessing the website. HDC synchronizes folder and file permissions between the database and network file share. As PHP has access to the database, it can check if a user has permission to access specific data. Additionally, Windows Server and IIS enable the use of the same authentication for websites and file access. This enables single sign-on (SSO), indicating that users only need one account and password to access both the network file share and Mata's website. Other systems connected to the domain can also be accessed using SSO. PHP can read the session's details from the server and therefore knows who is accessing the webpage. The user identification is the same as the one HDC synchronized between the servers' file store and database. As a result, PHP knows the users' permissions.

As discussed in Section 2.1.2.4, we will not explore details of the security implementation. The use of well-established systems, like Windows Server, Windows authentication, secure protocols, SQL and PHP enables system administrators to set up a secure implementation throughout the whole data storage system. This includes securing the network drive which is accessible from outside the institution through a virtual private network (VPN). The primary reasons for choosing SQL as the main database type was the well-established security it offers in comparison to NoSQL databases, as examined in Section 2.1.2.4.

3.2.5 Discussion

In this section, we explained how to improve the medical image storage system Mata introduced in Section 3.1. We discussed the underlying software and presented various parts of the storage system. This included the editing of metadata, visualisation and searching of metadata and system security. We combined the advantages of a file store with known functionality to the

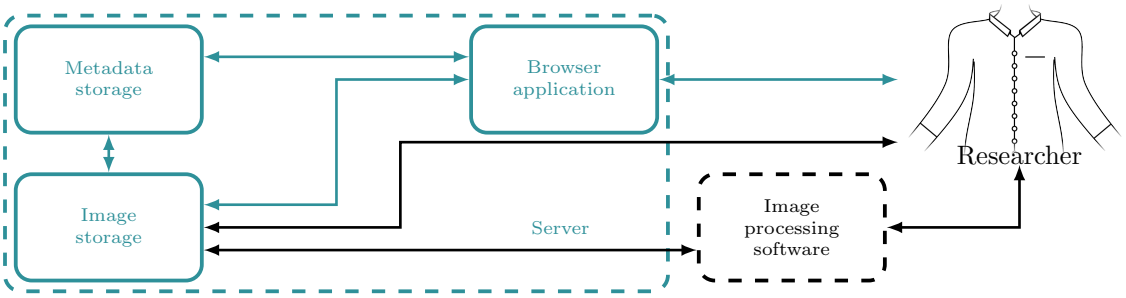


FIGURE 3.8: Research image storage system implemented by the second iteration of Mata as compared to the common implementation used today in Figure 3.1 and the first iteration in Figure 3.4. Light parts are system-specific in their implementation.

end user and established file-transfer protocols with the accessibility, visualisation and searching power of a website with a database (see Figure 3.8). The website was built very modularly and both the HDC file watcher and the PHP page allow for easy extension.

Metadata is created by users and the process of editing metadata is guided through the use of a dictionary. A tag cloud and dataset relation networks allow additional navigation options apart from a basic search. We also explained how we secured the system, including using an SSO solution to control data access. A further extension of Mata with two image previewers has been achieved and will be shown in Section 5.1 to demonstrate the flexibility of this set-up.

3.3 Summary

In this chapter, we implemented a storage system for medical images, Mata. In a first iteration, we tested the use of the DICOM receiver and found that it can be easily implemented apart from access permission issues, which require a moderator. Additionally, we ruled out the option of a system giving users access to their data through a website only. The main issue found was that large file uploads over a website are unreliable and inconvenient for users.

A second iteration of the system improved data access. The solution was to offer users a well-known, integrated way to access and manage their data through a network file store. The HDC file watcher was used to link file store and database, and a standard PHP-based website was built to provide access to the metadata. It offers integrated security and options for data and metadata modification and finding relevant data.

The final system, as shown in Figure 3.8, has solved the data access issues of the first iteration of Mata (shown in Figure 3.4 on page 49) and traditional research image management systems (shown in Figure 3.1 on page 45).

The fully functional image storage system is easy to use due to direct integration of the file store, and easy to extend its features through its modularity. A comparison of the predecessor HDC, the iterations of Mata and two existing storage systems is given in Table 3.1. This chapter has also demonstrated how this system can be expanded to allow users to easily find relevant data through data tagging and defining relations between datasets. Furthermore, we have shown that

functions for security can be integrated in an easy and user-friendly manner. With this system in place, images created by medical researchers can be organised and shared among them.

The remaining chapters will focus on other parts of the workflow. The next chapter will look at a way of extracting information out of the images by means of image segmentation.

TABLE 3.1: Comparison of the iterations of Mata to other image storage systems. Mata 1 has been discussed in Section 3.1 and Mata 2 in Section 3.2.

	OMERO	**BisQue**	**HDC**	**Mata 1**	**Mata 2**
Operating system	Linux (various)	Linux (various)	Windows Server	Windows Server	Windows Server
Server	OMERO.server	BisQue server (various)	Microsoft IIS	Microsoft IIS	Microsoft IIS
Database	PostgreSQL	XML	SQL Server	SQL Server	SQL Server
Middleware	Java	Python	Sharepoint 2010/ .NET 3.5	Webmatrix/ C#	PHP 7.1
Data access	Website/ Client software/ API	Website/ API	Website/ Network file share	Website	Website/ Network file share
Synchronisation	OMERO.dropbox	✗ (not applicable)	File watcher	✗ (not applicable)	File watcher (from HDC)
SSO authentication	✗	✗	✓	✗	✓ (partially from HDC)
Metadata editing	✓	✓	✓	✓	✓
Metadata import	✓ (individual only)	✓ (via file export and import)	✓	✗	✓
Metadata suggestion	✗	✓ (1-to-1 correspondence)	✗	✗	✓ (ranked dictionary)
Basic search (see Section 3.2.3.1)	✓	✓	✓	✓	✓
Sophisticated search (see Section 3.2.3.1)	✓	✓	✗	✗	✗
Tag cloud	✗	✗	✗	✗	✓
List of direct relatives	✗	✗	✓	✗	✓
Relation network	✗	✗	✓ (list of direct relatives)	✗	✓ (graph of all relatives)

Chapter 4

Image Processing

The work described in this chapter has been published as:

Wollatz, L., Johnston, S. J., Lackie, P. M., Cox, S. J., Dec. 2017. '3D histopathology—A lung tissue segmentation workflow for microfocus X-ray-computed tomography scans'. *J. Digit. Imaging* 30 (6), 772–781. doi:10.1007/s10278-017-9966-5

In the previous chapter, we described the design of a system that allows the storage of images coming from an imaging device and related metadata. It provides medical researchers with tools to find images of interest and relevance to a research project. It also provides access to raw image data.

Medical research cannot be based on the raw image data alone. Often, information is needed that is hidden inside the images but not readily available as metadata. Some information will be easy to spot by humans looking at an image, but the large number of images to analyse in a scientific study makes manual processing too slow for effective research. In other cases, the researcher might not know what they are looking for and require a machine to find irregularities or patterns in the data. This can be the highlighting of regions of interest; identification of biological structures; or measurement of parameters of interest, such as the thickness of membranes or cell density in a tissue. All such information can help doctors detect abnormalities and therefore aid them in their diagnosis.

Images are a collection of numbers with spacial or temporal reference. This makes them ideal for analysis by a computer algorithm. In Section 2.2, we reviewed different image processing algorithms. Many methods (Angenent et al., 2006; Dougherty, 2011; Pham et al., 2000; Robb, 2000; Sonka and Fitzpatrick, 2009) have been developed for automatic segmentation of images in medicine. Different images and research interests require different processing algorithms. There is no single solution to every medical research project. We will not attempt to develop a universal processing algorithm since existing algorithms, introduced in Section 2.2.6, are already sufficient in most scenarios. Instead, we will focus on the scalability of existing algorithms to very large images.

This chapter takes the example of µCT images of lung biopsies explained in Section 1.1 and discusses the development of a workflow to process these types of images and extract useful information in the form of semi-automated segmentation of airways and blood vessels. We implement this workflow as LungJ[1] using the free, open source software ImageJ (Abramoff

[1]Source code and documentation published as doi:10.5258/SOTON/401280 (Apache 2.0 License)

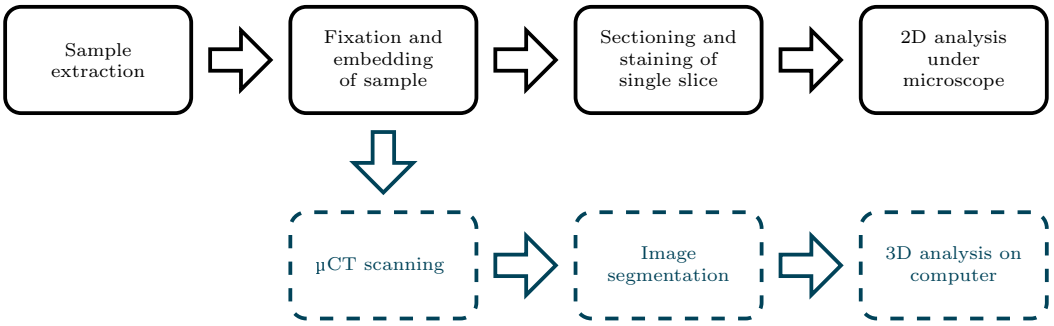


FIGURE 4.1: Current histology workflow (top) and proposed workflow (bottom). As µCT is non-invasive, this can be seen as an additional approach.

et al., 2004). The workflow is scalable to large 3D µCT scans and targets lung tissue samples. Section 4.1 provides details on sample preparation and image acquisition. The procedural steps of the image processing workflow itself are presented in Section 4.2. The background and choices made for individual workflow procedures are explained for each step. A summary of the results is provided in Section 4.3. We will discuss the application of the method behind the workflow to other image processing problems in Section 4.4. Finally, Section 4.5 discusses the workflow in the context of the end-to-end approach for dealing with large images.

In Section 1.1 we discussed histopathology and how it can benefit from the use of new imaging techniques such as µCT. For diagnostic purposes, sections are stained to identify the overall tissue structure and highlight specific tissue components before analysis under a microscope by a trained histopathologist (see Figure 4.1). An alternative approach proposed by researchers at the University of Southampton is the analysis µCTscans of lung biopsies (Scott et al., 2015).

µCT of soft-tissue embedded in wax produces relatively high-resolution ($\approx 7\,\mu m$ or better) but low-contrast 3D images. Identification of key features inside these images is challenging but important. Image segmentation describes the process of distinguishing between the areas of interest in an image and the remaining area. Parts of the image which are not of interest are commonly removed, or areas of different features are marked with different (digital) labels. Manually segmenting some features such as the walls of the airways and blood vessels is effectively impossible. Manual segmentation of the inner part of the tubular structures, the lumen, while possible (Scott et al., 2015), is time-consuming and can take weeks or months per dataset. Automatic or semi-automatic segmentation is therefore required to make this a usable method for research or diagnosis.

4.1 Materials and Methods

Several steps were required before the digital segmentation. The tissue was prepared for scanning while scanning conditions were optimized to provide the best possible resolution and contrast for the tissue. Scans also had to be pre-processed before segmentation.

4.1.1 Sample Preparation

Surgically resected human lung tissue was obtained with written informed consent from patients undergoing elective surgery at University Hospital Southampton. UK ethics approval was given by the National Research Ethics Service Committee, South Central—Southampton A, number 08/H0502/32. A representative sample was taken from a macroscopically healthy portion of the lung of a 75-year-old male non-smoker with GOLD stage 0 chronic obstructive lung disease (COPD) undergoing resection for lung cancer. The samples were fixed with para-formaldehyde in phosphate buffer for 8 hours and embedded in wax following a routine histology protocol using a Shandon Hypercenter tissue processor (Fisher Scientific, Loughborough, UK). This resulted in blocks of 1.5 cm^3, which were then scanned without any further treatment.

4.1.2 Image Acquisition and Pre-Processing

Samples were scanned using a custom-built Nikon Metrology µCT scanner at the micro-scale volume imaging system (µ-VIS) X-Ray Imaging Centre, Southampton, UK. An electron accelerating voltage of 50 kV was used, with a molybdenum reflection target, yielding mean energy in the order of 18 keV, along with characteristics peaks at around 17 and 19 keV. No filtration was used. A beam current of approximately 150 µA was selected, yielding a total beam energy below 8 W. Reconstructions with a voxel resolution of 8 µm were created, using standard filtered-back projection (Ram-Lak filter) within the Nikon CT Pro 2.0 package using 3142 projections (for a 360° rotation in 0.11° increments). Typical datasets included $1800 \times 2000 \times 2000$ isotropic voxels. The value of each voxel corresponded to the density of the material at that position and was reconstructed at 32 bit resolution. For comparison, CT systems at hospitals acquire $512 \times 512 \times 80$ voxel datasets at 16 bit resolution.

4.2 The Workflow

In order to simplify the workflow and avoid repetitive steps, a software plug-in for ImageJ was developed. This plug-in, called LungJ, had to be capable of handling the large data size of individual scans and segmenting images in a way that is meaningful to and can be interpreted by the final user. An image segmentation method had to be established, and images were post-processed to remove noise and artefacts. Finally, representation of the 3D image data was explored to ensure better understanding of the underlying structure. Each of these distinct steps is detailed below, highlighting important decisions made while creating this workflow.

4.2.1 Choice of Software

It was important that the software package of choice, which a plug-in would be programmed for, had to be one that was familiar to potential users. This would maximize usability, especially for users less familiar with image processing. Several software packages were evaluated for implementation of this method (Eliceiri et al., 2012). Paid software such as Amira 3D (Stalling et al., 2005) or OsiriX (Rosset, 2010) was not configurable to the required degree or limited

TABLE 4.1: Popularity of different image processing software in research, based on results obtained on 15th May 2016 from PubMed Central and Google Scholar searches. Search results for Amira 3D excluded authors with the name Amira. The second set of results in brackets includes a large number of results irrelevant to the software.

Software	**Type**	**Main Developer**	**PubMed Central Papers**[2]	**Google Scholar Results**[3]
ImageJ	Open source	Wayne Rasband	75 400	157 000
MATLAB	Commercial	Mathworks	67 305	2 010 000
NIS-Elements	Commercial	Nikon	6 184	18 000
Imaris	Commercial	Bitplane	4 842	12 300
SlideBook	Commercial	3i	2 807	6 010
ImagePro Plus	Commercial	MediaCybernetics	2 464	5 420
Amira 3D	Commercial	FEI	1 903	13 800
OsiriX	Commercial (free version available)	Pixmeo	1 515	9 320
CellProfiler	Open source	CellProfiler Team, MIT	1 055	3 350
ITK-SNAP	Freeware	University of Pennsylvania and University of Utah	505	2 610
Avizo 3D	Commercial	FEI	371	4 210
ParaView			192	6 870
VGStudio	Commercial	Volume Graphics	125	1 620
Vaa3D	Open source	Hanchuan Peng and Howard Hughes	77	216
BioImageXD	Open source	Pasi Kankaanpää, Lassi Paavolainen, Varpu Marjomäki, Jyrki Heino et al.	69	241
icy	Open source	Quantitative Image Analysis Unit at Institut Pasteur	(2 840)	(402 000)
OpenDX	Open source	IBM	(65)	(1 710)
VIPS	Open source	Martinez, K. and Cupitt, J.	(576)	(38 600)

to certain operating platforms. Table 4.1 shows that ImageJ (Abramoff et al., 2004; Schneider et al., 2012) is, together with Matlab, the most widely used image processing software in medical research. ImageJ is programmed in Java and is, therefore, independent of the platform. It has the advantage over Matlab of being open-source freeware. The pre-packaged distribution of ImageJ, Fiji (Fiji is just ImageJ), also comes with a plug-in for machine-learning-based image segmentation.

Based on the framework provided by Fiji, a plug-in was developed to tackle the different tasks of image segmentation. LungJ tools include a function for segmenting an image, as well as tools for pre- and post-processing the image. While all these processes can be done in Fiji, the tools provide important simplifications, as they reduce the amount of knowledge a user needs to have

[2]Result counts taken from www.ncbi.nlm.nih.gov/gquery (last accessed 31/07/2018)
[3]Result counts taken from www.scholar.google.com (last accessed 31/07/2018)

about the image (e.g. bit-depth or absolute values of voxels). Instead of opening the image, launching the image segmentation plug-in, loading plug-in configurations, applying them and then filtering out the relevant mask in separate steps, LungJ achieves all of this using a single graphical user interface, which calls the relevant functions. The overall structure of LungJ is very modular and therefore adaptive, in alignment with the ImageJ philosophy. While the original scan can be in any format handled by ImageJ, LungJ saves intermediate steps as loss-less Tagged Image Files (TIFs). By combining multiple modular functions in one GUI and allowing default settings to be defined, it was possible to reduce the number of steps of the workflow.

4.2.2 Handling Large Images

Each image file produced by the reconstruction of the µCT scan was several tens of gigabytes in size. Image processing requires memory multiple times the size of an image, i.e. several tens to hundreds of gigabytes of RAM. In order to be able to process these large image files, they were split into smaller sub-volumes and processed separately, enabling us to complete processing on a computer with only 16 GB of RAM while still allowing 3D interrelationships with the volume to be addressed. Currently, available methods such as the virtual image stack do not allow custom-sized image blocks. Tests using them, in conjunction with the image segmentation algorithm shown in the next section, caused an out-of-memory error. The image blocks were loaded individually and only on demand. Global properties of the image were saved in a separate file and used during further processing steps to ensure consistency of intensity values over the whole volume. This introduced a novel image representation method which required ImageJ to only hold a directory path instead of loading a full image. Using this representation, any available ImageJ filter can be applied to the whole image with exceptions being discussed in Section 4.4. Furthermore, functions for analysing the global image were written to extract image properties such as the global intensity distribution histogram. The use of smaller image blocks also enables future adaptations of this process in parallel computing or cloud computing.

Three functions for creating, processing and concatenating image sub-volumes were implemented. They form the backbone of the sub-volume image processing. As shown in Figure 4.2, an image was split prior to the segmentation, and a representable block was chosen to test the segmentation. This enabled faster and hence more efficient testing and troubleshooting of a segmentation technique compared to running a segmentation on a full scan. Once the segmentation procedure had been established, it was applied to all the sub-volumes by processing each one in turn. Finally, the sub-volumes were assembled into a single image by applying the concatenating function.

A further addition was the use of halos enclosing the image blocks, which is commonly used in parallel computing (Quinn, 1994). The use of halos avoided boundary artefacts, and a function for halo exchange was implemented, as detailed in Appendix B. Cubical sub-volumes were used when applying three-dimensional processing methods because of their smaller surface area compared to that of other cuboids. For 2D processing algorithms, it is more efficient to have large 2D slices processed.

With these tools in place, a segmentation was achieved in Section 4.2.3 on an Intel-i7-4770 at 3.40 GHz, with 16 GB RAM, within 12 hours, whereas manual segmentation of similar images

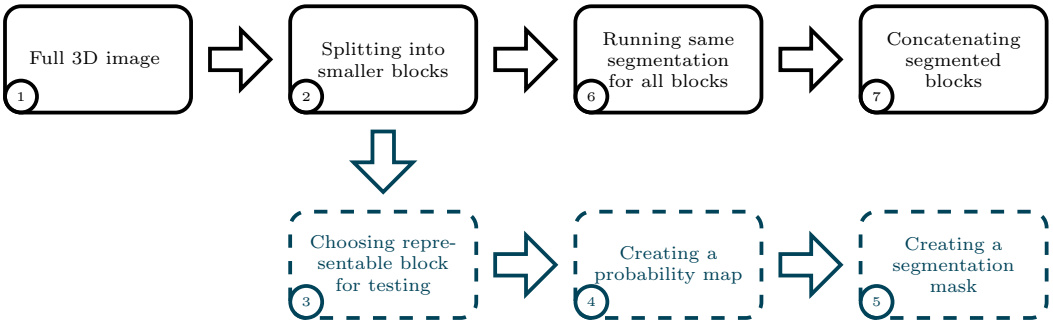


FIGURE 4.2: The modular approach of LungJ: After splitting the image into smaller blocks, the segmentation algorithm is tested for one representable block. It is then applied to all blocks and the blocks are concatenated to a full image.

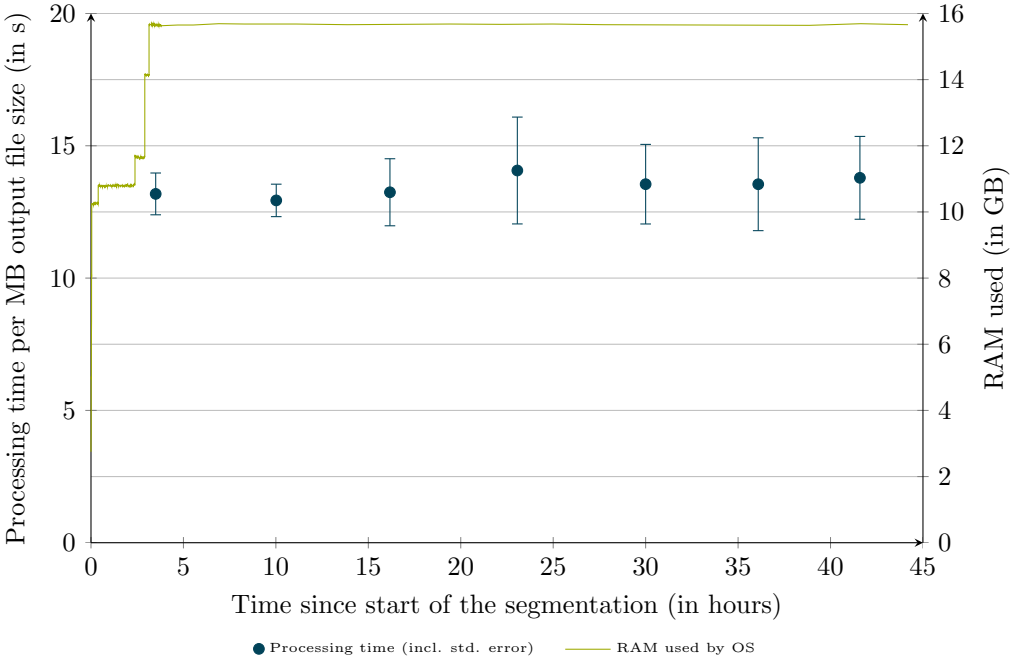


FIGURE 4.3: Processing speed of blocks based on the slowest of the runs (44 hours, Intel Pentium G2020T at 2.50 GHz and with 16 GB RAM). The average processing speed over 35 consecutive blocks was taken and standard errors were calculated. RAM usage is shown for reference.

could take between 3 and 6 months. As shown in Figure 4.3, for a run on a computer with a slower Intel Pentium G2020T at 2.50 GHz, the processing speed does not change significantly over time despite the Java memory management in ImageJ causing full RAM usage.

4.2.3 Image Segmentation

A machine learning algorithm was applied for the image segmentation. Fiji comes with a segmentation plug-in called TWS (Arganda-Carreras et al., 2014). For the segmentation, the random-forest algorithm (see Section 2.2.6.3) was chosen. This algorithm has been used for image segmentation of neuronal processes in biomedical applications as well as segmentation of larger vessels in CT scans of whole lungs previously (Kaynig et al., 2010; Rudyanto et al., 2014).

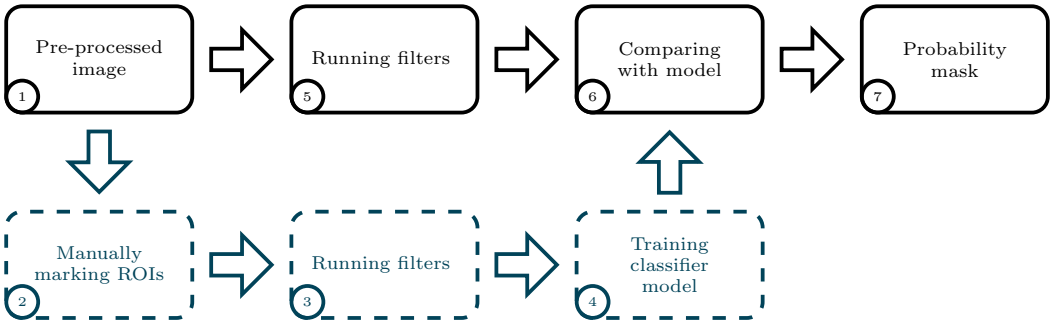


FIGURE 4.4: Machine learning algorithm. Based on partial manual segmentation, a classifier model is trained and then applied to the whole image.

TWS uses its own FastRandomForest[4] with 200 trees and 2 random features per node. The application of the machine learning algorithm involved the creation of a classifier model which had to be trained by using reference data and filters, as outlined in Figure 4.4. The algorithm could then determine the likelihood of each voxel of further images (or further parts of the same image) belonging to the feature of interest.

As values of the voxels are related to the sample's X-ray absorption data, any pre-processing methods affecting the histogram will also lead to loss of information within the image or relative to other scans. The scans available differed in their greyscale resolution and in the applied reconstruction algorithm. Therefore, no useful normalization or equalization could be performed. This loss of information underlines the importance of standardized sample preparation and scanning procedure to produce comparable images. With a defined reconstruction algorithm, comparable images with equal pixel depth can be produced. In such a case, normalisation across different scans is possible and allows for the application of a trained algorithm across different images. No noise reduction filter was applied apart from a Gaussian blur prior to other feature detecting filters (outlined at the end of this section).

Reference data was provided in the form of manually segmented images. For large parts of the image, it was not possible to clearly differentiate and manually segment the vessel wall and surrounding tissue. The progression from airways into alveolar areas is fluent. Only areas that could clearly be identified were used for the training to avoid training with false data. This means that there is a bias towards vessels that can be clearly identified. Areas that represent a mix between alveolar areas and airways are not recognized reliably. The small sample size was compensated using data augmentation (Raj, 2011). We performed 3D rotations on the datasets to increase the number of datasets without reducing the quality of the training data. Two airways and two blood vessels were selected as the foreground, and four representative areas that were not of interest were selected as background. Whilst selection of more training data can improve the classifier model, too much training data, in this case, resulted in over-segmentation due to the difficulty in accurately selecting training data manually. A selection of a wall including the inside area gave better results, as it included edges which are easier to identify than other features when using filters. The inside of a vessel matched the background, as both were filled with wax and therefore had the same density (see Figure 4.5). This caused an over-segmentation of the background, which was dealt with during the post-processing detailed in the following section.

[4]Available at: https://code.google.com/archive/p/fast-random-forest/ (last accessed 20/12/2018)

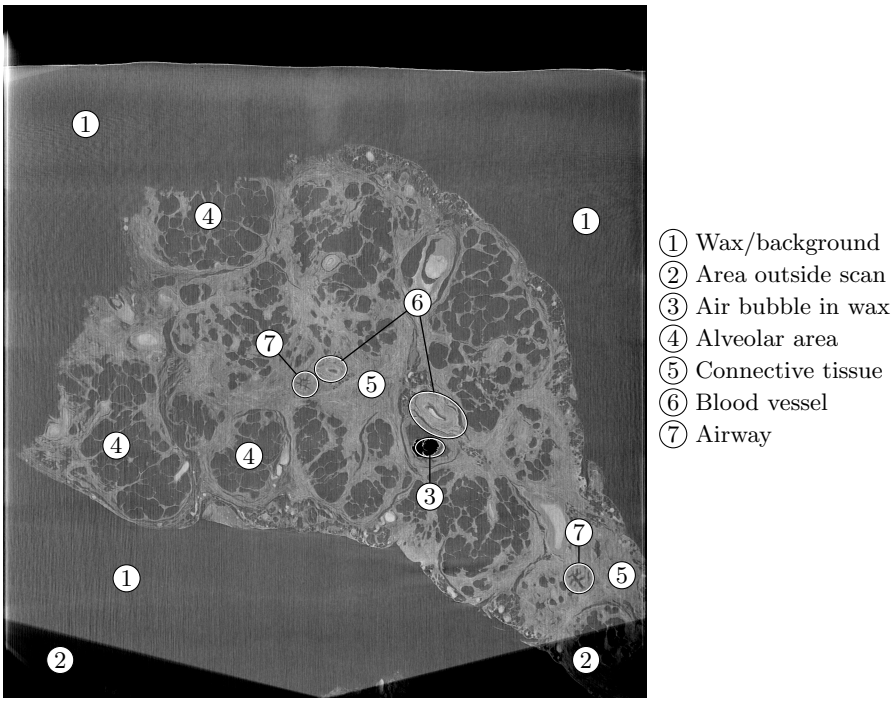
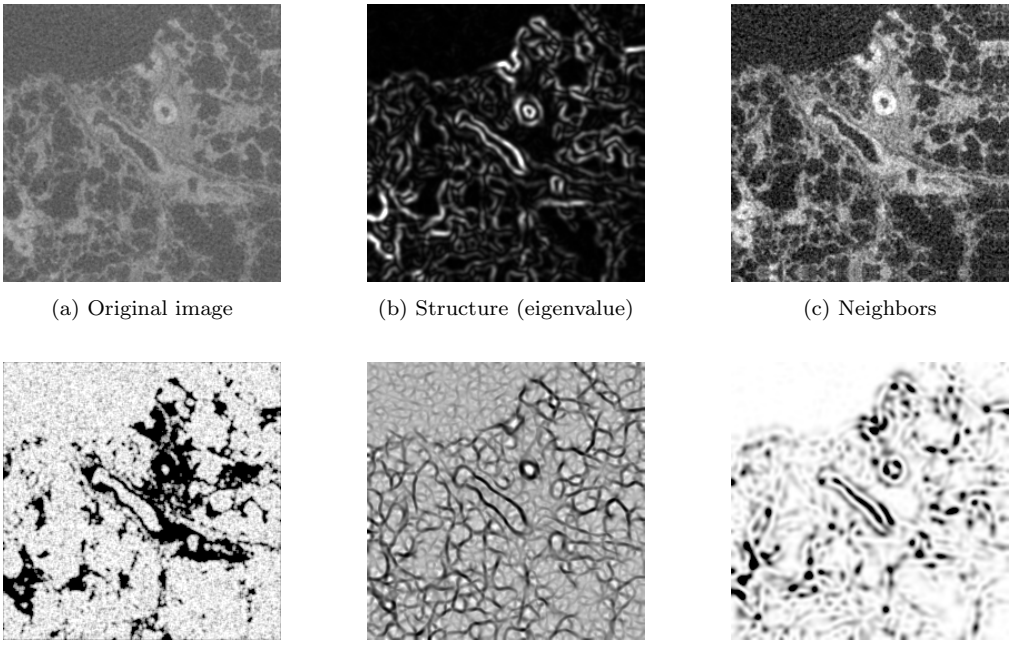


FIGURE 4.5: A single section of a µCT scan of wax embedded human lung tissue showing areas of wax, area outside the scan and distinct features of interest. Features 5, 6 and 7 were targeted for this project.

The TWS provides a set of image filters, and a selection of these was chosen based on the features to detect. Features of interest are shown in Figure 4.5. This project focused on detecting blood vessels and airways—key structural elements of the lung. This method can also be used to detect other features such as fibrotic structures, as will be shown later in Section 5.2. Vascular structures are characterized by a bright (X-ray dense) surrounding and a dark (X-ray lucent) centre. These attributes suggested the use of filters targeting edges. The choice of filters was based on literature review as well as screening of all available filters with a representable image as shown in Figure 4.6 and Appendix C. Filters such as 'Structure', 'Neighbors', 'Entropy' and 'Hessian' were able to separate areas of interest from the rest of the image. The Hessian filter highlights edges and was proposed for tube-like structure detection by Sato et al. (1998). The use of 'Entropy' filters for CT images has been encouraged by Bae et al. (2005).

Next, a classifier model was created. The reference segmentation, as well as the choice of filters, were provided to the TWS, which trained a classifier model.

Using the TWS fast-random-forest algorithm and 'Structure', 'Neighbors', 'Entropy', and 'Hessian' filters, it was possible to train a classifier model from a manual segmentation. This model was used to segment the whole µCT scan. The result was a probability map, where higher values represent a larger likelihood of a voxel being an airway or a blood vessel.



(a) Original image (b) Structure (eigenvalue) (c) Neighbors

(d) Entropy (e) Hessian (eigenvalue) (f) Hessian (orientation)

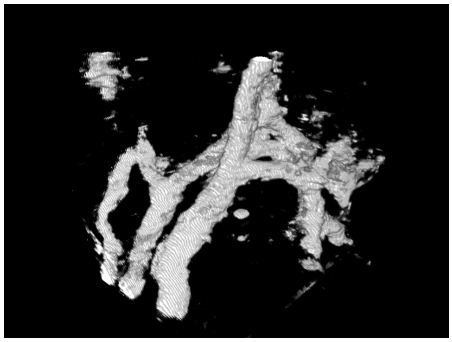
FIGURE 4.6: Result of different filters applied to an image: A slice of the original image and the result of the application of different filters are shown. These were the filters chosen for the classifier.

4.2.4 Image Post-Processing

The probability map was converted into a binary image (mask), where each voxel was given a value of 0 or 1, representing areas of no interest and areas of interest, respectively. This was done by manually defining and applying a threshold. This mask contained the vessel walls as well as some areas surrounding both the inside and outside of the wall. To remove the voxels from the mask which were not part of the vessel walls, first, the mask was applied to the original image. Then, another threshold was applied to remove the darker areas representing wax. This threshold was again chosen manually and a second mask was produced.

This second mask contained some undesired regions, such as air bubbles or noise. While erode and dilate operations are useful tools that remove noise (Antonelli et al., 2005), they affect the surface of correctly identified regions. Erosion of a mask removes all voxels within a radius from the mask's surface, while dilation adds voxels within a certain radius from the mask's surface to the mask. For both operators, the radius is usually equal to the size of one voxel. The combination of one erode operation and one dilate operation of equal radius in that order is referred to as '*opening*', and the opposite action of dilating before eroding is referred to as '*closing*'.

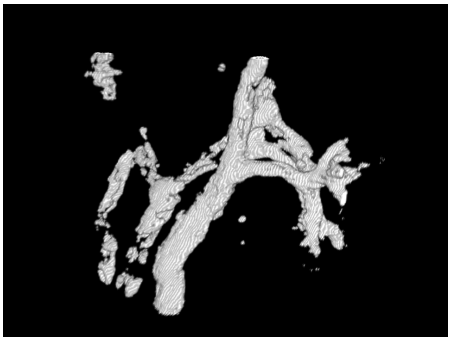
Erode and dilate algorithms have been previously used in medical image processing (Hu et al., 2001). The effect of applying them in comparison to an algorithm identifying and removing small connected regions has been studied. Results obtained for the minimum region size of the connected region algorithm and for two different implementations of erode and dilate functions are presented in Figure 4.7b and Figures 4.7c and 4.7d, respectively. Figure 4.7c used a modified version of the algorithm presented by Antonelli et al. (2005), which has been slightly optimized using the idempotent nature of the operators, whilst the others were based on a set of trial runs.



(a) Original mask

(b) Connected region algorithm with minimum region size 700

(c) Opening (erode, dilate) followed by closing (dilate, erode) operation



(d) Closing, opening and radius 2 opening operation

FIGURE 4.7: Difference between connected regions and erode/dilate algorithm for noise removal. Neither erosion nor dilation removed a lot of noise (c) or visibly removed part of the structure of interest (d).

It was found that algorithms for removing small connected regions reduce the loss of shape but are multiple times more time-consuming (see Table 4.2), adding about 1 hour to the total processing time. Noise removal through the removal of 'small connected regions', as shown in Figure 4.7b, gives better results than opening and closing operations, shown in Figures 4.7c and 4.7d. The more sophisticated algorithm leaves the actual boundary/surface of the vessels untouched while erode and dilate operations affect both noise and valid segmentation. For the large connected vessels, erode and dilate operations are not strong enough for effective noise removal.

After post-processing of the map, only the vessel walls were left and the noise had mostly been removed.

4.3 Discussion

Current histopathology relies on limited 2D analysis of tissue samples that are intrinsically 3D and are affected by local challenges and pathological changes that may be highly heterogeneous at several different length scales. The use of 3D scanning techniques to accompany current histology procedures can improve insight into tissue structures and accordingly the understanding and diagnosis of chronic lung diseases. The workflow for segmentation of multi-gigabyte µCT scans of lung tissue presented in this chapter demonstrates this alternative method.

TABLE 4.2: Timings for two different µCT images on two different machines. Image 1 was 1600 × 1600 × 1200 px in size at 16 bit and Image 2 1374 × 1497 × 500 px at 8 bit. Machine 1 had specifications of 16 GB RAM, 64 bit, Intel Pentium CPU G2020T @2.50 GHz and a local hard disk drive (HDD); Machine 2 had specifications of 16 GB RAM, 64 bit Intel i7-4770 @3.40 GHz and a network drive. Two runs were done and averaged for image 2 and one for image 1.

Image	**Computer**	**Blocks**	**Time (h:mm)**					
			Block Creation	**Segmentation**	**Thresholding**	**Erode & Dilate**	**Connected Regions**	**Total**
1	1	245 (no halo)	0:01	44:11	0:04	0:01	0:52	44:17 to 45:08
2	1	72 (16px halo)	0:00 (9 s)	17:59	0:01	0:01	1:14	18:01 to 19:14
1	2	245 (no halo)	0:19	11:38	0:16	0:17	0:42	12:30 to 12:55
2	2	72 (16px halo)	0:04	5:05	0:06	0:06	0:47	5:21 to 6:02

It was found that splitting large µCT images into smaller blocks permits handling of 3D images without requiring unreasonably powerful computers. At the same time, it enables fast testing of a segmentation strategy. Trainable image segmentation using 'Structure', 'Neighbors', 'Entropy' and 'Hessian' filters provided good results with vessels being clearly distinguishable.

Tests with five images showed that, currently, the training needs to be redone for every image in order to be effective. Two of the images were used for timed segmentation and the results are presented in Table 4.2. Other images are expected to give similar results. Currently the classifier had to be trained on a small part of each image. TWS generally allows classifiers to be applied accross different images. A more extensive study with identically acquired and pre-processed images would give better insight into the possible transferability of a trained classifier.

The connected regions algorithm was shown to be the more accurate noise removal algorithm compared to erode and dilate operations. The workflow was implemented in the popular image processing software ImageJ in a modular fashion, which allows for optimization of individual parts of the workflow.

Using this method, tubular structures were successfully segmented from an embedded tissue sample without making assumptions about the 3D structure beforehand. It was possible to apply the processing algorithm to images of many tens of gigabytes in size without reduction in the processing speed (see Figure 4.3). The total processing time was reduced from months of manual segmentation (see Section 1.1) to between half a day and two days of semi-automated segmentation (see Table 4.2). The results clearly identified a 3D network of vessels.

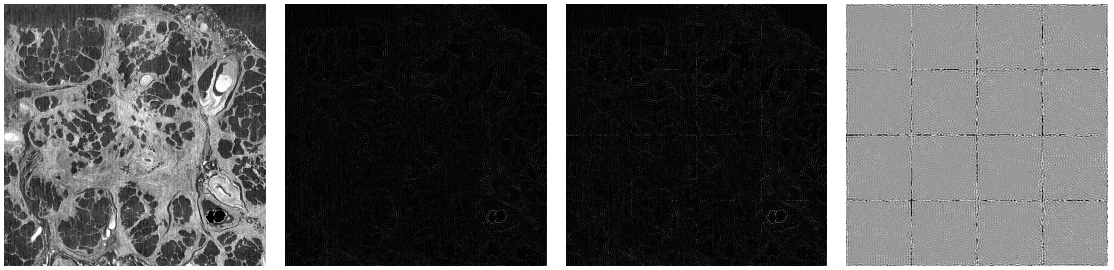
Another advantage of the proposed method is its implementation as a modular workflow in open source software. This allows individual segments of the image processing to be improved and optimized as required without having to revise the whole protocol. User input for the training of the algorithm allows flexible, interactive selection of features of interest. Reduction in the number of user steps required to operate LungJ allows much easier application of this method in medical research. The workflow has been designed to aid users and provide feedback of success.

As no other workflows for this task are known to us, evaluating the method empirically is difficult (Zhang, 1996). Method evaluation can be achieved using unsupervised image segmentation evaluation methods but is less meaningful (Zhang et al., 2008). A modular approach enables the proposed workflow to be adapted to other types of tissue or image conditions. Having this workflow in place shifts focus towards the efficiency of individual modules in future.

4.4 Generalisation of the LungJ Method

In Section 2.2.6, we introduced different groups of processing methods. As the name indicates, LungJ was initially designed for the segmentation of lung tissue. As demonstrated in previous sections of this chapter, LungJ can be used with machine learning algorithms. Machine learning includes other image processing algorithms (see Section 2.2.6.3), and we have used spatial filters as part of machine learning for this application.

LungJ splits the image into smaller regions and therefore part of the global information is lost when processing an image using LungJ. The approach used by LungJ can be generalised, and in



(a) Original image

(b) Image after applying an FFT high-pass filter

(c) Image after applying an FFT high-pass filter using LungJ blocks

(d) Difference between (b) and (c).

FIGURE 4.8: Application of a high-pass frequency filter to an image (a) with and without using LungJ. In (b) the high-pass filter is applied to the 1024×1024 pixel image. In (c) the high-pass filter is applied to individual squares of 256×256 pixel. As can be seen from the difference in image (d), the result of the filter is affected by the splitting process of LungJ.

this section, we will go through the different segmentation methods to clarify how each of them can be addressed.

As part of the filter-based algorithms introduced in Section 2.2.6.1, the first two methods presented were in-place filters and spatial filters. In-place filters only rely on basic global information, such as the global maximum pixel value. These values can be computed before or while splitting the image. The presented implementation of LungJ will calculate the global maximum and minimum as well as the mean pixel value and the standard deviation. This information is saved and can be provided to the algorithm. As long as the code does not read this information from the loaded image automatically, no modification is required to use LungJ with in-place filters. Spatial filters additionally require information from the neighbouring pixels. In order to provide the neighbouring pixels at the boundary of an image block, halos were introduced to LungJ in Section 4.2.2.

The last filter-based algorithm presented in Section 2.2.6.1 was frequency filtering. Frequency filters depend on variations in pixel values occurring over the whole image. They require the transformation of the image into a frequency domain prior to splitting the image and applying the filter. The frequencies are global properties and need to be computed for the whole image. Due to the separability of the transform, the application of a frequency filter is computationally inexpensive. Since the transformation involves the whole image, it requires a lot of memory and cannot be split as shown in Figure 4.8. This means that LungJ cannot process images using a frequency filter in its current form. Instead, the frequency transform needs to be applied prior to the splitting into LungJ blocks. Concatenation of the blocks needs to happen before performing the inverse transform.

Iterative methods (see Section 2.2.6.2) are slightly more complicated to implement with LungJ. The image can be split into blocks before the processing. Halos are required and need to be exchanged after every iteration or a given number of iterations. This means that LungJ can only be used with existing iterative ImageJ functions, which allow the iterations to be run individually.

Frequency filters rely on large amounts of global image information. They require an approach different to LungJ to reduce their memory requirements. For all other image processing algorithms, the approach of splitting the image into small sub-images can be applied.

4.5 Summary

In this chapter, the problem of processing extra-large images has been solved for the segmentation of images. We looked at histopathology as a particular use case and implemented code to apply machine learning to large µCT scans. We used a popular open source software for implementing our solution. The main code, LungJ, was kept generic so it can be applied to other image-processing scenarios and methods.

With this approach to image segmentation, a further part of the end-to-end workflow has been achieved. Collected images can now be enhanced and segmented to extract further information from them. Similar methods can be developed in future to support image recognition and image registration of large images to cover all aspects of image processing (see Section 2.2). The next chapter will look at how to present the extracted image information to a medical researcher.

Chapter 5

Image Visualisation

The work described in this chapter has been published as:

Wollatz, L., Konieczny, K., Vandervelde, C., Mitchell, T., Cox, S., Burgess, A., Ismail-Koch, H., Sep. 2016. 'The use of 3D-printed paediatric temporal bones as a training tool'. *In: BAPO Annu. Sci. Meet. (BAPO 2016)*. British Association for Paediatric Otolaryngology, Liverpool, United Kingdom, p. 24

Wollatz, L., Cox, S. J., Johnston, S. J., Sep. 2015. 'Web-based manipulation of multiresolution micro-CT images'. *In: Proc. 11th Int. Conf. eScience (eScience 2015)*. IEEE, Munich, Germany, pp. 308–311. doi:10.1109/eScience.2015.42

We developed a new approach for the processing of large images in the previous chapter. This allows medical researchers to take images from the management system discussed in Chapter 3 and process them from the raw digital format into a segmented digital image. So far, we have only considered images as digital files. As discussed in Section 1.2, the visualisation of numbers stored on a hard drive, so that they can be interpreted by the human eye, is the last remaining part of the medical image life-cycle. In this chapter, we will look at the presentation of digital image data to a human. This includes aspects of fast representation methods for large images, as well as alternative visualisation methods that can help humans understand multi-dimensional data obtained from the image files.

In Section 2.3, we looked at established visualisation methods in 2D and 3D, namely variations of 2D sectioning and 3D volume and surface rendering. We also looked at holograms, VR and AR for giving the eye a better perception of 3D objects retrieved from 3D scans. Further, we talked about 3D printing as a new fast manufacturing technique and how it can be used to the advantage of doctors in medicine.

This chapter will implement a selection of these visualisation techniques for three example cases. Section 5.1 will integrate 2D sectioning and 3D surface rendering into the website developed in Chapter 3. Alternative 2D and rendered 3D visualisations are implemented in Section 5.2 for local visualisation based on the image processing software mentioned in Chapter 4. Section 5.3 uses a new case study of paediatric surgery, to present the use of 3D printing and AR in medicine. A summary is provided in Section 5.4.

5.1 Web Visualisation

µCT data and the resulting processed data need to be available to medical researchers in an accessible and meaningful way. In Chapter 3, we developed a management system to ensure easy access of data and management of metadata, but we have not addressed visualisation of stored images yet. Fast previewing of images is critical, in order to allow researchers to decide which images are worth analysing before retrieving the full image data from a remote storage onto their system for processing. Commonly, medical images from a management system are previewed in a browser but, for large images, the use of browser-external software is still the primary solution (Allan et al., 2012; Engel and Ertl, 1999; Kvilekval et al., 2010; Trotts et al., 2007).

In this section, we will look at methods for image visualisation over the web and develop our own image viewer. We will then implement the viewer into Mata[1] from Section 3.2. As mobile devices are of increasing importance in today's world and also within the field of medicine, compatibility of applications with mobile devices is a key concern (Castiglione et al., 2015; Johnson et al., 2012). We will therefore focus on an implementation that can work on devices with limited processing capabilities and is less affected by slow internet connections. We will also focus on common features of CT visualisation software. Medical researchers who worked with CT visualisation software such as Siemens syngo fastView[2] expect tools to measure distances in the displayed image and adjust the image gray-scale (Kagawa et al., 2006).

For non-medical images, the display of very large images by the use of tiled images is a well-established method. Sending full-scale images over the web can take a long time. It is therefore advantageous to send only the necessary parts of the image. In order to quickly select these parts without further image processing, the image is saved in tiles. Each tile is of a reasonably small size (e.g. 256×256 pixels). Only those tiles which are visible are loaded onto the client's computer (dark tiles in Figure 5.1). It is important to balance the tile size between the amount of data to load from the server and number of server requests made (Gerhard et al., 2010).

If the user zooms out, the whole image is displayed on the screen. As this means that all the tiles need to be loaded, another technique is needed—the use of multi-resolution images. The image is scaled to several different resolutions and tiled at each of these resolutions. The result is a structure as shown in Figure 5.1. For images where the overhead from multiple tile requests is too large, loading multiple resolutions will still provide a faster response experience. The primary objective of the viewer is not to speed up the loading, but to limit the amount of work which needs to be done. In the case of image manipulation, reducing the amount of data to edit means reducing the footprint of the application.

In this section, the MCTV[3], an implementation of a multi-resolution tiled image viewer based on the Brain Maps API (Mikula et al., 2007), is described. It makes viewing images and processing the pixels on low-performance devices within the web browser possible. This improves the capabilities of existing web-based PACS and the practical usability of remote data storage for medical research. Using Portable Network Graphics (PNG) image tiles instead of commonly used

[1] Source code and documentation published as doi:10.5258/SOTON/D0430 (Apache 2.0 License)

[2] Official site: https://www.healthcare.siemens.co.uk/medical-imaging-it/advanced-visualization-solutions/syngo-fastview (last accessed 20/07/2018)

[3] Source code and documentation published as doi:10.5258/SOTON/400332 (Apache 2.0 License)

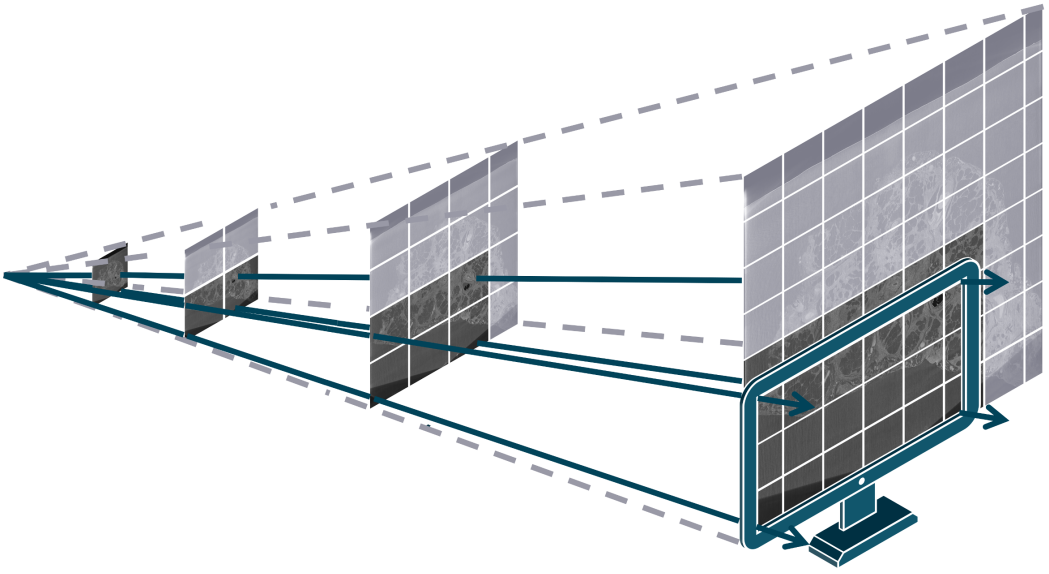


FIGURE 5.1: Multi-resolution tiled image: Each resolution level is four times the size of the previous. Tiles are loaded from the server for the current view only (dark).

Joint Photographic Experts Group (JPEG or JPG) files as tiles, better results were obtained, as shown in Section 5.1.2. Requirements for speed improvement are discussed in Section 5.1.2 and integration into an image server as well as the performance of the code are presented in Section 5.1.3.

