Quality Examination of Clinical DXA Images for Assessment of Hip Fracture Risk by Finite Element Modeling

By

Wanshi Yu

A thesis submitted to
the Faculty of Graduate Studies
in partial fulfilment of
the requirements for the degree of
Master of Science

Department of Mechanical Engineering
Faculty of Graduate Studies
University of Manitoba
Winnipeg, Manitoba

September 2019

Abstract

A DXA (Dual Energy X-ray Absorptiometry) based finite element model was previously developed and automated for clinical assessment of hip fracture risk caused by sideways fall. However, one remaining issue for the finite element model was that the accuracy of assessment is dependent on the quality of DXA images.

Requirements on DXA images for finite element analysis are much more stringent than those for clinical densitometry examination. Therefore, the objective of this study was to apply and implement algorithms to automatically examine clinical DXA images before finite element analysis. In this study, two algorithms were implemented to work in series for DXA quality examination, they are the *k*-NN algorithm and average distance between neighboring points on femur contour. Geometric features of common defects in clinical images were identified and quantified. After that, these features were used for training the algorithms. Clinical DXA images were acquired from St. Boniface General Hospital (SBGH) to validate the algorithms. A standalone computer program was wrapped using MATLAB packages.

The computer program was tested by SBGH and the assessment accuracy of hip fracture risk was substantially improved. However, a number of minor defects have been identified in the clinical testing, which will be implemented in a new version of the computer program.

Acknowledgments

I would like to express my gratitude to all those who helped me during the writing of this thesis.

My deepest gratitude goes first and foremost to my advisor, Dr. Yunhua Luo, for his constant guidance and encouragement. He has supported me and helped me during all the stages of my graduate study and guided me through my research. Without his consistent instruction and kind encouragement, this thesis could not have been completed.

Secondly, I would like to express my heartfelt gratitude to all of my friends in University of Manitoba for their help during my study and research.

Lastly, my thanks would go to my dear family for their love and support to my studying for these years.

In addition, please allow me to express my gratitude to all the professors who reviewed this thesis in their busy schedule.

Dedication

To my parents.

Contents

Front Matter

List of Tables

List of Figures

List of Abbreviations

2D	Two-dimensional
3D	Three-dimensional
3D-FEXI	3D FEA of X-ray Images
aBMD	Areal Bone Mineral Density
AJAX	Asynchronous JavaScript And XML
AUC	Area Under the Receiver-operating-characteristics Curve
BASH	Bourne Again Shell
BMC	Bone Mineral Content
BMD	Bone Mineral Density
CGI	Common Gateway Interface
CSS	Cascading Style Sheets
DOES	Dubbo Osteoporosis Epidemiology Study
DXA	Dual Energy X-ray Absorptiometry
ELF	Executable and Linkable Format
FEA	Finite Element Analysis
FRAX	Fracture Risk Assessment Tool
FRIs	Fracture Risk Indices
GUI	Graphical User Interface

HFRIs	Hip Fracture Risk Indices
HSA	Hip Structure Analysis
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
k-NN	*K*-nearest Neighbors
LAN	Local Area Network
LSR	Load-to-strength Ratio
MATLAB	Matrix Laboratory
NCSA	National Center for Supercomputing Applications
NOF	National Osteoporosis Foundation
OST	Osteoporosis Self Assessment Tool
PHP	Hypertext Preprocessor
PPV	Positive Predictive Value
QCT	Quantitative Computed Tomography
ROC	Receiver Operating Characteristics
ROIs	Regions of Interest
SMB	Server Messages Block
SWAN	Study of Women's Health Across the Nation
UI	User Interface
WHO	World Health Organization
XHTML	Extensible Hypertext Markup Language

Chapter 1

Introduction

1.1 Motivation and background

Osteoporosis is the most common bone disease and a significant concern for elderly people. It is a disease characterized by a reduced bone mineral density (BMD) and deterioration of bone tissue, which commonly occurs in people of 50 and older. Nowadays, the increase in the proportion of the aging population gives rise to an enormous increase in the number of osteoporosis patients [1]. If osteoporosis cannot be accurately diagnosed and effectively treated in time, the damage in bone structure can lead to fracture under some circumstances. In fact, statistical studies have shown that fractures caused by osteoporosis are more common than heart attack, breast cancer, and stroke combined, and over 80% of fractures in seniors (50+) are induced by osteoporosis [2]. Studies have indicated that healthcare costs of osteoporosis and fracture have been increasing rapidly mainly due to inadequate diagnosis technology [1]. According to the information from Osteoporosis Canada, the total yearly cost of treating osteoporosis and related fractures for Canadian healthcare system was over $2.3 billion in 2010 [2].

One of the most typical sites for fractures induced by osteoporosis is the hip [3]. Annually, 70-90% of 30,000 hip fractures in Canada are caused by osteoporosis, which uses more healthcare resources and results in higher mortality rate than stroke, diabetes or heart attack [2]. Based on age- and sex-specific incidence of osteoporosis since 1990, the number of hip fractures is expected to reach 2.6 million by the year 2025, and 4.5 million by 2050 [4]. Therefore, effective diagnosis and treatment of osteoporosis is essential to reduce healthcare costs and to maintain life quality of osteoporosis patients. To meet the challenge, many researchers have devoted their effort to develop techniques that can help reduce osteoporotic hip fractures.

To decrease the number of osteoporotic hip fractures, accurate, safe and cost-effective assessment of fracture risk is necessary. Strategically, it is always cost effective to identify high risk patients and to provide them effective treatment before fracture happens.

Currently, the most commonly used method for diagnosing osteoporosis and fracture risk is the areal bone mineral density (aBMD) measured by dual energy X-ray absorptiometry (DXA) [5]. However, its ability of aBMD in assessment of bone strength is limited due to the involvement of other risk factors such as bone geometry and impact force induced in fall [6, 7]. Various studies have suggested that aBMD alone cannot determine bone strength and patient-specific fracture risk [8, 9]. To resolve this problem, researchers have proposed alternative methods to consider more risk factors affecting fracture risk, for example Fracture Risk Assessment Tool (FRAX) [10] and hip structural analysis (HSA) [11]. However, FRAX does not consider patient-specific impact force, which makes it only useful for statistical analysis instead of patient-specific fracture risk assessment [12]. Additionally, the results using HSA were not more accurate compared with aBMD only [11, 12]. Although researchers have considered combining aBMD with clinical risk factors and bone geometry information, the results from those studies do not show any considerable improvement compared with aBMD alone [12, 10, 13].

For the above reasons, finite element modeling based on medical images has come to the attention of biomedical researchers. The developed finite element models can be divided into three-dimensional (3D) and two-dimensional (2D) ones. It has been shown from a number of studies that finite element modeling is a promising method for the assessment of patient-specific fracture risk [6].

3D or 2D medical imaging-based finite element models can be constructed from quantitative computed tomography (QCT) or DXA images. QCT-based 3D finite element modeling has a higher accuracy in the prediction of fracture, but it also requires higher cost and high exposure to radiation [16]. Compared with 3D finite element modeling, 2D finite element modeling based on DXA images has a relatively lower accuracy with much lower cost and radiation level [15, 16]. Researchers have made significant progress using DXA-based finite element models to predict femoral strength [16]. It has also been reported that using parameters derived from DXA images can have better performance than using BMD alone [19]. Therefore, DXA-based finite element modeling has become the most promising assessment method for hip fracture risk [20].

In a previous study conducted by our research group, a subject-specific DXA-based finite element tool for the assessment of hip fracture risk was developed with Matrix Laboratory (MATLAB) [20]. Different from other DXA-based finite element tools, the segmentation of the femur outline and the detection of critical femur cross-sections are fully automated in this model. A finite element mesh is generated from the contour coordinates. Therefore, the segmentation quality of proximal femur contour has an important impact on the analysis accuracy. The DXA-based finite element model has improved prediction of osteoporotic hip fracture and monitoring of treatment effectiveness. If qualified DXA images are provided, it can be considered as an automatic assessment of hip fracture risks using DXA images.

However, the computer program based on the model can be interrupted by DXA images with defects, which typically have poor quality of the segmented contour. In other words, this model cannot check and exclude DXA images of poor quality, and therefore cannot be applied to large-scale clinical assessment. The precision of hip fracture risk analysis in this model is also largely affected by DXA quality [21] that is dependent on the quality of segmented contour and consistence of patient positioning [20]. Particularly, the quality of segmented contour is critical, since it determines the quality of finite element mesh. Thus, clinical DXA images must be inspected prior to finite element modeling to achieve a fully automated operation and a high precision. Previously, the inspection of DXA images has been completed manually, which requires much time and effort. In order to reduce the related economic and human costs, an automatic algorithm is necessary for detecting the DXA images with poor contour quality and removing them before FEA.

Another problem of this DXA-based finite element model is the deficiency in the redistribution. Since it was fully developed with in-house MATLAB code, this model requires the installation of MATLAB and sufficient knowledge in MATLAB programming, hence, unsuitable for clinical application. As a result, an automatic and standalone tool needs to be developed based on this model to better fulfill the requirements of clinical hip fracture risk assessment.

In conclusion, developing an automatic algorithm for the quality examination of clinical DXA images will be the main goal of this study. The improved model, including the quality examination of DXA images and the finite element modeling, will be then redistributed for a more general clinical application.

1.2 Research objectives

With the DXA-based finite element model developed by our research group, the primary goal of this thesis is to provide an automatic algorithm for the quality examination of DXA images. This thesis will present two algorithms for identifying major defects in DXA images and excluding unqualified ones from the following FEA procedure, in order to achieve a fully automated operation. Then, a standalone computer program for the routine clinical assessment will be developed based on the quality examination algorithms of DXA images and finite element modeling. It will enable clinicians or physicians to conduct the assessment without MATLAB software or any programming knowledge. Manual manipulation in the assessment will be reduced to a minimal amount.

As a supplementary work in this thesis and a direction of future studies, the framework of a web-based biomechanical tool for hip fracture risk assessment will also be proposed.

With the work completed in this thesis, a large-scale clinical assessment of osteoporotic hip fracture risks will be much more convenient and efficient. Patients with high hip fracture risk can be easily identified and selected for treatment.

1.3 Proposed approach

For quality examination of DXA images, two algorithms will be implemented in MATLAB based on neighboring pixels or points on femur contours. The maximum area of pixel patches and the distance between neighboring points will be the features, respectively. In addition, *k*-NN algorithm will be used to identify the feature in the first algorithm. After that, these quality examination algorithms and the existing DXA-based finite element model will be packaged into a standalone program using MATLAB Compiler. A user-friendly interface will be designed for this program.

The framework of a web-based biomechanical tool will be developed in the Linux system based on Hypertext Markup Language (HTML) and PHP. A Linux Shell script will be used for setting up the environmental variables and the transmission of parameters. The in-house MATLAB codes will be packaged into an executable file that can be invoked by other programming languages. When the operation of the Linux Shell script is completed, users can have access to the assessment results through the Samba client.

1.4 Thesis organization

This thesis is arranged as follows:

Chapter 2 - In this chapter, literature related to the research topics covered in this study will be reviewed. The DXA-based finite element modeling for hip fracture risk assessment will be introduced, with detailed information of the finite element model used in this study. After that, there will be some discussion on the quality examination algorithms of DXA images. Moreover, the necessity of developing a standalone and automatic computer program for clinical assessment of hip fracture risks will be justified.

Chapter 3 - In this chapter, quality criteria of DXA images will be first introduced. Two algorithms for quality examination of DXA images will be then presented. The algorithms, their pros and cons will be explained in detail.

Chapter 4 - In this chapter, a standalone computer program will be developed using MATLAB GUIDE and MATLAB Compiler for clinical assessment of hip fracture risk. The DXA-based finite element model and the quality examination algorithms of DXA images proposed in this study will be implemented in this program.

Chapter 5 - In this chapter, the framework of a web-based biomechanical tool will be designed, which can be used by clinicians via internet for hip fracture risk assessment. It should be noted that the framework developed in this study is only a preliminary one and cannot be applied to any clinical uses.

Chapter 6 - In this chapter, the quality examination algorithms of DXA images developed in Chapter 3 and the standalone program constructed in Chapter 4 will be validated. The performance of the proposed quality examination algorithms will be discussed and compared. Furthermore, the advantages and disadvantages of using the standalone program and the web-based biomechanical tool will be illustrated.

Chapter 7 - In this chapter, conclusions, contributions, and limitations of this study will be presented. Directions of future research are proposed to resolve these limitations.

Chapter 2

Literature reviewing of biomechanical tools for hip fracture risk assessment

Based on a previous study of our reaserch group [20], I will improve the DXA-based finite element modeling procedure as shown in Figure 2-1, where the improvements in this study are marked in red. This literature review will include four relevant aspects.

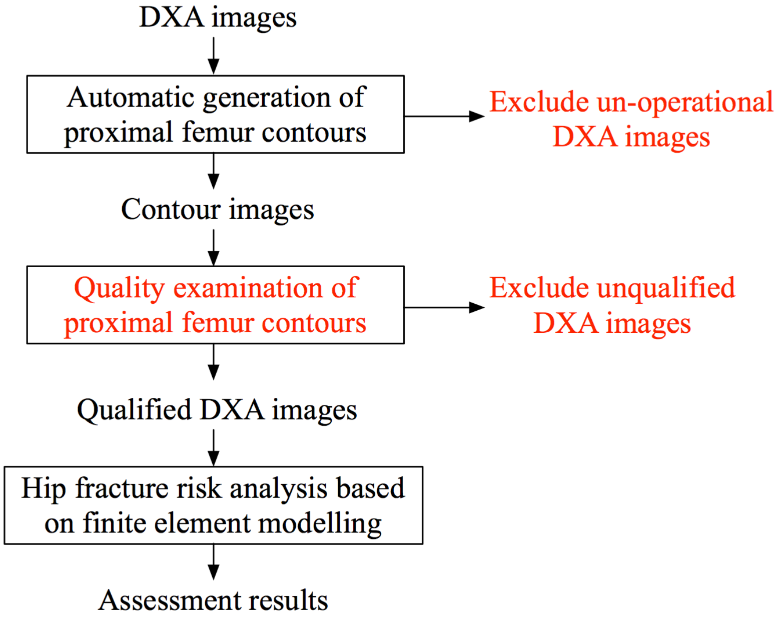

DXA images
Automatic generation of proximal femur contours
Exclude un-operational DXA images
Contour images
Quality examination of proximal femur contours
Exclude unqualified DXA images
Qualified DXA images
Hip fracture risk analysis based on finite element modelling
Assessment results

Figure 2-1: The procedure of the improved DXA-based finite element model

2.1 DXA-based finite element models for hip fracture risk assessment

For the accurate assessment of osteoporotic hip fracture risk, a number of studies have been conducted to develop clinically-applicable finite element models. This section of the literature review will focus on QCT and DXA based finite element models.

To assess hip fracture risks by FEA, 3D and 2D finite element models can be developed from QCT and DXA images respectively. In fact, compared with 2D FEA, 3D FEA did not show any evident increase in the sensitivity to changes in femoral density or geometry [22]. The study of J. Thevenot et al. [23] demonstrated that the 2D finite element model was able to discriminate hip fractures similar to complex 3D models. In addition, the cost and radiation level of DXA are much lower than those of QCT. For the above reasons, 2D finite element modeling based on DXA images is regarded as a practical approach for clinical assessment of hip fracture risk.

