IntechOpen

Computer Methods and Programs in Biomedical Signal and Image Processing

Edited by Lulu Wang

Computer Methods and Programs in Biomedical Signal and Image Processing

Edited by Lulu Wang

Published in London, United Kingdom

IntechOpen

Supporting open minds since 2005

Computer Methods and Programs in Biomedical Signal and Image Processing
http://dx.doi.org/10.5772/intechopen.73344
Edited by Lulu Wang

Contributors
Hong Seong Park, Donguk Yu, Ihab Zaqout, Peter Larbi, Daniel Larbi, Mouloud Adel, Hichem Guedri, Hafedh Belmabrouk, Lulu Wang

First published in London, United Kingdom, 2020 by IntechOpen
IntechOpen is the global imprint of INTECHOPEN LIMITED, registered in England and Wales, registration number: 11086078, 7th floor, 10 Lower Thames Street, London, EC3R 6AF, United Kingdom
Printed in Croatia

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

Additional hard and PDF copies can be obtained from orders@intechopen.com

Computer Methods and Programs in Biomedical Signal and Image Processing
Edited by Lulu Wang
p. cm.
Print ISBN 978-1-83880-242-4
Online ISBN 978-1-83880-245-5
eBook (PDF) ISBN 978-1-83880-246-2

Meet the editor

Lulu Wang is currently a distinguished professor of biomedical engineering at the College of Health Science and Environmental Engineering at Shenzhen Technology University in China. She received her ME (first class hons.) and PhD degrees from Auckland University of Technology, New Zealand, in 2009 and 2013, respectively. From 2013 to 2015, she was a research fellow with the Institute of Biomedical Technologies, Auckland University of Technology, New Zealand. In June 2015, Dr. Wang became an associate professor of biomedical engineering with the Hefei University of Technology. In June 2019, she became a distinguished professor of biomedical engineering at Shenzhen Technology University. Her research interests include medical devices, electromagnetic sensing and imaging, and computational mechanics. Over the past 5 years, Dr. Wang has authored more than 60 peer-reviewed publications, two ASME books, six book chapters, and four issued patents. Dr. Wang is a member of ASME, IEEE, MRSNZ, AAAS, PSNZ, and IPENZ. She is an active reviewer of numerous journals, books, and conferences. Dr. Wang has edited three books and two special issues of international journals. She has received multiple national and international awards from various professional societies and organizations.

Contents

Preface

Computers have significantly changed everyone's daily lives ever since the invention of the first computer. With the rapid development of medical care and biomedical technologies, the requirements for computational methods and programs in biomedical engineering and biotechnology are increasing. Traditional biomedical and clinical data collection and analysis methods are being replaced by computer programs that can predict clinical decisions. As a result, more cost-effective and powerful computing methods and programs have been replaced by expensive equipment, tests, and examinations.

There is no doubt that a large number of biomedical and clinical data, such as signals and images, cannot be extracted by human eyes. The rapid development of computer programs has provided much help in solving such problems. Computer programs can truly simulate abstract models of specific or equivalent clinical systems. Computer models have become a useful part of mathematical modeling of many natural systems, including psychology, physiology, biology, biomedicine, and biotechnology, as well as all branches of bioengineering, to better understand the work of the systems mentioned above.

With the fast development of biomedical engineering, electronics, and computer technologies, computational methods and programs have attracted much interest in biomedical signal and imaging processing over the past two decades. Practical computational approaches and devices need to be developed to analyze complex data and to provide clinicians and healthcare providers with a prominent recommendation and prediction. The applications of computer methods and programs in biological signals and imaging have attracted the attention of researchers all over the world. Many computational methods and devices have been proposed, tested, implemented, and analyzed in biotechnological and biomedical engineering applications. However, the application of computer methods and programs has both advantages and advantages. Obviously, there are more advantages than limitations, and it is certain that computer methods and programs will dominate all fields of biomedicine and bioengineering.

With the design of more advanced computational methods and programs, it is expected that the methods and equipment for processing the data generated by these systems need to be improved accordingly. These new methods not only help to extract new knowledge, which is difficult to obtain from existing traditional interpretations, but also lay a foundation for providing fast prediction information and helping clinicians make better clinical decisions. As shown in the chapters, these changes need to address both the size and complexity of the generated data.

This book aims to provide a brief update to the current status of and advances in the computational methods and programs used for the development of the theory and practice of biomedical signal and image communication. The book comprises a collection of invited manuscripts, written in a convenient way and of manageable length. These timely collections will provide an invaluable resource for initial inquiries about technologies and will encapsulate the latest developments and applications with reference sources for further detailed information. The methods described in this book cover a wide range of computational algorithms that are widely used in bioengineering and biomedicine. The content and format are specifically designed to stimulate the further development and application of these technologies by reaching out to nonspecialists across a broad audience.

I would like to express my gratitude to the publishers (IntechOpen) for their help with this book and the authors of accepted manuscripts for their work and patience.

Lulu Wang Ph.D.
Hefei University of Technology,
China

Shenzhen Technology of University,
China

Chapter 1

Introductory Chapter: Computational Methods in Biomedical Engineering and Biotechnology

Lulu Wang

1. Introduction

Computers have significantly changed everyone's daily lives since the first computer came to the world. With the rapid development of medical care and biomedical technologies, the requirements for computational methods and programs in biomedical engineering and biotechnology are becoming higher and higher [1]. Traditional biomedical and clinical data collection and analysis methods are replaced by computer programs that can predict clinical decisions. As a result, more cost-effective and powerful computing methods and programs have been replaced by expensive equipment, tests, and examinations.

There is no doubt that a large number of biomedical and clinical data, such as signals and images, cannot be extracted by human eyes. The rapid development of computer programs has provided great help to solve such problems. Computer programs can truly simulate abstract models of specific or equivalent clinical systems [2]. Computer models have become a useful part of mathematical modelling of many natural systems, including psychology, physiology, biology, biomedicine and biotechnology, as well as all branches of bioengineering, in order to better understand the work of the systems mentioned above.

With the fast development of biomedical engineering, electronics and computer technologies, computational methods and programs have attracted much interest in biomedical signal and imaging processing over the past 2 decades [3–5]. Effective computational approach and devices need to be developed to analyse complex data and to provide clinician and health care providers a prominent recommendation and prediction. The applications of computer methods and programs in biological signals and imaging have attracted the attention of researchers all over the world. Many computational methods and devices have been proposed, tested, implemented and analysed in biotechnology and biomedical engineering applications. Although the application of computer methods and programs has both advantages and advantages. Obviously, they have more advantages than their limitations, and it is certain that computer methods and programs will dominate all fields of biomedicine and bioengineering.

With the design of more advanced computer methods and programs, it is expected that the methods and equipment for processing the data generated by these systems need to be improved accordingly. These new methods not only help to extract new knowledge, which is difficult to obtain from existing traditional interpretations, but also lay a foundation for providing fast prediction information and helping clinicians make better clinical decisions. As shown in the chapters, these changes need to address both the size and complexity of the generated data.

The book 'Computer Methods and Programs in Biomedical Signal and Image Processing' aims to provide a brief update to the current status of and advances in the computer methods and programs used for the development of the theory and practice of biomedical signal and image communication. The book comprises a collection of invited manuscripts, written in a convenient way and manageable length. These timely collections will provide an invaluable resource for initial enquiries about technologies and encapsulating the latest developments and applications with reference sources for further detailed information. The methods described in this book cover a wide range of computational algorithms that are widely used in bioengineering and biomedicine. The content and format are specifically designed to stimulate the further development and application of these technologies by reaching out to the non-specialist across a wide audience.

This book is intended to expose the latest developments of scientists and engineers, covering a variety of complementary topics, with a view to enhancing people's overall understanding of the computer science and biomedical image communications. It will benefit students, scientists and researchers in applied computer science. Engineers and clinicians working in imaging will also find this book very useful. I hope you will enjoy this book.

Author details

Lulu Wang
School of Instrument Science and Opto-electronics Engineering,
Hefei University of Technology, China

*Address all correspondence to: luluwang2015@hfut.edu.cn

References

[1] Hii PC, Chung WY. A comprehensive ubiquitous healthcare solution on an android™ mobile device. Sensors. 2011;**11**:6799-6815

[2] Hosseini SAH, Sohrabpour A, He B. Electromagnetic source imaging using simultaneous scalp EEG and intracranial EEG: An emerging tool for interacting with pathological brain networks. Clinical Neurophysiology. 2018;**129**(1):168-187

[3] Jin KH, Mccann MT, Froustey E, Unser M. Deep convolutional neural network for inverse problems in imaging. IEEE Transactions on Image Processing. 2017:1-1

[4] Braojos R, Bortolotti D, Bartolini A, Ansaloni G, Benini L, Atienza D. A synchronization-based hybrid-memory multi-core architecture for energy-efficient biomedical signal processing. IEEE Transactions on Computers. 2017;**66**(4):575-585

[5] Mitov IP. A method for assessment and processing of biomedical signals containing trend and periodic components. Medical Engineering and Physics. 1998;**20**(9):660-668

Chapter 2

Adopting Microsoft Excel for Biomedical Signal and Image Processing

Peter Ako Larbi and Daniel Asah Larbi

Abstract

Microsoft Excel was recently added to the list of software applications for signal and image processing. The use of Excel as a powerful tool for teaching signal and image data processing techniques as demonstrated in agriculture and natural resource management can be easily adopted for biomedical applications. In the same vein, Excel's proven utility as a research tool can also be harnessed. This chapter expands the methodology of signal and image formation, visualization, enhancement, and image data fusion using Excel. Different types of techniques used in biomedical imaging are introduced, including: X-ray radiography (X-rays), computerized tomography (CT), ultrasound (U/S), magnetic resonance imaging (MRI), and optical imaging. However, the chapter mainly focuses on optical imaging involving a single spectrum or multiple spectra such as RGB. Specific illustrations of corresponding outputs from different techniques are discussed in the chapter for a better appreciation by the reader.

Keywords: optical imaging, multispectral imagery, image visualization, image enhancement, RGB image data, RGB2X, RGBExcel

1. Introduction

Biomedical signal and digital image processing pertains to the manipulation of signal and image data to obtain output images that are useful for human health diagnostics and therapeutic purposes. This may range from the capture of a static image of the condition of an organ or tissue to the capture of multiple images at different stages of a condition to monitor the physiological process of development. It involves using different approaches for visualizing biological tissues, altogether aimed at improving health. Related research efforts lead to the development of better diagnostic tools in clinical settings as well as the improvement in developing appropriate drugs and other therapies.

Different types of techniques are used in biomedical imaging including X-rays [1], computerized tomography (CT) [2], ultrasound (U/S) [3], magnetic resonance imaging (MRI), and optical imaging. The scientific and engineering principles and technology underlying these techniques and associated sensors, instrumentation, and software continue to evolve [1, 4]. This has led to more accessibility to health-care involving these techniques and associated technologies. [1] provided a survey of current commercial implementations and identification of essential differences and similarities in image and signal processing approaches. The fundamentals of biomedical image processing including terminology used and imaging modalities are presented by [5]. The need to obtain high quality images of diagnostic and therapeutic relevance remains at the heart of the practice. Both high-level and low-level image processing and visualization are of great significance. While high-level image processing and analysis as programmed in a given biomedical device may not be directly applicable to other needs, low-level processing and analysis can be used for research and development and/or teaching purposes to explore other possibilities. Also, despite the fact that computation approaches may differ, the output images similar purposed approaches may yield similar outputs.

Researchers in medicine and biomedical science and engineering often utilize spectroscopic techniques for research and clinical purposes. Measurements in the ultra-violet, visible, and near infrared wavelength ranges are performed for different medical and biomedical applications. The use of fiber optic based spectrometers allow for in vivo measurements and testing procedures that are inexpensive and disposable. The generation of diagnostic information based on processing and analyzing digital signal and imagery obtained from such techniques has also been applied in other disciplines including agriculture [6], biology [7], and geography at different spatial scales.

Digital imagery consists of dual-dimensional arrays (rows and columns) of square pixels having values that represent signal intensities captured by the imagery sensor's detector [8]. The imagery sensor may operate in a single band (monochrome) on the electromagnetic spectrum or in multiple bands (multispectral) which may be contiguous or discrete. The pixel values of a monochrome (8-bit) image have a range of 0 to 255 (i.e., 256 colors). Multispectral image data consisting of three bands are fused to obtain color images with 16,777,216 (i.e., $256 \times 256 \times 256$) possible colors. True color or RGB images such as are obtained by commercial digital cameras consist of data from the red (R), green (G), and blue (B) bands of the visible portion of the electromagnetic spectrum. In certain applications, different images can be layered on top of each other to give composite images containing details not found in a single image.

Several software applications have been commonly known for their use in performing low-level image processing and analysis. However, until recently, Microsoft Excel was not recognized for such utility although it had been attempted in the works of [9–11] and shown to have some potential. [9] utilized a Visual Basic for Applications (VBA) macro program for visualizing terrain image data. They determined that its function and quality were similar to a typical geographic information system (GIS). However, some macros do not work properly when used in versions of Excel other the versions used to create them. [10] also developed a clustering approach in Excel based on k-means algorithm. They made use of an Excel add-in called "loadImageArray" in acquiring RGB data from images. Likewise, [11] developed an Excel 2007 add-in known as "Image Manipulation", which can extract data (decomposed into R, G, and B data in separate workbooks) from a JPEG image, resize the image to fit the available number of columns, and export it back as JPEG images. After properly saving a required library file, the "Image Manipulation" add-in runs smoothly in later versions of Excel. It is possible to further manipulate the data using either the add-in's programmed methods or Excel's features. In view of the above examples, one of the major limitations for using Excel for image processing and analysis has been how to make the image data available in Excel 'painlessly'. Now, image data can be extracted via [12] and copied to an Excel workbook for processing and analysis.

Recently, several publications of Larbi and colleagues have extensively demonstrated the utility of Microsoft Excel for image processing and analysis.

First was a study which demonstrated similarities in processed data from images captured using different cameras [13]. Image data were extracted into Excel using the RGBExcel application [14]. The RGBExcel application was further improved as a new version called RGB2X, which provides several advantages over the RGBExcel including the ability to extract data from multiple image files at once into corresponding Excel workbooks [15]. An extensive demonstration of applying the methods of image processing and analysis is presented in [16] and further expanded in [17]. The Excel based methodology was useful in demonstrating the potential use of time-lapse cameras in studying changes in condition over time [6, 18]. It was also useful in determining the optimal timing for cover crop termination to control weeds while maintaining good soybean yields [19]. Although the examples demonstrated are mainly agricultural examples, it is well understood that the methods of image processing are broadly applicable and of interdisciplinary importance.

As such, based on the potential of using Microsoft Excel for processing and analyzing digital images as shown in previous literature, the main aim of this chapter is to demonstrate the potential use in biomedical optical imaging applications. Specific illustrations of corresponding outputs of different visualization, and image enhancement, and image data fusion are provided for a better appreciation by the reader.

2. Processing and analyzing biomedical images

For simplicity, image processing and analysis is discussed as an integrated methodology rather than separate steps or one being a subset of the other. Biomedical digital image processing generally implies using computers to manipulate digital image data pertaining to biomedical sciences to output other images possessing diagnostic information. Image analysis encompasses the entirety of steps used for quantitative assessment and abstract interpretation of biomedical images. Image processing and analysis includes a number of processes that adjust contrast and brightness, assist in distinguishing between different tissues, and others that perform measurements or assess the rate of development of certain processes.

Overall, the steps can be outlined as: image formation, visualization, enhancement, analysis, and interpretation. These steps and their order are not conclusive but suggestive as the specific order depends on the images being processed. For instance, different image enhancement techniques can be applied at different stages of image processing and analysis. For high-level analysis of automated systems, the appropriate steps are incorporated into algorithms with preset parameters for the target images. Such steps require a priori knowledge of the basic features and constitution of the images, and the algorithms typically cannot be directly transferred to other applications.

This chapter focuses on simple and commonly used methods for processing optical images such as for enhancing images prior to additional digital analysis and/or visual interpretation. The use of Excel in implementing the various image processing steps are demonstrated in the following sections. The general procedure used in Microsoft Excel for implementing these different image processing methods are outlined as follow: (1) creating a new (blank) output worksheet; (2) applying the method's formula to the upper left cell, i.e. A1, utilizing the original R, G, and B data; and (3) copying the formula to apply to the rest of the cells up to the extent of the original data. The original data is therefore preserved in the processing.

3. Image formation

Image formation encompasses all the necessary steps from image capture to creating a digital matrix of pixels. With the purpose of using Microsoft Excel for image processing, image formation also includes making the image data available in Excel. These are discussed further in sub-sections 3.1 and 3.2.

3.1 Acquiring digital images

Optical medical images can be obtained using a number of optical devices including endoscopes in their different kinds such as bronchoscopes, colonoscopes, and laparoscopes that allow physicians and clinicians to view inside various channels of anatomic tubes. Examples of such channels include the bronchial system, the gastrointestinal tract, and the genitourinary system. An extended list of the different types of endoscopes and their uses is provided by [20]. Endoscopic diagnostic procedures use a guided probe based system with a camera and lighting for imaging. More details on the procedures including pre-procedure preparation and post-procedure expectations can be accessed in [21].

Different materials reflect incident light differently. Measurements of light reflectance (proportion of incident light that is reflected) by different materials such as the human skin and internal tissues can unveil great amounts of diagnostic information about their conditions. An example of this kind of measurement is the use of a dermal auto-fluorescence measurement for detecting advanced glycogen end products (AGEs) in patients [22]. Near-infrared measurements of deep tissue reflectance is also useful for assessing the effectiveness of radiation therapy as decreasing cancerous masses. Such measurements depend on spectroscopic analysis including image processing.

Cameras or imaging systems such as discussed above output RGB or RGB-type images. For the purpose of demonstrating potential applications, image acquisition using common devices such as digital cameras and cellphones is also considered for capturing external parts of the human body to assess variability in skin condition resulting from disease or exposure to certain extreme environments. Examples of such targets include healthy skins, skins with sunburn, facial skins with acne, and skins showing different stages of a disease.

For the purpose of demonstration in this chapter, images were acquired from [23]. The scales of these images are uncertain. However, this does not diminish their usefulness for demonstration purposes. **Figure 1** shows images of tissues with various conditions: (a) eczema atopic dermatitis symptom skin; (b) allergic rash dermatitis eczema skin; (c) psoriasis vulgaris skin; and (d) colon affected by ulcerative colitis.

3.2 Image digitization

Regardless of file format, image data from the R, G, and B bands can be extracted from images to Microsoft Excel using the RGB2X software application [13, 17, 24]. The extracted data are populated in different worksheets and named accordingly. **Figure 2** shows worksheet tabs corresponding to R, G, and B band data, with a section of the G band data displayed. The UC worksheet originally has three adjacent columns having sequentially unidimensional R, G, and B datasets. This allows for certain analyses like histogram and clustering to be accomplished. Alternatively, the image data can be extracted by [12] and uploaded into Excel.

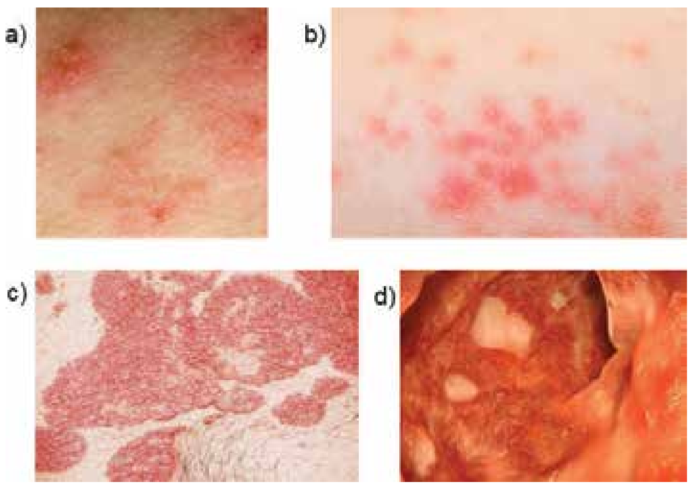
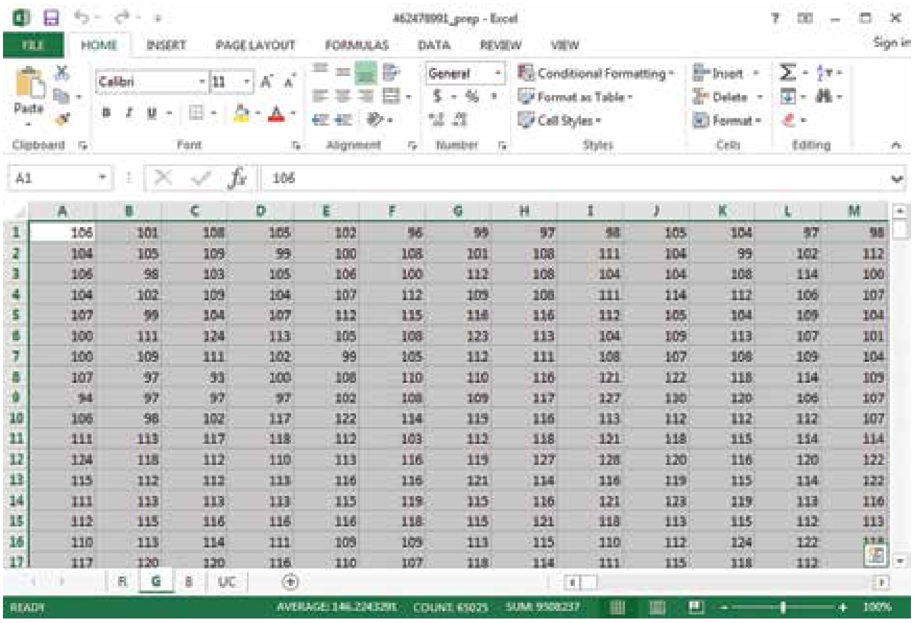


Figure 1.
Images used for demonstration: (a) eczema atopic dermatitis symptom skin; (b) allergic rash dermatitis eczema skin; (c) psoriasis vulgaris skin; and (d) colon affected by ulcerative colitis.



A1 = 106

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	106	101	108	105	102	96	99	97	98	105	104	97	98
2	104	105	109	99	100	108	101	108	111	104	99	102	112
3	106	98	103	105	106	100	112	108	104	104	108	114	100
4	104	102	109	104	107	112	109	108	111	114	112	106	107
5	107	99	104	107	112	115	118	116	112	105	104	109	104
6	100	111	124	113	105	108	123	113	104	109	113	107	101
7	100	109	111	102	99	105	112	111	108	107	108	109	104
8	107	97	93	100	108	110	110	116	121	122	118	114	109
9	94	97	97	97	102	108	109	117	127	130	120	106	107
10	106	98	102	117	122	114	119	116	113	112	112	112	107
11	111	113	117	118	112	103	112	118	121	118	115	114	114
12	124	118	112	110	113	116	119	127	128	120	116	120	122
13	115	112	112	113	116	116	121	114	116	119	115	114	122
14	111	113	113	113	115	119	115	116	121	123	119	113	116
15	112	115	116	116	116	118	115	121	118	113	115	112	113
16	110	113	114	111	109	109	113	115	110	112	124	122	[illegible]
17	117	120	120	116	110	107	118	114	111	115	118	112	

AVERAGE: 146.2243291 COUNT: 65025 SUM: 9508237

Figure 2.
Extracted image data in Excel.

4. Image visualization

Various chart options in Microsoft Excel can be used to visualize the image data once the data are available in Excel. With the desired dataset selected, the appropriate chart option can be accessed via the INSERT Ribbon. [16, 17] identified the 3D surface and contour as the preferred or effective depictions of the two-dimensional image data of the R, G, and B bands and further processed data. Both charts have a spatial dimension of 255 categories on the x-axis and 255 data series on the y-axis, and a spectral dimension of 256 pixel values (i.e. 0–255 gray levels) on the z-axis for the 3D surface chart. The pixel values (signal intensities) are displayed as different colors in the contour chart. However, in order to reduce the dimensionality of the data and the complexity of image processing, the 256 pixel values are displayed in only a few colors. However, once features in the image have been identified, the data can be further processed to reduce its dimensionality.

The one-dimensional datasets of the R, G, and B band which are shown in the UC worksheet in Excel and their derived datasets can be visualized in two ways. First, histograms of each dataset can be obtained utilizing the Data Analysis ToolPak. The Data Analysis ToolPak is an optional add-in tool in Excel. Second, similarly as described in the preceding paragraph, the X Y (scatter) chart can be accessed to produce cluster plots of any two datasets (columns). This can be useful for feature extraction and unsupervised classification purposes.

Figure 3 shows 3D surface charts, contour charts, and histograms (rows 1, 2, and 3, respectively) of R, G, and B band data (columns 1, 2, and 3, respectively) from **Figure 1a**. The 3D surface and contour charts were set to display only six colors with the data range. These charts show variations in the skin condition, however, the features are not very well separated. Also, as expected, the nature of the data is clearly different among the three bands due to differences in signal responses of different features.

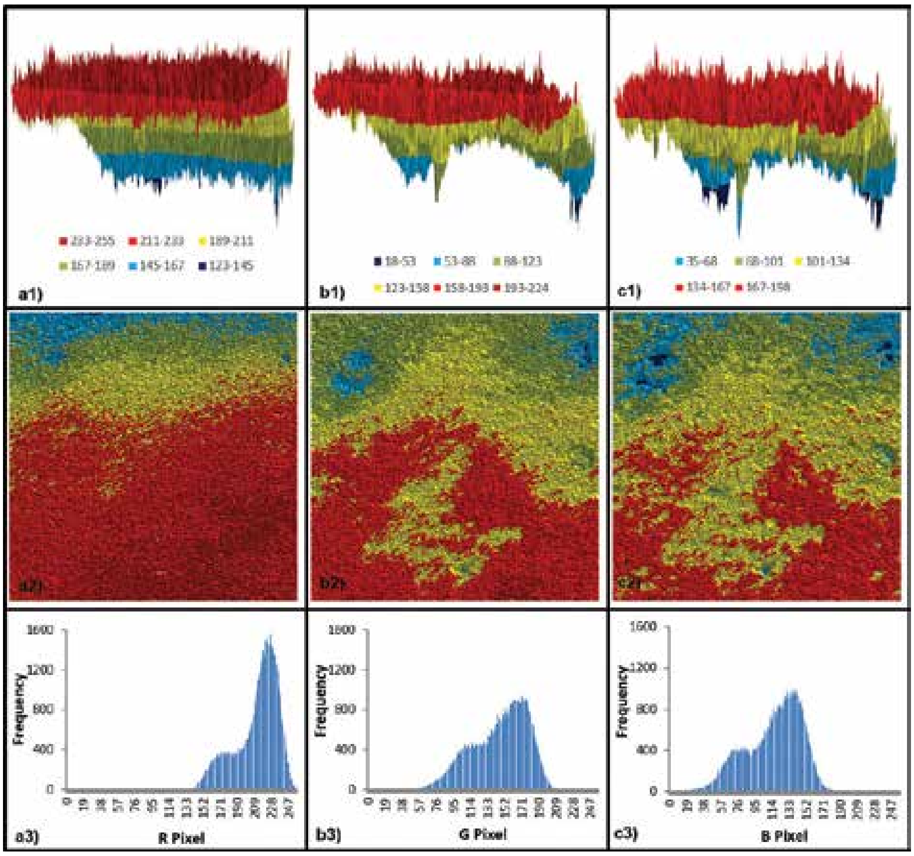


Figure 3.
Various visualizations of R, G, and B data: columns (a = R data; b = G data; and c = B data) and rows (1 = 3D surface; 2 = contour chart; and 3 = histogram).

5. Image enhancement

Image enhancement methods can be applied to biomedical image data to more effectually render the data for subsequent analysis and interpretation. The selected algorithm may manipulate: the contrast of a single image; the data of multiple images at the same time; or the spatial features in an original or derived image [25]. The output image data tends to have intensified visual distinctions that are useful for both digital and visual analyses. As already mentioned, applying image enhancement operations in Excel requires a new output worksheet for resampling, in which the appropriate formula is applied to one cell and then copied to additional cells within the dimension of the original data. The original source data worksheets are retained and referenced in entering the formulas. The procedures used to implement these enhancements have been exemplified in [16, 17].

5.1 Contrast manipulation

Three different forms of contrast manipulation are segmentation (a.k.a. gray level thresholding), level slicing, and contrast stretching. Segmentation generally connotes the division of an image into typically two contiguous regions [5], e.g. foreground versus background. The division is based on a threshold value identified to separate the levels appropriately. Different contrasting colors are assigned to the different classes. The algorithm is extended to obtain more than two levels using multiple threshold values (one less the number of classes), what is termed as level slicing. This could imply splitting an image into various diagnostically and therapeutically relevant areas, for instance, differentiating between anatomically healthy tissues and infected tissues at multiple levels. Thus, segmentation (or level slicing) can be considered as a pre-stage of image classification. It can be used to classify different features including raw pixels, edges, textures, and so on. On the other hand, contrast stretching is applied to image data to spread the data across the full range of 256 gray levels.

Figure 4 provides example contrast manipulation outputs of **Figure 1b** which portray different depictions of allergic rash dermatitis eczema on a human skin. **Figure 4a** is a segmented contrast stretched B band image where the eczema symptoms are shown as the foreground and the clear skin as the background. **Figure 4b** shows a five-level sliced R/G ratio image where the lowest level (lightest gray) represents clear skin while the subsequent darker grays indicate different severities of the symptoms. Multi-image manipulation operations such as the simple ratio of R data on G data are discussed in Section 5.2. **Figure 4c** is a fused contrast stretched false color RGB image where contrast stretching was applied to all three datasets before fusion. **Figure 4d** is a fused contrast stretched monochrome R band image.

To obtain the output in **Figure 4a**, the B band data (min = 2; max = 203) was first stretched to fill a range of 0–255 by using the corresponding function code in **Table 1**. Then, the data was split into two arbitrary values (0.3 and 0.8) for display based on a threshold value of 150 using the corresponding function code in **Table 1**.

The output of **Figure 4b** was accomplished by first obtaining a simple ratio data (min = 1.1; max = 6.8) of R band pixels on G band pixels (simple ratios are discussed in Section 5.2) and then slicing them into five arbitrary values (0.1, 0.3, 0.5, 0.7, and 0.9) for display based on four threshold values of 1.2, 1.4, 1.8, and 2.2. The function codes are shown in **Table 2**.

Figure 4c was obtained by implementing contrast stretching separately to R band data (min = 123; max = 255), G band data (min = 18; max = 224), and B

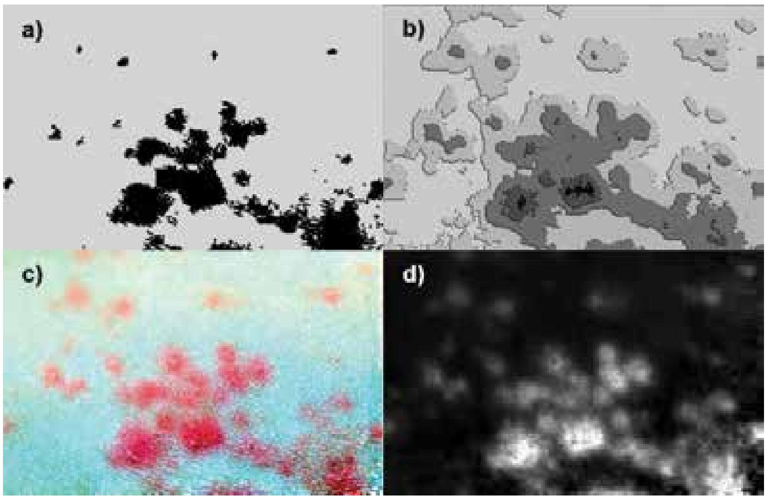


Figure 4.
Contrast manipulation outputs: (a) segmented contrast stretched B band image; (b) level sliced R/G ratio image; (c) fused contrast stretched false color RGB image; and (d) fused contrast stretched monochrome R band image.

Operation	Function code in cell A1 of output worksheet
Contrast stretching	=ROUNDUP(((B!A1-2)/(203-2))*255,0)
Segmentation	=IF(B_CS!A1>150,0.3,0.8)

Table 1.
Example code for implementing contrast stretching and segmentation in Excel.