5.1.1 Background

Image tiling is commonly used for image display on the web. Most people will know it from Google Maps, where a map of the entire earth is displayed over various zoom levels with individual tiles being loaded on demand. Another example is Deep Zoom, which is based on Silverlight plug-in. Deep Zoom was used to display 2 PB of information from Bing Maps (Pendleton, 2010). Deep Zoom was also applied to display medical images (Crichton et al., 2008).

In the medical field, the implementation of tiled image viewers is also being established (Arguiñarena et al., 2010; Mikula et al., 2007). This solution is limited to displaying images, however, and does not support image modification.

Dcm-Ar is an Adobe-Flash-based image viewer (Arguiñarena et al., 2010). To address the problem of slow image transmission from web-based PACS systems, Dcm-Ar was specifically designed to handle DICOM files a few tens of megabytes in size, as explained in Section 2.4. Dcm-Ar is split into a server, which can connect to an existing PACS, and a client-side application which displays the images to the user. The viewer handles 3D images using multi-planar reformatting. In order to load several images at a time, thumbnails are loaded and full images are only requested for a full-screen view. Besides being used for small images only, the concept of Dcm-Ar genuinely allows scaling to larger images with the limitation of long loading times for full resolution display.

The three-dimensional internet imaging protocol (IIP3D) (Husz et al., 2012) provides much better scalability than Dcm-Ar through the use of a special file format. This format called Woolz enables efficient access of specific regions of an image. With the help of Woolz, IIP3D is able to display 3D images of over a hundred GB in size at arbitrary angles. The calculations for the 3D rotations are done server side. The server then sends the requested tiles to the client viewer via asynchronous JavaScript and XML (AJAX). The computations require IIP3D to be run on a dedicated server. The advantage is that it is very scalable and can be extended through the use of a proxy server. Downsides of IIP3D include the lack of grayscale adjustments, which means that images can only be viewed at the original brightness. IIP3D is used by projects such as the eMouse Atlas Project (EMAP) (Richardson et al., 2013).

The Multiresolution Image Viewer (MIV)[4] (Mikula et al., 2007), later renamed to Brain Maps API[5], was originally developed for displaying high-resolution brain images. These images are taken with a microscope with a very high resolution compared to that of a µCT scan. Even though microscopes only allow a small part of the sample to be viewed at high magnification, methods exist to stitch these images back together to form a complete image of the sample (Appleton et al., 2005). After being stitched together, the image is converted into a multi-resolution, tiled image and stored in a predefined folder structure. Brain Maps API loads and places the image tiles needed for the current view.

The capabilities of the MIV were further extended by StackVis (Trotts et al., 2007), which is a standalone software but enables the display of coarsely spaced 3D image stacks and 2D images from a bird's-eye view at an arbitrary angle.

As Brain Maps API is open source, it is ideal for adaptation to any scenario. It was therefore decided for it to be used as a base for a CT viewer implementation.

5.1.2 Methods

Raw data can be stored in various ways, such as uncompressed bits or in a file format such as Tagged Image File Format (TIFF). The tiles for the image viewer need to be stored in a web-compliant format. Modern web-browsers support only standard 8 bit images natively—commonly JPEG, Graphics Interchange Formats (GIFs), PNGs, and Windows Bitmap (BMP) files.

TIFF allows storing 16 bit or 32 bit data as well as storing multiple images in one file, which is not possible with common image formats supported in the web. Unlike raw data files, it also presents a standard for storing keywords related to the image. As standard TIFF uses 4 B pointers, it can only support file sizes of up to 4 GB. Images larger than that can be stored as image stacks of several separate 2D TIFs. The JPEG format can be seen as a standard file format for tiled image viewers (Chui and Wang, 2005; Gerhard et al., 2010; Jeong et al., 2010; Trotts et al., 2007). However, server-side solutions also use their own multi-resolution file formats or compression methods (Gormish et al., 1997; Jeong et al., 2010). Deep Zoom supports JPEG, PNG and BMP files (Gerhard et al., 2010; Prosise, 2009). GIF files use an 8 bit indexed colour map, which allows for very high compression but has a low colour depth. They are not

[4]Available at: http://brainatomics.com (last accessed 03/08/2018)
[5]Official site: http://BrainMaps.org (last accessed 13/02/2018)

appropriate for displaying images including gradients but are useful for schematic or '*cartoon-like*' images (Miano, 1999). Lossless compressed JPEG files provide better compression than PNG files for untiled images but because of the overhead in small images produce tiles that are 5 % larger. The JPEG2000 format, which is becoming increasingly popular in medicine, applies partial lossy compression (Clunie, 2000). This section compares lossless compressed PNG, lossy compressed JPEG, and the uncompressed BMP format. The comparison is made in terms of file size and quality loss due to conversion and compression.

The original µCT images scanned by the µ-VIS X-Ray Imaging Centre at the University of Southampton were reconstructed at 32 bit and 16 bit. This was proven to be of sufficient quality for preserving features for medical purposes (Scott et al., 2015). TIFF allows greyscale resolution of up to 64 bit per channel. Standard image formats, such as BMP, PNG or JPEG, are based on 8 bit integers for greyscale ranging from 0 to 255. A total of 24 bit are used per pixel, since colours are encoded as three individual 8 bit RGB channels. A PNG32 format exists but only adds transparency to the 24 bit PNG format.

Radiologists can discern 720 to 917 different levels of luminance within one scene, corresponding to 720 to 917 individual grey shades. This value assumes variable visual adaption and is therefore an upper limit (Kimpe and Tuytschaever, 2007; NEMA, 2014b). This means that a 10-bit image format with 1024 grey shades is the minimum requirement for accurate display (Blume and Muka, 1995; Robb, 2000). A visible loss in intensity resolution therefore occurs, independent of the choice of the file format for tiling.

For normal display, the difference is very small, but one of the requirements established in Section 5.1 is the change of the density range displayed. This means stretching the histogram. The resulting losses are much greater, as illustrated in Figure 5.2. Two possible conversions to 8 bit PNG are compared: The upper row shows the output if the histogram is first converted to 8 bit and then stretched, while the lower row shows the histogram stretching at 32 bit in TIFF, followed by a conversion to 8 bit PNG. The clipping of the images on the client side after conversion to 8 bit is very lossy and has a very limited use for diagnostic purposes. A small improvement can be achieved by pre-analysing the original histogram as shown in the lower row of Figure 5.2. In the example, the original image uses only a small range of the available greyscales. A partial clipping can be done before the conversion in order to make better use of the limited 8 bit palette. As the images are in 3D, this pre-stretching has to follow the global 3D-histogram and can be applied equally to all image slices. This agrees with the comparatively good quality of the bottom right histogram.

Besides the quality loss due to conversion to 8 bit, the compression methods of the different image formats play a role. Commonly, the different image resolutions vary by a scaling factor of 2, meaning that each zoom level contains only a quarter of the number of pixels as the previous one. This means that, in total, the tiled image will contain at most 4/3 of the number of pixels, as

$$\sum_{n=0}^{N} \frac{1}{4^n} \leq \frac{4}{3}.$$

The overhead of having several files instead of a single one adds to this. JPEG uses a 2D frequency transform to convert the image into the frequency domain and only stores the most important frequencies (Nixon and Aguado, 2012, Chapter 2). This allows for much smaller file

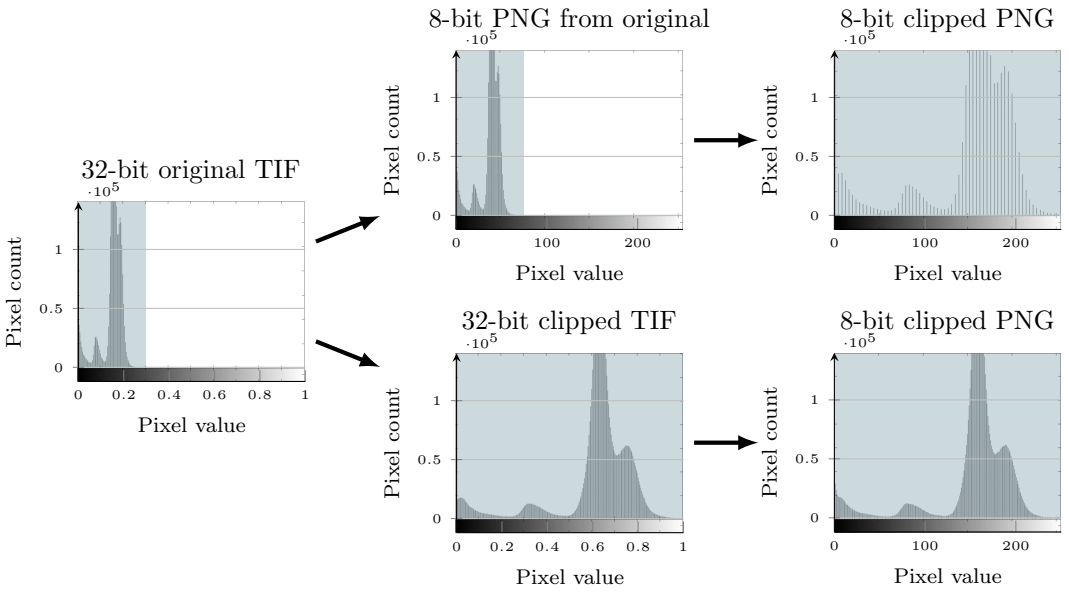


FIGURE 5.2: Quality of an image if first stretched and then compressed compared to first being compressed and then stretched. The number of grey shades visibly reduces. The image used for creating the histogram can be seen in Figure 4.5 on page 66.

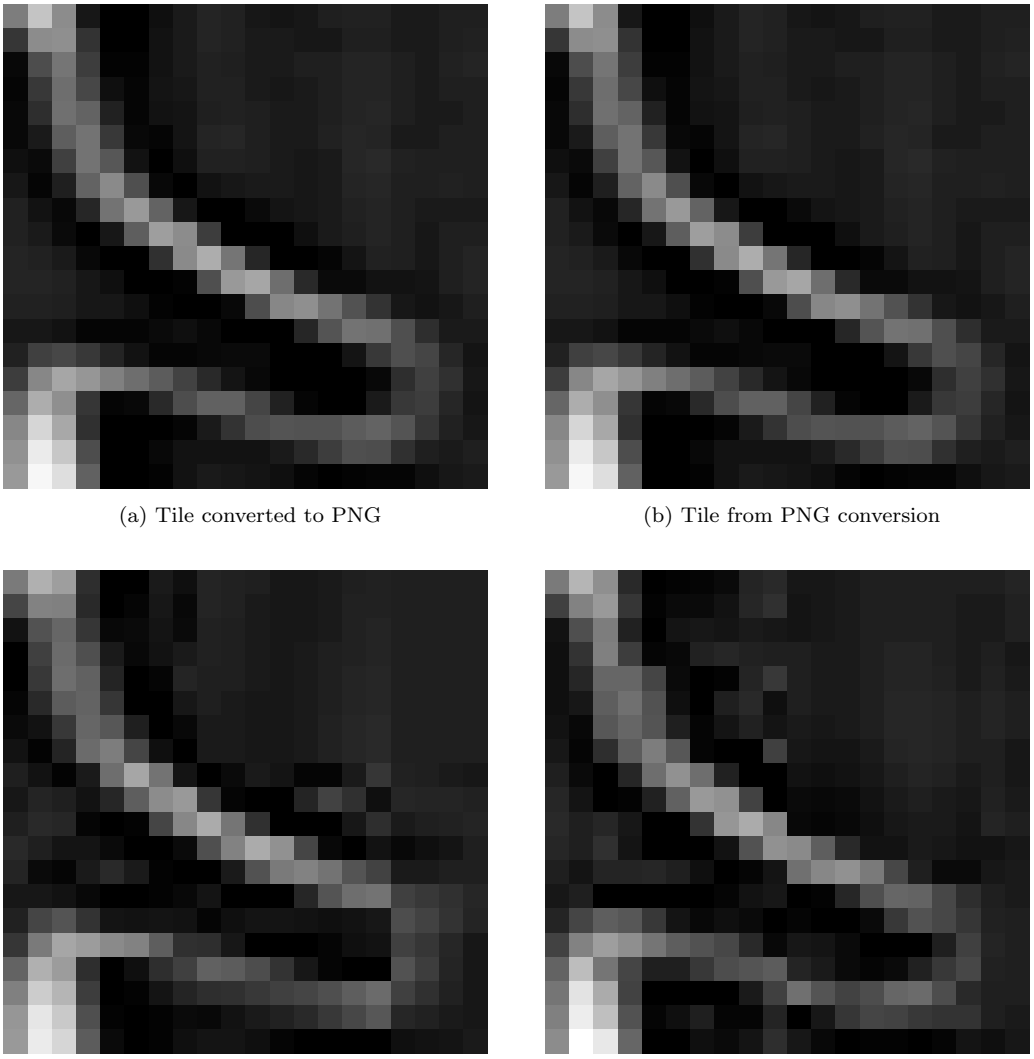
TABLE 5.1: Size of tiled images compared to original untiled TIF.

Image	**Orig. TIF**	**Tiled JPEG**	**Tiled PNG**	**Tiled BMP**
Sample 1 (32-bit)	8.13 GB	138 MB	855 MB	2.89 GB
Sample 2 (16-bit)	7.09 GB	265 MB	1.79 GB	5.01 GB
Sample 3 (16-bit)	60.8 GB	2.1 GB	9.53 GB	21.1 GB

sizes of lower quality. PNG uses lossless compression and BMP uses no compression or run-length encoding (RLE) compression only. Figure 5.3 shows the difference in terms of image quality. Due to the post-processing required, not all differences can be seen, but an increased number of artefacts for JPEG is visible. Lossless formats do not show major differences from the original image, even for large image scaling. The root mean square (rms) of the difference between the different images and the original TIF were computed for the selected tile. Both BMP and PNG resulted in an rms of 0.005 while converting the image to JPEG and then tiling it gave an rms of 0.0144 and tiling the image prior to the conversion resulted in an rms of 0.0134.

On using lossy JPEG compression, the image size of the multi-resolution image shown in Table 5.1 was smaller than the original TIF by a factor of 14. This excludes the effect of the number of bits per pixel. This is much less than that reported for normal images mainly due to tiling and the increased number of pixels of the multi-resolution image (Norcen et al., 2003). The PNG format achieved the expected compression ratio of a factor of 3 (Cosman et al., 1994).

If the increased amount of data transfer results in an unacceptable delay of the time until a webpage displays, PNG cannot be used for the tiled images. Otherwise, it is preferred, as it offers a much higher image quality compared to that of lossy compressed JPEG files. As the



(a) Tile converted to PNG

(b) Tile from PNG conversion

(c) Tile converted to JPEG

(d) Tile from JPEG conversion

FIGURE 5.3: Quality of JPEG compression compared to PNG based on a 20×20 pixel tile. Images on the left are a result of the TIF being tiled and then converted to the new file format. Images on the right have been first converted and then cut into smaller tiles. The JPEG images at the bottom show a lot of errors. The difference between the PNG images and the original TIF are too small to be visualised.

original data size is several tens of gigabytes, the larger overall data size of PNG tiles compared to that of JPEG tiles is not of major importance.

The speed of websites is a major issue (Gehrke and Turban, 1999). Server-, network- and coding-related improvements were explored in order to ensure short website response times.

The processing/rendering of the images should occur on the client side. For comparison, an AJAX-based loading of the images was implemented, allowing for server-side image processing. This proved to be much slower than client-side processing through pure JavaScript. This agrees with the observations regarding Deep Zoom that server-side image generation is slow as it increases the load on the server (Prosise, 2009). This would not be an issue if powerful server were used. As explained in Section 1.3, the purchase of powerful servers is not realistic for an early-stage research project.

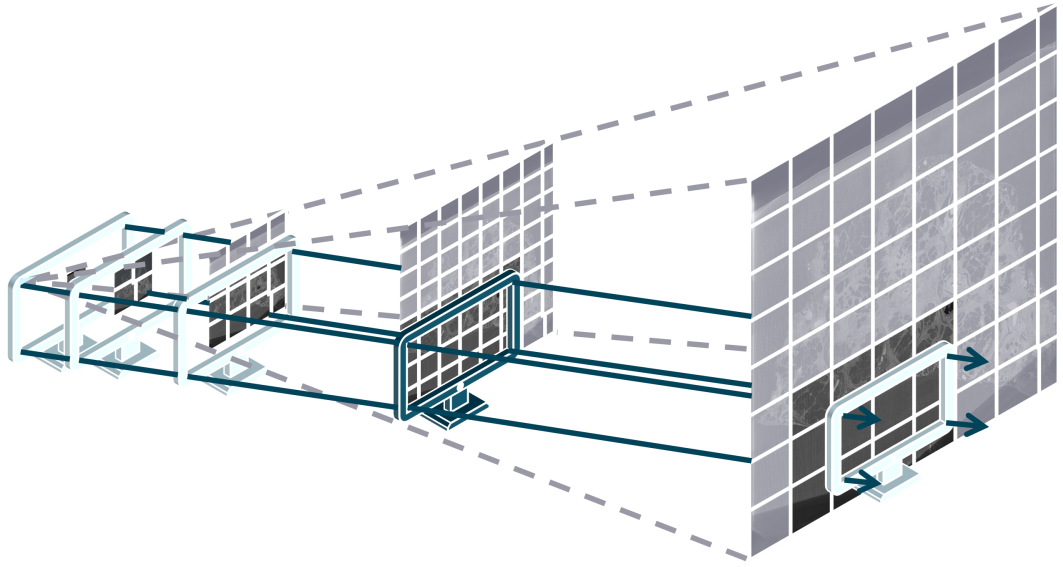


FIGURE 5.4: Zooming of a multi-resolution tiled image: for lower magnification, more of the image fits on the display, but since the tiles are always of the same size, the number of tiles to display (darker tiles from right to left) does not increase. The right-most zoom level shown goes beyond the image resolution and the tiles are scaled accordingly.

In order to decrease the page loading time, the JavaScript routines called on start-up were reduced to a minimum. The image loads on a comparatively small zoom level to keep the number of image files transferred on page load to a minimum (see Figure 5.4). For further speed-up, the code was modified to reduce the number of times tiles were recomputed and loaded (see Figure 5.5). A differentiation was made between the case of tiles needing to be updated and the case where the tiles visible might have changed. In the first case, all the tiles need to be recomputed. This occurs if the user changes the image manipulation parameters. The second case requires checking which tiles are within the field of view and loading those while removing the ones which moved out of the displaying area. This event needs to be triggered whenever the zoom level is changed or the image is panned.

As panning of the image is handled by observing the movement of the mouse, the event is triggered multiple times a second. Therefore, calling compute- or data-intensive tasks upon image panning is avoided. Images are only loaded after a drag event by the user has completed (see Figure 5.6). This allows much faster panning through the image. A low-resolution thumbnail (see Figure 5.7) was placed below the tiles in order to allow quick orientation while higher-resolution images are still loading. To create a better feeling of responsiveness, the thumbnail is always loaded before the tiles and placed in the background. Tiles then appear on top of the low-resolution image as they are processed. In the case of quick scrolling through the dataset, the low-resolution image loads much quicker than the tiles and improves the user orientation in the 3D stack.

Another change implemented was that the number of slices the users go through was made dependent on the zoom level and the amount of mouse-wheel movement. At higher magnification, the user moves only a single slice at a time, while at low resolution he can move 20 images at once. This way, it gives the feeling of moving through the image more quickly while reducing the number of in-between tiles that need to be loaded.

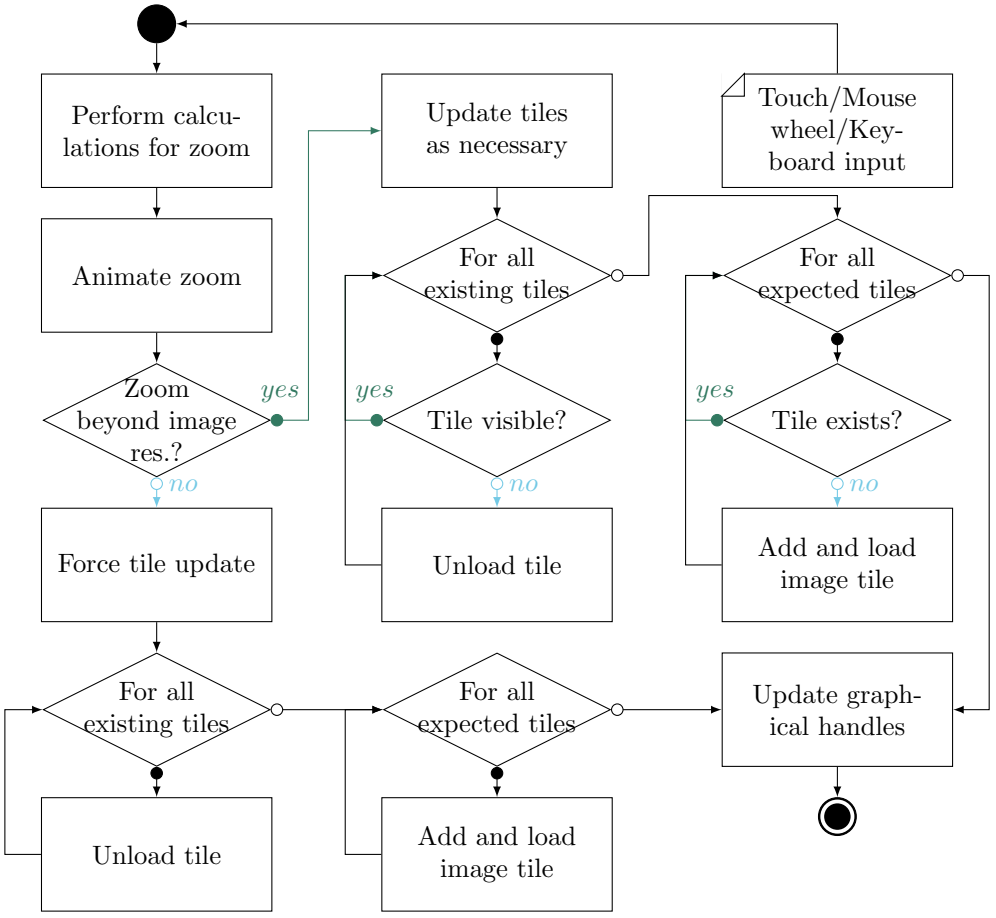


FIGURE 5.5: Flowchart to explain the zooming in MCTV.

5.1.3 Results of MCTV

In Section 5.1.2, we showed how to implement a multi-resolution tiled image viewer. As image files only contain information about the image inside the file itself, tiled image files contain no information about the overall 3D image. This metadata needs to be recorded either in a separate file or in a database, to be accessible to the viewer. For MCTV, image information is stored in a separate file that is requested by the viewer using AJAX. The pixel values are mapped back to the density of an object using the header file, as are the dimensions. The user can select a Hounsfield unit (HU) range to display within the boundaries of the scanned HU values, as shown in Figure 5.8.

For speed evaluation, the time to respond (TTR) and the time to display (TTD) were recorded. The average times are presented in Table 5.2. The TTR defines the time it took to display a preview after a user input while the TTD describes the amount of time it took till the full resolution image was displayed. Times were recorded for the scenario loading the maximum number of tiles for the device screen size. Each experiment was repeated at least 5 times to produce the average times and standard errors presented in Table 5.2. Results are shown for Opera and Firefox. The PC also allowed for comparison with Internet Explorer 11. The tablet used only supported Safari.

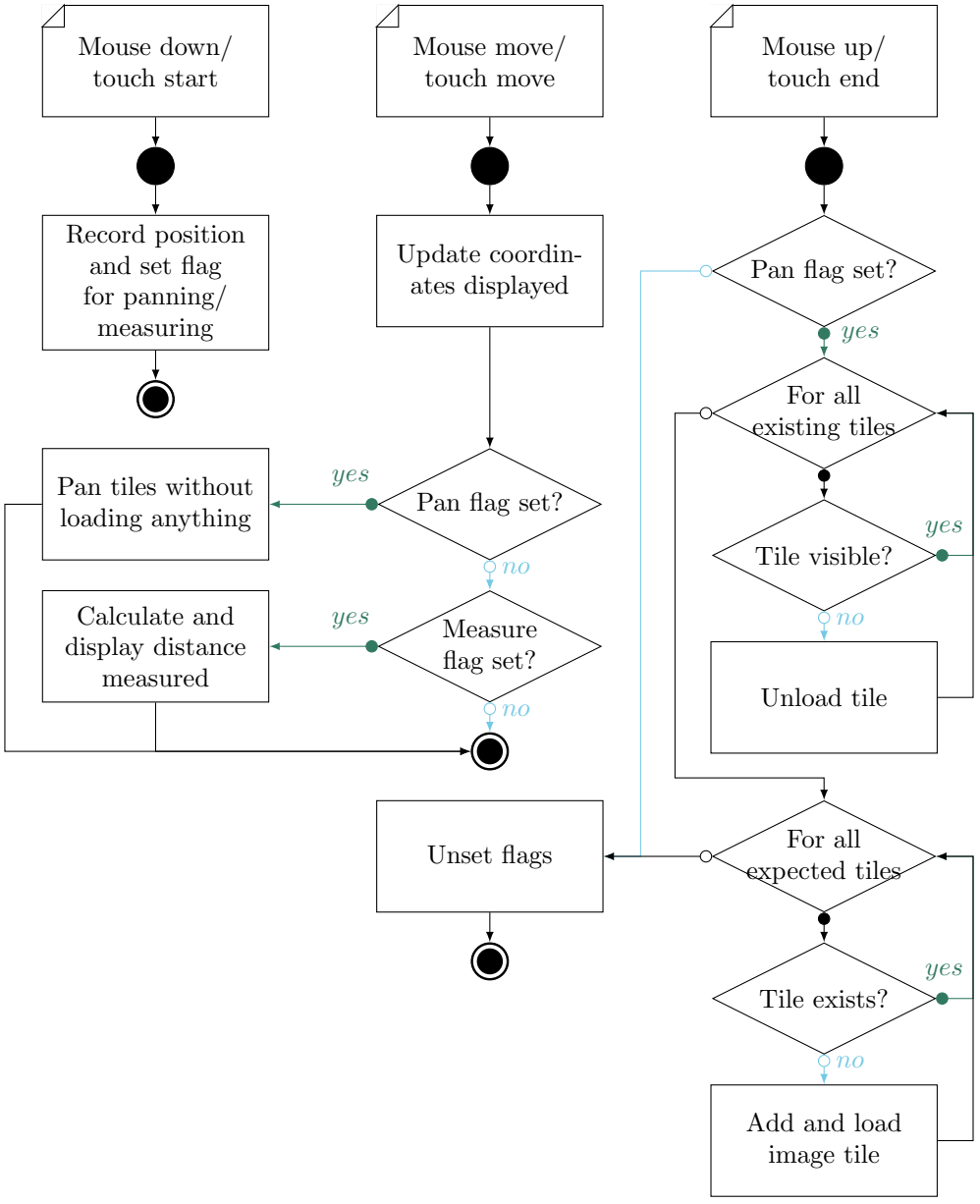


FIGURE 5.6: Flowchart to explain the panning in MCTV.

In general, TTD was 3 % to 4 % faster for JPEG tiles than for PNG tiles but 25 % to 60 % faster than for the un-tiled image. On mobile devices, the advantage of JPEG over PNG was much clearer, with a 16 % to 45 % speed improvement noted. Loading and editing the full image on a mobile device took 16 seconds on average, which decreased to 4 seconds on tiling. Due to multi-resolution loading, the TTR was much lower at 63 ms for the workstation and a fifth of a second for the mobile phone. For low zoom levels, the time to display new image slices averaged at 90 fps. Panning only requires loading a selection of new tiles and was even faster (around 6 ms).

The average speed of displaying a $2000 \times 2000 \times 2000$ voxel image does not differ from the speed of a $1987 \times 1850 \times 523$ voxel image as expected. Since the size of the tiles and the number of tiles that need to be loaded do not vary between images of different sizes, this approach ensures

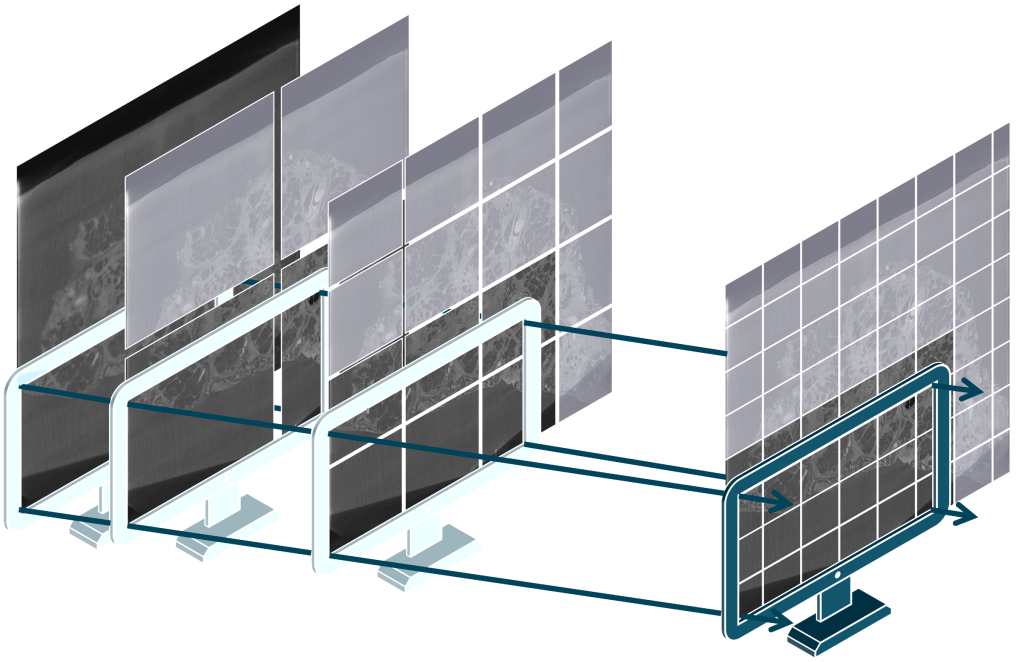


FIGURE 5.7: Loading a multi-resolution tiled image: it is possible to first load the lowest-resolution tile and then load higher-resolution tiles (darker tiles from left to right). The dark tiles are the same as in Figure 5.1. Therefore, all tiles are of the same size in pixels, meaning that the initial view will have a very bad resolution.

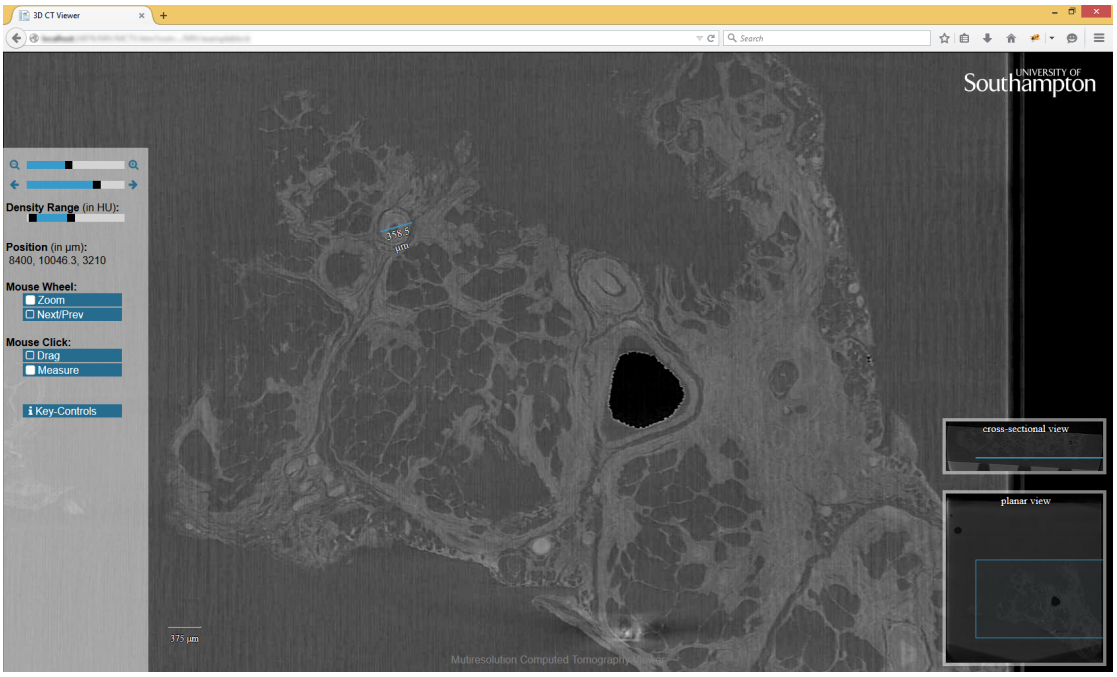


FIGURE 5.8: Screenshot of the viewer: a section of a lung tissue at twice the size of the raw data is shown.

TABLE 5.2: TTR and TTD using different file formats. The time to display the untiled PNG on the tablet was not recorded as the browser application crashed before finishing with the request.

Device	**Browser**	**Average**	**Average TTD [s]**			
		TTR [ms]	**JPG**	**PNG**	**BMP**	**Untiled PNG**
PC	Opera	37 ± 5	1.40 ± 0.11	1.46 ± 0.06	1.53 ± 0.07	3.51 ± 0.03
PC	IE	20 ± 2	1.61 ± 0.06	1.6 ± 0.4	1.64 ± 0.19	5.0 ± 0.2
PC	Firefox	94 ± 12	2.96 ± 0.17	3.06 ± 0.16	3.0 ± 0.1	3.98 ± 0.12
Mobile	Opera	300 ± 100	2.48 ± 0.11	3.06 ± 0.15	3.7 ± 0.6	20 ± 4
Mobile	Firefox	240 ± 160	5.9 ± 0.4	7 ± 1	8.6 ± 0.8	11.4 ± 1.5
Tablet	Safari	470 ± 80	6.84 ± 0.15	7.5 ± 0.2	8.2 ± 0.5	—

theoretically unlimited scalability in two dimensions. Since only two dimensions are displayed, scalability of the third dimension is also ensured.

Comparing PNG and JPEG showed minimal loading difference within the university network, whilst the compression rate of JPEG was five times better than that of PNG. For low-performance devices and networks, the difference in the loading time was very notable and JPEG was clearly favoured over PNG tiles. Yet, the lower quality of JPEG made PNG a better choice for medical applications given minimum device performance.

For the test, the zoom level was set to ensure that the maximum number of images had to be loaded. Speed was measured using the time taken from the point of user input to the point of image display. It was ensured that the images were not cached. These were the worst conditions possible, which was reflected in the results. For a test series where the zoom level was not restricted but all zoom levels were covered equally, the top 10 % TTDs were at 90 fps.

The tests were carried out in July 2015 within the University of Southampton network. The server was hosted with IIS on Windows 8 (16 GB RAM) and the times recorded on a Windows 7 PC (16 GB RAM) with Opera 28, Firefox 35 and Internet Explorer 11 via a gigabit network connection. Each request required 38 tiles to be loaded. Mobile speeds were recorded on a Samsung Galaxy Duos (Android 4, 645 MB RAM, Firefox & Opera, 8 tiles to load) and iPad 1 (iOS 5, Safari, 20 tiles to load) using Wi-Fi (IEEE 802.11ac) with a VPN. It should be noted that the image loading time fluctuated massively, especially on slower devices, with a standard deviation of up to 67 %. As this was not tested in an isolated environment, the workload on neither the server nor the client was constant.

5.1.4 Adapting Mata

In the previous two sections, a web-viewer for large 3D files has been presented. For testing purposes, it was implemented using the website presented in Section 3.1 but, as explained at the end of that section, the upload and download of large files over the internet is a major limitation in terms of time. At the end of Chapter 3, we suggested the use of a file watcher to overcome these issues and simplify data access for the end user and adapted the HDC for use with medical

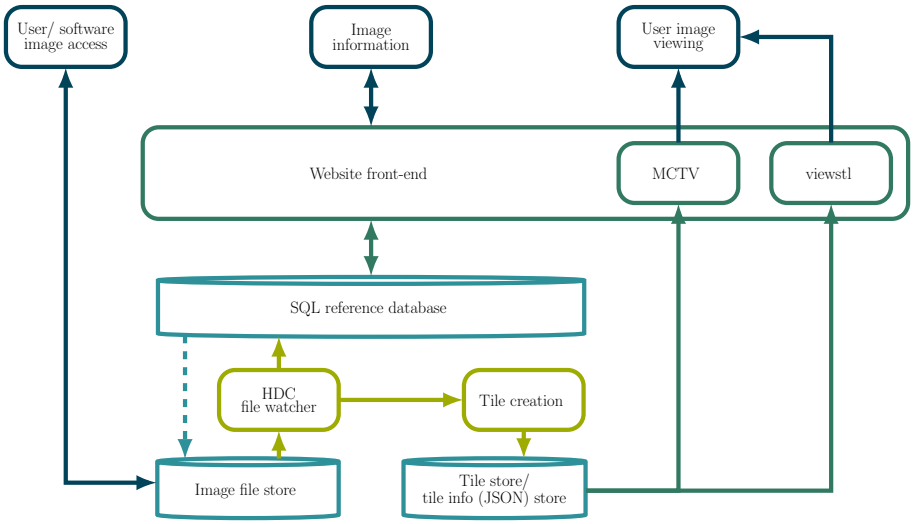


FIGURE 5.9: Architecture of Mata 2 with the addition of MCTV and viewstl for image viewing. This builds upon the Mata 2 architecture presented in Figure 3.5 on page 51. Matching colours indicate parts which fulfil a similar role within the architecture.

data. This section will discuss the implementation of the web-viewer with the HDC file watcher and the Mata website.

After the system was set up as described in Section 3.2.1, the next step was to link MCTV to the system. On the presentation layer, it can be directly embedded into Mata without additional installations. On the back end, a script had to be implemented to automatically create previews and metadata required by MCTV. The HDC file watcher already had a functionality to allow a script to be executed when a dataset changes. This was used by Mata to trigger a script creating a preview for a dataset each time it changes, as shown in Figure 5.9.

The process of creating tiles requires a large number of read and write operations and is therefore relatively slow. This means that it is likely that a dataset is being changed while a preview of the old version is still being created. For example, when a dataset comprising a stack of 2D image files is first uploaded, the file watcher triggers the plug-in for each file upload. When the first file is being uploaded, the file watcher starts the tile-creating script (the tiler). When the second file reaches the Mata file store, the tiler has not yet finished, but the file watcher starts another instance of it. For a 3D image with 2000 slices, this would mean that millions of duplicate tiles would be created.

Therefore, the script needed to cope with the processing of large images. The approach we took was to split the task of creating tiles into several subtasks which can then be cancelled remotely. To implement this, a local queue (as explained in Section 2.1.2.1) was set up to enable local deployment, as shown in Figure 5.10. Celery version 3.1.25 was used for the queue as it is the latest version compatible with Windows. It was served using Rabbit MQ 3.6.9. Scripts interacting with the queue were implemented in Python.

HDC executes a script that sends a job request to ‘readdir’ to the queue. The ‘readdir’ function first checks if the command has been executed for the same directory before and revokes any

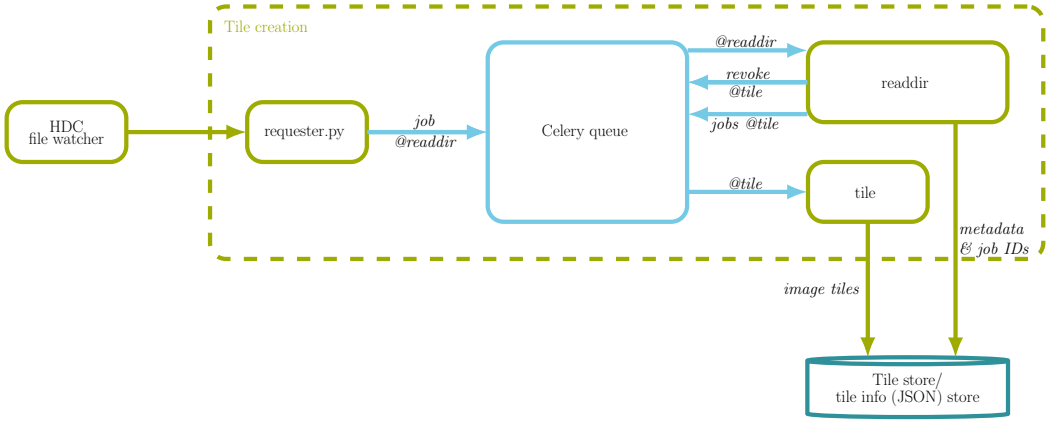


FIGURE 5.10: Schematic of the queue implemented for the tiler in Figure 5.9. Matching colours indicate parts which fulfil a similar role within the architecture.

outstanding jobs. As this version of Celery does not support accessing information about tasks in the queue, it is necessary to save the job identifications (IDs) in a separate file. The Python code then reads any metadata files it finds and attempts extracting remaining information such as the global minimum and maximum pixel values from the image files. For extracting the minimum and maximum pixel values, only a selection of the image files is read (or a random set of points taken from a raw file) and evaluated to reduce the overall time it takes to execute the script. Finally, one tiling-job request is created for each image slice, and the job ID is written to a file. The tiler then splits the image into smaller tiles and saves the tiles to disk as shown in the beginning of Section 5.1. The tiler jobs are started with a lower priority than that of the job reading the directory, to ensure that revoking jobs gets priority over working on other jobs. This reduces the overall queue length and avoids images being tiled more often than necessary.

It is important to make images accessible in a way meaningful to the medical researchers who are used to 2D images. With the script for creating the tiles in place and having it linked to the management system as shown in Figures 5.9 and 5.10, MCTV is able to read the tiles from the server file store and present them to the user as shown in Figure 5.11. Users' access to the tiles is restricted through the permissions on the network file share.

5.1.5 Adding a 3D Surface Image Viewer

The viewer presented in the previous section enables the viewing of volumetric images in the form of 3D image stacks and 2D images. It cannot deal with 3D viewing or with the viewing of surface files. The data which is inherently 3D also requires 3D visualisation in order to enable the analysis of 3D structures not represented in a 2D viewer. It was therefore decided to add viewstl[6] to Mata to display stereo-lithography (STL) and object (OBJ) files.

As with MCTV, viewstl does not require special plug-ins but works with JavaScript. Files are loaded into client-side memory and are not processed by any third party.

In order to use the encrypted version of the Hypertext Transfer Protocol (HTTP) for all resources, a local copy of Viewstl was created and modified to use the encrypted protocol.

[6]Available at http://www.viewstl.com/ (last accessed 03/08/2018)

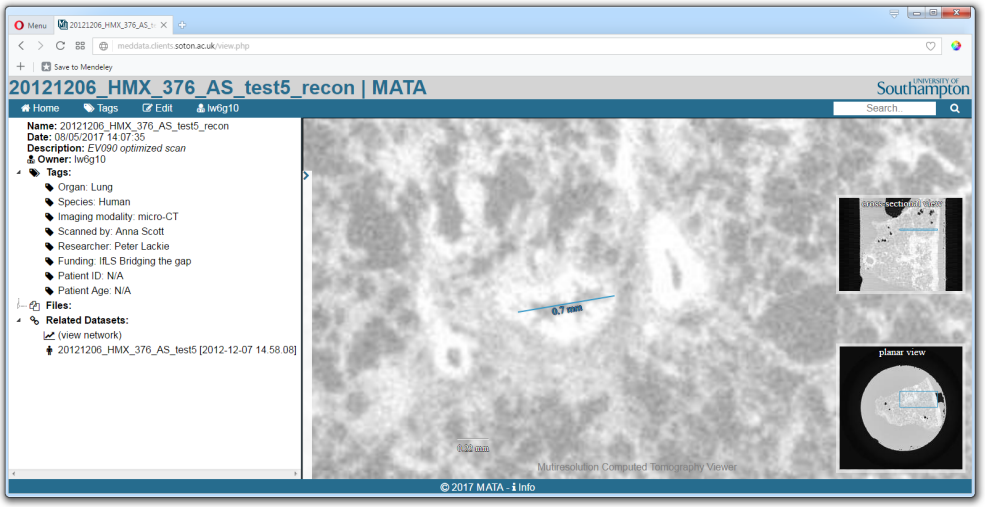


FIGURE 5.11: Screenshot of a dataset containing a TIF-stack viewed in Mata. The panel on the left shows the tags and related datasets. On the right a slice of a µCT scan of a lung biopsy can be seen with a planar and cross-sectional view as explained in Section 5.1.2.

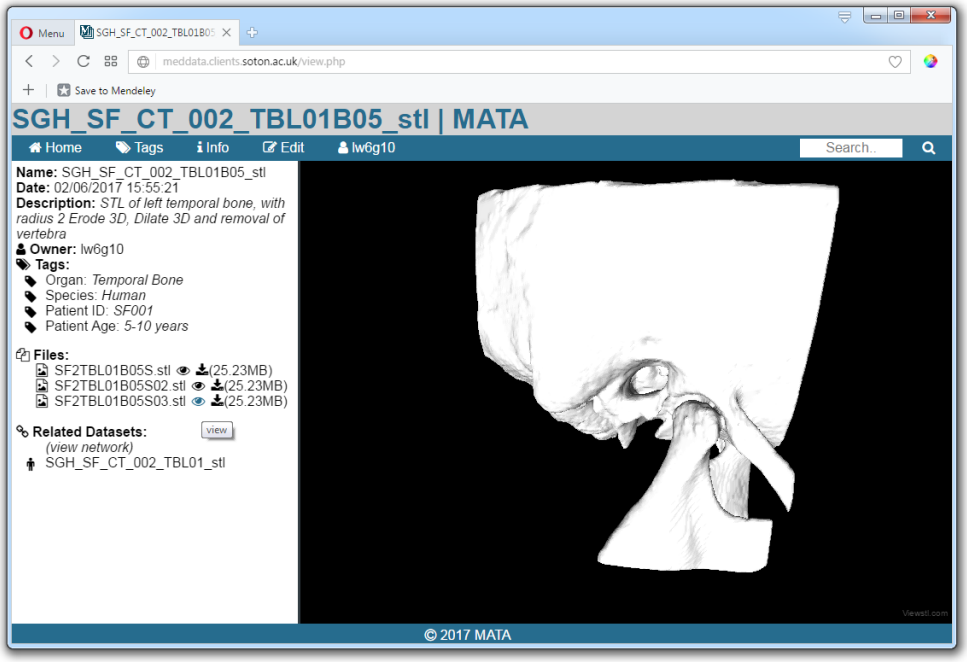


FIGURE 5.12: Screenshot of a dataset containing an STL file, viewed in Mata. The layout is the same as in Figure 5.11 but MCTV is replaced with viewstl showing the surface of a temporal bone. The generation of this file will be addressed in Section 5.3.2.

To integrate the viewer, an HDC file-watcher plug-in was created, which was executed for STL and OBJ files. It calls a Python script which uses the same queue as the MCTV tiler but only creates a file with the name of the first STL file in the datasets folder. On the web page for viewing a dataset, PHP is used to check if the dataset contains a 3D surface file and if the user has read permission. If this is the case, viewstl is loaded and is passed the file name, which is then displayed as shown in Figure 5.12.

Through the use of viewstl, the ability to view 3D surface files (specifically STL and OBJ files) over the web via Mata was demonstrated. The generation of 3D surface files will be addressed in Sections 5.2.3 and 5.3.2. The secure implementation of an external viewer also shows that, due to its modularity, Mata can be extended very easily.

5.1.6 Discussion

Advances in medical imaging devices has led to growing amounts of data requiring organisation and remote viewing. Viewing should not be dependent on browser-external solutions as this limits compatibility. Using multi-resolution images, these limitations can be avoided.

We created an image viewer and manipulator for 3D CT data and showed that it can be implemented into an existing system. We demonstrate the feasibility of using this method for medical applications with respect to several aspects:

(a) Use of tiled images improves the overall speed of the viewer compared to untiled approaches;

(b) Image formats with lossless compression require significantly more storage space than alternatives using lossy compression. For display on a computer, there is no significant difference, but on mobile devices they were 16 to 45 % slower;

(c) Basic requirements of medical image viewers such as pixel-based image processing can be implemented into a web-viewer;

(d) These features do not restrict the usability of the system for mobile or low-performance devices;

(e) Using JavaScript and HTML5, it is possible to implement these functionalities for all currently common browsers;

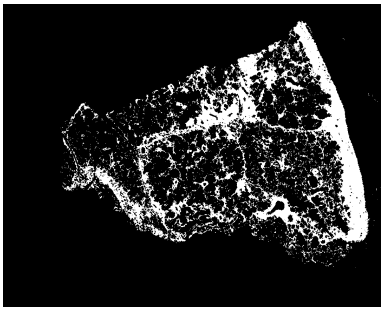
(f) MCTV can be added to existing image management systems.

Combining Mata and MCTV allows users to access experiments through a file store, as they are used to, as well as a website, where they can attach metadata to the datasets for easier searching as well as faster previewing of extra-large 3D images.

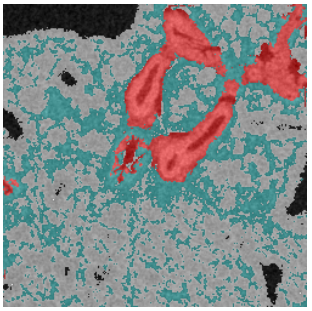
5.2 LungJ and Image Visualisation

In the previous section, the use of an established image visualisation technique was demonstrated for online use with any 3D voxel data. In this section, the option of generating more sophisticated visualisation options on a local machine will be discussed. In Section 2.2, the ImageJ plug-in LungJ[7] was introduced—a tool that enables application of segmentation workflows on very large images. ImageJ itself has an in-built feature to display multi-dimensional images as 2D slices. A plug-in for projected 3D viewing of volumes and surfaces is also available. This section will look at the implementation of image visualisation methods in LungJ. This includes 2D, projected 3D and 3D data representation, introduced in Section 2.3.

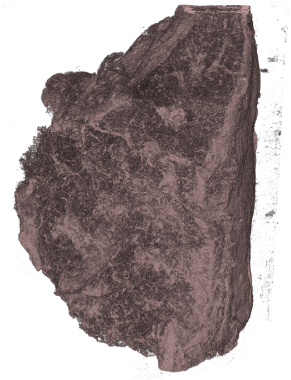
[7]Source code and documentation published as doi:10.5258/SOTON/401280 (Apache 2.0 License)



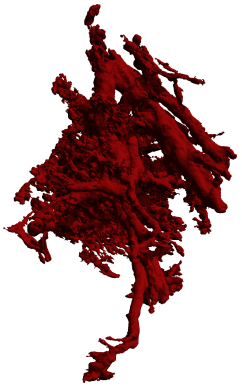
(a) Fibrotic area of cross-section.