The first 2D finite element model was developed by Testi et al. for the estimation and validation of proximal femur strength using a synthetic fiber-glass femur [6]. A 2D finite element model of the proximal femur was built based on DXA data to predict *in vitro* hip fracture risk. The model was then validated *in vitro* against strain-gauge measurement and a 3D finite element model. Good agreements were observed. However, it should be mentioned that the value of Young's modulus was determined based on areal BMD, which is a major limitation of their model. Additionally, the magnitude and direction of the loads were assumed to be constant instead of patient-specific. Despite of these limitations, their study proposed an effective method for clinical evaluation of femoral neck fractures in osteoporotic patients with non-invasive and low-cost screening technique. Testi et al. [24] also found that the identification accuracy of hip fracture patients could be remarkably improved by adding one biomedical index, which was defined by the maximum principal tensile strain obtained from FEA and increased the prediction accuracy from 77% to 82%. To validate Testi et al.'s model [6] with experimental data, Buijs et al. developed a subject-specific 2D finite element model for assessing femoral stiffness and strength [25]. The capability of 2D DXA-based model to predict femoral fracture loads during sideways falls was validated by digital image correlation analysis [25].

In the study by Langton et al. [16], a 3D finite element model was generated from 2D radiographic images, which were projected images of QCT scans. It was demonstrated that their model could predict the failure load of the proximal femur with accuracy similar to that of 3D FEA. The approaches proposed in their study can be applied to regular clinical DXA images. However, the sample size in their research for validation of 3D FEA of X-ray images (3D-FEXI) was quite small.

Both studies by Buijs et al. [25] and Langton et al. [16] indicated the importance and possibility of developing finite element models from DXA images for hip fracture risk assessment. However, DXA images used in their studies were generated from CT scans, which could lead to some inevitable problems. For instance, QCT-based FEA is hard to implement in a clinical environment since it requires expensive computational devices. The clinical performance of QCT-based finite element models still needs to be further evaluated [17]. Additionally, CT scanning is not regularly used in the clinical assessment of osteoporosis owing to its high cost and high radiation dose [16]. Therefore, an optimal method to solve these problems is the use of DXA images instead of CT scans to construct a 2D finite element model for fracture risk assessment.

To compare the effectiveness of DXA and CT based models, M. Danielson et al. used 2D analysis based on DXA data and more sophisticated 3D analysis based on QCT data to estimate the strength of the femoral neck in post-menopausal women [26]. The femoral neck geometry parameters calculated with Hip Structure Analysis (HSA) from DXA data and those derived from QCT data were well correlated in a large Study of Women's Health Across the Nation (SWAN) data set. However, the estimated proximal femur strengths did not correlate well. After comparing different indices for DXA- and CT- based approaches, they observed significant correlations between the femoral neck geometries derived from DXA and QCT. In conclusion, DXA-based models are more applicable to clinical uses due to the wide availability of DXA scanners and lower radiation level of DXA scan.

To determine the relationship between bone strength derived from a finite element model and hip fracture risk, K. E. Naylor et al. applied load-to-strength ratio (LSR) [19]. Firstly, the finite element model was directly developed from a segmented BMD map, with a constant but subject-specific thickness. A sideways fall situation was then simulated to estimate the impact force. Finally, LSR was determined as the attenuating impact force divided by the femoral strength. They found that the baseline LSR was notably higher in fracture cases as compared with the control group. Therefore, patients with high hip fracture risk can be effectively identified using a DXA-based finite element model.

To get a more accurate patient-specific assessment of osteoporotic hip fracture risk and improve the treatment monitoring, Luo et al. constructed a finite element model of the human femur from patients' DXA images [27]. In this model, fracture risk indices at three critical locations (i.e., the femoral neck, the intertrochanteric region and the sub-trochanteric region) were introduced and used for fracture risk assessment. The modeling procedure was then validated with six representative clinical cases. Furthermore, the three risk indices introduced in this study had shown potential uses in identifying the fracture location. After that, Luo et al. improved the precision of their model to meet the clinical requirement [21]. By calculating hip fracture risk indices at 10mm wide regions of interest (ROIs), the repeatability of the finite element model based on DXA images was significantly improved. However, some random errors could not be removed, for example errors induced by inconsistent subject positioning.

To reduce the inconsistency in previous studies and further improve the model, an automated DXA-based finite element tool was developed for the clinical assessment of hip fracture risk [20]. The modeling procedures of this finite element tool is introduced in the following section.

2.2 Procedure of automated DXA-based finite element modeling for hip fracture risk assessment

In this section, the procedure of automatic DXA-based finite element modeling with inhouse MATLAB code [28] for hip fracture risk assessment will be introduced based on several previous studies conducted by our research group [15, 25, 26]. As mentioned before, this finite element model will be further improved in this study for a more general clinical application.

Figure 2-2 describes the procedure to assess hip fracture risk and discriminate hip fractures from controls using the developed DXA-based finite element model, which includes three steps—pre-processing, FEA and post-processing.

Before constructing a finite element model, the contour of proximal femur was automatically segmented from a DXA scan of the patient, critical femur cross sections and ROIs are detected (Examples of segmented proximal femur contours can be found in Chapter 3.2). After that, a finite element mesh consisting of triangular elements was generated from the proximal femur contour.

With the finite element mesh, material properties determined from the ash density of bones were assigned to the model at element Gaussian integration points. Before that, the volumetric BMD was calculated by areal BMD and subject-specific specimen thickness, so that the ash density or apparent density could be estimated from volumetric BMD. Bone material is inhomogeneous and anisotropic. In DXA scans, inhomogeneous material properties are represented by BMD. However, since DXA images are 2D, they do not contain any anisotropy information of bone material. Therefore, this model was assumed to be inhomogeneous and isotropic based on results from a previous study, where isotropic material properties were used to construct a DXA-based finite element model [19]. The thickness of the model was approximated to be a patient-specific value, which was defined by the width of the narrowest femoral neck [20]. The loading and boundary conditions were then applied to the model to simulate a sideways fall from standing height.

Afterwards, finite element solutions were obtained and postprocessed. Hip fracture risk indices (HFRIs) were calculated at the critical femur cross sections and ROIs. The definition of HFRI was initially proposed by Luo et al. [27]. It was defined as the ratio between the effective - or von Mises - stress induced by the external force and the yield stress at cross sections or ROIs. Additionally, improvement in the hip fracture discriminability was measured using the area under receiver-operating-characteristics curve (AUC) value.

In this finite element model, the segmentation of proximal femur from clinical DXA scans was fully automated, followed by the subject-specific simulation of a sideways fall. The HFRIs were then calculated at two positions to discriminate hip fracture risk. The developed DXA-based finite element model has the potential to be applied in the routine clinical assessment of hip fracture risks. However, the main limitations are the lack of quality examination of DXA images and intellectual property restriction in redistribution of the computer codes, which will be further discussed in the next two sections.

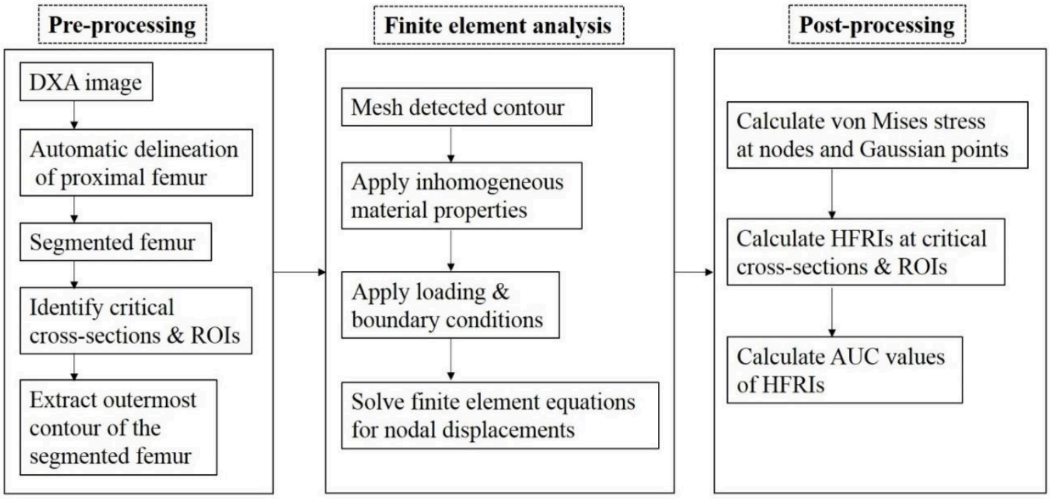

Pre-processing
DXA image
Automatic delineation of proximal femur
Segmented femur
Identify critical cross-sections & ROIs
Extract outermost contour of the segmented femur
Finite element analysis
Mesh detected contour
Apply inhomogeneous material properties
Apply loading & boundary conditions
Solve finite element equations for nodal displacements
Post-processing
Calculate von Mises stress at nodes and Gaussian points
Calculate HFRIs at critical cross-sections & ROIs
Calculate AUC values of HFRIs

Figure 2-2: Procedure to calculate HFRIs and to assess the hip fracture (adopted from [17])

2.3 Quality examination of DXA images

Based on a survey taken in 2006 [30], poor quality of DXA images is not uncommon in clinical practice, which has adverse effect on BMD accuracy and thus diagnosis accuracy. 25% of the clinicians or physicians responding to the survey claimed that they had encountered poor quality DXA images at least once a week. Almost all of the respondents believed that DXA images of poor quality are harmful to patients due to the unnecessary medical expenses and wrong treatment decisions that may lead to.

Although the DXA-based finite element model developed in the previous study [20] has improved accuracy and discriminability in prediction of hip fracture, its application to large scale clinical cases is still prevented by the lack of an algorithm for quality examination of DXA images. The quality of DXA image determines the quality of the segmented proximal femur contour, which affects the quality of finite element mesh and the precision and repeatability of HFRIs. A defective contour of proximal femur can cause fault stress distribution in femur and thus affect HFRI. The quality of the segmented femur also has an impact on ROI locations, e.g. the position of femoral neck cross-section. Therefore, using DXA images of high quality can further improve the discriminability and repeatability of HFRIs predicted by the DXA-based finite element model. Furthermore, the DXA-based finite element modeling process can be interrupted by defective DXA images. Therefore, quality examination of DXA images is an essential step before the DXA-based finite element model can be used by clinicians.

Possible solutions to quality control of DXA images have been studied by a number of researchers [24, 27, 28, 29]. Firstly, the technologists or clinicians of DXA scanning should be trained in bone densitometry [30]. In addition, they should regularly maintain DXA devices to avoid low-quality scans [31]. At each scanning, the DXA image should be examined to make sure that the positioning and segmentation are correct.

Apart from the measures mentioned above, the methods developed by Luo et al. [21] can be implemented to evaluate the quality of segmented contour for improving the repeatability of HFRIs. Poor contour quality reduces the repeatability in both ROI average areal BMD and fracture risk indices (FRIs). The errors introduced in scan positioning and segmentation of DXA images, also known as random errors, can reduce the precision of DXA-based finite element procedure. Unfortunately, random errors could not be eliminated by any refinement or improvement in finite element modeling. To resolve this problem, Luo et al. developed a quality index in their study as a measure of random errors (or the quality of proximal femur contour) [21]. With the quality index, segmented proximal femur contours can be classified into high, moderate or low quality. They also applied image processing techniques, e.g. image translation and rotation, to partially remove random errors. Moreover, strict guidelines should be adopted to regulate positioning in DXA scanning.

In this study, the quality examination of DXA images is an essential part for redistributing DXA-based finite element model to clinicians. A method or an algorithm is required to automatically identify the quality of DXA images and exclude the ones with defects prior to finite element analysis. However, so far there has not been a clinically applicable algorithm for quality assessment of DXA images. As mentioned before, using high-quality DXA images can improve the precision of this model and guarantee a fully automatic hip fracture risk assessment. Therefore, it is necessary to develop an algorithm to automatically examine quality of DXA images. The quality of segmented proximal femur contours should be assessed. Accordingly, DXA images can be classified into qualified and unqualified ones.

Therefore, the quality examination of DXA images in this study can be regarded as a binary classification problem, which can be solved by a number of algorithms, for example *k*-NN algorithm. As one of the most popular machine learning algorithms and a conventional non-parametric classifier, *k*-NN has already been used in the statistical estimation and pattern recognition starting in the 1970's [33]. The *k*-NN algorithm can classify a sample based on the cluster of its *k* nearest neighbors, as shown in Figure 2-3. An unclassified contour image can be identified based on its *k* nearest images. Compared with other machine learning algorithms, the principle of *k*-NN is simple and does not need any off-line training. It has been widely used in many studies and has achieved excellent performance [34]. Hence, *k*-NN algorithm will be selected for classifying the segmented femur contour images as well as the corresponding DXA images.

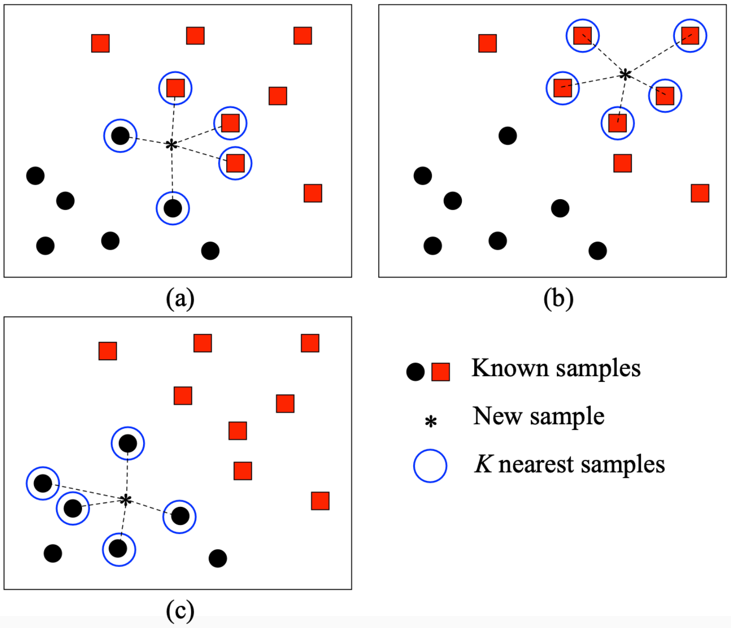


Figure 2-3: *K*-nearest neighbor classification

2.4 Necessity to develop a standalone program for hip fracture risk assessment

In addition to lack of quality examination of DXA images, another issue is intellectual property restriction to redistribute computer codes that implemented the DXA-based finite element model. As mentioned in Chapter 2.2, this finite element model has been developed using in-house MATLAB code, which makes it almost impossible for clinicians or physicians who have no knowledge in MATLAB to use [20]. It also requires users to install MATLAB software on their computers. Moreover, it is restricted by intellectual property regulations to redistribute MATLAB source codes involving previous students. Therefore, a standalone and user-friendly computer program should be developed based on the existing model to benefit the wide clinical community.

For redistribution purpose, in-house MATLAB codes can be packaged into a standalone computer program using MATLAB Compiler. A user-friendly interface can also be designed for this program in MATLAB. The packaged program uses MATLAB Runtime as its operating environment [35] and prevents exposure of the MATLAB source codes. It does not require any programming knowledge or the installation of MATLAB for users to run the program. Hence, the packaged program can be conveniently redistributed for the clinical assessment of hip fracture risks. However, it still requires the installation of MATLAB Runtime, which can take up some storage space in the user’s computer.