Operation	Function code in cell A1 of output worksheet
Simple ratio (R/G)	='R'!A1/G!A1
Level slicing	=IF(R_G!A1<=1.2,0.1,IF(R_G!A1<=1.4,0.3,IF(R_G!A1<=1.8,0.5, IF(R_G!A1<=2.2,0.7,0.9))))

Table 2.
Example code for implementing simple spectral ratioing and level slicing in excel.

band data (min = 2; max = 203) and then fusing the three datasets using a modified form of the Visual Basic for Applications (VBA) macro in the Appendix. A similar function code as in **Table 1** was used to accomplish the contrast stretching here. **Figure 4d** was achieved by fusing just the contrast stretched R band data in triplicate (in place of R, G, and B) to obtain a monochrome image. Image fusion here was also accomplished by using a modified form of the VBA macro in the Appendix.

5.2 Multi-image manipulation

By its name, multi-image manipulation means manipulating data of more than one image together. Multispectral-band integrated enhancement techniques include spectral ratioing, integrating biological components, canonical components, and intensity-hue-saturation color space transformations. All these techniques have been discussed by [25, 26], and several others. [16, 17] demonstrated the implementation of spectral ratioing in Excel with an agricultural example. Here, we extend the demonstration with example applications in biomedical sciences.

Spectral ratioing enhancements are accomplished either by correspondingly dividing pixel values of one band by pixel values of another band as in simple ratios, or by dividing the pixel difference of corresponding cells in two bands by the sum of the same pixel values as for normalized difference ratios. These ratioing operations are effective for feature extraction from the image data. An example output image of a simple ratio between the R band pixels and G band pixels (R/G; indicated in **Figure 3**) is displayed in **Figure 5**. The image quantitatively shows the eczema atopic dermatitis symptoms on a skin as was seen in **Figure 1a**. The three areas of the skin showing symptoms indicate different severities. This demonstrates the effectiveness of spectral ratioing in distinguishing between biological tissues of different conditions. A similar output data which was subsequently level sliced to obtain quantitative measures of different severities of allergic rash dermatitis eczema on a skin was shown in **Figure 4b**. The ability to express such variations quantitatively can allow for more precise treatment.

Another example output of R/G spectral ratioing is shown in **Figure 6** for the ulcerative colitis affected colon shown in **Figure 1d**. This results in identification of different tissues particularly capturing the colitis as white (**Figure 6a**). **Figure 6b** was obtained by further segmentation of **Figure 6a** data (min = 1.2; max = 14.3)) based on a threshold value of 2, where the colitis affected tissues showed pixel values less than or equal to the threshold value. The projected area of the image covered by colitis is approximately 21%, which was obtained as a ratio of the number of colitis infected pixels to the total number image pixels times 100.

5.3 Spatial feature manipulation

Examples of spatial feature manipulation techniques include edge enhancement, convolution, and spatial filtering. Edge enhancement methods attempt to achieve low frequency brightness information while at the same time preserving local contrast. These are obtained by combining the original pixel values with a high frequency component image of the same scene [25, 26]. The high frequency component can be created by spatial filtering which involves subtracting a convolved image from its original image. Convolution involves the use of an array of coefficients (e.g. 3 × 3 and 5 × 5) known as masks or kernels to apply a digital number to an output pixel corresponding to the central pixel of the original image. This can

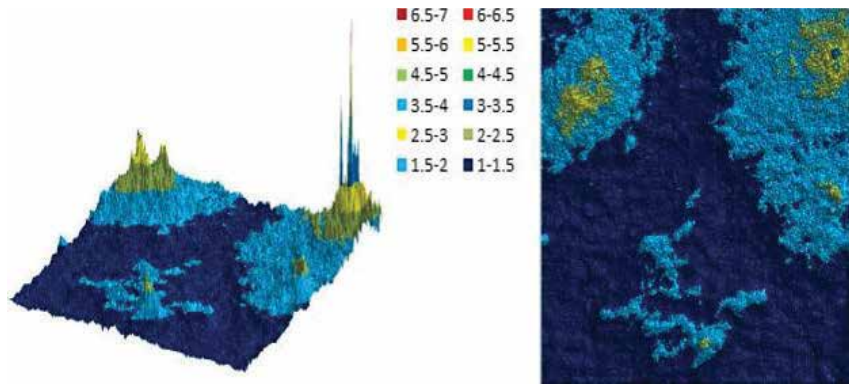


Figure 5.
*3D surface and contour chart outputs of a simple ratio of R band pixels on G band pixels showing eczema atopic dermatitis symptom on skin (***Figure 1a***).*

be implemented by simply finding the average or median [27] pixel value of the original image and entering that in the corresponding central pixel of the output image. The mask is moved to apply the operation throughout the original image to obtain the convolved image. Spatial filtering is done to emphasize or deemphasize image data of varied spatial frequencies (observed tonal variation roughness). Spatial filtering is a special case of convolution and is a pixel neighborhood operation. **Figure 7** shows sequential enhancements of the data of the psoriasis vulgaris skin image which was shown in **Figure 1c**.

Figure 7a is the original image which is shown here again for ease of comparison. **Figure 7b** is an output image of R/G ratioing while **Figure 7c** is an edge

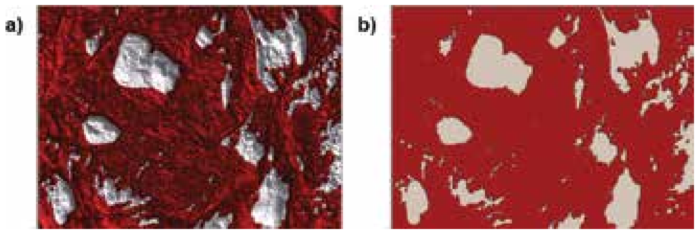


Figure 6.
Ulcerative colitis affected colon: (a) output of R/G ratio; and (b) subsequent segmented image. The colitis is represented by the whitish regions.

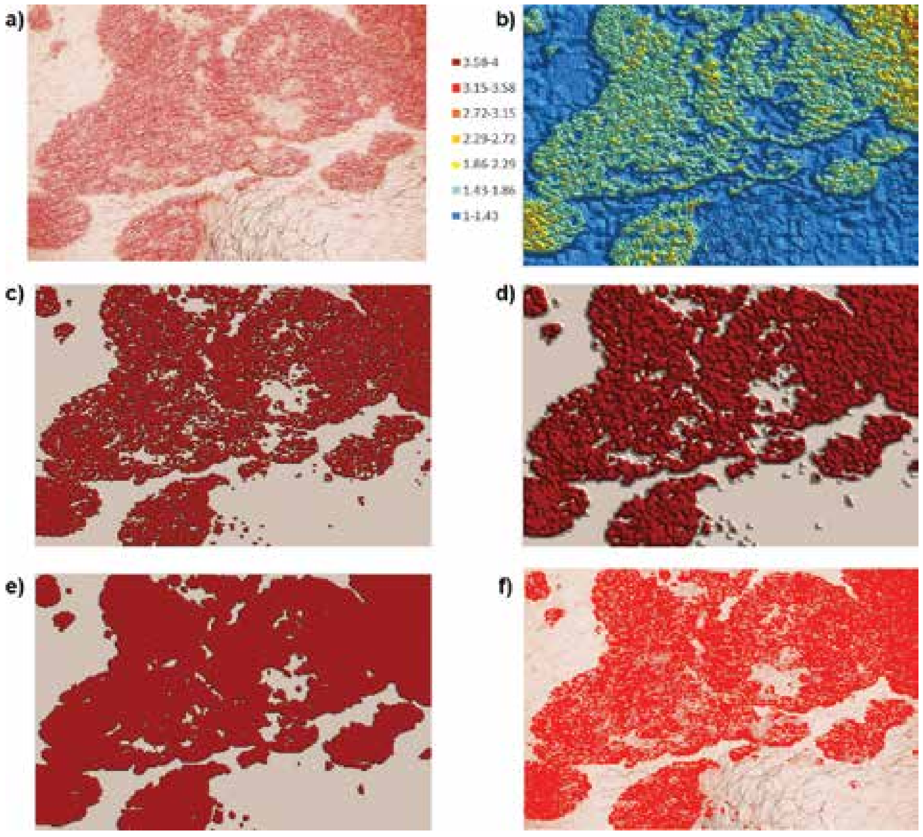


Figure 7.
Psoriasis vulgaris skin: (a) original image; (b) output of R/G ratio; (c) edge enhancement subsequent to 'a'; (d) convolution subsequent to 'b'; (e) segmentation subsequent to 'c'; and (f) true color RGB fusion with edge enhancement.

Operation	Function code in cell A1 of output worksheet
R/G Simple ratio (7b)	='R'!A1/G!A1
Edge detection (7c)	=IF(ABS(R_G!B1-R_G!B2)>=0.1,1,IF(ABS(R_G!A2-R_G!B2)>=0.1,1, IF(ABS(R_G!B3-R_G!B2)>=0.1,0,IF(ABS(R_G!C2-R_G!B2)>=0.1,1,0))))
Convolution (7d)	=AVERAGE(R_G_edge!A1:C3)
Segmentation (7e)	=IF(R_G_edge_conv!A1<=0.45,0.3,0.8)

Table 3.
Example codes for implementing spatial feature enhancements.

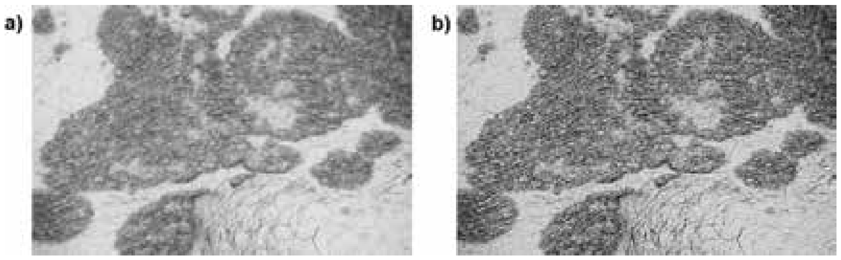


Figure 8.
Psoriasis vulgaris skin monochrome fused images: (a) G data; and (b) edge-enhanced G data.

enhancement applied subsequently. The edge enhancement was obtained by applying a multi-directional pixel differencing method to detect the edges. This was implemented by using a threshold difference value of 0.1 to create a binary data of, say '1' for 'edge' if the absolute difference with at least one value of its neighboring four-directional pixels was greater than the established threshold value and or '0' if otherwise [17] as in **Table 3**. It should be noted that the function code in **Table 3** was applied to cell B2 and copied on only the remaining inner cells as only those are sounded by four neighbors. The function codes were modified for the outer cells (pixels). [28, 29] provide more details on multi-directional pixel value differencing. **Figure 7d** shows the output image of a 3 × 3 window convolution operation performed subsequent to the edge enhancement. Similarly to edge detection and for the same reason, the function code shown in **Table 3** was applied to cell B2 and copied onto the remaining inner cells. **Figure 7e** was obtained as subsequent segmentation. Finally, **Figure 7f** is a true color RGB image with edge enhancement obtained by data fusion using the VBA macro in the Appendix.

Figure 8 demonstrates the implementation of spatial filtering in the G band data for the psoriasis vulgaris skin image shown in **Figure 1c**. **Figure 8a** is simply the fused image of the G data where only the G data was used for the R, G, and B components (see Appendix) to obtain a monochrome image output. This output image prior to the spatial filtering is not very sharp. **Figure 8b** was obtained after applying the spatial filtering. To accomplish this, the data was first convolved using a 3 × 3 window filter and the output data was subtracted from the original data. Finally, the result of the subtraction was added back to the original data. The final output image is edge enhanced and thus is sharper.

6. Further discussion on potential and limitation

The use of Microsoft Excel for implementing biomedical image processing and analysis methods as demonstrated in the preceding sections is a straightway possibility. This possibility is mainly because each blank worksheet acts as a fresh canvas where the cells are available for performing pixel by pixel computations or can be colored up to 16,777,216 colors based on RGB input data. Excel computations are also generally straightforward with some familiarity with entering formulas making use of common mathematical functions. Therefore, with the data available in Excel, already existing features and tools can be used to perform manipulations on the data. Further, as a general use spreadsheet program, Excel has a well-respected performance with data handling and now presents 2D and 3D graphing capabilities that are comparable to other image processing applications. Specifically, 3D surface and contour charts are two effective ways of visualizing the two-dimensional image data in Excel. These charts can be copied into other Microsoft Office Suite applications where they can further be directly formatted. This means that the final chart format does not need to be achieved in the Excel file. As an embedded chart in Word or PowerPoint, for instance, they can be shared between users and finalized later without requiring the original Excel file.

Although Excel does not currently have its own automated means of importing image data directly from an image file, the data can be copied from other resources such as [12]. However, a more efficient means of acquiring the image data from multiple files all at the same time is by using the RGB2X 1.0 software application [24]. RGB2X offers the advantages of multiple image file handling and creation of Excel files saved in the same location and having the same base name(s) as the original image file(s). Also, an additional worksheet in the Excel files having unidimensional forms of the R, G, and B data makes four data worksheets which can be used for performing certain analysis like histogram and clustering.

As a limitation, the RGB2X software application by design scales down images larger than 255 × 255 pixels during the data extraction process. This is to allow for viewing the data as charts in Excel. However, the reduced resolution may influence the quality of the raw and processed data. Also, solely using Excel's standard features for image processing and analysis can retard the processing speed in high throughput applications. Furthermore, re-exporting RGB-type images of output data as standard image files is limited compared to advanced professional image processing software. Nevertheless, the limitations of Excel's standard features and tools and that of re-exporting RGB-type images can be eliminated by including some VBA programming in Excel as depicted with image data fusion macro in the Appendix.

7. Conclusion

The potential use of Microsoft Excel for digital image processing and analysis was successfully extended to application in biomedical imaging. This potential depends on the ability to easily extract RGB data from a variety of image file formats into Excel which is now established by the use of the RGB2X 1.0 program. However, other resources including some being online may be used in extracting the data and copying into Excel. With the data being available in Excel, the possibilities of accomplishing a wide range of biomedical image processing and analysis in Excel combined with the use of Visual Basic in Applications (VBA) is unlimited. The selection of image processing and analysis methods demonstrated in this paper including the different image enhancement methods only serve as evidence of the enormous possibilities. Other methods would need to be tested in Excel covering a wide range of images captured using different biomedical imaging modalities. This should include using image metrics to compare Excel processed image data with processing based on other software platforms. Also, the limitations of reducing larger images for visualizing as charts in Excel would need to be addressed in future work. Finally, other aspects of image processing and analysis in Excel such as processing time requirement for completing different image processing tasks and image quality should be compared with other image processing software applications for increased understanding of the potentials and limitations of using Excel for biomedical image processing.

A. Appendix

Example image data fusion code (Source: [17]; reused with permission) (**Table A1**).

```
Sub RGBCells()
' RGBCells Macro
'
'SELECT SHEET TO OUTPUT TO
Dim myValue As Variant
myValue = InputBox("An output worksheet will be created. Type the exact name you want to use.", "Output
Sheet Name", "RGB#")
Sheets.Add After:=Sheets(Sheets.Count) ‘Creates a new output worksheet
ActiveSheet.Name = myValue ‘Renames the output worksheet to user-specified name
'RESIZE CELLS
Cells.Select
Selection.ColumnWidth = 0.83 'Changes cell width to 10 pixels
Selection.RowHeight = 7.5 'Changes cell width to 10 pixels
ActiveWindow.Zoom = 15 'Zooms out
ActiveWindow.LargeScroll ToRight:= −1
ActiveWindow.LargeScroll Down:= −1
'FIND ROW AND COLUMN DIMENSION
Sheets("R").Select
Dim sht As Worksheet
Dim LastRow As Long
Dim LastColumn As Long
Range("A1").Select
LastRow = ActiveSheet.Cells(Rows.Count, "A").End(xlUp).Row
LastColumn = ActiveSheet.Cells(1, Columns.Count).End(xlToLeft).Column
'COLOR CELLS
For i = 1 To LastRow
For j = 1 To LastColumn
  Sheets("R").Select 'Switches focus to worksheet containing data for fusion as R
  R = Cells(i, j) 'Assigns cell value as R value
  Sheets("G").Select 'Switches focus to worksheet containing data for fusion as G
  G = Cells(i, j) 'Assigns cell value as G value
  Sheets("B").Select 'Switches focus to worksheet containing data for fusion as B
  B = Cells(i, j) 'Assigns cell value as B value
  Sheets(myValue).Select 'Switches focus to worksheet containing data for fusion as B
  Cells(i, j).Select ‘Selects cell
With Selection.Interior
     .Pattern = xlSolid
     .PatternColorIndex = xlAutomatic
     .Color = RGB(R, G, B) ‘Applies RGB color to selected cell
     .TintAndShade = 0
     .PatternTintAndShade = 0
End With
Next j
Next i
End Sub
```

Table A1.
Example data fusion code used to obtain different image outputs after processing in Excel.

Author details

Peter Ako Larbi[1]* and Daniel Asah Larbi[2]

1 Agriculture and Natural Resources, University of California, Parlier, CA, United States

2 College of Medicine, Howard University, Washington, DC, United States

*Address all correspondence to: palarbi@ucanr.edu

References

[1] Flynn M. Digital Image Processing in Radiology. Detroit, MI: American Association of Physicists in Medicine, Henry Ford Health Systems; 2007

[2] U.S. Department of Health and Human Services. Radiation-Emitting Products: What is Computed Tomography? U.S. Food & Drug Administration. [Online] December 5, 2017. Available from: https://www.fda.gov/Radiation-EmittingProducts/RadiationEmittingProductsandProcedures/MedicalImaging/MedicalX-Rays/ucm115318.htm [Cited: 17 September 2018]

[3] U.S. Department of Health and Human Services. Radiation-Emitting Products: Ultrasound Imaging. U.S. Food & Drug Administration. [Online] August 29, 2018. Available from: https://www.fda.gov/radiation-emittingproducts/radiationemittingproductsandprocedures/medicalimaging/ucm115357.htm [Cited: 17 September 2018]

[4] Li L, Zhang Q, Huang D. A review of imaging techniques for plant phenotyping. Sensors. 2014;**14**:20078-20111

[5] Deserno TM. Fundamentals of biomedical image processing. In: Deserno TM, editor. Biomedical Image Processing. Heidelberg: Springer; 2011

[6] Larbi PA, Green S. Time series analysis of soybean response to varying atmospheric conditions for precision agriculture. Precision Agriculture. 2018;**2018**:14

[7] Uchida S. Image processing and recognition for biological images. Development, Growth & Differentiation. 2013;**55**:523-549

[8] Liew SC. Principles of remote sensing. In: Space View of Asia CD-ROM Tutorial. 2nd ed. Singapore: CRISP, NUS; 2001

[9] Lin C, Lin YL. Apply Excel VBA to terrain visualization. In: Proc. 22th IPPR Conf. on Computer Vision, Graphics and Image Processing. Chi-Tou, Taiwan: IPPR; 2009. pp. 1707-1711

[10] Aravind H, Rajgopal C, Soman KP. A simple approach to clustering in excel. International Journal of Computers and Applications. 7, 2010;**11**:19-25

[11] Waldock J. Applying mathematics to digital image processing using a spreadsheet. MSOR Connections. 2010;**10**:11-16

[12] Johnson Z. Get RGB. [Online] n.d. Available from: https://itg.beckman.illinois.edu/technology [Cited: 17 September 2018]

[13] Vong CN, Larbi PA. Comparison of Processed Images Captured by Different Cameras. ASABE Paper No. 162455510. St. Joseph, MI: ASABE; 2016

[14] Larbi PA. RGBExcel: An RGB image data extractor and exporter for excel processing. Signal and Image Processing: An International Journal. 2016;**7**:1-9

[15] Larbi PA. RGB2X: An RGB Image Data Extract-Export Tool for Digital Image Processing and Analysis in Microsoft Excel. ASABE Paper No. 162460787. St. Joseph, MI: ASABE; 2016

[16] Larbi PA. Advancing Microsoft Excel's Potential for Low-Cost Digital Image Processing and Analysis. ASABE Paper No. 162455503. St. Joseph, MI: ASABE; 2016

[17] Larbi PA. Advancing Microsoft Excel's potential for teaching digital image processing and analysis.

Applied Engineering in Agriculture. 2018;**34**:263-276

[18] Larbi PA, Green S. Time series study of soybean response based on adjusted green red index (AGRI). In: 13th International Conference on Precision Agriculture, Paper No. 2162. St. Louis, Missouri: International Society of Precision Agriculture; 2016

[19] Vemula S. UAS-based remote sensing for weed identification and cover crop termination determination. In: Engineering & Technology. Jonesboro, AR: Arkansas State University, College of Agriculture; 2018

[20] Warden A. Pro scope systems: Refurbished endoscopes, endoscope peripherals & accessories [Online]. Different Types of Endoscopes and What They're Used For. 2017. Available from: http://www.proscopesystems.com/uncategorized/different-types-endoscopes-theyre-used/ [Cited: 26 August 2018]

[21] American Society of Clinical Oncology. Types of endoscopy. Cancer.Net: Doctor-approved Patient Information from ASCO [Online]. 2017. Available from: https://www.cancer.net/navigating-cancer-care/diagnosing-cancer/tests-and-procedures/types-endoscopy/trackback [Cited: 26 August 2018]

[22] Avantes BV. Solutions for Medical & Biomedical Applications [Online]. AVANTES: Enlightening Spectroscopy. Available from: https://www.avantes.com/applications/markets/bio-medical [Cited: 26 August 2018]

[23] iStock by Getty Images [Online]. Available from: https://www.istockphoto.com

[24] Larbi PA. RGB2X: An RGB Image Data Extract-Export Tool for Digital Image Processing and Analysis in Microsoft Excel. ASABE Paper No. 162460787. St. Joseph, MI: ASABE; 2016. DOI: 10.13031/aim.20162460787

[25] Lillesand TM, Kiefer RW, Chipman JW. Remote Sensing and Image Interpretation. 7th ed. Hoboken, NJ: John Wiley & Sons; 2015

[26] Tempfli K et al. Principles of Remote Sensing: An Introductory Textbook. Enschede, The Netherlands: ITC; 2009

[27] Ahmed HSS, Nordin M. Improving diagnostic viewing of medical images using enhancement algorithms. Journal of Computer Science. 2011;**7**:1831-1838

[28] Verma V, Yadav I. An enhanced image steganographic method with high payload capacity using multidirectional block based pixel-value differencing. The International Journal of Innovative Research in Computer and Communication Engineering. 2013;**1**:1888-1896

[29] Dhruw T, Tiwari N. Different method used in pixel value differencing algorithm. IOSR Journal of Computer Engineering. 2016;**18**:102-109

Chapter 3

Reconstruction of Three-Dimensional Blood Vessel Model Using Fractal Interpolation

Hichem Guedri and Hafedh Belmabrouk

Abstract

Fractal method is used in the image processing and studying the irregular and the complex shapes in the image. It is also used in the reconstruction and smoothing of one-, two-, and three-dimensional data. In this chapter, we present an interpolating fractal algorithm to reconstruct 3D blood vessels. Firstly, the proposed method determines the blood vessel centerline from the 2D retina image, and then it uses the Douglas-Peucker algorithm to detect the control points. Secondly, we use the 3D fractal interpolation and iterated function systems for the visualization and reconstruction of these blood vessels. The results showed that the obtained reduction rate is between 71 and 94% depending on the tolerance value. The 3D blood vessels model can be reconstructed efficiently by using the 3D fractal interpolation method.

Keywords: three-dimensional (3D) interpolation fractal, Douglas-Peucker algorithm, iterated function system, blood vessel, retinal images

1. Introduction

Since a long time, the interpolation is used to model and visualize data. Indeed, a way to analyze experimental data is to represent them on a 2D or 3D graph; it fills this gap by adding intermediate points that could simulate a more detailed experiment. Our proposed approach is to achieve a 3D fractal interpolation, more adapted to many natural phenomena. We use the fractal interpolation to perform the image restoration. The principle of this technique is explained hereafter [1–18].

In this research, we are interested in the 3D geometrical models of human organs [2–11]; the three-dimensional reconstruction of the vascular networks presents a major medical interest for the diagnostic and prognostic monitoring of several diseases such as atheromatous disease. However, the disadvantage of three-dimensional reconstruction approaches is too costly in terms of storage capacity and transmission time. The objective of this research is to create a technique that allows generating with very little information at the beginning, a large amount of data with few errors. To accomplish this goal, we propose an original method based on mathematical morphology in order to provide a 3D reconstruction taking into account all the data of the problem [11–18].

A lot of available works focuses on 3D reconstruction by the 3D fractal interpolation. These approaches were used to generate a 3D model with a high degree of realism.

Guedri et al. [11] have proposed a method for the 3D model blood vessel reconstruction by using the 3D *iterated function* systems (IFS) and 3D fractal interpolation, and the approach is within the framework of 2D/3D techniques. Sun [12, 13] has presented the principles of the multifractal interpolation surface, and he described the methods of dividing local interpolation neighborhoods and determining multiple vertical compression ratios. Then, he gave a practical MATLAB program for the multifractal interpolation surface, and he explained the main parameters in his proposed program. Chen and Bi [14] have created fascinating fractal scenes by using the 3D IFS, and they proposed an efficient coloring, lighting, and mist effect scheme. Guérin et al. [15] have used a fractal model called projected iterated function system (IFS) model that allows the extension of the iteration space to a barycentric space R^{n2} by enriching the classical IFS model with a set of control points for approximating smooth or rough surfaces defined in R^3. He et al. [16] have proposed a probability-based method to speed up the fractal interpolation execution to three-dimensional terrain reconstruction by a few sparse points in the digital elevation model (DEM). Huang and Chen [17] have adopted two sets of real-world 3D terrain profile data to precede data reducing and then reconstruct them through 3D fractal reconstruction. Xiong [18] has analyzed the reasons that fractal can be used in 3D terrain surface reconstruction and introduced a fractional Brownian motion. Then, they presented an interpolating algorithm to reconstruct 3D terrain surface.

The chapter is organized as follows. In the first step, we give the image source and introduce its features. Then, we present preprocessing of the 2D image datasets and the 3D reconstruction model. Then, a linear simplification method is provided, the Douglas-Peucker method, in order to detect the control points and reduce the data. Thereafter, we develop a reconstruction algorithm which adds new data points by 3D fractal interpolation; this algorithm generates more data points to restore the 3D original model. To evaluate the results of the fractal interpolation method, some blood vessel samples extracted from a real retinal image are used to make the fractal interpolation.

2. Materials and methods

2.1 Algorithm

The aim of this research is to reconstruct a 3D blood vessel model from a retinal image by using the 3D fractal interpolation. **Figure 1** illustrates the different steps of our approach.

1. First, after giving the source of the used images, we will present the different vessel process detections (preprocessing of the 2D image):

    a. Binarization image

    b. Centerline detection

    c. Point classification

2. Then, we will detect the characteristic points by using the Douglas-Peucker algorithm.

3. Subsequently, we will present the 3D reconstruction model.

4. Finally, the fractal interpolation will be presented.

2.2 Image source

There are several databases of available retinal images. Most popular databases used for blood vessel segmentation approach and containing low-resolution images are the Digital Retinal Images for Optic Nerve Segmentation (DRIONS) Database [19], Structured Analysis of the Retina (STARE) [20], and Digital Retinal Images for Vessel Extraction (DRIVE) [21]. On the other hand, there are other databases such as MESSIDOR Digital Retinal Images (MESSIDOR) [22], Retinal Identification Database (RIDB) [23], Glaucoma Database (DB) [24], and high-resolution fundus (HRF) with higher resolutions [25]. In this chapter, we used images illustrated from the databases STARE and HRF.

2.2.1 High-resolution fundus (HRF)

The database contains three image sets: 15 images of healthy patients, 15 images of patients with diabetic retinopathy, and 15 images of glaucomatous patients. All images are captured using a fundus camera CANON CF-60UVi equipped with CANON EOS 20D digital cameras; the capture angle is 60° (FOV). The resolution is 3504×2336 pixels. In addition, this database provided binary gold standard vessel segmentation for each image. Also the masks determining the field of view (FOV) are provided for particular datasets [25]. **Figure 2** shows an example of an image from the database.

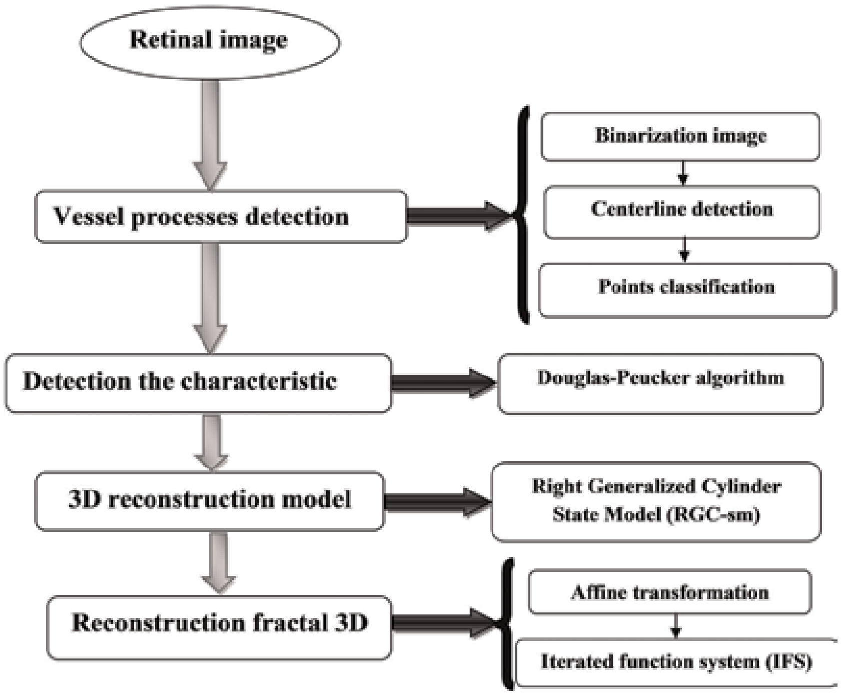

Retinal image
Vessel processes detection
Binarization image
Centerline detection
Points classification
Detection the characteristic
Douglas-Peucker algorithm
3D reconstruction model
Right Generalized Cylinder State Model (RGC-sm)
Reconstruction fractal 3D
Affine transformation
Iterated function system (IFS)

Figure 1.
3D fractal reconstruction algorithm.

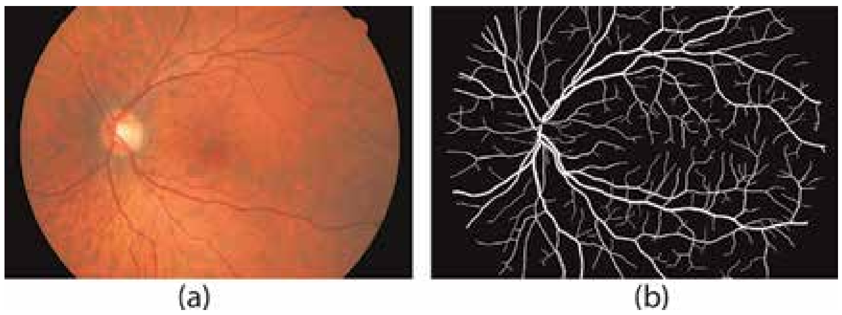


Figure 2.
Example fundus image from high-resolution fundus (HRF). Database: (a) raw image 02_h.jpg and (b) the manual segmentation of the blood vessels 02_h.tif.

2.2.2 Structured analysis of the retina (STARE)

We use the human retina images from the STARE database [20]. The fundus images from this database are acquired by a retinal camera TopCon TRV-50 type with a field of view 35° and which consist of 605 × 700 with 24 bits/pixel. **Figure 3a** shows an example of a raw image im0139.ppm, and **Figure 3b** shows vascular networks marked by hand provided by Valentina Kuznetsova, im0139.ah.ppm [26, 27].

2.3 Vessel process detection

In the first step, our approach starts with the extraction of vessel centerlines, i.e., the preprocessing phase.