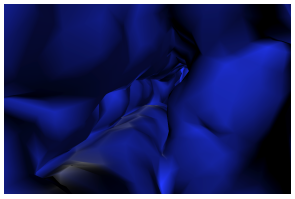
(b) Colourized overlay of masks on top of original image



(c) 3D view of the fibrotic structure

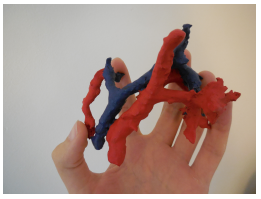


(d) 3D view of vessels



(e) 3D render of the inside of an airway. The inside wall is folded due to compression by the muscle around it.

(f) 3D render of the inside of a blood vessel.



(g) 3D print in acrylonitrile butadiene styrene (ABS) post-painted (airways are blue and blood vessels are red)

FIGURE 5.13: Image representations created after segmenting the whole tissue volume.

5.2.1 2D Representations

The default presentation method of multi-dimensional images in ImageJ is through 2D slices. These can be scrolled through in further dimensions. The BigDataViewer[8] plug-in also supports reslicing at an arbitrary angle (Pietzsch et al., 2015). Figure 5.13a shows what a segmentation mask created by LungJ looks like in ImageJ. Our target was to create a view more similar to stained microscopy views than default grayscale images of a µCT scan. A tool was created to take a set of masks from the same image and overlap them, giving each mask a different colour. The resulting view is seen in Figure 5.13b, wherein the vessels (red) and normal tissue (green) are shown clearly. Colouring of the images can be achieved on a block level before re-assembling the whole image by using the LungJ block approach. After reassembly, the process of viewing many coloured 2D sections of a µCT scan is similar to that of going through hundreds of 2D histology images (see Figure 1.2a on page 3).

[8]Official site: https://imagej.net/BigDataViewer (last accessed 22/01/2019)

5.2.2 Projected 3D Representations

The major advantage of the histopathology workflow proposed in Chapter 4 over established microscopy is the extraction of 3D data. This enabled us to create projected 3D views (2D representations of 3D models) of individual features. Fiji 3D Viewer (Schmid et al., 2010) allowed the display of interactive 3D surfaces and volumetric images, as shown in Figure 5.13c. More elaborate rendering of 3D surface views and videos was possible by using Avizo[9]. This could be outside views (Figure 5.13d) but also inside views of blood vessels or airways (Figures 5.13e and 5.13f, respectively) similar to virtual endoscopy (Robb, 2000). The inside view shown in Figure 5.13e allows clear identification of the airway; such a view can offer details about the vessel size while removing the complexity of the full network. Current research promises the ability to render large 3D images directly in Fiji without the use of proprietary software. This is done through the use of GPU acceleration provided by ClearVolume[10] (Arena et al., 2016).

5.2.3 3D Representations

Apart from projected 3D images, a physical 3D replica was also created. Conversion of the masks to 3D surface objects was possible using Fiji 3D Viewer, and this step was implemented as a single ImageJ function in LungJ. The use of Blender[11] allowed smoothing of the surface and creation of printable STL files, one of which was printed using a Tiertime UP! Plus 2 printer (Figure 5.13g). 3D-printed models improve the understanding of structures beyond the ability of 3D rendered graphics, which only show a 2D projection (Biglino et al., 2015). We will discuss the option of visualising images through 3D printing in more detail in Section 5.3.

5.2.4 Summary

In this section, we showed how it is possible to implement a variety of different visualisation methods in software. We looked at the approach of using different colours to represent structures identified by computer algorithms in 2D images, similar to highlighting areas of interest with dyes or agents, which users are more used to. We further looked at 3D representations in the form of projections and 3D printing. This allows users to see 3D structures which are not visible in true 2D representations. Our approaches were limited to extracts of images, however. In order to visualise large 3D images or create surface files for them, computers with more powerful processing units had to be used.

Having a variety of ways of representing the data improves the insight gained for the 3D structure inside an image.

The next section will focus on an example case for new 3D representation methods, mainly 3D printing but also AR, to improve the training of surgeons.

[9] Official site: https://www.fei.com/software/amira-avizo/ (last accessed 13/02/2018)
[10] Official site: http://imagej.net/ClearVolume (last accessed 22/01/2019)
[11] Official site: https://www.blender.org (last accessed 13/02/2018)

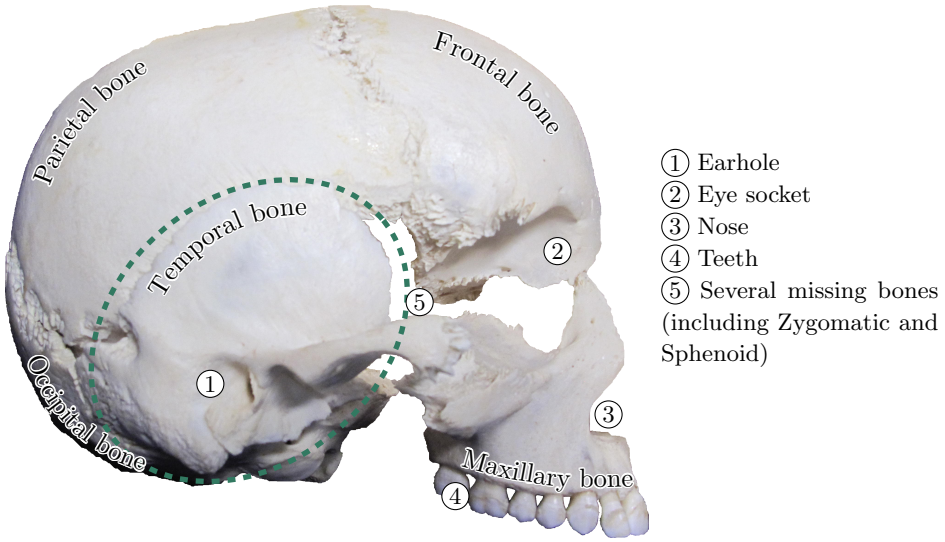


FIGURE 5.14: Parts of a paediatric skull with the temporal bone (Netter, 2014, p. 14).

5.3 Paediatric Temporal Bones—A Case Study

So far, we have shown that we can integrate basic 2D and 3D image viewing capability into the website of Mata from Chapter 3. We have also shown that more advanced views and even 3D prints can be created by extending the ImageJ plug-in LungJ developed in Chapter 4. In this section, we will take a closer look at 3D printing from medical scan image data using the example of paediatric temporal bones. The temporal bone is a large part of the skull by the ear, as shown in Figure 5.14.

5.3.1 Introduction

Temporal bone dissection is an essential component of otological training. The ideal temporal bone training platform should allow the surgeon to prepare for and undertake the steps of the actual operative procedure required prior to the operation on the patient. Several solutions exist for adult patients, ranging from the use of cadaveric bones to surgical simulations in virtual reality with haptic feedback (Wiet et al., 2002; Zirkle et al., 2007). Haptic devices and handcrafted plastic models struggle to reproduce the feeling of a real operation with respect to anatomy that is accurate but leaves room for variability as noted among different people (George and De, 2010). Therefore, cadaveric bone dissection remains the standard training method for trainee otologists (Duckworth et al., 2008).

For infants and children, this proves to be more difficult. Cadaveric paediatric temporal bones are generally not used due to ethical issues and availability. This leaves very little room for practice for a real operation, especially since the available plastic models of bones usually represent healthy bones. Cadaveric skulls are scarce compared to the number of trainees, owing to which George and De (2010) encouraged the use of alternative techniques, especially for junior trainees.

The solution proposed in this section is the use of image segmentation skills described in Chapter 4 to segment CT scans of children and 3D print the result so surgeons can practise

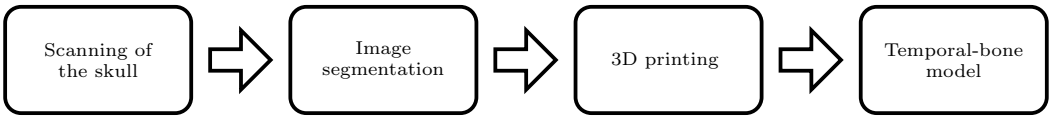


FIGURE 5.15: 3D printing workflow: creating a 3D-printed temporal bone from a CT scan.

drilling a 3D representation of a child's temporal bone. Using 3D printing for surgeries has been trialled before but usually relied on expensive multi-material printing. It has therefore been restricted to only a few prints (Chenebaux et al., 2014; Hochman et al., 2014; Longfield et al., 2015; Rose et al., 2015a). It was found to be useful in demonstrating spacial relations (Biglino et al., 2015). One of the challenges has been the image segmentation, since the studies relied on the use of µCT, which could only be applied to cadavers.

Research has been carried out at the University of North Carolina in the USA to test the feasibility of 3D-printed bone models for both training purposes and surgery preparation. However, only a single model was tested for each case, and details of the material used are not available. In both cases, highly expensive multi-material printers were used, which do not lead to a cost-efficient method (Rose et al., 2015a,b).

In France, Chenebaux et al. (2014) asked 25 otology experts to evaluate a 3D-printed temporal bone, who concluded that 3D printed samples are useful in replacing cadaveric samples for training purposes.

Researchers at the University of Manitoba in Canada examined the mechanical properties of different printed materials using objective mechanical analysis (yield, rupture, indentation forces and vibrational properties) and compared them to a sheep femur leg bone with a Young's modulus of (2.79 ± 0.14) GPa (Hochman et al., 2014). All polymers tested were used for powder printing. The most suitable ones were polyetheretherketone (PEEK) with cyanoacrylate with hydroquinone (CAH) and untreated PEEK. Untreated PEEK had the most similar Young's modulus to that of the sheep bone, with a value of (2.79 ± 0.06) GPa (Hochman et al., 2014). In similar experiments, the much cheaper ABS showed a Young's modulus of 2 GPa to 3 GPa (Cantrell et al., 2016; Grammatikopoulos et al., 2018). As 3D printing involves layering, the resulting objects have different properties in different directions, unlike natural materials. The comparisons, therefore, have to be interpreted carefully.

5.3.2 Methods

Using non-identifiable paediatric HRCT scans of temporal bones and machine learning algorithms, 3D printing of paediatric temporal bones has been trialled (Figure 5.15). CT scans from patients were used with the permission of their parents and anonymised by the hospital. Using the TWS, a classifier model was created to segment bone. As bone is much denser than the soft tissue surrounding it, segmentation was easier than for vessels, as described in Section 4.2.3.

After creating a classifier, it was then applied to the whole image following the LungJ workflow and segmenting the full scan. As the first step, the DICOM image stack was assembled into

(a) 3D-printed temporal bone with support material

(b) Drilling of 3D-printed temporal bone at the University Hospital Southampton.

FIGURE 5.16: Creation and evaluation of a 3D-printed temporal bone.

a 3D image. After classification, the voxels of the mask was made isotropic from the values obtained from the DICOM file to ensure correct interpretation. Results were post-processed in two different ways—using a very simple dilate and erode algorithm or using the LungJ filter to fill holes in the segmented regions. The filling of the holes was expected to counteract segmentation errors due to the porosity of the bones as 3D printing already introduces porous filling. We programmed a LungJ function for creating 3D-surface files based on Fiji 3D Viewer, following the process established in Section 5.2. This way, an STL file was created from the mask. Using Blender and integrated tools, the surface was checked for printing issues, and the visible outer surface was smoothed to mask the low resolution of the scan. The original segmentation result was printed without smoothing for reference purposes. Each file was printed on a Tiertime UP BOX printer with a high-density infill pattern and 0.2 mm layer height in ABS. Using additive manufacturing for complex shapes required a lot of support material to be printed, as seen in Figure 5.16a, which had to be removed subsequently.

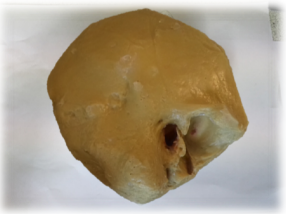
The 3D prints were then given to ear, nose and throat (ENT) specialists at the hospital, who mounted them on modelling clay. The prints were drilled using an electric drill and after cortical mastoidectomy internal structures were evaluated as shown in Figure 5.16b.

5.3.3 Results

3D printing of paediatric temporal bones has proved both economical and anatomically accurate for training purposes and operation training. HRCT scans allow cases of appropriate complexity to be selected for prints. 'The 3D-printed temporal bones are useful because they allow us to provide anatomical models allowing students to practise operations on' (Hasnaa Ismail-Koch, personal communication, 2018). 3D-printed models enabled cortical mastoidectomy to be undertaken but lacked sufficient detail of the inner ear and facial nerve. With a price of £16 for a 40 g print in May 2016, the cost of 3D-printed bones is much lower than that of traditional plastic cast models (see Table 5.3).

The evaluation by ENT specialists also revealed limitations of the process. Due to the limited scan resolution, parts of the facial nerve could not be segmented as they were less than 1 pixel in size. It would require an alternative segmentation method, such as a deformable model (compare Section 2.2.6.2), in order to estimate the position of the facial nerve. Furthermore, the printing

TABLE 5.3: Comparison of different resources for obtaining temporal bones.

3D-printed model	Plastic casted model	Cadaveric bone
		
Cost of £16 (excluding cost of labour and profit margin)	Cost of £65 (Pettigrew Temporal Bones, 2018)	Cost of £225 to 375 (Longfield et al., 2015)
High resemblance of the anatomy	Generic anatomy	Accurate anatomy

process incurs the use of a lot of support material (see Figure 5.16a), which is time-consuming to remove and can lead to the accidental removal of important parts, such as the ossicles (the small bones inside the middle ear). Alternative 3D-printing methods may reduce the amount of support required or use support material that is easier to remove. An example of this uses the 3D printer Ultimaker 3[12], which can print water-soluble support material. Other printing types, presented in Section 2.3.4, can help improve print results even further by giving more control over the support material used. Scalability of the creation of training sets can be achieved through the use of 3D-printed moulds. A third limitation is the strength of the material used. This is due to the printing occurring in layers, causing different sample strength in different directions. It is further caused by the non-solid infill, which again is one-directional and leads to varying strength in different directions. Alternative printing methods (see Section 2.3.4), such as liquid based 3D printing, could help improve this behaviour. In addition, it may be useful to choose a different plastic instead of ABS for better haptic sensation during the drilling. This would also make the 3D-printed temporal bones a useful option for surgeons to practise on an anatomically accurate model before an operation.

5.3.4 Possible Extension to Augmented Reality

An alternative to 3D printing is AR, which has a shorter waiting time from the creation of the scan to the final visualisation. In Section 2.3.3.2, we mentioned existing implementations of AR in medicine. While AR does not provide any haptic feedback, the display of CT scans in 3D can equally help paediatric surgeons to plan surgeries or be used as a teaching tool as 3D-printed models.

The possibility of using an AR approach to visualise CT scans has been tested, and the preliminary result is shown in Figure 5.17. The same image files created for the 3D printer were used to achieve this. After importing the surface representation file into Unity[13] engine, it was possible to port it to a HoloLens headset. The bone was displayed in front of the user and could be modified using hand gestures.

[12]Official Site: https://ultimaker.com/en/products/ultimaker-3 (last accessed 20/07/2018)
[13]Official site: https://unity3d.com/ (last accessed 31/05/2018)

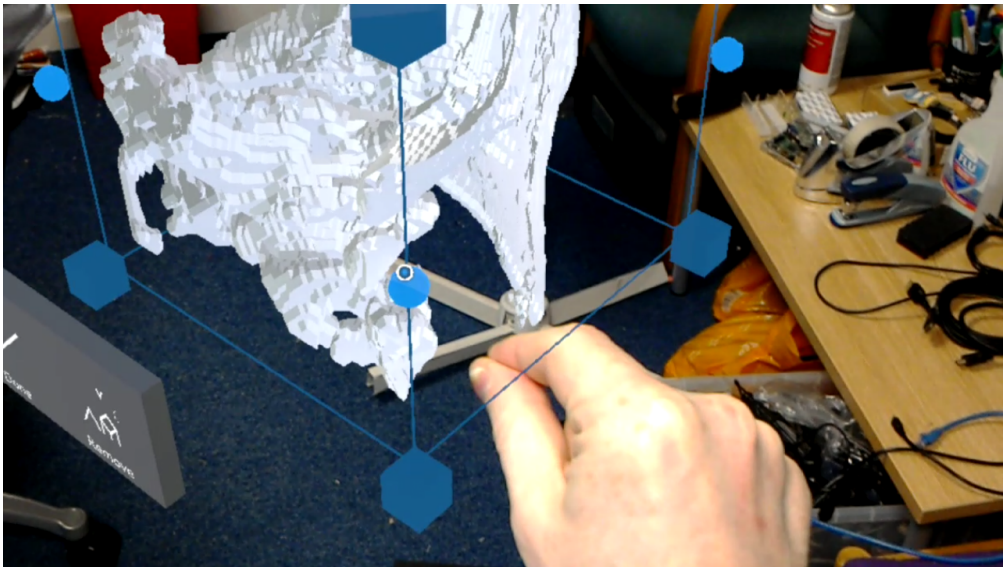


FIGURE 5.17: Screenshot of a temporal bone in AR viewed through HoloLens.

More sophisticated implementations of AR in medicine can help provide medical surgeons powerful tools to plan their operations or teach the next generation of medical students.

5.3.5 Discussion

3D-printed temporal bones from HRCT images provide optimal opportunities for paediatric temporal bone training and allow for more accurate delineation of structures. The 3D printing turns out to be cost efficient. For mass production of training models, the cost can be further reduced and speed increased through the use of 3D-printed moulds. It would also provide preoperative simulation opportunities in particular cases involving challenging anatomy. Selection of appropriate material also potentially offers improved haptics. Further research in alternative segmentation and additive manufacturing methods can lead to improved temporal bone models. Even though the scans used in the work described in this section were much smaller than the µCT and high-resolution 3D images this thesis targets, it illustrates what can be done in the near future. It also shows that the concepts do not limit the workflow to new large images but are 'backwards compatible' to commonly used radiography scans.

5.4 Summary

In this chapter, different visualisation techniques were demonstrated using three example cases: the viewing of 3D images on a website, the examination of image results from LungJ segmentation and the 3D printing of paediatric temporal bones. These covered the following segments of the image visualisation spectrum introduced in Section 2.3: 2D-section views, 3D-rendering views, augmented reality, and 3D printing.

An orthogonal 2D section view has been implemented as a web-viewer. It can be used for extra-large 3D images by splitting the image into smaller blocks or multi-resolution tiles. A web-viewer

for 3D CT images, MCTV, was developed and integrated into Mata, mentioned in Chapter 3. It only uses cascading style sheet (CSS), HTML and JavaScript to display large 3D images. It allows the display range to be edited and distances to be measured on the image. For LungJ, several visualisation options were created. These follow the LungJ principle of splitting the image into smaller blocks, processing them and reassembling. In 2D, image slices were shown and several segmentation results were overlaid using different colours to highlight different areas.

3D rendering for web applications was implemented for smaller images but it requires more computational power than preparing 2D section views and at the same time makes rational loading of only the necessary data more difficult. One can hope that future work finds a way to save multi-resolution 3D-tiles and only load the ones in the field of view, similar to multi-resolution polygon rendering technology used by game engines (Gobbetti and Marton, 2005; Rossignac and Borrel, 1993). For LungJ, a function was created to convert 3D voxel images into surface STL files. It was based on existing ImageJ plug-ins and the size of images it can be applied to was limited by these plug-ins. Using more powerful computers, it allows for projected 3D representations of large image files.

The surface files created by LungJ were also used for 3D printing of parts of images. The exploration of 3D printing was continued as part of the paediatric surgery case study. CT scans of temporal bones were taken, segmented and 3D printed. A feasible model was produced for surgeon training.

Finally, the use of AR illustrated what is possible in the near future based on the use of CT scans of temporal bones to create a 3D model of a patient's bone.

Altogether, both existing and upcoming ways of viewing image data have been covered in this chapter. It was possible to enhance existing visualisation techniques to support large images. Newer, more advanced representation methods have been implemented but require further improvement to be able to cope with large image data. The next chapter will discuss bringing all of the workflow steps from the last three chapters together and placing them in the bigger picture of medical research.

Chapter 6

Complete Workflow

As discussed in Chapter 1, a digital workflow for medical images is needed to be able to benefit from the digitalisation of medicine and new trends such as machine learning. It is also required to deal with the increasing image size. We identified three main areas of the medical workflow as image storage, processing, and visualisation and analysed each of these areas in the previous three chapters. Each chapter then proposed a solution to handling modern medical images in the form of one or more approaches to the corresponding area.

These different areas are linked through the medical image life-cycle, and in this chapter, we will show how the different approaches work together. We will briefly review the previous chapters in Section 6.1 and then discuss how the individual approaches fulfil the requirements of a medical image workflow in Section 6.2. In Section 6.3, we will discuss limitations and how they can be overcome. Finally, we will look at how this workflow integrates into the existing medical workflow using example use cases in Section 6.4. A summary of this chapter is provided in Section 6.5.

6.1 Review of Developed Approaches

In Chapter 3, a description has been provided for building a data repository that offers storage of data on a shared network drive. The file-store solution offers users a familiar way to access and manage their data. At the same time, access through a website enables the addition of metadata and extended functionality compared to a standard file store. The HDC file-watcher (see Section 2.5.3) was used to link file store and database. The website enables users to add metadata to datasets, offers options to find relevant data, and provides integrated security. The modularity of this system allows parts of it to be exchanged for alternative solutions. Table 6.1 shows how this compares to existing systems. With this system in place, images created by medical researchers can be organised and shared with other researchers.

In Chapter 4, we described the development of an image processing tool to segment large images. LungJ[1] is integrated into the image processing software ImageJ. Its key focus is on supporting large images by splitting them into smaller, processable blocks. As discussed in Section 4.3, we considered both usability and workflow modularity by choosing a popular open source software and splitting the workflow into several steps which can be individually improved or replaced.

[1]Source code and documentation published as doi:10.5258/SOTON/401280 (Apache 2.0 License)

TABLE 6.1: Comparison of Mata described in Section 3.2 and extended in Section 5.1.4 to other image management systems.

	OMERO	**BisQue**	**HDC**	**Mata 2 (including image viewers)**
Operating system	Linux (various)	Linux (various)	Windows Server	Windows Server
Server	OMERO.server	BisQue server (various)	Microsoft IIS	Microsoft IIS
Database	PostgreSQL	XML	SQL Server	SQL Server
Middleware	Java	Python	Sharepoint 2010/ .NET 3.5	PHP 7.1
Data access	Website/ Client software/ API	Website/ API	Website/ Network file share	Website/ Network file share
Synchronisation	OMERO.dropbox	✗ (not applicable)	File watcher	File watcher (from HDC)
SSO authentication	✗	✗	✓	✓ (partially from HDC)
Metadata editing	✓	✓	✓	✓
Metadata import	✓ (individual only)	✓ (via file export and import)	✓	✓
Metadata suggestion	✗	✓ (1-to-1 correspondence)	✗	✓ (ranked dictionary)
Basic search	✓	✓	✓	✓
Advanced search	✓	✓	✗	✗
Tag cloud	✗	✗	✗	✓
List of direct relatives	✗	✗	✓	✓
Relation network	✗	✗	✓ (list of direct relatives)	✓ (graph of all relatives)
CT slice viewer	✓	✓	✗ (but 2D viewer)	✓ (MCTV[a])
3D voxel viewer	✗	✓	✗	✗
3D surface viewer	✗	✗	✗	✓ (Viewstl)

[a]Source code and documentation published as doi:10.5258/SOTON/400332 (Apache 2.0 License)

Since LungJ splits an image into smaller blocks, it enables parallel computing for image processing, which has been listed as a requirement by Montagnat et al. (2004) even for processing algorithms originally designed for single thread processing. Once the medical research project (see Section 1.1) scales to more images, it will be possible to scale an implemented processing workflow without changing the tools used. Even without this additional speed-up, the current algorithm reduced the required time from months of manual segmentation to a day (see Section 4.3).

Finally, we looked at image visualisation in Chapter 5. Image visualisation is vital in enabling the user to interpret the image data. Different visualisation types can cause different interpretations. There is also the specific use case of using image pre-viewing to enable researchers to decide on data of relevance from the storage system referred to in Chapter 3. In Sections 5.1.2 and 5.1.3, we discussed the development of the web-viewer MCTV[2] which allows displaying large 3D volume images. Displaying large images has been made possible through the use of multi-resolution tiles, which also contributed to a fast response by reducing the amount of data transfer through the internet. We also integrated it together with a surface image web-viewer into Mata[3], as mentioned in Sections 5.1.4 and 5.1.5. In Section 5.2, 2D and 3D visualisation options were implemented for ImageJ using LungJ. We enabled researchers to experiment with different 2D and 3D visualisation methods to explore their datasets. Finally, we looked at 3D printing as a visualisation tool. The advantage of physical objects over 2D projections was shown through the use case of temporal bone printing in Section 5.3.

The next section will show to what extent these approaches fulfil the various requirements of a medical image workflow.

6.2 Workflow for Research Images

In this thesis, we identified the need for approaches that can deal with images of any size and show better usability than existing solutions. We addressed the three main aspects of the image life-cycle, as discussed in Sections 1.2 and 6.1.

In this section, the different features of the overall workflow are evaluated. In Section 1.4, key requirements for a medical image management system were identified: support for large-sized images, usability, system modularity, metadata support, data traceability, fast response and visualisation, and data security. The following sections will explain how each of these requirements is addressed by our approaches.

6.2.1 Support for Large Images

Mata uses a file store to store data. The file system is very scalable and can support very large images without problems (Scott, 2014). For the Windows Server 2012 R2 used for this system, the maximum file size is 16 TB (Microsoft Corporation, 2018, Appendix A). With increase in the file sizes in future, the file systems of operating systems will continue to grow to support

[2]Source code and documentation published as doi:10.5258/SOTON/400332 (Apache 2.0 License)
[3]Source code and documentation published as doi:10.5258/SOTON/D0430 (Apache 2.0 License)

these large images. At the same time, the database only holds one entry per dataset, making the number of datasets more important than the size of an individual dataset. The use of a shared network drive for image storage further allows data transmission to be handled by the user's operating system, which connects to the server. This well-established data transmission ensures reliable file transfer without the need for additional software.

LungJ splits the large images into smaller blocks as explained in Section 4.2.2. This way, images of theoretically any size can be cut down into manageable chunks, which are then processed using existing algorithms. This also applies to the visualisation part of LungJ.

MCTV works similarly to LungJ. Images are split into smaller parts, but, in this case, the splitting occurs over multiple resolutions, as explained in Section 5.1.2. The use of these multi-resolution tiles reduces the amount of data required to be transferred to display the image on the end user's device.

6.2.2 Usability

Usability of the system has been addressed from two different perspectives—one involves metadata access through the website, and the other concerns data access through the file store. The website uses tables and tag clouds to display metadata in ways commonly used across various websites. In contrast to existing solutions, the shared network drive means that there is no need for an external software to ensure reliable communication with the data store. Most users will know how to operate a file store. Any common image processing software can interact with a file store, which means that users are not restricted in their choice of software for processing their data. All of these benefits make Mata a very user-friendly system.

ImageJ is the most popular image processing software in medicine, as mentioned in Section 4.2.1. By creating a plug-in that can be implemented using the standard ImageJ plug-in manager and relying on existing processing algorithms rather than forcing users to use custom algorithms, we ensured the usability of LungJ.

Similarly, we define the usability of MCTV through the low number of requirements in order to be able to use it. As discussed in Section 5.1, it runs natively in a browser, only relying on JavaScript and CSS. It is therefore compatible with any major internet browser and does not require installing an additional plug-in.

6.2.3 System Modularity

The set-up of Mata is modular. By using a standard website, many tools developed by third parties can be integrated easily to support various types of data, as illustrated in Figure 6.1. The image-processing and image-management workflow has been split into its components, allowing the processing to be managed separately from the storage. Instead of using ImageJ in combination with LungJ, any other processing software can be used to read files from a file store and save the result back into the file store again. In Section 5.1.5, we showed how users could decide on the presentation solution they find suitable for their application by implementing the 3D viewer 'viewstl'.

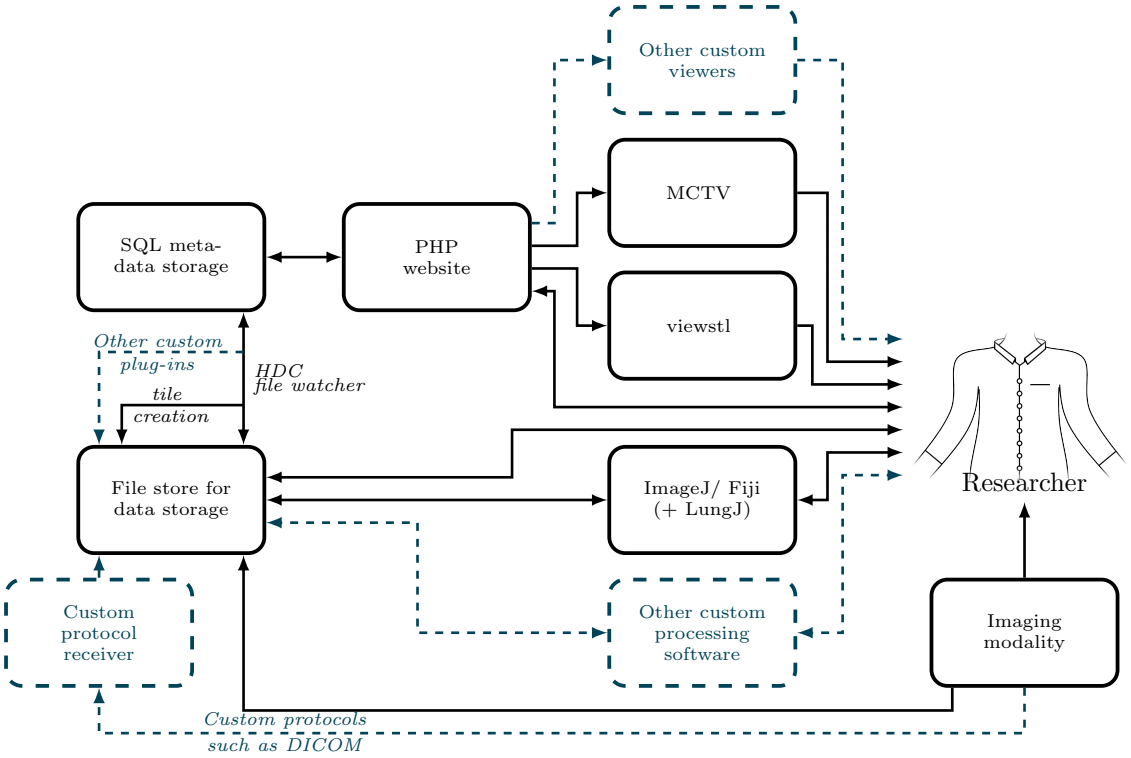


FIGURE 6.1: New proposed workflow in detail. Dashed parts show possible extensions by users. The storage system needs to be administered by a data librarian in the field. Other users can access shared datasets based on the file-store permissions mentioned in Section 2.5.3.

6.2.4 Metadata Support

HDC connects a file store with a database so that advanced metadata for files on the file system can be stored in the database. It also tracks data movement and modification to prevent metadata from being lost. Furthermore, features for modifying metadata and searching through it have been shown. Metadata can be grouped to create categories and can be displayed as part of a relationship network. In Section 3.2.3, we showed how users could use tag clouds and a simple search function to find relevant datasets.

6.2.5 Data Traceability

Montagnat et al. (2004) defined traceability as the ability to find the origin of an image. Features of defining origins for each dataset in combination with the network graph allow traceability over as many levels as required to find the original data to any image. We even enable users to find the origin where multiple original images are present and have been combined. Through the systematic use of metadata, it is possible to ensure provenance, tracking not only the origin, but keeping a log of all the steps leading to the current version of an image. This has to be done manually. Trusted provenance is not achieved with this system.

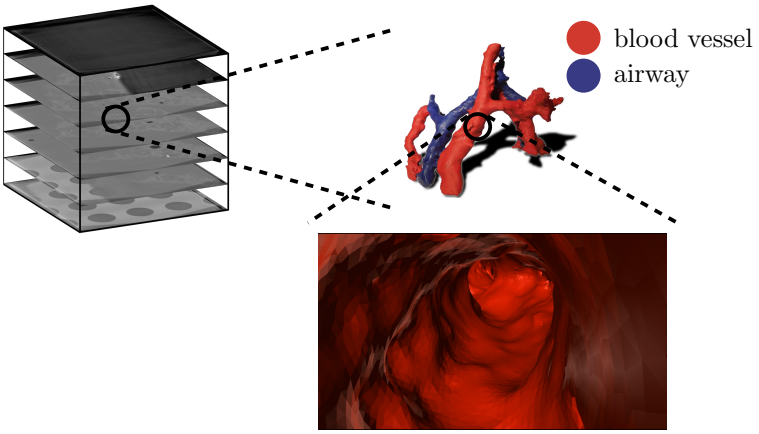


FIGURE 6.2: LungJ summary: Stacks of about 2000 image slices (top left) can be segmented to produce 3D visualizations of included blood vessel and airways (top right) and provide information about the inside of veins (bottom right). This can help to detect irregularities and identify diseases.

6.2.6 Fast Response & Visualisation

The website front-end of Mata is a lightweight PHP application and implements two viewers to visualise 3D image stacks and surface images. MCTV has been optimised for fast response time, as we showed in Section 5.1.3. We also discussed the implementation of further visualisation methods in Sections 5.2 and 5.3 to give medical researchers a better insight into 3D data. Our LungJ algorithm extracts 3D vessel structures from large image stacks, allowing insight into airways and veins (see Figure 6.2). By converting segmentation masks of medical scans into surface files, we were able to visualise them in AR and 3D print them.

6.2.7 Data Security

As discussed in Section 3.2.4, the image storage system uses well-established technologies such as SQL Server and Windows Server to benefit from the experience of large user communities, industry standards and established ways of securing the system.

6.3 Limitations and Extendability

In the previous section, we showed that the approaches mentioned in the previous chapters do fulfil the requirements for managing images throughout their life-cycle. We have not yet addressed some limitations to our approaches that are due to hardware or software limitations: we will discuss them in the following sections.

6.3.1 Hardware Limitations

In this thesis, we are focussing on software, and some hardware limitations cannot be overcome by software. While we have managed to find solutions that can run on 'low-performance computers'

(see the beginning of Chapter 1), we have not solved the issue of storage space requirements and network speed.

Large images require a large amount of storage and the only way to improve the situation from a software perspective is the use of compression algorithms. Many compression algorithms have been developed and implemented in file formats, such as JPEG and PNG. In Section 5.1.2, we showed how significant the saving of lossless or lossy compression is. We also established that lossy compression is not acceptable for raw scientific data (Blume and Muka, 1995; Robb, 2000). We know that compression algorithms vary between file formats and some formats are not designed to work with compression. While it may be desirable to compress all data on the storage in a lossless way, this would limit the capability of external software accessing the data. This is especially true in the case of a custom compression algorithm. Instead, we suggest that lossless compression algorithms are applied where available for the used file format. Apart from that, the storage problem should be considered a hardware problem. The cost of hard drives is getting cheaper, making it feasible to increase the provided storage proportional to the increase in demand. Services such as cloud storage can help satisfy these demands in the future.

Low network speeds are a critical limitation when it comes to file transfer. Optical fibres (Maurer and Schultz, 1972) have increased the speed of network traffic compared to copper wires, but they are not widely implemented. According to a report by the Organisation for Economic Co-operation and Development (OECD), even though the fibre optic technology promises download speeds of 134 Mbit/s, the average download speed measured for the OECD countries was only 13.6 Mbit/s in September 2014 (OECD, 2015, Chapter 2). This means that the download of a 10 GB image on average will take 1.6 hours. Download speeds at the University of Southampton were measured to be at 350 Mbit/s to 540 Mbit/s. A 90 GB µCT image would still take at least 22 minutes to download at optimal conditions. When considering maximum transfer speed, the full network route has to be considered, and the final speed is set by the slowest link. We implemented fast pre-viewing techniques, which allow users to preview data before downloading gigabytes of data from the internet. Keeping servers local to the research institute can improve the performance, but limits the data exchange between different institutions. Having a distributed server network would decrease the load experienced by each server and improve the performance for highly busy applications. Even though splitting over several servers was considered as an option, it would still require a substantial amount of work. It depends on the end user to decide which implementation fits their needs best.

6.3.2 Software Limitations

There are several software limitations that the approaches presented in this thesis do not overcome. This includes the lack of direct access to the metadata, limitations in the application of the processing plug-in, lack of a fast 3D surface visualisation and slow code execution.

While Mata allows accessing the data directly through a shared file store, metadata remain stored in a database only accessible through the web interface. One can argue that the main use of the metadata is to search through the available data and select the right one, but even this assumption leads to limitations. Data mining is becoming increasingly popular. A different research project may want to use data mining to find similarity in data or identify data useful

to them, not just from the actual images, but also from the attached metadata. Easy access to the metadata would allow researchers to implement their own data mining algorithms. One can think of two ways to implement this. The first would be to give users restricted access to 'database views' containing the metadata for datasets or store a copy of the metadata as a NoSQL database in the corresponding directory of the file store. This would have to be kept up to date by the file watcher. An alternative would be the use of an API similar to the one used by BisQue or OMERO. Researchers can write a code to interface with the API and query the metadata. The use of an API would also help establish a semi-automatic way of data provenance similar to OMERO and BisQue, as described in Sections 2.5.1 and 2.5.2.

In Section 2.2.6, we listed various image-processing algorithms. The idea of splitting images into manageable chunks and individually processing those chunks applies to processing algorithms that do not rely on the value of a neighbouring pixel or global image information. By storing a few global variables and implementing halos, it was possible to apply the approach to any spatial filter and machine learning algorithms relying on these filters. In Section 4.4, we also discussed the possibility of applying the same approach to segment images using iterative algorithms. As explained in Section 4.4, this approach cannot be applied to the following two categories of processing algorithms: frequency filters and model-based image segmentation methods. Frequency image filters require information obtained from the whole image because they need to undertake a transformation of the image into the frequency domain. Model-based image segmentation methods also require the detection of markers over the whole image. While both methods can be implemented using the splitting approach, frequency domain filters would always suffer in accuracy. They require a more refined approach. Model-based algorithms would require a substantial adaptation in their internal programming to evaluate the fit of the current model for each image chunk.

In Section 5.1, we introduced a web-viewer targeted at showing 2D slices of a 3D image. Even though Sections 5.2 and 5.3 demonstrate the possibility of 3D visualisation, they require the full image to be loaded into RAM. For the same reason, the 3D viewer presented in Section 5.1.5 is not suitable for large 3D surface files. While there are solutions available, most of them seem to rely on a powerful server that renders a 2D image which is then sent to the client (e.g. Kvilekval et al., 2010). Some achieve this by storing the data in a database, which can be of particular advantage where not a lot of new data is generated on the go. We did not come across a freely available viewer that allows the creation of previews and renders 3D volume or surface images efficiently. Significant research has been carried out in this area, and advances in the gaming industry suggest that the rendering of large 3D images over the web is a problem that can soon be solved (Evans et al., 2014).

Finally, some of the code used in this work is not very fast. Part of the reason for the code being slow is that it is intended for running on devices with limited processing power. For an early-stage research project, it was not considered reasonable to invest in a lot of hardware. Even in the longer run, high-performance hardware would be shared between members of the medical research group and, therefore, the ability to run code on the researcher's own low-performance computer would be advantageous. This means that, for example, LungJ enables the processing of images that previously could not be processed on computers with low performance. At the same time, processing an image in chunks in series on a high-performance computer is slower than directly processing the whole image. This is not seen as a limitation since the code can

always be omitted and purely acts as an enabler. LungJ also enables parallel processing on high-performance computers. The benefit of the code lies in the fact that it allows a direct trade of money invested into IT equipment for time taken for image processing. Other parts of the code are not fully optimised. The Python code creating the image tiles for MCTV is particularly slow and, for a large-scale implementation, one should consider speeding up the code using tools such as Cython or by translating it into an entirely different programming language. Additionally, it is advised to make use of the queue and share the work of creating image tiles over several servers. Again, cloud providers can offer the required flexibility for this kind of service. At the current stage of the project, a speed up was not required as the application is read-intensive and only a few images were available.

6.3.3 Summary of the Limitations

In this section, we reviewed the limitations of the work done in this thesis. We did not aim to solve the hardware limitations of low network speeds and storage space requirements when developing our image storage system. We explained why hardware problems are inevitable and cannot be solved by research alone. We further addressed the limitations of the developed software and how the software can be extended to resolve the limitations. This includes the development of an API to improve metadata access for the image storage system and allow for data mining. A further limitation is the execution of image processing filters in the frequency domain, which is not accurate when using the splitting approach developed in Chapter 4. The viewing of 3D data over the web is limited to 2D sections or small 3D files in our implementation, and we rely on research in other communities to deliver reliable solutions. In general, much of the code has not been optimised for speed but for low memory consumption. We did present a few options, such as the use of dynamic scaling, to improve the speed of the code executed on the server.

Now that the full implementation has been shown in Section 6.2 and its limitations have been discussed in Section 6.3, the next section will give examples of how the approaches can be used in a medical research scenario.

6.4 Workflow Use Cases

Though they are applicable to any data, the approaches developed in this thesis were tested using a scenario modelled after a research project on lung diseases at the University of Southampton. This project evaluates the use of µCT scanners for the analysis of biopsies in a non-destructive way.

This section presents the use cases for this scenario to show how the tools discussed can integrate with an existing image workflow. The use cases are illustrated in Figure 6.3.

As part of this scenario, a large number of images needs to be created using a newly developed µCT scanner. Matching microscopy images are retrieved, and images are processed to enhance the contrast. µCT and microscopy images are then aligned in 2D and 3D using point-based registration based on the scan conditions. Matching image areas are presented to experts to assess

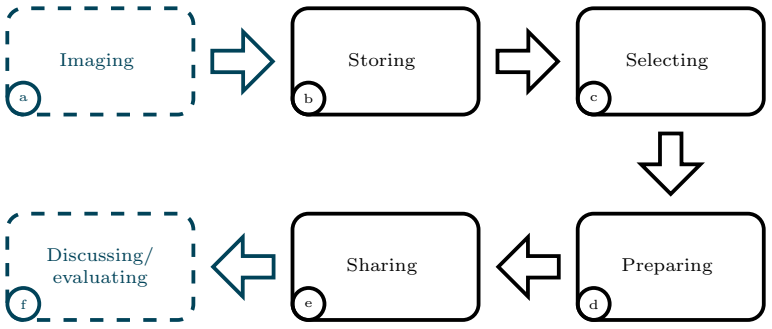


FIGURE 6.3: Simplified overview of use cases. Dashed parts are important to the project and the image workflow but are outside the scope of this thesis. Each of the steps (a) to (f) are described in more detail in Sections 6.4.1 to 6.4.5.

both biological content and image quality. Images of microscopy slices and µCT slices are then assessed for technical quality or biological data content by image experts or histopathologists.

6.4.1 Storing New Scans

Use case: **An image has been scanned on the µCT, and the scan operator wants to store the image. (Figure 6.3 a,b)**

By using Mata as described in Section 3.2, it is possible to store images generated by the newly developed medical µCT scanner by merely copying the image files into the file store.

The file watcher will detect the new files and create an entry in the database. It will also trigger the code for creating a preview of the image. The operator can add file permissions and metadata as appropriate. In many practical scenarios, it will make sense to add an HDC plug-in which reads out metadata from files and adds it automatically as described in Scott et al. (2014). For the project discussed in this thesis there was no relevant meta-data in the image files as they were saved as raw data.

If metadata is created in a consistent way by the scanner, a plug-in for the file watcher can be written to automatically add the metadata to the database as soon as the image gets uploaded. DICOM devices can also be supported by adding a DICOM receiver. An administrator is required to set permissions for any received data since DICOM does not transfer user information during file exchange. This is due to DICOM being patient centred.

6.4.2 Finding Relevant Images

Use case: **A medical researcher tries to find and get hold of images relevant to their project. For that, they want to find images of a lung taken by µCT and preview them to confirm that they fit with their needs. (Figure 6.3 c)**

The researcher has various options for finding relevant images, as discussed in Section 3.2.3. A basic search for a relevant keyword or browsing through the tags can help obtain a list of relevant datasets. If images are not in the desired format or processing stage, it is possible to find related

datasets through the network graph. With the help of MCTV and viewstl (Section 5.1), it is possible to preview images before downloading them, even if their size of several tens of GB exceeds the machine's RAM. Download of images can be conveniently achieved through the file explorer.

6.4.3 Finding Related Images

Use case: **The researcher wants to retrieve the microscopy scan of the same sample as the CT scan they are currently viewing. (Figure 6.3 c)**

Again, the researcher has various options. The two images are likely to share a tag, such as a sample identification number, or they are related through the network.

6.4.4 Processing Images

Use case: **The researcher found a relevant image and tries to process it for presentation to the experts. (Figure 6.3 d)**

After relevant images have been found, they can be processed using any suitable processing software.

LungJ, as introduced in Chapter 4, can be helpful in changing the image contrast or segmenting relevant areas. Since the images are accessible via a network file share, ImageJ can read the files directly from the remote storage. LungJ can then be used to split the image into blocks which can be processed individually (see Section 4.2.2) either using machine learning as proposed in Section 4.2.3 or using any other, possibly custom-developed algorithm to segment images. Resulting images can be reassembled with LungJ.

A point-based registration algorithm for the alignment of µCT and microscopy images can be rewritten to work with partial image data from LungJ or be executed on a high-specification computer. This is not limited to ImageJ: any processing software or self-written code that can read from a file store can be used for this task.

After processing, the new data is stored in a new folder on the file store to create a new dataset in Mata and share it with other specified researchers. Newly found metadata can also be added to Mata, and a custom HDC plug-in for importing this metadata can be developed once medical researchers decided on a processing workflow.

6.4.5 Sharing and Evaluating Images

Use case: **A set of images is to be shared with experts for evaluation. (Figure 6.3 e, f)**

In order to share a dataset with a colleague, the web link to that dataset can be sent to the other person. Provided that the other person is a user of Mata, they can access the dataset as soon as the owner gives them dataset access rights through the web interface. This can be read or read/write access as explained in Section 2.5.3.

For quick evaluation, web-viewers for volumetric and surface files are included in the storage system. Depending on the intention of the researcher, images can be visualised in 2D using tools implemented in LungJ and compared to other 2D images. Alternatively, the researcher can generate a surface file from the segmentation of a volumetric image and either display it using 3D rendering or by creating a 3D print.

6.5 Summary

In this chapter, we summarised the various approaches for medical images and focused on the workflow as a whole. We showed to what extent the approaches fulfil the requirements we defined for a medical image workflow: data security, metadata support, data traceability, system modularity, fast response and visualisation, large image support and usability. We discussed limitations to the workflow, such as hardware limitations and discussed possible extensions of our work. We evaluated if and how each of these can be addressed. Finally, we showed that the existing approaches work together to enable medical research using large images.

The next chapter will summarise this thesis, evaluate the thesis objectives and propose future work.

Chapter 7

Conclusions

IMAGES produced for medical research are growing in size. The underlying workflow for handling image data during its life-cycle is not prepared to cope with this increased amount of data.

In this thesis, we looked at the different stages of the workflow and developed approaches to ensure that large images can be handled. We created an image storage system, Mata[1], that can store large images and related metadata and give users direct access to the data. We developed an approach for processing large images using existing algorithms and implemented it as a plug-in, LungJ[2], for the popular processing software ImageJ. We also considered a variety of visualisation options and developed MCTV[3], a web-viewer, as well as visualisation functions for LungJ and an approach for visualising digital images through 3D printing.

In this final chapter, we will review the thesis objectives in Section 7.1 and explain how each of the objectives has been achieved. We will also revisit the research hypothesis in Section 7.2. Finally, we will show possible improvements to the presented work and how they can lead to future research in Section 7.3.

7.1 Review of the Objectives

In this section, we will review each of the objectives defined in Section 1.5.2 and see how they have been met in the thesis.

1. To provide a framework for image storage that can be utilised by medical researchers and by the software that is commonly used by the researchers.

We discussed the development of the Mata framework in Chapter 3. Using Mata, data can be retrieved via a shared file store. This means that no plug-in or external software is required for medical researchers to interact with their data. It also allows the researchers to utilise any image processing software that can read files from a file store in their research.

Image storage includes the management of related metadata (Section 3.2.2). We ensured that users of the framework can add dataset descriptions and metadata in the form of tags. We

[1]Source code and documentation published as doi:10.5258/SOTON/D0430 (Apache 2.0 License)
[2]Source code and documentation published as doi:10.5258/SOTON/401280 (Apache 2.0 License)
[3]Source code and documentation published as doi:10.5258/SOTON/400332 (Apache 2.0 License)

further enabled the ability to add relations between datasets. Both tags and relations can be visualised through various options, as described in Section 3.2.3, to quickly find related and relevant datasets.

2. To develop a method that enables the processing of very large images using existing software and processing algorithms.
In Chapter 4, we created a novel approach to process large images. We ensured that this approach can be applied to images of any size by splitting images into smaller blocks depending on the performance capabilities of the hardware. These blocks then get processed individually using existing processing algorithms. Global information is gathered when the blocks are created and can be used during the processing of the blocks. The use of iterative methods or a combination of several processing methods consecutively can be achieved by performing a halo exchange between individual iterations.

We also explained how this approach can be applied to different types of image processing algorithms. The approach extends to any spatial filter and can be extended to iterative image processing methods, as well as machine learning methods based on these. The approach is limited in its application to frequency-based filters since the domain transfer has to occur prior to the chunking and therefore does not benefit from the smaller image size.

3. To implement the processing method from objective 2 as a software or plug-in and show that it can be applied to an existing image processing algorithm.
We implemented the approach for processing large images as LungJ, a plug-in for ImageJ. We were able to include all the different parts, i.e. the chunk creation, halo exchange and re-assembly of image blocks into a single result image. We then used this plug-in to process large µCT images. We successfully demonstrated that the tool can apply a machine learning algorithm to segment a large image in Section 4.2.3 and Section 4.3.

4. To develop visualisation methods for medical researchers to view large 3D images in a way familiar to them.
We created MCTV and integrated it into Mata from Section 3.2 in Section 5.1. MCTV allows viewing 2D slices of images as is common in radiology and similar to histology slices. We ensured that it can handle large images by using a tiling approach to create a multi-resolution tiled version of the image to display. We analysed the speed of this technique and concluded that it offers consistent performance independent of the image size (Section 5.1.3).

In Section 5.2, we added further visualisation methods into LungJ. These allowed us to add colours to segmented features and, therefore, create colour images that highlight the features of interest, similar to a microscopy image of a stained sample.