A web-based biomechanical tool implementing the DXA-based finite element model can become an alternative for FRAX, which is currently the most popular web-based tool for fracture risk assessment and based on statistical modeling [10]. With a web-based assessment tool, physicians or patients can upload DXA images and calculate the HFRIs immediately, avoiding the need to install software or operating environment. Furthermore, it can make the assessment more convenient without taking up any space on the computer, while the standalone program needs to be installed on the client's computer. For the above reasons, the third topic of this thesis is to develop a framework of a web-based tool for the routine clinical assessment of hip fracture risks based on the existing DXA-based finite element model.

Despite the recent development in scanning technologies, many patients with high risk of osteoporosis still cannot be diagnosed accurately [36]. For a more convenient clinical diagnosis of osteoporosis and related fractures, various web-based assessment tools have been developed. The most widely used one is the FRAX tool developed by World Health Organization (WHO) Collaborating Centre for Metabolic Bone Diseases at Sheffield, UK [37] - first launched in 2008. FRAX is a web-based tool used to estimate 10-year probability of major osteoporosis fractures and hip fracture for both men and women [12]. It integrates different clinical fracture risk factors, including age, sex, and low body mass index [10], with or without information on BMD [38]. So far, FRAX is available in over 60 countries or regions and can be calibrated for different populations [39].

FRAX aims at identifying individuals with high risk of fracture [37]. Since 1st June of 2011, over 21,152,000 individuals with fracture risk have been identified using FRAX [39], and the website has an average of 55,000 hits daily [12]. It is undoubtedly that FRAX has been a significant achievement in assessment of fracture risks. As the only model based on the data derived from multiple cohorts, FRAX has been widely accepted in treatment decisions, such as in the National Osteoporosis Foundation (NOF) guidelines [40].

However, FRAX has a number of limitations such as incomplete list of variables in the assessment and not considering dose-response relationships [40]. It is not suitable for clinical application since there are still challenges in the construction of new clinical guidelines and consideration of drug registration [12], and it cannot be used for patients that undergo osteoporosis treatment [41]. Additionally, there is no FRAX algorithm for the pediatric population, since the relationship between BMD and the fracture risk for children has not been established [37].

Apart from FRAX, there are other web-based tools for fracture risk assessment that are being less widely used, examples are the Garvan tool [37], QFracture [42], and Osteoporosis Self Assessment Tool (OST) [44, 45]. Since OST only considers age and body weight in the assessment [45], while others include more risk factors, this section does not discuss OST in detail or compare it with other tools.

The Garvan calculator, or Garvan Institute Assessment Tool, was developed based on the data collected in the Dubbo Osteoporosis Epidemiology Study (DOES), which included approximately 2500 men and women 60 years or older [46]. Compared with FRAX, the Garvan calculator includes the history of falls over the last 12 months and fractures since the age of 50 in the assessment; and does not include some other factors that are considered in FRAX, such as smoking or alcohol consumption [47].

QFracture was based on a UK open cohort study using data derived from 357 clinical practices on over 2 million men and women in the age range of 30 to 85 years [48]. It is used to predict the 10-year risk of hip fracture or osteoporotic fracture at multiple locations. QFracture can be applied to men and women between 30 and 99 years old, who have no history of osteoporosis fracture [42]. Unlike the other two calculators, QFracture does not include BMD in the assessment process at all [46].

The detailed comparison among these three web-based tools, including the specific differences due to the input variables, output fracture risk, data acquisition and calibration ability, has been reviewed by J. A. Kanis et al. [46]. The web-based assessment tools discussed in this section all have certain advantages and limitations; for example, the Garvan calculator is challenging to be calibrated for new population [49], while the modeling of QFracture is based on a limited data source [46]. It is therefore impractical to conclude which tool is the best for assessing osteoporotic fracture risk based on the aforementioned reasons. However, the common and major limitations of these assessment tools are not able to perform the subject-specific assessment and having different requirements on the subjects, for example, FRAX can only be applied for untreated patients [41]. Therefore, they are not suitable for the clinical assessment. On the contrary, not all pysicians have Internet access in their examination rooms [50], which prevents them from using web-based tools. In addition, it should be noted that the purpose of these web-based tools is not completely replacing clinical diagnosis required by a recognized practitioner [37] [40]. These tools have an enormous impact on identifying potential fractures and providing a more targeted prevention or treatment process.

In conclusion, a standalone program should be first developed based on the DXA-based finite element model for the clinical assessment of hip fracture risks. Meanwhile, inspired by the web-based assessment tools discussed above, the framework of a similar web-based tool will also be presented in this study as another redistribution method and a direction for future development. Since the web-based tool will be developed based on a subject-specific finite element model, the common and major limitation of the existing tools can be resolved, with the subject-specific assessment for hip fracture risk being achieved. Consequently, the routine clinical assessment of osteoporotic hip fracture risk will be easier with the standalone program developed in this study, while the future studies can be conducted based on the framework of the web-based tool.

Chapter 3

Quality examination of DXA images

In this chapter, an introduction is made to DXA images and the examination of DXA image quality. Two algorithms for examining DXA quality are presented. The validation of these algorithms will be presented in Chapter 6.1.

3.1 Bone imaging techniques

At the end of last century, non-invasive assessment methods for human skeleton imaging experienced a rapid evolution [51]. In 1987, DXA was commercially introduced for the measurement of BMD based on the technique of X-ray spectrophotometry.

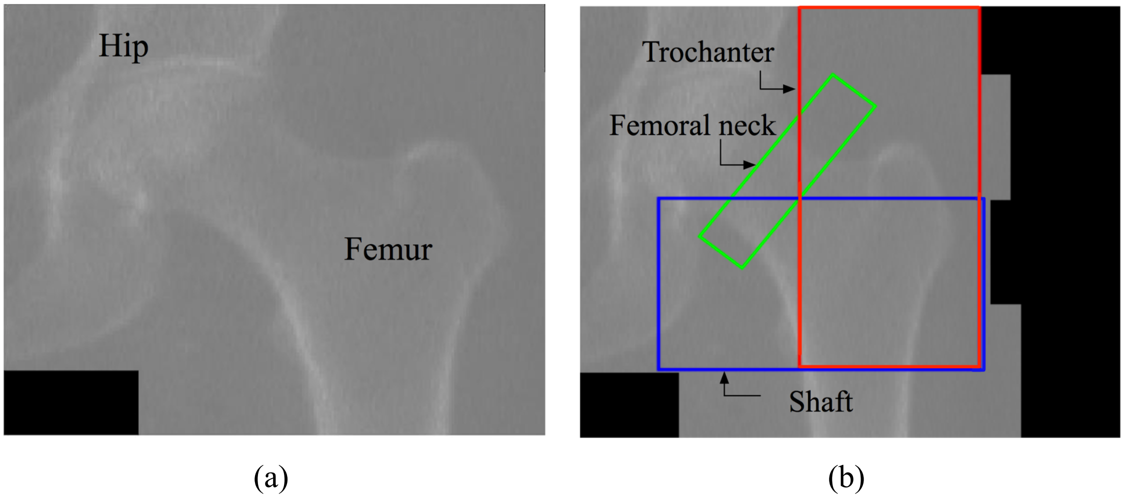
In addition to DXA, another popular non-invasive BMD assessment method was QCT [52]. QCT can measure the true physical density of the bone with the unit of $\mathrm{g/cm^3}$. However, one of the major limitations of QCT was the higher ionizing radiation level required, compared with DXA [53]. Hence, the use of QCT in clinical applications is not prioritized as DXA has been developing rapidly.

The fundamental physical principle of DXA is to measure the energy decay of two X-rays with high- and low-photon energies following transmission through the body [52]. Same as other radiologic techniques, it relies on different absorption level of X-rays to identify tissues with different radiographic densities. For a selected ROI with the unit of cm^2, bone mineral content (BMC) is measured at each pixel based on its X-rays attenuation. The BMD at each pixel can be calculated through dividing BMC by the pixel's area. For the whole ROI, DXA measures BMD ($\mathrm{g/cm}^2$) as the integral of bone mineral over the projected area, which is in contrast to QCT. Despite that, DXA is still widely used in clinical practice due to its low radiation dosage to patients, short scanning time and low cost [52]. One of the most critical applications of DXA is establishing the relationship between BMD and fracture risk in either the prediction or clinical treatment of osteoporosis. Besides, DXA has been used for measuring body fat composition and evaluating the effects of pharmaceutical therapy [28]. Figure 3-1 shows a typical DXA scanner used in clinics or clinical studies. DXA has been adopted by WHO as the official evaluation tool for hip fracture risks [29]. The aBMD measured by DXA is the most commonly used measurement criterion for diagnosing osteoporosis [28].

Figure 3-1: A DXA scanner [54]

In this study, a DXA image is accessed both in the native manufacturer format ('enCore') and as a MATLAB file ('mat'). The file in the native manufacturer format contains information on ROIs, while the MATLAB file contains BMD distribution in terms of pixel values [17]. The ROIs enables the automatic segmentation of proximal femur contour. A sample DXA image can be seen in Figure 3-2(a), where the gray scale of any pixel represents the BMD at that point. Figure 3-2(b) shows the clinical ROIs, which are inherent to DXA devices. The coordinates corresponding to the femoral neck, trochanter and shaft regions were extracted from the 'enCore' header file.

In Figure 3-3, self-defined ROIs and the corresponding critical cross-sections are defined. Critical cross-sections are the regions that are most prone to fracture, which are the narrowest femoral neck, the intertrochanter and the subtrochanter. The narrowest femoral neck is the section with the minimum diameter at the femoral neck [55]. Femoral neck axis is orthogonal to and bisects the narrowest femoral neck section, and the femoral shaft axis is defined as the axis of femoral shaft [56]. With these two axes, the intertrochanter cross-section is the bisecting line of the angle formed by the neck and the shaft axes. The distance between subtrochanter cross-section and the femoral neck-shaft intersection point is 1.5 times of the narrowest femoral neck width [55]. The self-defined ROIs are 10mm wide rectangular regions around the three critical cross-section lines.



(a) (b)

Figure 3-2: A sample DXA image and clinical ROIs

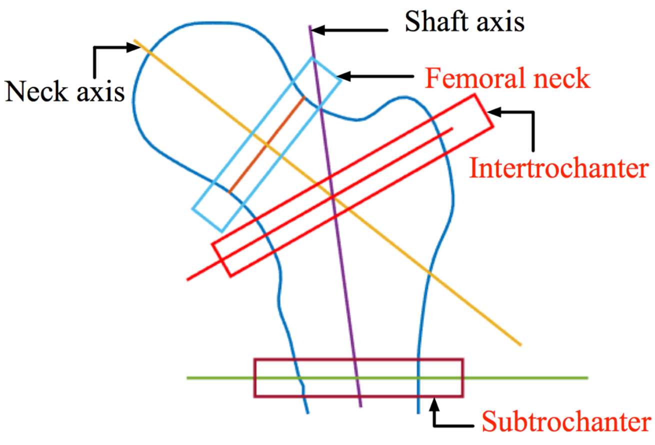
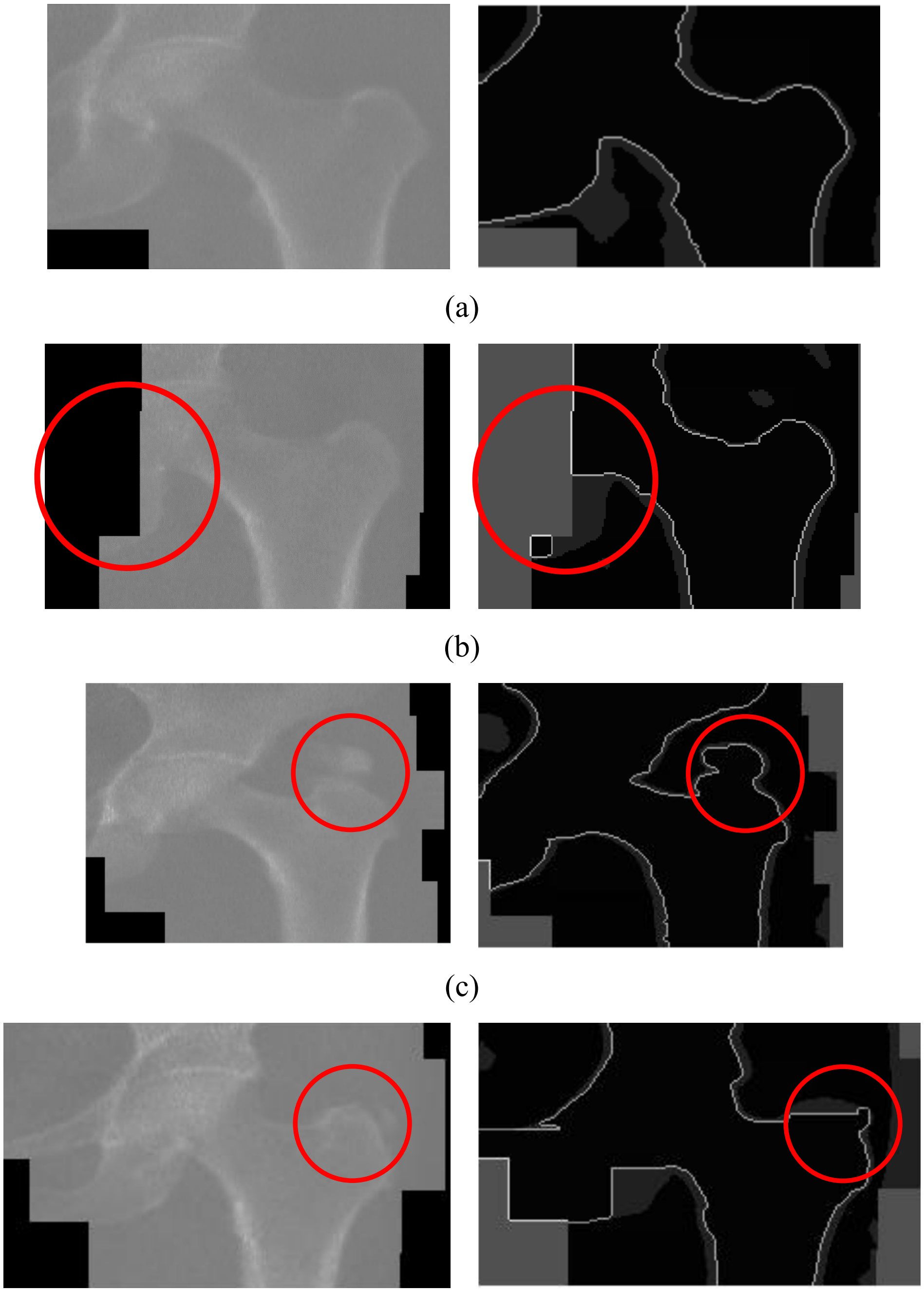


Figure 3-3: Critical cross-sections and self-defined ROIs

Examples of qualified and unqualified DXA images with the corresponding hip-shaft outlines are shown in Figure 3-4. In a qualified DXA image, the bone edges and the hip-femoral shaft outline should be clear. Moreover, the scan length should be long enough to identify all the ROIs. In the sample unqualified DXA images (Figure 3-4 (b), (c) and (d)), the hip outline or part of the trochanter ROI is missing, which are marked on the figures. As discussed before, DXA images with poor quality are not uncommon in clinical practice, and they can affect the precision of HFRI and the operation of FEA. Many mis-operations such as short scan length, improper or inconsistent positioning can induce quality issues to DXA scans.