2.3.1 Binarization phase

The binarization phase is used to convert the gray image to a binary image using the thresholding technique [28–32]. The intensity Io (x, y) of the binary image is transformed using the following equation:

$$I_o(x,y) = \begin{cases} 1 & I_i(x,y) > \text{Threshold} \\ 0 & \text{Otherwise} \end{cases} \quad (1)$$

where Ii (x, y) is the intensity of the initial image.

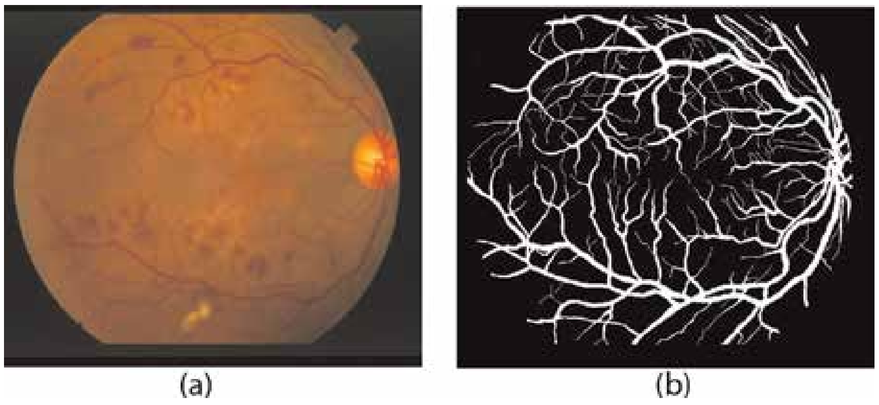


Figure 3.
Example fundus image from the STARE database. (a) Example raw retinal image from the proposed database im0139.ppm. (b) The manual segmentation of the vessels im0139.ah.ppm.

2.3.2 Retinal vessel centerline detection

One of the very useful applications in image processing and in shape recognition is skeletonization. The skeletons represent interconnected lines at the center of an object. They do not only describe the shape but also some mathematical properties of objects, such as length or surface [33–35]. The skeletons of the blood vessels are produced by a thinning technique [35–38]. In this study, the skeleton makes it possible to represent in a compact manner the properties of the blood vessels of the human retina in its particular shape. Its purpose is to describe each vessel by a set of infinitely fine lines. To achieve this objective, we have used the thinning method; the basic idea of this method is to reduce an object iteratively. The algorithm of this method consists of two main steps: the first step is detecting the contour of the object, and in the second step, the contour points are deleted only if they do not change the object's topology.

2.3.3 Feature point extraction

In order to extract the shape characteristic points of the curve, we used a matrix 3×3, and we calculate the point neighbor numbers (PNN) of a pixel Pi containing the information (Pi and the neighboring points must be equal to 1) [39–42]:

- If PNN = 0, P is an isolated point.
- If PNN = 1, P is an endpoint.
- If PNN = 2, P is a connection point.
- If PNN = 3, P is a bifurcation point.
- If PNN $\geq$ 4; P is a crossing point.

2.4 Determination of the control points

Instead of using all the points of blood vessel centerline, it is worthwhile to be contented only with the control points.

Several algorithms are available to archive this reduction [43–46]. In this work, we will use the Douglas-Peucker algorithm since it gives the best results according to the works of White [43]. This algorithm is still used in many applications.

The Douglas-Peucker method is an automated algorithm, and its principle is given by the following steps:

We connect the start pixel to the end pixel of the blood vessel curve (**Figure 4a**) by a straight line (denoted by S; see **Figure 4b**).

Then, we compute the perpendicular distance (PD) of each point of the curve to the line S (**Figure 4b**).

We compare PD with tolerance ε, and we determine the maximal perpendicular distance MPD (**Figure 4b**).

If MPD > ε, then we take this point as a starting point of the new interval and repeat step 1 (**Figure 4b** and **c**).

This process is repeated until all the distances become smaller than a tolerance value ε (**Figure 4b–d**, and **f**).

Figure 4 illustrates the evolution of the progressive determination of the control points.

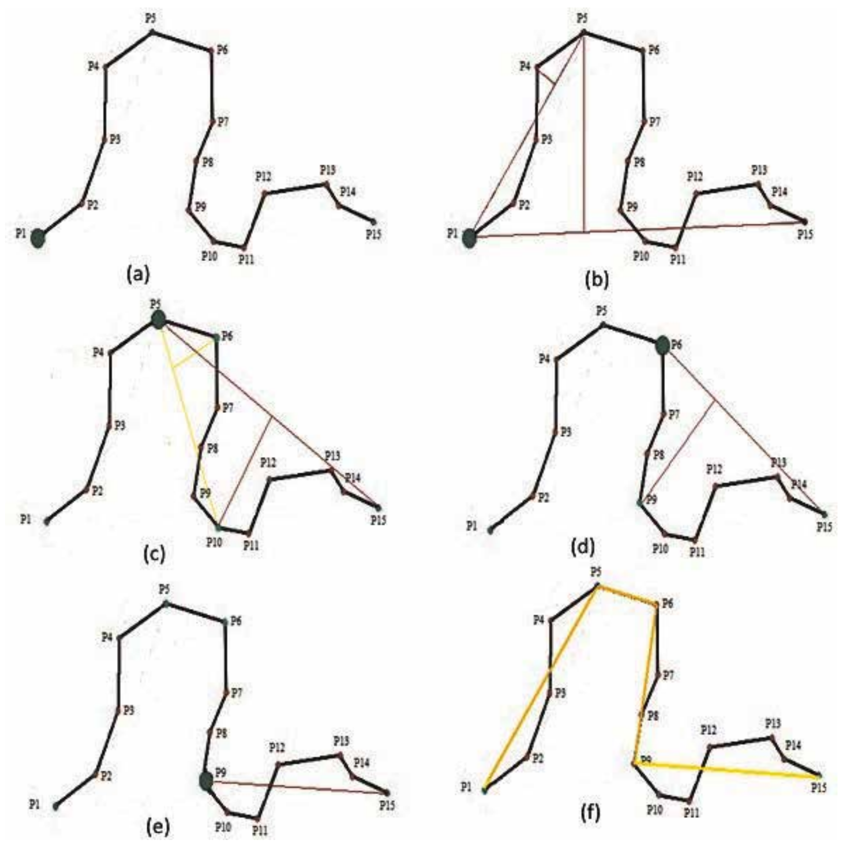


Figure 4.
Douglas-Peucker algorithm.(a) Initial curve, (b) the connection between the two end points of the curve with a line (S) and and the determining the max perpendicular distance, (c), (d) and (e) detection of the sub-new interval and repeat the same procedure, (f) Final result achieved.

2.5 3D reconstruction

The proposed method relied on on the natural shape of the blood vessel, it has a deformable cylindrical shape, which can be represented as a combination of an axis and a surface [8, 47, 48]; more precisely, we used the approach the right generalized cylinder state model (RGC-sm); then to create the cylindrical shape, we need a set of successive 3D circles with its axes being the blood vessel center (as shown in **Figure 5**). The RGC parametric equations include a coupling of an ordinary axis H

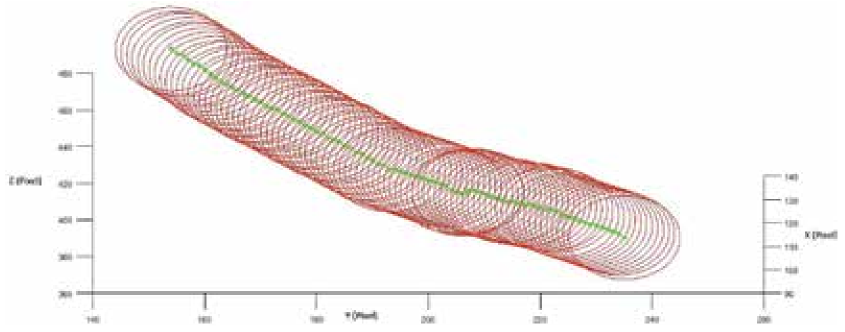


Figure 5.
Representation of the cylindrical surface with given axis.

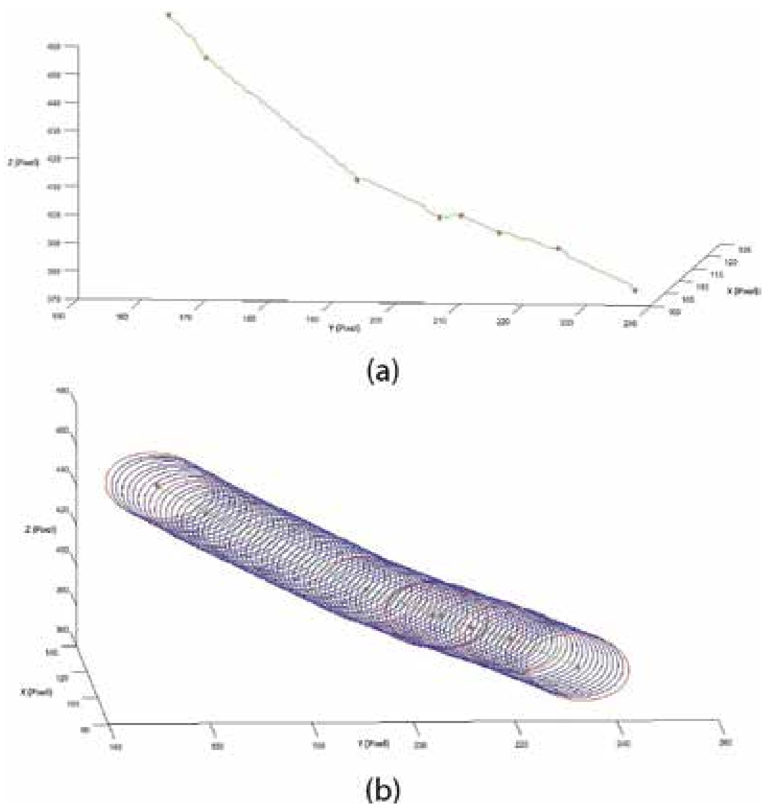


Figure 6.
Detection of the control point in 3D surface. (a) Determination of the axis control points. (b) Determination of the control circles.

and a regular surface S (radius parameter rand w is the surface azimuthal parameter) to represent the cylinder as a volumetric object:

Figure 6a shows the control points obtained for the blood vessel represented in **Figure 5**.

It is clear that the number of control points is greatly reduced, and therefore the image transmission time will be reduced with a limited information loss.

Actually, the 3D reconstruction is not performed to all the centerline, but it is limited only to the control points (**Figure 6b**).

2.6 Fractal interpolation

The principle of fractal interpolation studied by Barnsley [49, 50] is to construct a continuous fractal function, passing by a number of giving points of the shape $\{(x_n; F_n)$: n = 0, 1, 2,..., N$\}$ with x0 < x1 < x2 < ... < xN. An interpolation function corresponding to this data set is a continuous function f: [x0; xN] passing with the interpolation points $(x_n; F_n)$ and checking $f(x_n) = F_n$ with n = 0, 1, 2,..., N. In this section, we will study constructions of 3D IFS, whose attractors, f, are the graphs of continuous functions, with affine transformation W_n [50–53]. We use the same principle to reconstruct our 3D model by using the 3D fractal interpolation method.

2.6.1. Affine transformation

The affine transformations affect the rotations, translations, scaling, and shear of a data set. He generates each point in a space R^n to another point in the R^n. The affine transformation contains two major parameters, A and t, where he represented in the AP + T form. The affine transformations on a data set in 2D space are is defined by the following equation [50–53]:

$$A = \begin{pmatrix} a_i & b_i \\ c_i & d_i \end{pmatrix}, T = \begin{pmatrix} e_i \\ f_i \end{pmatrix}, \text{and } P_i \begin{pmatrix} x_i \\ y_i \end{pmatrix} \text{ the data set of control points} \tag{2}$$

The equation system provides five equations for five parameters, so d_i, the vertical scaling factor, is computed by using the fractal dimension (DF) calculated by the box-counting algorithm [54, 55]. We can solve the above equations for a_i, c_i, d_i, e_i, F_i which are defined as

$$\begin{aligned} a_i &= \frac{x_i - x_{i-1}}{x_N - x_1} \\ c_i &= \frac{y_i - y_{i-1}}{x_N - x_1} - d_i * \frac{y_N - y_1}{x_N - x_1} \\ d_i &= (N-1)^{DF-2} \\ e_i &= \frac{x_N x_{i-1} - x_1 x_i}{x_N - x_1} \\ f_i &= \frac{x_n y_{i-1} - x_1 y_i}{x_N - x_1} - d_i * \frac{x_N y_1 - x_1 y_N}{x_N - x_1} \end{aligned} \tag{3}$$

where N is the affine transformation map number.

2.6.2. Moving from 2D to 3D

In the 2D affine transformation, it only requires a single equation in AP + t form. On the other hand, the 3D affine transformation composes two 2D affine transformations, where this is represented by W1(P) = A1P+ t1 and the other is represented by W2(P) = A2P + t2; all the elements of a 2D transformation matrix could be solved and used to generalize the 3D transformations. Indeed, 3D space (x-y-z) is divided into two 2D spaces (x-y space and x-z space). To determine the 3D affine transformation parameters, we calculate, at first, the parameters of W1 in the space x-y. For that, the parameter bi (bi = 0) is eliminated, so that the y-axis has no term of rotation. The function W1 is defined as follows:

$$P' = W_1(P) = \begin{cases} x'_i = a_i x_i + e_i \\ y'_i = c_i x_i + d_i y_i + f_i \end{cases} \tag{4}$$

Similarly for the second affine transformation W2 in the space x-z, we use the same approach:

$$P' = w_2(P) \begin{cases} x'_i = a_i x_i + e_i \\ z'_i = c_i x_i + d_i z_i + f_i \end{cases} \tag{5}$$

2.6.3. Iterated function system (IFS)

Considering W_N is the 3D affine transformation (W_1,..., W_n), we select an initial set circle which can be selected at random; then we compute iteratively the new

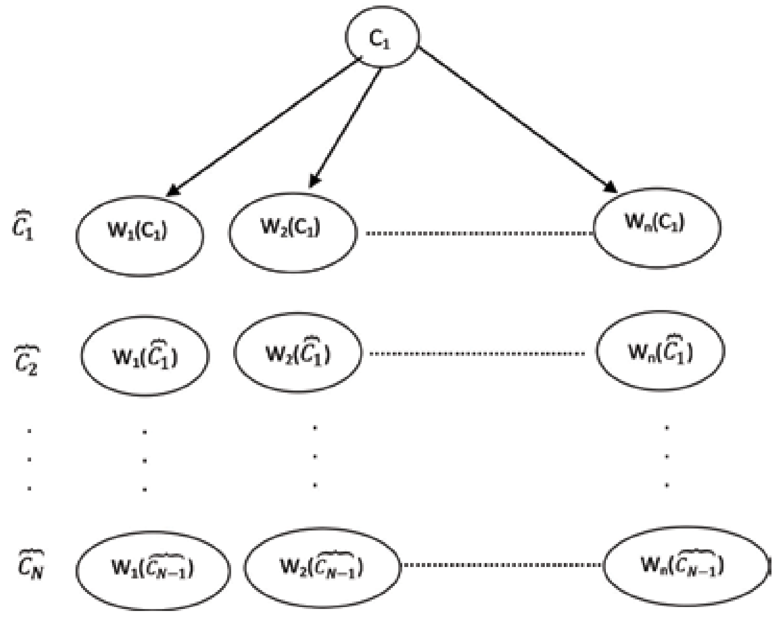


Figure 7.
Constructions by iterated function system (IFS) method.

data sets. In practice, we start with an the initial circle set and a whole group of affine functions to generate the first set circle. Then, using this circle set and a whole group of affine functions to generate the second circle set, we continue to generate circle set until the generated circle set conforms the requirements of the desired shape, as can be seen in **Figure 7** [53].

3. Results

In order to test the method presented in this work, a demo has been designed and coded in MATLAB. This program has been run on a workstation equipped with an Intel Pentium B960 CPU at 2.20 GHz and 4GB of RAM processor. The reconstruction time takes into account the time required for image segmentation and fractal interpolation.

3.1. Image segmentation

Figure 8 provides two examples of blood vessel skeletons; they are created from the retinal images shown in **Figures 2** and **3**. These images have been transformed to binary image using thresholding algorithm (**Figure 8a** and **c**). Then, the thinning technique is used to produce a blood vessel skeleton that preserves all geometric and topological features of these vessels (**Figure 8b** and **d**).

After performing the skeleton of the retinal vasculature, we use the proposed method for the detection and extraction of the feature points. **Figure 9** illustrates an example for the detection of endpoints and bifurcation points. In these examples, the red pixels correspond to the endpoints, and the blue pixels correspond to the bifurcation points.

In addition, **Table 1** presents some typical results of endpoints and bifurcation point detection for two test images (image from STARE database and image from

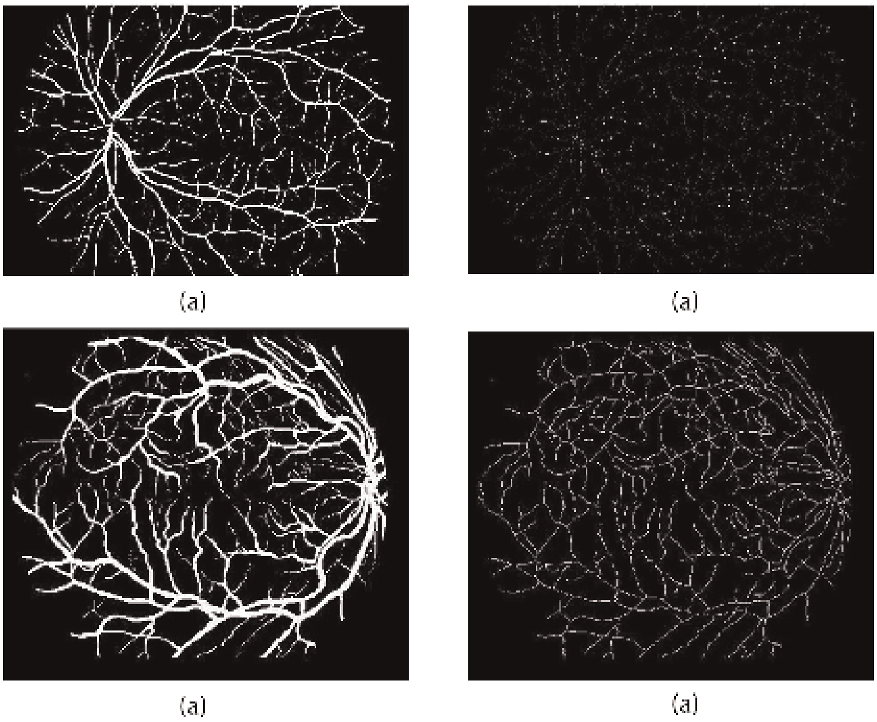


Figure 8.
(a) Result of the binary morphology for image 02_h.jpg. (b) The resulting skeleton of the image 02_h.jpg. (c) Result of the binary morphology for image im0139.vk.jpg. (d) The resulting skeleton of the image im0139.vk.jpg.

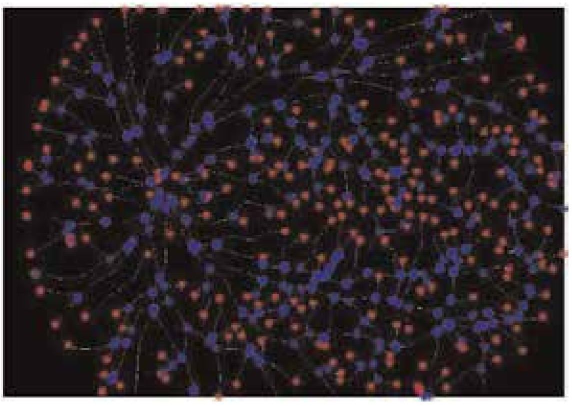


Figure 9.
Minutiae detection (the red pixels show the endpoints, and the blue pixels show the bifurcation points).

Database	Images	Endpoints	Bifurcation points
STARE	im0139.vk.jpg	200	450
HRF	02_h.jpg	296	982

Table 1.
Detection of endpoints and bifurcation points for image from high-resolution image database (HRF) and image from STARE database.

HRF database). According to the results, we can see that the images from HRF database give a number of endpoints and bifurcation points higher than the images from the other used database; it is due to the image quality and the presence of a higher number of blood vessels in these images.

3.2. Douglas-Peucker algorithm

In the first step, the aim is to compress data by using the Douglas-Peucker method to calculate the data-reduction rate (DR). We use the following equation:

$$D_R = \frac{(P_O - P_R)}{P_O} \tag{6}$$

where P_O is the number of the original data points and P_R is the number of the reduced data points.

Table 2 summarizes the results obtained by the Douglas-Peucker algorithm. The simplification tolerance is given in line "ε." We also show the simplification rate in % for each value of ε.

The results in **Table 2** show that the rate of simplification exceeds 71% and can reach up to 95%. This simplification rate increases proportionally with the tolerance value "ε." For example, for tolerance values between 0.5 and 2, the simplification rate for the image "02_h.tif" is between 72 and 95%. For the same values of ε and for the image "im0139.vk.jpg," the rate of simplification is between 71.43 and 93.35%. It could be noted that the first image simplification rate is slightly higher than that of the second image. This is due to the complexity of their geometry of the blood vessel curves.

Figure 10 shows a result of control point test for a tolerance ε = 0.5. This figure shows clearly that the number of original points (blue points) is larger than the number of control points (red points).

ε (pixels)	**0.5**	**1**	**1.5**	**2**
02_h.tif	72.2%	89.86%	93.6%	95.05%
im0162.ah.jpg	71.43%	87.23%	91.95%	93.35%

Table 2.
Simplification rate for different tolerance values ε.

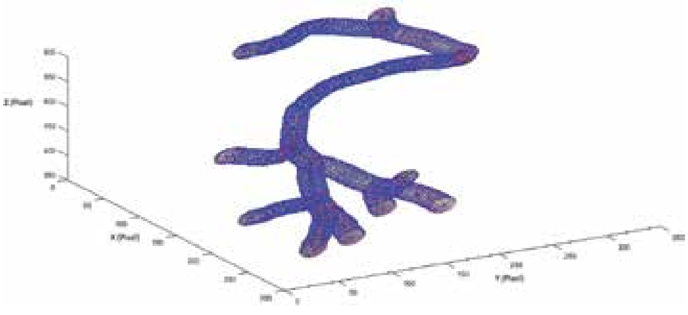


Figure 10.
Determination of the control points (blue point, 3D original model; red point, control points).

3.3. 3D interpolation fractal

The control point's density is large in the regions having a large curvature. The original shape of the blood vessel is preserved. These control points are used to reconstruct the 3D blood vessel via fractal interpolation. The affine transformations W_n are performed out. The execution time, the number of interpolated points, and the error are calculated versus the iteration number.

In addition, to analyze the performance of our proposed interpolation algorithm, we calculate the error rate between the original image and the interpolated image in the equation below:

$$Er(\%) = \frac{NE\,P}{TP} \times 100 \tag{7}$$

where *NEP* is the number of erroneous pixels and *TP* is the total pixel. The number of erroneous pixels is the pixel numbers that do not appear in the interpolated image and the total pixels are the numbers of pixels in the original image.

Table 3 shows the evolution of the iteration number, as well as the error between original and interpolated image and the number of iterations for two test images used; for a small iteration number (N = 50), the error is about 7%. Moreover, For N= 400 iterations, the error decreases to 2.3% for the first image 12 (im0139.vk.jpg) and 2% for the second image used (02_h.tif) For N = 1000, the minimum error rate (0.3%) is obtained from the two test images.

The algorithm described in this document takes a lower time of 2 s to find the curve for a blood vessel and the 3D reconstruction. Moreover, the remnants of execution times are devoted to 3D fractal interpolation. The execution times were between 4 and 14 s. For a small iteration number (N = 50), the execution times is about 4 s. By looking at **Table 3**, we can see that the value of the execution time increases in the last three tests; this is justified by the increased number of interpolated point, for example, for N = 1000 iterations, the execution time is about 14 s.

Increasing the number of iterations reduces the error but dramatically increases the number of interpolated points and execution times. A trade-off should be looked from these two parameters.

Image	N° iteration	50	100	200	400	600	800	1000
02_h.tif	Error in %	6.52	5.11	4.02	1.89	1.08	0.66	0.25
	Execution time (s)	3.9	6.2	7.1	7.9	9.8	11.2	13.7
im0139.vk.jpg	Error in %	6.71	5.41	4.30	2.26	1.28	0.86	0.27
	Execution time (s)	3.7	5.7	6.8	7.5	9.3	10.7	13.1

Table 3.
Performance evaluation for the image "im0139.vk.jpg" and image "02_h.tif".

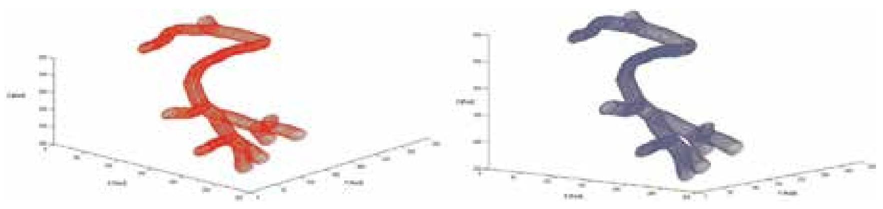


Figure 11.
(a) 3D models reconstructed. (b) The original models.

Figure 11a shows the test experimental results for the reconstructed blood vessel using 3D fractal interpolation with ε = 0.5 and N = 1000 iterations from image im0139.ah.ppm with different arc lengths (from 50 to 385 pixels); **Figure 11b** depicts the original 3D blood vessels.

The obtained results are qualitatively similar and quantitatively more than the original data.

4. Discussion

The aim of this work is not only to make a 3D reconstruction for visualization but also to describe the geometric shape-based fractal interpolation. The benefits of the proposed method to perform 3D reconstruction by 3D fractal interpolation method lies mainly in reducing storage memory for 3D blood vessel images. Indeed, the data-reduction rate ranges from 72 to 94% for the first test image and between 73 and 95% for the second image used, when the tolerance value varies from ε = 0.5 to 2 pixels. This allows us to reduce costs and transmission time. Another advantage in terms of execution time is obtained. This execution time range is varying from 4 to 14 s, according to the iteration number. Furthermore, an accurate form of description is reached since the algorithm does not only interpolate the original points but also determines the interconnection between these points. However, the weakness of our algorithm is the space complexity of algorithm calculation. It requires a specialized computer to minimize the execution time.

In previous works, the result obtained by Guedri et al. [11] offers an innovative approach for 3D reconstruction with fractal calculation time between 4 and 40 s. The execution time of the present method is about 4–14 s for the same number of iterations. Thus, we can note that the present method is faster than other methods.

Regarding the error evolution of the curvature by using the 3D fractal interpolation, Guedri et al. [11] obtained an error between 0.8 and 8%. The new method is indeed better than the other method in terms of error. The error value is between 0.27 and 6.7% and between 0.25 and 6.5% for the first test image and for the second test image, respectively.

This comparison shows that our proposed method can correctly follow all the 3D branches of the blood vessels and reduces the execution time while having a minimal error.

5. Conclusion

In this work, we propose a new method for reconstructing a 3D vascular tree model from the human retina image by fractal interpolation. Firstly, we describe the method to extract the human retina vascular tree, and then we present a 3D reconstruction algorithm of the blood vessel. In the second step, we applied the Douglas-Peucker algorithm of simplification to detect control points and to compress the data in this 3D model. In the last step, the 3D fractal interpolation and control points are used to generate new data points and restore the original 3D model. From the obtained result, it is found that the Douglas-Peucker method has a high reduction ratio between 72 and 94% for the first test image and between 73 and 95% for the second image. According to the obtained results after the application of the proposed 3D fractal interpolation, we find that the error is small and ranging between 0.2 and 2% for the iteration number greater than 400 iterations. On the other hand, the error increases while decreasing the iteration number. The advantage of this method is that there are more interpolated data points than the original data points; therefore, the reconstruction model is powerful and provides valuable information.