5. To explore the possible use of new and sophisticated visualisation methods in medicine.
We considered 3D printing as a visualisation technique and evaluated 3D printing for creating training models in Section 5.3. We received positive responses from surgeons that these models are a useful alternative to existing training models since they provide higher anatomical accuracy. We also explored the possibility of AR as a faster way to view 3D images in a 3D way. We concluded that 3D printing and AR are two feasible solutions to visualising medicine images.

6. To show that all the developed tools are coherent with the requirements for a medical research image workflow.
In Section 1.4, we defined the requirements for an image life-cycle. Section 6.2 showed in detail how these requirements are satisfied by the various approaches developed in this thesis. We consistently ensured that our approaches target large images. We also paid particular attention to usability by utilising existing software to implement our approaches. By ensuring that individual parts of the workflow are exchangeable, we achieved our aim of designing the approaches in a modular way. Future research in a particular area can, therefore, be implemented without having to reinvent the whole workflow.

7.2 Review of the Research Question

In Section 1.5.1, we defined the main hypothesis of this thesis. In this section, we will answer it based on the research completed in this thesis.

What approaches support large images in medical research without significantly changing the established workflow?
Images are getting larger, and the available software and hardware are no longer capable of coping with the size of the images produced. Changing an existing workflow is not desirable since this requires researchers to learn new tools and reduces compatibility with existing research projects. Updating the hardware is a solution but comes at a cost that is not compensated by the benefits. The approaches we developed in this thesis aim to add to the existing image life-cycle workflows instead of changing the established procedures, software, or hardware. The approaches are modular to enable flexible adaptations without disturbing existing systems or workflows.

The proposed framework, for example, does not require any changes in the imaging software or the image processing software. Instead, it provides software native access to a network share. The image processing approach is designed to be very generic, and we showed how it can be applied to existing image processing algorithms. The algorithms did not have to be changed in any way. Instead, we decreased the image size and ran the algorithm more often to achieve the same result. We used a similar approach for image visualisation, by developing tools to display images in a way familiar to medical researchers, before exploring new visualisation options.

This shows that we successfully developed approaches to support the use of large images in medical research without significantly changing the established workflow.

7.3 Further Work

The limitations of the approaches proposed in this thesis have been discussed in Section 6.3. Based on these limitations, we suggest the following directions for future work.

7.3.1 Improved Image Processing Approaches

In this thesis, we developed an approach to process very large images. As discussed in Section 6.3, this approach cannot be applied to processing methods, which require extensive amounts of global parameters.

We recommend further research towards providing medical researchers full freedom when choosing image processing algorithms for processing their image data. This can involve the development of new processing tools but ideally would result in a more widely applicable strategy of approaching large images. When developing such an approach, it is important to enable medical researchers to use it with existing algorithms and software so they do not require programming.

7.3.2 Improved Metadata Search Functionality

The currently used search functionality in Mata 2 is very limiting, as it only allows querying for exact phrases. Other data management systems offer more sophisticated search capabilities. The adaptation of a search function that enables querying for several disjointed phrases or the use of Boolean search operators is considered a useful addition to the image storage system.

7.3.3 Standardised Metadata Access

We store data in a shared file store to allow users direct access to their data. This gives users a wide choice of software they can use to access the images from the storage system without having to download them. We did not find a way to enable the same convenient way for accessing the metadata, since there is no standard for image software to read metadata other than that included in a file. Accordingly, future work in this area should focus on developing a standard for accessing the metadata of a research dataset. To increase its benefits, this standard can be applied to a larger community than only medical researchers. APIs, as provided by OMERO or BisQue, offer a starting point to resolving this problem.

7.3.4 Provenance

The proposed system allows for the manual logging of information about the origin of a dataset. Other image management systems reviewed provide tools for implementing trusted provenance for a workflow. An API can improve not only metadata access but also data provenance. API functions need to be provided to log information about image origin, image processing and operating user in a standardised way. This would allow programmers to implement a workflow with trusted provenance.

7.3.5 Web-Based Visualisation of Large 3D Surface Files

In Section 5.3, we explored new visualisation techniques. Based on the exploration, we did not manage to fully exhaust this field to show the benefits that new visualisation techniques can

give. We recommend further research to find fast visualisation methods for large 3D surface files over the internet. This may well be based on advances in game engines. We also recommend further investigation of the use of 3D printing and AR to help medical researchers understand 3D image data in general.

7.3.6 Automated Image Comparison

By storing images in a central storage system, many hospital samples can be digitalised and accumulated over time. The use of metadata to find useful images is limited and more powerful tools will be required in the future. Automatically detecting properties of the scanned organs or tissues would make it possible to compare images and determine their similarity. This allows showing researchers images that are similar to the one they are currently investigating.

As a further step, the practise of finding similarity could be applied to images in diagnosis. If processing and feature detection of new images can be achieved fast enough, they can aid doctors in their diagnosis. The system can present not only the result of the imaging done on the patient but also other similar looking scans and the diagnosis for those similar cases. In this way, doctors can draw from a much larger knowledge base than currently available when assessing new cases.

7.4 Overall Summary

Digitalisation of medical research and diagnosis is becoming increasingly important because it allows us to offer the best possible treatment to patients by exploiting technological advances. Due to continuous improvements in imaging techniques, the produced images have grown in size in proportion to improvements in computation. This thesis addressed the problems caused by extra-large images in three main areas of image-based research: storage, processing and visualisation of images. We created a system for managing image data throughout its life-cycle, developed an approach for processing extra-large images using existing processing algorithms, and built tools and workflows for a variety of visualisation options. We combined these tools in an end-to-end workflow, which enables medical researchers to make use of cutting-edge imaging techniques without requiring expensive cutting-edge computing hardware and software. The tools developed in this work improve the flexibility of medical researchers to adopt new imaging techniques and therefore help to discover new ways of treating humans in the future.

Appendix A

Tag Cloud Implementation

In Section 3.2.3, we discussed different ways of visualising metadata and finding relevant data in a management system based on metadata. Specifically in Section 3.2.3.2, tag clouds are introduced an a possibility to allow users quickly find relevant metadata in a folksonomy (see Section 2.1.1). This appendix will talk about the implementation of the tag cloud in a bit more detail.

In order to prioritize tags in a tag cloud, they are represented in different font sizes (and sometimes colour or layout). Tags that appear more often are larger so that very specific uncommon tags or typographical errors that only appear for a few datasets are smaller and less obvious. The main finding of Halvey and Keane (2007) was that alphabetically ordered tag clouds help users to find very specific tags. Several studies found highlight that tag clouds are slower when users try to search for a specific keyword compared to simple lists due to the varying font size (Halvey and Keane, 2007; Kuo et al., 2007; Seifert et al., 2008). In the experiments conducted in those studies, the font size was not related to the likelihood of a user trying to search for that term. As pointed out by Sinclair and Cardew-Hall (2008), tag clouds do have a use in scenarios wherein users search for generic information rather than specific key terms. Tag clouds alone are not sufficient for all scenarios of users searching for datasets. We took all the mentioned studies into account and maintained an alphabetical order of tags, and minimized the spread of font size between tags.

Zipf's Law (Zipf, 1949) was assumed to be applicable to the occurrence of specific tags. This is a reasonable assumption, as shown by Li (1992). For the scaling of words in our tag cloud, we mapped the power-law distribution from Zipf's Law to a linear distribution. Zipf's Law for language states that the probability of occurrence of the n-th most common word in a text is inversely proportional to its rank n, or more precisely:

$$p(n) = \frac{1}{n \ln 1.78N},$$

where N is the total number of elements. This means that the n-th most common word in a text occurs $1/n$ times as often as the most common word. Taking $\tilde{s}$ as the spread of font size and $\hat{s}$ as the maximum font size, one can linearise occurrences following Zipf's Law as

$$\frac{\hat{x}}{n} \to \hat{s} - \frac{n-1}{N-1}\tilde{s},$$

where $\hat{x}$ the maximum number of occurrences of any single element. Therefore, the font size of an element that appears x times is

$$s(x) = \hat{s} - \frac{\frac{\hat{x}}{x} - 1}{N - 1}\tilde{s}$$

One can see that the factor in front of $\tilde{s}$ is smaller than or equal to 1 as long as the most common tag does not occur more often than the total number of tags ($\hat{x} \leq 1$). As potentially s can turn negative with this linearisation, a minimum font size has to be hard-coded. This is even more important as Zipf's Law for word frequencies breaks down after three to four orders of magnitude (relatively soon for small sample sizes) (Williams et al., 2015).

In this appendix, we showed how the tag cloud was implemented for browsing user-defined tags using Zipf's Law. Unfortunately, not enough metadata is stored in the file system to allow for a formal evaluation of the tag cloud. We still implemented the tag cloud as an important part of a GUI, as suggested by Halvey and Keane (2007) and Sinclair and Cardew-Hall (2008).

Appendix B

Halo Exchange Implementation

As part of the LungJ workflow that splits large images into blocks, we implemented the option of using halos and performing a halo exchange. Every small block of the large images lies next to each other, in order to cover the whole image. Halos are added to each side of the block, causing an overlap of pixels between adjacent blocks. For image processing algorithms that are dependent on the boundary of the image, this adjacent data can be used for updating the actual content of the block. After performing the processing algorithm, the halos themselves usually contain inaccurate or unchanged data. In order to perform a sequential algorithm, they need to be updated with correct data from the neighbouring blocks. This is referred to as a '*halo exchange*'. A function to achieve this kind of exchange has been implemented as part of LungJ, in order to enable the application of LungJ to more complicated processing methods.

Since LungJ was implemented to avoid large amounts of memory being used, we considered it important to also reduce the amount of memory a halo exchange consumes. One could load all the blocks at the same time and perform the exchanges before saving the updated blocks again. This would require a lot of memory, since the number of pixels loaded at the same time equals the size of the image plus the size of all the halos of the blocks. The opposite extreme would be to load only two images at a time, perform the exchange between them and then load the next two. This requires only two blocks and their halos to be loaded at any one time, but also means that every image needs to be loaded several times during the complete halo exchange. In the following section, we explain how we limit the amount of data loaded at any time. Appendix B.2 will then describe the implementation of the halo exchange in more detail, with Appendix B.3 paying specific attention to Java and ImageJ; before Appendix B.4 discusses the chances involved in and limitations of our approach.

B.1 Limiting the Maximum Memory Used

As mentioned in the introduction to this appendix, loading all the image blocks at the same time requires a lot of memory. The larger the image is, the more memory would be required. It makes sense to limit the amount of memory required by having a maximum number of blocks loaded at the same time. The smallest possible number of maximum simultaneously loaded blocks is 2, since at least two blocks are required to perform a data exchange. Loading only a small number of blocks at the same time means that some blocks need to be loaded more than once. This

means that a small maximum memory leads to a large number of file loads, which take a long time. Hence, we need to find a balance between having as few blocks as possible loaded at the same time and having as few total file loading operations as possible.

As the first step, we will calculate the two extreme cases as references.

In the case where all blocks are loaded at the same time, the total number of file loads is B, where B is the number of blocks in the image. The maximum memory requirement in terms of blocks is also B.

In the case of only 2 blocks being loaded at the same time, the total number of file loads will be of the order of $O(13B)$. This is because there are 26 neighbours per block (ignoring blocks on the boundary) and one exchange will update the data for each of the two neighbouring blocks. The maximum memory required in terms of blocks is only 2, however.

The next obvious choice is 8 blocks. We managed to implement an algorithm that performs the halo exchange with 8 simultaneously loaded blocks in $(k-1)(m-1)(n-1)$ iterations, where each iteration uses half of the blocks of the previous iteration, thus loading only 4 new blocks. This means that the total number of file loads of this algorithm for one halo exchange is

$$4(k-1)(m-1)(n-1)+4$$

$$= 4kmn - 4(km+kn+mn) + 4(k+m+n) = O(4B)$$

The details of the implementation can be found in the next section. This does not prove that this is the most efficient possible implementation, but it was the most obvious for us, as we attempted to avoid loading files more often than necessary while keeping the iterations as simple as possible. Given that with each file load a maximum of 7 new exchanges can be done, the theoretical optimum is of the order of $O(\frac{13}{7}B)$. It is possible to show that one cannot carry out all halo exchanges in a way that every file load achieves the maximum of 7 new halo exchanges. Therefore, the optimum algorithm would be $O(1.8B) < x \leq O(4B)$. With $O(4B)$ file loads, the 8 block limit is significantly better than the 2 block limit with $O(13B)$ file loads. It is 4 times worse than the implementation loading all the blocks, but it has a constant memory usage, which means that the halo exchange can be performed for any image size, as long as 8 image blocks including their halos can be comfortably loaded into memory. No detailed analysis comparing the maximum number of blocks loaded to the total number of block loads was made. The use of 8 blocks was sufficient for our purposes.

B.2 Implementing the Halo Exchange

In the previous section, we explained why we decided to load 8 blocks at the same time. In this section, we will go into details of the implementation.

First, we will look at the iterations taken by the algorithm for choosing the blocks to perform an exchange on. For illustration purposes, Figure B.1 shows how the algorithm works in a 2D case with four simultaneously loaded blocks. We simplify the drawing by showing blocks as equal-sized squares. In reality, they can have any rectangular shape. In particular, the blocks

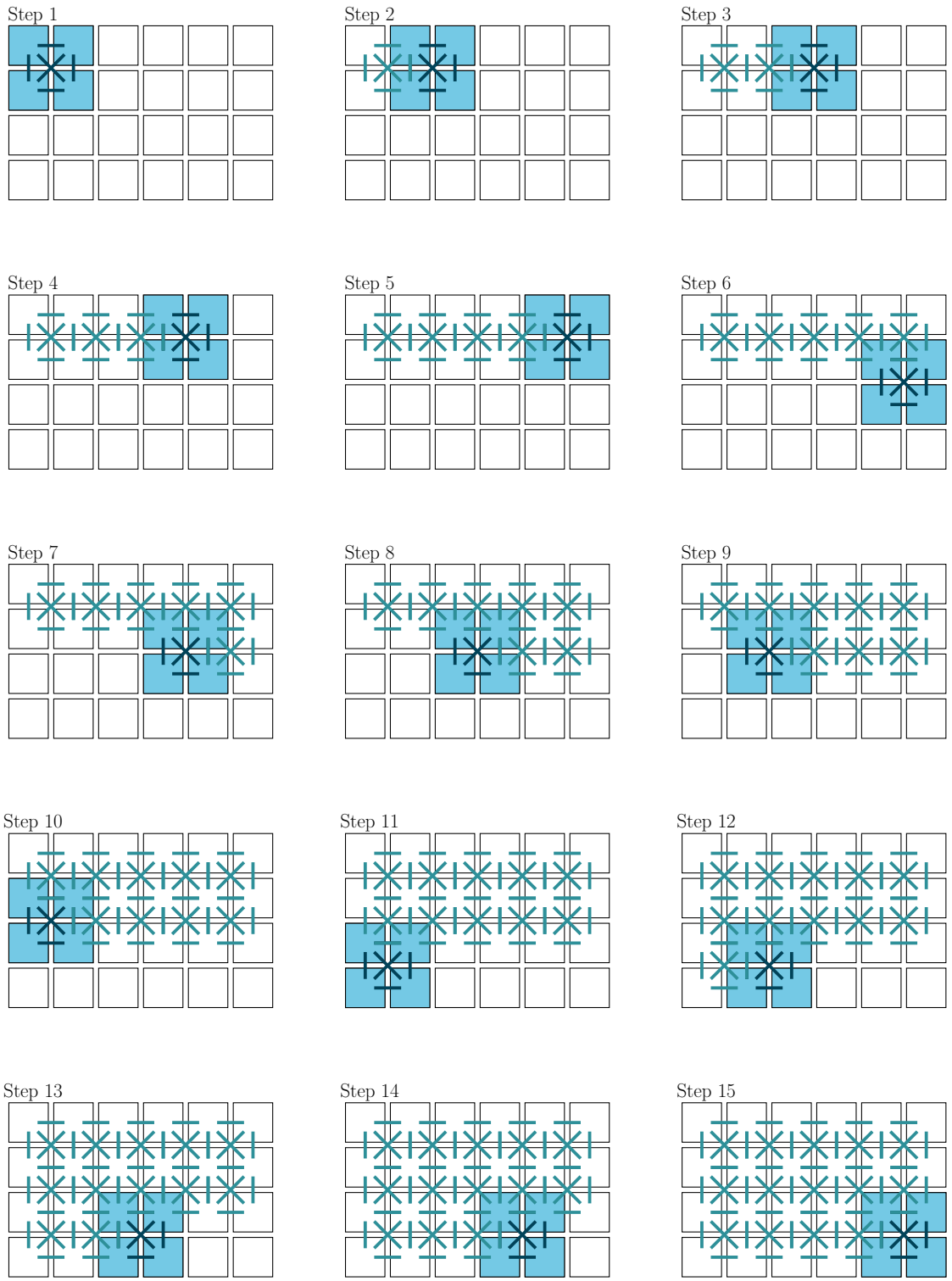


FIGURE B.1: Halo exchange in 2D: example block selection for a 4×6 block image.

on the far edges may be smaller than the remaining blocks, since the block size does not need to be a multiple of the image size. A further simplification is that we omitted the halos. Instead, only the core of each block is shown. Further, the blocks are drawn at a slight distance to make them more clearly distinguishable.

Following Figure B.1, one starts in the top left corner and loads 4 blocks in a square. Now one progresses along one axis, loading 2 new squares in each iteration (Steps 2 to 5) until the end of the image is reached. The blocks are then moved one unit into the other axis direction (Step 6) and moved in the opposite direction along the first axis (Steps 7 to 10). This process repeats until the block cannot move anymore. It then has gone through all the required iterations. In each iteration, all the new halo exchanges possible are performed (red lines). In 3D, the process is the same, but with 8 blocks and 3 axes to move along.

B.3 Managing Memory in Java

Java works with a garbage collector to manage its memory usage (Oracle, 2009, Chapter 3). Variables declared in Java are references to objects which are stored in a heap. Java will run a garbage collection to release the memory from any objects no longer in use. In order to speed up the process, the heap memory is split into several parts. In one part, the nursery, newly created objects reside. The other parts contain old objects that have been in use for longer. The garbage collection will first attempt to clear the nursery before going through the old objects. Large objects will directly go into the part for old objects (Oracle, 2009, Chapter 3).

While the garbage collection is great for programming, since programmers do not have to worry about de-allocating memory, the specific implementation of the garbage collection can cause trouble when dealing with many large objects. From our experience, ImageJ seems to have issues releasing memory allocated to images no longer open. This means that is important to ensure that data no longer needed can be collected as soon as possible by the Java garbage collector to release memory. A full memory slows down other applications. The Java application itself is not affected, since it will eventually clear old objects memory if it requires the space itself.

In order to help the memory release, image objects were loaded and assigned to the same variables. This means that old image blocks have no variable pointing to them, so that the garbage collection can remove them.

B.4 Summary

The use of halos is especially useful when working with processing methods that rely on boundary values such as spatial filters. It becomes a necessity in the case of iterative processing methods or spatial filters being executed consecutively. In this appendix, we discussed the implementation of a halo exchange for LungJ blocks in ImageJ. We showed that limiting the number of simultaneously loaded blocks to 8 allows to set a boundary to the maximum memory requirement in terms of image blocks. At the same time, it requires fewer file loads than lower maximum block

numbers. Choosing the ideal number of blocks depends on the specific system. Through the choice of appropriate block and halo sizes, the user can ensure the halo exchange implementation runs independently of the total image size and system RAM.

We gave a detailed explanation of how this algorithm chooses the blocks for an exchange by moving along the axis of the image in a snake-like motion. We also took issues arising from Java memory management into consideration. Even though no mathematical proof is given, to show if this implementation is the most efficient for 8 blocks, we can say that it is very efficient compared to alternative implementations.

Overall, this implementation presents a useful tool for working with LungJ blocks and helps make LungJ applicable to a larger variety of applications.

Appendix C

Trainable WEKA Segmentation Filters

This appendix lists all filters used by the TWS and visualizes them for an example input. Information and text taken from the official documentation yet the source code was analysed and some information was added, corrected or modified to match the current implementation. Each filter was evaluated for the 8-bit grayscale version of a lung tissue image shown in Figure C.1.

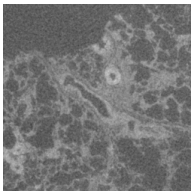
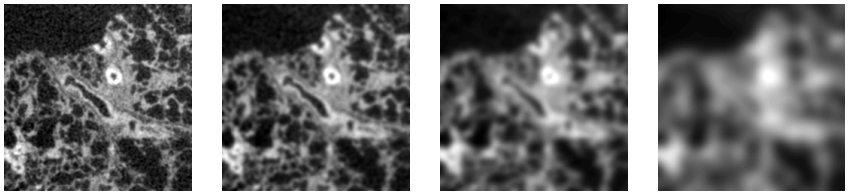


Figure C.1: Example image prior processing

Available implementations of the filters in Fiji are listed in the beginning of each section. Often a 3D version of the filter is available. At the time LungJ was created, the TWS mostly used its own implementation of filters and always used 2D filters. In the most recent release, a special 3D version of the TWS exists.

C.1 Gaussian Blur

The Gaussian blur performs n individual convolutions with Gaussian kernels with σ equal to $1, 2, 4...2^{n-1}$. By default, $n = 5$ but due to space limitations we will present the results for $n = 4$ in this appendix. The larger the radius is, the more blurred the image becomes until the pixels are homogeneous. An example output is shown in Figure C.2.

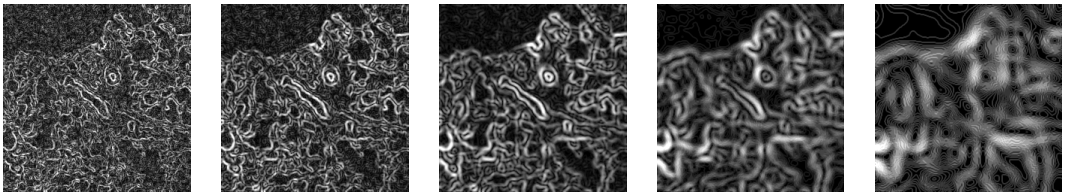


$\sigma = 1$ $\sigma = 2$ $\sigma = 4$ $\sigma = 8$

FIGURE C.2: Gaussian blur applied to the image in Figure C.1

C.2 Sobel Filter

The Sobel filter calculates the gradient at each pixel. This enhances the edges as shown in Figure C.3. Gaussian blurs with $\sigma = 1, 2, 4...2^{n-1}$ are performed prior to the filter.



$\sigma = 0$ $\sigma = 1$ $\sigma = 2$ $\sigma = 4$ $\sigma = 8$

FIGURE C.3: Sobel filter applied to the image in Figure C.1

C.3 Hessian

This filter calculates a Hessian matrix H at each pixel. Prior to the application of any filters, a Gaussian blur with $\sigma = 1, 2, 4...2^{n-1}$ is performed. The Hessian matrix (see Figure C.4) is computed using the 3×3 Sobel operator. The output of the Hessian filter enhances edges.

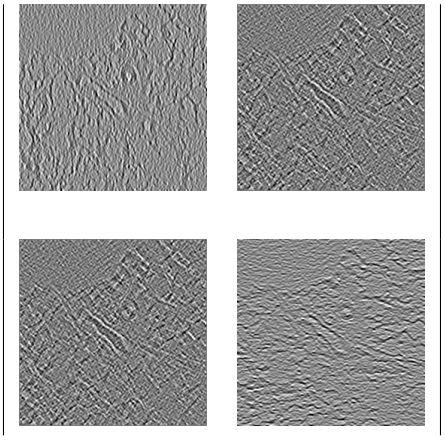
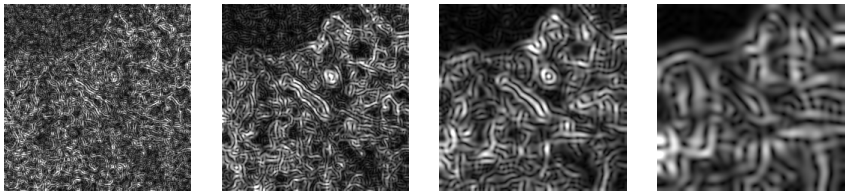


FIGURE C.4: Hessian matrix based on the image in Figure C.1



$\sigma = 1$ $\sigma = 2$ $\sigma = 4$ $\sigma = 8$

FIGURE C.5: Module of the Hessian matrix in Figure C.4

The final features used for pixel classification, given the Hessian matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$, are then calculated:

- Module: $\sqrt{a^2 + bc + d^2}$ (see Figure C.5).
- Trace: $a + d$ (see Figure C.6).
- Determinant: $ad - cb$ (see Figure C.7).
- First eigenvalue: $\frac{a+d}{2} + \sqrt{\frac{4b^2+(a-d)^2}{2}}$ (see Figure C.8).
- Second eigenvalue: $\frac{a+d}{2} - \sqrt{\frac{4b^2+(a-d)^2}{2}}$ (see Figure C.9).
- Orientation: $\frac{1}{2}\operatorname{acos}\left(4b^2 + (a-d)^2\right)$ This operation returns the orientation for which the second derivative is maximal. It is an angle returned in radians in the range $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$ and corresponds to an orientation without direction. The orientation for the minimal second derivative can be obtained by adding (or subtracting) $\frac{\pi}{2}$ (see Figure C.10).
- Gamma-normalized square eigenvalue difference: $t^4(a-d)^2\left((a-d)^2 + 4b^2\right), t = 1^{3/4}$. Note that t is a parameter commonly chosen as a number raised to the power of 0.75. In this implementation, it is hardcoded as 1 (see Figure C.11).
- Square of gamma-normalized square eigenvalue difference: $t^2\left((a-d)^2 + 4b^2\right), t = 1^{3/4}$. Note that t is a parameter commonly chosen as a number raised to the power of 0.75. In this implementation, it is hardcoded as 1 (see Figure C.12).

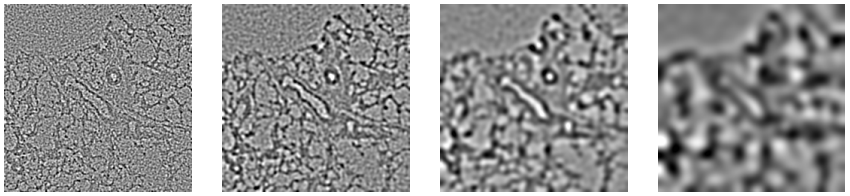


FIGURE C.6: Trace of the Hessian matrix in Figure C.4

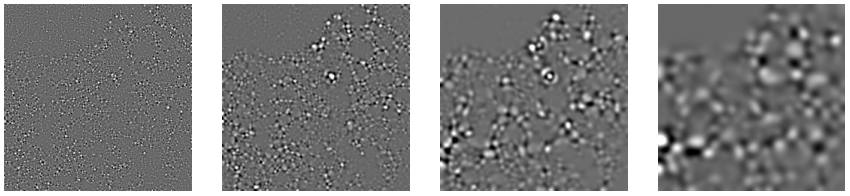


FIGURE C.7: Determinant of the Hessian matrix in Figure C.4

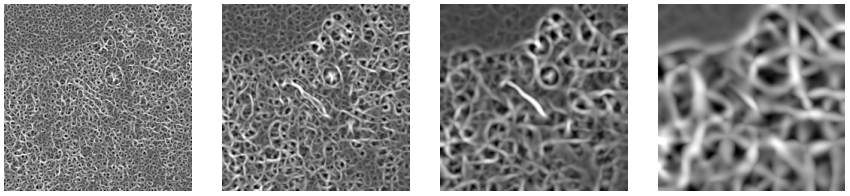


FIGURE C.8: First eigenvalue of the Hessian matrix in Figure C.4

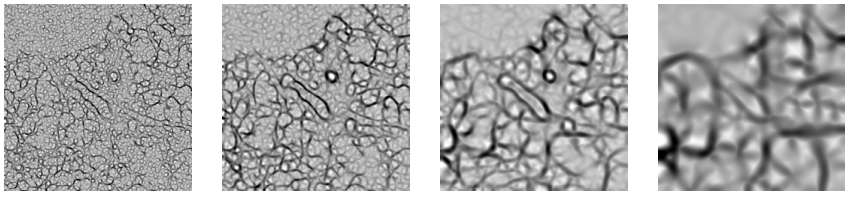


FIGURE C.9: Second eigenvalue of the Hessian matrix in Figure C.4

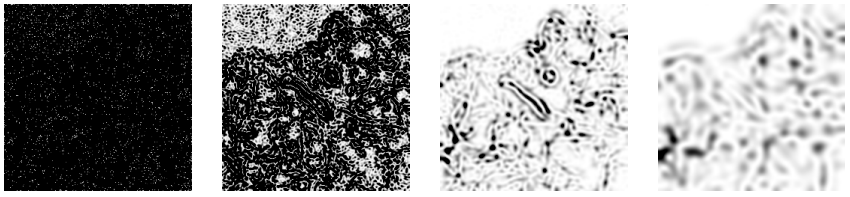


FIGURE C.10: Orientation of the Hessian matrix in Figure C.4

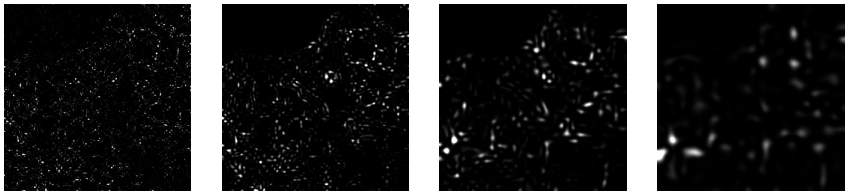


FIGURE C.11: Gamma-normalized square eigenvalue difference of the Hessian matrix in Figure C.4

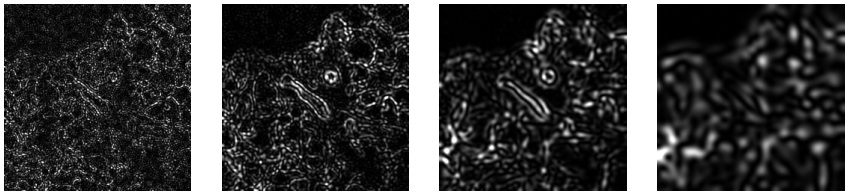


FIGURE C.12: Square of gamma-normalized square eigenvalue difference of the Hessian matrix in Figure C.4

C.4 Difference of Gaussian

This filter calculates two Gaussian blur images from the original image and subtracts one from the other. σ values are again set to $1, 2, 4...2^{n-1}$, so $n(n-1)/2$ feature images are added to the stack. The result can be seen in Figure C.13.

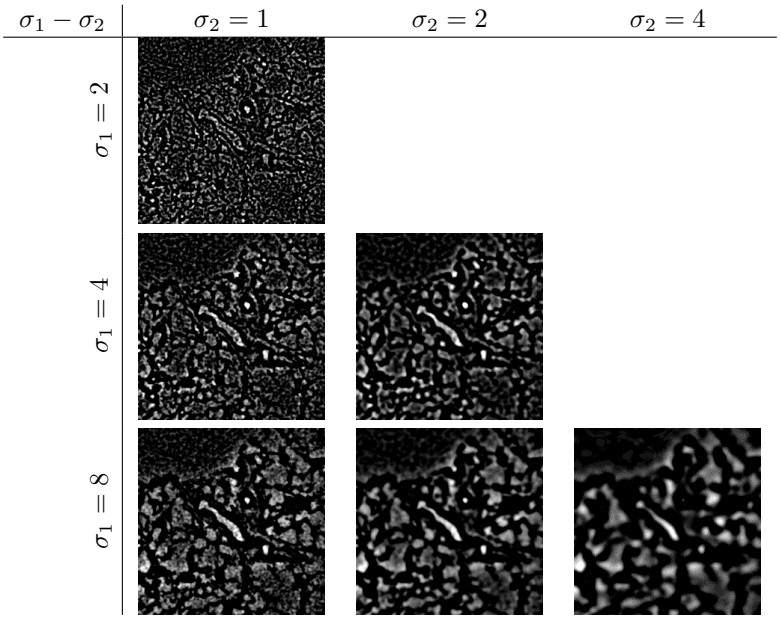


FIGURE C.13: Difference of Gaussian of the image in Figure C.1

C.5 Membrane Projections

The initial kernel for this operation is hardcoded as a 19×19 zero matrix with the middle column entries set to 1. The membrane size, t_m, determines how many columns are filled with ones. Multiple kernels are created by rotating the original kernel by 6 degrees up to a total rotation of 180 degrees, giving 30 kernels. Rotations refer to in-built image rotations. Each kernel is convolved with the image and then the set of 30 images are Z-projected into a single image via 6 methods:

1. Sum of the pixels in each image (see Figure C.14).

2. Mean of the pixels in each image (see Figure C.15).

3. Standard deviation of the pixels in each image (see Figure C.16).

4. Median of the pixels in each image (see Figure C.17).

5. Maximum of the pixels in each image (see Figure C.18).

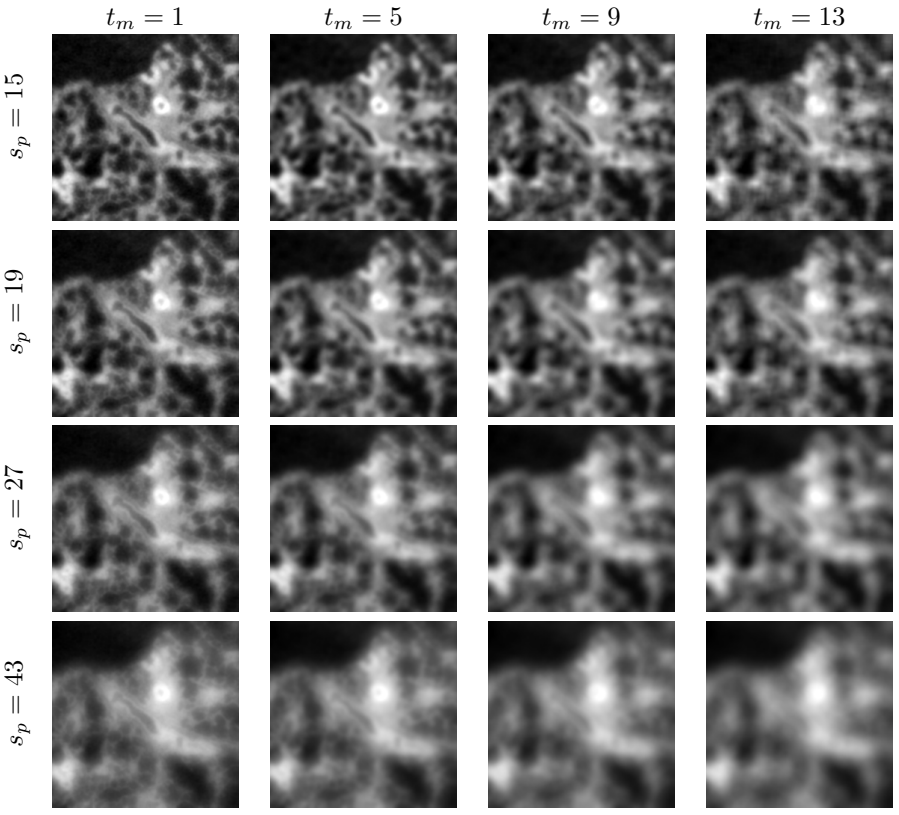


FIGURE C.14: Sum of the pixels in each membrane of the image in Figure C.1

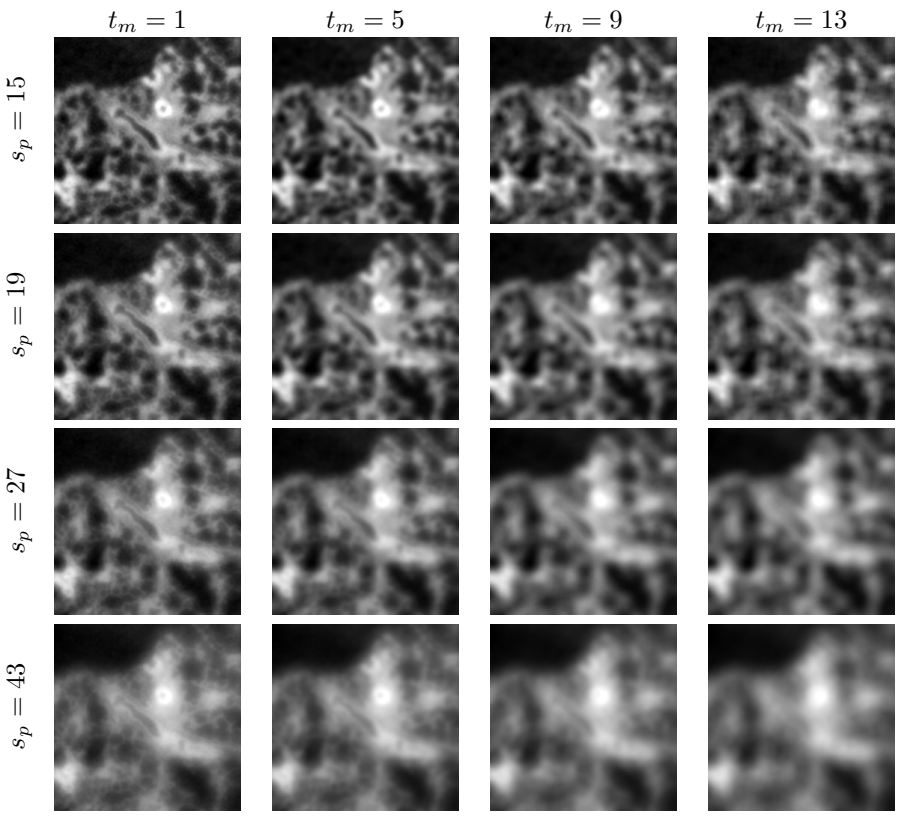


FIGURE C.15: Mean of the pixels in each membrane of the image in Figure C.1

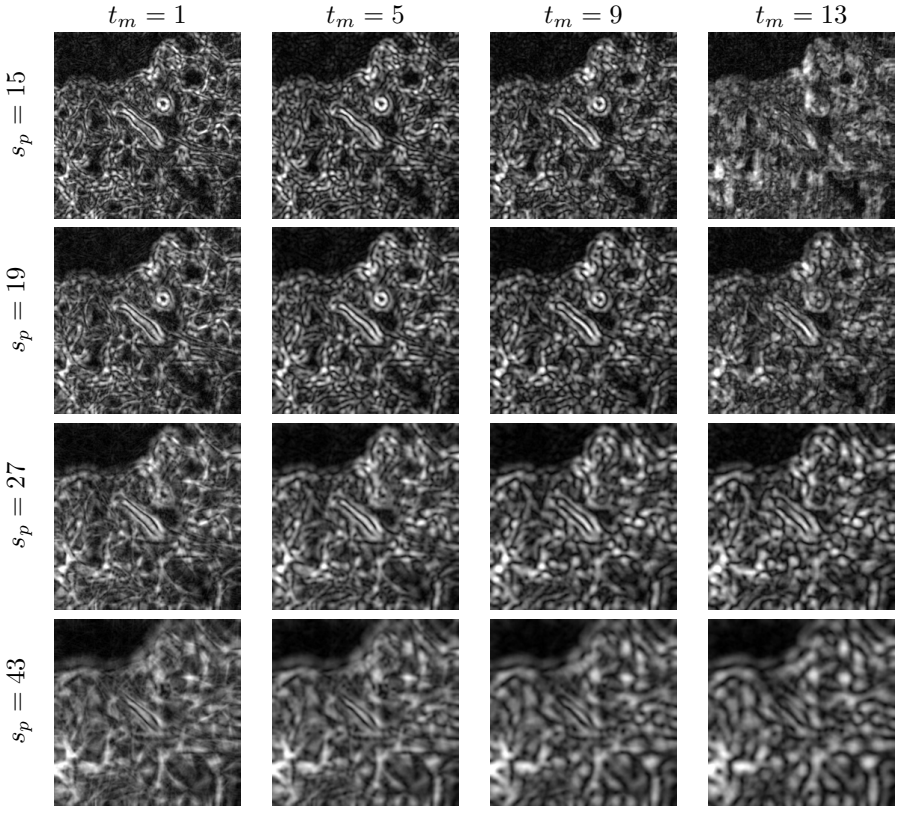


FIGURE C.16: Standard deviation of the pixels in each membrane of the image in Figure C.1

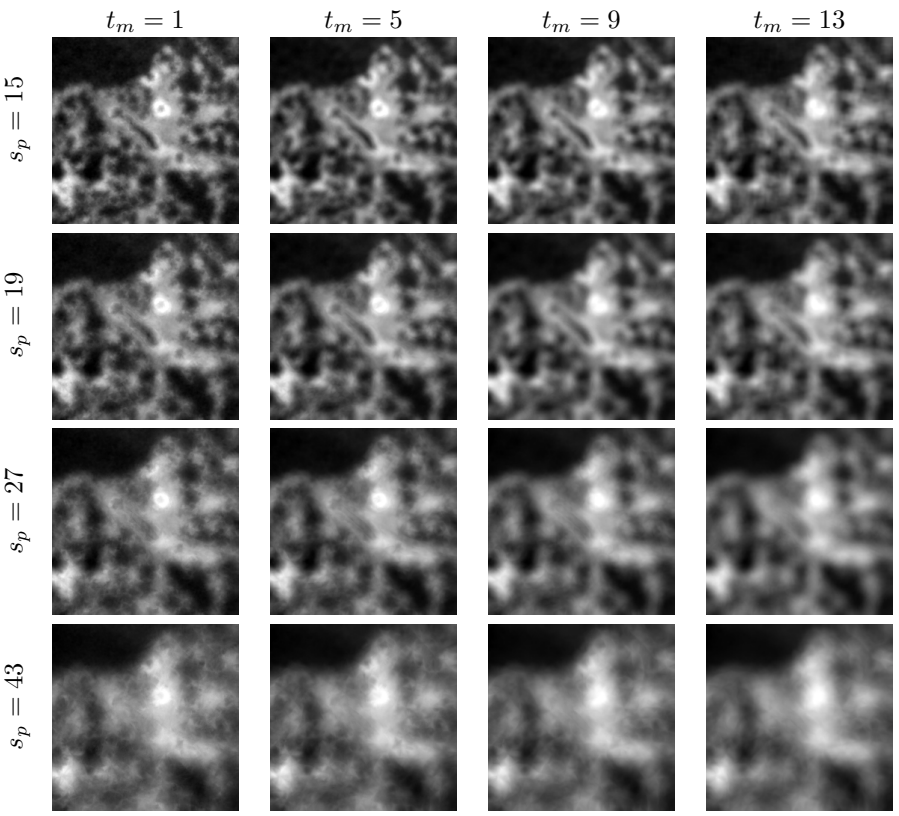


FIGURE C.17: Median of the pixels in each membrane of the image in Figure C.1

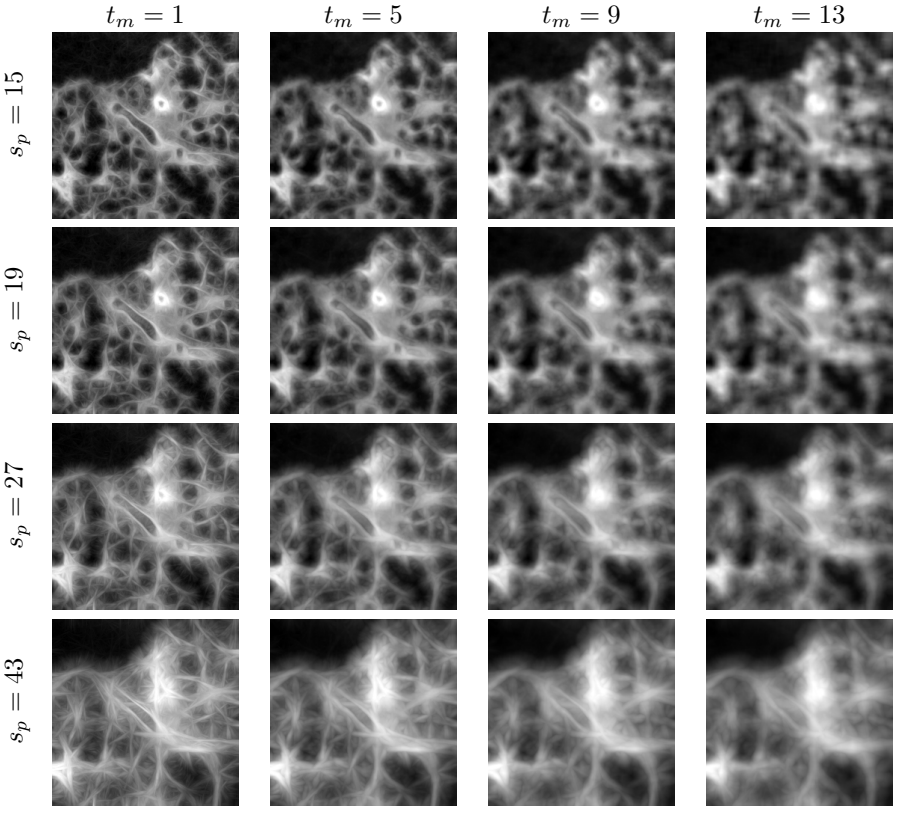


FIGURE C.18: Maximum of the pixels in each membrane of the image in Figure C.1

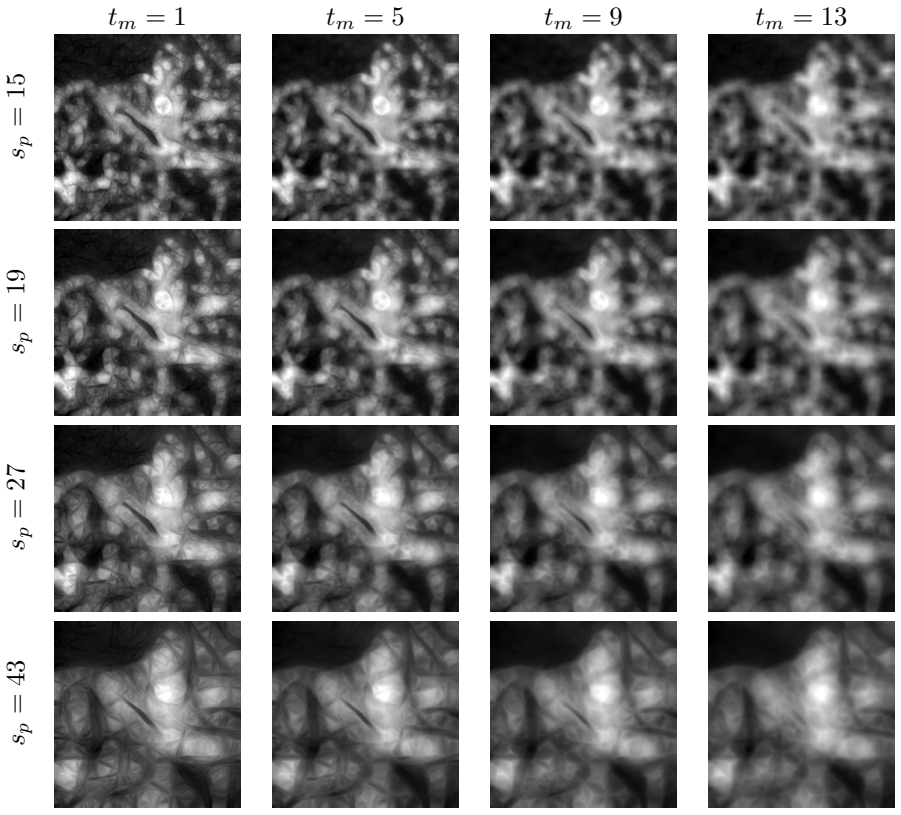


FIGURE C.19: Minimum of the pixels in each membrane of the image in Figure C.1

6. Minimum of the pixels in each image (see Figure C.19).

Each of the 6 resulting images is a feature. Hence, pixels in lines of similarly valued pixels in the image that are different from the average image intensity will stand out in the Z-projections.

C.6 Variance

The pixels within a radius of $1, 2, 4...2^{n-1}$ pixels from the target pixel are subjected to the pertinent operation (variance) and the target pixel is set to that value. This highlights structures, as can be seen in Figure C.20.

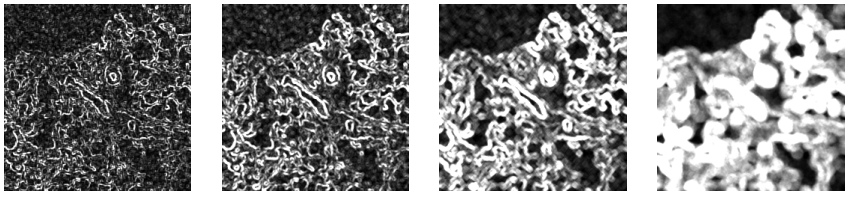
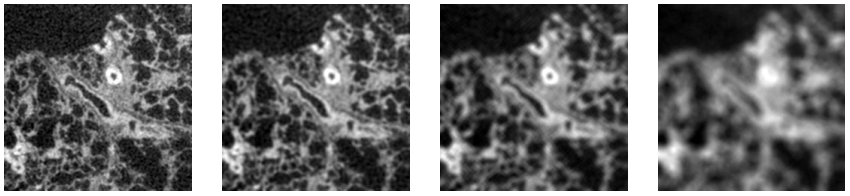


FIGURE C.20: Variance of the image in Figure C.1

C.7 Mean

The pixels within a radius of $1, 2, 4...2^{n-1}$ pixels from the target pixel are subjected to the pertinent operation (mean) and the target pixel is set to that value. The result averages the image as seen in Figure C.21.

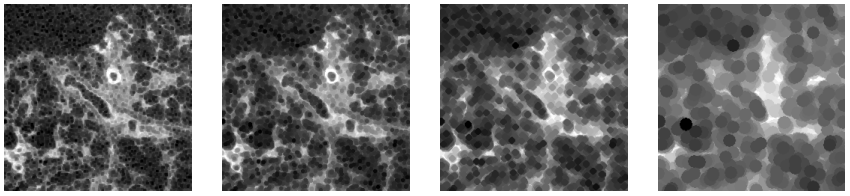


$r = 1$ $r = 2$ $r = 4$ $r = 8$

FIGURE C.21: Mean of the image in Figure C.1

C.8 Minimum

The pixels within a radius of $1, 2, 4...2^{n-1}$ pixels from the target pixel are subjected to the pertinent operation (min) and the target pixel is set to that value. The result averages the image as can be seen in Figure C.22.



$r = 1$ $r = 2$ $r = 4$ $r = 8$

FIGURE C.22: Minimum of the image in Figure C.1

C.9 Maximum

The pixels within a radius of $1, 2, 4...2^{n-1}$ pixels from the target pixel are subjected to the pertinent operation (max) and the target pixel is set to that value. The result averages the image as can be seen in Figure C.23.