Defects in DXA images, which will cause failures in DXA-based finite element modeling, are reflected as certain features in the segmented proximal femur contour. Therefore, it is critical to identify these features in the proximal femur contour. To ensure a successful assessment of hip fracture risks, DXA images need to be automatically classified into qualified and unqualified ones in advance based on the generated proximal femur contours. Previously, this classification process was completed by visual inspection.



(a)

(b)

(c)

(d)

Figure 3-4: (a) Sample qualified DXA image and hip-shaft outline; (b), (c), (d) Sample unqualified DXA images and hip-shaft outlines

A qualified proximal femur contour should be clear and smooth, as shown in Figure 3-5. Typical unqualified proximal femur contours can be approximately divided into four categories, which are described in Figure 3-6.

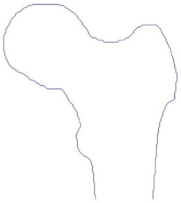
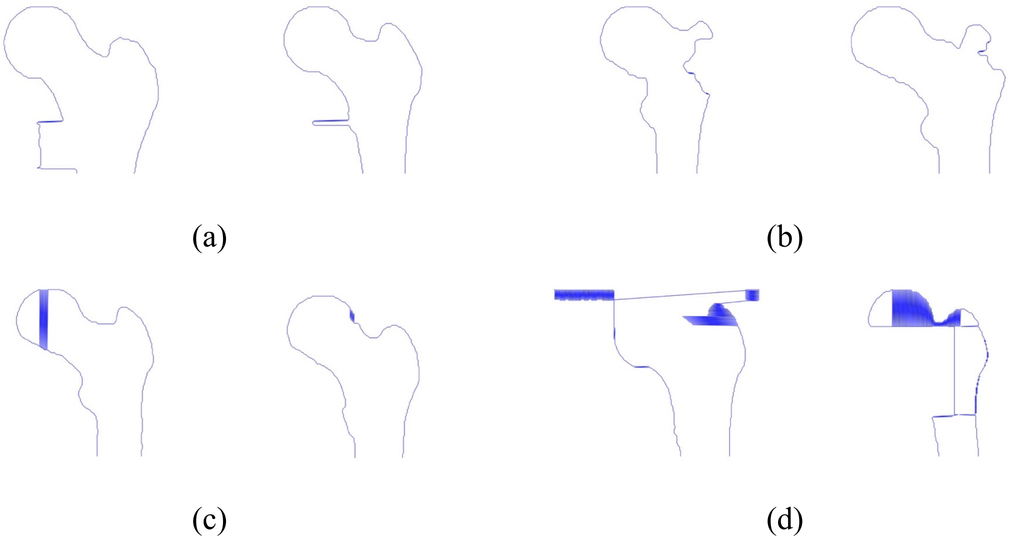


Figure 3-5: A proximal femur contour generated from the DXA image



(a)

(b)

(c)

(d)

Figure 3-6: Unqualified proximal femur contours with (a) redundant part, (b) missing part, (c) fluctuation, (d) incorrect shapes

3.2 Quality examination of DXA images based on geometric features

3.2.1 Geometric features extracted from proximal femur contour images

Before undergoing quality examination, geometric features of poor quality in femur contours must be identified and quantified, which will be used as criteria in classification of DXA images. To identify these features, image processing is required.

There are two types of pixels in a contour image, foreground and background pixels. Foreground pixels refer to the pixels that form the femur contour, while the rest are background pixels. The purpose of image processing is to bring out the foreground pixels and further isolate any defective area. Afterwards, geometric features of unqualified contours have to be more distinct and able to be quantified, so that computer algorithms can easily identify them.

Proximal femur contour images segmented from DXA images are RGB images with three color codes, i.e. red, green and blue. In other words, each pixel in the contour image is made of the combination of three colors. For a more convenient feature extraction, these contour images need to be converted into gray-scale images. This step can be considered as the basis of the following image-processing procedures. In a gray-scale image, pixels are represented by their gray-scale values, from 0 (black) to 1 (white). After loaded into MATLAB, the proximal femur contour image is converted into a gray-scale image and subtracted from 1, which is the grayscale value of white, as shown in Eq. (3-1). A sample contour image and the corresponding gray-scale image are presented in Figure 3-7.

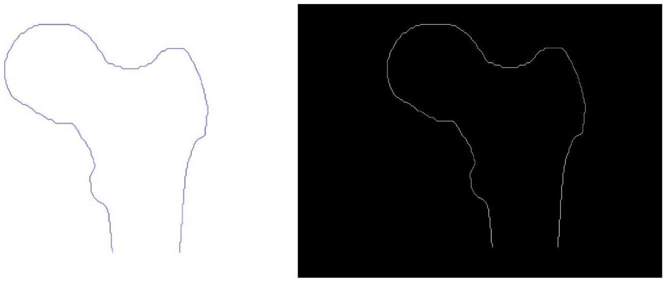


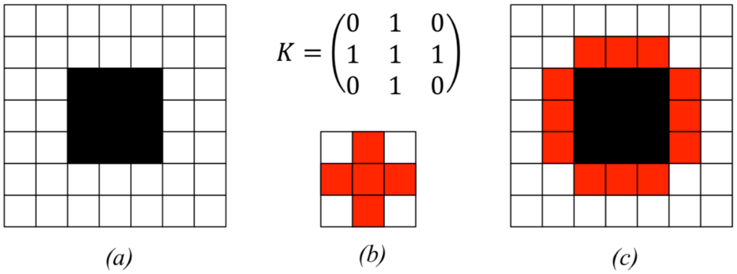
Figure 3-7: The proximal femur contour image and corresponding gray-scale image

$$\begin{gathered} img(x, y) = rgb2gray\big(img(x, y)\big) \\ img(x, y) = 1 - img(x, y) \end{gathered} \qquad \text{Eq. (3-1)}$$

The gray-scale image is then dilated to extend edges of the femur contour and enlarge any defective area. As one of the basic operations of mathematical morphology, the main effect of image dilation is to gradually enlarge the boundaries of the foreground pixels (in this case, white pixels), which are the pixels used for the object(s) in an image, while the rest are called background pixels. The operation of image dilation takes two matrixes as inputs, i.e. the source image that is to be dilated and the kernel (or the structuring element) that determines how the source image will be transformed. In Eq. (3-2), img_{src} and img_{dst} represent the source image and the dilated image. Since image dilation is a particular type of convolution operation, K can be considered as the kernel used during convolving. Eq. (3-3) shows the specific equation for image dilation. Additionally, the process of image dilation is illustrated in Figure 3-8, with the dilated image shown in Figure 3-9.

$$img_{dst} = Dilation(img_{src}, K) \qquad \text{Eq. (3-2)}$$

$$dst(x, y) = \max_{K(x', y') \neq 0} src(x + x', y + y') \qquad \text{Eq. (3-3)}$$


$K = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$

(a) (b) (c)

Figure 3-8: Dilation (c) of the image (a) with the structuring element (b)

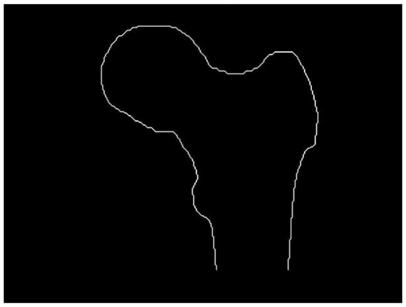
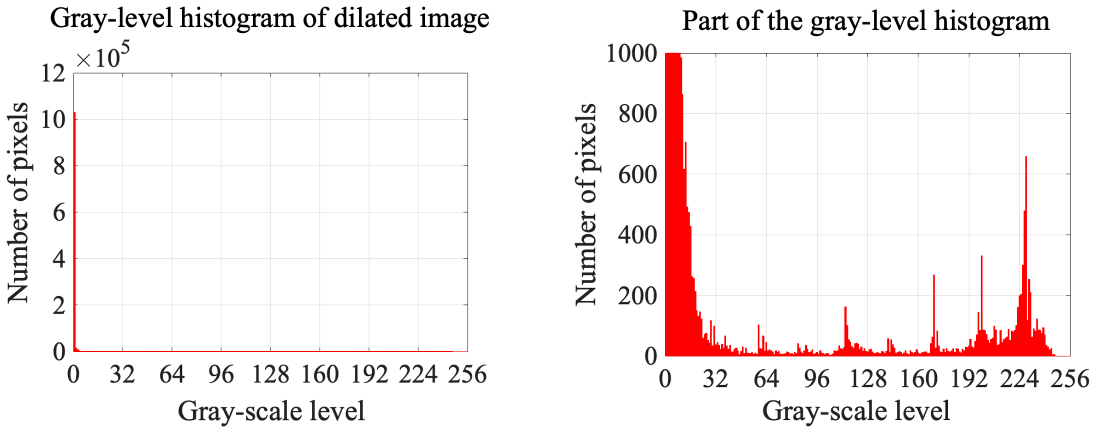


Figure 3-9: The dilated image

After image dilation, a threshold is applied to the image to convert the current image into a binary image, which is beneficial to obtain clearer edges. A pixel on a binary image has only two possible values, i.e. black or white. Therefore, the relationship between pixels can be uniquely expressed in a binary image [57]. In many cases, such as medical image processing or character recognition, image binarization is conducted for a more convenient analysis.

To determine the optimal threshold of image binarization, the gray-scale distribution of dilated pixels needs to be observed. Gray-scale values of every pixels on the dilated image are accounted for in a histogram, which is shown in Figure 3-10 (a). Besides, a partial view (Figure 3-10 (b)) is provided where the largest count of pixels is 1000.



(a) (b)

Figure 3-10: Gray-level histogram of the dilated image

According to the histogram, foreground and background pixels have no obvious separation, and there exist multiple peaks of gray-level between them. The proportion of foreground pixels in whole image is unknown, since the scanning image is subject-specific. Therefore, Otsu's method [58] is selected among the common methods. This method aims to search for a threshold that maximizes the inter-class variance of class possibilities. For example, for an image containing *L* pixels, pixels are separated into two classes, *1~T* (object) and *T~L-T* (background). The average gray-scale value is calculated for two classes based on the possibility of each gray-level occurs. Then, the desired threshold will be the value of *T* that can maximize the inter-class variance. Hence, the difference between femur contour and background can also be maximized.

Otsu's method is implemented for all dilated images in the train and validation sets (see Table 3-1) to achieve the optimal threshold of image binarization for each case. Since image binarization influences the determination of foreground pixels, a certain and universal threshold should be applied for all cases in the computer algorithm instead of using different thresholds. And this certain value can be determined based on the achieved optimal thresholds for each case.

Over 96% of achieved threshold values are located between 0.3863 and 0.4. Overall, these values have a fairly small variation, the range between maximum and minimal values is 0.0471, and the coefficient of variation (see the following equation) is 1.12%. Therefore, the mean of these values is selected as the optimal threshold in this study, which is 0.3952. Figure 3-11 shows the binary image, which has more well-defined edges compared with the dilated image.

$$Coefficient\ of\ variation = \frac{\sqrt{\frac{\sum(x-\overline{x})^2}{n}}}{\overline{x}} \times 100\% \qquad \text{Eq. (3-4)}$$

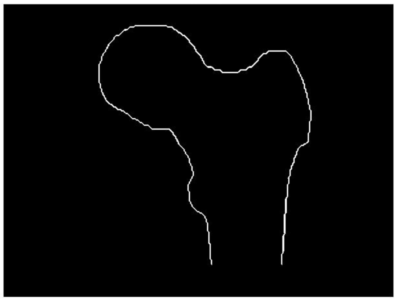
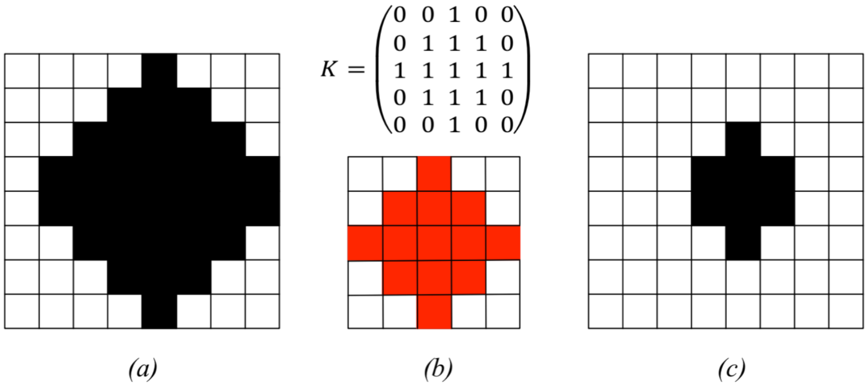


Figure 3-11: The binary image

The binary image is then subject to image erosion for thinner edges, removing non-defective areas and better isolating the defective ones. Similar to image dilation, image erosion is also a basic operation of mathematical morphology. The fundamental effect of image erosion is to erode pixels along the boundaries of foreground pixels (in this case, white pixels), which can be regarded as the opposite of image dilation.

The operation of image erosion also takes two matrixes as inputs, which are the source image that needs to be eroded and the kernel - akin to the kernel matrix used in the image dilation. In Eq. (3-5), img_{src} and img_{dst} represent the source image and the eroded image.

Eq. (3-6) describes the evaluation of image erosion. Figure 3-12 illustrates the process of image erosion. Figure 3-13 presents the eroded image, which is almost a full black image with several small white segments. The features used for quality examination are extracted from the eroded image.



$$K = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Figure 3-12: Erosion (c) of the image (a) with the structuring element (b)

$$img_{dst} = Erosion(img_{src}, K) \qquad \text{Eq. (3-5)}$$

$$dst(x, y) = \min_{K(x',y')\neq 0} src(x + x', y + y') \qquad \text{Eq. (3-6)}$$

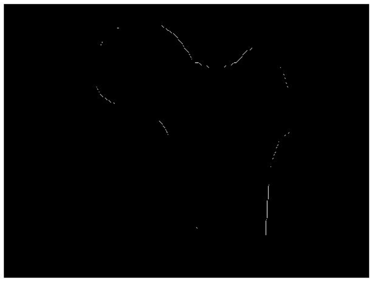


Figure 3-13: The eroded image

For an unqualified contour with fluctuations, the effect of image dilation and erosion for the defective area can be seen from Figure 3-14. The boundary of whole contour is enlarged with image dilation, including the defective pixels. Then, image erosion deletes most non-defective pixels to segment the defective areas. Therefore, the defective area becomes more distinct after these image operations, as a patch of pixels instead of a line of pixels. As for a qualified contour, the maximum number of pixels in a patch left in the eroded image will be much less than the ones of unqualified contours, since there are no defective areas. As a result, the maximum number of continuously neighboring pixels in the eroded image can be used for separating the unqualified contours from the qualified ones.