Author details

Hichem Guedri* and Hafedh Belmabrouk
Electronics and Microelectronics Laboratory, Physics Department,
Faculty of Sciences, Monastir University, Monastir, Tunisia

*Address all correspondence to: himougu@yahoo.fr

References

[1] Spiegel M, Redel T, Struffert T, Hornegger J, Doerfler A. A 2D driven 3D vessel segmentation algorithm for 3D digital subtraction angiography data. Physics in Medicine and Biology. 2011; **56**:6401-6419. DOI: 10.1088/0031-9155/56/19/015

[2] Bauer C, Pock T, Sorantin E, et al. Segmentation of interwoven 3D tubular tree structures utilizing shape priors and graph cuts. Medical Image Analysis. 2010;**14**:174-184. DOI: 10.1016/j.media.2009.11.003

[3] Risser L, Plouraboué F, Descombes X. Gap filling of 3-D microvascular networks by tensor voting. IEEE Transactions on Medical Imaging. 2008; **27**(5):674-687

[4] De Bruijne M, Van Ginneken B, Niessen W, Viergever M. Adapting active shape models for 3D segmentation of tubular structures in medical images. In: Information Processing in Medical Imaging. Lecture Notes in Computer Science. Vol. 2732. Berlin/Heidelberg: Springer; 2003. pp. 136-147. DOI: 10.1007/978-3-540-45087-0_12

[5] Jourdain M, Meunier J, Sequeira J, et al. A robust 3D IVUS transducer tracking using single-plane Cineangiography. IEEE Transactions on Information Technology in Biomedicine. 2008;**12**(3):307-314. DOI: 10.1109/TITB.2008.921043

[6] Birkeland A, Ulvang DM, Nylund K, et al. Doppler-based 3D blood flow imaging and visualization. In: SCCG'13 Proceedings of the 29th Spring Conference on Computer Graphics; Smolenice, Slovakia; 2013. pp. 115-122

[7] Hua L, Yezzi A. Vessels as 4-D curves: Global minimal 4-D paths to extract 3-D tubular surfaces and centerlines. IEEE Transactions on Medical Imaging. 2007;**26**(9):1213-1223. DOI: 10.1109/TMI.2007.903696

[8] Guedri H, Chouchene F, Belmabrouk H. 3D model reconstruction of blood vessels in the retina with tubular structure. International Journal on Electrical Engineering and Informatics. 2015;**7**(4):724-734. DOI: 10.15676/ijeei.2015.7.4.14

[9] Martinez-Perez ME, Espinosa-Romero A. Three-dimensional reconstruction of blood vessels extracted from retinal fundus images. Optics Express. 2012;**20**(10): 11451-11465. DOI: 10.1364/OE.20.011451

[10] Fuller AR, Zawadzki RJ, Choi S, et al. Segmentation of three-dimensional retinal image data. IEEE Transactions on Visualization and Computer Graphics. 2007;**13**(6):1719-1726. DOI: 10.1109/TVCG.2007.70590

[11] Guedri H, Malek J, Belmabrouk H. Three-dimensional reconstruction of blood vessels of the human retina by fractal interpolation. Journal of Nanotechnology in Engineering and Medicine. 2015;**6**:0310031-0310035. DOI: 10.1115/1.4032170

[12] Sun H. A practical MATLAB program for multifractal interpolation surface. In: 8th International Conference on Natural Computation (ICNC); Chongqing, China; 2012. pp. 909-913

[13] Sun H. The theory of fractal interpolated surface and its MATLAB program. In: IEEE Symposium on Electrical & Electronics Engineering (EEESYM); Kuala Lumpur, Malaysia; 2012. pp. 231-234

[14] Chen YQ, Bi G. 3-D IFS fractals as real-time graphics model. Computers

and Graphics. 1997;**21**(3):367-370. DOI: 10.1016/S0097-8493(97)00014-9

[15] Guérin E, Tosan E, Baskurt A. Fractal approximation of surfaces based on projected IFS attractors. The Eurographics Association. 2001;**9**(1): 95-103

[16] He W, Niu Z, Liang L. An improved fractal construction on 3D DEM terrain profile. In: IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2009; Cape Town, South Africa; 2009

[17] Huang YM, Chen CJ. 3D Fractal reconstruction of terrain profile data based on digital elevation model. Chaos. Solitons and Fractals. 2009;**40**: 1741-1749. DOI: 10.1016/j. chaos.2007.09.091

[18] Xiong J. The fractal interpolating algorithm of 3D terrain surface reconstruction. Applied Mechanics and Materials. 2015;**734**:526-529. DOI: 10.4028/www.scientific.net/ AMM.734.526

[19] Digital Retinal Images for Optic Nerve Segmentation Database (DRIONS). Available from: http://www. ia.uned.es/∼ejcarmona/DRIONS-DB.h tml [Accessed: 10 February 2017]

[20] Structured Analysis of the Retina (STARE). Available from: http://cecas.c lemson.edu/∼ahoover/stare/index.html [Accessed: February 10, 2018]

[21] Digital Retinal Images for Vessel Extraction (DRIVE). Available from: http://www.isi.uu.nl/Research/Da tabases/DRIVE/ [Accessed: 09 February 2018]

[22] MESSIDOR Digital Retinal Images (MESSIDOR). Available from: http:// www.adcis.net/en/Download-Third-Pa rty/Messidor.htmlindex-en.php [Accessed: 09 February 2018]

[23] Retinal Identification Database (RIDB). Available from: http://biomisa. org/ridb.html [Accessed: 10 February 2018]

[24] Glaucoma Database (GlaucomaDB). Available from: http://biomisa. org/glaucomadb.html [Accessed: 09 February 2018]

[25] High-Resolution Fundus Image Database (HRF). Available online: https://www5.cs.fau.de/research/da ta/fundus-images/ [Accessed: 10 February 2018]

[26] Hoover A, Kouznetsova V, Goldbaum M. Locating blood vessels in retinal images by piece-wise threshold probing of a matched filter response. IEEE Transactions on Medical Imaging. 2000;**19**(3):203-210. DOI: 10.1109/ 42.845178

[27] Ben Abdallah M, Malek J, Azar AT, et al. Automatic extraction of blood vessels in the retinal vascular tree using multiscale medialness. International Journal of Biomedical Imaging. 2015;**1**: 1-16. DOI: 10.1155/2015/519024

[28] Akram MU, Tariq A, Khan SA. Retinal image blood vessel segmentation. In: International Conference on Information and Communication Technologies, ICICT '09; Karachi, Pakistan; 2009

[29] Sezgin M, Sankur B. Survey over image thresholding techniques and quantitative performance evaluation. Journal of Electronic Imaging. 2004; **13**(1):146-168. DOI: 10.1117/1.1631316

[30] Lee SU, Chung SY, Park RH. A comparative performance study of several global thresholding techniques for segmentation. Computer Vision, Graphics, and Image Processing. 1990; **52**:171-190. DOI: 10.1016/0734-189X (90)90053-X

[31] Mendonça AM, Campilho A. Segmentation of retinal blood vessels by combining the detection of centerlines and morphological reconstruction. IEEE Transactions on Medical Imaging. 2006; **25**(9):1200-1213. DOI: 10.1109/TMI.2006.879955

[32] Hantos N, Iván S, Balázs P, Palágyi K. Binary image reconstruction from a small number of projections and the morphological skeleton. Annals of Mathematics and Artificial Intelligence. 2015;**75**:195-216. DOI: 10.1007/s10472-014-9440-8

[33] Bertrand G, Couprie M. Isthmus based parallel and symmetric 3D thinning algorithms. Graphical Models, Elsevier. 2015;**80**:1-15. DOI: 10.1016/j.gmod.2015.05.001

[34] Ritter GX, Wilson JN. Thinning and skeletonizing. In: Computer Vision Algorithms in Image Algebra. Boca Raton/London/New York/Washington, DC: CRC Press; 2001. pp. 174-191

[35] Lam L, Lee S, Suen C. Thinning methodologies—A comprehensive survey. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1992; **14**:868-885. DOI: 10.1109/34.161346

[36] Youssef R, Sevestre-Ghalila S, Ricordeau A. Statistical control of thinning algorithm with implementation based on hierarchical queues. In: International Conference of Soft Computing and Pattern Recognition; Tunis, Tunisia; 2014. pp. 365-370

[37] Li D, Wu X, He X. Depth-based thinning: A new non-iterative skeletonization algorithm for 2D digital images. 2014 IEEE 9th Conference on Industrial Electronics and Applications (ICIEA); Hangzhou, China; 2014. pp. 1193-1197

[38] Couprie M, Bertrand G. Asymmetric parallel 3D thinning scheme and algorithms based on isthmuses. Pattern Recognition Letters. 2016;**76**:22-31. DOI: 10.1016/j.patrec.2015.03.014

[39] Farina A, Kovács-Vajna ZM, Leone A. Fingerprint minutiae extraction from skeletonized binary images. Pattern Recognition. 1999;**32**(5):877-889. DOI: 10.1016/S0031-3203(98)00107-1

[40] Jain AK, Hong L, Bolle R. On-line fingerprint verification. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1997;**19**(4): 302-314. DOI: 10.1109/34.587996

[41] Ratha NK, Chen S, Jain AK. Adaptive flow orientation-based feature extraction in fingerprint images. Pattern Recognition. 1995;**28**(11):1657-1672. DOI: 10.1016/0031-3203(95)00039-3

[42] Rutovitz D. Pattern recognition. Journal of the Royal Statistical Society. 1996;**129**(4):504-530. DOI: 10.2307/2982255

[43] White ER. Assessment of line-generalization algorithms using characteristic point. American Cartographer. 1985;**12**(1):17-27. DOI: 10.1559/152304085783914703

[44] Marino JS. Identification of characteristic points along naturally occurring lines—An empirical study. Cartographica: The International Journal for Geographic Information and Geovisualization. 1979;**16**(1):70-80. DOI: 10.3138/AG00-3264-1Q31-P216

[45] Douglas DH, Peucker TK. Algorithms for the reduction of the number of points required to represent a line or its caricature. The Canadian Cartographer. 1973;**10**(2):112-122. DOI: 10.3138/FM57-6770-U75U-7727

[46] Saalfeld A. Topologically consistent line simplification with the Douglas-Peucker algorithm. Cartography and Geographic Information Science. 1999;

26(1):7-18. DOI: 10.1559/152304097824249010

[47] Azencot J, Orkisz M. Deterministic and stochastic state model of right generalized cylinder (RGC-sm): Application in computer phantoms synthesis. Graphical Models. 2003; **65**(6):323-350. DOI: 10.1016/S1524-0703(03)00073-0

[48] Shafer S. Shadows and silhouettes in computer vision. In: The Kluwer International Series in Engineering and Computer Science. Boston/Dordrecht/Lancaster: Kluwer Academic; 1985

[49] Barnsley MF, Elton J, Hardin D, Massopust P. Hidden variable fractal interpolation. SIAM Journal of Math Analysis. 1989;**20**(5):1218-1242. DOI: 10.1137/0520080

[50] Barnsley MF. Fractal functions and interpolation. Constructive Approximation. 1986;**2**:303-329. DOI: 10.1007/BF01893434

[51] Barnsley MF. Fractals Everywhere. 2nd ed. Atlanta, Georgia: Iterated Systems, Inc.; 1993

[52] Hichem G, Malek J. Extraction of characteristic points and comparison between the fractal and linear interpolation for the reconstruction of retinal vasculature. In: 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT); Sousse, Tunisia; 2012

[53] Chen C, Lee T, Huang YM, Lai F. Extraction of characteristic points and fractal reconstruction for terrain profile data. Chaos, Solitons and Fractals. 2009; **39**:1732-1743. DOI: 10.1016/j.chaos.2007.06.074

[54] Azemin MZC, Hamid FA, Wang JJ, et al. Box-counting fractal dimension algorithm variations on retina images. Advanced Computer and Communication Engineering Technology. 2015;**362**:337-343. DOI: 10.1007/978-3-319-24584-3_27

[55] Ostwald MJ, Vaughan J. Introducing the box-counting method: The fractal dimension of architecture. In: Mathematics and the Built Environment. Vol. 1. US: Springer; 2016. pp. 39-66

Chapter 4

Alzheimer's Disease Computer-Aided Diagnosis on Positron Emission Tomography Brain Images Using Image Processing Techniques

Mouloud Adel, Imene Garali, Xiaoxi Pan, Caroline Fossati, Thierry Gaidon, Julien Wojak, Salah Bourennane and Eric Guedj

Abstract

Positron emission tomography (PET) is a molecular medical imaging modality which is commonly used for neurodegenerative disease diagnosis. Computer-aided diagnosis (CAD), based on medical image analysis, could help with the quantitative evaluation of brain diseases such as Alzheimer's disease (AD). Ranking the effectiveness of brain volume of interest (VOI) to separate healthy or normal control (HC or NC) from AD brain PET images is presented in this book chapter. Brain images are first mapped into anatomical VOIs using an atlas. Different features including statistical, graph, or connectivity-based features are then computed on these VOIs. Top-ranked VOIs are then input into a support vector machine (SVM) classifier. The developed methods are evaluated on a local database image as well as on Alzheimer's Disease Neuroimaging Initiative (ADNI) public database and then compared to known selection feature methods. These new approaches outperformed classification results in the case of a two-group separation.

Keywords: machine learning, computer-aided diagnosis, first order statistics, feature selection, positron emission tomography, classification, Alzheimer's disease

1. Introduction

Alzheimer's disease (AD) is a degenerative and incurable brain disease which is considered the main cause of dementia in elderly people worldwide. At present, there are around 90 million people who have been diagnosed with AD, and it is estimated that the number of AD patients will reach 300 million by 2050 [1, 2]. The diagnosis of this disease is done by clinical, neuroimaging, and neuropsychological assessments. Neuroimaging evaluation is based on nonspecific features such as cerebral atrophy, which appears very late in the progression of the disease.

Advances in neuroimaging biomarkers help to identify AD in its prodromal stage, mild cognitive impairment (MCI) [3]. Therefore, developing new approaches for early and specific recognition of AD is of crucial importance [4]. Positron emission tomography (PET) is nowadays widely used in neuroscience to characterize brain molecular mechanisms involved in healthy and pathological models [5]. Applied to the brain, PET imaging provides a noninvasive evaluation of various biomarkers, such as the metabolic rate of glucose, the cerebral blood flow, and the neurotransmission, but also the evaluation of some pathological processes such as the neuroinflammation, amyloid deposits, and more recently tubulin-associated unit (TAU) aggregates. In this line, PET using 18-fluoro-deoxy-glucose (18FDG) has been proposed as a biomarker of Alzheimer's disease [6–10]. 18F-FDG-PET provides 3D-volumetric brain imaging of the cerebral metabolic rate of glucose, thought to reflect the synaptic activity and thus the functional brain. However, these abnormalities, evident on average within a group of patients, or in individual cases with advanced disease, can be individually more difficult to confirm in a purely visual interpretation, particularly in the earliest stages of the disease when the current treatment (and those under development) would be yet the most effective. This visual interpretation is in addition subjective and also highly impacted by the experience of the physician. Individual, quantitative, and computer-aided approaches are thus needed for medical management of brain diseases. There has been a growing interest in using the cerebral glucose metabolism rate for AD classification and prediction of conversion from MCI to AD [11–13]. Four main groups of methods have been studied: voxels as feature (VAF)-based [14], discriminative voxel selection-based [15, 16], atlas-based [17–20], and projection-based methods [21]. Recently deep learning and more specifically convolutional neural networks (CNN) and recurrent neural networks (RNN) have been investigated for image classification in the case of Alzheimer's disease computer-aided diagnosis. These approaches show better results than many of the machine learning techniques that have been proposed for recognition tasks [22–27].

In this chapter, a description on the main steps of a computer-aided diagnosis system for AD based on 18F-FDG-PET brain images is given. Section 2 is devoted to data acquisition and preprocessing. Section 3 describes computed features extracted from brain PET images. A feature selection description is reported in Section 4. In Section 5 classification step is explained. Finally a conclusion and future work are given.

2. Materials and methods

2.1 Overview of the computer-aided diagnosis system

A computer-aided diagnosis (CAD) (see **Figure 1**) system consists in different important stages including image acquisition and preprocessing, feature extraction, feature selection, and classification.

In this chapter, the described CAD system focuses on 18F-FDG-PET images from local and public databases and is atlas-based. This means that each PET brain image is mapped into anatomical volume of interests (VOIs), on which features are extracted and then input to a classifier, instead of inputting the whole voxels of each brain image. The goal of such a system is to provide doctors with tools that help them to classify AD and HC subjects.

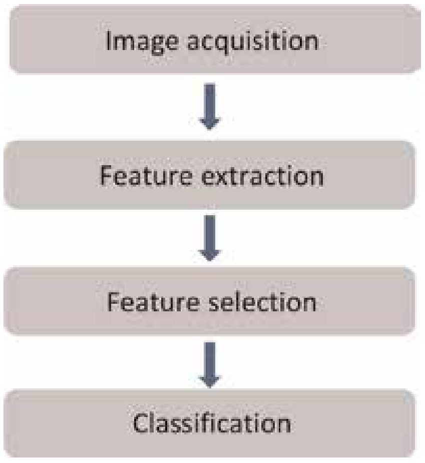


Figure 1.
Flowchart of a CAD system.

2.2 Image dataset and preprocessing

2.2.1 Image dataset

Two datasets are presented in this chapter and used to evaluate the two approaches described in the following. The local database (**Table 1**) consists in 18F-FDG-PET scans that were collected from the "La Timone" University Hospital, in the Nuclear Medicine Department (Marseille, France). The local database image enrolled 171 adults 50–90 years of age, including 81 patients with AD and 61 health control (HC) and 29 mild cognitive impairment (MCI). HC were free from neurological/psychiatric disease and cognitive complaints and had a normal brain MRI. AD subjects exhibited NINCDS-ADRDA [28] clinical criteria for probable AD.

	HC	AD	MCI
Number	61	81	29
Male/female	(24/37)	(32/49)	(12/17)
Ages (Mean [Min.Max])	68.18 [50.86]	70.60 [50.90]	67.55 [50.85]

Table 1.
Demographic and clinical information of subjects of the local database.

Characteristic	HC	AD	MCI
Number of subjects	90	94	88
Female/male	34/56	38/56	32/56
Age (Mean ± SD)	76.08 ± 5.01	75.83 ± 7.37	76.71 ± 6.63
MMSE (Mean SD)	23.46 ± 2.14	28.97 ± 1.15	26.92 ± 1.62

Table 2.
Demographic and clinical information of subjects of ADNI database.

The second used dataset (**Table 2**) was obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (adni.loni.usc.edu). The ADNI was launched in 2003 as a public-private partnership, led by Principal Investigator Michael W. Weiner, MD. The primary goal of ADNI has been to test whether serial MRI, PET, other biological markers, and clinical and neuropsychological assessment can be combined to measure the progression of MCI and early AD.

Therefore, 272 post-processed baseline FDG-PET data were obtained from ADNI, including 94 subjects with AD, 88 subjects with MCI, and 90 NC subjects.

2.2.2 Image preprocessing

Image comparison with brains from different subjects is difficult due to the complexity and anatomical variations of brain structures. For that purpose image data were preprocessed into three steps: spatial normalization, smoothing, and intensity normalization. Spatial normalization was done by registration at voxel level using SPM8 software [11]. The data was spatially normalized onto the Montreal Neurological Institute atlas (MNI). These images were then smoothed using a Gaussian filter with an 8 mm full width at half maximum (FWHM) to increase the signal-to-noise ratio (SNR) [20]. After spatial normalization, intensity normalization was required in order to perform direct image comparison between different subjects. It consisted in dividing the intensity level of each voxel by the intensity level mean of the brain's global gray matter VOI.

2.3 Feature extraction

Each 3D PET brain image was segmented into 116 volumes of interest (VOIs) using an automated anatomical labeling (AAL) atlas. In this research project, the ability of VOIs to best distinguish AD from HC subjects was studied. Different parameter combinations for each VOI were used to select and rank VOIs according to their ability to separate AD group from HC one. The top-ranked VOIs were then introduced into a classifier. Several levels of features were extracted from VOIs. Two approaches have been investigated to achieve this goal.

2.3.1 Separation power factor

In the first approach, only features that extract the statistical information from each VOI are computed. First order statistics and the entropy are extracted from the histogram $h(x)$ of each VOI:

$$h(x) = \frac{number\ of\ voxels\ in\ a\ given\ ROI\ with\ grel\ level\ x}{total\ number\ of\ pixels\ in\ the\ given\ ROI} \tag{1}$$

where x is a gray-level value of a voxel belonging to a VOI and l_{min} and l_{max} are the minimum and the maximum gray-level values in VOI, respectively.

$$P_1 = \sum_{x=l_{min}}^{x=l_{max}} xh(x) \text{ Mean} \tag{2}$$

$$P_2 = \sum_{x=l_{min}}^{x=l_{max}} (x - P_1)^2 h(x) \text{ Variance} \tag{3}$$

$$P_z = \sum_{x=l_{min}}^{x=l_{max}} \frac{(x-P_1)^z h(x)}{(P_1)^{z/2}} \text{ z } \epsilon\{3,4\} \text{ Skewness and Kurtosis} \tag{4}$$

$$P_5 = \sum_{x=l_{min}}^{x=l_{max}} h(x) \log_2(h(x)) \text{ Entropy} \tag{5}$$

For a given VOI, we compute a set of parameter values {Pp|pϵ {1...K}} = {P_1, P_2, P_3, P_4, P_5}. For easier readability, {P_p} is used instead of {P_p|pϵ {1...K}} in the following. HC and AD subjects are plotted in a N feature space, which represents a subset of {P_p}, denoted {P_p}$_N$, $N \leq K$ among subsets. N-Dimensional sphere (N-D sphere) is created over the group of healthy subjects (HC) (N of length one correspond to an interval, N of length two correspond to a disk, N of length greater than or equal to three correspond to a sphere). The N-D sphere's center is the mass center of healthy subjects' distribution. **Figure 2** shows the case of a VOI based on three parameters: the mean, P_1; the standard deviation, P_2; and the kurtosis, P_4. It is a 3D sphere with N = {P_1, P_2, P_4}. At various radii of the N-D sphere, we compute the true positives (TPR) and the false positive rates (FPR).

The ROC curve is created by plotting the true positive rate (TPR) vs. the false positive rate (FPR) for different radii of the N-D sphere settings as it is shown in **Figure 3**. The SPF is defined as the area under the ROC curve (AUC) and is within the range [0, 1].

2.3.1.1 "Combination matrix" analysis

SPF is taken as a key factor to build "combination matrix." For each VOI, we compute this factor all over the combinations of parameter P_p with a length varying from 1 to K (number of feature parameters K = 5). "Combination matrix" is then built and contains 2^K-1 columns.

Each line of this matrix represents a VOI, and each column represents the SPF (noted α_v−N or α_v−{P_p}$_N$) computed on a subset {P_p}$_N$ of N elements of {P_p}.

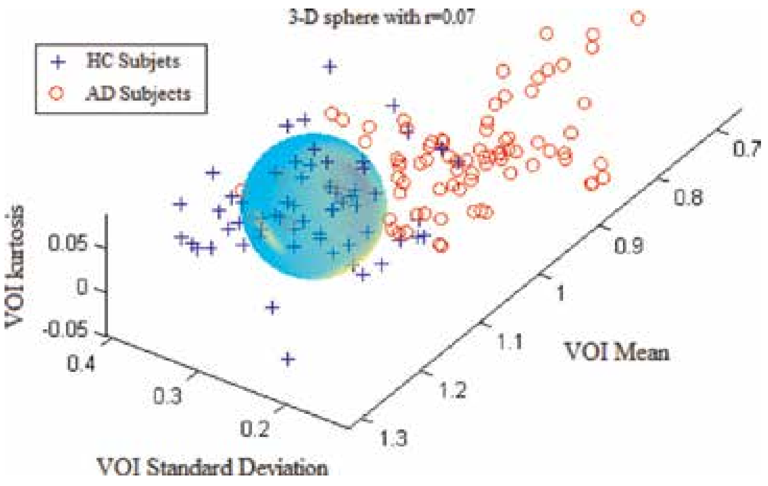


Figure 2.
The separation between AD and HC groups relative to the region "Cingulum_Post_Left" with three parameters: the mean, the standard deviation, and the kurtosis.

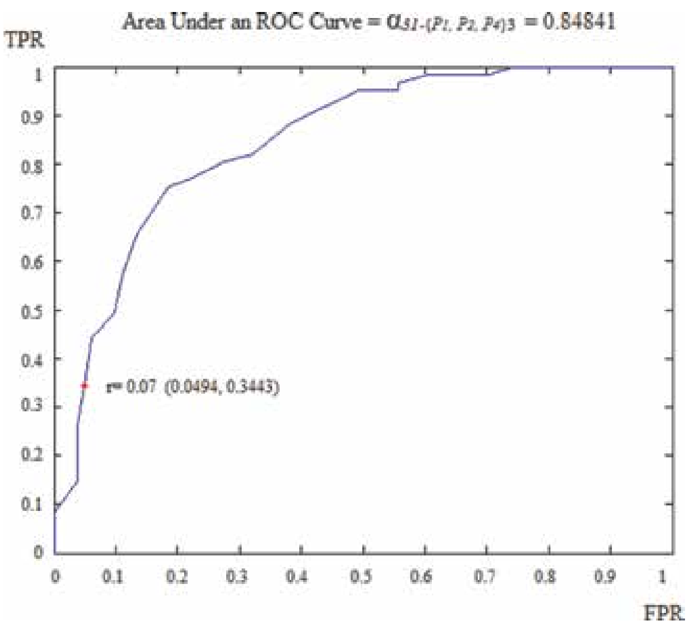


Figure 3.
ROC curve obtained for the region "Cingulum_Post_Left" using three parameters: the mean, the standard deviation, and the kurtosis.

The "combination matrix" has then L lines and 2^K-1 columns. L = 116 is the number of VOIs.

2.3.1.2 "Combination matrix" 1: mono-parametric analysis

Mono-parametric analysis consists in ranking VOIs according to their higher values of SPF in the "combination matrix." The set of the top-ranked VOIs are selected for the classification step.

2.3.1.3 "Combination matrix" 2: multi-parametric analysis

Multi-parametric analysis for "combination matrix" depends on both the combination of the parameters (subset of $\{P_p\}$) for each VOI and the combination of the VOIs, subsets of $\{V_v,|v\epsilon\ \{1\ldots\ L\}\}$.

The procedure begins with the choice of the first two combination VOIs depending on all the possible combinations of VOIs of length 2 ($\{V_j, V_q\}$, $1 \leq j$, $q \leq L$, $j \neq q$). Thereafter, it consists on an iterative process according to which we add one VOI at each step to a VOIs list, sequential forward selection (SFS) [29].

At each step of the iterative procedure, HC and AD subjects are plotted in a new feature space by combination of parameters for that selection of VOIs, and examine the SPF value based on this combination, which is named cumulated CSPF. The VOIs that provide the best cumulated SPF value are then added to the final VOI combination. Our algorithm stops when the same or a lower cumulated SPF is obtained.

2.3.2 Multilevel representation

In the second approach, we investigate the multilevel feature representation, which considers both region properties (first level), connectivity between any pair of VOIs (second level), and an overall connectivity between one region and the other regions (third level). The proposed method ranks regions from highly to slightly affected by the disease. A classifier selection strategy is proposed to choose a pair of classifiers with high diversity.

2.3.2.1 First-level feature

The first-level feature consists in computing the first order statistics of voxels within each VOI. The nth subject can be represented as:

$$\boldsymbol{r}_n^m = \left[r_{n1}^m, r_{n2,\dots\dots\dots}^m r_{np}^m\right]$$

$$\mathbf{r}_n^s = \left[r_{n1}^s, r_{n2,\dots\dots\dots}^s r_{np}^s\right]$$

where r_n^m and r_n^s are the mean intensity and standard deviation of each VOI, respectively, and p is the number of VOIs; here p = 116.

2.3.2.2 Second-level feature

The second-level feature consists in computing a similarity-based connectivity parameter w_{ij} between VOIs:

$$w_{ij} = \begin{cases} e^{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2} & i \neq j \\ 0 & i = j \end{cases} \tag{6}$$

where x_i is the feature vector containing the mean and standard deviation of the ith VOI and w_{ij} is the similarity coefficient between the *i*th and the *j*th VOIs. The second-level features of any subject is denoted W_r which is a symmetric matrix.

The second-level feature is composed of similarity coefficients between all the 116 VOIs, totally 6670 dimension (upper part of matrix $\mathbf{W}_r$), which is clearly not an optimal dimension for the subsequent classification. Therefore, $\mathbf{W}_r$ is further decomposed into three subsets of features. Similar to the way of computing similarity coefficients between VOIs, we can obtain similarity coefficients between subjects for a specific VOI:

$$w_{uv} = \begin{cases} e^{-\|\boldsymbol{x}_u - \boldsymbol{x}_v\|^2} & u \neq v \\ 0 & u = v \end{cases} \tag{7}$$

where u and v stand for the uth and vth subjects. For any VOI, a symmetric matrix for subjects, $\mathbf{W}_s$ is computed.

The dimension of $\mathbf{W}_s$ is determined by the number of subjects, N, in a group (AD, NC, MCI). Since each subject is segmented into 116 VOIs, thus there are 116 matrices like $\mathbf{W}_s$.

On the one hand, a VOI that is not affected by AD will give similar coefficients between AD and NC subjects. On the other hand, a VOI affected by AD will give different similarity coefficients for the two groups. In order to quantify the difference, we compute the frequency distribution histogram of the upper triangle values of $\mathbf{W}_s$. **Figure 4** shows the cumulative probability curve of similarity coefficients obtained for region angular L (c), region hippocampus L (b), and region cerebellum 10 R (a), respectively. There is a clear difference between the AD and NC groups in

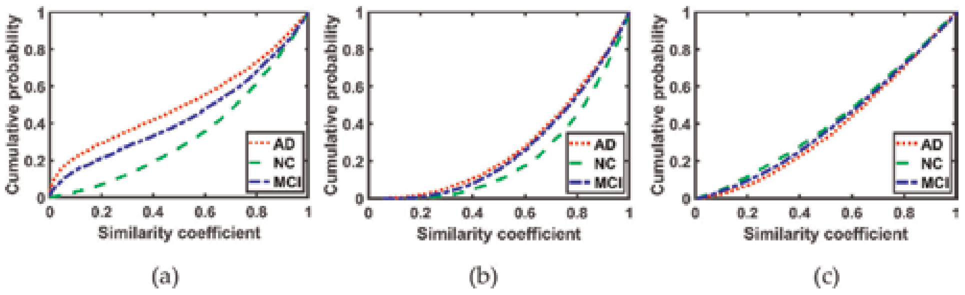


Figure 4.
Statistics of the similarity coefficients between subjects for certain VOIs. (a) VOI: angular L; (b) VOI: hippocampus L; (c) VOI: cerebellum 10 R.

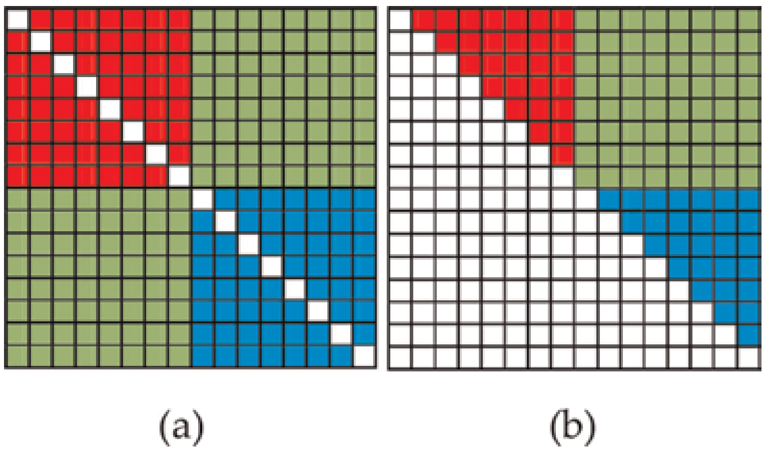


Figure 5.
Instance of the division for a similarity matrix. Highly involved VOIs in AD (red); Less involved VOIs in AD (blue); Connectivities between highly and slightly influenced VOIs appear in green.

Figure 4a, and for the other two VOIs, the difference decreases gradually. Cerebellum 10 R appears as the VOI that is almost unaffected by AD, while VOI angular L is the one that is most affected by AD. The area under curve, denoted S, quantifies differences between VOIs.

After ranking all the VOIs, the similarity matrix $\mathbf{W}_r$ is recalculated according to the new order of VOIs. $\mathbf{W}_r$ is divided into four equal parts, as shown in **Figure 5a**. VOIs that are highly involved in AD appear in red and are denoted $\mathbf{W}^h$. VOIs that are less involved in AD appear in blue and are denoted $\mathbf{W}^l$. Connectivities between highly and slightly influenced VOIs, denoted $\mathbf{W}^m$, appear in green.

Since W_r is symmetric, only upper triangular matrix is taken into consideration, like in **Figure 5b**. Therefore, the second-level feature $\mathbf{W}_r$ is divided into three sets, and after converting them to vectors, the second-level feature for the *n*th sample is represented as w_n^h, w_n^m, and w_n^l, respectively. For w_n^h and w_n^l (red and blue parts in **Figure 5b**), the dimension is 1653 (58× (58–1)/2), and for w_n^m (green part), it is 3364 (58 × 58). Apparently, compared to 6670 (red, blue, and green parts), the dimension is decreased by about 50–75%.

2.3.2.3 Third-level feature

The third-level feature is extracted from a graph, which represents the overall connectivity between a VOI and the others. A graph G = (V, E) is defined by a finite set V of vertices (VOIs) and a finite set $E \subseteq V \times V$ of edges (similarity coefficient between the *i*th and the *j*th VOI is denoted α_{ij}).

After constructing a graph for a subject, several graph measures can be computed [30]. The third-level feature is represented by two graph measures: strength and clustering coefficient.

Strength: the sum of a vertex's neighboring link weights [30]:

$$s_i = \sum_{j=1}^{p} \alpha_{ij} \tag{8}$$

where s_i is the strength of a vertex or a VOI.

Clustering coefficient: the geometric mean of all triangles associated with each vertex [30]:

$$c = \frac{diag\left[\left(W_r^{\frac{1}{3}}\right)^3\right]}{\boldsymbol{d}(\boldsymbol{d}-1)} \tag{9}$$

where $diag()$ is an operator which takes the diagonal values from a matrix, c is a clustering coefficient vector, and d is a degree vector in which the element d_i is

$$d_i = \sum_{j=1}^{p} a_{ij} \tag{10}$$

where a_{ij} is the connection status between the *i*th vertex and the *j*th vertex: $a_{ij} = 0$ when $w_{ij} = 0$, otherwise $a_{ij} = 1$.

These features exhibit different ranges of values. Thus a procedure of feature normalization is necessary by z-score prior to classification:

$$z_{mn} = \frac{f_{mn} - \mu_m}{\delta_m} \tag{11}$$

where f_{mn} is the value of the mth feature of the nth sample and μ_m and δ_m are the mean value and standard deviation of the mth feature, respectively. Most of the f_{mn} values are within the range $[-1, 1]$, while out-of-range values are clamped to either -1 or 1.

2.4 Classification results

2.4.1 Separation power factor approach

Classification was performed using a support vector machine (SVM) classifier. These classifiers map pattern vectors to a high-dimensional feature space where a "best" separating hyperplane (the maximal margin hyperplane) is build. In the present work, we used a linear kernel SVM classifier [29]. The reduced number of subjects (142 patients) for this first approach leads to use a leave-one-out (LOO) strategy as the most suitable for classification validation. This technique iteratively holds out a subject for test while training the classifier with the remaining subjects, so that each subject is left out once. Parameter C is used during the training phase and tells how much outliers are taken into account in calculating support vectors. A good way to estimate the better C value to be used is to perform it with cross-validation. As result, the estimation value is fixed to 10 (C = 10) depend upon database.