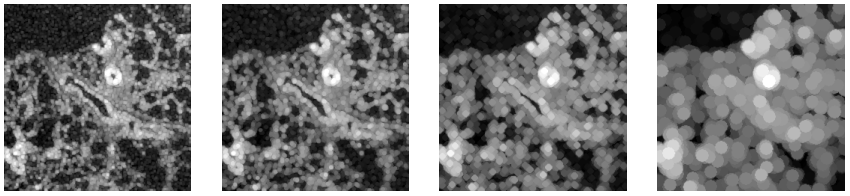


FIGURE C.23: Maximum of the image in Figure C.1

C.10 Median

The pixels within a radius of $1, 2, 4...2^{n-1}$ pixels from the target pixel are subjected to the pertinent operation (median) and the target pixel is set to that value. The result averages the image as can be seen in Figure C.24.

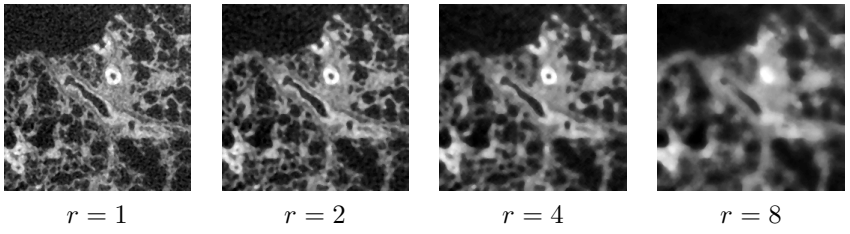


FIGURE C.24: Median of the image in Figure C.1

C.11 Anisotropic Diffusion

Anisotropic diffusion uses the image gradient to compute a diffusion tensor which, over several iterations, drives the diffusion of pixel values. This filter smoothes the image. The TWS implements anisotropic diffusion filtering from Fiji with 20 iterations, with $s = 1, 2, 4...2^{n-1}$ smoothing per iterations, where $a_1 = 0.10, 0.35$, $a_2 = 0.9$, and an edge threshold set to the membrane size. The result is an averaged image as can be seen in Figure C.25.

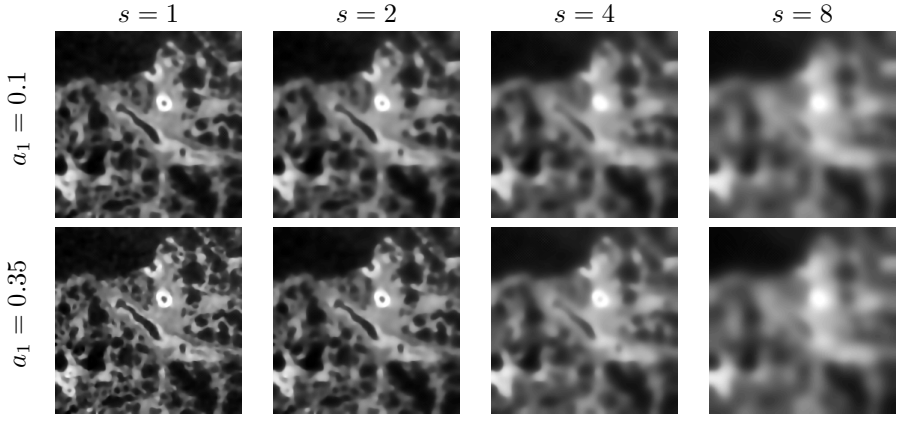


FIGURE C.25: Anisotropic diffusion filter applied to the image in Figure C.1

C.12 Bilateral

The bilateral filter is very similar to the mean filter in Appendix C.10 but better preserves edges while averaging/blurring other parts of the image. The filter accomplishes this task by only averaging the values around the current pixel that are close in color value to the current pixel. The 'closeness' of other neighborhood pixels to the current pixels is determined by the specified threshold, i.e. for a value of 10, each pixel that contributes to the current mean have to be within 10 values of the current pixel. In our case, we combine a 'spatial radius' of 5, 10 and 20, with a 'range radius' of 50 and 100. The result can be seen in Figure C.26.

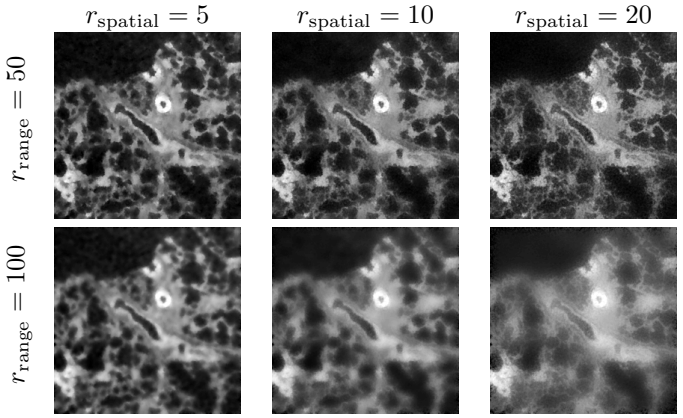


FIGURE C.26: Bilateral filter applied to the image in Figure C.1

C.13 Lipschitz

This filter is derived from the Stencel plugin. This plugin implements the Lipschitz cover of an image, which is equivalent to a grayscale opening by a cone. The Lipschitz cover can be applied for the elimination of a slowly varying image background by subtraction of the lower Lipschitz cover (a top-hat procedure). A sequential double scan algorithm by Rosenfeld was used. The TWS uses the down and top hats combination, where the slope = 5, 10, 15, 20, 25. The result can be seen in Figure C.27.

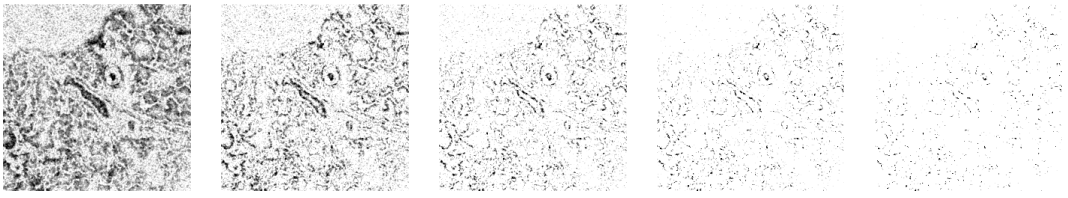
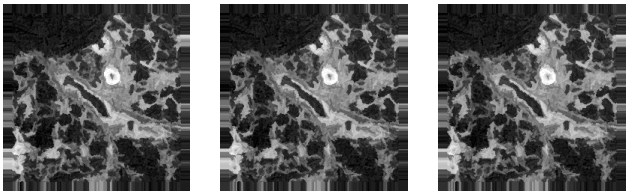


FIGURE C.27: Lipschitz filter applied to the image in Figure C.1

C.14 Kuwahara

This is another noise-reduction filter that preserves edges. It is a version of the Kuwahara filter that uses linear kernels rather than square ones. The TWS uses the 'membrane patch size' as the kernel size, 30 angles, and three different criteria (variance, variance/mean and variance/mean^2). The result can be seen in Figure C.28.



variance variance/mean variance/mean^2

FIGURE C.28: Kuwahara filter applied to the image in Figure C.1

C.15 Gabor

The Gabor filter is an edge detection filter, which convolves several kernels at different angles with an image. Each point in a kernel is calculated as

$$\frac{\cos\left(2\pi f\frac{x_p}{s_x}+\psi\right)e^{-0.5\left(\frac{x_p^2}{\sigma_x^2}+\frac{y_p^2}{\sigma_y^2}\right)}}{2\pi\sigma_x\sigma_y}.$$

Gabor filters are band-pass filters and therefore implement a frequency transformation. At the moment, this option may take some time and memory because it generates a very diverse range of Gabor filters (22). At the time of writing this thesis, it was not possible to run the Gabor filter on the sample image. According to the main developer, this filter may undergo changes in the future.

C.16 Derivatives

This filter enhances edges by computing high-order derivatives of the input image (from the 4th to the 10th derivative) using FeatureJ. It is affected by the smoothing coefficient σ and the order of derivative. Only even derivatives are computed (i.e. $\frac{\partial^4}{\partial x^2\partial y^2}$, $\frac{\partial^6}{\partial x^3\partial y^3}$...). The result can be seen in Figure C.29.

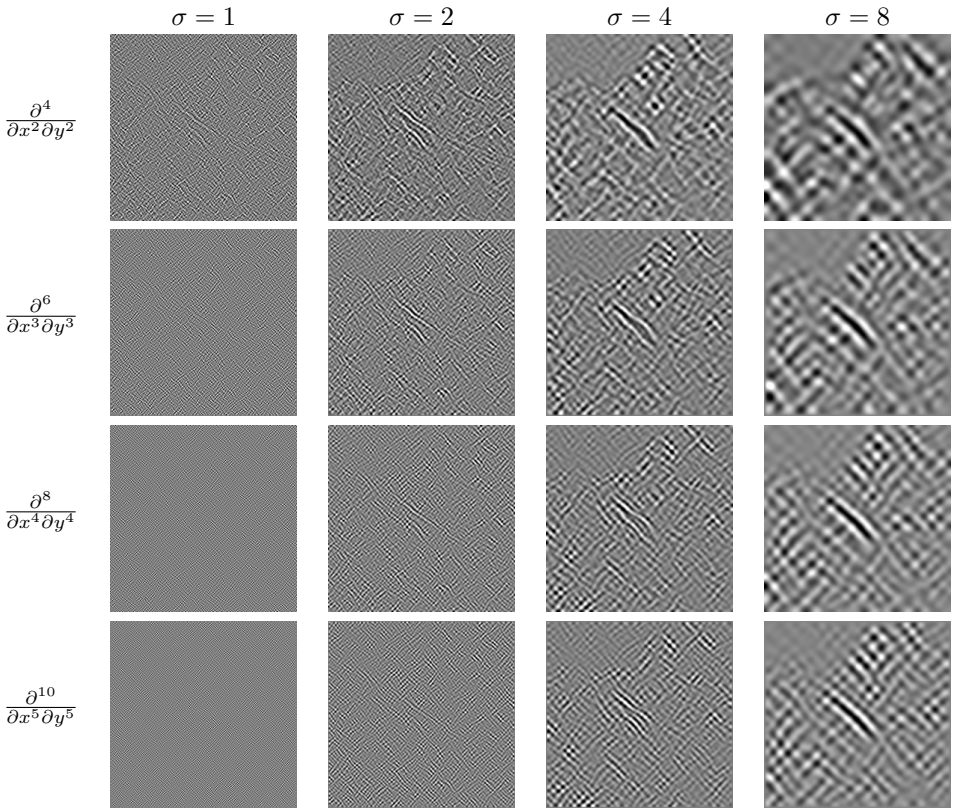


FIGURE C.29: Derivatives filter applied to the image in Figure C.1

C.17 Laplacian

This filter enhances edges by computing the Laplacian of the input image using FeatureJ. It uses a smoothing scale $\sigma = 1, 2, 4...2^{n-1}$. The result can be seen in Figure C.30.

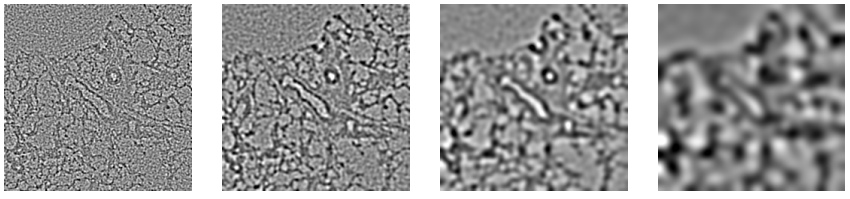


FIGURE C.30: Laplacian filter applied to the image in Figure C.1

C.18 Structure

This filter calculates the eigenvalues (smallest and largest) of the so-called structure tensor for all elements in the input image using FeatureJ. It uses smoothing scale $\sigma = 1, 2, 4...2^{n-1}$ and integration scales $s = 1$ and 3.

- Smallest eigenvalue (see Figure C.31).

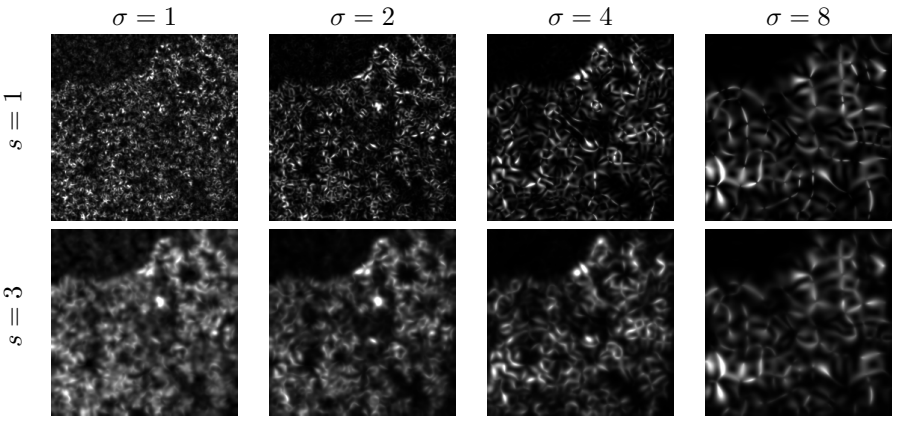


FIGURE C.31: Smallest eigenvalue of the structure filter applied to the image in Figure C.1

- Largest eigenvalue (see Figure C.32).

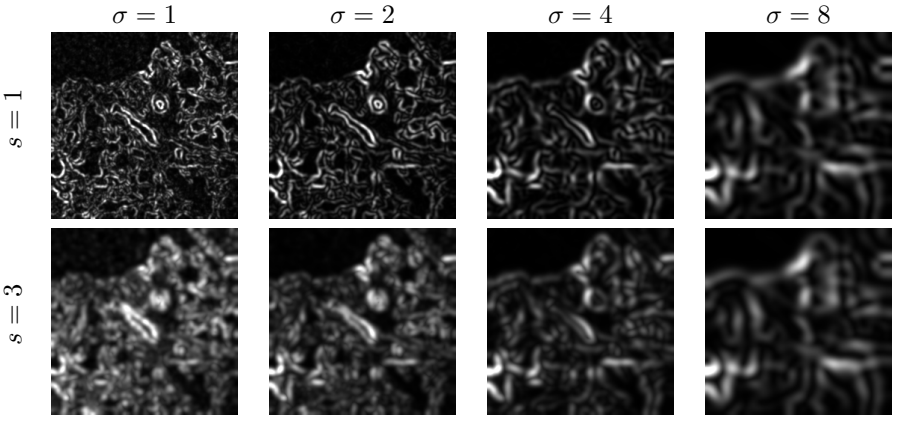


FIGURE C.32: Largest eigenvalue of the structure filter applied to the image in Figure C.1

C.19 Entropy

The entropy filter highlights structures of an image. First a circle of radius r is drawn around each pixel and the histogram of the pixel inside that circle is calculated with 'nBins' chunks. Then, the entropy is calculated as

$$\sum_{p \text{ in histogram}} -p * \log_2(p),$$

where p is the probability of each chunk in the histogram. 'nBins' is varied from 32 to 256 in powers of 2. r is varied from $\sigma_{\min}$ to $\sigma_{\max}$. The result can be seen in Figure C.33.

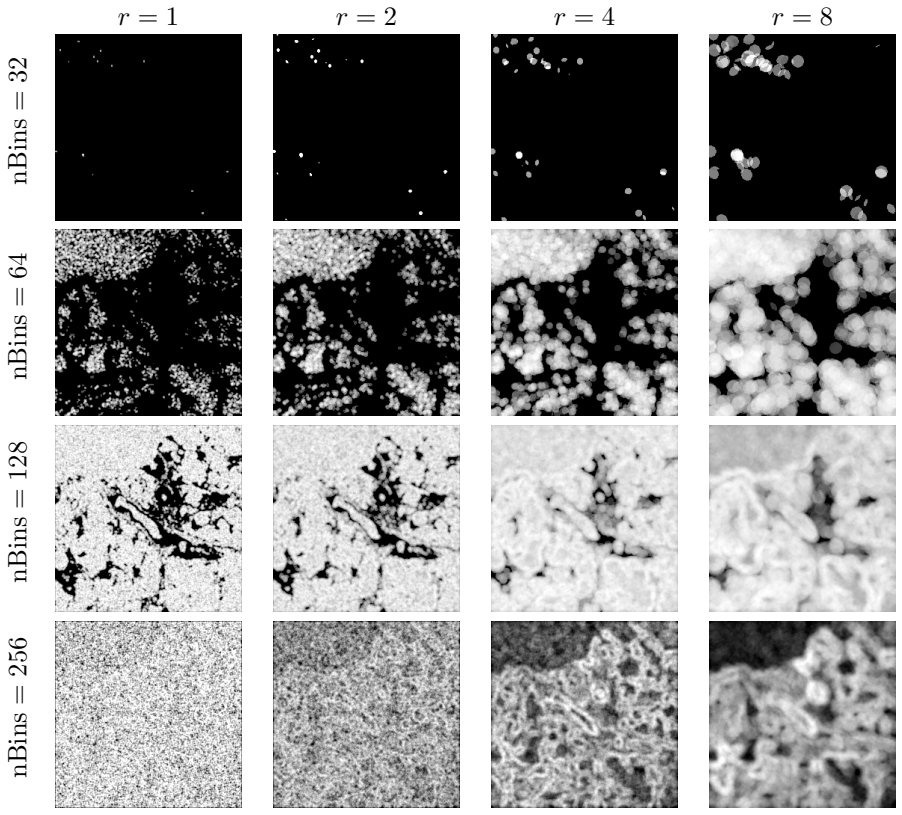


FIGURE C.33: Entropy filter applied to the image in Figure C.1

C.20 Neighbors

The neighbors filter shifts the image in 8 directions by a certain number of pixels, σ. It therefore creates $8n$ feature images. The change in the image is barely visible to the human eye at this size. We still included all variations for completeness in Figures C.34, C.35, C.36 and C.37.

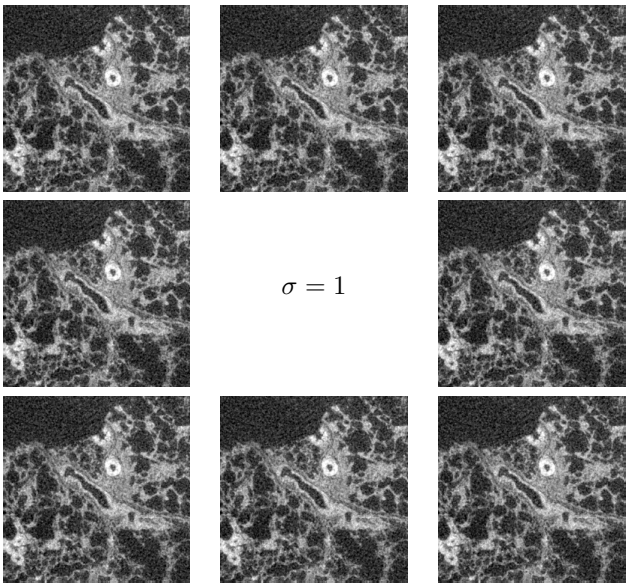


FIGURE C.34: Neighbors filter with $\sigma = 1$ applied to the image in Figure C.1

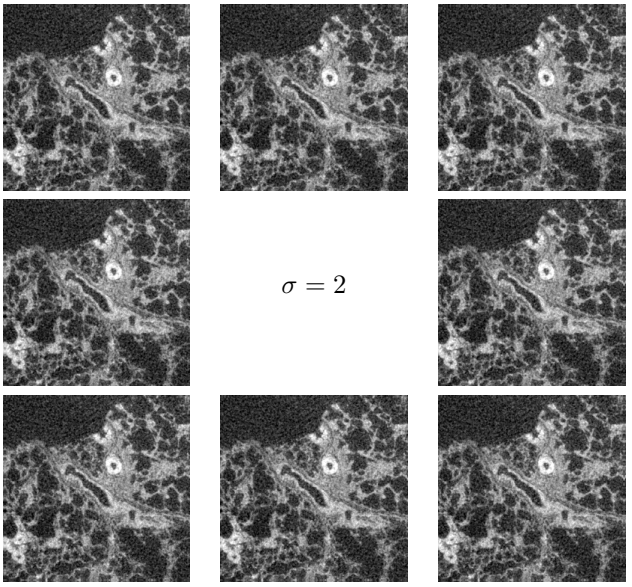


FIGURE C.35: Neighbors filter with $\sigma = 2$ applied to the image in Figure C.1

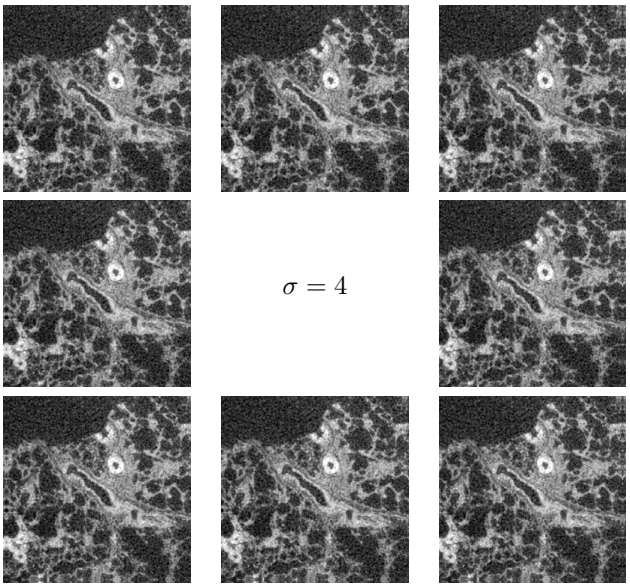


FIGURE C.36: Neighbors filter with $\sigma = 4$ applied to the image in Figure C.1

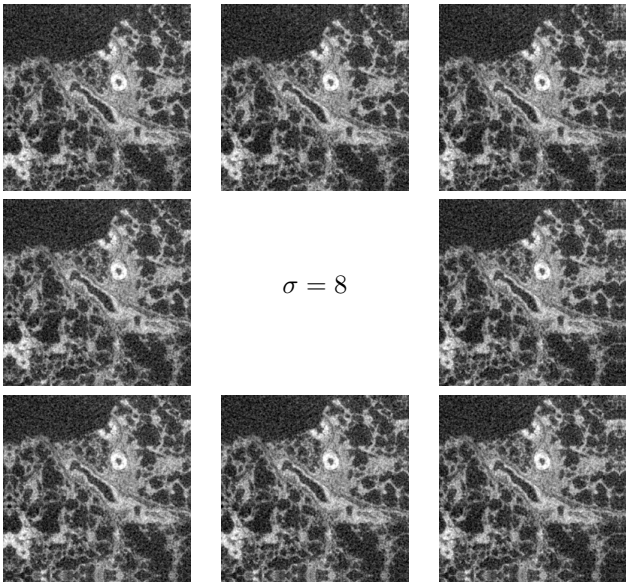


FIGURE C.37: Neighbors filter with $\sigma = 8$ applied to the image in Figure C.1

C.21 Summary

All filter options are compared in Table C.1. The comparison includes the type of filter according to Section 2.2.6 and the amount of features produced (relates to memory consumption). The TWS plug-in allows to vary the membrane thickness and patchsize, as well as minimum and maximum sigma. The number of produced features depends on the difference between the minimum and the maximum sigma. This difference shall be denoted n, where $n = b - a$ with $\sigma_{\min} = 2^a$ and $\sigma_{\max} = 2^{b-1}$.

TABLE C.1: Summary of the different filters provided by the TWS

Name	**Section**	**Type**	**Feature images produced**
Gaussian blur	Appendix C.1	Spatial filter	n
Sobel filter	Appendix C.2	Spatial filter	$n + 1$
Hessian	Appendix C.3	Spatial filter	$8n$
Difference of Gaussian	Appendix C.4	Spatial filter	$0.5n(n - 1)$
Membrane projections	Appendix C.5	Spatial filter	6
Variance	Appendix C.6	Spatial filter	n
Mean	Appendix C.7	Spatial filter	n
Minimum	Appendix C.8	Spatial filter	n
Maximum	Appendix C.9	Spatial filter	n
Median	Appendix C.10	Spatial filter	n
Anisotropic diffusion	Appendix C.11	Iterative algorithm	$2n$
Bilateral	Appendix C.12	Spatial filter	4
Lipschitz	Appendix C.13	Frequency filter	5
Kuwahara	Appendix C.14	Spatial filter	3
Gabor	Appendix C.15	Frequency filter	22
Derivatives	Appendix C.16	Spatial filter	$4n$
Laplacian	Appendix C.17	Spatial filter	n
Structure	Appendix C.18	Spatial filter	$4n$
Entropy	Appendix C.19	Spatial filter	$4n$
Neighbors	Appendix C.20	Spatial filter	$8n$

Appendix D

Publications

The work has been published at several occasions and the published papers are presented in this appendix as follows:

- Wollatz, L., Scott, M., Johnston, S. J., Lackie, P. M., Cox, S. J., Oct. 2018. 'Curation of image data for medical research'. *In: Proc. 14th Int. Conf. eScience (eScience 2018)*. IEEE, Amsterdam, Netherlands, pp. 105–113. doi:10.1109/eScience.2018.00026
- Wollatz, L., Johnston, S. J., Lackie, P. M., Cox, S. J., Dec. 2017. '3D histopathology—A lung tissue segmentation workflow for microfocus X-ray-computed tomography scans'. *J. Digit. Imaging* 30 (6), 772–781. doi:10.1007/s10278-017-9966-5
- Wollatz, L., Konieczny, K., Vandervelde, C., Mitchell, T., Cox, S., Burgess, A., Ismail-Koch, H., Sep. 2016. 'The use of 3D-printed paediatric temporal bones as a training tool'. *In: BAPO Annu. Sci. Meet. (BAPO 2016)*. British Association for Paediatric Otolaryngology, Liverpool, United Kingdom, p. 24
- Wollatz, L., Cox, S. J., Johnston, S. J., Sep. 2015. 'Web-based manipulation of multiresolution micro-CT images'. *In: Proc. 11th Int. Conf. eScience (eScience 2015)*. IEEE, Munich, Germany, pp. 308–311. doi:10.1109/eScience.2015.42

Curation of Image Data for Medical Research

Lasse Wollatz*, Mark Scott*, Steven J. Johnston*, Peter M. Lackie†, and Simon J. Cox*
*Faculty of Engineering and Physical Sciences, University of Southampton, Southampton, UK
†Faculty of Medicine, University of Southampton, Southampton, UK
{L.Wollatz, Mark.Scott, S.J.Johnston, P.M.Lackie, S.J.Cox}@soton.ac.uk

***Abstract*—Microfocus X-ray computed tomography (µCT) and 3D microscopy scanning create scientific data in the form images. These images are each several tens of gigabytes in size. E-Scientists in medicine require a user-friendly way of storing the data and related metadata and accessing it. Existing management systems allow computer scientists to create automatic image workflows through the use of application programming interfaces (APIs) but do not offer an easy alternative for users less familiar with programming. We present a new approach to the management and curation of biomedical image data and related metadata. Our system, Mata, uses a network file share to give users direct access to their data and also provides access to metadata. Mata also enables a variety of visualization options as required by e-Scientists in medicine.**

***Index Terms*—data curation, image management, metadata, medical research**

I. Introduction

Data curation is an essential part of medical research. New imaging techniques and the fast-changing nature of research lead to quick changes in the image analysis tools required by e-Scientists in medicine.

Current systems, such as the Open Microscopy Environment Remote Object (OMERO) platform [1] and the Bio-Image Semantic Query User Environment (BisQue) [2], rely on custom plug-ins and internal software to allow users to interact with the data stored by them. Additional steps are needed for the integration of new software into the image management workflow. Either the users export data to their local file store, to then import the data into the new software, or a plug-in is programmed to allow the software to retrieve data through the systems' application programming interfaces (APIs).

Many researchers and technicians are working to ensure that e-Scientists in medicine can conduct research using the latest advances in computer science. The medical research itself has to be undertaken by medical specialists. It is vital that e-Scientists in medicine do not have to consult a computer scientist to test and experiment with the vast number of software tools and algorithms included in those tools. We consider easy access to data to be more critical than programmatic access in the form of an API [3].

Heterogeneous Data Centre (HDC) [4] is a tool that manages users' data without imposing such restrictions. A network share enables users to access their data directly through their operating system's (OS) file manager application. A file-system monitor keeps track of the data and synchronizes it with the metadata in a database.

In this paper, we adapt HDC for the use of biomedical research images. The resulting software is called Mata[1] (a combination of medicine and data) and provides additional functionalities for working with metadata as well as previewing capabilities targeted to e-Scientists in medicine.

This paper makes the following contributions:

- It proposes an alternative approach to the curation of research data in medicine;
- It shows how HDC can be configured to medical research and the steps taken to create Mata;
- It designs distinct use cases to show how Mata can benefit medical researchers.

The rest of this paper is structured as follows. An overview of HDC and reasons for choosing it are given in section II. Section III discusses Mata and how it adapts HDC to medical research. Section IV presents different use cases and shows how Mata can benefit medical researchers. We give a conclusion in section V.

II. Motivation

In [5], we tested a website-based system for image management. The upload of several tens of gigabytes of data through a web browser is slow and unreliable due to browser limitations [6], [7]. The upload speed is mainly dependent on the internet performance, which cannot be controlled by the system. There are two common solutions to implement the upload of large files. Either a webbrowser application is used, or a separate software connects to both the local file store and the remote one. We used Plupload[2] in [5] to overcome the issue of file-size when uploading large datasets, but it was not able to increase reliability compared to that offered by software-based file upload as used by other management systems [1], [2], [8]. The alternative approach, as used by [1], [2], [8], is a software running on the e-Scientists computer that interacts with an API on the management server. The downside of a system-specific software for image upload and download is that it reduces user-friendliness since it requires users to learn the operation of the image management software. Providing access to data only via an API introduces an additional step to connect other software. Any new software will require a computer scientist to integrate

Funding: This work was supported by the Engineering and Physical Sciences Research Council [grant number 1511465]

[1]Published as: doi:10.5258/SOTON/D0430
[2]Official site: http://www.plupload.com/ (last accessed 22/05/2018)

it with the management system via an API. The APIs used by existing management systems are not standardized. The existing clinical standard for data management and transfer, Digital Imaging and Communications in Medicine (DICOM), relies on detailed patient information and cannot be used in a research environment, where the patient needs to be anonymous [9], [10].

The reason for choosing HDC [4] was that it gives users direct access to the server file store. Using a network file share gives users a familiar environment for accessing their data while ensuring robust file transfer and storage methods that evolve with the file store.

HDC has a file-system monitor, which will be referred to as the "file watcher" for the rest of this paper. Its task is to compare the state of the file system to that of the database storing the metadata and modify each of them to reach a consistent state. The file watcher is triggered by file-system events to analyze a particular folder and checks every folder in the system at regular intervals. All folders on a defined level are considered datasets in the database. Information about the datasets' files is stored in hidden sub-directories of the data folder. With the help of this information, HDC can determine and differentiate between additions, removals, renames and copies on a folder and file level [4], [11]. Another functionality provided by the file watcher is the automatic execution of plug-ins if specific file-types are added. This allows extending the capabilities of the file watcher further by interfacing with other external software [4].

III. The Image Management System

In this section, we will explain how we created Mata by modifying HDC. HDC implements a variety of features for the management of data, most of which were deemed useful for medical research. For Mata, we retained the Windows server to integrate with companies' Windows networks, as well as the Structured Query Language (SQL) database used by HDC for metadata storage. The layout of the database was copied since it already implemented the versatility required for this project.

Other parts of HDC were considered unnecessary for this particular implementation. Specifically, the Microsoft SharePoint[3] interface was removed and replaced with a more lightweight PHP (PHP: Hypertext Preprocessor) website. SharePoint integrates well with Microsoft environments since it is part of the Microsoft Office suite. The number of third-party applications for SharePoint is limited and integration of custom plug-ins, though possible, requires programming them in .NET or finding another way of implementing them. The original interface was built with SharePoint 2010 and .NET 3.5. That version of SharePoint is out of date, and we found it to be slow. We did not use many of the SharePoint features that were implemented with the HDC front-end, either. Instead of reimplementing the SharePoint front-end in a more recent release, we created a PHP site for Mata that was easier to prototype and is fully open source. The use of PHP also allowed easy reimplementation of essential features, such as the editing of user permissions and dataset metadata.

[3]Official site: www.sharepoint.com/ (last accessed: 22/05/2018)

The rest of this section will address various parts of the management system and the corresponding solutions we have developed. We will briefly discuss feature re-implementations of HDC and then introduce any additions made to the system for each part. We distinguish between the following modules of the management system: editing of metadata, visualization and searching of metadata, visualization of datasets, and system security.

A. Editing of Metadata

Metadata here refers to key-value pairs [4], which we refer to as tags in order to match users' expectations. Tags allow users to assign metadata to datasets. Tags can help maintain information about data that is important to medical researchers but cannot be stored as part of the image file. For example, a tag can describe the species seen in the image, which may be a human, a sheep, or a zebrafish. As with HDC, editors of the dataset are allowed to sort tags in a particular order and create hierarchies for them. These hierarchies allow users to structure the tags according to their needs.

A challenge faced when allowing users to edit the metadata is the lack of consistency, which makes a comparison of tags more challenging [12]. We used two solutions to overcome this problem.

One solution we adapted from HDC to avoid users using different terms to describe the same thing involved the ability to import tags from parents. It assumes that users will tend to use this functionality to save the effort of copying common tags. When importing tags from a parent, all tags from that parent, where the key has not yet been assigned any value for the dataset, are copied over to the dataset. This approach ensures consistency among tags between different datasets, users, and sites. Not copying values of keys which have already been defined avoids overwriting data or having duplicate keys.

In addition, Mata implements a dictionary to guide users into using the same tags for describing the same features [13]. The advantage is that synonyms can be avoided. In this project, a dictionary has been implemented for guidance to avoid typographical errors and reduce the number of grammatical variations of the same term. The system suggests existing tags to users based on the user's input. Tags that have been used more often have been ranked higher amongst the suggested tags.

B. Visualization and Searching of Metadata

One of the main requirements of a data management system is the ability to look up data in an efficient manner. Our solution provides three different options for finding datasets: a basic search, tag clouds, and relation networks.

1) Basic Search: One functionality re-implemented is a basic search function which looks for the exact phrase in the datasets title, description and tags. It returns any matches found in the order of the number of matches of the search term

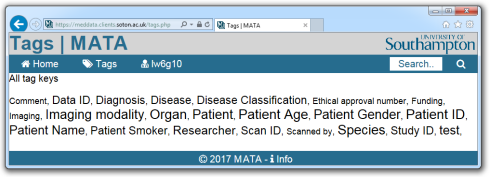


Fig. 1. Screenshot showing the rectangular tag cloud for a small number of tags. Entries are in alphabetical order. More common tags have a larger font-size than less common tags.

found in the database. This search function can be expanded to be more sophisticated if needed [14]–[17].

2) Tag Cloud: A page with all of the different tags was created, to enable users to browse the tags. Tags were arranged in an elementary rectangular tag cloud as shown in Fig. 1, to aid users in finding important tags quicker. The implementation of more sophisticated tag clouds is possible and has been discussed in many papers [18]–[20]. We implemented our own tag cloud here and will outline it in the following paragraphs.

In order to prioritize tags in a tag cloud, they are represented in different font sizes, colours, or layouts. Tags that appear more often are larger, such that specific, unusual tags or typographical errors that only appear for a few datasets are smaller and less noticeable.

We assume that more frequently used tags are more likely to be of interest to a common user and therefore should be larger and easier to find. Meanwhile, to help users searching for specific tags, we minimized the variation of font size between tags. We used Zipf's Law [21] to map the occurrence of specific tags to a linear distribution. We also maintained an alphabetical ordering of tags to help users find specific tags [19].

3) Relation Network: When editing a dataset, users can define parent datasets, meaning datasets which the current dataset was derived from [4]. Examples are a segmentation of a scan or follow-up data, which can be linked to the original image dataset. By default, a dataset is expected to have precisely one parent unless it is an original scan, in which case it does not have any parent. In some cases, several images may be combined into one. An example is the correlation of a single photon emission computed tomography (SPECT) image and a computed tomography (CT) image. Areas of abnormality found by a SPECT can be located relative to internal organs, detected by the CT [22]. In such cases, more than one parent may exist. Having the origin of an image defined ensures that they can be traced back. A network graph (see Fig. 2) is generated, using recursion to search for all related datasets in the database and vis.js[4] for display. In this graph, each node represents a dataset and arrows connecting the nodes indicate their relation, as seen in Fig. 2. This graph helps users finding other datasets derived from the same scan with just a few clicks.

This view has also been combined with the tags explained in the previous section. As a result, it can display a hierarchy of tags for a folksonomy [12], [23] by assuming that two tags are equal, if their keys, their values, and child tags are all the same. The order of the children is disregarded.

As seen in Fig. 2, the datasets "*Sample_CT002*" and "*test-Merge of twins CT*" are tagged with the same patient, as all the child tags related to "*Patient*" are the same. However, they are different from the patient of "*Sample_CT001*", as they only share a single attribute.

Fig. 2 also illustrates the limitation of this strict definition of equality, since the "*Patient*" in "*Sample_CT001*" and "*Sample_CT001-Copy*" are considered different even though they describe the same person. Not all child tags of "*Patient*" are the same, because the researcher in one dataset listed the species as part of the "*Patient*" information. The gravitational physics model behind vis.js solves this issue by ensuring that such tags, which are almost equal, will remain close together since they share many children.

The rather strict definition avoids false positives. In medicine, critical metadata, such as the patient, have multiple identifiers. This reduces the chance of tags falsely being identified as equal.

The comparison of tags and their children, as mentioned before, occurs over several levels. As a result, the tag "*Imaging*" may include subcategories, for example, "*Scan settings*", containing a list of settings such as the exposure rate and projections. In the web interface, every tag (for example a single patient) links to a page that shows all datasets containing that tag.

C. Visualization of Datasets

Microfocus X-ray computed tomography (µCT) data and resulting processed data need to be available to medical researchers in a secure, accessible, and meaningful way. One of the critical demands identified was fast previewing of images. It allows users to decide which images are worth analyzing before retrieving the full image from a remote storage onto their systems for processing. In order to display files over the web, we used the Multiresolution Computed Tomography Viewer (MCTV) [5] and Viewstl and integrated them as shown in Fig. 3.

1) Adding a 2D Volumetric Image Viewer: MCTV allows the viewing of extra-large 3D CT files in a web browser without requiring plug-ins [5]. It can also present 2D images as a special case of a 3D image stack. On the presentation layer, it can be directly embedded into Mata without additional installations. On the back end, a script had to be implemented to automatically create previews and metadata required by MCTV. The HDC file watcher already had a functionality to allow a script to be executed when a dataset changes. This was used by Mata to trigger a script creating a preview for a dataset each time it changes, as shown in Fig. 3.

[4]Accessible at http://visjs.org (last accessed 22/05/2018)

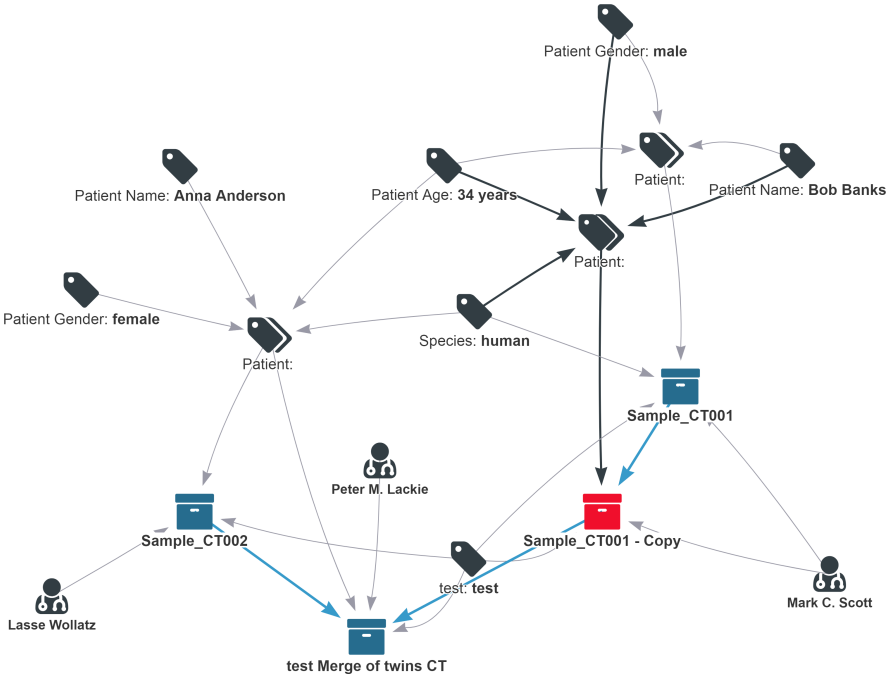


Fig. 2. Screenshot of a dataset relation network graph using fictional datasets for demonstration purposes. It allows to find all datasets (blue) related to the currently selected dataset (red). We invented patient names and details to make it easier to follow the example.

The process of creating tiles requires a large number of read and write operations and is therefore relatively slow. This means that it is likely that a dataset is being changed while a preview of the old version is still being created. For example, when a dataset comprising a stack of 2D image files is first uploaded, the file watcher triggers the plug-in for each file upload. When the first file is being uploaded, the file watcher starts the tile-creating script (the tiler). By the time the second file reaches the Mata file store, the tiler has not yet finished, but the file watcher starts another instance of it. For a 3D image with 2000 slices, this would mean that millions of duplicate tiles would be created.

Therefore the script needed to cope with processing large images. The approach we took was to split the task of creating tiles into several subtasks which can then be canceled remotely. To implement this, a local queue was set up to enable local deployment, as shown in Fig. 4. Celery version 3.1.25 was used for the queue as it is the latest version compatible with Windows. Scripts interacting with the queue were implemented in Python.

HDC executes a script that sends a job request to "readdir" to the queue. The "readdir" function first checks if the command has been executed for the same directory before and revoke any outstanding jobs. As this version of Celery does not support accessing information about tasks in the queue, it was necessary to save the job identifications (IDs) in a separate file. The Python code will then read any metadata files it finds and attempts extracting remaining information such as the global minimum and maximum pixel values from the image files. For extracting the minimum and maximum pixel values, only a selection of the image files is read (or a random set of points taken from a raw file) and evaluated to reduce the overall time it takes to execute the script. Finally, one tiling-job request is created for each image slice, and the job ID is written to a file. The tiler then splits the image into smaller tiles and saves the tiles to disk [5]. The tiler jobs are started with a lower priority than that of the job reading the directory, to ensure that revoking jobs gets priority over working on other jobs. This reduces the overall queue length and avoids images being tiled more often than necessary.

It is important to make images accessible in a way meaningful to the medical researchers who are used to 2D images. With the script for creating the tiles in place and having it linked as shown in Figs. 3 and 4, MCTV is able to read the

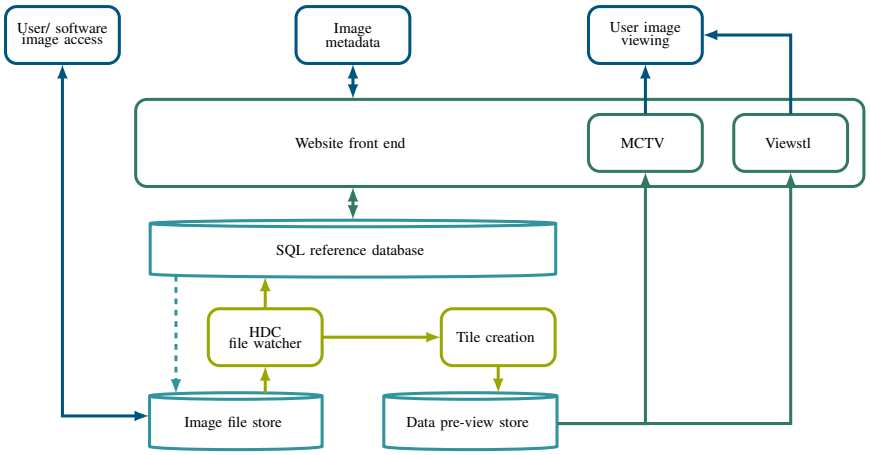


Fig. 3. The new proposed system using HDC. The image file store, HDC file watcher and SQL reference database are taken from the original HDC implementation. The website front-end, the implementation of visualization tools and related data pre-processing scripts are newly created for this paper.

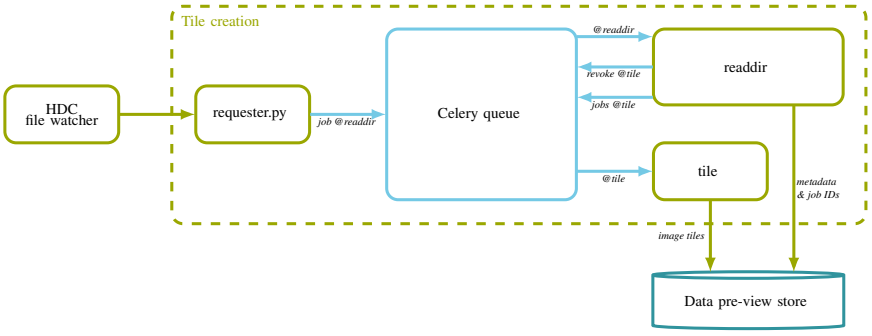


Fig. 4. Schematic of the queue implemented for the tiler in Fig. 3

tiles from the server file store and present them to the user as shown in Fig. 5. Users' access to the tiles is restricted through the permissions on the network file share.

2) Adding a 3D Surface Image Viewer: The viewer presented in the previous section enables the viewing of volumetric images in the form of 3D image stacks and 2D images. It cannot deal with 3D viewing or with the viewing of surface files. The data which are inherently 3D also requires a 3D visualization in order to enable the analysis of 3D structures not represented in a 2D viewer. It was therefore decided to add Viewstl[5] to display stereo-lithography (STL) and object (OBJ) files.

As with MCTV, Viewstl does not require special plug-ins but works with JavaScript. Files are loaded into client-side memory and are not processed by any third party.

In order to use the encrypted version of the Hypertext Transfer Protocol (HTTP) for all resources, a local copy of Viewstl was created and modified to use the encrypted protocol.

To integrate the viewer, an HDC file-watcher plug-in has been created, which is executed for STL and OBJ files. It calls a Python script which uses the same queue as the MCTV tiler but only creates a file with the name of the first STL file in

[5]Official site: http://www.viewstl.com/ (last accessed 22/05/2018)

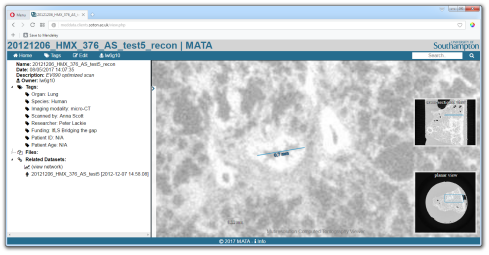


Fig. 5. Screenshot of a dataset containing a TIF-stack viewed in Mata. The panel on the left shows the tags and related datasets. On the right a slice of a µCT scan of a lung biopsy can be seen with a planar and cross-sectional view as explained in [5].

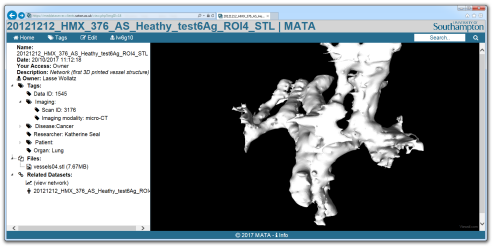


Fig. 6. Screenshot of a dataset containing an STL file, viewed in Mata. The layout is the same as in Fig. 5 but MCTV is replaced with Viewstl showing the surface of a vascular structure from a lung biopsy.

the datasets folder. On the web page for viewing a dataset, PHP is used to check if the dataset contains a 3D surface file and if the user has read permission. If this is the case, Viewstl is loaded and is passed the file name, which is then displayed as shown in Fig. 6.

Through the use of Viewstl, the ability to view 3D surface files over the web via Mata was demonstrated. The secure implementation of an external viewer also shows that, due to its modularity, Mata can be extended very easily.

D. System Security

The importance of security for medical databases has been highlighted through recent incidents [24], [25]. For the security of a service, various functional components have to be considered and evaluated. This is particularly important when dealing with sensitive medical data. Components that could be vulnerable include the user authentication and the database. Most of these components have been discussed in [4], [11]. The specific permissions in Mata were kept the same as in the case of HDC. Everyone can list folder content but access to file content is limited to certain users. The owner (the creator) of the dataset can decide whom to give access to, and people with access can preview and edit the data apart from their permission in a browser. HDC also forwards the permissions as read and write permissions to the server file store and the connected network file share.The file access on the network file share is secured through Windows Authentication. Applications, such as the PHP interpreter, running on the server responsible for the website need elevated permissions in order to access the database and other data. Further, the website authentication has to be re-implemented with PHP.

When webpage content is generated, it is essential that confidentiality of the data is maintained. Accordingly, when rendering the content of a webpage, PHP needs to know who is allowed to view or edit individual content and who is accessing the website. HDC synchronizes folder and file permissions between the database and network file share. As PHP has access to the database, it can look up if a user has permission to access specific data. Additionally, Windows Server and Microsoft Internet Information Service (IIS) enable the use of the same authentication for websites and file access. This enables single sign-on (SSO), so that users only need one account and password to access both, the network file share and the website of Mata as well as other systems connected to the domain. PHP can read the session's details from the server and therefore knows who is accessing the webpage. The user identification is the same as the one HDC synchronized between the servers' file store and database. As a result, PHP knows the users' permissions.

The primary reasons for choosing SQL as the main database type was the well-established security it offers in comparison to not only SQL (NoSQL) databases [26]–[28]. The use of well-established systems, like Windows Server, Windows authentication, SQL and PHP enables system administrators to set up a secure implementation throughout the whole data management system.

E. Summary

In this section, we introduced the medical image management system Mata. We discussed the underlying software and presented various parts of the management system. This included the editing of metadata, visualization and searching of metadata, visualization of datasets and system security. Metadata is created by users and the editing of it is guided through the use of a dictionary. A tag cloud and dataset relation networks allow additional navigation options apart from a basic search. We also showed the implementation of a 2D and a 3D viewer, and explained how we secured the system. A comparison of HDC and Mata is given in Table I.

IV. Use Cases

Though it is applicable to any sort of data, Mata was tested using a scenario modelled after a research project on lung diseases, at the University of Southampton. This project evaluates the use of µCT scanners for the analysis of biopsies in a non-destructive way.

This section presents the use cases for this scenario to show how the tools discussed can integrate with an existing image workflow. The use cases are illustrated in Fig. 7.