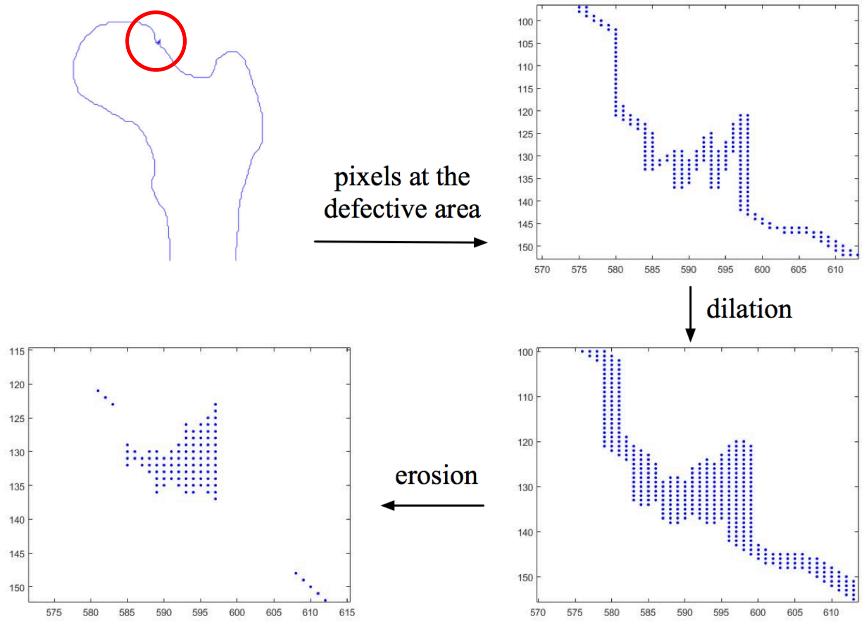


Figure 3-14: Effects of image dilation and erosion

In the eroded image, a patch of pixels can be considered a connected component, which is formed by adjacent foreground pixels with the same value [59]. For feature extraction, the area of each patch in the eroded image is calculated, and the maximum area is selected as the 'feature' for the quality examination (see Eq. (3-7)). These features are then saved into a MATLAB file ('mat') for any further uses. The difference between the features extracted from qualified and unqualified contours will be demonstrated in the next section.

$$feat = \max\{area_i | i = 1, 2, \dots n\} \qquad \text{Eq. (3-7)}$$

3.2.2 Differences between qualified and unqualified proximal femur contours

Figure 3-15 displays the proximal femur contours segmented from sample qualified and unqualified DXA images. In this case, it is evident that there exist fluctuation or folding lines in the femoral neck region of the unqualified contour. After image processing, the difference between the qualified and unqualified contours become more evident in the eroded images, as shown in Figure 3-16.

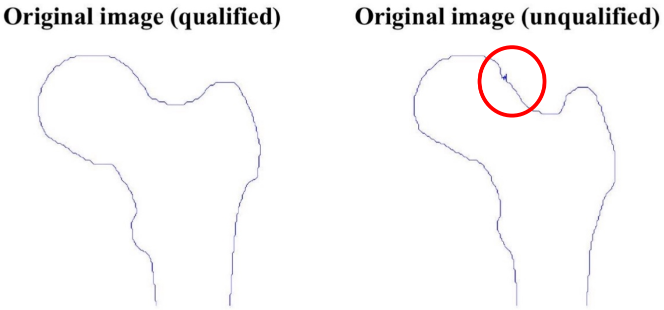


Figure 3-15: Segmented contours from qualified and unqualified DXA images

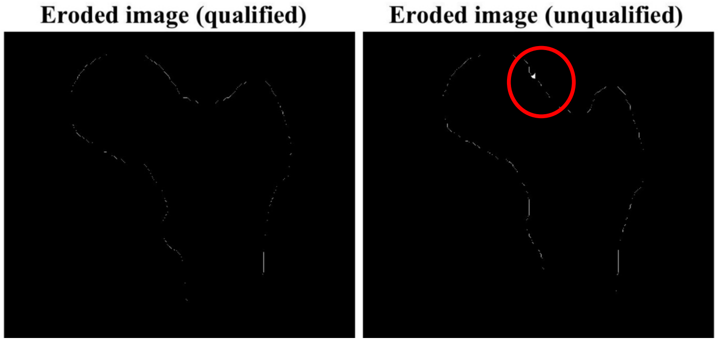


Figure 3-16: Eroded images of qualified and unqualified contours

Figure 3-17 shows the zoom-in views of the same part on the sample qualified and unqualified contours. The visual difference between non-defective and defective areas is quite obvious in this figure.

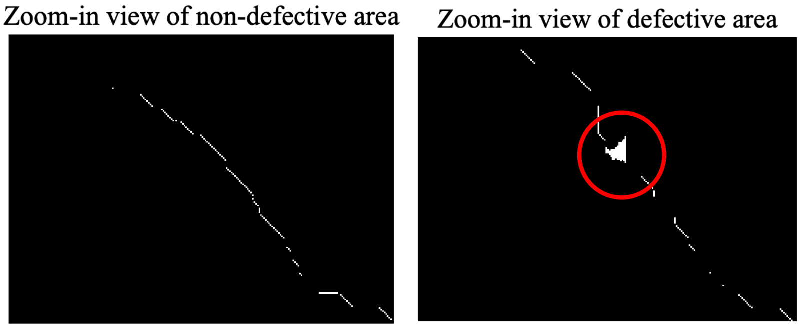

Zoom-in view of non-defective area
Zoom-in view of defective area

Figure 3-17: Zoom-in views of non-defective and defective areas

As explained before, the maximum number of continuously neighboring pixels, or the maximum area of pixel patches, can be used as the feature for quality examination. Therefore, after image erosion, areas of every pixel patches should be calculated with the unit of pixel, then the maximum one is selected as the feature. In Figure 3-18, the features extracted from all the qualified and unqualified samples in the train set (see Table 3-1) are plotted into histograms. It can be observed that the features of qualified and unqualified contours have different distributions with nearly no overlap with each other. In other words, the extracted feature of a single contour is either located in the 'qualified zone' or the 'unqualified zone.'

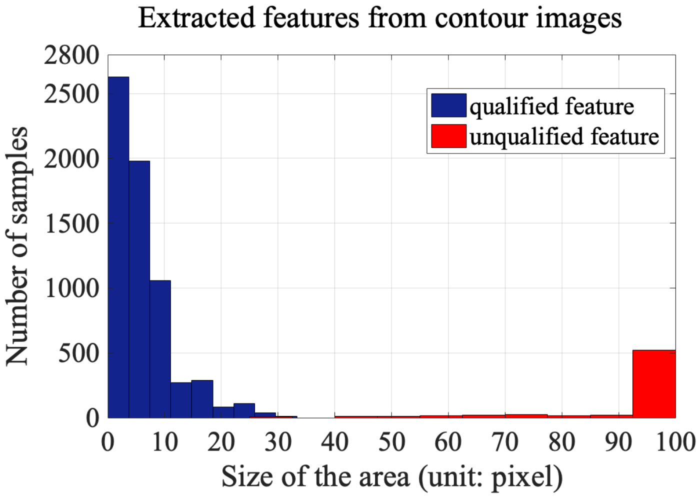
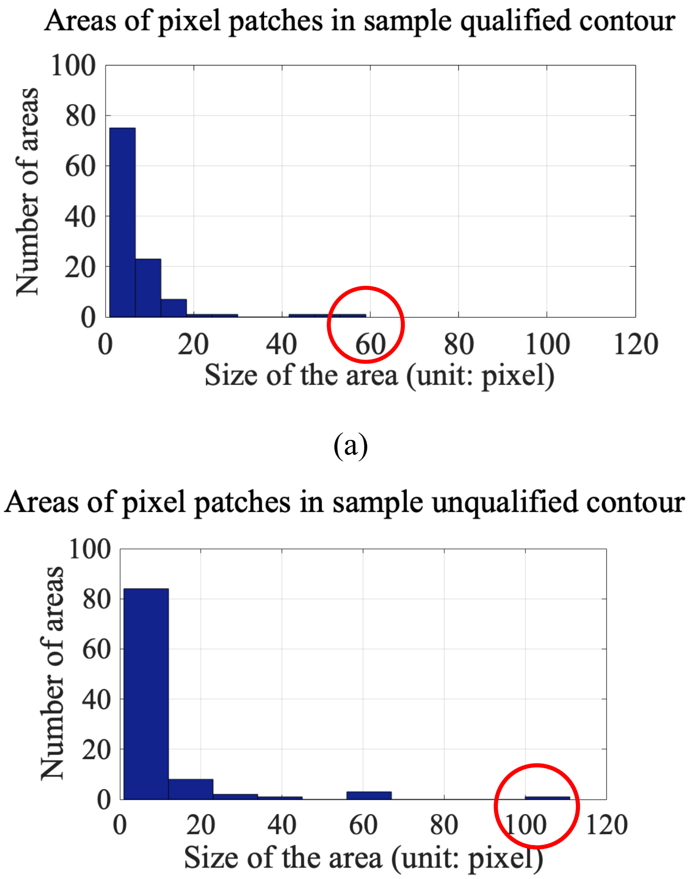


Figure 3-18: Statistic features of DXA images in the train set

For example, features of the sample images provided in this section can be seen in Figure 3-19. It can be observed that the feature of qualified sample is smaller than 60 pixels, while the unqualified one is significantly larger, which is over 100 pixels. Hence, these two samples can be easily distinguished based on the extracted features.



(a)

(b)

Figure 3-19: Features extracted from the sample contours

In conclusion, it is feasible to classify qualified and unqualified DXA images based on the extracted feature, which is the maximum area of pixel patches in the eroded image. By using an algorithm, such as *k*-NN, to recognize the features, unqualified DXA images are easy to be identified and excluded from the finite element modeling procedures.

From Figure 3-20 to Figure 3-23, the features used for identifying the four types of unqualified contours are demonstrated. Compared with non-defective part, detects in the femur contour can be transformed into considerably larger pixel patches in the eroded image.

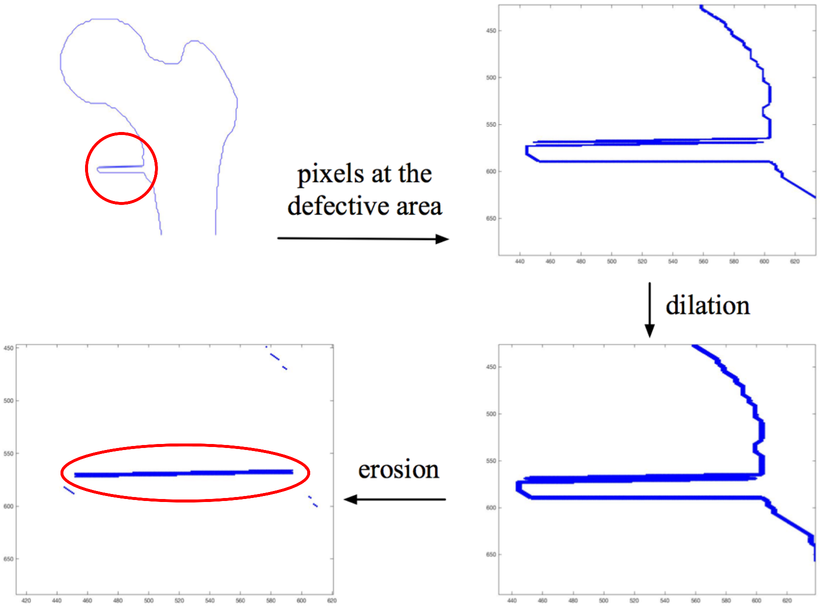


Figure 3-20: Feature extraction of an unqualified contour with redundant part

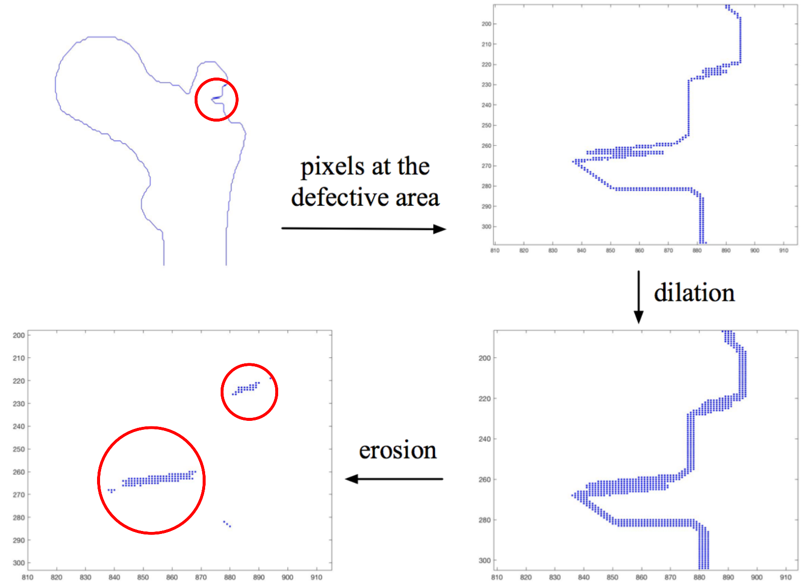


Figure 3-21: Feature extraction of an unqualified contour with missing part

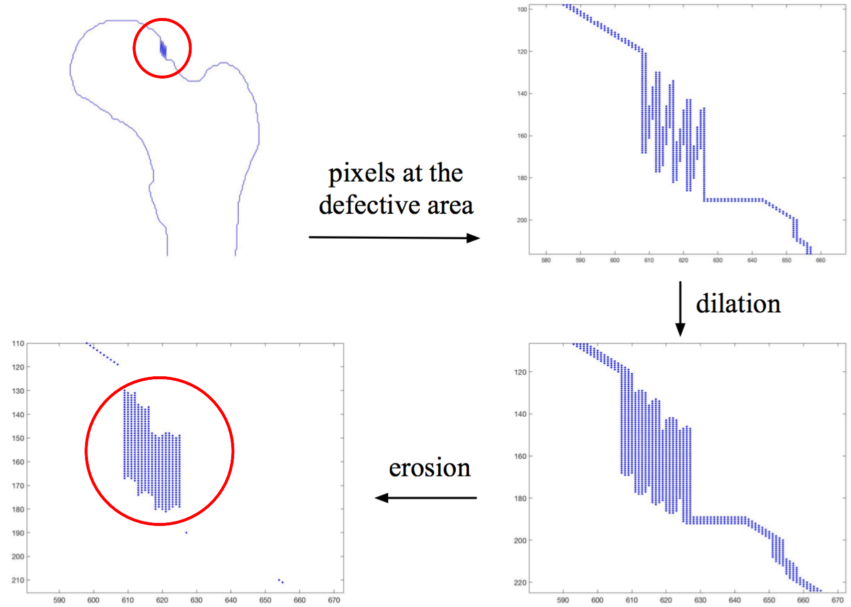


Figure 3-22: Feature extraction of an unqualified contour with fluctuation

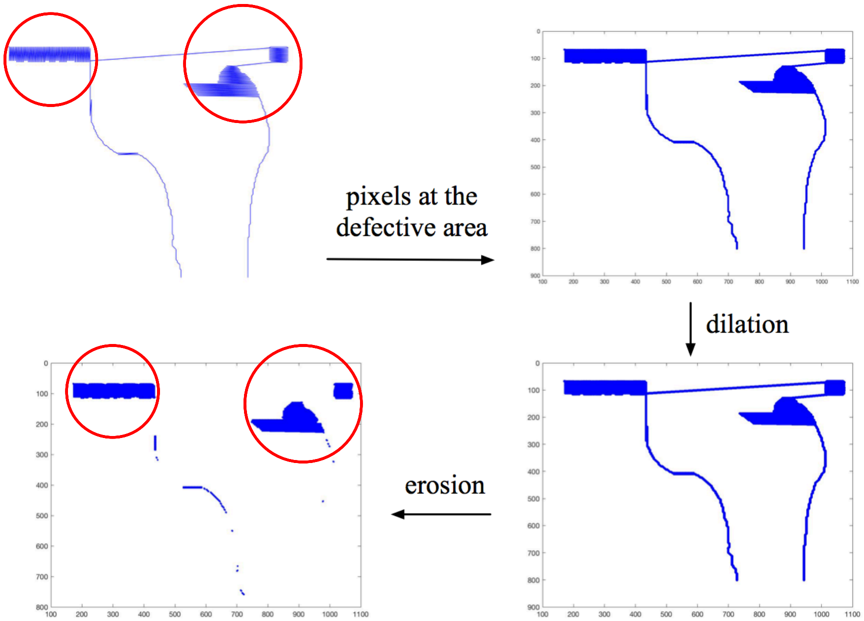


Figure 3-23: Feature extraction of an unqualified contour with incorrect shape

3.2.3 Quality examination using *k*-NN algorithm

After feature extraction, *k*-NN algorithm is applied to classify the features and examine the quality of DXA images. It is a simple, non-parametric algorithm that can store all available samples and classify new samples based on a measure of similarity, for example the cosine distance function [60]. In the *k*-NN algorithm, each sample can be represented by its *k* nearest neighbors, where *k* is a positive content-defined integer. Hence, an unclassified sample should be assigned to the category which is most common among its closest *k* nearest neighbors. Certainly, the classes of these neighbors need to be determined prior to the classification. This process can be considered as the 'training phase' of the *k*-NN algorithm, as there is no need for explicit training. To implement the *k*-NN algorithm in this study, the training set and validation data set shown in Table 3-1 were used.