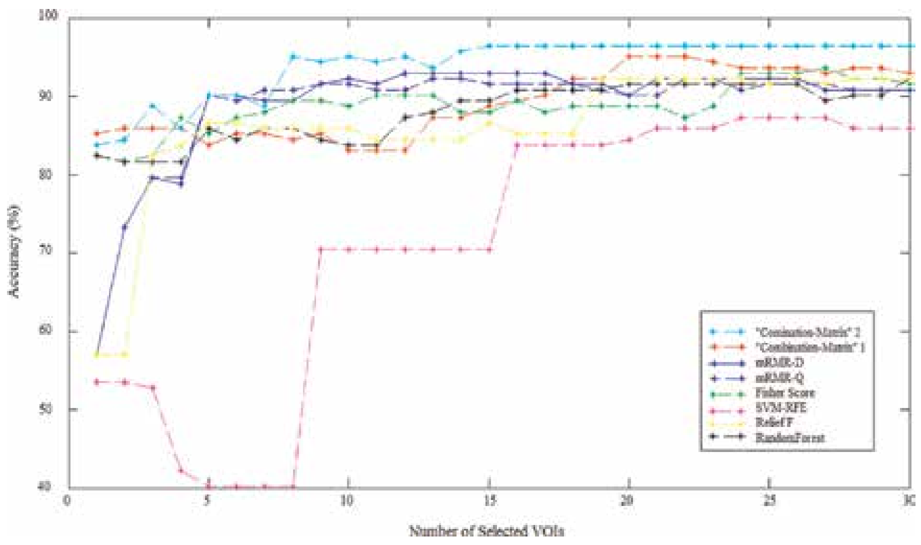


Figure 6.
Average accuracy obtained with SVM classifier varying number of features for different VOI selection analyses and applying LOO cross-validation with estimation value C = 10.

The results achieved using the proposed method with SVM are shown in **Figure 6**. These results are compared to different feature selection methods: Fisher score [31], support vector machine-recursive feature elimination (SVM-RFE) [32], feature selection with random forest [33], ReliefF [34], and minimum redundancy maximum relevance (mRMR) [35]. Cross-validation average accuracy results are shown as a function of the number of selected VOIs. We note that by injecting 19, 20, or 21 VOIs with their combination of parameters in the SVM with "combination matrix" 1, mono-parametric analysis, we obtain a classification rate of 95.07%. The "combination matrix" 2 achieved higher accuracy with lower number of features (14 VOIs). The best classification results were obtained on the "combination matrix" 2, achieving 96.47%. Therefore, the proposed feature selection from PET images is very effective providing a good discrimination between AD subjects and HC, where we considered the VOIS in the brain image illustrated in **Figure 7**.

2.4.2 Multilevel approach

The support vector machine (SVM) classifier was also applied in this second approach for classifying AD or MCI from NC subjects. This second approach takes into account three levels of features, the total of which is seven types of features that are input to seven linear SVMs. The margin parameter C of all the SVMs is fixed to one for a fair comparison. The final decision is made through a majority voting of the seven classifiers' outputs:

$$Y = sgn\left(\sum_{t=1}^{t=7} y_t\right) \tag{12}$$

where sgn() is a sign function and y_t denotes the labels of SVMt.

Classification concerns AD vs. NC and MCI vs. NC. Evaluation was done using four different parameters: classification accuracy (ACC), sensitivity (SEN), specificity (SPE), and area under the curve (AUC). A tenfold cross-validation technique

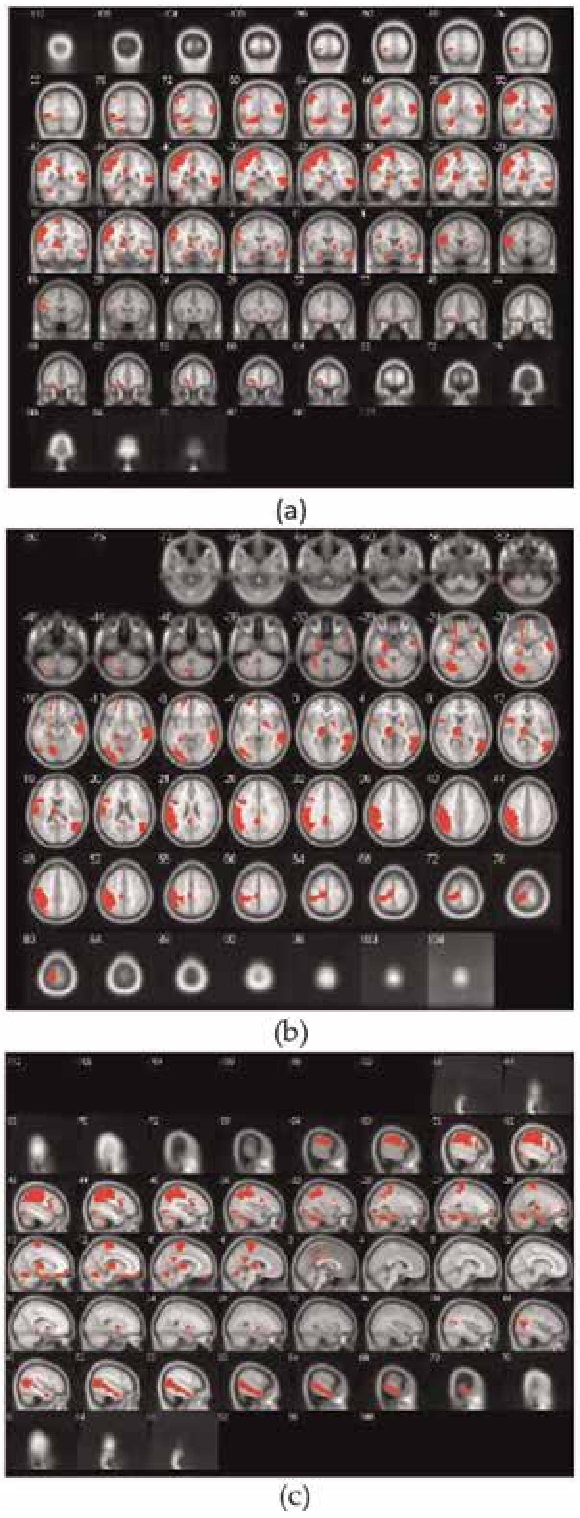


Figure 7.
15 VOIs' representation of the "combination matrix" 2 on a coronal plane (a), on a transverse plane (b), and on a sagittal plane (c).

was used to assess the performance and repeated 10 times to reduce the possible bias. A least absolute shrinkage and selection operator (LASSO) was used for feature selection. LASSO parameter λ retained for the experiments was obtained by nested cross-validation on the training dataset. **Tables 3** and **4** show the obtained results for AD vs. NC and MCI vs. NC when using each feature, respectively. As can be seen, the first-level and second-level features outperformed the third-level feature for AD and MCI diagnosis. This could be explained since the two first-level features are more linked to VOI's property or connectivity between each pair of VOIs, while the third-level feature represents an overall connectivity between a VOI and the others.

This second proposed method was compared with the state-of-the-art methods, including Hinrichs's method [14], Gray's method [12], Li's method [36], and Padilla's method [21], which were applied to FDG-PET data. The results are shown

Feature	ACC	SEN	SPE	AUC
Mean intensity	85.13	86.61	83.97	93.39
Standard deviation	85.49	84.98	86.24	93.84
Connectivity wh	85.05	86.24	84.56	93.01
Connectivity wm	86.88	88.82	85.17	93.88
Connectivity wl	83.98	84.31	83.37	91.37
Strength	80.77	80.29	81.50	88.63
Clustering coefficient	83.89	84.03	84.26	92.05

Table 3.
Performance of different types of feature for AD vs. NC (%).

Feature	ACC	SEN	SPE	AUC
Mean intensity	73.55	75.01	72.87	81.36
Standard deviation	78.19	78.31	78.69	86.67
Connectivity wh	72.78	70.63	74.35	83.19
Connectivity wm	74.67	76.06	73.65	83.27
Connectivity wl	74.89	77.01	72.68	78.94
Strength	71.72	70.62	72.01	80.07
Clustering coefficient	72.31	74.73	70.26	80.36

Table 4.
Performance of different types of feature for MCI vs. NC (%).

Method	ACC	SEN	SPE	AUC
Hinrichs et al. [14]	84	84	82	87.16
Gray et al. [12]	88.4	83.2	93.6	—
Li et al. [36]	89.1	92	86	97
Padilla et al. [21]	86.59	87.50	85.36	—
Our method	90.48	90.58	89.38	95.95

Table 5.
Performance comparison for AD vs. NC (%).

Method	ACC	SEN	SPE	AUC
Gray et al. [12]	81.3	79.8	82.9	—
Li et al. [36]	63.2	65	62	72
Our method	81.09	80.99	81.25	87.65

Table 6.
Performance comparison for MCI vs. NC (%).

in **Tables 5** and **6**. The proposed approach outperformed these methods in terms of ACC and SEN for AD diagnosis. For MCI diagnosis, our method outperforms the other methods in SEN and AUC, and the difference with the best result is 0.21 and 1.65% for ACC and SPE, respectively. Moreover, compared with **Tables 3** and **4**, the significant improvements indicate the effectiveness of the ensemble classification, thereby explaining the multilevel features are necessary.

3. Conclusion

In this chapter, two novel methods for VOI ranking are developed to classify brain PET images better. The first approach consists in ranking VOIs using ROC curves and quantifies the ability of a VOI to classify HC from AD subjects thanks to the area under curve for AUC.

The second approach which uses multilevel features is proposed to address the PET brain classification problem. Three levels of features are extracted from PET brain images and ranked in order to feed a SVM. Different models are trained by using different types of features. The final decision is made through the majority voting of different models' outputs. According to experiments on ADNI dataset, the proposed method can improve the performance of AD and MCI diagnosis when compared with those state-of-the-art methods which are also developed under FDG-PET.

To go further in computer-aided diagnosis tasks, other features like texture and gradient computed on VOIs have to be joined to first order statistical parameters in order to enrich information. Modern machine learning based on deep learning on neural network will be included in our future work.

Acknowledgements

This work has been conducted in the framework of DHU-Imaging thanks to the support of the A*MIDEX project (n°ANR-11-IDEX-0001-02) (« Investissements d'Avenir » French Government programme, managed by the French National Research Agency (ANR)).

Author details

Mouloud Adel[1,3]*, Imene Garali[1,3], Xiaoxi Pan[2,3], Caroline Fossati[2,3], Thierry Gaidon[2,3], Julien Wojak[1,3], Salah Bourennane[2,3] and Eric Guedj[1,3]

1 Aix-Marseille Université, CNRS, Marseille, France

2 Centrale Marseille, Marseille, France

3 Institut Fresnel UMR-CNRS, Marseille, France

*Address all correspondence to: mouloud.adel@univ-amu.fr

References

[1] Zhan L, Zhou J, Wang Y, Jin Y, Jahanshad N, Prasad G, et al. Comparison of nine tractography algorithms for detecting abnormal structural brain networks in Alzheimer's disease. Frontiers in Aging Neuroscience. 2015;**7**(48). DOI: 10.3389/fnagi.2015.00048

[2] Zhu X, Suk HI, Zhu Y, Thung KH, Wu G, Shen D. Multiview classification for identification of Alzheimer's disease. Machine Learning in Medical Imaging. 2015;**9352**:255-262. DOI: 10.1007/978-3-319-24888-2_31

[3] Ricea L, Bisdasb S. The diagnostic value of FDG and amyloid PET in Alzheimer's disease—A systematic review. European Journal of Radiology. 2017;**94**:16-24

[4] Garali I, Adel M, Bourennane S, Guedj E. Histogram-based features selection and volume of interest ranking for brain PET image classification. IEEE Journal of Translational Engineering in Health and Medicine. 2018;**6**:1-12; 2100212; DOI:10.1109/JTEHM.2018.2796600

[5] Varrone A, Asenbaum S, Vander Borght T, Booij J, Nobili F, Någren K, et al. EANM procedure guidelines for PET brain imaging using [^{18}F] FDG, version 2. European Journal of Nuclear Medicine and Molecular Imaging. 2009; **36**(12):2103-2110. DOI: 10.1007/s00259-009-1264-0

[6] Pan X, Adel M, Fossati C, Gaidon T, Guedj E. Alzheimer disease diagnosis with FDG-PET brain images by using multilevel features. IEEE International Conference on Image Processing. Athens, Greece; October 7-10, 2018

[7] Chen K, Langbaum JB, Fleisher AS, Ayutyanont N, Reschke C, Lee W, et al. Twelve-month metabolic declines in probable Alzheimer's disease and amnestic mild cognitive impairment assessed using an empirically pre-defined statistical region-of-interest: Findings from the Alzheimer's disease neuroimaging initiative. NeuroImage. 2010;**51**(2):654-664

[8] Albert MS, DeKosky ST, Dickson D, Dubois B, Feldman HH, Fox NC, et al. The diagnosis of mild cognitive impairment due to Alzheimer's disease: Recommendations from the national institute on aging-Alzheimer's association workgroups on diagnostic guidelines for Alzheimer's disease. Alzheimer's & Dementia. 2011;**7**(3): 270-279

[9] Davatzikos C, Bhatt P, Shaw LM, Batmanghelich KN, Trojanowski JQ. Prediction of MCI to AD conversion, via MRI, CSF biomarkers, and pattern classification. Neurobiology of Aging. 2011;**32**(12):19-27

[10] Cheng B, Liu M, Zhang D, Munsell BC, Shen D. Domain transfer learning for MCI conversion prediction. IEEE Transactions on Biomedical Engineering. 2015;**62**(7):1805-1817

[11] Friston K, Ashburner J, Kiebel S, Nichols T, Penny W. Statistical Parametric Mapping: The Analysis of Functional Brain Images. Amsterdam: Academic Press; 2007

[12] Gray KR, Wolz R, Heckemann RA, et al. Multi-region analysis of longitudinal FDG-PET for the classification of Alzheimer's disease. NeuroImage. 2012;**60**(1):221-229

[13] Toussaint PJ, Perlbarg V, Bellec P, Desarnaud S, Lacomblez L, Doyon J, et al. Resting state FDG-PET functional connectivity as an early biomarker of Alzheimer's disease using conjoint univariate and independent component analyses. NeuroImage. 2012;**63**:936-946

[14] Hinrichs C, Singh V, Mukherjee L, Xu G, Chung MK, Johnson SC. Spatially augmented LP boosting for AD classification with evaluations on the ADNI dataset. NeuroImage. 2009;**48**: 138-149

[15] Cabral C, Morgado PM, Campos Costa D, Silveira M. Predicting conversion from MCI to AD with FDG-PET brain images at different prodromal stages. Computers in Biology and Medicine. 2015;**58**:101-109

[16] Eskildsen SF, Coupe P, Garcia-Lorenzo D, Fonov V, Pruessner JC, Collins DL. Prediction of Alzheimer's disease in subjects with mild cognitive impairment from the ADNI cohort using patterns of cortical thinning. NeuroImage. 2013; **65**:511-521

[17] Pagani M, De Carli F, Morbelli S, Oberg J, Chincarini A, Frisoni GB, et al. Volume of interest-based [^{18}F] fluorodeoxyglucose PET discriminates MCI converting to Alzheimer's disease from healthy controls. A European Alzheimer's disease consortium (EADC) study. NeuroImage: Clinical. 2015;**7**: 34-42

[18] Pan X, Adel M, Fossati C, Gaidon T, Guedj E. Multi-level feature representation of FDG-PET brain images for diagnosing Alzheimer's disease. IEEE Journal of Biomedical and Health Informatics. July, 2018:1-9. DOI 10.1109/JBHI.2018.2857217

[19] Garali I, Adel M, Bourennane S, Ceccaldi M, Guedj E. Brain region of interest selection for ^{18}FDG positrons emission tomography computer-aided image classification. Innovation and Research in Biomedical Engineering Journal (IRBM). 2016;**37**:23-30

[20] Garali I, Adel M, Bourennane S, Guedj E. Brain region ranking for ^{18}FDG-PET computer-aided diagnosis of Alzheimer's disease. Biomedical Signal Processing and Control Journal. 2016;**27**:15-23

[21] Padilla P, López M, Górriz JM, Ramírez J, Salas-Gonzalez D, Alvarez I, et al. NMF-SVM based CAD toll applied to functional brain images for the diagnosis of Alzheimer disease. IEEE Transactions on Medical Imaging. 2012; **2**:207-216

[22] Liu S, Liu S, Cai W, Pujol S, Kikinis R, Fen D. Early diagnosis of Alzheimer's disease with deep learning. In: 2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI); IEEE; 2014. pp. 1015–1018

[23] Suk H-I, Lee S-W, Shen D, Initiative ADN, et al. Hierarchical feature representation and multimodal fusion with deep learning for AD/MCI diagnosis. NeuroImage. 2014;**101**: 569-582

[24] Payan A, Montana G. Predicting Alzheimer's disease: A neuroimaging study with 3D convolutional neural networks. 2015. arXiv: 1502.02506

[25] Ortiz A, Munilla J, Gorriz JM, Ramirez J. Ensembles of deep learning architectures for the early diagnosis of the Alzheimer's disease. International Journal of Neural Systems. 2016;**26**(07): 1650025

[26] Hosseini-Asl E, Keynton R, El-Baz A. Alzheimer's disease diagnostics by adaptation of 3d convolutional network. In: Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP); Lausanne: IEEE; 2016. pp. 126–130. DOI: 10.1109/ICIP.2016.7532332

[27] Liu M, Cheng D, Yan W. Classification of Alzheimer's disease by combination of convolutional and recurrent neural networks using FDG-PET images. June 19, 2018:12-35. DOI: 10.3389/fninf.2018.00035

[28] McKhann G, Drachman D, Folstein M, Katzman R, Price D, Stad-lan EM. Clinical diagnosis of Alzheimer's disease: Report of the NINCDS-ADRDA work group under the auspices of Department of Health and Human Services Task Force on Alzheimer's disease. Neurology. 1984;**34**(7):939-944

[29] Webb AR, Copsey KD. Statistical Pattern Recognition. 3rd ed. West Sussex, United Kingdom: John Wiley & Sons Ltd; 2011

[30] Rubinov M, Sporns O. Complex network measures of brain connectivity uses and interpretations. NeuroImage. 2010;**52**:1059-1069

[31] Duda RO, Hart PE, Stork DG. Pattern Classification. 2nd ed. New York, NY: John Wiley & Sons Ltd

[32] Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support vector machines. Machine Learning. 2002; **46**:389-422

[33] Liaw A, Wiener M. Classification and regression by randomForest. R News. 2002;**2**(3):18-22

[34] Sikonja MR, Kononenko I. Theoretical and empirical analysis of relief and ReliefF and RReliefF. Machine Learning Journal. 2003;**53**(1–2):23-69

[35] Peng H, Long F, Ding C. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2005;**27**(8): 1226-1238

[36] Li R, Perneczky R, Yakushev I, et al. Gaussian mixture models and model selection for [^{18}F] fluorodeoxyglucose positron emission tomography classification in Alzheimer's disease. PLoS One. 2015;**10**

Chapter 5

An Efficient Block-Based Algorithm for Hair Removal in Dermoscopic Images

Ihab Zaqout

Abstract

Hair occlusion in dermoscopy images affects the diagnostic operation of the skin lesion. Segmentation and classification of skin lesions are two major steps of the diagnostic operation required by Dermatologists. We propose a new algorithm for hair removal in dermoscopy images that includes two main stages: hair detection and inpainting. In hair detection, a morphological bottom-hat operation is implemented on Y-channel image of YIQ color space followed by a binarization operation. In inpainting, the repaired Y-channel is partitioned into 256 nonoverlapped blocks and for each block, white pixels are replaced by locating the highest peak of using a histogram function and a morphological close operation. Our proposed algorithm reports a true positive rate (sensitivity) of 97.36%, a false positive rate (fall-out) of 4.25%, and a true negative rate (specificity) of 95.75%. The diagnostic accuracy achieved is recorded at a high level of 95.78%.

Keywords: dermoscopy image, melanoma, hair detection, hair removal, inpainting, skin lesion

1. Introduction

Melanoma, otherwise called malignant melanoma, is a kind of cancer that is created from the pigment-containing cells known as melanocytes. Melanomas commonly occur in the skin, but may rarely occur in the mouth, digestion tracts, or eye. Malignant melanoma is the most forceful and hazardous skin disease. It is created in the cells that give the skin its color (melanocytes) and has a high inclination to spread to different parts of the body. The cure rates depend incredibly on the phase of melanoma, when it is discovered. On the off-chance that melanoma is perceived and treated early, it is quite often repairable; however, in the event that it is not, the disease can progress and spread to different parts of the body, where it turns out to be difficult to treat and can be lethal. While it is not the most well-known of the skin cancers, it causes the most deaths.

The timeline of melanoma is summarized in **Table 1**, portraying particularly significant disclosures and advances in treatment against the disease. Malignant melanoma is the most genuine type of skin cancer. An early detection and diagnosis of skin cancer prevent its advancement to later stages. Menzies method, the seven-point checklist, the CASH (Color, Architecture, Symmetry, and Homogeneity) algorithm, and the broadly used algorithm is the ABCD/ABCDE (Asymmetric,

Year/period	Key developments
Prior to 1750	Hippocrates is a Greek physician and is a standout among the most exceptional figures ever of. He is thought to be the first to record a depiction of melanoma, which he portrays as melas, which means dark, and oma, which means tumor, in Greek [1]. Various references to "lethal black tumors with metastases and black liquid in the body" are discussed precisely by [2].
1750s–1830s	John Hunter is recorded as the first to work on a patient and Laennec is the first to recognize melanoma as a sickness isolated from others. Carswell presents the term melanoma [1].
1840s–1900s	Information advances in treatment. Careful anesthesia is received [2], and rules for careful treatment against melanoma are combined. Propelled melanoma is perceived as untreatable [1].
Twentieth century	The etiology and hereditary contribution in melanoma are found. Qualities like skin, hair and eye shading are found to have an effect on melanoma improvement [1]. Driver hereditary transformations in melanoma are found [2].
1970s – 1990s	A developing number of concentrates in this period recommend that sun presentation assumes a critical part in the improvement of a few melanomas. In the 1980s, the general wellbeing network and promotion bunches start forewarning the general population about the potential dangers of sun introduction. Dermoscopy ends up accessible in the 1990s [3].
Present time	Today, melanoma is dealt with by medical procedure, immunotherapy, directed treatment, chemotherapy and radiation treatment [4]. Melanoma is more typical in regions that are generally Caucasian [5].

Table 1.
Timeline of melanoma.

Border, Color, and Dermatoscopic structures) method are computational algorithms have been developed using image processing techniques to help dermatologists in early diagnosis of skin lesions [6–9]. Several methods for evaluating melanocytic lesions by dermoscopy are described precisely in Section 3.

Extraordinary endeavors have been done by researchers to make compelling and dependable computerized demonstrative techniques for skin lesions, yet very little research has been centered on the hair removal problem. Clearly, human body might be totally covered by hair, which has a scope of various surfaces, orientations, and colors; in this way influence incompletely/totally the presence of skin lesions as appeared in **Figure 1**.

Hair removal is an important step in dermoscopy images to classify the skin lesion correctly into benign, suspicious, or malignant. Various techniques were applied to remove hairs automatically from dermoscopic images are discussed in detail by [11, 12]. The rest of this research is organized as follows: Section 2 describes an overview of related work. Section 3 describes an overview of common

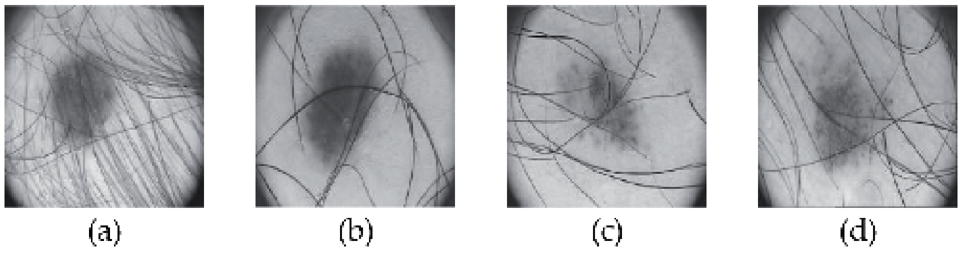


Figure 1.
Sample of digital dermoscopic images with hair pixels are collected from PH^2 dataset [10].

methods used for diagnosis of skin lesions while Section 4 describes the proposed technique. The implementation is presented and discussed in Section 5 followed by some remarks and future work.

2. Related work

The pixel-based interpolation technique was proposed by Satheesha et al. [13] to locate a quadratic curve, which detects bended hairs in the binary image mask for removal and replacement. Gabor filtering and PDE-based image reconstruction was proposed by Nasonova et al. [14] for hair removal problem. Moreover, for edge sharpening, they have utilized a warping algorithm to move pixels from the neighborhood of the blurred edge closer to the edge while the overall luminosity and texture patterns of the skin lesions are preserved. In [15], two fundamental steps are proposed to automatically detect and remove hairs from dermoscopy images: generation of a binary image mask by isolating hairs and ruler marking. From the RGB dermoscopy image, a red channel is utilized to perform noise removal followed by a generation of the binary image mask via an adaptive canny edge detector. Furthermore, a repaired task in view of polygons inpainting is implemented on the white regions of the generated mask.

The work of [16] relied on two classes of images: gray scale and RGB images. In gray scale images, in light of edge property, a circular mask is used to remove the nonskin pixels followed by a repair operation accomplished by a normalization process of pixel values. In RGB images, in light of histogram values, a frequency of occurrence of each bin is measured followed by the calculation of minimum distance among neighborhood pixels. An algorithm presented by Abbas et al. [17] for automatically detecting and repairing hair occlusion in dermoscopy images. In the detection stage, hairs are segmented utilizing MF-FDOG, thresholding, and morphological edge-based methods connected for improvement. In the repair stage, to inpaint the image without loss of texture patterns of skin lesions, the fast marching technique is implemented. MRF-based Multi-Label Optimization and Dual-Channel Quaternion Tubularness Filters are proposed by Mirzaalian et al. [18] for hair improvements in dermoscopy images. Their method was approved and contrasted with different methods regarding: hair segmentation accuracy, image inpainting quality, and image classification accuracy. To remove hairs by detecting hair pixels in a binary image mask followed by replacement through pixel interpolation is implemented via the Generalized Radon Transform (GRT). The Radon Transform was chosen to locate quadratic curves characterized by rational angle and scaling [19].

An effective detection of artifacts proposed by Okuboyejo et al. [20] consists of a two-stage artifact detection termed: fast image restoration (FIR) by means of canny algorithm and line segment detection (LSD) operation. To remove artifacts from dermoscopic images, the fast marching method (FMM) was applied at each stage while preserving morphological features during artifacts removal. A threshold set model for digital hair removal from dermoscopic images proposed by Okuboyejo et al. and Koehoorn et al. [21, 22]. They proposed a gap-detection algorithm to find hairs for every threshold and merge results in a single mask image. To locate hairs in the generated mask, morphological filters and medial descriptors are combined. The proposed work of [23] automatically detects and removes hairs and ruler markings from dermoscopy images. In detection stage, they used a curvilinear structure and modeling, and additional feature guided exemplar-based inpainting stage. Extensions to the fast marching method are introduced by Hearn [24] with the aim to enhance the segmentation of medical image data. The proposed algorithm used

Method	Hair detector	Inpainting method	#Test images
DullRazor [31]	Generalized morphological closing	Bilinear interpolation	5
E-shaver [32]	Prewitt edge detector	Color averaging	5
Fiorese et al. [33]	Top-hat operator	PDE-based [34]	20
Huang et al. [35]	Multiscale matched filters	Median filtering	20
Xie et al. [36]	Top-hat operator	Anisotropic diffusion [37]	40
Abbas et al. [11]	Derivative of Gaussian	Coherence transport [38]	100
Koehoorn et al. [21, 22]	Multiscale skeletons and morphological operators	Fast marching [39]	≅ 300
Our method	Top-hat operator	Block-based histogram function & morphological close	200

Table 2.
Comparison of existing digital hair removal methods.

to limit the occurrence of bleeding across boundaries, including automatic starting point selection and statistical region combination.

Two removal hair approaches are conducted by Sultana et al. [25]. The first method is based on a simple morphological closing operation with a disk-shaped structural element while the top-hat transform combined with a bicubic interpolation utilized in the second approach. An effective hair removal algorithm for dermoscopic imagery is implemented by Bibiloni et al. [26]. They utilized soft color morphology operators that able to cope with color images. The hair removal filter used is basically made out of a morphological curvilinear object detector and a morphological-based inpainting algorithm. A simple approach to automatic hair and consequently noise removal were discussed by Acebuque-Salido and Ruiz [27]. The process starts with a median filter on each color space of RGB, a bottom-hat filter, a binary conversion, a dilation and morphological opening, and then the removal of small connected pixels. The detected hair regions are then filled up using harmonic inpainting. Their experiments were carried out on the PH^2 dataset and compared to DullRazor method. Furthermore, they generated synthetic hair on skin images and measured the reconstruction quality using peak signal-to-noise ratio. In the work done by Al-abayechi et al. [28], a hair was removed, and reflective light was reduced using morphological operations and a median filter.

An algorithm for the automated hairs detection was implemented by Chakraborti et al. [29] to 50 dermoscopic melanoma images. They used an adaptive, canny edge-detection method, followed by morphological filtering and an arithmetic addition operation. Their proposed method produced 6.57% segmentation error (SE), 96.28% true detection rate (TDR) and 3.47% false positive rate (FPR). The proposed algorithm by Toossi et al. [30] divided into two phases: detection and removal. In detection, light and dark hairs and ruler marking are segmented via an adaptive canny edge detector and refinement by morphological operators. In removal, the hairs are repaired in view of multi-resolution coherence transport inpainting.

In addition to the above-mentioned hair removal methods, several aspects are captured in **Table 2**.

3. An overview of common methods

The purpose of this section is to review state-of-the-art melanomas diagnosis methods and technologies that have the potential to reduce melanoma mortality.

Current methods for the recognition of melanoma go from populace-based instructive crusades and screening to the utilization of calculation driven imaging innovations and execution of measures that distinguish markers of change. Each one of the following methods is used for the dermoscopic separation between benign melanocytic lesions and melanoma based on its own features.

3.1 Menzies method

The Menzies technique is an improved dermoscopy strategy for diagnosing melanomas [40, 41]. In the first arrangement, it had an affectability of 92% and a specificity of 71% for the analysis of melanoma. This strategy utilizes purported "negative" and "positive" highlights. For a melanoma to be analyzed, none of the two "negative highlights" ought to be found and no less than 1 of the 9 "positive highlights" must be available. An injury is suspicious of melanoma on the off chance that it has in excess of one shading and is hilter kilter in design. Suspect lesions showing any of the nine positive highlights for melanoma are thought to be melanoma except if demonstrated something else.

- Negative features (benign lesions): symmetrical pattern (colors and structure) and single color.

- Positive features (melanoma): Blue-white veil, multiple brown dots, pseudopods, radial streaming, scar-like depigmentation, multiple (5–6) colors, multiple blue/gray dots, and broadened network.

3.2 Seven-point checklist method

In the last years, a great deal of investigative techniques in view of scored calculations have been acquainted both with streamline the dermoscopic learning and to enhance the early melanoma discovery. The seven-point checklist, distributed in 1998, speaks to a standout amongst the most and most recent approved dermoscopic calculations because of its high affectability and specificity, likewise when used by nonspecialists. The seven criteria were initially tried on 342 melanocytic lesions (117 melanomas and 225 atypical nevi) and were decided for their successive relationship with melanoma [41–43]. Three of them were characterized as significant criteria (atypical system (score 2), blue-white veil (score 2) and atypical vascular pattern score 2)), though the staying four were considered to be minor (irregular streaks (score 1), irregular dots/globules (score 1), irregular blotches (score 1), and regression structures (score 1)). The seven-point checklist for the dermoscopic separation between benign melanocytic lesions and melanoma (scores in sections). At least three shows melanoma.

3.3 CASH method

The CASH [44] is used for the dermatoscopic differentiation between benign melanocytic lesions and melanoma (scores in brackets) as shown in **Table 3**. Add up the scores for a total CASH score (2–17): CASH score of 7 or less is likely benign, otherwise the lesion is suspicious of melanoma.