TABLE I
COMPARISON OF HDC AND MATA.

	HDC	Mata
Operating system	Windows Server	Windows Server
Server	Microsoft IIS	Microsoft IIS
Database	SQL Server	SQL Server
Middleware	Sharepoint 2010/ .NET 3.5	PHP 7.1
Data access	Network file share	Network file share
Synchronisation	File watcher	File watcher (from HDC)
SSO authentication	✓	✓ (partially reimplemented)
Metadata editing	✓	✓ (reimplemented)
Metadata import	✓	✓ (reimplemented)
Metadata suggestion	✗	✓ (ranked dictionary)
Basic search	✓	✓ (reimplemented)
Tag cloud	✗	✓
List of direct relatives	✓	✓
Relation network	✗	✓ (graph of all relatives)
CT slice viewer	✗ (but 2D viewer)	✓ (MCTV)
3D surface viewer	✗	✓ (Viewstl)

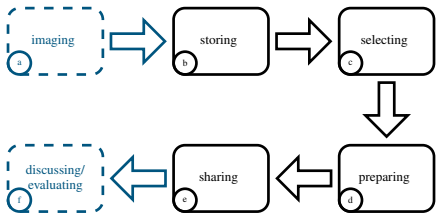


Fig. 7. Simplified overview of the use cases. Dashed parts are important to the project and the image workflow but not part of the presented system. Each of the steps (a) to (f) are described in more detail in sections IV-A-IV-E.

As part of this scenario, a large number of images needs to be created using a newly developed µCT scanner. Matching microscopy images are retrieved, and images are processed to enhance the contrast. µCT and microscopy images are then aligned in 2D and 3D using point-based registration based on the scan conditions. Matching image areas are presented to experts to assess both biological content and image quality. Images of microscopy slices and µCT slices are then assessed for technical quality or biological data content by image experts or histopathologists.

A. Storing New Scans

Use case: **An image has been scanned on the µCT, and the scan operator wants to store the image. (Fig. 7 a, b)**
By using Mata developed in section III, it is possible to store the images generated by the newly developed medical µCT scanner by simply copying the image files into the network file share.

The file watcher detects the new files and creates an entry in the database. It also triggers the code for creating a preview of the image. The operator can add file permissions and metadata as appropriate. In many practical scenarios, it will be beneficial to add an HDC plug-in which reads out metadata from files and adds it automatically as described in [4].

If metadata are created in a consistent way by the scanner, a plug-in for the file watcher can be written, to automatically add this metadata to the database as soon as the image gets uploaded. DICOM devices can also be supported by adding a DICOM receiver. An administrator is required to set permissions for any received data since DICOM does not transfer user information during file exchange. This is due to DICOM being patient centred.

B. Finding Relevant Images

Use case: **A medical researcher tries to find and get hold of images relevant to the project. For that, they want to find images of a lung taken by µCT and preview them to confirm that they align with the projects requirements. (Fig. 7 c)**
The researcher has various options of finding relevant images, as discussed in section III-B. A basic search for a relevant keyword or browsing through the tags can help obtaining a list of relevant datasets. If images are not in the desired format or processing stage, it is possible to find related datasets through the network graph. With the help of MCTV and stlviewer (section III-C), it is possible to preview images before downloading them, even if their size of several tens of GB exceeds the machine's Random Access Memory (RAM). Download of images can be conveniently achieved through the file explorer.

C. Finding Related Images

Use case: **The researcher wants to retrieve the microscopy scan of the same sample. (Fig. 7 c)**
The researcher has various options to find a related dataset. The two images are likely to share a tag, like the sample identification number. The researcher can therefore find the data by searching for this tag. A more direct approach is to look at the relationship network graph. Any linked dataset can be found and accessed in this way.

D. Processing Images

Use case: **The researcher found a relevant image and tries to process it for presentation to the experts. (Fig. 7 d)**
After relevant images have been found, they can be processed using any suitable processing software.

Since the images are accessible via network file share, any processing software the researcher wants to use can read the files directly from the remote storage.

After the processing, the new data are stored in a new folder on the network file share to create a new dataset in Mata and share them with other, specified researchers. Newly found metadata can also be added to Mata, and a custom HDC plug-in for importing this metadata can be developed if suitable.

E. Sharing Images

Use case: **A set of images is to be shared with an expert for evaluation. (Fig. 7 e, f)**

In order to share a dataset with an expert, the web link to that dataset can be sent to the other person. Provided that the other person is a user of Mata, they can access the dataset as soon as the owner has added them to the list of viewers/editors of the dataset through the web interface.

F. Summary

In this section, we introduced a possible scenario for the use of an image management system in medical research. We showed how Mata can be used as a management system for this scenario by discussing different use cases based on the scenario. We demonstrated that Mata can tackle the various steps involved for image management in medical research. All of the steps can be achieved without programming scripts or plug-ins to link external software to the system.

V. Conclusion

It is essential to provide access in an user-friendly way, to enable e-Scientists in medicine to access data without requiring programming skills. Current image management systems are restrictive in the way they incorporate other software.

We took a data curation system that gives users direct access to their data through a network file share, eliminating the need for special software to access the data. It also simplifies the use of new software for image processing and visualisation with the existing system, without requiring custom plug-ins to load image data via a non-standard API. Since most software can read files from a file store, no additional programming is required to allow them to use image data stored in our management system.

We added a website front-end to the system, which allows medical researchers to fulfill common use cases. This includes the management of metadata through the web-interface as well as different ways of visualizing and searching metadata and datasets. We showed how additional data visualization tools can be implemented. We used well established standards all along the way to ensure that the system is easy to transfer to and secure in a different environment.

Future work involves the addition of an API to allow scripts to access metadata through code. It would be beneficial for the biomedical image community to develop a standard for research images similar to DICOM for clinical, medical images. Declaring an existing API, like the one used by OMERO or BisQue, as a standard may well be an option for this. We will also show how this management system fits into the bigger workflow of images in medical research.

Acknowledgment

We would like to thank Anna Scott for providing her dataset as an example figure for this paper. All data supporting this study are openly available from the University of Southampton repository at https://doi.org/10.5258/SOTON/D0430.

References

[1] C. Allan, J. M. Burel, J. Moore, C. Blackburn, M. Linkert *et al.*, "OMERO: flexible, model-driven data management for experimental biology," *Nat. Methods*, vol. 9, pp. 245–253, 2012.

[2] K. Kvilekval, D. Fedorov, B. Obara, A. Singh, and B. S. Manjunath, "BisQue: a platform for bioimage analysis and management," *Bioinformatics*, vol. 26, no. 4, pp. 544–552, Feb 2010.

[3] J. Dryndos, A. S. S. Kazi, D. Langenberg, H. Löh, and R. Stark, "Collaborative virtual engineering for smes: Technical architecture," in *IEEE Int. Technol. Manag. Conf.* Lisbon, Portugal: IEEE, Jun 2008, pp. 1–8.

[4] M. Scott, R. P. Boardman, P. A. Reed, and S. J. Cox, "Managing heterogeneous datasets," *Inf. Syst.*, vol. 44, pp. 34–53, 2014.

[5] L. Wollatz, S. J. Cox, and S. J. Johnston, "Web-based manipulation of multiresolution micro-CT images," in *Proc. 11th Int. Conf. eScience*. Munich, Bavaria, Germany: IEEE, Sep 2015, pp. 308–311.

[6] E. Lawrence, "File upload and download limits," Mar 2011, accessed on 06/12/2017. [Online]. Available: https://blogs.msdn.microsoft.com/ieinternals/2011/03/10/file-upload-and-download-limits/

[7] Ephox, "Plupload: Multi-runtime file-uploader frequently asked questions," 2017, accessed on 06/12/2017. [Online]. Available: http://www.plupload.com/docs/v2/Frequently-Asked-Questions#when-to-use-chunking-and-when-not

[8] D. S. Marcus, T. R. Olsen, M. Ramaratnam, and R. L. Buckner, "The extensible neuroimaging archive toolkit," *Neuroinformatics*, vol. 5, no. 1, pp. 11–33, Mar 2007.

[9] NEMA, *Digital Imaging and Communications in Medicine (DICOM) Standard*, National Electrical Manufacturers Association Standard 2018a, 2018. [Online]. Available: ftp://medical.nema.org/medical/dicom/2018a/

[10] M. J. Warnock, C. Toland, D. Evans, B. Wallace, and P. Nagy, "Benefits of using the dcm4che DICOM archive," *J. Digit. Imaging*, vol. 20, no. Suppl. 1, pp. 125–129, Nov 2007.

[11] M. Scott, "Research data management," Thesis, University of Southampton, May 2014.

[12] S. A. Golder and B. A. Huberman, "Usage patterns of collaborative tagging systems," *J. Inf. Sci.*, vol. 32, no. 2, pp. 198–208, 2006.

[13] A. Noruzi, "Folksonomies - why do we need controlled vocabulary?" *Webology*, vol. 4, no. 2, Jun 2007.

[14] S. Dessloch and N. Mattos, "Integrating SQL databases with content-specific search engines," in *Proc. 23rd VLDB Conf.*, Athens, Greece, 1997, pp. 528–536.

[15] W. Kießling and G. Köstler, "Preference SQL: design, implementation, experiences," in *Proc. 28th Int. Conf. Very Large Data Bases (VLDB '02)*. Hong Kong, China: VLDB Endowment, 2002, pp. 990–1001.

[16] Y. Tsuruoka, J. Tsujii, and S. Ananiadou, "FACTA: a text search engine for finding associated biomedical concepts," *Bioinformatics*, vol. 24, no. 21, pp. 2559–2560, Nov 2008.

[17] S. Brin and L. Page, "Reprint of: The anatomy of a large-scale hypertextual web search engine," *Comput. Networks*, vol. 56, no. 18, pp. 3825–3833, Dec 2012.

[18] B. Y.-L. Kuo, T. Hentrich, B. M. Good, and M. D. Wilkinson, "Tag clouds for summarizing web search results," in *Proc. 16th Int. Conf. World Wide Web (WWW '07)*. New York, New York, USA: Association for Computing Machinery (ACM), 2007, pp. 1203–1204.

[19] M. J. Halvey and M. T. Keane, "An assessment of tag presentation techniques," in *Proc. 16th Int. Conf. World Wide Web (WWW '07)*. New York, New York, USA: Association for Computing Machinery (ACM), 2007, pp. 1313–1314.

[20] C. Seifert, B. Kump, W. Kienreich, G. Granitzer, and M. Granitzer, "On the beauty and usability of tag clouds," in *Proc. 12th Int. Conf. Inf. Vis. (IV 2008)*. London, UK: IEEE, Jul 2008, pp. 17–25.

[21] G. K. Zipf, *Human Behaviour and the Principle of Least-Effort*. Cambridge, MA: Addison-Wesley, 1949.

[22] G. Soo and T. Cain, "SPECT-CT scan," Jul 2017, accessed on 25/05/2018. [Online]. Available: https://www.insideradiology.com.au/spect-ct-scan/

[23] A. Mathes, "Folksonomies - cooperative classification and communication through shared metadata," 2004, accessed on 25/05/2018. [Online]. Available: http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html

[24] TrapX Research Labs, "MEDJACK.2 hospitals under siege," TrapX Research Labs, Tech. Rep., 2016, accessed on 25/05/2018. [Online]. Available: https://media.scmagazine.com/documents/242/trapx_medjack2_60312.pdf

[25] T. Fox-Brewster, "Medical devices hit by ransomware for the first time in US hospitals," May 2017, accessed on 04/07/2017. [Online]. Available: https://www.forbes.com/sites/thomasbrewster/2017/05/17/wannacry-ransomware-hit-real-medical-devices/#741ce643425c

[26] L. Okman, N. Gal-Oz, Y. Gonen, E. Gudes, and J. Abramov, "Security issues in NoSQL databases," in *Proc. Int. Jt. Conf. IEEE Trust. ICESS-11/FCST-11*, IEEE. IEEE, 2011, pp. 541–547.

[27] A. Ron, A. Shulman-Peleg, and A. Puzanov, "Analysis and mitigation of NoSQL injections," *IEEE Secur. Priv.*, vol. 14, no. 2, pp. 30–39, 2016.

[28] D. Chahal, L. Kharb, and M. Gupta, "Challenges and security issues of NoSQL databases," *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, vol. 2, no. 5, pp. 976–982, 2017.

3D Histopathology – A Lung Tissue Segmentation Workflow for Microfocus X-ray Computed Tomography Scans

Lasse Wollatz [1*], Steven J. Johnston [1], Peter M. Lackie [2] and Simon J. Cox [1]

Abstract

Lung histopathology is currently based on the analysis of 2D sections of tissue samples. The use of microfocus X-ray computed tomography (μCT) imaging of unstained soft tissue can provide high resolution 3D image datasets in the range of 2-10μm without affecting the current diagnostic workflow. Important details of structural features such as the tubular networks of airways and blood vessels are contained in these datasets but are difficult and time consuming to identify by manual image segmentation.
Providing 3D structures permits a better understanding of tissue functions and structural inter-relationships. It also provides a more complete picture of heterogeneous samples. In addition 3D analysis of tissue structure provides the potential for an entirely new level of quantitative measurements of this structure that have previously been based only on extrapolation from 2D sections.
In this paper a workflow for segmenting such 3D images semi-automatically has been created using and extending the ImageJ open-source software and key steps of the workflow have been integrated into a new ImageJ plug-in called LungJ.
Results indicate an improved workflow with a modular organization of steps facilitating the optimization for different sample and scan properties with expert input as required. This allows for incremental and independent optimization of algorithms leading to faster segmentation. Representations of the tubular networks in samples of human lung, building on those segmentations, have been demonstrated using this approach.

Keywords: Lung, Image Segmentation, Computed Tomography, Histopathology, ImageJ, Vascular Network

1. Introduction

Histopathology provides structural details of tissue samples on a cellular level allowing disease-associated changes to be identified. It offers a key diagnostic tool for fibrotic lung diseases, particularly those that cannot be clearly identified on the basis of patient computed tomography (CT) or high resolution CT (HRCT) [1 - 4] and is often regarded as the gold standard [5, 6]. For routine histopathology, surgical biopsies are taken from a patient, providing three dimensional (3D) tissue samples that are chemically fixed to preserve tissue structure and then embedded in wax to allow for histological sectioning. For diagnostic purposes, sections are stained to identify the overall tissue structure and highlight certain tissue components before analysis under a microscope by a trained histopathologist (see Fig. 1). These microscopy images are increasingly provided through digital scanning of tissue slices. This allows analysis based on the digital image (virtual microscopy). Systems for computer assisted diagnosis (CAD) used to highlight relevant features or suggest a possible diagnosis are also becoming available [5, 7].

The established methods in histopathology have proven to be critical to the diagnosis of many lung diseases. There are some diseases like idiopathic pulmonary fibrosis (IPF), non-specific interstitial pneumonia (NSIP) or extrinsic allergic alveolitis (EAA) where diagnosis is difficult and an agreement between histopathologists is generally only fair to moderate (kappa agreement coefficients ranging from 0.2 to 0.7) [8]. By producing a three dimensional scan of the sample additional information about the tissue will be gained [9]. Instead of single slices, 100 to 1000 times the number of slices can be viewed, at various slice orientations. This aids the identification of rarer structural changes and reveals the degree of heterogeneity in tissue structure. Further benefits include analysis of 3D structures, specifically 3D networks, revealing tissue function as well as interrelationships

[1] Faculty of Engineering and the Environment, University of Southampton, SO17 1BJ, United Kingdom

[2] Faculty of Medicine, University of Southampton, SO17 1BJ, United Kingdom

[*] Corresponding Author: Lasse Wollatz, email: L.Wollatz@soton.ac.uk

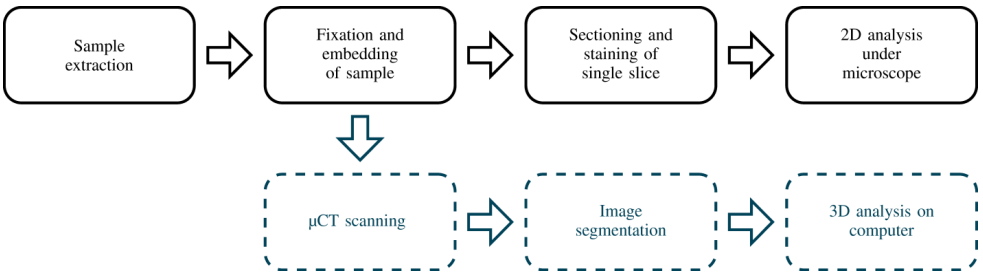


Fig. 1: Current histology workflow (top) and proposed workflow (bottom). As µCT is non-invasive this can be seen as an additional approach.

between objects [9, 10] enabling the application of established stereological methods [11].

Microfocus X-ray computed tomography (µCT) of soft-tissue embedded in wax produces relatively high resolution (~7µm or better), but low contrast, 3D images. Identification of key features inside these images is challenging but important. Image segmentation describes this process of distinguishing between the areas of interest in an image and the remaining area. Parts of the image which are not of interest are commonly removed or areas of different features are marked with different (digital) labels. Segmenting some features such as the walls of the airways and blood vessels manually is effectively impossible; manual segmentation of the lumen, while possible [12] is time consuming taking weeks or months per dataset. Automatic or semi-automatic segmentation is therefore required to make this a useable method for research or diagnosis.

Many methods [13 - 17] have been developed for automatic segmentation of images in medicine. Standard approaches such as intensity thresholding [18], watershed algorithms [19] and contour region growing [15] do not have the required sensitivity or specificity when used for noisy low contrast images like soft tissue. Atlas based approaches [20] require an ideal model which cannot be created for small and very variable structures like alveoli. Machine learning approaches can reduce the amount of human input and are flexible enough to deal with low contrast images. They have previously been used for computer aided diagnosis in histology and for airway detection in lung patient CT [21, 22].

We present here a workflow that allows semi-automated segmentation of airways and blood vessels and its implementation as LungJ using the free, open source software ImageJ [23]. The workflow is scalable to large 3D µCT scans and targets lung tissue samples. Section 2, Materials & Methods, provides details on sample preparation and image acquisition. The procedural steps of the image processing workflow itself are presented in section 3. Background, as well as choices made for individual workflow procedures are explained for each step. A short summary of the workflow is provided in the conclusions in section 4.

2. Materials & Methods

A number of steps were required before the digital segmentation: Tissue was prepared for scanning, while scanning conditions were optimized to provide the best possible resolution and contrast for the tissue; Scans also had to be pre-processed before segmentation.

2.1. Sample Preparation

Surgically resected human lung tissue was obtained with written informed consent from patients undergoing elective surgery at University Hospital Southampton. UK ethics approval was given from the National Research Ethics Service Committee, South Central - Southampton A, number 08/H0502/32. A representative sample was taken from a macroscopically healthy portion of

lung of a 75 year old male non-smoker with GOLD stage 0 chronic obstructive pulmonary disease (COPD) undergoing resection for lung cancer. The samples were fixed with para-formaldehyde in phosphate buffer for 8 hours and embedded in wax following a routine histology protocol using a Shandon Hypercenter tissue processor (Fisher Scientific, Loughborough, UK). This resulted in blocks of about 1.5cm^3 which were then scanned without any further treatment.

2.2. Image Acquisition and Pre-Processing

Samples were scanned using a custom-built Nikon Metrology µCT scanner (µ-VIS, Southampton, UK). An electron accelerating voltage of 50kV was used, with a molybdenum reflection target, yielding a mean energy in the order of 18keV, along with characteristics peaks around 17 and 19keV. No filtration was used. A beam current of approximately 150µA was selected, yielding a total beam energy below 8W. Reconstructions with voxel resolution of 8µm were created, using standard filtered-back projection (Ram-Lak filter) within the Nikon CT Pro 2.0 package using 3142 projections (360˚ rotation in 0.11˚ increments). Typical datasets included 1800x2000x2000 isotropic voxels. The value of each voxel corresponds to the density of the material at that position and was acquired at 32-bit resolution. As a comparison: Hospital CTs acquire 512 x 512 x 80 voxel datasets at 16-bit resolution.

3. The Workflow

In order to simplify the workflow and avoid repetitive steps, a software plug-in for ImageJ was developed. This plug-in, called LungJ, had to be capable of handling the large data size of individual scans and segmenting images in a way that is meaningful to and can be interpreted by the final user. An image segmentation method had to be established and images were post-processed to remove noise and artifacts. Finally, representation of the 3D image data was explored to ensure better understanding of the underlying structure. Each of these distinct steps are detailed below, highlighting important decisions made while creating this workflow.

3.1. Choice of Software

It was important that the software package of choice, which a plug-in would be programmed for, had to be one that was familiar to potential users. This would maximize usability, especially for users less familiar with image processing. Several software packages were evaluated for implementation of this method [24]. Paid software, such as Amira 3D [25] or OsiriX [26] was not user configurable to the required degree or limited to certain operating platforms. Table 1 shows that ImageJ [23, 27] is, together with Matlab, the most widely used image processing software in medical research. ImageJ runs in Java and is therefore platform independent. It has the advantage over Matlab of being open-source freeware. The pre-packaged Fiji (Fiji Is Just ImageJ) distribution of ImageJ also comes with a plug-in for machine learning based image segmentation.

Based on the framework provided by Fiji, a plug-in was developed to tackle the different tasks of image segmentation. LungJ [28] tools include a function for segmenting an image, as well as tools for pre- and post-processing the image. While all these processes can be done in Fiji, the tools provide important simplifications, as they reduce the amount of knowledge a user needs to have about the image (e.g. bit-depth or absolute values of voxels). Instead of opening the image, launching the Trainable WEKA Segmentation (TWS) [29], loading the classifier, applying it and then filtering out the relevant mask, with LungJ all of this can be done from a single graphical user interface (GUI) which in turn calls the relevant TWS functions. The overall structure of LungJ is very modular and therefore adaptive, in order to align with the ImageJ philosophy. While the original scan can be in any format handled by ImageJ, LungJ saves intermediate steps as loss-less tagged image files (TIF). By combining multiple modular functions in one GUI and allowing default settings to be defined, it was possible to reduce the number of steps of the workflow.

Table 1: Popularity of different imaging software in research, based on results from 15th May 2016 (www.ncbi.nlm.nih.gov/gquery and scholar.google.com). Search results for Amira 3D excluded authors with the name Amira.

Software	Type	Developer	PubMed Central Papers	Google Scholar Results
ImageJ	Open Source	Wayne Rasband	75 400	157 000
MATLAB	Commercial	Mathworks	67 305	2 010 000
NIS-Elements	Commercial	Nikon	6 184	18 000
Imaris	Commercial	Bitplane	4 842	12 300
Amira 3D	Commercial	FEI	1 903	13 800
SlideBook	Commercial	3i	2 807	6 010
ImagePro Plus	Commercial	MediaCybernetics	2 464	5 420
OsiriX	Commercial (Free version available)	Pixmeo	1 515	9 320
CellProfiler	Open Source	CellProfiler Team, MIT	1 055	3 350
ITK-SNAP	Freeware	University of Pennsylvania and University of Utah	505	2 610
Avizo 3D	Commercial	FEI	371	4 210
VGStudio	Commercial	Volume Graphics	125	1 620
Vaa3D	Open Source	Hanchuan Peng and Howard Hughes	77	216
BioImageXD	Open Source	Pasi Kankaanpää Lassi Paavolainen Varpu Marjomäki Jyrki Heino et al.	69	241

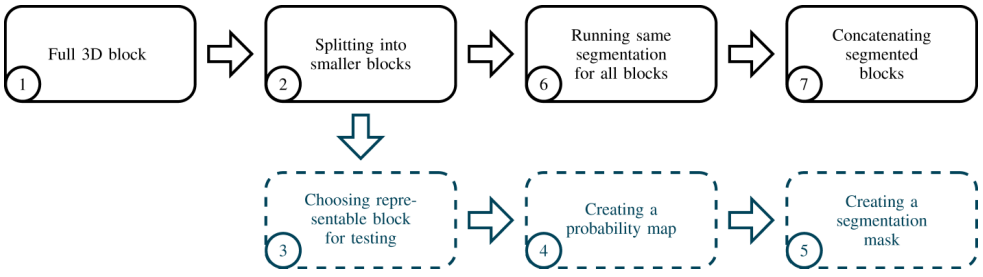


Fig. 2: Modular approach of LungJ solving several problems of the image segmentation.

3.2. Handling Large Images

The output image volumes were several tens of gigabytes each. Image processing requires multiple times the memory size of an image, i.e. several tens to hundreds of GB Random Access Memory (RAM). In order to be able to process these large image-files they were split into smaller sub volumes and processed separately, enabling us to complete processing on a computer with

only 16GB of RAM while still allowing 3D interrelationships with the volume to be addressed. Currently available methods, like the virtual image stack do not allow custom sized image blocks and tests using them, in conjunction with the image segmentation algorithm shown in the next section, caused an out of memory error. The image blocks are loaded individually and only on demand. Global properties of the image were saved in a separate file and used during further processing steps to ensure consistency of intensity values over the whole volume. This introduced a novel image representation method which required ImageJ to only hold a directory path instead of loading a full image. Using this representation, any available ImageJ filter can be applied to the whole image. Furthermore, image properties such as the intensity distribution histogram can be produced. The use of smaller image blocks also enables future adaption of this process in parallel computing or cloud computing.

Three functions for creating, processing and concatenating image sub-volumes were implemented. They form the backbone of the sub-volume image processing. As shown in Fig. 2, an image was split prior to the segmentation and a representable block was chosen to test the segmentation. This enabled faster and hence more efficient testing and troubleshooting of a segmentation technique compared to running a segmentation on a full scan. Once the segmentation procedure had been established, it was applied to all the sub-volumes by processing each one in turn. Finally the sub-volumes were assembled into a single image by applying the concatenating function.

A further addition was the use of halos enclosing the image blocks, as commonly used in parallel computing. The use of halos avoided boundary artefacts, and a function for halo-exchange was implemented. Cubical sub-volumes were used when applying three dimensional processing methods because of their smaller surface area compared to other cuboids.

With these tools in place a segmentation was achieved on an Intel-i7-4770, with 3.40GHz and 16GB RAM within 12 hours whereas manual segmentation of similar images could take between 3 and 6 months. As shown in Fig. 3 for a run on a computer with a slower Intel Pentium G2020T at 2.50GHz, the processing speed does not change significantly over time despite a memory leak in ImageJ causing full RAM usage.

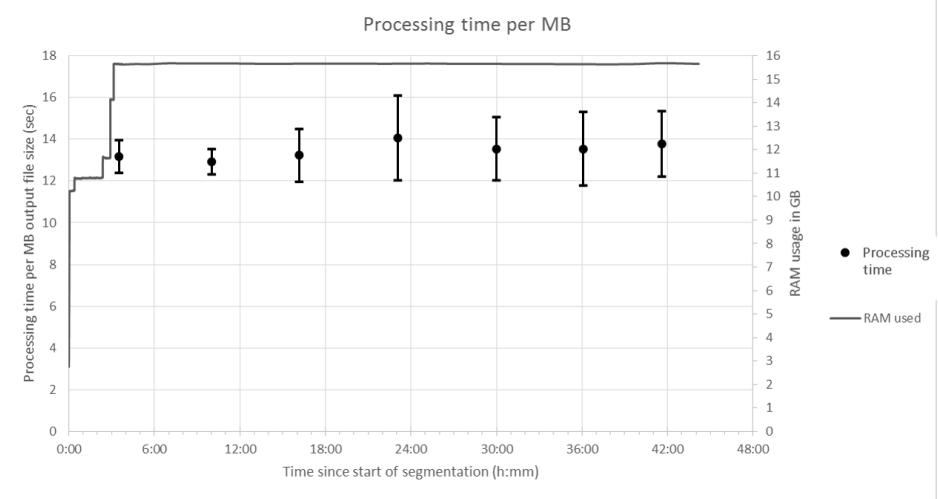


Fig. 3: Processing speed of blocks based on the slowest of the runs (44 hours, Intel Pentium processor). The average processing speed over 35 consecutive blocks was taken and standard errors calculated. RAM usage is shown for reference.

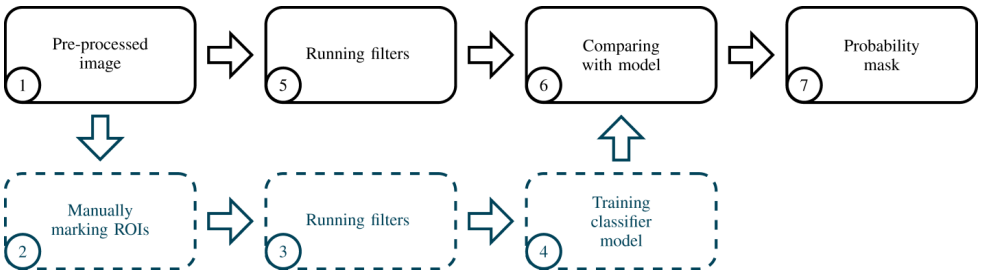
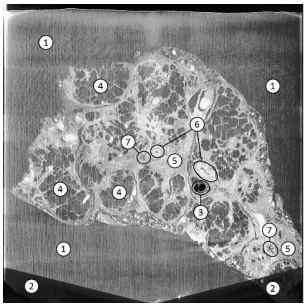
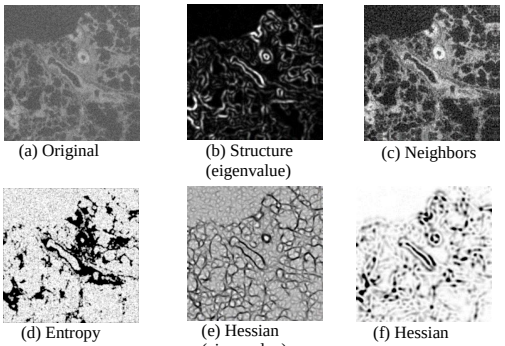


Fig. 4: Machine learning algorithm. Based on partial manual segmentation a classifier model is trained and then applied to the whole image.



① wax/ background
② area outside of scan
③ air bubble in wax
④ alveolar area
⑤ connective tissue
⑥ blood vessel
⑦ airway

Fig. 5: A single section of a µCT scan of wax embedded human lung tissue showing areas of wax, area outside the scan and distinct features of interest. Features 5, 6 and 7 were targeted for this project



(a) Original (b) Structure (eigenvalue) (c) Neighbors

(d) Entropy (e) Hessian (eigenvalue) (f) Hessian (orientation)

Fig. 6: Original image and result of different filters applied.

3.3. Image Segmentation

A machine learning algorithm was applied for the image segmentation. Fiji comes with a segmentation plug-in called Trainable WEKA Segmentation [29]. For the segmentation the random-forest algorithm was chosen. This algorithm has been used for image segmentation of neuronal processes in biomedical applications as well as segmentation of larger vessels in CT scans of whole lungs before [30, 31]. The application of the machine learning algorithm involved the creation of a classifier model which had to be trained by using reference data and filters, as outlined in Fig. 4. The algorithm could then determine the likelihood of each voxel of further images (or further parts of the same image) belonging to the feature of interest.

As values of the voxels are proportional to the sample's X-ray absorption, any pre-processing methods affecting the histogram will also lead to a loss of information within the image or relative to other scans. Therefore no normalization or equalization was performed. This underlines the importance of a standardized sample preparation and scanning procedure to produce comparable images. No noise reduction filter was applied apart from a Gaussian blur prior to other feature detecting filters (outlined in the end of this section).

Reference data was provided in the form of manually segmented images. For large parts of the image it was not possible to clearly differentiate and manually segment vessel wall and surrounding tissue. Accordingly manual training slices were selected, which showed the boundaries of the vessel wall as clearly as possible. Two airways and two blood vessels were selected as foreground and four representative areas that were not of interest were selected as background. Whilst selection of more training data can improve the classifier model, too much training data in this case resulted in over-segmentation due to the difficulty in accurately selecting training data manually. A selection of a wall including the inside area gave better results, as it included edges

which are easier to identify than other features when using filters. The inside of a vessel matched the background, as both were filled with wax and therefore had the same density (see Fig. 5). This caused an over-segmentation of the background which was dealt with during the post-processing detailed in the following section.

A set of image filters is provided by the TWS and a selection of these were chosen based on the features to detect. Features of interest are shown in Fig. 5. As key structural elements of the lung, this project focused on detecting blood vessels and airways. This method can also be used to detect other features such as fibrotic structures as shown in the Data Presentation section. Vascular structures are characterized by a bright (X-ray dense) surrounding and a dark (X-ray lucent) center. These attributes suggested the use of filters targeting edges. The choice of filters was based on literature review as well as screenings of all available filters with a representable image as shown in Fig. 6. Filters like 'Structure', 'Neighbors', 'Entropy' and 'Hessian' were able to separate areas of interest from the rest of the image. The Hessian filter highlights edges and was proposed for tube-like structure detection by Sato et al. [32]. The use of 'Entropy' filters for CT images is encouraged by Bae et al. [33].

Next a classifier model was created. The reference segmentation as well as the choice of filters were provided to the TWS which trained a classifier model.

Using the TWS fast-random-forest algorithm and 'Structure', 'Neighbors', 'Entropy', and 'Hessian' filters, it was possible to train a classifier model from a manual segmentation. This model was used to segment the whole µCT scan. The result was a probability map where higher values represent a larger likelihood of a voxel being an airway or a blood vessel.

3.4. Image Post-Processing

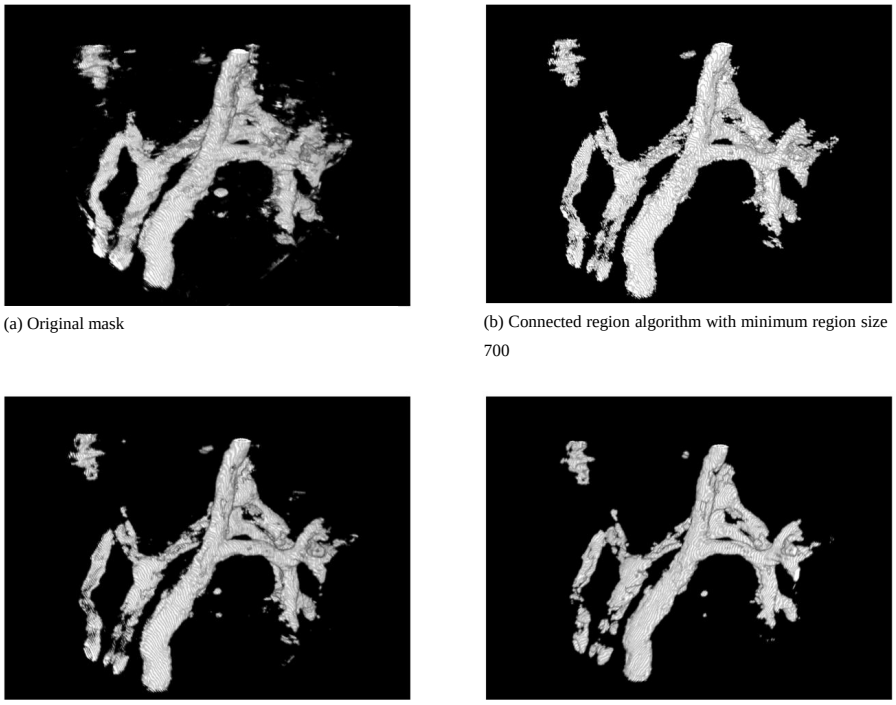
The probability map was converted into a binary image (mask), where each voxel was given a value of 0 or 1, representing areas of no interest and areas of interest, respectively. This was done by manually defining and applying a threshold. This mask contained the vessel walls as well as some areas surrounding both the inside and outside of the wall. To remove the voxels from the mask which were not part of the vessel walls, the mask was applied to the original image and then another threshold was applied to remove the darker areas representing wax. This threshold was again chosen manually.

This second mask contained some undesired regions, such as air bubbles or noise. Whilst erode and dilate operations are useful tools to remove noise [34], they do affect the surface of correctly identified regions. Erosion of a mask removes all voxels within a radius from the masks' surface, while dilation adds voxels within a certain radius from the masks' surface to the mask. For both operators the radius is usually equal to the size of one voxel. The combination of one erode operation and one dilate operation of equal radius is referred to as opening and the opposite action of dilating before eroding is referred to as closing.

Erode and dilate algorithms have been used in medical image processing previously [35]. The effect of applying them in comparison to an algorithm identifying and removing small connected regions has been studied. Results of the minimum region size of the connected region algorithm, as well as two different implementations of erode and dilate functions are presented in Fig. 7 (b) and Fig. 7 (c)-(d), respectively. Fig. 7 (c) used a modified version of the algorithm presented by Antonelli et al. [34], which has been slightly optimized using the idempotent nature of the operators, whilst the others were based on a set of trial runs.

It was found that algorithms for removing small connected regions reduce the loss of shape but are multiple times more time-consuming (see Table 2), adding about 1 hour to the total processing time. Noise removal through removal of "small connected regions" shown in Fig. 7 (b) gives better results than opening and closing operations shown in Fig. 7 (c)-(d). The more sophisticated algorithm leaves the actual boundary/surface of the vessels untouched while erode and dilate operations affect both noise and valid segmentation. For the large connected vessels erode and dilate is not strong enough for efficient noise removal.

After post-processing of the map, only the vessel walls were left and noise had largely been removed.



(a) Original mask

(b) Connected region algorithm with minimum region size 700

(c) Opening (erode, dilate) followed by closing (dilate, erode) operation

(d) Closing,opening and radius 2 opening operation

Fig. 7: Difference between connected regions and erode/dilate algorithm for noise removal. Note how erosion and dilation either do not remove a lot of noise (c) or visibly remove part of the structure of interest (d).

Table 2: Timings for two different images on two different machines. Image 1 was 1600x1600x1200px in size at 16bit and Image 2 1374x1497x500px at 8bit. Machine 1 had 16GB RAM, 64-bit, Intel Pentium CPU G2020T @2.50 GHz and a local HDD Drive, Machine 2 has 16GB RAM, 64bit Intel i7-4770 @3.40GHz and a network drive. 2 runs were done and averaged for image 2 and 1 for image 1.

Image	**Computer**	**Blocks**	**Time (h:mm)**					
			Block creation	**Segmen-tation**	**Thres-holding**	**Erode Dilate**	**Connected Regions**	**Total**
1	1	245 (no halo)	0:01	44:11	0:04	0:01	0:52	44:17 to 45:08
2	1	72 (16px halo)	0:00 (9sec)	17:59	0:01	0:01	1:14	18:01 to 19:14
1	2	245 (no halo)	0:19	11:38	0:16	0:17	0:42	12:30 to 12:55
2	2	72 (16px halo)	0:04	5:05	0:06	0:06	0:47	5:21 to 6:02

3.5. Data presentation

Once a segmentation mask had been created, a selection of presentation views were generated. This included 2D, projected 3D and 3D data representation [17, 36].

The major advantage of the proposed workflow over established microscopy is the extraction of 3D data. Projected 3D views (2D representations of 3D models) of individual features were created. Fiji 3D allowed the display of interactive 3D surface and volumetric images as in Fig. 8 (b). More elaborate rendering of 3D surface views and videos was possible by using Avizo. This could be outside views (Fig. 8 (d)) but also inside views of blood-vessels or airways (Fig. 8 (e) and (f) respectively) similar to virtual endoscopy [17]. The inside view shown in Fig. 8(e) allows clear identification of the airway and can show details about vessel size while removing the complexity of the full network.

Apart from projected 3D images, a physical 3D replica was also created. Conversion of the masks to 3D surface objects was possible using Fiji 3D. The use of Blender allowed smoothing of the surface and creation of printable stereo lithography (STL) files, one of which was printed using a Tiertime UP! Plus 2 printer (Fig. 8 (g)). 3D printed models improve the understanding of structures beyond the ability of 3D rendered graphics which only show a 2D projection [37].

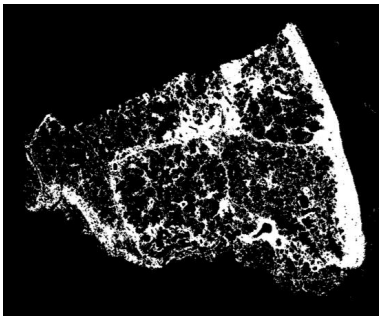
Having a variety of ways of representing the data improved the insight gained into the 3D structure, since a relation to known representations as well as a new 3D representation were created.

4. Conclusions

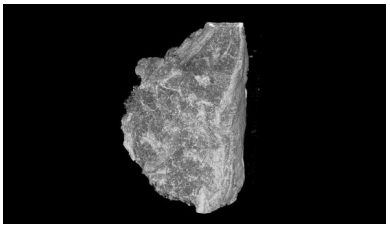
Current histopathology relies on a limited 2D analysis of tissue samples that are intrinsically 3D and are affected by local challenges and pathological changes that may be highly heterogeneous at several different length scales. The use of 3D scanning techniques to accompany current histology procedures can improve insight into tissue structures and accordingly understanding and diagnosis of chronic lung diseases. The workflow for segmentation of multi-gigabyte µCT scans of lung tissue presented in this paper demonstrates this alternative method. It was found that splitting the large µCT images into smaller blocks permits handling of 3D images without requiring unreasonably powerful computers. At the same time it enables fast testing of a segmentation strategy. Trainable image segmentation using ‘Structure’, ‘Neighbors’, ‘Entropy’ and ‘Hessian’ filters provided good results with vessels being clearly distinguishable. The connected regions algorithm was shown to be the more accurate noise removal algorithm compared to erode and dilate operations. Different ways of data representation were discussed including 2D color maps, 3D rendering and 3D printing. The workflow was implemented in the popular image processing software ImageJ in a modular fashion which allows for optimization of each single part of the workflow.

Using this method, tubular structures were successfully segmented from an embedded tissue sample without making assumptions about the 3D structure beforehand. Not only was it possible to reduce the processing time from months of manual segmentation to half a day to 2 days of semi-automated segmentation, but also to be able to scale it to volumes of many tens of GBs without reduction in processing speed. The results clearly identified a 3D network of vessels and various display methods were used to reveal more information about the tissue than previously possible through microscopy.

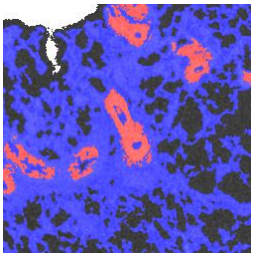
Another advantage of the proposed method is its implementation as a modular workflow in open source software. This allows individual segments of the image processing to be improved and optimized as required without revising the whole protocol. User input for training of the algorithm allows flexible interactive selection of features of interest. Reduction of the number of user steps required to operate LungJ allows much easier application of this method in medical research. The workflow has been designed to aid users, provide feedback of success, and facilitate visual presentation of results.



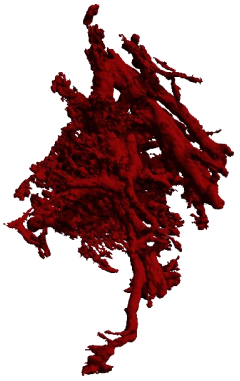
(a) Fibrotic area of cross-section.



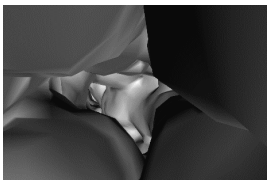
(b) 3D view of the fibrotic structure



(c) Colorized overlay of masks on top of original image

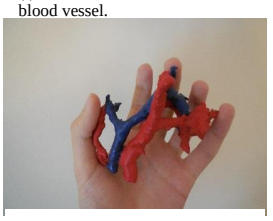


(d) 3D view of vessels



(e) 3D render of the inside of an airway.

(f) 3D render of the inside of a blood vessel.



(g) 3D print in ABS post-painted (airways blue, blood vessels red)

Fig. 8: Image representations created after segmenting the whole tissue volume.

As no other workflows for this task are known to the authors, evaluating the method empirically is difficult [38]. Method evaluation can be achieved using unsupervised image segmentation evaluation methods [39]. The modular approach enables the proposed workflow to be adapted to other types of tissue or image conditions. Having this workflow in place shifts focus towards the efficiency of individual modules in future. The implementation of this workflow in an end-to-end system, including dataset management [40], is also envisioned. Furthermore, we hope to be able to apply this method to images of different diseases to identify parameters of importance and in the long run improve the diagnostic certainty of histopathology.

Acknowledgment

The authors would like to acknowledge both Stephanie Robinson and James Cairns for their contribution to the manual segmentation of training data. Orestis L. Katsamenis and Ian Sinclair provided advice and did the imaging of the tissue sample using the µCT scanners at the µ-VIS X-Ray Imaging Centre. The authors would also like to thank the staff of the Biomedical Imaging Unit at University Hospital Southampton for their support and Shoufeng Yang for the access to 3D printing facilities. Lasse Wollatz was funded in part by a grant from Engineering and Physical Research Council (EPSRC 1511465). Data supporting this study are openly available from the University of Southampton repository (http://dx.doi.org/10.5258/SOTON/401280).

References

[1] Brown KK: Chronic cough due to nonbronchiectatic suppurative airway disease (bronchiolitis): ACCP evidence-based clinical practice guidelines. Chest 129:132S–137S, 2006

[2] Dawson JK, Graham D R, Desmond J, Fewins H E, Lynch M P: Investigation of the chronic pulmonary effects of low-dose oral methotrexate in patients with rheumatoid arthritis: a prospective study incorporating HRCT scanning and pulmonary function tests. Rheumatology 41:262–267, 2002

[3] Webb WR, Müller NL, Naidich DP: High-Resolution CT of the Lung, 5: Lippincott Williams & Wilkins, 2014

[4] Ko JP, Bargo LJ, Broderick LS, Hunsaker A, Chung JH, Lynch D, Stern EJ: ACR-STR practice parameter for the performance of high-resolution computed tomography (HRCT) of the lungs in adults. The American College of Radiology 17, 2015. Available at http://www.acr.org/Quality-Safety/Standards-Guidelines/Practice-Guidelines-by-Modality/Chest. Accessed 2 December 2016

[5] He L, Long LR, Antani S, Thoma GR: Computer Assisted Diagnosis in Histopathology. In Zhao Z: Sequence and Genome Analysis: Methods and Applications, 3, Hong Kong: iConcept Press:271–287, 2011

[6] Raghu G, Rochwerg B, Zhang Y, Garcia CAC, Azuma A, Behr J, Brozek JL, Collard HR, Cunningham W, Homma S, Johkoh T, Martinez FJ, Myers J, Protzko SL, Richeldi L, Rind D, Selman M, Theodore A, Wells AU, Hoogsteden H, Schünemann HJ: An official ATS/ERS/JRS/ALAT clinical practice guideline: Treatment of idiopathic pulmonary fibrosis: An update of the 2011 clinical practice guideline. Am J Respir Crit Care Med 192:e3–e19, 2015

[7] Ross MH, Pawlina W: Histology: A Text and Atlas, 6: Lippincott Williams & Wilkins, 2011

[8] Nicholson AG, Fulford LG, Colby TV, du Bois RM, Hansell DM, Wells AU: The relationship between individual histologic features and disease progression in idiopathic pulmonary fibrosis. Am J Respir Crit Care Med 166:173–177, 2002

[9] Hsia CCW, Hyde DM, Ochs M, Weibel ER: An official research policy statement of the American Thoracic Society/European Respiratory Society: Standards for quantitative assessment of lung structure. Am J Respir Crit Care Med, 181:394–418, 2010

[10] Mitzner W, Weibel ER: Standards for quantitative assessment of lung structure. J Appl Physiol 109:934, 2010

[11] Hsia CCW, Hyde DM, Ochs M, Weibel ER: How to measure lung structure - What for? On the `Standards for the quantitative assessment of lung structure'. Respir Physiol Neurobiol 171:72–74, 2010

[12] Scott AE, Vasilescu DM, Seal KAD, Keyes SD, Mark NM, Hogg JC, Sinclair I, Warner JA, Hackett T-L, Lackie PM: Three dimensional imaging of paraffin embedded human lung tissue samples by micro-computed tomography. PLoS One 10, DOI: 10.1371/journal.pone.0126230, June 1, 2015

[13] Pham DL, Xu C, Prince JL: Current methods in medical image segmentation. Annu Rev Biomed Eng 2:315–337, 2000

[14] Dougherty G: Medical Image Processing - Techniques and Application: Springer, 2011

[15] Angenent I, Pichon E, Tannenbaum A: Mathematical methods in medical image processing. Bull New Ser Am Math Soc 43:365–396, 2006

[16] Sonka M, Fitzpatrick JM: Handbook of Medical Imaging - Medical Image Processing and Analysis, 2: SPIE Press, 2009

[17] Robb RA: Biomedical Imaging, Visualization, and Analysis: John Wiley & Sons, 2000

[18] Gonzalez RC, Woods RE: Digital Image Processing, 2, USA: Prentice Hall, 2002

[19] Grau V, Mewes AUJ, Alcañiz M, Kikinis R, Warfield SK: Improved watershed transform for medical image segmentation using prior information. IEEE Trans Med Imaging 23:447–458, 2004

[20] Michopoulou SK, Costaridou L, Panagiotopoulos E, Speller R, Panayiotakis G, Todd-Pokropek A: Atlas-based segmentation of degenerated lumbar intervertebral discs from MR images of the spine. IEEE Trans Biomed Eng 56:2225–2231, 2009

[21] Doyle S, Madabhushi A, Feldman M, Tomaszeweski J: A boosting cascade for automated detection of prostate cancer from digitized histology. Med Image Comput Comput Assist Interv 4191:504–511, 2006

[22] Park W, Hoffman EA, Sonka M: Segmentation of intrathoracic airway trees: A fuzzy logic approach. IEEE Trans Med Imaging 17:489–497, 1998

[23] Abramoff MD, Magalhaes PJ, Ram SJ: Image processing with ImageJ. Biophotonics Int 11:36–42, 2004

[24] Eliceiri KW, Berthold MR, Goldberg IG, Ibáñez L, Manjunath BS, Martone ME, Murphy RF, Peng HC, Plant AL, Roysam B, Stuurman N, Swedlow JR, Tomancak P, Carpenter AE: Biological imaging software tools. Nat Methods 9:697–710, 2012

[25] Stalling D, Westerhoff M, Hege H-C: Amira: A Highly Interactive System for Visual Data Analysis. In Hansen CD, Johnson CR: The

Visualization Handbook. USA: Elsevier Butterworth-Heinemann:749–767, 2005

[26] Rosset A: OsiriX MD. 510(k) Summary of Safety and Effectiveness K101342. Bernex: Food and Drug Administration, August, 2010. Available at https://www.accessdata.fda.gov/cdrh_docs/pdf10/K101342.pdf. Accessed 15 May 2015

[27] Schneider CA, Rasband WS, Eliceiri KW: NIH Image to ImageJ: 25 years of image analysis. Nat Methods 9:671–675, 2012

[28] Wollatz L, Johnston SJ, Lackie P, Cox SJ: LungJ. Dataset, University of Southampton, DOI: 10.5258/SOTON/401280, November, 2016

[29] Arganda-Carreras I, Kaynig V, Schindelin J, Cardona A, Seung HS: Trainable Weka Segmentation: A machine learning tool for microscopy image segmentation. Neuroscience 2014 73–80, 2014

[30] Kaynig V, Fuchs T, Buhmann JM: Neuron geometry extraction by perceptual grouping in ssTEM images. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2902–2909, 2010

[31] Rudyanto RD, Kerkstra S, van Rikxoort EM, Fetita C, Brillet PY, Lefevre C, Xue W, Zhu X, Liang J, Öksüz İ, Ünay D, Kadipaşaoğlu K, Estépar RSJ, Ross JC, Washko GR, Prieto JC, Hoyos MH, Orkisz M, Meine H, Hüllebrand M, Stöcker C, Mir FL, Naranjo V, Villanueva E, Staring M, Xiao C, Stoel BC, Fabijanska A, Smistad E, Elster AC, Lindseth F, Foruzan AH, Kiros R, Popuri K, Cobzas D, Jimenez-Carretero D, Santos A, Ledesma-Carbayo MJ, Helmberger M, Urschler M, Pienn M, Bosboom DGH, Campo A, Prokop M, de Jong PA, Ortiz-de-Solorzano C, Muñoz-Barrutia A, van Ginneken B: Comparing algorithms for automated vessel segmentation in computed tomography scans of the lung: The VESSEL12 study. Med Image Anal 18:1217–1232, 2014

[32] Sato Y, Nakajima S, Shiraga N, Atsumi H, Yoshida S, Koller T, Gerig G, Kikinis R: Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images. Med Image Anal 2:143–168, 1998

[33] Bae KT, Kim J-S, Na Y-H, Kim KG, Kim J-H: Pulmonary nodules: Automated detection on CT images with morphologic matching algorithm - Preliminary results. Radiology, DOI: 10.1148/radiol.2361041286, December 10, 2005

[34] Antonelli M, Lazzerini B, Marcelloni F: Segmentation and reconstruction of the lung volume in CT images. 2005 ACM Symposium on Applied Computing 255–259, 2005

[35] Hu S, Hoffman EA, Reinhardt JM: Automatic lung segmentation for accurate quantitation of volumetric X-ray CT images. IEEE Trans Med Imaging 20:490–498, 2001

[36] Hansen CD, Johnson CR: The Visualization Handbook, USA: Elsevier, 2005

[37] Biglino G, Capelli C, Wray J, Schievano S, Leaver L-K, Khambadkone S, Giardini A, Derrick G, Jones A. Taylor AM: 3D-manufactured patient-specific models of congenital heart defects for communication in clinical practice: Feasibility and acceptability. BMJ Open 5:e007165, 2015

[38] Zhang YJ: A survey on evaluation methods for image segmentation. Pattern Recognit 29:1335–1346, 1996

[39] Zhang H, Fritts JE, Goldman SA: Image segmentation evaluation: A survey of unsupervised methods. Comput Vis Image Underst 110:260–280, 2008

[40] Scott M, Boardman RP, Reed PA, Cox SJ: Managing heterogeneous datasets. Inf Syst 44:34–53, 2014

The Use of 3D Printed Paediatric Temporal Bones as a Training Tool

Lasse Wollatz[1], Kasia Konieczny[2], Tim Mitchell[2], Simon J. Cox[1], Andrea Burgess[2], Hasnaa Ismail-Koch[2]

[1]Faculty of Engineering and the Environment, University of Southampton, Southampton; [2]Department of Otorhinolaryngology, University Hospital Southampton NHS Foundation Trust, Southampton

Introduction

Temporal bone dissection is an essential component of otological training. The ideal temporal bone training platform should allow the surgeon to prepare and undertake the actual operative procedure required prior to the operation on the patient. Paediatric temporal bones are generally not available or used due to ethical issues and availability.