Table 3-1: Details of train and validation sets

	Number of positive samples (qualified DXA images)	Number of negative samples (unqualified DXA images)
Train set	6464	636
Validation set	2586	254

The most critical and basic components of *k*-NN algorithm are the distance matrix and the *k*-value. Figure 3-24 shows the influence of distance function and *k*-value to the testing accuracy using the same validation set (see Table 3-1). Statistically, data can be recognized as numerical, categorical or mixed. Numerical data have the meaning as a measurement or a count, while categorical data represent characteristics without any mathematical meaning. Based on an experimental study on the distance function effect [60], Euclidean distance (see Eq. (3-8)) can make the *k*-NN classifier perform the best for numerical datasets. Chi-square distance (see Eq. (3-9)) is the optimal one for mixed type of datasets, which includes both numerical and categorical data. For categorical data, the performance of these two functions varies with the number of attributes. Although the feature in this study, the maximum area of pixel patches, is a type of numerical data, the performances of both Euclidean and Chi-square distances were compared, as shown in Figure 3-24. Overall, the Euclidean distance achieved better results and thus was selected.

$$D = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \qquad \text{Eq. (3-8)}$$

$$D = \frac{1}{2}\sum_i \frac{(x_i - y_i)^2}{(x_i + y_i)^2} \qquad \text{Eq. (3-9)}$$

In above equations, D is the distance between x and y; k is the number of dimensions; x_i and y_i represent the coordinates of two objects at this dimension.

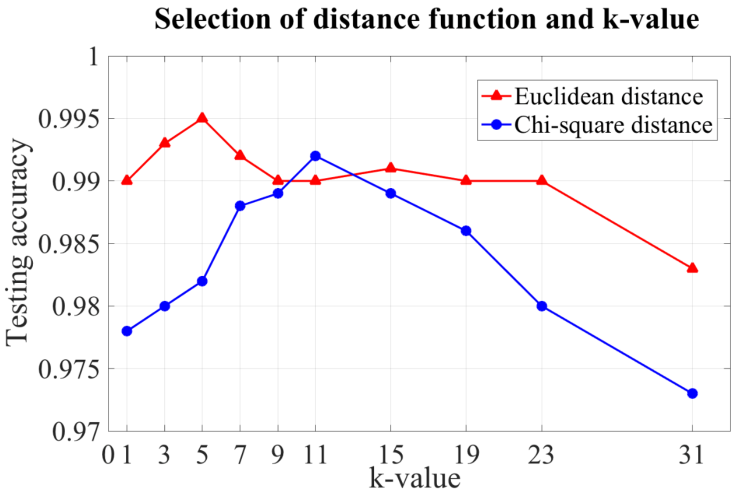


Figure 3-24: Testing accuracy for each *k*-value using Euclidean or Chi-square distances

As for the *k*-value, it normally is a positive, content-defined integer that is not larger than 20 or smaller than the square root of the number of training samples. To avoid over-fitting or a high sensitivity to noise, cross validation was conducted to find the optimal *k*-value. Based on Figure 3-24, the testing accuracy reached to its highest value when *k* was 5, using Euclidean distance. When the *k*-value was relatively larger, for example 31, the testing accuracy experienced a rapid decrease. Therefore, the *k*-value is set to 5 for the *k*-NN algorithm in this study.

In the feature extraction stage (see Chapter 3.2.1), features were extracted from the training samples in the train set and saved as MATLAB files ('mat') for the quality examination of new DXA images. After that, the hyper-parameter *k* in the *k*-NN algorithm was set according to the provided data and the extracted features in this model. With the developed model based on *k*-NN algorithm, the quality examination process for new DXA images is illustrated in Figure 3-25.

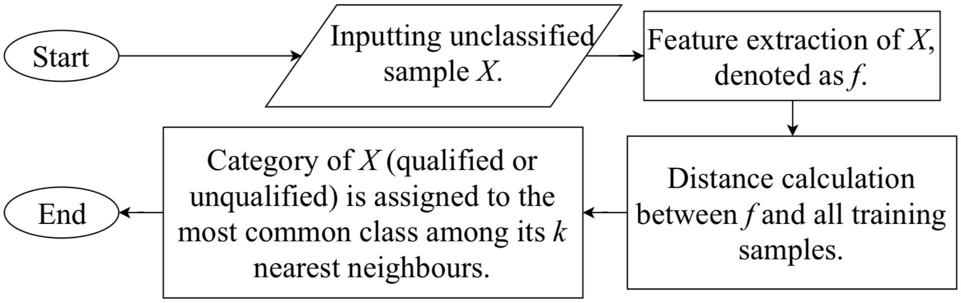

Start
Inputting unclassified sample *X*.
Feature extraction of *X*, denoted as *f*.
Distance calculation between *f* and all training samples.
Category of *X* (qualified or unqualified) is assigned to the most common class among its *k* nearest neighbours.
End

Figure 3-25: Quality examination of DXA images based on *k*-NN

3.3 Quality examination of DXA images based on the distance between neighboring points

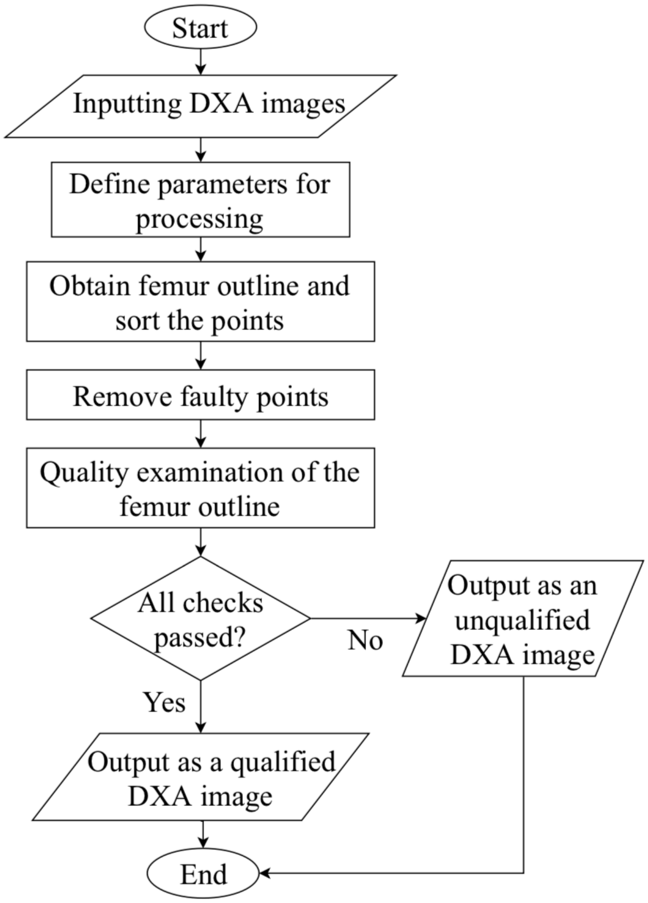

Start
Inputting DXA images
Define parameters for processing
Obtain femur outline and sort the points
Remove faulty points
Quality examination of the femur outline
All checks passed?
No
Output as an unqualified DXA image
Yes
Output as a qualified DXA image
End

Figure 3-26: Quality examination based on distance between neighboring points

With the examination algorithm based on geometric features and *k*-NN, DXA images with the aforementioned four types of defects can be identified. However, this algorithm can fail under certain circumstances. For example, if the redundant or missing parts of a femur contour is formed by straight lines, the eroded pixels cannot form any larger pixel patch, this algorithm will fail to identify the defects. As a result, another algorithm based on the distance between neighboring points was also developed in this study. Figure 3-26 illustrates the idea of this algorithm.

After inputting DXA images and defining parameters for processing, the proximal femur outline is obtained, and sorted using the nearest neighboring method. Afterwards, a loop structure is used for removing faulty points, or outliers, from the sorted outline. Firstly, a distance parameter *P* is created to represent the path length of a point trace along the femur outline. Calculation of *P* is shown in the following equation, where *x*, *y* and *i* represent the coordinates and numbering of a point.

$$P_i = P_{i-1} + \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \qquad \text{Eq. (3-10)}$$

In this loop structure, location of each point can be estimated from 5 points prior to it. For example, at the beginning of the loop, P_1 is assumed as 0, then P_2 ~P_6 can be calculated one by one. The estimated location of the 6th point is obtained by PCHIP (Piecewise Cubic Hermite Interpolating Polynomial) function, where P_6 is the query point, *P* and *x* represent the previous 5 points, as seen in the following equation.

$$x_{6\_estimated} = pchip(P, x, P_6) \qquad \text{Eq. (3-11)}$$

The distance between estimated and real locations of the 6th point is then compared with a predetermined tolerance (i.e., 0.35*P_6) to decide whether to keep or delete this point from the outline. If the distance exceeds the tolerance, this point will be determined as a faulty point and removed from the outline. An example of femur outline with faulty points can be seen in Figure 3-27. After each loop, points remained on the outline need to be resorted.

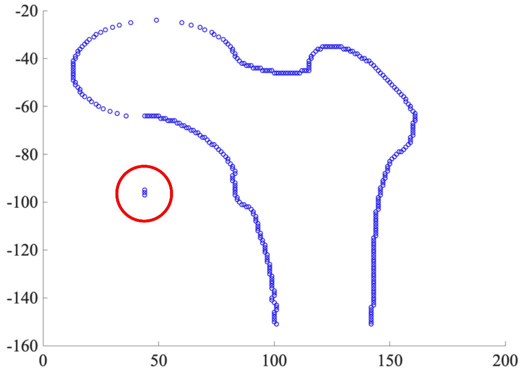


Figure 3-27: Femur outline with faulty points

Once all the faulty points are removed, the femur outline can be checked for its quality. It is aforementioned that this algorithm aims to identify defects that involves long distance between neighboring points. Therefore, the longest distance between two neighboring points on the femur outline is the feature in this algorithm. If this distance is larger than the predefined threshold, the femur outline, or the corresponding DXA image, will be identified as unqualified, such as Figure 3-28. In addition, outlines with too few left points or too many removed points are likely to have problems, such as Figure 3-29 with 44 removed points, so they are unqualified. A qualified image must pass all the checks.

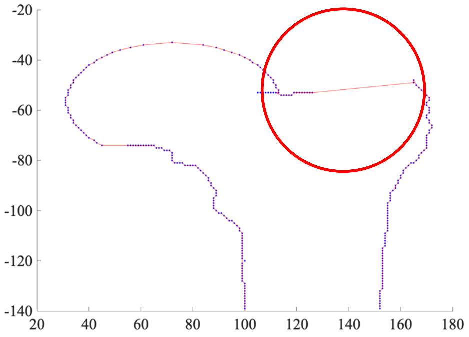


Figure 3-28: Femur outline that fails the distance check

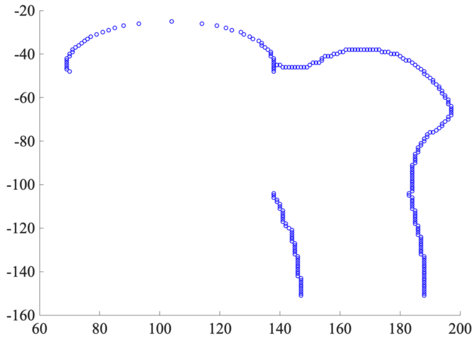


Figure 3-29: Femur outline with too many removed points or too few left points

The critical parameters in this algorithm are summarized in Table 3-2. Except for the number of points for interpolating polynomial function, other parameters are determined by statistical studies. Many femur outlines have been generated from our database of DXA images and manually divided into qualified and unqualified ones. Based on observation of these femur outlines, we have achieved appropriate values for these parameters.

Table 3-2: Critical parameters in this algorithm

Number of points for interpolating polynomial function	5
Multiplier for identifying faulty points	0.35
Maximum number of removed points	20
Minimum number of left points in femur outline	100
Threshold of distance between neighboring points	30

3.4 Comparison and combination of two quality examination algorithms

The major differences between these two algorithms are the feature for quality examination and the approaches to define or identify the features, as shown in Table 3-3. In the algorithm based on geometric features, the feature is the maximum area of pixel patches in the eroded image. Features of known cases are defined by training process, then the feature of an unknown case is identified by *k*-NN to determine its property, i.e., qualified or unqualified. In the algorithm based on distance between neighboring points, an explicit threshold of distance is predefined by statistical studies. Then, property of a new feature is directly determined by being compared with the threshold.

Table 3-3: Major differences between two algorithms

	Feature	Approach to define and identify the feature
Algorithm based on geometric features	Maximum area of pixel patches in the eroded image	Training process and *k*-NN
Algorithm based on neighboring points	Longest distance between two neighboring points	Compared with a predefined threshold value

However, utilizing a feature alone can cause errors under certain circumstances, which can be seen in Figure 3-30. For example, the algorithm using distance feature can make false positive identification when the fluctuation range of points is smaller than the threshold (see Figure 3-30 (a)). If two adjacent points on the femur outline are connected by a straight line that is larger than the threshold (see Figure 3-30 (b)), the distance feature will lead to a false negative identification. On the other hand, if the redundant part of an unqualified outline is formed by straight lines (see Figure 3-30 (c)), which have no effect to the pixels, the algorithm based on geometric features will fail.

Therefore, these two algorithms need each other as a supplementary measure for a better representative of the DXA image's quality. Accordingly, the quality examination of DXA images will be more effective if these algorithms are combined.