3.4 The ABCD method

The ABCD rule of dermoscopy was the primary dermoscopy calculation made to help separate benign from malignant melanocytic tumors. This calculation, which

Criteria	Low	Medium	High
Colors: few vs. many White, black, red, blue, Light brown, dark brown (Score: 1 point/color)	1–2 colors (1–2 points)	3–4 colors (3–4 points)	5–6 colors (5–6 points)
Architecture: order vs. disorder (Score: 0–2 points)	None or mild disorder (no points)	Moderate disorder (1 point)	Marked disorder (2 points)
Symmetry vs. asymmetry, Contor, colors and structures (Score: 0–2 points)	Symmetry in 2 axes (no points)	Symmetry in 1 axis (1 point)	No symmetry (2 points)
Homogeneity vs. heterogeneity, dots/globules, blotches, pigment network, blue-white veil, polymorphous vessels, regression, streaks (Score: 1 point/structure)	One structure (1 point)	Two types of structure (2 points)	≥3 structures (3–7 points)

Table 3.
Suspicion for melanoma.

was depicted by Stolz, was created to quantitatively address the significant inquiry in dermoscopy of whether a melanocytic skin lesion under scrutiny is considerate, suspicious (borderline), or malignant. Construct just in light of four dermatoscopic criteria, this technique is moderately simple to learn and apply. The ABCD dermoscopy technique has been widely contemplated, and it has been demonstrated that it enhances the symptomatic execution of clinicians assessing pigmented skin injuries. It is the conclusion of some that this strategy might be especially appropriate for clinicians with constrained dermoscopy encounter. The criteria that consolidate to make the ABCD rule of dermoscopy are asymmetry, border, color, and differential structures. To use these criteria, a scoring framework was produced to compute the total dermoscopy score (TDS) utilizing a straight condition. With this TDS, a reviewing of lesions is conceivable regarding the likelihood that the lesions under investigation are malignant [41, 42]. The likelihood of melanoma depends on adding up the scores of different features as shown in **Table 4**.

3.5 CHAOS and clues

A modified form of pattern analysis [45] looks for CHAOS (asymmetry of structure and/or color) and no less than one clue to diagnose malignancy. It can be

Criteria	Score	Weight	Result
Asymmetry, Perpendicular axes: contour, colors and structures	0–2	1.3	0–2.6
Borders, 8 segments: abrupt ending of pigment pattern	0–8	0.1	0–0.8
Colors, White, black, red, blue-gray, light-brown (tan), dark-brown	1–6	0.5	0.5–3.0
Dermatoscopic structures or differential structural components (pigment network, aggregated globules, branched streaks, structureless areas, dots)	1–5	0.5	0.5–2.5
Total score	Benign		<4.76
	Suspicious		4.76–5.45
	Melanoma		>5.45

Table 4.
Total dermoscopic score of ABCD rule.

Benign	If not, then consider the following:	
Lonely	Unsightly duckling	Score 1
Irregular	Asymmetrical pigmentation pattern or > 1 color	Score 1
Nervous	Nervous patient/**C**hanging lesion	Score 1
Known	Known clues to malignancy	Score 1

Table 5.
Evaluation operation of BLINCK algorithm.

connected to melanocytic and non-melanocytic lesions. Patterns are portrayed by different components of a similar sort: lines, dots, clods, circles, pseudopods (a line with a bulbous end), and structureless regions. Structureless zones are comprised of colors: black, dark brown, light brown, gray, blue, orange, yellow, white, red, and purple.

- A single pattern or a single color is a symmetrical structure, that is, benign.
- Two patterns can have one pattern inside the other pattern or the two patterns may be regularly distributed. Such lesions have a symmetrical structure. Two patterns can be distributed asymmetrically.
- Multiple patterns/colors may result in symmetrical structure if forming concentric zones. Otherwise, they result in an asymmetrical structure.

Asymmetrical patterns should prompt searching for particular clues to malignancy. The clues to malignancy are: thick reticular lines, gray or blue structures of any kind, pseudopods or radial lines at the periphery, black dots in the periphery, eccentric structureless area of any color, polymorphous vascular pattern, white lines, parallel lines on ridges, and large polygons.

3.6 The BLINCK algorithm

The BLINCK algorithm has been conceived to recognize malignant lesions, especially nodular melanoma, as this tumor regularly needs customary dermatoscopic highlights. It can likewise be utilized for non-melanocytic lesions [46]. **Table 5** summarizes the evaluation operation of the BLINCK algorithm. Clues to malignancy are: atypical network, segmental streaks, irregular black dots/globules/clods, eccentric structureless zone, irregular blue or gray color, polymorphous/arborising/glomerular vessels, and parallel ridge pattern or diffuse irregular brown/black pigmentation in acral lesion. A score of ≥ 2 requires biopsy.

3.7 TADA

TADA is an acronym for Triage Amalgamated Dermoscopic Algorithm. TADA does not require a determination to be made to choose if the lesion ought to be extracted or alluded to a specialist [47]. TADA is accounted for to have sensitivity 94.8% and specificity 72.3% for malignant skin lesions. The first step is to determine whether the lesion has features of: angioma, dermatofibroma, and/or seborrhoeic keratosis. If yes, exclude from further analysis. If no, is there any architectural disorder? If there is architectural disorder, does the lesion have one or more of the following six predictive factors?: starburst pattern, blue-black or gray structures, shiny white structures, negative network, ulcer/erosion, and/or vessels. If yes, consider excision or refer. If no, the lesion is likely to be benign. Any doubt, follow-up or refer.

4. The proposed technique

In this research, we propose a novel technique to remove hair pixels from dermoscopic images. The YIQ (luminance (Y), hue (I), and saturation (Q). The first component, luminance, represents gray scale information, while the last two components make up chrominance (color information)) or National Television System Committee (NTSC: the analogue television system used in North America and Japan) color space is chosen because the hair pixels are well demonstrated by only luminance (Y-channel) image, for example, compared to RGB as shown in **Figure 2**. In addition to Red, Green, and Red (RGB) color space, the Hue, Saturation, and Value (HSV) and YCbCr (Y is the brightness (luma), Cb is blue minus luma (B-Y), and Cr is red minus luma (R-Y)) color spaces present the hair pixels in more than one channel too. This issue complicates the hair removal task and may affect the performance.

The Y-channel image is partitioned into 256 non-overlapped blocks. During experimental studies, several block sizes are tested such as 4×4, 8×8, 16×16, and so on. We concluded that the implementation of block size 16×16 introduced better results for inpainting stage as compared with other block sizes. For each block, morphological operators and histogram analysis are implemented to detect hair pixels and inpainting operation as well to replace hair pixels by non-hair skin pixels. This section describes the proposed algorithm for automatic hair detection and inpainting operations. To achieve the aims of this research, **Figure 3** describes the work mechanism, and each step is described in the following subsections.

4.1 Color space conversion

As depicted in **Figure 4**, the conversion operation from the input image (RGB) into YIQ color space.

4.2 Hair detection

To detect hair pixels, a morphological "bottom-hat" operation is implemented on Y-channel image, returning the image minus the morphological closing of the image (dilation followed by erosion) to highlight dark hair on a light background as shown in **Figure 5**. Because the image closing expands the white areas in an

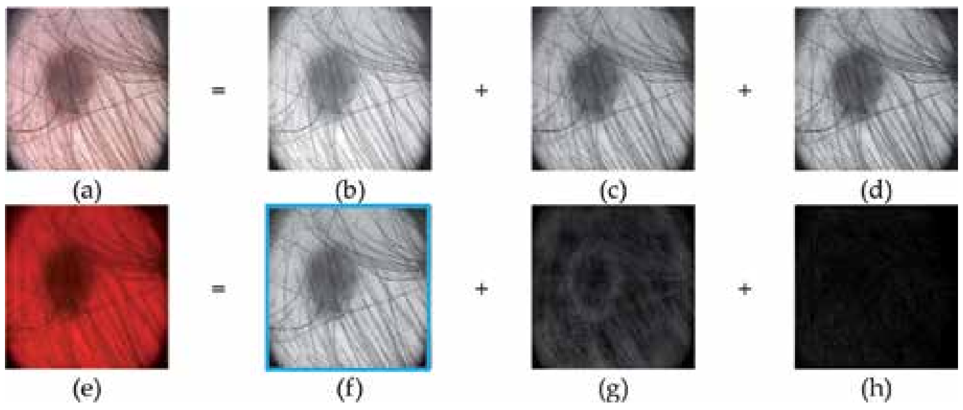


Figure 2.
A digital dermoscopic image presented in RGB (a-d) and YIQ (e-h) color spaces.

image but does not significantly alter those areas which are already white, the only areas left after subtracting the original are those that were originally black but surrounded by white. In general, bottom-hat filtering produces highlighted areas, which more truly follow the shape of the hair. However, the main motivation behind utilizing a bottom-hat filter is still the ability to better preserve the true shape of the hair.

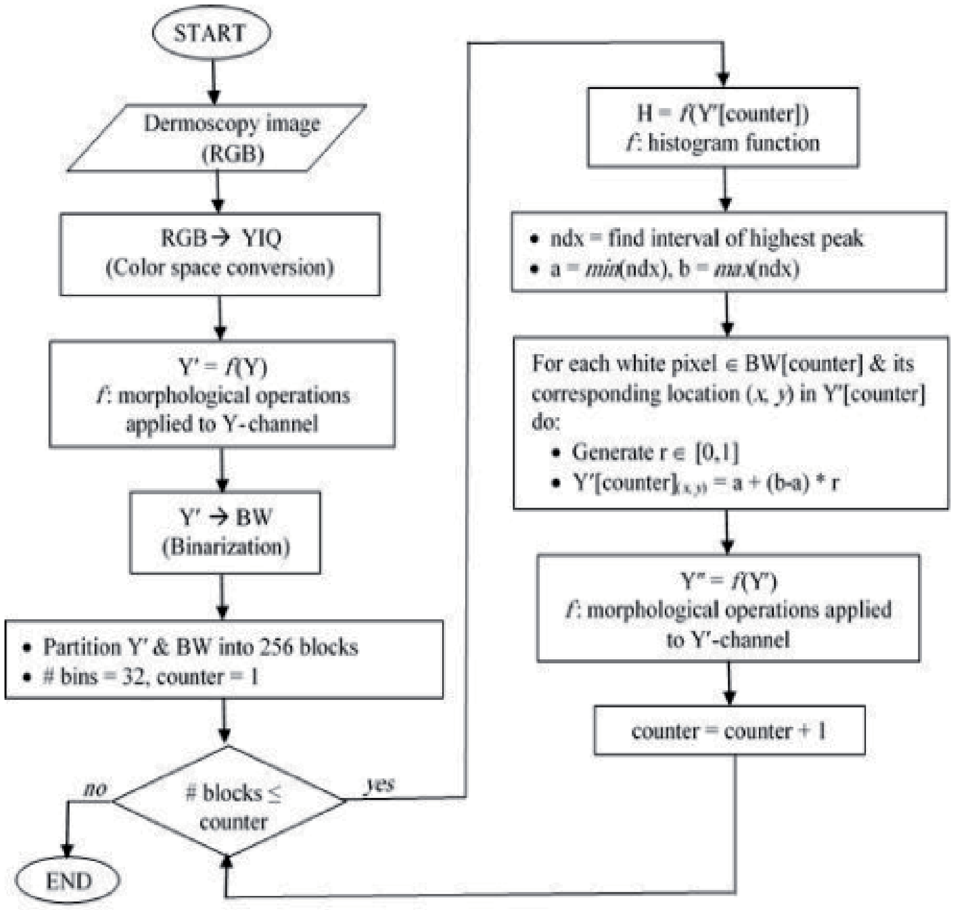

START
Dermoscopy image (RGB)
RGB→ YIQ (Color space conversion)
Y' = f(Y) f: morphological operations applied to Y-channel
Y' → BW (Binarization)
• Partition Y' & BW into 256 blocks
• # bins = 32, counter = 1
# blocks ≤ counter
no
yes
END
H = f(Y'[counter]) f: histogram function
• ndx = find interval of highest peak
• a = min(ndx), b = max(ndx)
For each white pixel ∈ BW[counter] & its corresponding location (x, y) in Y'[counter] do:
• Generate r ∈ [0,1]
• Y'[counter](x,y) = a + (b-a) * r
Y'' = f(Y') f: morphological operations applied to Y'-channel
counter = counter + 1

Figure 3.
Flowchart of the proposed method.

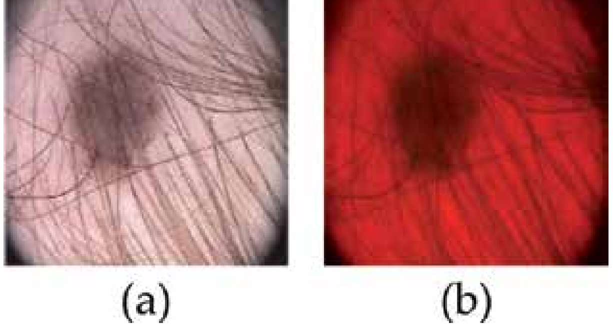


Figure 4.
RGB (a) and YIQ (b) color spaces.

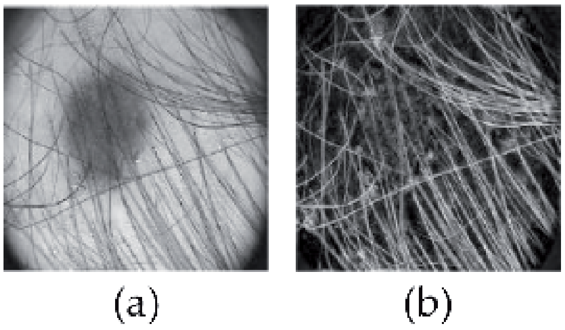


Figure 5.
Hair detection. (a) Y-channel image. (b) Result of bottom-hat operation.

4.3 Binary image conversion

As shown in **Figure 6** is the binarization operation of the image resulted from the implementation of bottom-hat operation on repaired Y-channel image.

4.4 Inpainting operation

Divide the repaired Y-channel and the binarized image into 256 non-overlapped blocks. During experimental studies, several block sizes are tested such as 4×4, 8×8, 16×16, and so on. We concluded that the implementation of block size 16×16 introduced better results for inpainting stage as compared with other block sizes.

a. *For each block do*

- Apply histogram function using 32 bins. The histogram function is *imhist* constructed from the image processing toolbox in the MATLAB software. The first parameter used is the sub-image of size 16x16 and the second parameter is the number of bins which is equal to 32 bins. Based on experimental studies, several number of bins tested and found that 32 bins are sufficiently utilized the intensity pixels ranged in [0, 1] into 32 intervals of size 0.0313 each. Furthermore, there were no improvements when number of binds was increased over 32 bins.

- Find the bin number that contains maximum occurrences (highest peak) of gray scale pixels in each sub-image or block.

- Find locations of white pixels in binary subimage.

- Let a = interval lower value and b = interval upper value.

- *For each white pixel do*

  1. Generate a random number r in [0,1].

  2. Replace the pixel in the Y-channel by using Eq. (1):

$$a + (b - a) \cdot r \tag{1}$$

where the purpose of having r is to keep a dynamic change in the repaired pixel value among all repaired pixels in each block.

- *End*
- Perform the morphological "close" operation (dilation followed by erosion) on repaired Y-channel image as depicted in **Figure 7(b)**.

b. *End*

4.5 Repaired RGB image

The resulted image from the inpainting process as discussed earlier in the previous subsection is subsequently used as an input image to the conversion operation to the RGB color space as depicted in **Figure 8**.

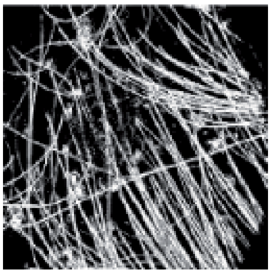


Figure 6.
Result of the binarization operation.

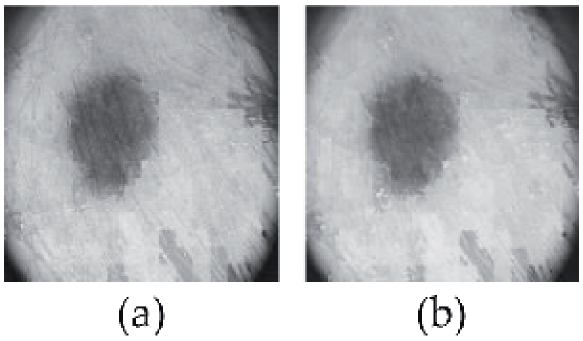


Figure 7.
Repaired Y-channel. (a) Y′-channel before close operation and (b) Y″-channel after close operation.

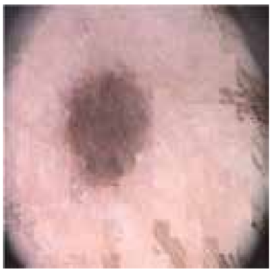


Figure 8.
The repaired RGB image.

5. Results and discussions

The experiments are executed on processor Intel, core i3-2330 M @ 2.20GHz and RAM 4GB. The system type is windows 7 ultimate of 64-bit OS and the software used for research implementation is MATLAB R2013a. The proposed methodology is tested on PH^2 dataset [10]. It consists of 200 8-bit RGB dermoscopic images of melanocytic lesions with a resolution of 768×560 pixels. The dermoscopic images were obtained at the Dermatology Service of Hospital Pedro Hispano, Portugal under the same conditions through Tuebinger Mole Analyzer system using a magnification of 20×. The efficiency of the proposed algorithm is the detection and removal of thin/thick and light/dark hair from dermoscopic images with the preservation of the texture pattern, shape, and colors of skin lesion. Furthermore, any dermoscopic image does not contain hair, the algorithm preserves its features. **Figure 9** depicts a sample of results consists of five input images as an initial stage sorted in the first row, and accordingly, their output images as a final stage appear in the last row after the implementation process of the proposed algorithm as discussed earlier represented by bottom arrows as an intermediate step.

The statistical analysis based on the metrics of sensitivity, specificity, and diagnostic accuracy was used to determine the performance of hair detection and inpainting operation. Our proposed algorithm reports a true positive rate (sensitivity) of 97.36%, a false positive rate (fall-out) of 4.25%, and a true negative rate (specificity) of 95.75%. The diagnostic accuracy achieved is recorded at level high of 95.78%. To estimate the accuracy of the proposed algorithm and to quantify the automatic hair detection error, quantitative evaluations were performed using three statistical metrics: sensitivity or true detection rate (TDR), specificity or true negative rate (TNR), and diagnostic accuracy (DA). TDR measures the rate of pixels which were classified as hair by both the automatic algorithm and the medical expert, and FPR measures the rate of pixels which were not classified as hair by both the automatic segmentation and the medical expert. These metrics are calculated using Eqs. (2)–(5) as follows:

$$\text{sensitivity (TDR)} = \frac{TP}{TP + FN} \times 100 \tag{2}$$

$$\text{specificity (TNR)} = \frac{TN}{TN + FP} \times 100 \tag{3}$$

$$\text{fall-out (FPR)} = \frac{FP}{FP + TN} \times 100 \tag{4}$$

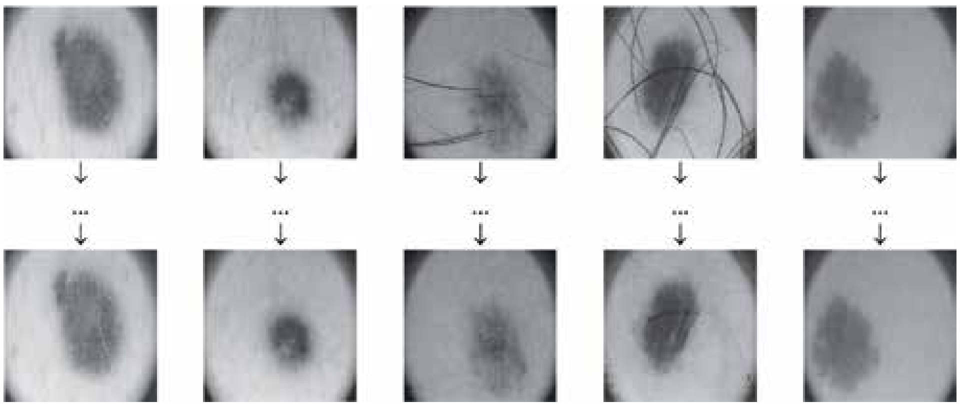


Figure 9.
Sample of results.

$$\text{diagnostic accuracy (DA)} = \frac{TP + TN}{TP + FN + FP + TN} \times 100 \tag{5}$$

where TP, FP, FN, and TN stand for the number of true positive, false positive, false negative, and true negative, respectively. The quantitative results of the proposed algorithm are summarized in **Table 6**. They were calculated as follows:

- False negative (FN): find the differences between the repaired Y-channel (Y″) and the original Y-channel, apply a binarization operation, and then count the white pixels. The results of these sequence of operations are depicted in **Figure 10**.

- True positive (TP): apply the binarization operation on the hair segmented image (Y′) yields to the hair segmented binary image (BW). Visually, it is better to represent the white pixels which are hair pixels in red color and black pixels for non-hair pixels in white background as shown in **Figure 11(a, c)**. The white pixels exist in BW and not exist in the images shown in **Figure 10(d, h)** are counted and preserved in another images as true positive pixels shown in **Figure 11(b, d)**.

- True negative (TN): perform the complement operation on the hair segmented binary image (as shown **Figure 11(a, c)**) yields to the images shown in **Figure 12(a, b)**, respectively. The TN is the count of the white pixels exist in the complement image.

- False positive (FP): count of the remained white pixels.

Count		**# Hair pixels (predicted)**	
		Class = Yes	**Class = No**
# Hair pixels (actual)	Class = Yes	TP (1,924,779)	FN (52,256)
	Class = No	FP (3,664,600)	TN (82,521,688)

Table 6.
Performance evaluation (confusion matrix).

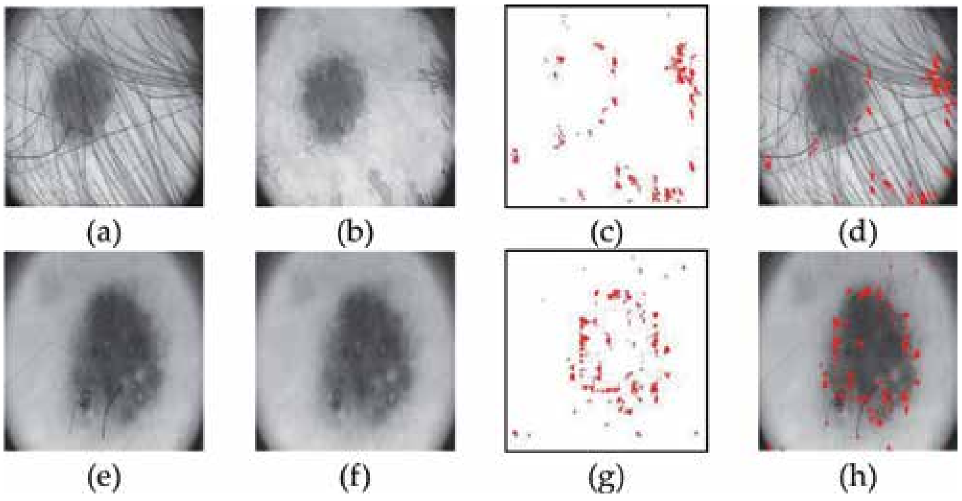


Figure 10.
False negative calculation. (a, e) Y-channel. (b, f) Repaired Y-Channel (Y″). (c, g) Differences between (a, b) and (e, f) illustrated by red dots. (d, h) Y-channel with false negative pixels represented by red dots.

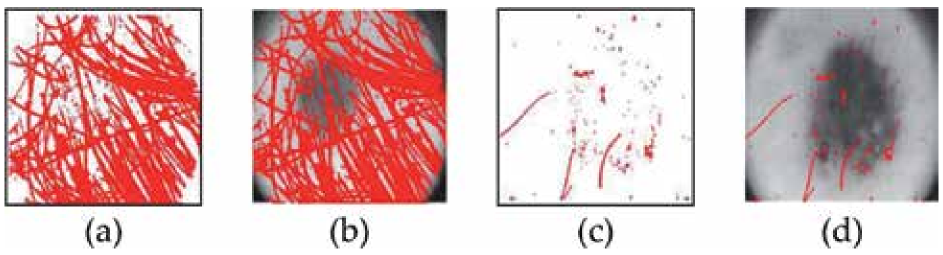


Figure 11.
True positive calculation. (a, c) Hair segmented binary image. (b, d) Truly classified hair pixels.

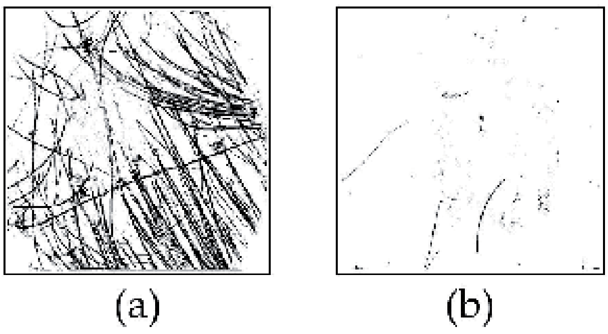


Figure 12.
Results of complement operation performed on binarized images.

Artifact detection method	TDR (%)	TNR (%)	FPR (%)	DA (%)	# test images
The proposed algorithm	97.36	95.75	4.25	95.78	200
Multi-resolution [30]	93.2	—	4	88.3	50
Top-hat operator [36]	—	—	—	72.5	40
DullRazor [31]	70.2	—	33.4	48.6	50
Fast image restoration (FIR) + line segment detection (LSD) [20]	98.27	93.75	—	96.10	299

Table 7.
Comparison of the hair detection algorithms.

Unfortunately, we could not find a common database that can be shared with other researchers and there is no related work used PH^2 dataset [10] to compare the proposed algorithm with others. However, **Table 7** compares the proposed hair detection algorithm with some other methods.

6. Conclusion and future work

In this study, a fast and effective method is proposed for hair-occluded removal in dermoscopic images. The implementation of the hair removal process is divided into two main stages: hair detection and inpainting. In hair detection, a morphological bottom-hat operation is implemented on Y-channel image of the YIQ color space followed by a binarization operation. In inpainting, the repaired Y-channel is partitioned into 256 non-overlapped blocks and for each block, white pixels are replaced by locating the highest peak of using a histogram function and a morphological close operation.

Our achieved results indicate high accuracy, and the proposed method can be dedicated to Dermatologists as a pre-processing stage before the lesion segmentation and classification. However, our proposed algorithm reports a true positive rate (sensitivity) of 97.36%, a false positive rate (fall-out) of 4.25%, and a true negative rate (specificity) of 95.75%. The diagnostic accuracy achieved is recorded at a high level of 95.78%.

The following opportunities are suggested for future work:

- Allocate a dataset to be common among researchers.
- Other artifacts such as air bubbles can be added for further studies.

Acknowledgements

The author would like to thank Dr. Saeb Zaqout for his valuable comments and constructive criticism of this manuscript.

Author details

Ihab Zaqout
Department of Information Technology, Faculty of Engineering and Information Technology, Al-Azhar University, Gaza, Gaza Strip, Palestine

*Address all correspondence to: i.zaqout@alazhar.edu.ps

References

[1] Melanoma History [Internet]. 2016. Available from: https://www.news-medical.net/health/Melanoma-History.aspx [Accessed: 13 August 2016]

[2] Rebecca VW, Sondak VK, Smalley KS. A brief history of melanoma: From mummies to mutations. Melanoma Research. 2012;**22**:114-122. DOI: 10.1097/CMR.0b013e328351fa4d. PMC 3303163. PMID 22395415

[3] Progress and timeline of melanoma [Internet]. 2016. Available from: http://www.cancerprogress.net/timeline/melanoma [Accessed: 14 August 2016]

[4] Melanoma skin cancer treatment [Internet]. 2016. https://www.cancer.org/cancer/melanoma-skin-cancer/treating.html [Accessed: 15 August 2016]

[5] Stewart BW, Wild CB. World Cancer Report 2014. World Health Organization, International Agency for Research on Cancer. Geneva, Switzerland: WHO Press; 2014. DOI: 10.3945/an.116.012211

[6] Iyatomi H. Computer-based diagnosis of pigmented skin lesions. In: New Developments in Biomedical Engineering. Japan: InTech; 2010. pp. 183-200. DOI: 10.5772/7621

[7] Garnavi R. Computer-aided diagnosis of melanoma [thesis]. Australia: University of Melbourne; 2011

[8] American Academy of Dermatology (AAD). Benchmark "Skin Cancer" [Internet]. 2016. Available from: http://www.aad.org/skin-conditions/dermatology-a-to-z/skin-cancer [Accessed: 05 December 2016]

[9] Zaqout I. Diagnosis of skin lesions based on dermoscopic images using image processing techniques. International Journal of Signal Processing, Image Processing and Pattern Recognition. 2016;**9**(9):189-204. DOI: 10.14257/ijsip.2016.9.9.18

[10] PH^2 Dataset [Internet]. 2016. Available from: https://www.fc.up.pt/addi/ph2%20database.html [Accessed: 05 December 2016]

[11] Abbas Q, Celebi ME, Garcia IF. Hair removal methods: A comparative study for dermoscopy images. Biomedical Signal Process and Control. 2011;**6**(4):395-404. DOI: 10.1016/j.bspc.2011.01.003

[12] Prathamesh AS, Gumaste PP. A review of existing hair removal methods in dermoscopic images. IOSR Journal of Electronics and Communication Engineering (IOSR-JECE). 2015;**1**:73-76. DOI: 10.1515/mathm-2016-0001

[13] Satheesha TY, Satyanarayana D, Giriprasad MN. A pixel interpolation technique for curved hair removal in skin images to support melanoma detection. Journal of Theoretical and Applied Information Technology. 2014;**70**(3):559-565

[14] Nasonova A, Nasonov A, Krylov A, Pechenko I, Umnov A, Makhneva N. Image warping in dermatological image hair removal. In: 11th International Conference on Image Analysis and Recognition (ICIAR '14); 22-24 October 2014; Vilamoura, Portugal. 2014. pp. 159-166

[15] Mahmood AF, Mahmood HA. Artifact removal from skin dermoscopy images to support automated melanoma diagnosis. Al-Rafadain Engineering Journal. 2015;**23**(5):22-30

[16] Das P, Gangopadhyay M. Removal of hair particles from skin disease images using pixel based approach. International Journal of Computer Science and Information Technologies. 2015;**6**(5):4154-4158

[17] Abbas Q, Garcia IF, Celebi ME, Ahmad W. A feature-preserving hair removal algorithm for dermoscopy images. Skin Research and Technology. 2011;**19**(1):27-36. DOI: 10.1111/j.1600-0846.2011.00603.x

[18] Mirzaalian H, Lee TK, Hamarneh G. Hair enhancement in dermoscopic images using dual-channel quaternion tubularness filters and MRF-based multi-label optimization. IEEE Transactions on Image Processing. 2014;**23**(12):5486-5496. DOI: 10.1109/TIP.2014.2362054

[19] Kretzler M. Automated curved hair detection and removal in skin images to support automated melanoma detection [thesis]. Case Western Reserve University; 2013

[20] Okuboyejo D, Olugbara O, Odunaike S. Unsupervised restoration of hair-occluded lesion in dermoscopic images. In: Medical Image Understanding and Analysis Conference (MIUA '2014); London, UK. 2014. pp. 91-96

[21] Koehoorn J, Sobiecki A, Boda D, Diaconeasa A, Doshi S, Paisey S, Jalba A, Telea A. Automated digital hair removal by threshold decomposition and morphological analysis. In: 12th International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing (ISMM' 15); Reykjavik, Iceland. 2015. pp. 15-26

[22] Koehoorn J, Sobiecki A, Rauber P, Jalba A, Telea A. Efficient and effective automated digital hair removal from dermoscopy images. Mathematical Morphology—Theory and Applications. 2016;**1**(1):1-17. DOI: 10.1515/mathm-2016-0001

[23] Zhou H, Chen M, Gass R, Rehg JM, Ferris L, Ho J, Drogowski L. Feature-preserving artifact removal from dermoscopy images. In: Proceeding SPIE 6914; Medical Imaging, San Diego, CA. 2008. pp. 1-9