Methods

Using specialised computer programmes and non-identifiable paediatric high resolution computed tomography (HRCT) scans of temporal bones 3D printing of paediatric temporal bones has been trialled.

Results

3D printing of paediatric temporal bones has proved both an economical and anatomically accurate tool for training. Using HRCT scans to print from allows cases of appropriate complexity to be selected.

Conclusion

3D printed temporal bones from HRCT images provide optimal opportunities for paediatric temporal bone training and allow for more accurate delineation of structures. Furthermore, they provide improved haptics through choice of appropriate material, especially in complex cases and where anatomy may be difficult.

Web-Based Manipulation of Multiresolution Micro-CT Images

Lasse Wollatz, Simon Cox and Steven Johnston
Faculty of Engineering and the Environment
University of Southampton
United Kingdom
Email: L.Wollatz@soton.ac.uk

***Abstract*—Micro Computed-Tomography (µCT) scanning is opening a new world for medical researchers. Scientific data of several tens of gigabytes per image is created and usually requires storage on a common server such as Picture Archiving and Communication Systems (PACS). Previewing this data online in a meaningful way is an essential part of these systems. Radiologists who have been working with CT data for a long time are commonly looking at two-dimensional slices of 3D image stacks. Conventional web-viewers such as Google Maps and Deep Zoom use tiled multiresolution-images for faster display of large 2D data. In the medical area this approach is being adapted for high resolution 2D images. Solutions that include basic image processing still rely on browser external solutions and high-performance client-machines. In this paper we optimized and modified Brain Maps API to create an interactive orthogonal-sectioning image-viewer for medical µCT scans, based on JavaScript and HTML5. We show that tiling of images reduces the processing time by a factor of two. Different file formats are compared regarding their quality and time to display. As well a sample end-to-end application demonstrates the feasibility of this solution for custom made image acquisition systems.**

I. Introduction

Images have always been important in medicine for diagnostics. Medical instruments such as microscopes and µCT scanners produce images that are several tens of gigabytes in size. The increasing amount of data obtained provides new opportunities for medical researchers but also poses challenges in terms of data storage and retrieval.

In histology sections of *in-vitro* tissue-samples are stained and analyzed under a microscope. The results coming from such analysis are limited to one 2D sample of the extracted tissue. Creating 3D data of the whole tissue using non-destructive µCT prior to the sectioning allows to create a more complete picture of diseases included in the tissue.

µCT data and resulting processed data need to be available to medical researchers in an accessible and meaningful way. One of the critical elements identified was fast previewing of images, in order to allow researchers to decide which images are worth analyzing before retrieving the full image data from a remote storage onto their system for processing. As mobile devices are of increasing importance in today's world and also within medicine, compatibility with mobile devices becomes a key concern [1].

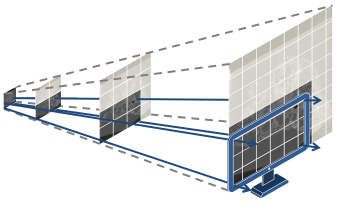


Fig. 1. Multiresolution tiled image - Each resolution level is four times the size of the previous. Tiles are loaded from server for the current view only.

Sending full scale image over the web can take a long time. It is therefore of advantage to save the image as small tiles and only send those parts of the image which are required (compare Fig. 1). It is important to balance the tile size between amount of data to load from the server and amount of server requests made [2].

If the user zooms out, the whole image is displayed on the screen. As this means all the tiles need to be loaded, the image is scaled to several different resolutions and tiled at each of these resolutions. The result is a multiresolution image as shown in Fig. 1. For images where the overhead from multiple tile requests is too large, loading multiple resolutions will still provide a faster response experience. The major idea of the viewer is not to speed up the loading, but to limit the amount of work which needs to be done. For non-medical images, the display of very large images over the web by the use of multiresolution tiled images is a well-established method. Examples are Google Maps, where a satellite image of the entire earth is displayed over various zoom levels with individual tiles being loaded on demand, Deep Zoom, which is based on Silverlight Plug-In, and Grid DataBlade by BCS, which runs in Java [3], [4].

In the medical field the implementation of tiled image viewers is also being established [5], [6]. These solutions are limited to displaying images and do not support image filters.

The Multiresolution Image Viewer (MIV) [5], later renamed to Brain Maps API, is an open source code for displaying high resolution brain images. These images are taken with microscopes, which have a very high resolution

compared to µCT. Even though microscopes only allow to view a small part of the sample at high magnification, methods exist to stitch these images back together to form a complete image of the sample [7]. After being stitched together, the image is converted into a multiresolution, tiled image and stored in a predefined folder structure. Brain Maps API loads and places the image tiles needed for the current view and was extended by StackVis [8], which is browser external, to enable the display of coarsely spaced 3D image stacks and 2D images with a birds-view perspective.

In this paper we describe an implementation of a multiresolution tiled image viewer based on the Brain Maps API [5] that makes it possible to view images and process them pixel based on low-performance devices within web-browser. This improves the capabilities of existing Web-PACS and the practical usability of remote data storage for e-Health. Using Portable Network Graphics (PNG) image tiles instead of commonly used Joint Photographic Experts Group (JPEG or JPG) tiles, better results are obtained as shown in section II. Requirements for speed improvement are discussed in section III and an integration into an image server as well as performance of the code are presented in section IV.

II. Image Format Consideration

Raw data can be stored in various ways, such as uncompressed bits or in a file format like Tagged Image File Format (TIFF). TIFF allows storing 16-bit or 32-bit data as well as being able to store multiple images in one file. Compared to raw data files, it also presents a standard for storing keywords related to the image. As standard TIFF uses 4-byte pointers it can only store file-sizes up to 4GB. Images larger than that are stored as image stacks of several separate 2D TIFFs.

The tiles for the image viewer need to be stored in a web-compliant format. Modern web-browsers support only standard 8-bit images natively commonly JPG, Graphics Interchange Format (GIF) and PNG but also Windows Bitmap (BMP). GIF, which uses an 8-bit indexed color map, has been replaced by PNG due to licensing restrictions [9]. PNG provided 5% better lossless compression than JPEG for small images. JPG can be seen as a standard file format for tiled image viewers [2], [8], [10], [11] but server side solutions also use their own multiresolution file formats or compression methods [11], [12]. JPEG2000, which is increasingly popular in medicine, applies partial lossy compression [13]. This paper compares lossless compressed PNG and lossy compressed JPG and as an uncompressed format BMP. Comparison is made in terms of quality loss due to conversion and quality loss due to compression as well as file-size.

Radiologists can discern 800 to 1000 Just Noticeable Differences (JNDs) within one scene, corresponding to 800 to 1000 individual gray-shades. This means that a 10-bit image format with 1024 gray shades is a minimum requirement for accurate display [14], [15]. A visible loss

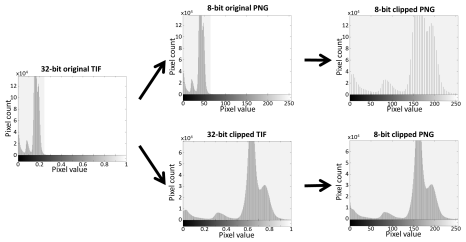


Fig. 2. Quality of image for two possible conversions to 8-bit are compared: The upper row shows the case of the histogram first converted to 8-bit and then stretched, while the lower row shows the histogram first stretched and then converted. The number of gray shades visibly reduces in the first case.

in intensity-resolution is therefore given independent of the choice of file format for the tiling. The original images from the µCT come at 32-bit but are reduced to 16-bit. This is sufficient quality for medical purposes [16]. TIFF allows for high resolution but standard image formats are based on 8-bit integers for gray-scale ranging from 0 to 255. The difference becomes noticeable if the displayed density range is changed and the histogram stretched as illustrated in the top of Fig. 2. A small improvement can be achieved, by partial clipping of the original histogram to make better use of the limited 8-bit palette if the original image uses only a small range of the available gray-scales as shown at the bottom of Fig. 2.

Besides the quality loss due to conversion to 8-bit, the compression-methods of the different image-formats play a role. The different image resolutions vary by a scaling factor of two, meaning that each zoom level contains only a quarter of the pixels of the previous one. This means that in total the tiled image will contain 4/3 the amount of pixels. The overhead of having several files instead of a single one adds to this. JPEG uses a 2D frequency transform to convert the image into the frequency domain and only stores the most important frequencies. This allows for much smaller file sizes but less quality. PNG uses lossless compression and BMP uses none or run-length encoding (RLE) compression only. Fig. 3 shows the difference in terms of image quality. An increased amount of artifacts for JPEG is visible. Lossless formats do not show major differences to the original image, even for large image scaling. The Root Mean Square (RMS) of the difference between the different images and the original TIF were computed for the selected tile. Both BMP and PNG resulted in an RMS of 0.005 while converting the image to JPEG and then tiling it gave a difference RMS of 0.0144 and tiling the image prior to the conversion one of 0.0134. Lossy JPEG compression reduced the image size of the multiresolution image shown in Table I by a factor of 15 which is much less than reported for normal images. This is mainly due to tiling and the increased amount of pixels of the multiresolution image [17]. PNG achieved the expected

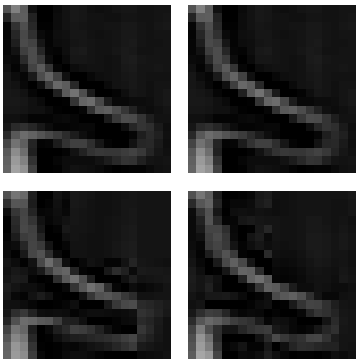


Fig. 3. Quality of JPEG compression compared to PNG based on a 20 by 20 pixel tile. Images on the left side result from the TIF being tiled and then converted to the new file format. Images on the right have first been converted and then cut into tiles. The JPEG images at the bottom show a lot of errors.

TABLE I
SIZE OF TILED IMAGES COMPARED TO THE ORIGINAL UNTILED TIF.

Image	Orig. TIF	Tiled JPG	Tiled PNG	Tiled BMP
Sample 1 (32-bit)	8.13GB	138MB	855MB	2.89GB
Sample 2 (16-bit)	7.09GB	265MB	1.79GB	5.01GB
Sample 3 (16-bit)	60.8GB	2.1GB	9.53GB	21.1GB

compression ratio of a factor of 3 [18].

If the increased amount of data transfer results in an unacceptable delay of page display, then PNG cannot be used for the tiled images. Otherwise it is preferred, as the quality is much greater. If the original data is several tens of gigabytes in size the larger overall data size of PNG compared to JPG-tiles has no major significance.

III. SPEED CONSIDERATION

The speed of web-sites is a major issue [19]. Server, network and coding related improvements were explored in order to ensure short website response times.

The processing/ rendering of the images occurs on the client side. For comparison an AJAX based loading of the images was implemented allowing for server side image processing but was much slower than the client-side processing through pure JavaScript. This agrees with the observations regarding Deep Zoom that server side image generation is slow as it increases the load on the server [20].

In order to decrease the page loading time, the JavaScript routines called on start-up were reduced to a minimum. The image loads on a comparatively small zoom level to keep the number of image files transferred on page load low. For further speed-up the code was modified to reduce the amounts of time tiles were recomputed and loaded. A differentiation was made between the case of tiles needing to be updated and the case where the tiles visible might have changed. In the first case all the tiles need to be recomputed. This occurs if the user changes the image manipulation parameters. The second case requires checking which tiles are within the field of view and loading those, while removing the ones that moved out of the displaying area. This event needs to be triggered whenever the zoom-level is changed, or the image is panned.

A low-resolution thumbnail was placed behind the tiles to allow quick orientation while higher resolution images are loading. To create a better feeling of responsiveness, the thumbnail is always loaded before the tiles and placed in the background. Tiles will then appear on top of the low resolution image, as they are processed. In case of quick scrolling through the dataset, the low resolution image loads much quicker than the tiles and improves the user orientation in the 3D stack.

Another change was made in the amount of slices the user goes through: it was made dependent on the zoom level and the angle of mouse-wheel movement. At higher magnification the user moves only a single slice at a time, while at low resolution he can move 20 images at once. This way it gives the feeling of moving through the image more quickly, while reducing the number of that need to be loaded.

IV. RESULTS

A sample website was built using Webmatrix 3 and PL-upload. If the server receives new images, a message is sent to a queue hosted on Azure which defines the image path. A Python script on the server checks the queue frequently for outstanding messages and creates the image tiles in a subdirectory of the original image. It also creates a header file for the image viewer that contains information about the overall 3D image. The viewer requests that file using AJAX. The pixel values and dimensions are mapped back to the density of an object using the header file. The user can select a Hounsfield unit (HU) range to display within the boundaries of the scanned HU values. By adding the queue request into the uploading process and linking the content request of the Image viewer to the systems database, this viewer can be added to an existing web image server.

For speed evaluation, the time to respond (TTR) and the time to display (TTD) were recorded. The average times are presented in Table II. The TTR defines the time it took to display a preview after a user input, while the TTD describes the amount of time it took till the full resolution image was displayed. In general TTD was 3 to 4% faster for JPG compared to PNG but 25% to 60% faster than for the untiled image. For low zoom levels, the TTD new image slices averaged at 90fps for panning only required loading a selection of new tiles and was even faster (around 6ms). On mobile devices, the advantage of JPG over PNG is much clearer with 16 to 45% speed improvement. Loading and editing the full image on a mobile device took 16 seconds on average but was decreased to 4 seconds by tiling. Due to the multiresolution loading the TTR is much lower at 63ms for a workstation and

TABLE II
TTR AND TTD USING DIFFERENT FILE FORMATS. TTD THE UNTILED PNG ON THE TABLET WAS NOT RECORDED, AS THE BROWSER APPLICATION CRASHED BEFORE COMPLETING THE REQUEST.

Device	Browser	Average	Average TTD [ms]			
		TTR [ms]	JPG	PNG	BMP	Untiled PNG
PC	Opera	32	1401	1456	1530	3514
PC	IE	20	1613	1633	1643	5013
PC	Firefox	94	2962	3056	3025	3975
Mobile	Opera	354	2478	3060	3734	20436
Mobile	Firefox	239	5855	6988	8611	11429
Tablet	Safari	465	6835	7481	8194	—

a fifth of a second for the mobile phone. The average speed of displaying different size images did not differ as expected. The reduction to equal sized tiles allows massive scalability not only in two but also in three dimensions. Results are shown for Opera, Firefox (FF) and Internet Explorer (IE).

Comparing PNG and JPG, differences in compression and speed are negligible. For the given medical applications, PNG was the better choice due to the lower quality of JPG.

For the test, the zoom level was set to ensure a maximum number of images were loaded. Speed was measured from the point of user input to the point of image displayed. It was ensured that images were not cached. These were the worst conditions possible and this is reflected in the results.

The server was hosted with IIS on Windows 8 and the times recorded on a Windows 7 (both 16GB RAM) with Opera 28, the results were compared to FF 35 and IE 11. Each request required 38 tiles to be loaded. Mobile speeds were recorded on a Samsung Galaxy Duos (Android 4, 645MB RAM, FF & Opera, 8 Tiles to load) and iPad (iOS 5, Safari, 20 Tiles to load) using Wi-Fi with VPN. The image loading time fluctuated massively, especially on slower devices. As this was not tested in an isolated environment, the workload on both - server and client - was not constant.

V. CONCLUSION

Advances in the application of µCT in e-Health lead to growing amounts of data requiring organization and remote viewing. Viewing should not be dependent on browser-external solutions, as this limits the compatibility. Multiresolution images can avoid these limitations. We created an image viewer and manipulator for 3D CT data and showed that it can be implemented into an existing system. We demonstrate the feasibility of this method for medical applications in several aspects: a) Using lossless compressed image formats does improve the overall speed of the viewer compared to untiled approaches; b) Basic requirements including pixel based image processing of medical image viewers can be implemented into a web-viewer; c) These features do not restrict the usability of the system for mobile or low-performance devices; d) Using JavaScript and HTML5 these functionalities can be implemented for all main browsers.

In the future we plan to extend the support of input formats and reformatting options to create an image server for practical use. For that we would like to explore direct integration of DICOM as well as security aspects of the transmission of medical images over the web.

REFERENCES

[1] P. T. Johnson, S. L. Zimmerman, D. Heath, J. Eng, K. M. Horton, W. W. Scott, and E. K. Fishman, "The iPad as a mobile device for CT display and interpretation: diagnostic accuracy for identification of pulmonary embolism," *Emerg. Radiol.*, vol. 19, no. 4, pp. 323–327, Aug. 2012.

[2] L. Gerhard, A. Szoforan, and R. Nickolov, "Scalable mutable tiled multi-resolution texture atlases," U.S. Grant 8 385 669, Feb. 26, 2013.

[3] C. Crichton, J. Davies, J. Gibbons, A. Tsui, J. Brenton, C. Caldas, and L. Morris, "Deep Zoom and touch screen for tissue microarray image scoring," in *Proc. of ICSE*, vol. 70, no. 2003. ACM Press, 2008, pp. 217–243.

[4] C. Pendleton, "The world according to Bing," *IEEE Comput. Graph.*, vol. 30, no. 4, pp. 15–17, Jul./Aug. 2010.

[5] S. Mikula, I. Trotts, J. M. Stone, and E. G. Jones, "Internet-enabled high-resolution brain mappingand virtual microscopy," *Neuroimage*, vol. 35, no. 1, pp. 9–15, 2007.

[6] E. J. C. Arguiñarena, J. E. Macchi, P. P. Escobar, M. del Fresno, J. M. Massa, and M. A. Santiago, "Dcm-Ar: a fast Flash-based web-PACS viewer for displaying large DICOM images," in *Proc. of EMBC*. Buenos Aires: IEEE, 2010, pp. 3463–3466.

[7] B. Appleton, A. P. Bradley, and M. Wildermoth, "Towards optimal image stitching for virtual microscopy," in *Proc. of DICTA'05*. Queensland, Australia: IEEE, Dec. 2005, pp. 44–51.

[8] I. Trotts, S. Mikula, and E. G. Jones, "Interactive visualization of multiresolution image stacks in 3D," *Neuroimage*, vol. 35, no. 3, pp. 1038–1043, Apr. 2007.

[9] J. Miano, *Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP*. Addison-Wesley Professional, 1999.

[10] C. K. Chui and H. Wang, "System and method for tiled multiresolution encoding/decoding and communication with lossless selective regions of interest via data reuse," US Grant 09/999,760, Jun. 7, 2005.

[11] W.-K. Jeong, J. Schneider, S. G. Turney, B. E. Faulkner-Jones, D. Meyer, R. Westermann, R. C. Reid, J. Lichtman, and H. Pfister, "Interactive histology and of large-scale and biomedical image and stacks," in *IEEE T. Vis. Comput. Gr.*, vol. 16, no. 6. IEEE, 2010, pp. 1386–1395.

[12] M. J. Gormish, E. L. Schwartz, A. Keith, M. Boliek, and A. Zandi, "Lossless and nearly lossless compression for high-quality images," in *Proc. of Very High Resolution and Quality Imaging II*, vol. 3025. SPIE, Apr. 1997, pp. 62–70.

[13] D. A. Clunie, "Lossless compression of grayscale medical images: effectiveness of traditional and state-of-the-art approaches," in *Proc. of Medical Imaging*, G. J. Blaine and E. L. Siegel, Eds., vol. 3980. San Diego, CA: SPIE, Feb. 2000.

[14] H. R. Blume and E. Muka, "Hard copies for digital medical images: an overview," in *Proc. of Color Hard Copy and Graphic Arts IV*, vol. 2413. San Jose, CA: SPIE, Feb. 1995.

[15] R. A. Robb, *Biomedical Imaging, Visualization, and Analysis*. John Wiley & Sons, 2000.

[16] A. E. Scott, D. M. Vasilescu, K. A. D. Seal, S. D. Keyes, N. M. Mark, J. C. Hogg, I. Sinclair, J. A. Warner, T.-L. Hackett, and P. M. Lackie, "Three dimensional imaging of paraffin embedded human lung tissue samples by micro-computed tomography," *PLOS ONE*, vol. 10, no. 6, Jun. 2015.

[17] R. Norcen, M. Podesser, A. Pommer, H.-P. Schmidt, and A. Uhla, "Confidential storage and transmission of medical image data," *Comput. Biol. Med.*, vol. 33, no. 3, pp. 277–292, May 2003.

[18] P. Cosman, R. Gray, and R. Olshen, "Evaluating quality of compressed medical images: Snr, subjective rating, and diagnostic accuracy," in *Proc. of the IEEE*, vol. 82, no. 6. IEEE, Jun. 1994, pp. 919–932.

[19] D. Gehrke and E. Turban, "Determinants of successful website design: relative importance and recommendations for effectiveness," in *Proc. of the HICSS-32*, vol. 5. Maui, HI, USA: IEEE, Jan. 1999, p. 8.

[20] J. Prosise, "Wicked code: Taking Silverlight Deep Zoom to the next level," *MSDN Magazine*, vol. 24, no. 7, p. 90, Jul. 2009.

Bibliography

Abramoff, M. D., Magalhaes, P. J., Ram, S. J., 2004. 'Image processing with ImageJ'. *Biophotonics Int.* 11 (7), 36–42. doi:10.1117/1.3589100

Abrial, J., Apr. 1974. 'Data semantics'. *In: Proc. IFIP Work. Conf. Data Base Manag.* North Holland Pub. Co., Cargèse, France, pp. 1–60.

Adamanskiy, A., Denisov, A., Dec. 2013. 'EJDB—Embedded JSON database engine'. *In: 2013 Fourth World Congr. Softw. Eng.* IEEE, Hong Kong, China, pp. 161–164. doi:0.1109/SC.2014.40

Adams, F., Qiu, T., Mark, A., Fritz, B., Kramer, L., Schlager, D., Wetterauer, U., Miernik, A., Fischer, P., Apr. 2017. 'Soft 3D-printed phantom of the human kidney with collecting system'. *Ann. Biomed. Eng.* 45 (4), 963–972. doi:10.1007/s10439-016-1757-5

Adams, R., Bischof, L., Jun. 1994. 'Seeded region growing'. *IEEE Trans. Pattern Anal. Mach. Intell.* 16 (6), 641–647. doi:10.1109/34.295913

Agarwala, M., Bourell, D., Beaman, J., Marcus, H., Barlow, J., Mar. 1995. 'Direct selective laser sintering of metals'. *Rapid Prototyp. J.* 1 (1), 26–36. doi:10.1108/13552549510078113

Albrecht, T., Lüthi, M., Vetter, T., 2009. 'Deformable models'. *In: Li, S. Z., Jain, A. (Eds.), Encycl. Biometrics*, 1st Edition. Springer US, Boston, MA, USA, pp. 210–215. doi:10.1007/978-0-387-73003-5_88

Allan, C., Burel, J. M., Moore, J., Blackburn, C., Linkert, M., Loynton, S., Macdonald, D., Moore, W. J., Neves, C., Patterson, A., Porter, M., Tarkowska, A., Loranger, B., Avondo, J., Lagerstedt, I., Lianas, L., Leo, S., Hands, K., Hay, R. T., Patwardhan, A., Best, C., Kleywegt, G. J., Zanetti, G., Swedlow, J. R., 2012. 'OMERO: Flexible, model-driven data management for experimental biology'. *Nat. Methods* 9, 245–253. doi:10.1038/nmeth.1896

Amat, F., Höckendorf, B., Wan, Y., Lemon, W. C., McDole, K., Keller, P. J., Oct. 2015. 'Efficient processing and analysis of large-scale light-sheet microscopy data'. *Nat. Protoc.* 10 (11), 1679–1696. doi:10.1038/nprot.2015.111

Angenent, I., Pichon, E., Tannenbaum, A., 2006. 'Mathematical methods in medical image processing'. *Bull. New. Ser. Am. Math. Soc.* 43 (3), 365–396. doi:10.1090/S0273-0979-06-01104-9

Antonelli, M., Lazzerini, B., Marcelloni, F., Mar. 2005. 'Segmentation and reconstruction of the lung volume in CT images'. *In: 2005 ACM Symp. Appl. Comput. (SAC 2005).* ACM Press, Santa Fe, NM, USA, pp. 255–259. doi:10.1145/1066677.1066738

Appleton, B., Bradley, A. P., Wildermoth, M., Dec. 2005. 'Towards optimal image stitching for virtual microscopy'. *In: Proc. Digit. Image Comput. Tech. Appl. (DICTA '05)*. IEEE, Queensland, Australia, pp. 44–51. doi:10.1109/DICTA.2005.79

Arena, E. T., Rueden, C. T., Hiner, M. C., Wang, S., Yuan, M., Eliceiri, K. W., Dec. 2016. 'Quantitating the cell: turning images into numbers with ImageJ'. *Wiley Interdisciplinary Reviews: Developmental Biology* 6 (2), e260. doi:10.1002/wdev.260

Arganda-Carreras, I., Kaynig, V., Schindelin, J., Cardona, A., Seung, H. S., 2014. 'Trainable weka segmentation: A machine learning tool for microscopy image segmentation'. *In: Neurosci. 2014 Short Course 2 - Adv. Brain-scale, Autom. Anat. Tech. Neuronal Reconstr. Tract Tracing, Atlasing*. Society for Neuroscience, Washington, DC, USA, pp. 73–80.

Arguiñarena, E. J. C., Macchi, J. E., Escobar, P. P., del Fresno, M., Massa, J. M., Santiago, M. A., Sep. 2010. 'Dcm-Ar: A fast flash-based web-PACS viewer for displaying large DICOM images'. *In: Proc. 32nd Annu. Int. Conf. IEEE EMBS (EMBC 2010)*. IEEE, Buenos Aires, Argentina, pp. 3463–3466. doi:10.1109/IEMBS.2010.5627827

Armstrong, T. G., Ponnekanti, V., Borthakur, D., Callaghan, M., 2013. 'LinkBench: A database benchmark based on the Facebook social graph'. *In: Proc. 2013 ACM SIGMOD Int. Conf. Manag. Data (SIGMOD '13)*. ACM, New York, NY, USA, pp. 1185–1196. doi:10.1145/2463676.2465296

Azuma, R. T., Aug. 1997. 'A survey of augmented reality'. *Presence Teleoperators Virtual Environ.* 6 (4), 355–385. doi:10.1162/pres.1997.6.4.355

Bae, K. T., Kim, J.-S., Na, Y.-H., Kim, K. G., Kim, J.-H., Jul. 2005. 'Pulmonary nodules: Automated detection on CT images with morphologic matching algorithm—Preliminary results'. *Radiology* 236 (1). doi:10.1148/radiol.2361041286

Bakshi, K., Mar. 2012. 'Considerations for big data: Architecture and approach'. *In: Proc. IEEE Aerosp. Conf.* IEEE, Big Sky, MT, USA, pp. 1–7. doi:10.1109/AERO.2012.6187357

Bankhead, P., Loughrey, M. B., Fernández, J. A., Dombrowski, Y., McArt, D. G., Dunne, P. D., McQuaid, S., Gray, R. T., Murray, L. J., Coleman, H. G., James, J. A., Salto-Tellez, M., Hamilton, P. W., 12 2017. 'QuPath: Open source software for digital pathology image analysis'. *Scientific Reports* 7 (1). doi:10.1038/s41598-017-17204-5

Barten, P. G. J., 1999. *Contrast Sensitivity of the Human Eye and its Effects on Image Quality*. SPIE Press, Washington, DC, USA. ISBN: 0-8194-3496-5

Basharat, I., Azam, F., Muzaffar, A. W., 2012. 'Database security and encryption: A survey study'. *Int. J. Comput. Appl.* 47 (12), 28–34. doi:10.5120/7242-0218

BBC, Sep. 2016. '3D printed body parts to help trauma surgeons'. Online, accessed on 01/08/2018. URL http://www.bbc.co.uk/news/uk-england-nottinghamshire-37497182

BBSRC, Mar. 2017. 'BBSRC data sharing policy'. Tech. Rep. 1.22, BBSRC. URL https://bbsrc.ukri.org/documents/data-sharing-policy-pdf/

Beaman, J. J., Deckard, C. R., 1990. 'Selective laser sintering with assisted powder handling', The University of Texas System. Patent No, 4938816.

Bertino, E., Sandhu, R., Apr. 2005. 'Database security—Concepts, approaches, and challenges'. *IEEE Trans. Dependable Secur. Comput.* 2 (1), 2–19. doi:10.1109/TDSC.2005.9

Biglino, G., Capelli, C., Wray, J., Schievano, S., Leaver, L.-K., Khambadkone, S., Giardini, A., Derrick, G., Jones, A., Taylor, A. M., 2015. '3D-manufactured patient-specific models of congenital heart defects for communication in clinical practice: Feasibility and acceptability.'. *Br. Med. J. Open* 5 (4), e007165. doi:10.1136/bmjopen-2014-007165

Bikas, H., Stavropoulos, P., Chryssolouris, G., Mar. 2016. 'Additive manufacturing methods and modelling approaches: A critical review'. *Int. J. Adv. Manuf. Technol.* 83 (1–4), 389–405. doi:10.1007/s00170-015-7576-2

Blakstad, O., May 2011. 'Renaissance medicine'. Online, accessed on 14/03/2018. URL https://explorable.com/renaissance-medicine

Bledsoe, W. W., 1964. 'The model method of facial recognition'. Tech. rep., Panoramic Research Inc., California. URL https://archive.org/details/firstfacialrecognitionresearch

Bloem, J., Veninga, M., Shepherd, J., Mar. 1995. 'Fully automatic determination of soil bacterium numbers, cell volumes, and frequencies of dividing cells by confocal laser scanning microscopy and image analysis'. *Applied and Environmental Microbiology* 61, 926–936. URL https://aem.asm.org/content/aem/61/3/926.full.pdf

Blume, H. R., Muka, E., Apr. 1995. 'Hard copies for digital medical images: An overview'. *In: Proc. Color Hard Copy Graph. Arts IV*. Vol. 2413. SPIE Proceedings, San Jose, CA, USA, pp. 206–241. doi:10.1117/12.207579

Bray, B., Zeller, M., Mar. 2018. 'What is mixed reality?'. Online, accessed on 12/06/2018. URL https://docs.microsoft.com/en-us/windows/mixed-reality/mixed-reality

Breiman, L., Oct. 2001. 'Random forests'. *Machine Learning* 45 (1), 5–32. doi:10.1023/a:1010933404324

Brewer, E., Feb. 2012. 'CAP twelve years later: How the 'rules' have changed'. *Computer (Long. Beach. Calif).* 45 (2), 23–29. doi:10.1109/MC.2012.37

Bricken, W., Sep. 1990. 'Virtual reality: Directions of growth notes from the SIGGRAPH '90 panel'. Tech. rep. URL http://www.wbricken.com/pdfs/03words/02vr/01describe/03dir-of-growth.pdf

Brin, S., Page, L., Dec. 2012. 'Reprint of: The anatomy of a large-scale hypertextual web search engine'. *Comput. Networks* 56 (18), 3825–3833. doi:10.1016/j.comnet.2012.10.007

Brown, K. K., 2006. 'Chronic cough due to nonbronchiectatic suppurative airway disease (bronchiolitis): ACCP evidence-based clinical practice guidelines'. *Chest* 129 (1), 132S–137S. doi:10.1378/chest.129.1_suppl.132S

Brown, L. G., Dec. 1992. 'A survey of image registration techniques'. *ACM Comput. Surv.* 24 (4), 325–376. doi:10.1145/146370.146374

Brownlee, J., Jul. 2018. 'When to use mlp, cnn, and rnn neural networks'. Online, accessed on 15/11/2018. URL https://machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/

Bueno, J. M. J., Chino, F., Traina, A. A. J. M., Jr., C. T., Azevedo-Marques, P. M. P., Traina, C., Azevedo-Marques, P. M. P., Jun. 2002. 'How to add content-based image retrieval capability in a PACS'. *In: Proc. 15th IEEE Symp. Comput. Med. Syst. (CBMS 2002)*. Maribor, Slovenia, pp. 321–326. doi:10.1109/CBMS.2002.1011397

Burel, J.-M., Besson, S., Blackburn, C., Carroll, M., Ferguson, R. K., Flynn, H., Gillen, K., Leigh, R., Li, S., Lindner, D., Linkert, M., Moore, W. J., Ramalingam, B., Rozbicki, E., Tarkowska, A., Walczysko, P., Allan, C., Moore, J., Swedlow, J. R., 7 2015. 'Publishing and sharing multi-dimensional image data with OMERO'. *Mammalian Genome* 26 (9-10), 441–447. doi:10.1007/s00335-015-9587-6

Cantrell, J., Rohde, S., Damiani, D., Gurnani, R., DiSandro, L., Anton, J., Young, A., Jerez, A., Steinbach, D., Kroese, C., Ifju, P., Jun. 2016. 'Experimental characterization of the mechanical properties of 3D-printed ABS and polycarbonate parts'. *In: Adv. Opt. Methods Exp. Mech. – Proc. 2016 Annu. Conf. Exp. Appl. Mech.* Vol. 3. Springer, Cham, Orlando, FL, USA, pp. 89–105. doi:10.1007/978-3-319-41600-7_11

Castiglione, A., Pizzolante, R., De Santis, A., Carpentieri, B., Castiglione, A., Palmieri, F., Feb. 2015. 'Cloud-based adaptive compression and secure management services for 3D healthcare data'. *Futur. Gener. Comput. Syst.* 43-44, 120–134. doi:10.1016/j.future.2014.07.001

Celesti, A., Maria, F., Romano, A., Bramanti, A., Bramanti, P., Villari, M., 2017. 'An OAIS-based hospital information system on the cloud: Analysis of a NoSQL column-oriented approach'. *IEEE J. Biomed. Heal. Informatics*. doi:10.1109/JBHI.2017.2681126

Cellan-Jones, R., May 2017. 'Ransomware and the NHS—The inquest begins'. Online, accessed on 01/08/2018. URL http://www.bbc.co.uk/news/technology-39917278

Chahal, D., Kharb, L., Gupta, M., Sep. 2017. 'Challenges and security issues of NoSQL databases'. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.* 2 (5), 976–982. URL http://ijsrcseit.com/paper/CSEIT1725215.pdf

Chenebaux, M., Lescanne, E., Robier, A., Kim, S., Bakhos, D., Jul. 2014. 'Evaluation of a temporal bone prototype by experts in otology'. *J. Laryngol. Otol.* 128 (7), 586–590. doi:10.1017/S0022215114001297

Chlipala, E., Elin, J., Eichhorn, O., Krishnamurti, M., Long, R. E., Sabata, B., Smith, M., oct 2010. 'Archival and retrieval in digital pathology systems'. Tech. rep., Digital Pathology Association (DPA), Madison WI. URL https://digitalpathologyassociation.org/_data/files/Archival_and_Retrieval_in_Digital_Pathology_Systems.pdf

Choplin, R. H., Boehme, J. M., Maynard, C. D., Jan. 1992. 'Picture archiving and communication systems: An overview'. *RadioGraphics* 12 (1), 127–129. doi:10.1148/radiographics.12.1.1734458

Chui, C. K., Wang, H., Jun. 2005. 'System and method for tiled multiresolution encoding/decoding and communication with lossless selective regions of interest via data reuse', LightSurf Technologies, Inc. Patent No, 6904176.

Clunie, D. A., Feb. 2000. 'Lossless compression of grayscale medical images: Effectiveness of traditional and state-of-the-art approaches'. *In: Blaine, G. J., Siegel, E. L. (Eds.), Proc. Med. Imaging 2000*. Vol. 3980. SPIE, San Diego, CA, USA, pp. 74–84. doi:10.1117/12.386389

Compton, K., Oosterwijk, H., 2012. 'Requirements for medical imaging monitors'. Tech. rep., OTech Inc. URL http://www.otechimg.com/publications/pdf/wp_medical_image_monitors.pdf

Cooper, B. F., Silberstein, A., Tam, E., Ramakrishnan, R., Sears, R., Jun. 2010. 'Benchmarking cloud serving systems with YCSB'. *In: Proc. 1st ACM Symp. Cloud Comput. (SOCC '10)*. ACM, Indianapolis, IN, USA, pp. 143–154. doi:10.1145/1807128.1807152

Cornelisse, D., Apr. 2018. 'An intuitive guide to convolutional neural networks'. Online, accessed on 20/12/2018. URL https://medium.freecodecamp.org/an-intuitive-guide-to-convolutional-neural-networks-260c2de0a050

Cosman, P. C., Gray, R. M., Olshen, R. A., Jun. 1994. 'Evaluating quality of compressed medical images: SNR, subjective rating, and diagnostic accuracy' 82 (6), 919–932. doi:10.1109/5.286196

Crichton, C., Davies, J., Gibbons, J., Tsui, A., Brenton, J., Caldas, C., Morris, L., May 2008. 'Deep Zoom and touch screen for tissue microarray image scoring'. *In: Proc. ICSE Work. Softw. Eng. Heal. Care (ISCE 2008)*. Vol. 70. ACM Press, Leipzig, Germany, pp. 217–243.

Crump, S. S., 1992. 'Apparatus and method for creating three-dimensional objects', Stratasys Inc. Patent No, 5121329.

Cuingnet, R., Prevost, R., Lesage, D., Cohen, L. D., Mory, B., Ardon, R., Oct. 2012. 'Automatic detection and segmentation of kidneys in 3D CT images using random forests'. *In: Ayache, N., Delingette, H., Golland, P., Mori, K. (Eds.), Med. Image Comput. Comput. Interv. (MICCAI 2012)*. Springer, Nice, France, pp. 66–74. doi:10.1007/978-3-642-33454-2_9

Cupitt, J., Martinez, K., 2 1996. 'VIPS: An image processing system for large images'. *In: Algazi, V. R., Ono, S., Tescher, A. G. (Eds.), Very High Resolution and Quality Imaging*. Vol. 2663. SPIE. doi:10.1117/12.233043

Das, A. K., Musen, M. A., 1994. 'A temporal query system for protocol-directed decision support.'. *Methods Inf. Med.* 33 (4), 358–370. doi:10.1055/s-0038-1635036

Dawson, J. K., Graham, D. R., Desmond, J., Fewins, H. E., Lynch, M. P., 2002. 'Investigation of the chronic pulmonary effects of low-dose oral methotrexate in patients with rheumatoid arthritis: A prospective study incorporating HRCT scanning and pulmonary function tests'. *Rheumatology* 41 (3), 262–267. doi:10.1093/rheumatology/41.3.262

DB-Engines, 2018. 'DB-Engines ranking'. Online, accessed on 21/08/2018. URL https://db-engines.com/en/ranking

Deckard, C. R., 1988. 'Method and apparatus for producing parts by selective sintering', The University of Texas System. Patent No, WO/1988/002677.

Deng, Y., Zhang, T., Nov. 2003. 'Generating panorama photos'. *In: Smith, J. R., Panchanathan, S., Zhang, T. (Eds.), Proc. SPIE 5242, Internet Multimed. Manag. Syst. IV*. SPIE, Orlando, FL, USA, pp. 270–279. doi:10.1117/12.513119

Dessloch, S., Mattos, N., Aug. 1997. 'Integrating SQL databases with content-specific search engines'. *In: Proc. 23rd Int. Conf. Very Large Data Bases (VLDB '97)*. Morgan Kaufmann Publishers Inc., Athens, Greece, pp. 528–536.

Díaz-Galiano, M., Martín-Valdivia, M., Ureña-López, L., Apr. 2009. 'Query expansion with a medical ontology to improve a multimodal information retrieval system'. *Comput. Biol. Med.* 39 (4), 396–403. doi:10.1016/j.compbiomed.2009.01.012

Dougherty, G. (Ed.), 2011. *Medical Image Processing: Techniques and Application*. Springer-Verlag, New York, NY, USA. ISBN: 978-1-4419-9769-2

Duckworth, E. A., Silva, F. E., Chandler, J. P., Batjer, H. H., Zhao, J.-c., Jan. 2008. 'Temporal bone dissection for neurosurgery residents: Identifying the essential concepts and fundamental techniques for success'. *Surg. Neurol.* 69 (1), 93–98. doi:10.1016/j.surneu.2007.07.054

Eini, O., May 2010. 'That no SQL thing: Column (family) databases'. Online, accessed on 01/08/2018. URL http://ayende.com/blog/4500/that-no-sql-thing-column-family-databases

Eliceiri, K. W., Berthold, M. R., Goldberg, I. G., Ibáñez, L., Manjunath, B. S., Martone, M. E., Murphy, R. F., Peng, H. C., Plant, A. L., Roysam, B., Stuurman, N., Swedlow, J. R., Tomancak, P., Carpenter, A. E., 2012. 'Biological imaging software tools'. *Nat. Methods* 9 (7), 697–710. doi:10.1038/nmeth.2084

Elmasri, R., Navathe, S. B. (Eds.), 2011. *Database Systems*, 6th Edition. Addison-Wesley, Boston, MA, USA. ISBN: 978-0-136-08620-8

Engel, K., Ertl, T., 1999. 'Texture-based volume visualization for multiple users on the world wide web'. *In: Gervautz, M., Schmalstieg, D., Hildebrand, A. (Eds.), Virtual Environ. '99.* Eurographics. Springer, Vienna, pp. 115–124. doi:10.1007/978-3-7091-6805-9_12

Ephox, 2017. 'Plupload: Multi-runtime file-uploader'. Online, accessed on 01/08/2018. URL https://www.plupload.com/docs/v2/Frequently-Asked-Questions

EPSRC, Oct. 2004. 'Clarifications of EPSRC expectations on research data management'. Tech. rep., EPSRC. URL https://epsrc.ukri.org/files/aboutus/standards/clarificationsofexpectationsresearchdatamanagement/

Ercan, M. Z., Lane, M., Dec. 2014. 'An evaluation of NoSQL databases for EHR systems'. *In: Proc. 25th Australas. Conf. Inf. Syst.* ACIS, Auckland, New Zealand. URL https://eprints.usq.edu.au/26225/

Evans, A., Romeo, M., Bahrehmand, A., Agenjo, J., Blat, J., Jun. 2014. '3D graphics on the web: A survey'. *Comput. Graph.* 41, 43–61. doi:10.1016/j.cag.2014.02.002

Fabian, B., Ermakova, T., Junghanns, P., Mar. 2015. 'Collaborative and secure sharing of healthcare data in multi-clouds'. *Inf. Syst.* 48, 132–150. doi:10.1016/j.is.2014.05.004

Feygin, M., 1987. 'Apparatus and method for forming an integral object from laminations', CUBIC TECHNOLOGIES Inc. Patent No, WO/1987/007538.

Fialka, O., Čadík, M., 2006. 'FFT and convolution performance in image filtering on GPU'. *In: Tenth International Conference on Information Visualisation (IV'06).* IEEE, pp. 609–614. doi:10.1109/iv.2006.53

Fox, A., Brewer, E. A., Mar. 1999. 'Harvest, yield, and scalable tolerant systems'. *In: Proc. 7th Work. Hot Top. Oper. Syst.* IEEE, Rio Rico, AZ, USA, pp. 174–178. doi:10.1109/HOTOS.1999.798396

Fox-Brewster, T., May 2017. 'Medical devices hit by ransomware for the first time in US hospitals'. Online, accessed on 01/08/2018. URL https://www.forbes.com/sites/thomasbrewster/2017/05/17/wannacry-ransomware-hit-real-medical-devices/#741ce643425c

Freeman, D., Reeve, S., Robinson, A., Ehlers, A., Clark, D., Spanlang, B., Slater, M., Oct. 2017. 'Virtual reality in the assessment, understanding, and treatment of mental health disorders'. *Psychol. Med.* 47 (14), 2393–2400. doi:10.1017/S003329171700040X

Fuchs, H., Livingston, M. A., Raskar, R., Colucci, D., Keller, K., State, A., Crawford, J. R., Rademacher, P., Drake, S. H., Meyer, A. A., 1998. 'Augmented reality visualization for laparoscopic surgery'. *In: Wells, W. M., Colchester, A., Delp, S. (Eds.), Med. Image Comput. Comput. Interv.* Springer, Berlin, Heidelberg, pp. 934–943. doi:10.1007/BFb0056282

Fumo, D., Jun. 2017. 'Types of machine learning algorithms you should know'. Online, accessed on 15/11/2018. URL https://towardsdatascience.com/types-of-machine-learning-algorithms-you-should-know-953a08248861

GE Healthcare, 2017. 'Revolution CT'. Online, accessed on 02/08/2018. URL http://www3.gehealthcare.com/en/products/categories/computed_tomography/revolution_ct#tabs/tab8CB733F2821249EDA32FB25F78A47A70

Gehrke, D., Turban, E., Jan. 1999. 'Determinants of successful website design: Relative importance and recommendations for effectiveness'. *In: Proc. 32nd Annu. Hawaii Int. Conf. Syst. Sci. (HICSS-32).* Vol. 5. IEEE, Maui, HI, USA, p. 8. doi:10.1109/HICSS.1999.772943

Genereaux, B. W., Dennison, D. K., Ho, K., Horn, R., Silver, E. L., O'Donnell, K., Kahn, C. E., Jun. 2018. 'DICOMweb™: Background and application of the web standard for medical imaging'. *J. Digit. Imaging* 31 (3), 321–326. doi:10.1007/s10278-018-0073-z

George, A. P., De, R., Feb. 2010. 'Review of temporal bone dissection teaching: How it was, is and will be'. *J. Laryngol. Otol.* 124 (2), 119–125. doi:10.1017/S0022215109991617

Gerhard, L., Szoforan, A., Nickolov, R., Sep. 2010. 'Scalable mutable tiled multi-resolution texture atlases', Microsoft Corporation. Patent No, 20100226593.

Ghabayen, A. S., Noah, S. A., 2014. 'Exploiting social tags to overcome cold start recommendation problem'. *J. Comput. Sci.* 10 (7), 1166–1173. doi:10.3844/jcssp.2014.1166.1173

Gibson, I., Rosen, D. W., Stucker, B., 2010. 'Photopolymerization processes'. *In: Addit. Manuf. Technol.* Springer US, Boston, MA, pp. 78–119. doi:10.1007/978-1-4419-1120-9_4

Gobbetti, E., Marton, F., Jul. 2005. 'Far voxels: A multiresolution framework for interactive rendering of huge complex 3D models on commodity graphics platforms'. *ACM Trans. Graph. - Proc. ACM SIGGRAPH 2005* 24 (3), 878–885. doi:10.1145/1073204.1073277

Golder, S. A., Huberman, B. A., 2006. 'Usage patterns of collaborative tagging systems'. *J. Inf. Sci.* 32 (2), 198–208. doi:10.1177/0165551506062337

Golio, M., Oct. 2015. 'Fifty years of Moore's law'. *Proc. IEEE* 103 (10), 1937.

Gonzalez, R. C., Woods, R. E., 2002. *Digital Image Processing*, 2nd Edition. Prentice Hall, Upper Saddle River, NJ, USA. ISBN: 0-201-18075-8

Gormish, M. J., Schwartz, E. L., Keith, A., Boliek, M., Zandi, A., Apr. 1997. 'Lossless and nearly lossless compression for high-quality images'. *In: Very High Resolut. Qual. Imaging II (Electronic Imaging '97)*. Vol. 3025. SPIE, San Jose, CA, USA, pp. 62–70. doi:10.1117/12.270058

Grammatikopoulos, A., Banks, J., Temarel, P., Jun. 2018. 'Experimental dynamic properties of ABS cellular beams produced using additive manufacturing'. *In: 18th Eur. Conf. Compos. Mater. (ECCM 18)*. ESCM, Athens, Greece, (in press).