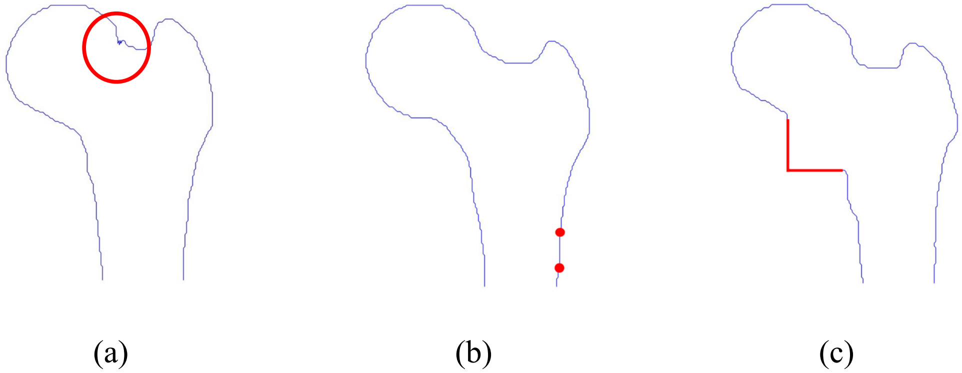


Figure 3-30: Some problematic femur contours

With these two algorithms combined, all unqualified DXA images with known defects can be identified. However, the database used to develop these algorithms has its own limitation. The number of cases is limited, and all images used in this study are obtained from scanners of the same model, which can be problematic. Future work needs to be done to investigate more images from more DXA scanners of different models.

Chapter 4

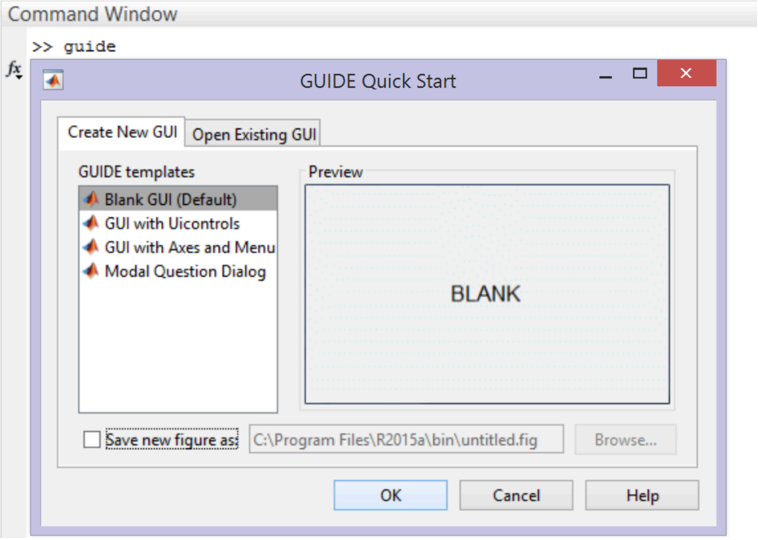
Development of a standalone program for assessment of hip fracture risk

In MATLAB, development of a standalone program, or an application, consists the following three steps: (1) design of Graphical User Interface (GUI); (2) hook of callback functions to different controls; and (3) packaging of functions and generation of executable application. The three steps are described in detail in the following sections.

4.1 Design of GUI in MATLAB GUIDE

A GUI is a graphical display that contains components that enable users to perform interactive tasks. Usually, a GUI consists of controls such as pushbuttons, sliders, menus and so forth. MATLAB GUIDE, or GUIDE, is the GUI development environment integrated within MATLAB. It provides a set of tools for creating GUIs and setting the User Interface (UI) controls regarding user interactions [61]. The process of designing and programming GUIs can be largely simplified by utilizing GUIDE.

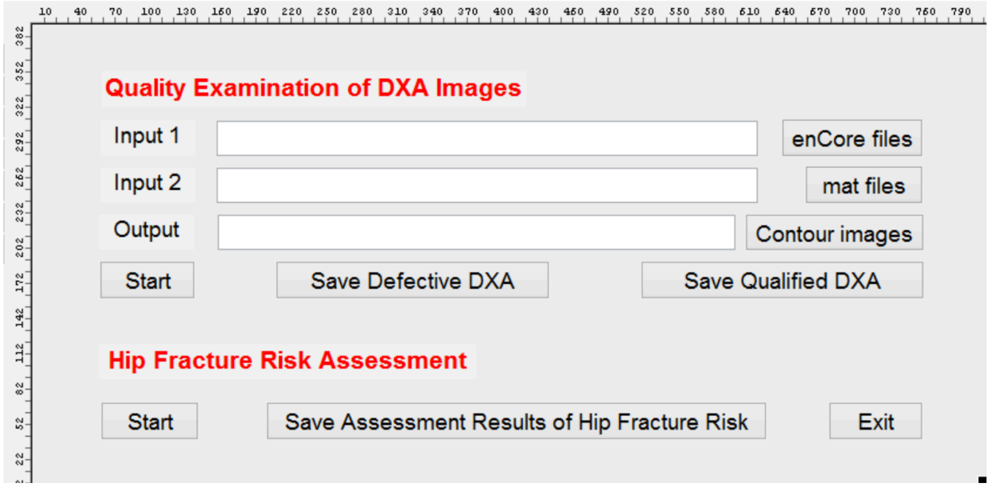
GUIDE can be started by inputting ‘guide’ in the MATLAB Command Window, as seen in Figure 4-1. In this study, the Blank GUI (Default) template was selected to create a GUI. Once the designing process is finished, the GUI can be saved as a FIG file, which contains the complete information about the GUI window and all the control functions. Meanwhile, GUIDE automatically generates the corresponding M-file skeleton to control how the GUI works.



```
>> guide
```

Save new figure as: C:\Program Files\R2015a\bin\untitled.fig

Figure 4-1: The start interface of GUIDE



Quality Examination of DXA Images
Input 1
enCore files
Input 2
mat files
Output
Contour images
Start
Save Defective DXA
Save Qualified DXA
Hip Fracture Risk Assessment
Start
Save Assessment Results of Hip Fracture Risk
Exit

Figure 4-2: The GUI designed for the standalone program

Figure 4-2 presents the GUI designed in MATLAB GUIDE. This program consists of two parts, which are the quality examination of DXA images and the hip fracture risk assessment of qualified ones.

Table 4-1: Functions of different buttons on the designed GUI

Button on the GUI	Function of the button
enCore files	Select the input directory of ‘enCore’ files
mat files	Select the input directory of ‘mat’ files
Contour images	Select the output directory of contour images
Save Defective DXA	Move defective DXA images into the assigned directory
Save Qualified DXA	Move qualified DXA images into the assigned directory
Save Assessment Results of Hip Fracture Risk	Save the result of hip fracture risk assessment into the assigned directory
Start	Run the program
Exit	Exit the program

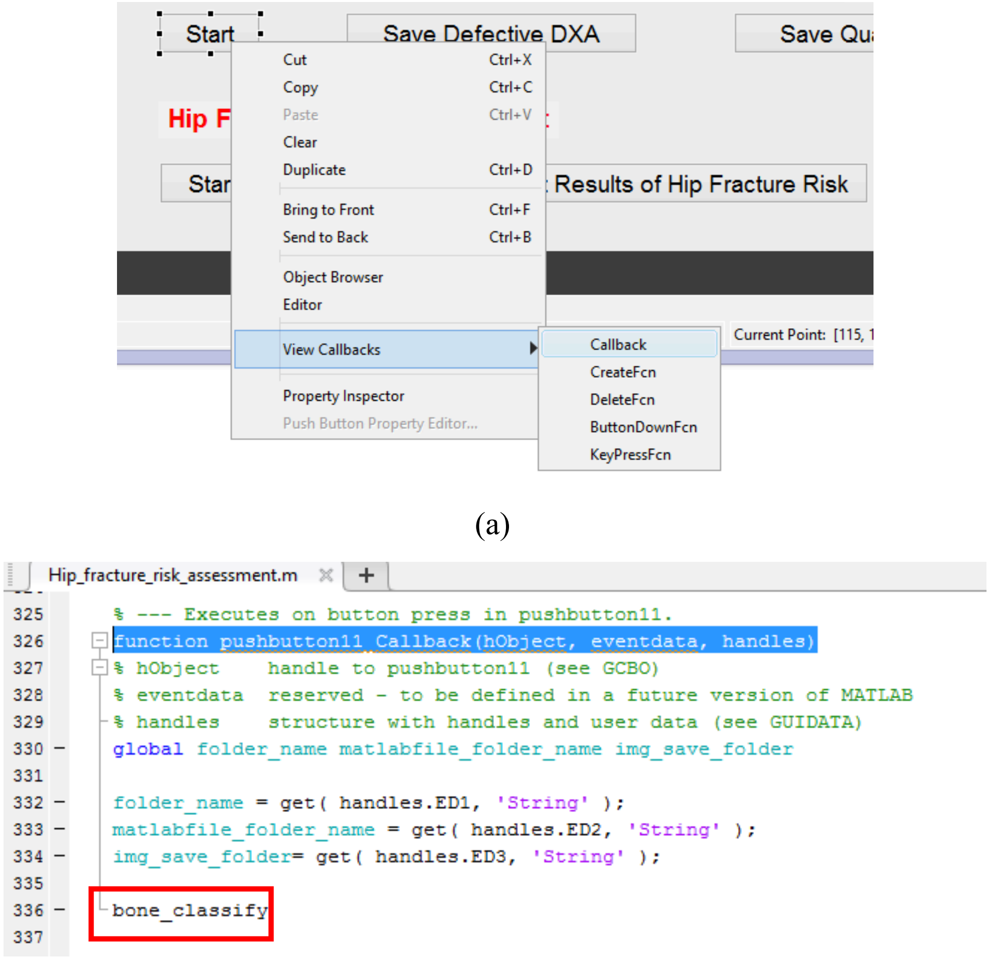
Firstly, the clinical DXA scans are read in both the native manufacturer format (‘enCore’) and as MATLAB files (‘mat’) to generate proximal femur contour images, as shown in Figure 4-2. Hence, the first part of this GUI contains two input boxes and one output box, respectively for the directories of DXA images and for outputting contour images, which are labelled as ‘Input 1’, ‘Input 2’ and ‘Output’ in Figure 4-2. Users can either type the directory paths or click on the corresponding buttons to select specific folders. After the quality examination, qualified and unqualified contour images will be put into two subfolders under the output directory.

In the second part of the GUI, which is the hip fracture risk assessment, the qualified DXA images are read as input parameters by ‘Input 1’ and ‘Input 2’. Therefore, input boxes are not required in this part. During the assessment, a progress bar (see Chapter 6.2) indicates the remaining time. In addition, designed function of each button on this GUI is explained in Table 4-1.

4.2 Hook of callback functions to different controls

After designing the purpose for each button, we need to edit their callback functions in the M-file generated along with the GUI, in order to set the corresponding UI controls. Callback functions of all buttons on the GUI are listed in the Appendix.

To explain how to hook the callback function to a control button, the 'Start' button in the first part of GUI is selected as an example. In Figure 4-3 (a), the 'Start' button is right-clicked, then 'View Callbacks' and 'Callback' are selected. Once we click on 'Callback', GUIDE Tool will automatically redirect the interface to this button's callback function in the M-file, as seen in Figure 4-3 (b). This button is designated to generate contour images and conduct quality examination for these images. In its callback function, we need to call the corresponding MATLAB file that can realize these functions, which is 'bone_classify.m'. Prior to that, 'folder_name', 'matlab_folder_name' and 'img_save_folder' are defined, since these are parameters required for running the function. During the operation of GUI, these parameters are acquired through input and output boxes. With the edited callback function, 'bone_classify.m' can be invoked with the inputted parameters by pressing the 'Start' button.



(a)

```
% --- Executes on button press in pushbutton11.
function pushbutton11_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global folder_name matlabfile_folder_name img_save_folder

folder_name = get( handles.ED1, 'String' );
matlabfile_folder_name = get( handles.ED2, 'String' );
img_save_folder= get( handles.ED3, 'String' );

bone_classify

```

(b)

Figure 4-3: Edit callback function for a button

4.3 Packaging of functions and generation of executable application

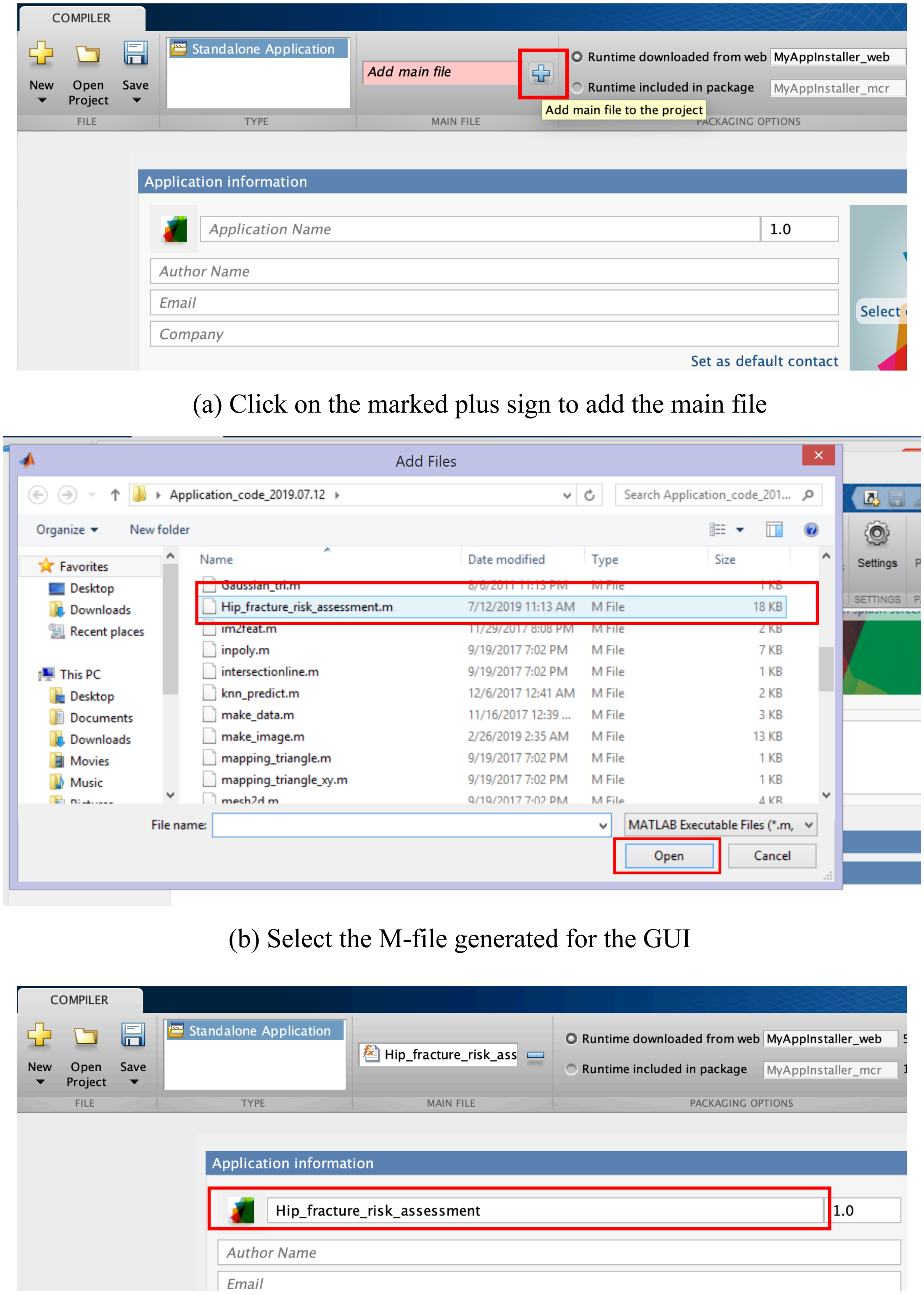
Subsequently, MATLAB Compiler was used to package the GUI and in-house MATLAB code into a standalone program. By using MATLAB Compiler, which is referred to as Application Compiler in the software interface, MATLAB programs can be shared as standalone applications (or programs) for redistribution to end users who do not need

MATLAB or have any knowledge in MATLAB. For the programs developed by MATLAB Compiler, it is essential to have MATLAB Runtime for royalty-free development [35]. MATLAB Runtime can be either packaged with the standalone program or downloaded by users during installation.