[24] Hearn J. Competitive medical image segmentation with the fast marching method [thesis]. Case Western Reserve University; 2008

[25] Sultana A, Ciuc M, Radulescu T, Wanyu L, Petrache D. Preliminary work on dermatoscopic lesion segmentation. In: 20th European Signal Processing Conference (EUSIPCO '12); 27-31 August 2012, Bucharest, Romania; 2012. pp. 2274-2277

[26] Bibiloni P, Gonzalez-Hidalgo M, Massanet S. Skin hair removal in dermoscopic images using soft color morphology. In: 16th Conference on Artificial Intelligence in Medicine (AIME '17); 21-24 June 2017, Vienna, Austria. 2017. pp. 322-326

[27] Acebuque-Salido A, Ruiz C. Using morphological operators and inpainting for hair removal in dermoscopic images. In: Proceedings of the Computer Graphics International Conference (CGC '17); 27-30 June 2017; Yokohama, Japan. 2017

[28] Al-Abayechi A, Guo X, Tan W-H, Jalab HA. Automatic skin lesion segmentation with optimal color channel from dermoscopic images. ScienceAsia. 2014;**40S**(1):1-7. DOI: 10.2306/scienceasia1513-1874.2014.40S.001

[29] Chakraborti D, Kaur R, Umbaugh S, LeAnder R. An effective hair detection algorithm for dermoscopic melanoma images of skin lesions. In: SPIE Proceedings on Applications of Digital Image Processing XXXIX; 27 September 2016; San Diego, California, USA. 2016. p. 9971. DOI: 10.1117/12.2236565

[30] Toossi MT, Pourreza HR, Zare H, Sigari MH, Layegh P, Azimi A. An effective hair removal algorithm for

dermoscopy images. Skin Research and Technology. 2013;**19**(3):230-235. DOI: 10.1111/srt.12015

[31] Lee T, Ng V, Gallager R, Coldman A, McLean D. DullRazor: A software approach to hair removal from images. Computers in Biology and Medicine. 1997;**27**(6):533-543. DOI: 10.1016/S0010-4825(97)00020-6

[32] Kiani K, Sharafat AR. E-Shaver: An improved DullRazor for digitally removing dark and light-colored hairs in dermoscopic images. Computers in Biology and Medicine. 2011;**41**(3):139-145. DOI: 10.1016/j.compbiomed.2011.01.003

[33] Fiorese M, Peserico E, Silletti A. VirtualShave: Automated hair removal from digital dermatoscopic images. In: 33rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS '11); Boston, Massachusetts, USA. 2011. pp. 5145-5148

[34] Bertalmio M, Sapiro G, Caselles V, Ballester C. Image inpainting. In: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, ACM SIGGRAPH; NY, USA. 2000. pp. 417-424

[35] Huang A, Kwan S, Chang W, Liu M, Chi M, Chen G. A robust hair segmentation and removal approach for clinical images of skin lesions. In: Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS '13); 2013. pp. 3315-3318

[36] Xie FY, Qin SY, Jiang ZG, Meng RS. PDE-based unsupervised repair of hair-occluded information in dermoscopy images of melanoma. Computerized Medical Imaging & Graphics. 2009;**33**(4):275-282. DOI: 10.1016/j.compmedimag.2009.01.003

[37] Perona P, Malik J. Scale-space and edge detection using anisotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI). 1990;**12**(7):629-639. DOI: 10.1109/34.56205

[38] Bornemann F, März T. Fast image inpainting based on coherence transport. Journal of Mathematical Imaging and Vision. 2007;**28**(3):259-278. DOI: 10.1007/s10851-007-0017-6

[39] Telea A. An image inpainting technique based on the fast marching method. Journal of Graphics Tools. 2004;**9**(1):25-36. DOI: 10.1080/10867651.2004.10487596

[40] Menzies SW, Ingvar C, Crotty KA, McCarthy WH. Frequency and morphologic characteristics of invasive melanomas lacking specific surface microscopic features. Archived Dermatology. 1996;**132**(10):1178-1182. DOI: 10.1001/archderm.1996.03890340038007. PMID: 8859028

[41] Johr RH. Dermoscopy: Alternative melanocytic algorithms-the ABCD rule of dermatoscopy, Menzies scoring method, and 7-point checklist. Clinics in Dermatology. 2002;**20**(3):240-247. DOI: 10.1016/S0738-081X(02)00236-5. PMID: 12074859

[42] Argenziano G, Fabbrocini G, Carli P, De Giorgi V, Sammarco E, Delfino M. Epiluminescence microscopy for the diagnosis of doubtful melanocytic skin lesions. Comparison of the ABCD rule of dermatoscopy and a new 7-point checklist based on pattern analysis. Archives of Dermatology. 1998;**134**(12):1563-1570. DOI: 10.1001/archderm.134.12.1563. PMID: 9875194

[43] Argenziano G, Catricalà C, Ardigo M, Buccini P, De Simone P, Eibenschutz L, et al. Seven-point checklist of dermoscopy revisited.

British Journal of Dermatology. 2011;**164**(4):785-790. DOI: 10.1111/j.1365-2133.2010.10194.x. PMID: 21175563

[44] Henning JS, Dusza SW, Wang SQ, Marghoob AA, Rabinovitz HS, Polsky D, Kopf AW. The CASH algorithm for dermoscopy. Journal of the American Academy of Dermatology. 2007;**56**(1):45-52. DOI: 10.1016/j.jaad.2006.09.003. PMID: 17190620

[45] Rosendah C, Cameron A, Tschandl P, Bulinska A, Zalaudek I, Kittler H. Prediction without pigment: A decision algorithm for non-pigmented skin malignancy. Dermatology Practical & Conceptual. 2014;**4**(1):59-66. DOI: 10.5826/dpc.0401a09. PMID: 24520516

[46] Bourne P, Rosendahl C, Keir J, Cameron A. BLINCK—A diagnostic algorithm for skin cancer diagnosis combining clinical features with dermatoscopy findings. Dermatology Practical & Conceptual. 2012;**2**(2):0202a12. DOI: 10.5826/dpc.0202a12. PMID: 23785600

[47] Rogers T, Marino M, Dusza SW, Bajaj S, Marchetti MA, Marghoob A. Triage amalgamated dermoscopic algorithm (TADA) for skin cancer screening. Dermatology Practical & Conceptual. 2017;**7**(2):39-46. DOI: 10.5826/dpc.0702a09. PMID: 28515993

Chapter 6

How to Keep the Binary Compatibility of C++ Based Objects

Donguk Yu and Hong Seong Park

Abstract

This chapter proposes the binary compatibility object model for C++ (BiCOMC) to provide the binary compatibility of software components in order to share objects among C++ based executable files such as .exe, .dll, and .so. In addition, the proposed model provides the method overriding and overloading, multiple inheritance, and exception handling. This chapter illustrates how to use the proposed model via a simple example in the Windows and Linux environment. The proposed method is validated by application examples and comparisons with known object models such as C++, COM, and CCC in terms of the call time of a method during execution and the binary compatibility such as reusability due to interface version and the types of compilers. Also this chapter shows that BiCOMC-based components compiled with Microsoft Visual C++ and GCC can call each other and the interface version problems are resolved.

Keywords: binary compatibility, component, C++, interface, object model, Windows

1. Introduction

Nowadays, lots of software have been developed based on components because of reusability and composability which can make development and maintenance easier. The component-based approach has some advantages that the cost and time required for maintenance and development can be reduced through the combination of components and the property of encapsulation. In particular, the robot software platforms such as OPRoS [1–3], openRTM [4, 5], and OROCOS [6], which are examples of component-based systems, have been using components of dynamic libraries, such as .dll and .so, in order for components to be able to be developed and maintained with ease. Despite these advantages, there are some hurdles in reusing of the components. The biggest hurdle is the binary compatibility issue of C++ based components, which is whether or not the components in the binary code compiled by a type of compiler are executing together on the same operating system with other components compiled by its old version compiler or other types of compilers. In practice, the number of components implemented with programming languages such as Java and Python has been increasing because those languages do not cause the binary compatibility problems. However, C++ is an important programming language needed for the control of automation machines/devices such as robots and SW-based PLCs because it provides fast performance [7, 8]. In addition, there have been lots of C++ based components or modules developed and stably used till now for industrial/office/home automation. Because the components were compiled by different types and/or versions of compilers, it is necessary to reuse them effectively. Therefore, the binary compatibility of C++ components (or objects) should be resolved. Note that examples of the components (or objects) are classes, variables, and methods, where a class can include variables, methods, and zero or more classes.

There are following two types of methods in dynamical sharing of C++ classes: the C-based dynamic library method and the sharing method supported by a compiler. Because the C-based dynamic library method cannot directly share classes, the executable files such as .exe, .dll, and .so refer to the abstract class of the same header files and deliver the address of the instance of the class. The sharing method by compiler can directly share classes. But the method cannot share instances of classes compiled by different types of compilers [9]. In other words, there is a serious problem that the sharing of classes is applicable only to the same compiler, which makes spreading of C++ based components difficult.

For instance, let us consider two types of components in the Windows environment, which are made using Visual C++ from Microsoft (MSVC) and GCC from GNU, respectively. A MSVC-based component and a GCC-based component are not mutually compatible in most cases even though they have been made in the same Windows environment. This situation occurs due to the binary compatibility problem. The binary compatibility problem generally causes the situations where the methods of the object cannot be suitably called or its operation is not properly executed. And then the system can enter into a down (or dead) state. To solve the problem, it is necessary to design an object structure compatible in all types of compilers [10].

There have been some researches to solve the binary compatibility problem of C++ objects, examples of which are COM [11], CCC [12], and ZL [13]. COM solves the binary compatibility problem of C++ objects but has some limitations that it should operate in a Windows environment and be supported only by the MSVC compiler [11]. CCC is a library that the classes are composed of only header files to maintain the binary compatibility and designed to use the C++ 11 features [14] to make binary compatible objects with ease. But CCC has a limitation that it can be used only by compilers that support the C++ 11 standard. In other words, CCC does not share objects compiled by compilers not supporting the C++ 11 standard. In the C++ compatible language ZL [13], the binary compatible classes are supported by customized preprocessors and macros. It does not however support multiple inheritances and provides the binary compatibility only for GCC because the compiler for ZL is made from modified version of GCC. In addition COM and CCC do not support the method overloading, but ZL partially supports it. CCC supports exception handling, COM does it in the restricted manner, but ZL does not support it.

This chapter proposes the binary compatibility object model for C++ (BiCOMC) for reusability of software components which provide binary compatibility for sharing objects between C++ executable files in the Windows or the Linux environment. In addition, the proposed model provides the method overloading and overriding, multiple inheritance, and exception handling. And BiCOMC makes each other share the objects generated by different types of compilers such as MSVC, GCC, and ICC. This chapter provides macros of C++ preprocessor for sharing the binary compatible objects easily and independently of the types and versions of compilers. This chapter illustrates how to use the proposed model via a simple example in the Windows and Linux environment. To validate the proposed method, BiCOMC is compared with COM and CCC in terms of the call time during execution and the binary compatibility among interface versions and the types of compilers. Moreover, it is shown that BiCOMC-based components made using both MSVC and GCC can call the methods of each other and the interface version problems are resolved.

In the next section, a binary compatibility object model for C++ (BiCOMC) is proposed, which has the structures of virtual function tables including multiple inheritance and the casting algorithm for conversion of interfaces. It is shown that method overloading and multiple inheritances are supported. In Section 3, the component based on BiCOMC is defined, and its implementation is shown using examples. Section 4 suggests two examples. One is a simple example to illustrate how to use the proposed method in Windows and Linux environment. The other is an example of a robot application to validate the proposed method, where the application consists of three components compiled by different types of compilers. In Section 5, the binary compatibility and the performance measure of the call time of methods are evaluated. Finally, the conclusions are given in Section 6.

2. Binary compatibility object model for C++ (BiCOMC)

2.1 Object model

A BiCOMC is shown in **Figure 1** and has the interface Object as the root of the hierarchy of the class diagrams. Since interfaces are public and can be used by many objects, they do not have any member variables so that binary compatibility is maintained. Note that the interface Object has one pointer variable as shown in **Figure 1** for sharing of a virtual function table. The point variable is the vftptr pointer for accessing of the table and exists at the topmost of the interface. The structure of the virtual function table is explained later.

Let us consider the case where an executable file A can create an object but a different executable file B can delete the object. In this case, the memory allocated by the file A cannot normally be deleted using the C++ delete operator in the file B because the delete operator of the file B cannot invoke the destructor of the class in

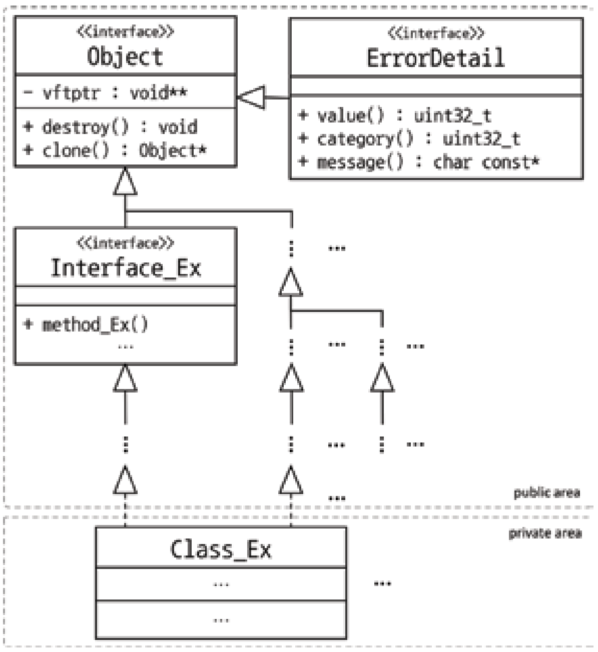


Figure 1.
Class diagram of Object and ErrorDetail interface for BiCOMC.

the file A. To solve this problem, the interface Object has a destroy() method so that objects can be deleted by the executable file which created the objects.

Because objects that are shared externally are exposed in the form of an interface, they cannot know the prototype of the object and then cannot be replicated through the C++ copy constructor. The interface Object has a clone() method that is allowed to clone the object.

When interface methods are implemented, the order of the methods in the virtual function table should be the same as the declaration order of the methods in the interface. So the order of the methods in the interface should not be changed once the interface is open in public. Adding a new method to the interface is allowed if its order is not changed. In other words the insertion of a new method is allowed only as the last method of the interface.

All methods can cause exceptions which are related to the interface ErrorDetail in **Figure 1**. Therefore, only objects implemented in the interface ErrorDetail should be thrown at the occurrence of the exceptions. The interface ErrorDetail has a value() method, a category() method, and a message() method that return an error value, an error category, and a description of the error, respectively.

An interface can inherit only one parent interface, but a class can inherit multiple interfaces. When a class has inherited multiple interfaces, the class has multiple vftptr pointers, which are the addresses of virtual function tables for individually inherited interfaces. The class, which is inherited multiple interfaces, refers to as many virtual function tables as the number of the inherited interfaces, which is shown in **Figure 3**.

2.2 Structure of a virtual function table

The vftptr pointer of the interface Object points to a virtual function table. A virtual function table contains the address of the overridden method and the interface information for the interface. **Figure 2** shows the structure of the virtual

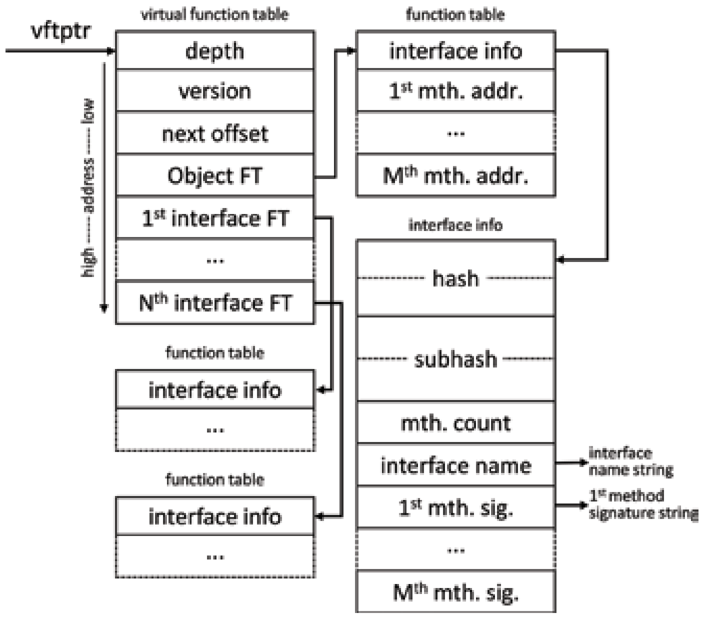


Figure 2.
Structure of a virtual function table.

function table. The sizes of depth, version, and next offset are equal to the size of a void pointer (or void*), respectively.

A virtual function table contains all the parent interfaces inherited by the interface. The first element, depth, of a virtual function table in **Figure 2** stores the number of inheritances from the interface Object to the last interface. If the interface directly inherited by the implementation class is the interface Object, the value of depth is 0. The second element, version, means the version number of BiCOMC for future extension of BiCOMC. The third element, next offset, is used for the objects that inherit multiple interfaces and represents the offset between the current vftptr and the next vftptr, which is explained later. The fourth element Object FT refers to the address of the function table of the interface Object where the actual addresses of overridden methods and the interface information are stored. The remaining elements i-th interface FT points to the function table of the i-th interface (i = 1... N).

The entry interface info in the function table means the interface information. The entry j-th mth. addr. (j = 1... M) is the address of the j-th method. Note that the addresses of the methods are stored in the same order as the order of declaration of the methods.

The interface info includes some elements such as hash, subhash, mth. count, interface name, and j-th mth. sig. (j = 1... M). The hash is a 64bit value, and it is calculated with the name of the interface and the names of inherited interfaces. The same hash means that the name of the interface and the names of the inherited interfaces are the same. The subhash is also a 64bit value, and it is calculated with M method signatures which contain the name of the method, a return type, and parameters' types. The same subhash means that the method definitions of two interfaces are the same. The sizes of the hash and the subhash of the interface may not be the same as the size of void*. Their sizes are calculated in (1) in bytes. If the hash values of two objects are different, the objects are incompatible. The subhash values are used for the method overloading and backward compatibility. So the method signatures should be checked if the subhash values are not matched. Note that the interface name and the j-th mth. sig. (j = 1...M) are null-terminated character strings encoded by UTF-8 and j-th mth. sig. consists of the method name, the return type, and types of parameters:

$$\text{size} = \lceil 16/\text{sizeof}(\text{void}^*) \rceil \times \text{sizeof}(\text{void}^*) \tag{1}$$

As mentioned above, the method signature can distinguish other methods with the same name because the method's signature is based on the method name as well as the types of parameters and the return type. So it can be said that BiCOMC supports the method overloading.

2.3 Structure of virtual function tables in multiple inheritance

In BiCOMC, an object that inherits multiple interfaces has as many vftptr pointers as the number of inherited interfaces. Note that the basic structure of the virtual function table is described in Section 2.2. **Figure 3** shows the structure of the virtual function tables of an object inheriting three interfaces.

In the case where three interfaces have been inherited, three vftptr pointers exist as shown in **Figure 3**. Assume that a 32bit system is used and the next offset of the virtual function **Table 1** is 4, which is the difference between the address of vftptr 2 and the address of vftptr 1 in the instance of class. And the next offset of the virtual function **Table 3** is −8, which is the difference between the address of vftptr 1 and the address of vftptr 3.

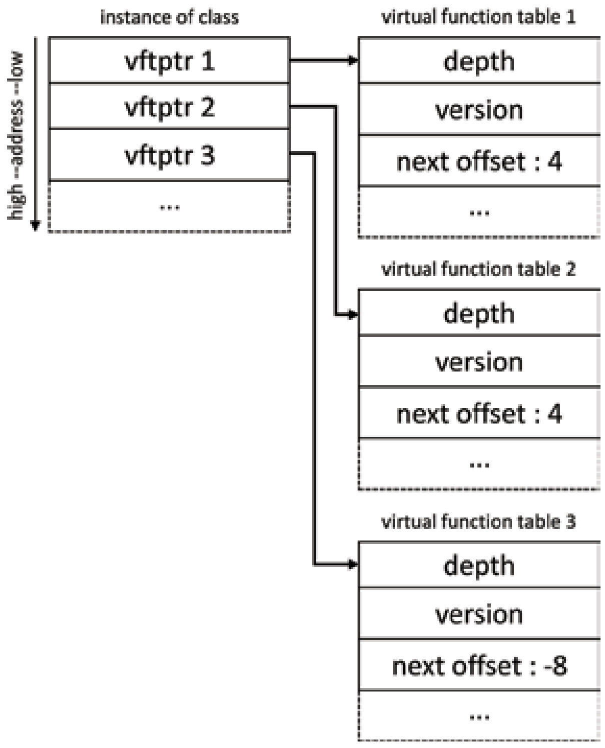


Figure 3.
Structure of virtual function table in the case of multiple inheritance.

BiCOMC			Dynamic library (DLL)					
			MSVC		GCC		ICC	
			9	14	4.5	5.2	14	16
Executable (EXE)	MS VC	9	O	O	O	O	O	O
		14	O	O	O	O	O	O
	GCC	4.5	O	O	O	O	O	O
		5.2	O	O	O	O	O	O
	ICC	14	O	O	O	O	O	O
		16	O	O	O	O	O	O

Table 1.
Tests of the binary compatibility in BiCOMC.

2.4 Interface casting

General casting methods of C++ such as static_cast and dynamic_cast cannot be used among objects created by different types of compilers. Therefore, methods that can cast BiCOMC objects are necessary regardless of a type of compilers used. **Figure 4** shows an algorithm for casting BiCOMC objects to other interfaces.

The casting algorithm gets virtual function tables using two parameters of obj and tgtTable. The algorithm compares the tables of obj with the tables of the target interface in order to check whether or not two interfaces are compatible. NULL is returned if two interfaces are not compatible each other. The seventh, tenth, and

```
 1: function CAST(obj, tgtTable)                 ▷ obj is a instance of interface Object.
 2:     $cntIdx \leftarrow \lceil 16/\text{sizeof(void*)} \rceil$
 3:     $p \leftarrow obj$
 4:     repeat
 5:         $table \leftarrow p.vftptr$
 6:         if $tgtTable[1] \neq table[1]$ then                 ▷ version mismatch
 7:             $p \leftarrow p + table[2]$                   ▷ move to the next interface
 8:             continue
 9:         else if $tgtTable[0] > table[0]$ then  ▷ incompatible count of methods
10:             $p \leftarrow p + table[2]$                   ▷ move to the next interface
11:             continue
12:         end if
13:         $isCompatible \leftarrow$ True
14:         for $i \leftarrow 0, tgtTable[0]$ do
15:             $tInfo \leftarrow tgtTable[i + 3][0]$
16:             $info \leftarrow table[i + 3][0]$
17:             if $tInfo[cntIdx] > info[cntIdx]$ then
18:                 $isCompatible \leftarrow$ False
19:                 break
20:             end if
21:             if $tInfo.hash \neq info.hash$ then               ▷ hash mismatch
22:                 $isCompatible \leftarrow$ False
23:                 break
24:             end if
25:             if $tInfo.subhash \neq info.subhash$ then      ▷ subhash mismatch
26:                 for $j \leftarrow 0, tInfo[cntIdx]$ do
27:                     $tSig \leftarrow tInfo[cntIdx + j + 1]$
28:                     $sig \leftarrow info[cntIdx + j + 1]$
29:                     if $tSig \neq sig$ then
30:                         $isCompatible \leftarrow$ False
31:                         break
32:                     end if
33:                 end for
34:                 if $isCompatible$ = False then
35:                     break
36:                 end if
37:             end if
38:         end for
39:         if $isCompatible$ = True then
40:             return $p$
41:         end if
42:         $p \leftarrow p + table[2]$                       ▷ move to the next interface
43:     until $p \neq obj$
44:     return NULL
45: end function
```

Figure 4.
Algorithm for interface casting.

forty-second lines in **Figure 4** are for processing of multiple inheritance. Lines 21–24 check hash values to test the names of interfaces and inheritance relationships. That is, lines 21–24 compare the hash values and detect whether they are compatible by detecting whether a new interface is added or a different interface name exists. Lines 25–37 compare subhash values, and two interfaces are considered as compatible interfaces if the values are the same. If they are different, it examines method signatures of interface information to check compatibility. An example of interface casting is shown in **Figure 6**.

3. Definition and implementation of component based on BiCOMC model

In this section the component based on BiCOMC is defined, and its implementation is shown using examples. **Figure 5** shows an example of the class Comp_1 based on BiCOMC, where the interface Interface_1 and the interface Interface_2 inherit the interface Object and Interface_3 inherits Interface_2. The definition of the interfaces in **Figure 5** is shown in **Figure 7**. The class Comp_1 is one of the components that provide interfaces Interface_1 and Interface_3. Interface Interface_1 consists of void mth_1(int) and int. mth_2(), and interface Interface_3 inherits interface Interface_2 and consists of void mth_1() and int. mth_2(int).

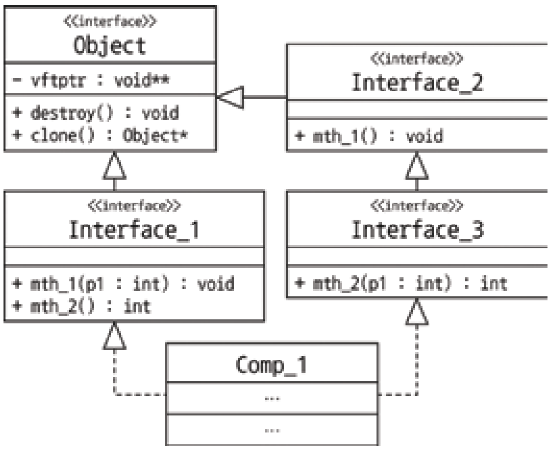


Figure 5.
Class diagram example of relationship between component and interface.

Figure 6 shows an example of the interface casting algorithm in **Figure 4**, which shows that Interface_1 is converted to Interface_2 using bicomc_cast based on the algorithm in **Figure 4**.

The macro BICOMC_INTERFACE defines interfaces as shown in **Figure 7**, using one or two parameters. The first parameter is the name of the interface, and the second parameter is the name of the parent interface and optional. In the case where the second parameter is empty, the interface Object is inherited. The macro BICOMC_DECL_METHOD in **Figure 7** declares the method of the interface. The macro has three parameters as follows: the first parameter is the name of the method, the second parameter is the function type of the method, and the third parameter is the number of parameters of the method. For example, let us consider BICOMC_DECL_METHOD(mth_2, int.(), 0). This macro represents the method int. mth_2() in **Figure 8**.

The interface definition in **Figure 7** is converted into the C++ code of **Figure 8** by C++ preprocessor. Note that C++ code for Interface_2 is not represented. The macro BICOMC_INTERFACE in **Figure 7** is converted to the C++ code related to the inputted interface name such as Interface_1. As seen in the 1st and 17th lines of **Figure 8**, the macro BICOMC_INTERFACE (Interface_1) and BICOMC_ INTERFACE (Interface_3, Interface_2) create the codes of class Interface_1 public Object and class Interface_3 public Interface_2, respectively. The macro BICOMC_ DECL_METHOD is converted to a method which consists of a name, a return type, and parameters. The second, third, and eleventh lines in **Figure 7** are converted into the third–seventh lines, ninth–fourteenth lines, and nineteenth to twenty-fourth lines in **Figure 8**, respectively. From the virtual function table pointed at by the vftptr pointer of the interface Object, the addresses of functions(or methods) are acquired using the inheritance depth of the interface and the order of declarations of the interface methods as shown in the 5th, 11th, and 21st lines in **Figure 8**, and then the actual overridden methods are called. These methods are converted into the function type as shown in **Figure 9** and then stored. Note that ErrorCode is a wrapper class of ErrorDetail in **Figure 1**.

```
1: Interface_1* p1 = new Comp1();
2: Interface_2* p2 = bicomc_cast<Interface_2*>(p1);
```

Figure 6.
An example of interface casting.

```
BICOMC_INTERFACE(Interface_1)
  BICOMC_DECL_METHOD(mth_1, void(int), 1)
  BICOMC_DECL_METHOD(mth_2, int(), 0)
BICOMC_INTERFACE_END(Interface_1)

BICOMC_INTERFACE(Interface_2)
  BICOMC_DECL_METHOD(mth_1, void(), 0)
BICOMC_INTERFACE_END(Interface_2)

BICOMC_INTERFACE(Interface_3, Interface_2)
  BICOMC_DECL_METHOD(mth_2, int(int), 1)
BICOMC_INTERFACE_END(Interface_3)
```

Figure 7.
Definition of interface using macro in **Figure 5**.

```
 1: class Interface_1 : public Object {
 2:   ...
 3:   void mth_1(int p1) {
 4:     void* r;
 5:     if (ErrorDetail* e = vftptr[4][1](this, &r, p1))
 6:       throw e;
 7:   }
 8:   ...
 9:   int mth_2() {
10:     int r;
11:     if (ErrorDetail* e = vftptr[4][2](this, &r))
12:       throw e;
13:     return r;
14:   }
15: };
16: ...
17: class Interface_3 : public Interface_2 {
18:   ...
19:   int mth_2(int p1) {
20:     int r;
21:     if (ErrorDetail* e = vftptr[5][1](this, &r, p1))
22:       throw e;
23:     return r;
24:   }
25: };
```

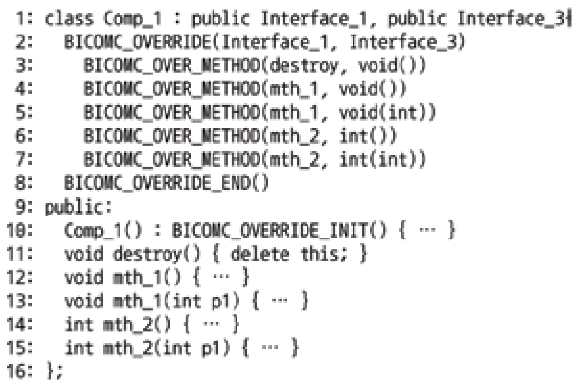
Figure 8.
C++ code of **Figure 7** *after preprocessing.*

In **Figure 9**, parameter I* is the address of the interface instance, the second parameter R* is the address of the variable receiving the return value of the method, and other parameters (P_1...P_N) are parameters of the method. The method stored in the function type is called with the same function-calling convention. Note that exception information is returned using the interface ErrorDetail. When the exception information has been returned, the C++ code of the method throws the exception on behalf of the method as shown in the 6th, 12th, and 22nd lines in **Figure 8**.