Greulich, M., Greul, M., Pintat, T., Mar. 1995. 'Fast, functional prototypes via multiphase jet solidification'. *Rapid Prototyp. J.* 1 (1), 20–25. doi:10.1108/13552549510146649

Griffin, J., Treanor, D., Jan. 2017. 'Digital pathology in clinical use: Where are we now and what is holding us back?'. *Histopathology* 70 (1), 134–145. doi:10.1111/his.12993

Grigorik, I., Jul. 2018. 'HTTP caching'. Online, accessed on 19/07/2018. URL https://developers.google.com/web/fundamentals/performance/optimizing-content-efficiency/http-caching

Gu, D., 2015. 'Laser additive manufacturing (AM): Classification, processing philosophy, and metallurgical mechanisms'. *In: Laser Addit. Manuf. High-Performance Mater.* Springer, Berlin, Heidelberg, Ch. 2, pp. 15–71. doi:10.1007/978-3-662-46089-4_2

Gupta, A., 2009. *Data Provenance*. Springer. ISBN: 978-0-387-35544-3

Hadjigeorgiou, C., Aug. 2013. 'RDBMS vs NoSQL: Performance and scaling comparison'. MSc Thesis, The University of Edinburgh.

Haggerty, M., 1991. 'Virtual reality dominates SIGGRAPH [selective update]'. *IEEE Comput. Graph. Appl.* 11 (5), 14–17. doi:10.1109/MCG.1991.6210989

Halvey, M. J., Keane, M. T., 2007. 'An assessment of tag presentation techniques'. *In: Proc. 16th Int. Conf. World Wide Web (WWW '07)*. Association for Computing Machinery (ACM), New York, NY, USA, pp. 1313–1314. doi:10.1145/1242572.1242826

Hammond, T., Hannay, T., Lund, B., Scott, J., Apr. 2005. 'Social bookmarking tools'. *D-Lib Mag.* 11 (4). URL http://www.dlib.org/dlib/april05/hammond/04hammond.html

Han, J., Haihong, E., Le, G., Du, L., Oct. 2011. 'Survey on NoSQL database'. *In: Proc. 6th Int. Conf. Pervasive Comput. Appl. (ICPCA/SWS 2011)*. IEEE, Port Elizabeth, South Africa, pp. 363–366. doi:10.1109/ICPCA.2011.6106531

Hansen, C. D., Johnson, C. R. (Eds.), 2004. *The Visualization Handbook*, 1st Edition. Elsevier Butterworth-Heinemann, Burlington, MA, USA. ISBN: 978-0-12-387582-2

Hartmann, A. Ø., Tjellesen, F. W., 2012. 'Three-dimensional printer', BLUEPRINTER APS. Patent No, 20120201960.

He, L., Long, L. R., Antani, S., Thoma, G. R., 2011. 'Computer assisted diagnosis in histopathology'. *In: Zhao, Z. (Ed.), Seq. Genome Anal. Methods Appl.* Vol. 3. iConcept Press, Hong Kong, Ch. 15, pp. 271–287.

He, N., Zhang, X., Zhao, J., Zhao, H., Qiang, Y., 2017. 'Pulmonary parenchyma segmentation in thin CT image sequences with spectral clustering and geodesic active contour model based on similarity'. *In: 9th Int. Conf. on Digit. Image Process. (ICDIP 2017)*. Vol. 10420. SPIE. doi:10.1117/12.2281942

Higgins, S., Dec. 2008. 'The DCC curation lifecycle model'. *Int. J. Digit. Curation* 3 (1), 134–140. doi:10.2218/ijdc.v3i1.48

Hipp, J., Fernandez, A., Compton, C., Balis, U., 2011. 'Why a pathology image should not be considered as a radiology image'. *J. Pathol. Inform.* 2 (1), 26. doi:10.4103/2153-3539.82051

Hochman, J. B., Kraut, J., Kazmerik, K., Unger, B. J., Feb. 2014. 'Generation of a 3D printed temporal bone model with internal fidelity and validation of the mechanical construct'. *Otolaryngol. - Head Neck Surg.* 150 (3), 448–454. doi:10.1177/0194599813518008

Hong, Z.-Q., Jan. 1991. 'Algebraic feature extraction of image for recognition'. *Pattern Recognit.* 24 (3), 211–219. doi:10.1016/0031-3203(91)90063-B

Hounsfield, G. N., Dec. 1973. 'Computerized transverse axial scanning (tomography): Part 1. description of system'. *Br. J. Radiol.* 46 (552), 1016–1022. doi:10.1259/0007-1285-46-552-1016

Hsia, C. C. W., Hyde, D. M., Ochs, M., Weibel, E. R., 2010a. 'How to measure lung structure—What for? On the 'standards for the quantitative assessment of lung structure''. *Respir. Physiol. Neurobiol.* 171, 72–74. doi:10.1016/j.resp.2010.02.016

Hsia, C. C. W., Hyde, D. M., Ochs, M., Weibel, E. R., 2010b. 'An official research policy statement of the American Thoracic Society/European Respiratory Society: Standards for quantitative assessment of lung structure'. *Am. J. Respir. Crit. Care Med.* 181 (4), 394–418. doi:10.1164/rccm.200809-1522ST

Hu, S., Hoffman, E. A., Reinhardt, J. M., 2001. 'Automatic lung segmentation for accurate quantitation of volumetric X-ray CT images'. *IEEE Trans. Med. Imaging* 20 (6), 490–498. doi:10.1109/42.929615

Huang, S. H., Liu, P., Mokasdar, A., Hou, L., Jul. 2013. 'Additive manufacturing and its societal impact: A literature review'. *Int. J. Adv. Manuf. Technol.* 67 (5-8), 1191–1203. doi:10.1007/s00170-012-4558-5

Huff, S. M., Berthelsen, C. L., Pryor, T. A., Dudley, A. S., Nov. 1991. 'Evaluation of an SQL model of the HELP patient database.'. *In: Proc. 15th Annu. Symp. Comput. Appl. Med. Care.* AMIA, Washington, DC, USA, pp. 386–390. URL https://www.ncbi.nlm.nih.gov/pubmed/1807629

Hull, C. W., 1986. 'Apparatus for production of three-dimensional objects by stereolithography', UVP Inc. Patent No, 4575330.

Husz, Z. L., Burton, N., Hill, B., Milyaev, N., Baldock, R. A., 2012. 'Web tools for large-scale 3d biological images and atlases'. *BMC Bioinformatics* 13. doi:10.1186/1471-2105-13-122

Işık, V., Apr. 2014. 'Classification of holograms and types of hologram used in holographic art'. *In: Proc. 3rd Int. Conf. Commun. Media, Technol. Des. (ICCMTD 2014)*. Istanbul, Turkey, pp. 35–41. URL http://www.cmdconf.net/2014/pdf/6.pdf

IMPERVA, 2015. 'Top ten database security threats—The most significant risks of 2015 and how to mitigate them'. Tech. rep., IMPERVA. URL http://www.imperva.com/docs/wp_topten_database_threats.pdf

Independent Security Evaluators, 2016. 'Securing hospitals a research study and blueprint'. Tech. rep., ISE. URL https://securityevaluators.com/hospitalhack/securing_hospitals.pdf

Jagadish, H. V., Olken, F., Jun. 2004. 'Database management for life sciences research'. *ACM SIGMOD Rec.* 33 (2), 15–20. doi:10.1145/1024694.1024697

Jeong, W.-K. K., Schneider, J., Turney, S. G., Faulkner-Jones, B. E., Meyer, D., Westermann, R., Reid, R. C., Lichtman, J., Pfister, H., Oct. 2010. 'Interactive histology and of large-scale and biomedical image and stacks' 16 (6), 1386–1395. doi:10.1109/TVCG.2010.168

Johnson, P. T., Zimmerman, S. L., Heath, D., Eng, J., Horton, K. M., Scott, W. W., Fishman, E. K., Aug. 2012. 'The iPad as a mobile device for CT display and interpretation: Diagnostic accuracy for identification of pulmonary embolism'. *Emerg. Radiol.* 19 (4), 323–327. doi:10.1007/s10140-012-1037-0

Jones, D. M., Jun. 2006. 'Utilization of DICOM GSDF to modify lookup tables for images acquired on film digitizers'. *J. Digit. Imaging* 19 (2), 167–171. doi:10.1007/s10278-005-9241-z

Kagawa, T., Fukunari, F., Shiraishi, T., Yamasaki, M., Ichihara, T., Kihara, Y., Zeze, R., Nogami, K., Yuasa, K., Dec. 2006. 'Development of a simple image viewer designed for small X-ray field CT equipment 3DX'. *Oral Radiol.* 22 (2), 47–51. doi:10.1007/s11282-006-0046-7

Kalinić, H., Nov. 2009. 'Atlas-based image segmentation: A survey'. Tech. rep., Universiy of Zagreb, Zagreb, Croatia, accessed on 13/07/2018. URL https://bib.irb.hr/datoteka/435355.jnrl.pdf

Karaliotas, C., Dec. 2011. 'When simulation in surgical training meets virtual reality'. *Hell. J. Surg.* 83 (6), 303–316. doi:10.1007/s13126-011-0055-9

Karch, M., Oct. 2016. 'What is folksonomy?'. Online, accessed on 03/08/2018. URL https://www.lifewire.com/what-is-folksonomy-1616321

Kass, M., Witkin, A., Terzopoulos, D., 1988. 'Snakes: Active contour models'. *Int. J. Comput. Vision*, 321–331. URL http://www.cs.ait.ac.th/~mdailey/cvreadings/Kass-Snakes.pdf

Kaynig, V., Fuchs, T., Buhmann, J. M., Jun. 2010. 'Neuron geometry extraction by perceptual grouping in ssTEM images'. *In: Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* IEEE, San Francisco, CA, USA, pp. 2902–2909. doi:10.1109/CVPR.2010.5540029

Kemerink, M., Dierichs, T. J., Dierichs, J., Huynen, H. J., Wildberger, J. E., van Engelshoven, J. M. A., Kemerink, G. J., May 2011. 'Characteristics of a first-generation X-ray system'. *Radiology* 259 (2), 534–539. doi:10.1148/radiol.11101899

Kießling, W., Köstler, G., Aug. 2002. 'Preference SQL: Design, implementation, experiences'. *In: Proc. 28th Int. Conf. Very Large Data Bases (VLDB '02)*. VLDB Endowment, Hong Kong, China, pp. 990–1001. URL https://dl.acm.org/citation.cfm?id=1287457

Kimpe, T., Tuytschaever, T., Dec. 2007. 'Increasing the number of gray shades in medical display systems—How much is enough?'. *J. Digit. Imaging* 20 (4), 422–432. doi:10.1007/s10278-006-1052-3

Ko, J. P., Bargo, L. J., Broderick, L. S., Hunsaker, A., Chung, J. H., Lynch, D., Stern, E. J., 2015. *ACR-STR Practice Parameter for the Performance of High-Resolution Computed Tomography (HRCT) of the Lungs in Adults*. The American College of Radiology. URL http://www.acr.org/quality-safety/standards-guidelines/practice-guidelines-by-modality/ct

Kodama, H., Nov. 1981. 'Automatic method for fabricating a three-dimensional plastic model with photo-hardening polymer'. *Rev. Sci. Instrum.* 52 (11), 1770–1773. doi:10.1063/1.1136492

Korenblum, D., Rubin, D., Napel, S., Rodriguez, C., Beaulieu, C., Aug. 2011. 'Managing biomedical image metadata for search and retrieval of similar images'. *J. Digit. Imaging* 24 (4), 739–748. doi:10.1007/s10278-010-9328-z

Kubacki, W. M., Jun. 2010. 'SQL vs. "NoSQL"'. Tech. rep., Uni Hannover. URL http://www.se.uni-hannover.de/priv/lehre_2010sommer_wwwseminar/11-SQL_vs_NoSQL-Ausarbeitung.pdf

Kumar, A., Shaik, F., 2016. 'Importance of image processing'. *In: Image Process. Diabet. Relat. Causes*. Forensic and Medical Bioinformatics. Springer, Singapore, Ch. 2, pp. 5–7. doi:10.1007/978-981-287-624-9_2

Kuo, B. Y.-L., Hentrich, T., Good, B. M., Wilkinson, M. D., May 2007. 'Tag clouds for summarizing web search results'. *In: Proc. 16th Int. Conf. World Wide Web (WWW '07)*. ACM Press, Banff, Canada, pp. 1203–1204. doi:10.1145/1242572.1242766

Kvilekval, K., 2007. 'BISQUIK internals'. Online, accessed on 03/08/2018. URL http://biodev.ece.ucsb.edu/projects/bisquik/raw-attachment/wiki/WikiStart/bisquik-internals.ppt

Kvilekval, K., Fedorov, D., Obara, B., Singh, A., Manjunath, B. S., Feb. 2010. 'BisQue: A platform for bioimage analysis and management'. *Bioinformatics* 26 (4), 544–552. doi:10.1093/bioinformatics/btp699

Laal, M., Jun. 2013. 'Innovation process in medical imaging'. *Procedia - Soc. Behav. Sci.* 81, 60–64. doi:10.1016/j.sbspro.2013.06.388

Larson, R., 1998. 'Method and device for producing three-dimensional bodies', Arcam Ltd. Patent No, 5786562.

Lawrence, E., Mar. 2011. 'File upload and download limits'. Online, accessed on 30/07/2018. URL https://blogs.msdn.microsoft.com/ieinternals/2011/03/10/file-upload-and-download-limits/

Lawrence, S., Giles, C. L., Tsoi, A. C., Back, A. D., 1997. 'Face recognition: a convolutional neural-network approach'. *IEEE Transactions on Neural Networks* 8 (1), 98–113. doi:10.1109/72.554195

Lee, K. K.-Y., Tang, W.-C., Choi, K.-S., Apr. 2013. 'Alternatives to relational database: Comparison of NoSQL and XML approaches for clinical data storage'. *Comput. Methods Programs Biomed.* 110 (10), 99–109. doi:10.1016/j.cmpb.2012.10.018

Levoy, M., Ginsberg, J., Shade, J., Fulk, D., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Jul. 2000. 'The digital Michelangelo project: 3D scanning of large statues'. *In: Proc. 27th Annu. Conf. Comput. Graph. Interact. Tech. (SIGGRAPH '00)*. ACM Press, New Orleans, LA, USA, pp. 131–144. doi:10.1145/344779.344849

Li, W., Nov. 1992. 'Random texts exhibit Zipf's-law-like word frequency distribution'. *IEEE Trans. Inf. Theory* 38 (6), 1842–1845. doi:10.1109/18.165464

Lim, S.-J., Jeong, Y.-Y., Lee, C.-W., Ho, Y.-S., 2004. 'Automatic segmentation of the liver in CT images using the watershed algorithm based on morphological filtering'. *In: Proc. Med. Imaging 2004: Image Process.* Vol. 5370. SPIE Proceedings, San Diego, CA, USA, pp. 1658–1666. doi:10.117/12.533586

Linkert, M., Rueden, C. T., Allan, C., Burel, J.-M. J.-M., Moore, W., Patterson, A., Loranger, B., Moore, J., Neves, C., MacDonald, D., Tarkowska, A., Sticco, C., Hill, E., Rossner, M., Eliceiri, K. W., Swedlow, J. R., 2010. 'Metadata matters: Access to image data in the real world'. *J. Cell Biol.* 189 (5), 777–782. doi:10.1083/jcb.201004104

Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., van der Laak, J. A., van Ginneken, B., Sánchez, C. I., 2017. 'A survey on deep learning in medical image analysis'. *Med. Image Anal.* 42, 60–88. doi:10.1016/j.media.2017.07.005

Liu, S., Lin, G., Chai, B., Dai, J., Zhou, A., Chen, F., Pan, J., Dec. 2017. 'The future of graph database applications: An electric utility perspective'. *In: 2017 IEEE 2nd Inf. Technol. Networking, Electron. Autom. Control Conf.* IEEE, pp. 223–227. doi:10.1109/ITNEC.2017.8284941

Longfield, E. A., Brickman, T. M., Jeyakumar, A., Jun. 2015. '3D printed pediatric temporal bone: A novel training model'. *Otol. Neurotol.* 36 (5), 793–795. doi:10.1097/MAO.0000000000000750

Maintz, J. B. A., Viergever, M. A., Mar. 1998. 'A survey of medical image registration'. *Med. Image Anal.* 2 (1), 1–36. doi:10.1016/S1361-8415(01)80026-8

Malik, M., Patel, T., Mar. 2016. 'Database security—Attacks and control methods'. *Int. J. Inf. Sci. Tech.* 6 (1/2), 175–183. doi:10.5121/ijist.2016.6218

Marcus, D. S., Olsen, T. R., Ramaratnam, M., Buckner, R. L., Mar. 2007. 'The extensible neuroimaging archive toolkit'. *Neuroinformatics* 5 (1), 11–33. doi:10.1385/NI:5:1:11

Mariani, J. A., 1992. 'Oggetto: An object oriented database layered on a triple store'. *Comput. J.* 35 (2), 108–118. doi:10.1093/comjnl/35.2.108

Mathes, A., Dec. 2004. 'Folksonomies—Cooperative classification and communication through shared metadata'. Tech. rep., University of Illinois Urbana-Champaign. URL http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html

Maurer, R. D., Schultz, P. C., 1972. 'Fused silica optical waveguide', Corning Inc. Patent No, 3659915.

Miano, J., 1999. *Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP*. Addison-Wesley Professional. ISBN: 0-201-60443-4

Microsoft Corporation, Mar. 2018. '[MS-FSA]: File System Algorithms'. Tech. rep., Microsoft. URL https://winprotocoldoc.blob.core.windows.net/productionwindowsarchives/MS-FSA/[MS-FSA].pdf

Mikula, S., Trotts, I., Stone, J. M., Jones, E. G., 2007. 'Internet-enabled high-resolution brain mappingand virtual microscopy'. *Neuroimage* 35 (1), 9–15. doi:10.1016/j.neuroimage.2006.11.053

Milgram, P., Kishino, F., Dec. 1994. 'A taxonomy of mixed reality visual displays'. *IEICE Trans. Inf. Syst.* E77-D (12), 1321–1329.

Mitzner, W., Weibel, E. R., 2010. 'Standards for quantitative assessment of lung structure'. *J. Appl. Physiol.* 109 (3), 934. doi:10.1152/japplphysiol.00228.2010

Montagnat, J., Bellet, F., Benoit-Cattin, H., Breton, V., Brunie, L., Duque, H., Legré, Y., Magnin, I. E., Maigne, L., Miguet, S., Pierson, J. M., Seitz, L., Tweed, T., 2004. 'Medical images simulation, storage, and processing on the european datagrid testbed'. *J. Grid Comput.* 2 (4), 387–400. doi:10.1007/s10723-004-5744-y

Moore, G. E., Apr. 1965. 'Cramming more components onto integrated circuits'. *Electronics* 38 (8), 114–117.

Müller, H., Michoux, N., Bandon, D., Geissbuhler, A., Feb. 2004. 'A review of content-based image retrieval systems in medical applications—Clinical benefits and future directions'. *Int. J. Med. Inform.* 73 (1), 1–23. doi:10.1016/j.ijmedinf.2003.11.024

Mullins, C. S., Dec. 2017. 'The pros and cons of database scaling options'. Tech. rep., Mullins Consulting, Inc. URL http://go.nuodb.com/rs/139-YPK-485/images/Database-Scaling-Options.pdf

Nance, C., Losser, T., Iype, R., Harmon, G., Mar. 2013. 'NoSQL vs RDBMS—Why there is room for both'. *In: Proc. South. Assoc. Inf. Syst. Conf. (SAIS 2013)*. AIS, Savannah, GA, USA, pp. 111–116. URL http://aisel.aisnet.org/sais2013/27/

NEMA, 2014a. *Digital Imaging and Communications in Medicine (DICOM) Standard*. No. 2014c. NEMA Publications, Rosslyn, VA, USA. URL ftp://medical.nema.org/medical/dicom/2014c/

NEMA, 2014b. *Grayscale Standard Display Function*. Vol. PS3.14 of NEMA (2014a). URL http://dicom.nema.org/medical/dicom/2014c/output/pdf/part14.pdf

NEMA, 2014c. *Information Object Definitions*. Vol. PS3.3 of NEMA (2014a). URL http://dicom.nema.org/medical/dicom/2014c/output/pdf/part03.pdf

NEMA, 2014d. *Introduction and Overview*. Vol. PS3.1 of NEMA (2014a). URL http://dicom.nema.org/medical/dicom/2014c/output/pdf/part01.pdf

NEMA, 2014e. *Network Communication Support for Message Exchange*. Vol. PS3.8 of NEMA (2014a). URL http://dicom.nema.org/medical/dicom/2014c/output/pdf/part08.pdf

Netter, F. H., 2014. *Atlas of Human Anatomy*, 6th Edition. Elsevier Health Sciences. Saunders Elsevier. ISBN: 978-1-4557-0418-7

New, P. F. J., Scott, W. R., Schnur, J. A., Davis, K. R., Taveras, J. M., Jan. 1974. 'Computerized axial tomography with the EMI scanner'. *Radiology* 110 (1), 109–123. doi:10.1148/110.1.109

Nicholson, A. G., Fulford, L. G., Colby, T. V., du Bois, R. M., Hansell, D. M., Wells, A. U., 2002. 'The relationship between individual histologic features and disease progression in idiopathic pulmonary fibrosis'. *Am. J. Respir. Crit. Care Med.* 166 (2), 173–177. doi:10.1164/rccm.2109039

Nielsen, J., Kasperchik, V., 2004. 'Methods and systems for producing an object through solid freeform fabrication', Hewlett-Packard Development Company, L.P. Patent No, WO/2004/062892.

Nixon, M. S., Aguado, A. S., 2012. *Feature Extraction & Image Processing for Computer Vision*, 3rd Edition. Elsevier Academic Press. ISBN: 978-0-123-96549-3

Norcen, R., Podesser, M., Pommer, A., Schmidt, H.-P., Uhla, A., May 2003. 'Confidential storage and transmission of medical image data'. *Comput. Biol. Med.* 33 (3), 277–292. doi:10.1016/S0010-4825(02)00094-X

Noruzi, A., Jun. 2007. 'Folksonomies—Why do we need controlled vocabulary?'. *Webology* 4 (2). URL http://hdl.handle.net/10760/10308

Nottingham Trent University, Oct. 2016. '"Lifelike" human body to help train surgeons'. Online, accessed on 10/07/2018. URL https://www.ntu.ac.uk/about-us/news/news-articles/2016/10/lifelike-human-body-to-help-train-surgeons

OECD, Jul. 2015. *OECD Digital Economy Outlook 2015*. OECD Publishing. ISBN: 978-92-642-3227-3

Okman, L., Gal-Oz, N., Gonen, Y., Gudes, E., Abramov, J., 2011. 'Security issues in NoSQL databases'. *In: Proc. Int. Jt. Conf. IEEE Trust. ICESS-11/FCST-11*. IEEE, IEEE, pp. 541–547. doi:10.1109/TrustCom.2011.70

Oliveira, F. P., Tavares, J. M. R. S., 2014. 'Medical image registration: A review'. *Comput. Methods Biomech. Biomed. Engin.* 17 (2), 73–93. doi:10.1080/10255842.2012.670855

Oracle, 2009. 'Oracle jrockit jvm diagnostics guide'. Tech. rep., Oracle. URL https://docs.oracle.com/cd/E13150_01/jrockit_jvm/jrockit/geninfo/pdf/diagnos.pdf

Osborne, C., Jun. 2013. 'The top ten most common database security vulnerabilities'. Online, accessed on 03/08/2018. URL http://www.zdnet.com/article/the-top-ten-most-common-database-security-vulnerabilities/

Pantanowitz, L., Aug. 2010. 'Digital images and the future of digital pathology'. *J. Pathol. Inform.* 1 (15). doi:10.4103/2153-3539.68332

Patel, D., Aug. 2018. 'Different types of machine learning'. Online, accessed on 15/11/2018. URL https://www.digitalvidya.com/blog/types-of-machine-learning/

Pendleton, C., 2010. 'The world according to Bing'. *Comput. Graph. Appl. IEEE* 30 (4), 15–17. doi:10.1109/MCG.2010.77

Perozo, J., Leung, M. L., Ramírez, E., May 2016. 'HoloMed: A low-cost gesture-based holographic system to learn normal delivery process'. *In: Proc. 4th Simp. Científico y Tecnológico en Comput.* Centro de Computación Gráfica, pp. 160–168. URL https://arxiv.org/pdf/1607.05812.pdf

Pettigrew Temporal Bones, Aug. 2018. 'Buy temporal bones'. Online, accessed on 29/08/2018. URL https://www.temporal-bone.com/buy-temporal-bones/cat_1.html

Pham, D. L., Xu, C., Prince, J. L., 2000. 'Current methods in medical image segmentation'. *Annu. Rev. Biomed. Eng.* 2, 315–337. doi:10.1146/annurev.bioeng.2.1.315

Phé, V., Cattarino, S., Parra, J., Bitker, M.-O., Ambrogi, V., Vaessen, C., Rouprêt, M., Jun. 2017. 'Outcomes of a virtual-reality simulator-training programme on basic surgical skills in robot-assisted laparoscopic surgery'. *Int. J. Med. Robot. Comput. Assist. Surg.* 13 (2), e1740. doi:10.1002/rcs.1740

Pickhardt, P. J., Choi, J. R., Hwang, I., Butler, J. A., Puckett, M. L., Hildebrandt, H. A., Wong, R. K., Nugent, P. A., Mysliwiec, P. A., Schindler, W. R., Dec. 2003. 'Computed tomographic virtual colonoscopy to screen for colorectal neoplasia in asymptomatic adults'. *N. Engl. J. Med.* 349 (23), 2191–2200. doi:10.1056/NEJMoa031618

Pietzsch, T., Saalfeld, S., Preibisch, S., Tomancak, P., Jun. 2015. 'BigDataViewer: visualization and processing for large image data sets'. *Nat. Methods* 12 (6), 481–483. doi:10.1038/nmeth.3392

Prosise, J., Jul. 2009. 'Wicked code: Taking Silverlight Deep Zoom to the next level'. *MSDN Mag.* 24 (7), 90. URL https://msdn.microsoft.com/en-us/magazine/dd943052.aspx

Quinn, M. J., 1994. *Parallel Computing: Theory and Practice*. McGraw-Hill Education. ISBN: 978-0-07-113800-0

Raghu, G., Rochwerg, B., Zhang, Y., Garcia, C. A. C., Azuma, A., Behr, J., Brozek, J. L., Collard, H. R., Cunningham, W., Homma, S., Johkoh, T., Martinez, F. J., Myers, J., Protzko, S. L., Richeldi, L., Rind, D., Selman, M., Theodore, A., Wells, A. U., Hoogsteden, H., Schünemann, H. J., 2015. 'An official ATS/ERS/JRS/ALAT clinical practice guideline: Treatment of idiopathic pulmonary fibrosis: An update of the 2011 clinical practice guideline'. *Am. J. Respir. Crit. Care Med.* 192 (2), e3–e19. doi:10.1164/rccm.201506-1063ST

Raj, B., Apr. 2011. 'Data augmentation: How to use deep learning when you have limited data?—part 2'. Online, accessed on 19/12/2018. URL https://medium.com/nanonets/how-to-use-deep-learning-when-you-have-limited-data-part-2-data-augmentation-c26971dc8ced

Ray, S., Simion, B., Brown, A. D., Apr. 2011. 'Jackpine: A benchmark to evaluate spatial database performance'. *In: Proc. IEEE 27th Int. Conf. Data Eng.* IEEE, Hannover, pp. 1139–1150. doi:10.1109/ICDE.2011.5767929

Rebouças Filho, P. P., Cortez, P. C., da Silva Barros, A. C., Albuquerque, V. H. C., Tavares, J. M. R. S., 2017. 'Novel and powerful 3D adaptive crisp active contour method applied in the segmentation of CT lung images'. *Med. Image Anal.* 35, 503–516. doi:10.1016/j.media.2016.09.002

Rengier, F., Mehndiratta, A., von Tengg-Kobligk, H., Zechmann, C. M., Unterhinninghofen, R., Kauczor, H.-U., Giesel, F. L., Jul. 2010. '3D printing based on imaging data: review of medical applications'. *Int. J. Comput. Assist. Radiol. Surg.* 5 (4), 335–341. doi:10.1007/s11548-010-0476-x

Ribeiro, J., Alves, V., Silva, S., Campos, J., 2015. 'A 3D computed tomography based tool for orthopedic surgery planning'. *In: Tavares, S. M. R. S., Jorge, R. N. (Eds.), Dev. Med. Image Process. Comput. Vis.* Vol. 19 of Lecture Notes in Computational Vision and Biomechanics. Springer, Cham, Ch. 8, pp. 121–137. doi:10.1007/978-3-319-13407-9_8

Richardson, L., Venkataraman, S., Stevenson, P., Yang, Y., Moss, J., Graham, L., Burton, N., Hill, B., Rao, J., Baldock, R. A., Armit, C., nov 2013. 'EMAGE mouse embryo spatial gene expression database: 2014 update'. *Nucleic Acids Research* 42 (1), D835–D844. doi:10.1093/nar/gkt1155

Robb, R. A., 2000. *Biomedical Imaging, Visualization, and Analysis*. John Wiley & Sons. ISBN: 0-471-28353-3

Ron, A., Shulman-Peleg, A., Puzanov, A., 2016. 'Analysis and mitigation of NoSQL injections'. *IEEE Secur. Priv.* 14 (2), 30–39. doi:10.1109/MSP.2016.36

Röntgen, W. C., 1898. 'Weitere Beobachtungen über die Eigenschaften der X-Strahlen'. *Ann. Phys.* 300 (1), 18–37. doi:10.1002/andp.18983000104

Rose, A., Kimbell, J., Webster, C., Harrysson, O., Formeister, E., Buchman, C., Jul. 2015a. 'Multi-material 3D models for temporal bone surgical simulation'. *Ann. Otol. Rhinol. Laryngol.* 124 (7), 528–536. doi:10.1177/0003489415570937

Rose, A. S., Webster, C. E., Harrysson, O. L., Formeister, E. J., Rawal, R. B., Iseli, C. E., May 2015b. 'Pre-operative simulation of pediatric mastoid surgery with 3D-printed temporal bone models'. *Int. J. Pediatr. Otorhinolaryngol.* 79 (5), 740–744. doi:10.1016/j.ijporl.2015.03.004

Rosen, J. M., Soltanian, H., Redett, R. J., Laub, D. R., 1996. 'Evolution of virtual reality: From planning to performing surgery'. *IEEE Eng. Med. Biol. Mag.* 15 (2), 16–22. doi:10.1109/51.486713

Ross, M. H., Pawlina, W., 2011. *Histology: A Text and Atlas*, 6th Edition. Lippincott Wiliams & Wilkins. ISBN: 978-0-7817-7200-6

Rosset, A., Aug. 2010. 'OsiriX MD 510(k) summary of safety and effectiveness'. Tech. rep., Food and Drug Administration, Bernex. URL https://www.accessdata.fda.gov/cdrh_docs/pdf10/K101342.pdf

Rossignac, J., Borrel, P., Jul. 1993. 'Multi-resolution 3D approximations for rendering complex scenes'. *In: Proc. Model. Comput. Graph.* Springer, B.H., Genova, Italy, pp. 455–464. doi:10.1007/978-3-642-78114-8_29

Rudyanto, R. D., Kerkstra, S., van Rikxoort, E. M., Fetita, C., Brillet, P. Y., Lefevre, C., Xue, W., Zhu, X., Liang, J., Öksüz, I., Ünay, D., Kadipaşaoğlu, K., Estépar, R. S. J., Ross, J. C., Washko, G. R., Prieto, J. C., Hoyos, M. H., Orkisz, M., Meine, H., Hüllebrand, M., Stöcker, C., Mir, F. L., Naranjo, V., Villanueva, E., Staring, M., Xiao, C., Stoel, B. C., Fabijanska, A.,

Smistad, E., Elster, A. C., Lindseth, F., Foruzan, A. H., Kiros, R., Popuri, K., Cobzas, D., Jimenez-Carretero, D., Santos, A., Ledesma-Carbayo, M. J., Helmberger, M., Urschler, M., Pienn, M., Bosboom, D. G. H., Campo, A., Prokop, M., de Jong, P. A., Ortiz-de Solorzano, C., Muñoz-Barrutia, A., van Ginneken, B., 2014. 'Comparing algorithms for automated vessel segmentation in computed tomography scans of the lung: The VESSEL12 study'. *Med. Image Anal.* 18 (7), 1217–1232. doi:10.1016/j.media.2014.07.003

Rueden, C. T., Schindelin, J., Hiner, M. C., DeZonia, B. E., Walter, A. E., Arena, E. T., Eliceiri, K. W., Nov. 2017. 'ImageJ2: ImageJ for the next generation of scientific image data'. *BMC Bioinformatics* 18 (529). doi:10.1186/s12859-017-1934-z

Rupp, K., Feb. 2018. '42 years of microprocessor trend data'. Online, accessed on 26/06/2018. URL https://www.karlrupp.net/2018/02/42-years-of-microprocessor-trend-data/

Sachs, E., Cima, M., Williams, P., Brancazio, D., Cornie, J., Nov. 1992. 'Three dimensional printing: Rapid tooling and prototypes directly from a CAD model'. *J. Eng. Ind.* 114 (4), 481. doi:10.1115/1.2900701

Sato, Y., Nakajima, S., Shiraga, N., Atsumi, H., Yoshida, S., Koller, T., Gerig, G., Kikinis, R., 1998. 'Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images'. *Med. Image Anal.* 2 (2), 143–168. doi:10.1016/S1361-8415(98)80009-1

Schmid, B., Schindelin, J., Cardona, A., Longair, M., Heisenberg, M., 2010. 'A high-level 3D visualization API for Java and ImageJ'. *BMC Bioinformatics* 11 (1), 274. doi:10.1186/1471-2105-11-274

Schneider, C. A., Rasband, W. S., Eliceiri, K. W., Jun. 2012. 'NIH image to ImageJ: 25 years of image analysis'. *Nat. Methods* 9, 671–675. doi:10.1038/nmeth.2089

Scott, A. E., Vasilescu, D. M., Seal, K. A. D., Keyes, S. D., Mavrogordato, M. N., Hogg, J. C., Sinclair, I., Warner, J. A., Hackett, T.-L., Lackie, P. M., Jun. 2015. 'Three dimensional imaging of paraffin embedded human lung tissue samples by micro-computed tomography'. *PLoS One* 10 (6). doi:10.1371/journal.pone.0126230

Scott, M., May 2014. 'Research data management'. Ph.D. thesis, University of Southampton. doi:10.5258/SOTON/374711

Scott, M., Boardman, R. P., Reed, P. A., Cox, S. J., 2014. 'Managing heterogeneous datasets'. *Inf. Syst.* 44, 34–53. doi:10.1016/j.is.2014.03.004

Sears, R., van Ingen, C., Gray, J., 2006. 'To blob or not to blob: Large object storage in a database or a filesystem'. Tech. rep., Microsoft. URL https://www.microsoft.com/en-us/research/publication/to-blob-or-not-to-blob-large-object-storage-in-a-database-or-a-filesystem/#

Seifert, C., Kump, B., Kienreich, W., Granitzer, G., Granitzer, M., Jul. 2008. 'On the beauty and usability of tag clouds'. *In: Proc. 12th Int. Conf. Inf. Vis. (IV 2008)*. IEEE, London, UK, pp. 17–25. doi:10.1109/IV.2008.89

Sheppard, A. P., Sok, R. M., Averdunk, H., Aug. 2004. 'Techniques for image enhancement and segmentation of tomographic images of porous materials'. *Phys. A Stat. Mech. its Appl.* 339 (1-2), 145—151. doi:10.1016/j.physa.2004.03.057

Shojaii, R., Alirezaie, J., Babyn, P., 2005. 'Automatic lung segmentation in CT images using watershed transform'. *In: IEEE Int. Conf. Image Process. (ICIP 2005)*. IEEE, Genova, Italy, p. 4. doi:10.1109/ICIP.2005.1530294

Shum, H.-Y., Szeliski, R., Jan. 1998. 'Construction and refinement of panoramic mosaics with global and local alignment'. *In: 6th Int. Conf. Comput. Vis. (ICCV 1998)*. IEEE, Bombay, India, pp. 953–956. doi:10.1109/ICCV.1998.710831

Siemens Healthcare GmbH, 2016. 'SOMATOM Perspective'. Tech. rep., Siemens, Erlangen, Germany. URL https://static.healthcare.siemens.com/siemens_hwem-hwem_ssxa_websites-context-root/wcm/idc/groups/public/@global/@imaging/@ct/documents/download/mdaw/njgx/{~}edisp/ct_somatom_perspective_brochure_0313-00449304.pdf

Sinclair, J., Cardew-Hall, M., Feb. 2008. 'The folksonomy tag cloud: When is it useful?'. *J. Inf. Sci.* 34 (1), 15–29. doi:10.1177/0165551506078083

Smalley, D. E., Nygaard, E., Squire, K., Wagoner, J. V., Rasmussen, J., Gneiting, S., Qaderi, K., Goodsell, J., Rogers, W., Lindsey, M., Costner, K., Monk, A., Pearson, M., Haymore, B., Peatross, J., Jan. 2018. 'A photophoretic-trap volumetric display'. *Nature* 553, 486–490. doi:10.1038/nature25176

Smith, S. W., 1997. *The Scientist and Engineer's Guide to Digital Signal Processing*, 1st Edition. California Technical Publishing. ISBN: 0-9660176-3-3

Sonka, M., Fitzpatrick, J. M. (Eds.), 2009. *Handbook of Medical Imaging: Medical Image Processing and Analysis*. Vol. 2. SPIE Press. ISBN: 0-8194-3622-4

Soo, G., Cain, T., Jul. 2017. 'SPECT-CT scan'. Online, accessed on 03/08/2018. URL https://www.insideradiology.com.au/spect-ct-scan/

Sorace, J., Aberle, D. R., Elimam, D., Lawvere, S., Tawfik, O., Wallace, W. D., Dec. 2012. 'Integrating pathology and radiology disciplines: An emerging opportunity?'. *BMC Med.* 10 (1), 100. doi:10.1186/1741-7015-10-100

Stalling, D., Westerhoff, M., Hege, H.-C., 2005. 'Amira: A highly interactive system for visual data analysis'. *In: Hansen, C. D., Johnson, C. R. (Eds.), Vis. Handb.* Elsevier Butterworth-Heinemann, Burlington, MA, USA, Ch. 38, pp. 749–767. doi:TK7882.I6V59

Steuer, J., Dec. 1992. 'Defining virtual reality: Dimensions determining telepresence'. *J. Commun.* 42 (4), 73–93. doi:10.1111/j.1460-2466.1992.tb00812.x

Strauch, C., Jun. 2011. 'NoSQL databases'. Tech. rep., Stuttgart Media University. URL http://christof-strauch.de/nosqldbs.pdf

Suryawati, S., 2014. 'CT basic & neuroradiology'. Online, accessed on 26/06/2018. URL http://slideplayer.com/slide/9405067/

Suzuki, M., Hagiwara, A., Ogawa, Y., Ono, H., 2007. 'Rapid-prototyped temporal bone and inner-ear models replicated by adjusting computed tomography thresholds.'. *J. Laryngol. Otol.* 121 (11), 1025–1028. doi:10.1017/S0022215107006706

Tay, S., Blanche, P.-A., Voorakaranam, R., Tunç, A. V., Lin, W., Rokutanda, S., Gu, T., Flores, D., Wang, P., Li, G., Hilaire, P. S., Thomas, J., Norwood, R. A., Yamamoto, M., Peyghambarian, N., Feb. 2008. 'An updatable holographic three-dimensional display'. *Nature* 451, 694—698. doi:10.1038/nature06596

The Open Microscopy Environment, Oct. 2017. 'Omero events and provenance'. Online, accessed on 12/12/2018. URL https://docs.openmicroscopy.org/omero/5.4.0/developers/Server/Events.html

The Open Microscopy Environment, Oct. 2018. 'Scaling omero'. Online, accessed on 12/12/2018. URL https://docs.openmicroscopy.org/omero/5.4.9/developers/Server/ScalingOmero.html

Thomas, S. W., Jackson, W. K., Key, R. D., White, G. L., Elwell, K. D., Fluegel, K. G., 1997. 'Flexible reinforced rubber part manufacturing process utilizing stereolithography tooling', E-Systems Inc. Patent No, 5603797.

Thomsen, A. S. S., Bach-Holm, D., Kjærbo, H., Højgaard-Olsen, K., Subhi, Y., Saleh, G. M., Park, Y. S., la Cour, M., Konge, L., Apr. 2017. 'Operating room performance improves after proficiency-based virtual reality cataract surgery training'. *Ophthalmology* 124 (4), 524–531. doi:10.1016/j.ophtha.2016.11.015

Tomaszuk, D., 2010. 'Document-oriented triplestore based on RDF/JSON'. *Stud. Logic, Gramm. Rhetor.* 22 (35), 125–140.

Tong, J., Zhou, J., Liu, L., Pan, Z., Yan, H., Apr. 2012. 'Scanning 3D full human bodies using kinects'. *IEEE Trans. Vis. Comput. Graph.* 18 (4), 643–650. doi:10.1109/TVCG.2012.56

Torbati, N., Ayatollahi, A., Kermani, A., Jan. 2014. 'An efficient neural network based method for medical image segmentation'. *Comput. Biol. Med.* 44, 76–87. doi:10.1016/j.compbiomed.2013.10.029

Toshiba Medical, 2016. 'Aquilion ONE / GENESIS Edition—Transforming CT'. Online, accessed on 03/08/2018. URL https://medical.toshiba.com/download/ct-br-aq-one-genesis

Traina Jr., C., Traina, A. J. M., Araújo, M. R. B., Bueno, J. M., Chino, F. J. T., Razente, H., Azevedo-Marques, P. M., Dec. 2005. 'Using an image-extended relational database to support content-based image retrieval in a PACS'. *Comput. Methods Programs Biomed.* 80 (1), S71–S83. doi:10.1016/S0169-2607(05)80008-2

TrapX Research Labs, 2016. 'MEDJACK.2 hospitals under siege'. Tech. rep., TrapX Research Labs. URL http://deceive.trapx.com/rs/929-JEW-675/images/AOA_Report_TrapX_MEDJACK.2.pdf

Treanor, D., Quirke, P., Jul. 2007. 'The virtual slide and conventional microscope—A direct comparison of their diagnostic efficiency'. *In: 4th Jt. Meet. Br. Div. Int. Acad. Pathol. Pathol. Soc. Gt. Britain Irel. (Glasgow Pathology 2007)*. Glasgow, UK. URL http://www.virtualpathology.leeds.ac.uk/research/Publications/pub_docs/dt/Treanor%20and%20Quirke%20-%202007%20-%20The%20virtual%20slide%20and%20conventional%20microscope.pdf

Trotts, I., Mikula, S., Jones, E. G., Apr. 2007. 'Interactive visualization of multiresolution image stacks in 3D'. *Neuroimage* 35 (3), 1038–1043. doi:10.1016/j.neuroimage.2007.01.013

Tsiotsios, C., Petrou, M., may 2012. 'On the choice of the parameters for anisotropic diffusion in image processing'. *Pattern Recognit.* 46 (5), 1369–1381. doi:10.1016/j.patcog.2012.11.012

Tsuruoka, Y., Tsujii, J., Ananiadou, S., Nov. 2008. 'FACTA: A text search engine for finding associated biomedical concepts'. *Bioinformatics* 24 (21), 2559–2560. doi:10.1093/bioinformatics/btn469

Tudorica, B. G., Bucur, C., Jun. 2011. 'A comparison between several NoSQL databases with comments and notes'. *In: RoEduNet International Conference 10th Edition: Networking in Education and Research (RoEduNet 2011)*. IEEE, Iasi, Romania. doi:10.1109/RoEduNet.2011.5993686

University of Dundee, 2014. 'About OMERO'. Online, accessed on 01/08/2018. URL http://www.openmicroscopy.org/site/products/omero

Vaish, G., 2013. *Getting Started with NoSQL*. Packt Publishing Ltd. ISBN: 978-1-8496-9499-5

Vincent, L., Soille, P., Jun. 1991. 'Watersheds in digital spaces: An efficient algorithm based on immersion simulations'. *IEEE Trans. Pattern Anal. Mach. Intell.* 13 (6), 583–598.

Voelkel, D. D., Mazumder, J., Apr. 1990. 'Visualization of a laser melt pool'. *Appl. Opt.* 29 (12), 1718–1720. URL https://www.osapublishing.org/DirectPDFAccess/B6CEDB40-AF11-2EDA-F82560144B0A2E48_166047/ao-29-12-1718.pdf

Wang, W., Chen, L., Zhang, Q., Sep. 2015. 'Outsourcing high-dimensional healthcare data to cloud with personalized privacy preservation'. *Comput. Networks* 88, 136–148. doi:10.1016/j.comnet.2015.06.014

Warnock, M. J., Toland, C., Evans, D., Wallace, B., Nagy, P., Nov. 2007. 'Benefits of using the DCM4CHE DICOM archive'. *J. Digit. Imaging* 20 (Suppl. 1), 125–129. doi:10.1007/s10278-007-9064-1

Wasson, M., Nov. 2017. 'Web-queue-worker architecture style'. Online, accessed on 28/02/2018. URL https://docs.microsoft.com/en-us/azure/architecture/guide/architecture-styles/web-queue-worker

Webb, W. R., Müller, N. L., Naidich, D. P., 2014. *High-resolution CT of the lung*, 5th Edition. Lippincott Williams & Wilkins. ISBN: 978-1-451-17601-8

Weghorst, S., Aug. 1997. 'Augmented reality and parkinson's disease'. *Commun. ACM* 40 (8), 47–48. doi:10.1145/257874.257884

Weickert, J., 1998. *Anisotropic Diffusion in Image Processing*. B.G. Teubner, Stuttgart, Germany. ISBN: 978-3-519-02606-8

White, D., 2003. 'Ultrasonic object consolidation', Solidica, Inc. Patent No, 6519500.

White, J., Oct. 2015. 'Report: Hospitals need better IT security protocol'. Online, accessed on 03/08/2018. URL http://www.healthcarebusinesstech.com/hospital-it-security-protocol/

Wiet, G. J., Stredney, D., Sessanna, D., Bryan, J. A., Welling, B. D., Schmalbrock, P., Jul. 2002. 'Virtual temporal bone dissection: An interactive surgical simulator'. *Otolaryngol. - Head Neck Surg.* 127 (1), 79–83. doi:10.1067/mhn.2002.126588

Williams, J. R., Lessard, P. R., Desu, S., Clark, E. M., Bagrow, J. P., Danforth, C. M., Dodds, P. S., Aug. 2015. 'Zipf's law holds for phrases, not words'. *Sci. Rep.* 5 (12209). doi:10.1038/srep12209

Willoughby, C., Bird, C. L., Coles, S. J., Frey, J. G., Nov. 2014. 'Creating context for the experiment record. user-defined metadata: Investigations into metadata usage in the LabTrove ELN'. *J. Chem. Inf. Model.* 54 (12), 3268–3283. doi:10.1021/ci500469f

Wittman, T., Nov. 2014. 'Lecture 10: Snakes'. Tech. rep., The Citadel. URL http://macs.citadel.edu/wittman/490/Lectures/Lec10_490.pdf

Wollatz, L., Cox, S. J., Johnston, S. J., Sep. 2015. 'Web-based manipulation of multiresolution micro-CT images'. *In: Proc. 11th Int. Conf. eScience (eScience 2015)*. IEEE, Munich, Germany, pp. 308–311. doi:10.1109/eScience.2015.42

Wollatz, L., Johnston, S. J., Lackie, P. M., Cox, S. J., Dec. 2017. '3D histopathology—A lung tissue segmentation workflow for microfocus X-ray-computed tomography scans'. *J. Digit. Imaging* 30 (6), 772–781. doi:10.1007/s10278-017-9966-5

Wollatz, L., Konieczny, K., Vandervelde, C., Mitchell, T., Cox, S., Burgess, A., Ismail-Koch, H., Sep. 2016. 'The use of 3D-printed paediatric temporal bones as a training tool'. *In: BAPO Annu. Sci. Meet. (BAPO 2016)*. British Association for Paediatric Otolaryngology, Liverpool, United Kingdom, p. 24.

Wollatz, L., Scott, M., Johnston, S. J., Lackie, P. M., Cox, S. J., Oct. 2018. 'Curation of image data for medical research'. *In: Proc. 14th Int. Conf. eScience (eScience 2018)*. IEEE, Amsterdam, Netherlands, pp. 105–113. doi:10.1109/eScience.2018.00026

Wong, A., Lou, S. L., 2000. 'Medical image archive, retrieval, and communication'. *In: Bankman, I. N. (Ed.), Handb. Med. Imaging*. Biomedical Engineering. Elsevier, Ch. 47, pp. 771–781. doi:10.1016/B978-012077790-7/50055-2

Wu, H., Ambavane, A., Mukherjee, S., Mao, S., 2017. 'A coherent healthcare system with RDBMS, NoSQL and GIS databases'. *In: Proc. 32nd Int. Conf. Comput. Their Appl. (CATA 2017)*. ISCA, Honolulu, HI, USA, pp. 313–318. URL http://hdl.handle.net/1805/14911

Zhang, H., Fritts, J. E., Goldman, S. A., May 2008. 'Image segmentation evaluation: A survey of unsupervised methods'. *Comput. Vis. Image Underst.* 110 (2), 260–280. doi:10.1016/j.cviu.2007.08.003

Zhang, Y. J., Aug. 1996. 'A survey on evaluation methods for image segmentation'. *Pattern Recognit.* 29 (8), 1335–1346. doi:10.1016/0031-3203(95)00169-7

Zipf, G. K., 1949. *Human Behaviour and the Principle of Least-Effort*. Addison-Wesley, Cambridge, MA, USA. ISBN: 978-1-614-27312-7 (2002 reprint)

Zirkle, M., Roberson, D. W., Leuwer, R., Dubrowski, A., 2007. 'Using a virtual reality temporal bone simulator to assess otolaryngology trainees'. *Laryngoscope* 117 (2), 258–263. doi:10.1097/01.mlg.0000248246.09498.b4

Zitová, B., Flusser, J., Oct. 2003. 'Image registration methods: A survey'. *Image Vis. Comput.* 21 (11), 977–1000. doi:10.1016/S0262-8856(03)00137-9