Figure 10 shows the implementation of the interfaces in **Figure 7**. The interfaces in **Figure 7** are inherited, and the methods are overridden in order to define the class Comp_1. The interfaces and methods for overriding are set using the macros BICOMC_OVERRIDE and BICOMC_OVER_METHOD. Parameters of the macro

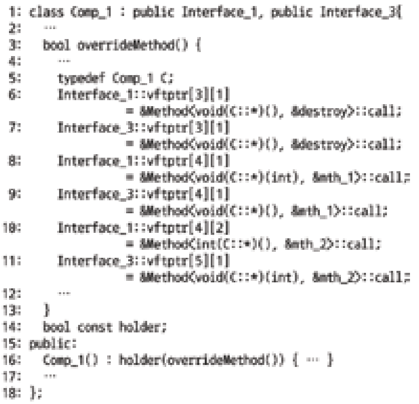
```
ErrorDetail* (*) (I*, R*, P_1, P_2, ..., P_N)
```

Figure 9.
Function type stored in the virtual function table.



```
 1: class Comp_1 : public Interface_1, public Interface_3{
 2:   BICOMC_OVERRIDE(Interface_1, Interface_3)
 3:     BICOMC_OVER_METHOD(destroy, void())
 4:     BICOMC_OVER_METHOD(mth_1, void())
 5:     BICOMC_OVER_METHOD(mth_1, void(int))
 6:     BICOMC_OVER_METHOD(mth_2, int())
 7:     BICOMC_OVER_METHOD(mth_2, int(int))
 8:   BICOMC_OVERRIDE_END()
 9: public:
10:   Comp_1() : BICOMC_OVERRIDE_INIT() { … }
11:   void destroy() { delete this; }
12:   void mth_1() { … }
13:   void mth_1(int p1) { … }
14:   int mth_2() { … }
15:   int mth_2(int p1) { … }
16: };
```

Figure 10.
Implementation of interfaces and overriding.



```
 1: class Comp_1 : public Interface_1, public Interface_3{
 2:   …
 3:   bool overrideMethod() {
 4:     …
 5:     typedef Comp_1 C;
 6:     Interface_1::vftptr[3][1]
                = &Method<void(C::*)(), &destroy>::call;
 7:     Interface_3::vftptr[3][1]
                = &Method<void(C::*)(), &destroy>::call;
 8:     Interface_1::vftptr[4][1]
                = &Method<void(C::*)(int), &mth_1>::call;
 9:     Interface_3::vftptr[4][1]
                = &Method<void(C::*)(), &mth_1>::call;
10:     Interface_1::vftptr[4][2]
                = &Method<int(C::*)(), &mth_2>::call;
11:     Interface_3::vftptr[5][1]
                = &Method<void(C::*)(int), &mth_2>::call;
12:     …
13:   }
14:   bool const holder;
15: public:
16:   Comp_1() : holder(overrideMethod()) { … }
17:   …
18: };
```

Figure 11.
C++ code of **Figure 10** *after preprocessing.*

BICOMC_OVERRIDE are names of all interfaces that the component inherits. Parameters of the macro BICOMC_OVER_METHOD are the names and signatures of the overridden methods. **Figure 11** shows C++ code converted from **Figure 10** by C++ preprocessor. The macro BICOMC_OVERRIDE is converted to bool overrideMethod(), and parameters of the macro such as Interface_1 and Interface_3 are used for clear conversion of overridden methods related to the type of the interface. The BICOMC_OVER_METHOD macros on lines 3–7 in **Figure 10** are converted to codes on lines 6–11 in **Figure 11** and the macro BiCOMC_OVERRIDE_INIT() to holder(overrideMethod()) on the 16th line in **Figure 11**.

4. Examples of BiCOMC-based application

4.1 A simple example

This section describes in sequence how to make a file copy application, which is a simple example based on BiCOMC. The file copy application is designed to run on Windows and Linux and is built using MSVC and GCC, respectively. The dynamic library providing the file copy function, which named utility, is created, and the executable file using this dynamic library, which named app, is generated as the application. The interface is first defined to share an object that provides a method for copying files. The interface to provide this functionality is defined in the utility.h file as the ICopy interface. **Figure 12** shows that the ICopy interface is defined based on the macro of BiCOMC and is the source code of the dynamic library named *utility.cpp*.

The BiCOMC macro is defined in *object.h*, which is added as shown on line 5 in **Figure 12** to use it. *ICopy* has a *copy()* method that receives two file names of *src* and *dst* as the input parameters and copies *src* to *dst*, which is shown on the eighth line in **Figure 12**.

In order to implement the *ICopy* interface in **Figure 12**, the *Copy* class is written made as shown in the *utility.cpp* of **Figure 13**. In addition, *copy()* method of *ICopy* is overridden in **Figure 13**. Note that the *utility.cpp* file is the source code for the *utility* dynamic library.

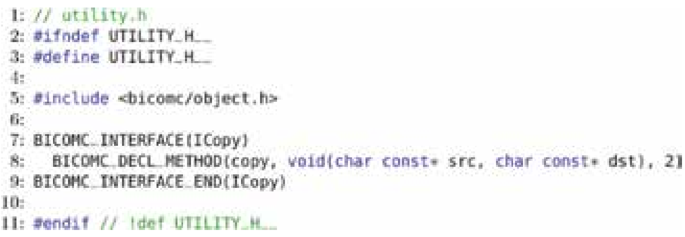
The *utility.h* which defines the *ICopy* interface is added as one of the header files on the second line in **Figure 13**, and some header files for the *Copy* class implementation on lines 4–13 are defined for Windows and Linux. The *Copy* class inherits the *ICopy* interface on the 15th line, and the overriding methods are explicitly specified using the BiCOMC macro on lines 16–19, where the specified methods are written on lines 23–37. The *create()* function on lines 40–46 creates an instance of the *Copy* class in order to pass its instance outside of the dynamic library called *utility*. The *main()* function of *app* using *utility* dynamic library is written in the *app.cpp* file, which is illustrated in **Figure 14**.

The *create()* function is called on the 21th line in **Figure 14** to get an instance of the *Copy* class in **Figure 13** but casted to *ICopy** using *bicomc_cast()* since *create()* returns *Object**. The *copy()* method is called to perform file copy on the 24th line. This method is used in try-catch because it generates an exception when *copy()* fails. The commands shown in **Figure 15** are executed to build *utility.cpp* into *utility* dynamic library and *app.cpp* into *app* executable file, respectively.

4.2 BiCOMC-based component for robot application

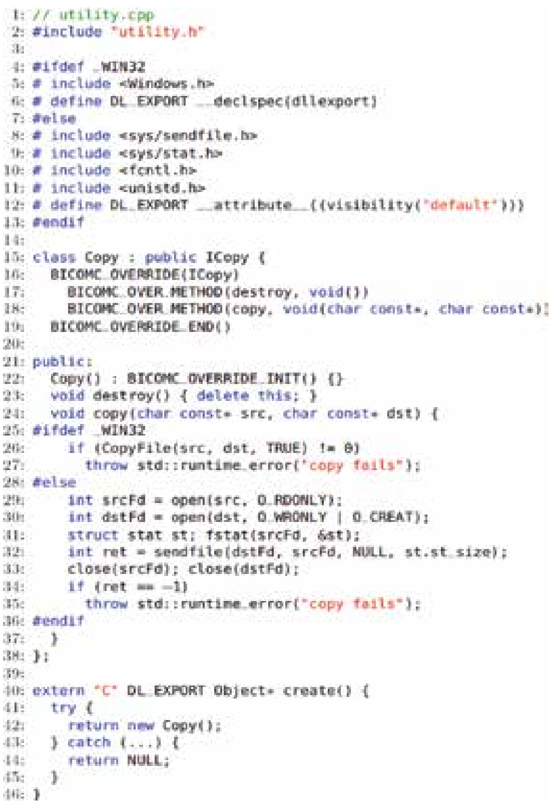
This section explains how to implement BiCOMC-based components using a robot applications. The robot application consists of three components in a Windows environment as follows: the *ManipulatorComp*, the *MobileComp*, and the *AppComp* in **Figure 16**.

The component *ManipulatorComp* is compiled with GCC 5.2 and controls the manipulator. The component *MobileComp* is compiled with MSVC 14 and controls the mobile platform. The component *AppComp* is compiled with MSVC 10 and coordinates the components *MobileComp* and *ManipulatorComp*. The component *AppComp* accesses the components *MobileComp* and *ManipulatorComp* through the interfaces *IMobile* and *IManipulator*, respectively. Definitions of these interfaces are illustrated in **Figure 17**. **Figure 18** shows the definition of the component *MobileComp* that inherits the interface *IMobile* and overrides the methods of the



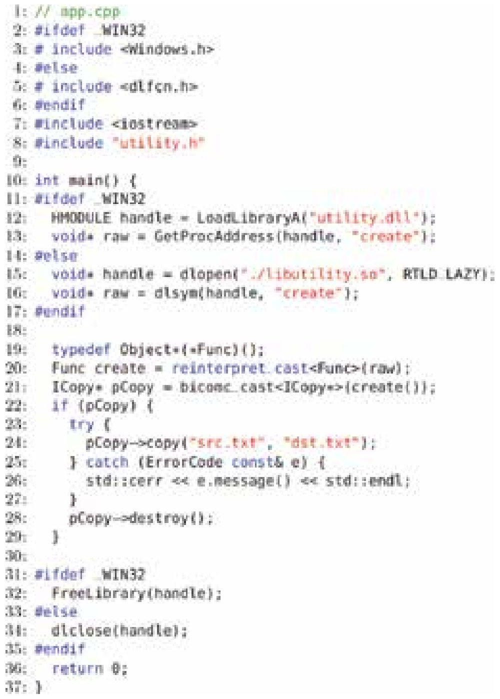
```
 1: // utility.h
 2: #ifndef UTILITY_H__
 3: #define UTILITY_H__
 4:
 5: #include <bicomc/object.h>
 6:
 7: BICOMC_INTERFACE(ICopy)
 8:   BICOMC_DECL_METHOD(copy, void(char const* src, char const* dst), 2)
 9: BICOMC_INTERFACE_END(ICopy)
10:
11: #endif // !def UTILITY_H__
```

Figure 12.
C++ code of utility.h.



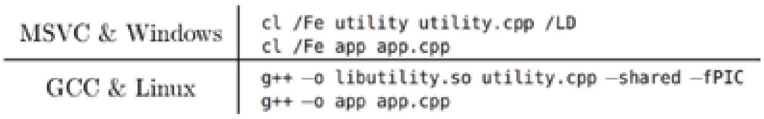
```cpp
// utility.cpp
#include "utility.h"

#ifdef _WIN32
# include <Windows.h>
# define DL_EXPORT __declspec(dllexport)
#else
# include <sys/sendfile.h>
# include <sys/stat.h>
# include <fcntl.h>
# include <unistd.h>
# define DL_EXPORT __attribute__((visibility("default")))
#endif

class Copy : public ICopy {
  BICOMC_OVERRIDE(ICopy)
    BICOMC_OVER_METHOD(destroy, void())
    BICOMC_OVER_METHOD(copy, void(char const*, char const*)
  BICOMC_OVERRIDE_END()

public:
  Copy() : BICOMC_OVERRIDE_INIT() {}
  void destroy() { delete this; }
  void copy(char const* src, char const* dst) {
#ifdef _WIN32
    if (CopyFile(src, dst, TRUE) != 0)
      throw std::runtime_error("copy fails");
#else
    int srcFd = open(src, O_RDONLY);
    int dstFd = open(dst, O_WRONLY | O_CREAT);
    struct stat st; fstat(srcFd, &st);
    int ret = sendfile(dstFd, srcFd, NULL, st.st_size);
    close(srcFd); close(dstFd);
    if (ret == -1)
      throw std::runtime_error("copy fails");
#endif
  }
};

extern "C" DL_EXPORT Object* create() {
  try {
    return new Copy();
  } catch (...) {
    return NULL;
  }
}
```

Figure 13.
C++ code of utility.h.



```cpp
// app.cpp
#ifdef _WIN32
# include <Windows.h>
#else
# include <dlfcn.h>
#endif
#include <iostream>
#include "utility.h"

int main() {
#ifdef _WIN32
  HMODULE handle = LoadLibraryA("utility.dll");
  void* raw = GetProcAddress(handle, "create");
#else
  void* handle = dlopen("./libutility.so", RTLD_LAZY);
  void* raw = dlsym(handle, "create");
#endif

  typedef Object*(*Func)();
  Func create = reinterpret_cast<Func>(raw);
  ICopy* pCopy = bicomc_cast<ICopy*>(create());
  if (pCopy) {
    try {
      pCopy->copy("src.txt", "dst.txt");
    } catch (ErrorCode const& e) {
      std::cerr << e.message() << std::endl;
    }
    pCopy->destroy();
  }

#ifdef _WIN32
  FreeLibrary(handle);
#else
  dlclose(handle);
#endif
  return 0;
}
```

Figure 14.
C++ code of app.cpp.



MSVC & Windows	`cl /Fe utility utility.cpp /LD`
	`cl /Fe app app.cpp`

GCC & Linux	`g++ -o libutility.so utility.cpp -shared -fPIC`
	`g++ -o app app.cpp`

Figure 15.
Compilation commands of app.cpp and utility.cpp.

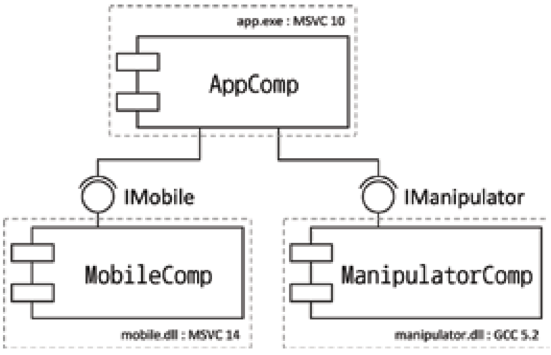
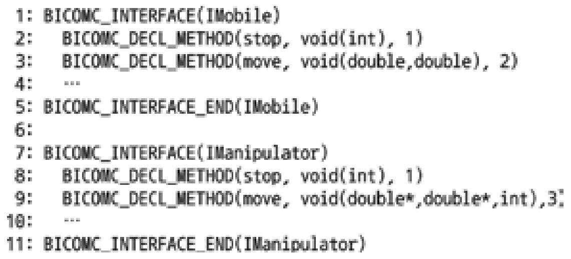
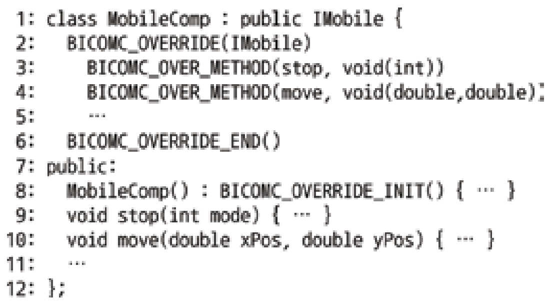


Figure 16.
Configuration example of components for a robot application.



```
BICOMC_INTERFACE(IMobile)
  BICOMC_DECL_METHOD(stop, void(int), 1)
  BICOMC_DECL_METHOD(move, void(double,double), 2)
  ···
BICOMC_INTERFACE_END(IMobile)

BICOMC_INTERFACE(IManipulator)
  BICOMC_DECL_METHOD(stop, void(int), 1)
  BICOMC_DECL_METHOD(move, void(double*,double*,int),3
  ···
BICOMC_INTERFACE_END(IManipulator)
```

Figure 17.
Definition of interfaces IMobile and IManipulator.

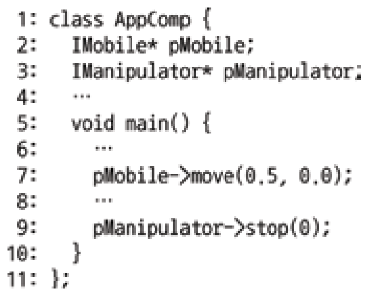


```
class MobileComp : public IMobile {
  BICOMC_OVERRIDE(IMobile)
    BICOMC_OVER_METHOD(stop, void(int))
    BICOMC_OVER_METHOD(move, void(double,double)
    ···
  BICOMC_OVERRIDE_END()
public:
  MobileComp() : BICOMC_OVERRIDE_INIT() { ··· }
  void stop(int mode) { ··· }
  void move(double xPos, double yPos) { ··· }
  ···
};
```

Figure 18.
Definition of component MobileComp.

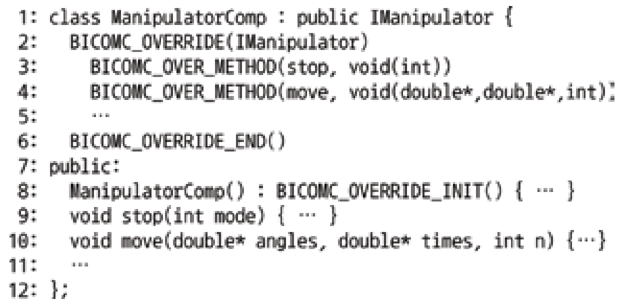
interface. The component *ManipulatorComp* and the component *AppComp* can be defined in the similar manner, which is shown in **Figures 19** and **20**, respectively.

Figure 21 shows the operation results of the robot after three components in **Figure 16** are successfully implemented, which are parts captured from a video clip [15]. It can be observed from **Figure 21** that the BiCOMC-based components function properly regardless of the types of compilers.



```
class AppComp {
  IMobile* pMobile;
  IManipulator* pManipulator;
  ...
  void main() {
    ...
    pMobile->move(0.5, 0.0);
    ...
    pManipulator->stop(0);
  }
};
```

Figure 19.
Example of component AppComp.



```
class ManipulatorComp : public IManipulator {
  BICOMC_OVERRIDE(IManipulator)
    BICOMC_OVER_METHOD(stop, void(int))
    BICOMC_OVER_METHOD(move, void(double*,double*,int));
    ...
  BICOMC_OVERRIDE_END()
public:
  ManipulatorComp() : BICOMC_OVERRIDE_INIT() { … }
  void stop(int mode) { … }
  void move(double* angles, double* times, int n) {…}
  ...
};
```

Figure 20.
Definition of component ManipulatorComp.

Figure 21.
Video capture of robot application running.

5. Evaluation

This section evaluates the binary compatibility occurred among different types of compilers and verifies whether backward compatibility can be maintained when the interface version has been changed. There are some binary compatibility checking tools like ABI Compliance Checker [16], shlib-compat [17], libabigail [18], and ABICheck [19], but these tools are used to check API/ABI in C language level or to compare virtual function tables generated by the supported compiler. So they cannot check the binary compatibility of C++ objects created by different types of compilers. In addition these tools cannot detect the compatibility maintained by BiCOMC. For checking of the binary compatibility, this chapter uses the proposed methods explained in Sections 5.1 and 5.2. Finally the call times as performance measures are compared among BiCOMC, COM, and CCC using different types of compilers.

5.1 Evaluation of binary compatibility between compilers

Compilers generally reorder a virtual function table according to each compiler's ABI, which makes the binary compatibility difficult. The interface *CompatibilityChecker* is suggested to check whether the methods of objects shared by different binaries (or executable files) created by different types of compilers normally call each other. The interface *CompatibilityChecker* has the methods *check1()*, *check2()*, and *check3()* arranged in an unordered fashion and is illustrated in **Figure 22**. The methods simply return the values 1, 2, and 3, respectively. In addition, all methods' calling convention is controlled as the same.

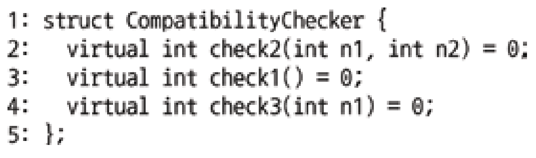
These methods are called to check whether the normal return value is delivered. The tests are considered successful if the three methods are normally called and are judged to have failed if any of the methods are not normally called or the program has shut down. The experiments using the interface *CompatibilityChecker* are performed using MSVC 9 and 14, GCC 4.5 and 5.2, and the Intel C++ Compiler 14 and 16 in Windows 10 in order to verify binary compatibility among different types of compilers. Note that the GCC of MinGW-w64 is used. In **Tables 1–3**, "O" means pass, "X" means failure, and "-" means not testable. ICC is the abbreviation of the Intel C++ Compiler. In other words, "O" means that the three methods are normally called.

It can be seen in **Tables 1** and **2** that BiCOMC and CCC guarantee the binary compatibility among MSVC, GCC, and ICC. However, CCC is only available in compilers supporting C++ 11. **Table 3** shows that the binary compatibility between MSVC and ICC is guaranteed as stated in [20], but these two compilers are not compatible with GCC. As stated earlier, the different types of compilers reorder the virtual function table, and then the reordered methods are not called normally. So C++ methods are called abnormally by different types of compilers as shown in **Table 3**, whereas BiCOMC and CCC worked normally because they prevent to reorder a virtual function table.

5.2 Evaluation of backward compatibility of different interface versions

In developing the program, it is necessary to add a new method to the existing interface. That is, the interface version is changed in this case, which is shown in **Figure 23**.

Figure 23 shows two interfaces of *IfaceA* and *IfaceB*, where *IfaceB* inherits *IfaceA*. In the version *v1*, *IfaceA* has a method, *mth_1()*. But the interface *IfaceA* should be upgraded to the new version *v2* because a new method of *mth_3()* is added. Let us consider the following: the v1-related interfaces and the v2-related interfaces are used in the caller and the callee programs, respectively.



```
struct CompatibilityChecker {
  virtual int check2(int n1, int n2) = 0;
  virtual int check1() = 0;
  virtual int check3(int n1) = 0;
};
```

Figure 22.
Interface for binary compatibility test.

CCC			Dynamic library (DLL)					
			MSVC		GCC		ICC	
			9	14	4.5	5.2	14	16
Executable (EXE)	MS VC	9	—	—	—	—	—	—
		14	—	O	—	O	—	O
	GCC	4.5	—	—	—	—	—	—
		5.2	—	O	—	O	—	O
	ICC	14	—	—	—	—	—	—
		16	—	O	—	O	—	O

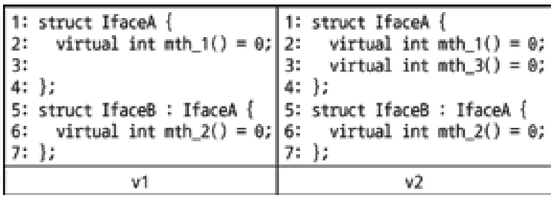
Table 2.
Tests of the binary compatibility in CCC.

CCC			Dynamic library (DLL)					
			MSVC		GCC		ICC	
			9	14	4.5	5.2	14	16
Executable (EXE)	MS VC	9	O	O	X	X	O	O
		14	O	O	X	X	O	O
	GCC	4.5	X	X	O	O	X	X
		5.2	X	X	O	O	X	X
	ICC	14	O	O	X	X	O	O
		16	O	O	X	X	O	O

Table 3.
Tests of the binary compatibility in C++.

This test also uses simple methods that return values of 1, 2, and 3, respectively, which are similar to methods used in Section 5.1. The experiments are done using MSVC 14, GCC 5.2, and ICC 16 in Windows 10.

Table 4 shows that BiCOMC can enable the binary compatibility between interfaces of versions *v1* and *v2*, but CCC, COM, and C++ are not compatible between codes with different interface versions. Therefore, BiCOMC supports the backward compatibility of interface versions. MSVC, GCC, and ICC generate a virtual function table in contiguous memory space regardless of inheritance. In this structure of the table, if a new method is added in the parent interface, the offset of child's methods will be changed. For this reason, C++ does not provide the backward compatibility. COM and CCC also have the same reason. BiCOMC has the structure



```
struct IfaceA {
   virtual int mth_1() = 0;

};
struct IfaceB : IfaceA {
   virtual int mth_2() = 0;
};
```

v1

```
struct IfaceA {
   virtual int mth_1() = 0;
   virtual int mth_3() = 0;
};
struct IfaceB : IfaceA {
   virtual int mth_2() = 0;
};
```

v2

Figure 23.
Interface for backward compatibility test per version.

Method compiler	BiCOMC	CCC	COM	C++
MSVC 14	O	X	X	X
GCC 5.2	O	X	—	X
ICC 16	O	X	X	X

Table 4.
Tests of the backward compatibility.

of **Figure 2** so that the parent's methods and the child's methods can be stored at independent memory space of each other. Thus BiCOMC supports the backward compatibility of interface versions.

5.3 Call time evaluation

Call times as performance measures are measured in Windows 10 using MSVC 14, GCC 5.2, and ICC 16. The methods of objects are called 10 million times, and the call times are obtained as the average values of all call times. These are shown in **Figures 24–26**, in which MSVC, GCC, and ICC compilers are used for evaluation, respectively. In these figures, two notations such as *XXX* and *XXX-R* are used, where *XXX* is one of BiCOMC, CCC, CPP (or C++), and COM and *-R* means that the method used in the test has a return value.

Tables 1–4 show that BiCOMC provides the best binary compatibility among different types of compilers. **Figures 24–26** show that C++ has the best call time, but the call time of BiCOMC is similar to those of C++ and COM. CPP is generally faster than others because the method calling accesses an optimized virtual function table which accesses directly addresses of methods. COM is also similar to CPP. Note that GCC does not support COM. CCC is slower as std::function, one of the C++ 11 features [14], is basically used [12]. BiCOMC is slower slightly than CPP and COM because it accesses addresses of methods in the *function table* pointed by the *virtual function table*. But the test results show no or little significant difference among BiCOMC and CPP/COM.

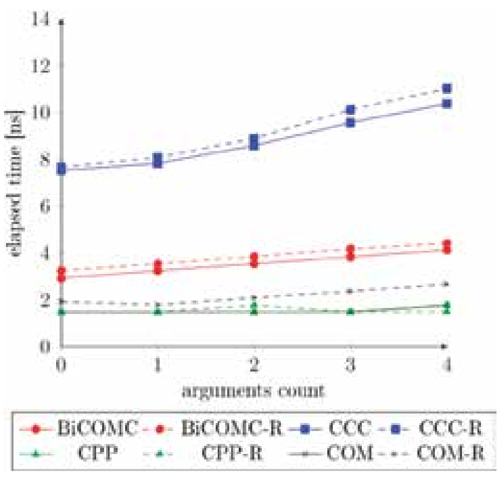


Figure 24.
Call time in MSVC.

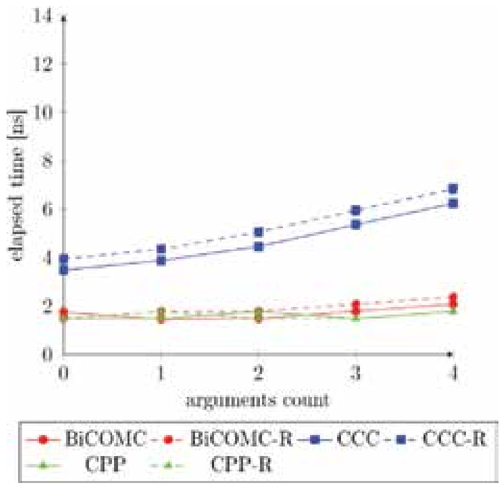


Figure 25.
Call time in GCC.

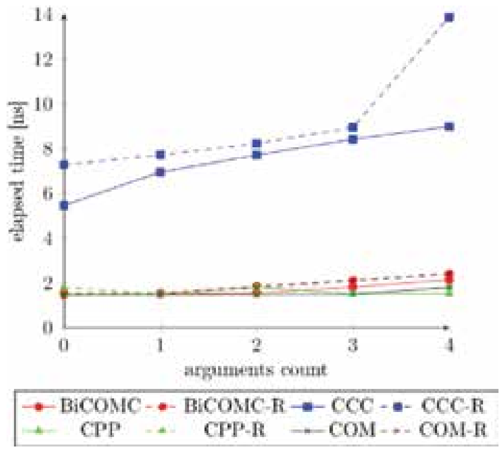


Figure 26.
Call time in ICC.

6. Conclusion

This chapter proposed the binary compatibility object model for C++ (BiCOMC) to provide the binary compatibility of objects necessary for reusability of software components in the Windows and Linux environment in order to share objects among C++ based executable files such as .exe, .dll, and .so. The interfaces for the component, method overloading and overriding, multiple inheritance, and the exception handling were suggested based on BiCOMC model.

The proposed model was validated by application examples and comparisons with commonly known object models such as C++, COM, and CCC in terms of the call time of a method during execution and the binary compatibility such as reusability. The application examples showed that components compiled by GCC and MSVC call each other without any restrictions. From **Tables 1–3**, it can be seen that the BiCOMC provides better binary compatibility in a Windows environment than object models in C++, COM, and CCC, which are compiled in GCC, MSVC, and ICC. The BiCOMC was compared with C++, COM, and CCC in terms of the call times of methods during run time. The results showed that the call time of the BiCOMC was similar to C++/COM. In other words, the application examples and the evaluation results verified that the proposed method was provided for the binary compatibility among different types of compilers.

In future we will develop and distribute BiCOMC-based components for various applications such as industrial/medical robot applications and factory/home automation application, which can be used regardless of the types of compilers.

Author details

Donguk Yu and Hong Seong Park*
Department of Electrical and Electronics Engineering, Kangwon National University, Chuncheon, Republic of Korea

*Address all correspondence to: hspark@kangwon.ac.kr

References

[1] Han S, Kim M, Park HS. Open software platform for robotic services. IEEE Transactions on Automation Science and Engineering;**9**(3):467-481

[2] Jang C et al. OPRoS: A new component-based robot software platform. ETRI Journal;**32**(5):646-656

[3] OPRoS Site [Online]. Available from: http://ropros.org [Accessed: 03 January 2018]

[4] Ando N et al. Software deployment infrastructure for component based RT-systems. Journal of Robotics and Mechatronics;**23**(3):350-359

[5] OpenRTM Site [Online]. Available from: http://www.openrtm.org [Accessed: 03 January 2018]

[6] OROCOS Site [Online]. Available from: http://www.orocos.org [Accessed: 04 January 2018]

[7] Gherardi L, Brugali D, Comotti D. A Java Vs. c++ performance evaluation: A 3d modeling benchmark. In: Proceedings of International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR 2012); 5-8 November 2012; Tsukuba. Berlin, Heidelberg: Springer; 2012. p. 161-172. DOI: 10.1007/978-3-642-34327-8

[8] Hildstrom G. Programming Language Performance Comparison [Online]. Available from: http://hildstrom.com/projects/langcomp/index.html [Accessed: 04 January 2018]

[9] Sayfan G. Building Your Own Plugin Framework [Online]. Available from: http://www.drdobbs.com/cpp/building-your-own-plugin-framework-part/204202899 [Accessed: 04 January 2018]

[10] Goldstein TC and Sloane AD. The object binary interface—C objects for evolvable shared class libraries. In: Proceedings of the 6th Conference on USENIX Sixth C++ Technical Conference (CTEC'94). Vol. 6; 11-14 April 1994; Cambridge, MA

[11] COM: Component Object Model Technologies [Online]. Available from: https://msdn.microsoft.com/en-us/library/windows/desktop/ms680573(v=vs.85).aspx [Accessed: 04 January 2018]

[12] Bandela JR. Cross Compiler Call [Online]. Available from: https://github.com/jbandela/cross_compiler_call [Accessed: 04 January 2018]

[13] Atkinson K. ABI compatibility through a customizable language. In: Proceedings of the Ninth International Conference on Generative Programming and Component Engineering (GPCE’10); 10-13 October 2010; Eindhoven. The Netherlands. DOI: 10.1145/1868294.1868316

[14] ISO/IEC 14882:2011 Information Technology—Programming Languages—C++, ISO/IEC JTC 1/SC 22

[15] Yu D. Video of Robot Application Running [Online]. Available from: https://www.youtube.com/watch?v=gMMsNINp14g [Accessed: 04 January 2018]

[16] ABI Compliance Checker [Online]. Available from: https://lvc.github.io/abi-compliance-checker [Accessed: 05 January 2018]

[17] Kurtsou G. Shlib-Compat: ABI Compatibility Checker for Shared Libraries with Symbol Versioning [Online]. Available from: https://github.com/glk/shlib-compat [Accessed: 04 January 2018]

[18] The ABI Generic Analysis and Instrumentation Library [Online].

Available from: https://sourceware.org/libabigail [Accessed: 05 January 2018]

[19] Abicheck [Online]. Available from: http://abicheck.sourceforge.net [Accessed: 06 January 2018]

[20] Intel C++ Compiler Compatibility with Microsoft Visual C++ [Online]. Available from: https://software.intel.com/en-us/articles/intel-c-compiler-compatibility-with-microsoft-visual-c [Accessed: 06 January 2018]

Edited by Lulu Wang

This book aims to provide a brief update to the current status of and advances in computational methods and programs used for the development of the theory and practice of biomedical signal and image communication. The book comprises a collection of invited manuscripts, written in a convenient way and of manageable length. These timely collections will provide an invaluable resource for initial inquiries into technologies and will encapsulate the latest developments and applications with reference sources for further detailed information. The methods described in this book cover a wide range of computational algorithms that are widely used in bioengineering and biomedicine. The content and format are specifically designed to stimulate the further development and application of these technologies by reaching out to non-specialists across a broad audience. This book is intended to expose the latest developments of scientists and engineers covering a variety of complementary topics, to enhance people's overall understanding of computer science and biomedical image communications. It will benefit students, scientists, and researchers in applied computer science. Engineers and clinicians working in imaging will also find this book useful.

Published in London, UK

IntechOpen

ISBN 978-1-83880-246-2