The software interface of the Application Compiler can be seen in following figures in this section. It enables customization of the related information, such as the author, splash screen, and version number. In this study, the main file of the program was the M-file generated for the GUI. Once the main file was added, all of the other files that were dependent on the main file were automatically identified and selected as ‘Files required for your application to run’. Moreover, any additional data or help files can be packaged into the final program.

MATLAB Compiler (or Application Compiler) can either be found in the Application Deployment section of MATLAB APPS gallery or be activated by inputting ‘applicationCompiler’ in the Command Window.

The first steps of packaging were picking the main file and adding it into the Compiler, as shown in Figure 4-4. The main file in this study was the M-file generated with the GUI, which was created for controlling the program. Once the main file was added, all the other files being used by the main file should be added into ‘Files required for your application to run.’ It should be noted that although MATLAB Compiler can recognize the main file and identify required files, sometimes it is possible to miss some types of files. Therefore, it is better to add the required files manually.


COMPILER
New
Open Project
Save
FILE
Standalone Application
TYPE
Add main file
MAIN FILE
Add main file to the project
Runtime downloaded from web MyAppInstaller_web
Runtime included in package MyAppInstaller_mcr
PACKAGING OPTIONS
Application information
Application Name
1.0
Author Name
Email
Company
Set as default contact
Select

(a) Click on the marked plus sign to add the main file

Add Files
Application_code_2019.07.12
Search Application_code_201...
Organize
New folder
Favorites
Desktop
Downloads
Recent places
This PC
Desktop
Documents
Downloads
Movies
Music
Name
Date modified
Type
Size
Gaussian_trl.m 8/6/2011 11:15 PM M File 1 KB
Hip_fracture_risk_assessment.m 7/12/2019 11:13 AM M File 18 KB
im2feat.m 11/29/2017 8:08 PM M File 2 KB
inpoly.m 9/19/2017 7:02 PM M File 7 KB
intersectionline.m 9/19/2017 7:02 PM M File 1 KB
knn_predict.m 12/6/2017 12:41 AM M File 2 KB
make_data.m 11/16/2017 12:39 ... M File 3 KB
make_image.m 2/26/2019 2:35 AM M File 13 KB
mapping_triangle.m 9/19/2017 7:02 PM M File 1 KB
mapping_triangle_xy.m 9/19/2017 7:02 PM M File 1 KB
mesh2d.m 9/19/2017 7:02 PM M File 4 KB
File name:
MATLAB Executable Files (*.m,
Open
Cancel
Settings
SETTINGS

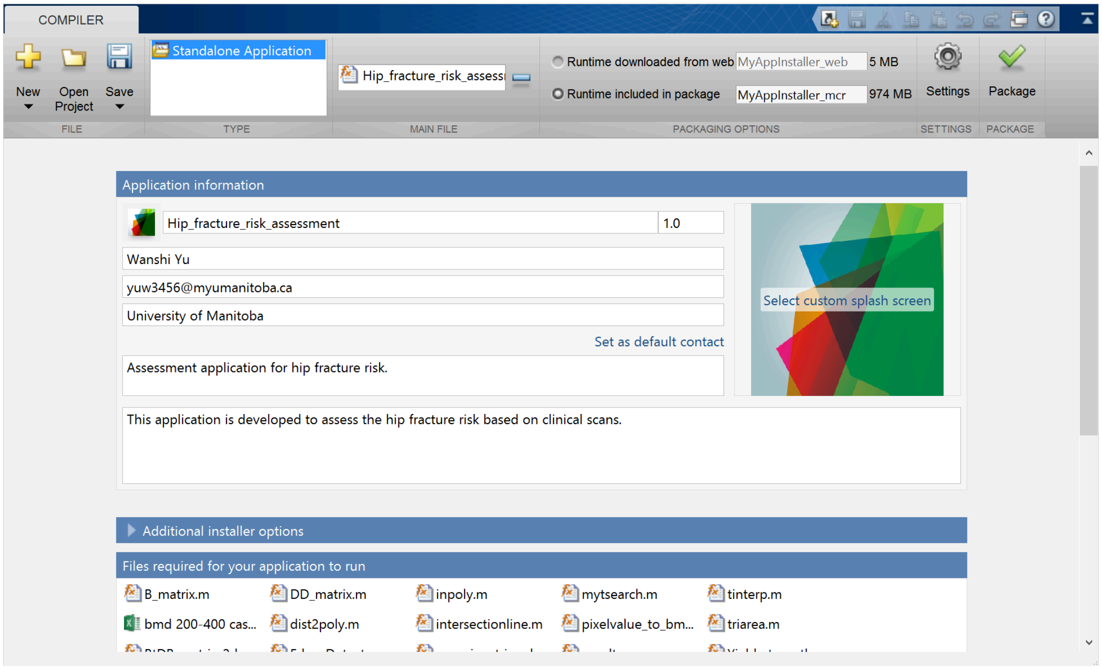
(b) Select the M-file generated for the GUI

COMPILER
New
Open Project
Save
FILE
Standalone Application
TYPE
Hip_fracture_risk_ass
MAIN FILE
Runtime downloaded from web MyAppInstaller_web
Runtime included in package MyAppInstaller_mcr
PACKAGING OPTIONS
Application information
Hip_fracture_risk_assessment
1.0
Author Name
Email

(c) The main file has been added

Figure 4-4: Add main file into MATLAB Compiler

Figure 4-5 shows the packaging process in MATLAB Compiler interface. In addition to adding required files into the Compiler, another critical task was to provide some useful information about this program, such as the name of the program, name of the author, contact information of the author, summary and so on.



○ Runtime downloaded from web MyAppInstaller_web 5 MB

● Runtime included in package MyAppInstaller_mcr 974 MB

Application information

Hip_fracture_risk_assessment 1.0

Wanshi Yu

yuw3456@myumanitoba.ca

University of Manitoba

Set as default contact

Assessment application for hip fracture risk.

This application is developed to assess the hip fracture risk based on clinical scans.

Additional installer options

Files required for your application to run

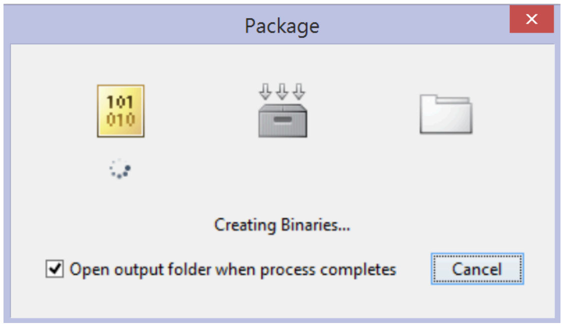
B_matrix.m, DD_matrix.m, inpoly.m, mytsearch.m, tinterp.m

bmd 200-400 cas..., dist2poly.m, intersectionline.m, pixelvalue_to_bm..., triarea.m

Figure 4-5: Packaging process of application in MATLAB Compiler

Moreover, the author can decide whether to include MATLAB Runtime in the package or not. In 'PACKAGING OPTIONS', if 'Runtime included in package' is selected, the size of the program will be larger. On the other side, if 'Runtime downloaded from web' is selected, users will need to download MATLAB Runtime when they start the program for the first time.

The final step was to click on the 'Package' button on the interface to initiate the packaging process, as seen in Figure 4-6. Once the process was finished, the packaged program would be saved as an executable installer file ('exe') under the selected output directory. Users or clients can use this installer file instead of the in-house MATLAB code. As a result, the in-house MATLAB code can be strongly encrypted, and the intellectual property will be better protected.



Creating Binaries...

☑ Open output folder when process completes

Figure 4-6: Interface during the packaging process

Chapter 5

A preliminary framework of web-based biomechanical tool for assessment of hip fracture risk

In this chapter, the framework of a web-based biomechanical tool is proposed for the clinical assessment of hip fracture risk. It includes the quality examination of DXA images based on the distance between neighboring points and the subject-specific hip fracture risk assessment.

5.1 Fundamental structure of the web-based tool

In the web-based tool developed in this study, the assessment of hip fracture risk is completed by the in-house MATLAB codes. Therefore, the main challenge in the developing process is how to invoke the MATLAB code following user's input at the front-end (webpage, or the HTML file). Instead of being embedded in the back-end, the MATLAB code is packaged into an executable file, which is an Executable and Linkable Format (ELF) file, and invoked by the Linux shell script. Before that, the shell script is initiated by the back-end through the Common Gateway Interface (CGI). In this case, the ELF file can be isolated from the back-end, which is more convenient for any future modification of MATLAB code.

The structure, or the data flow, of this web-based tool is shown in Figure 5-1. It is hosted on the Apache server in the Linux system and could be accessed externally. Firstly, user's inputs are collected at the front-end by HTML forms. The Linux shell script and the ELF file are then invoked correspondingly. When the operation of the ELF file is finished, the assessment results are saved in a shared folder that can be accessed by the user through Samba.

It should be noted that Apache and Samba are both free software, which can greatly reduce the cost of this study. Meanwhile, they have been actively maintained with updated features. In addition, the programming languages used in this study are all open-sourced and have high flexibility. The detailed introduction of the software and programming languages are presented in the following sections of this chapter.

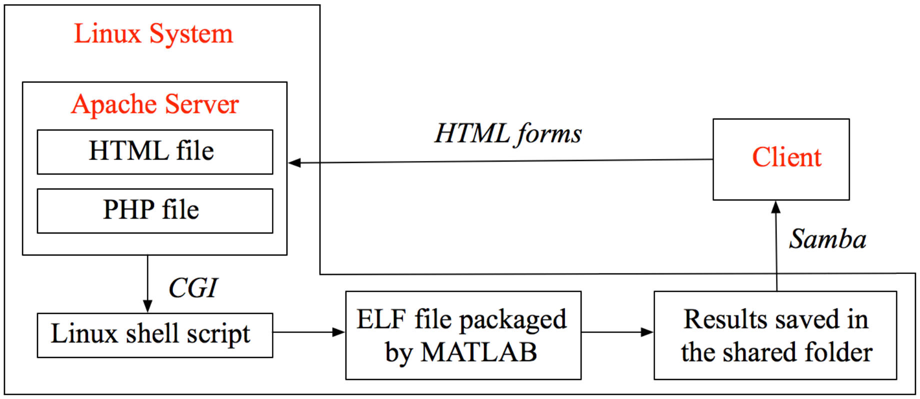


Figure 5-1: Structure of the web-based biomechanical tool developed in this study

5.2 Introduction to Apache and PHP

For the web-based biomechanical tool, Ubuntu 16.04.4 LTS, which is the release of the Linux system, was installed in VMware [62] as the development system. Additionally, MATLAB 2015b, Apache 2.4.18 and Hypertext Preprocessor (PHP) were installed in the Linux system.

As an open source software developed by the Apache Software Foundation and launched in 1995, Apache HTTP Server, which is usually called Apache, has been the most widely used web server software since 1996 [63]. Based on a research report from the Security Space [64], most of the Apache servers (over 90%) are installed in Linux at the moment. Apache is also available on other operating systems, such as Mac OS or Windows [65]. With the installation of Apache, web servers can be created to host one or more Hypertext Transfer Protocol (HTTP)-based websites. In other words, Apache, or any other web server, can be considered as the environment for the developed website to run in. There are many benefits from using Apache instead of other web servers since it is free for both personal and commercial uses [63]. Meanwhile, it has been actively maintained and kept being updated with new features and modules, which make Apache one of the most feature-filled web servers.

The server-side programming language of this web-based tool is PHP, which is also required to be manually installed in the Linux system. PHP is a free and widely-used scripting language that is executed on the server [66]. In web programming, the primary functions of PHP are to process the client's request in the engine and create a corresponding response [67], as shown in Figure 5-2.

PHP can run on different operating systems, including Windows, Linux, and Mac OS, and support many databases [66]. It is compatible with almost all recognized servers, including Apache, which is the one used in this study. Additionally, the cost of this project can be largely reduced since PHP is an open source language.

In this study, PHP is used for performing calculations, or the hip fracture risk assessment, with the data or information collected from the HTML page.

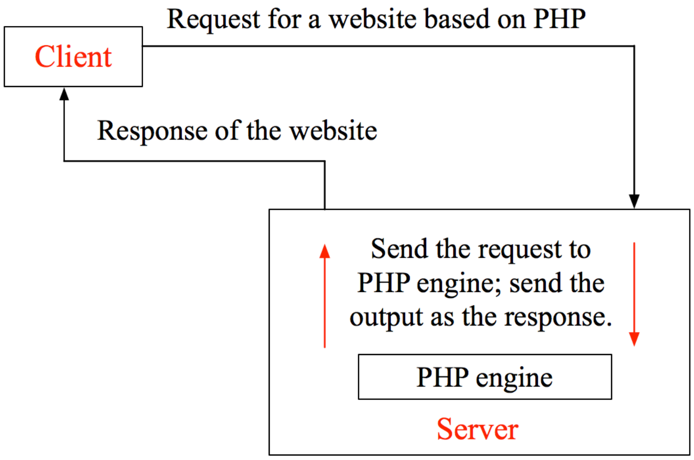

Request for a website based on PHP
Client
Response of the website
Send the request to PHP engine; send the output as the response.
PHP engine
Server

Figure 5-2: The function of PHP in web programming

5.3 Linux shell script

A Linux shell script is a computer program designed to be run by the Linux shell [68]. The Linux shell is a command-line interpreter, where shell refers to the UI of the system. The shell can provide an interface for the interactive operations between users and the kernel of the operating system. By using a shell, inputting commands from users can be translated into binary commands, or source code, that can be identified and processed by the kernel of the computer. A shell script is similar to programming languages since they all include variables and Control Flow Statements [68]. However, a shell script does not need to be compiled.

Along with the Linux system, there are many different versions of Linux shell, such as Bourne Shell, BASH (Bourne Again Shell), Korn Shell and C Shell [69]. In this study, BASH was selected. It was written by Brian Fox for the GNU Project as a free software replacement for the Bourne shell and first released in 1989. Nowadays, BASH has become the default shell for many Linux system packages due to its enhanced functionality and features [70]. The '/bin/sh' in a Linux system is usually the symbolic link towards '/bin/bash.'

The first step of developing the Linux shell script in this study is to package the existing in-house MATLAB code into an executable file (or an ELF file) by using the Application Compiler. Except for not including a GUI written by MATLAB, the steps of using the Application Compiler in this chapter are similar to those in Chapter 4. In addition, the MATLAB program needs specific modification to be running on the Linux system. In the main MATLAB function, four directories are set as four parameters, which are the input directories of DXA images, output directory of contour images and the storage directory of the assessment results. These parameters are also the inputs of the front-end (see Figure 5-4). After that, the packaged executable file is tested for its feasibility in the terminal of Linux system.

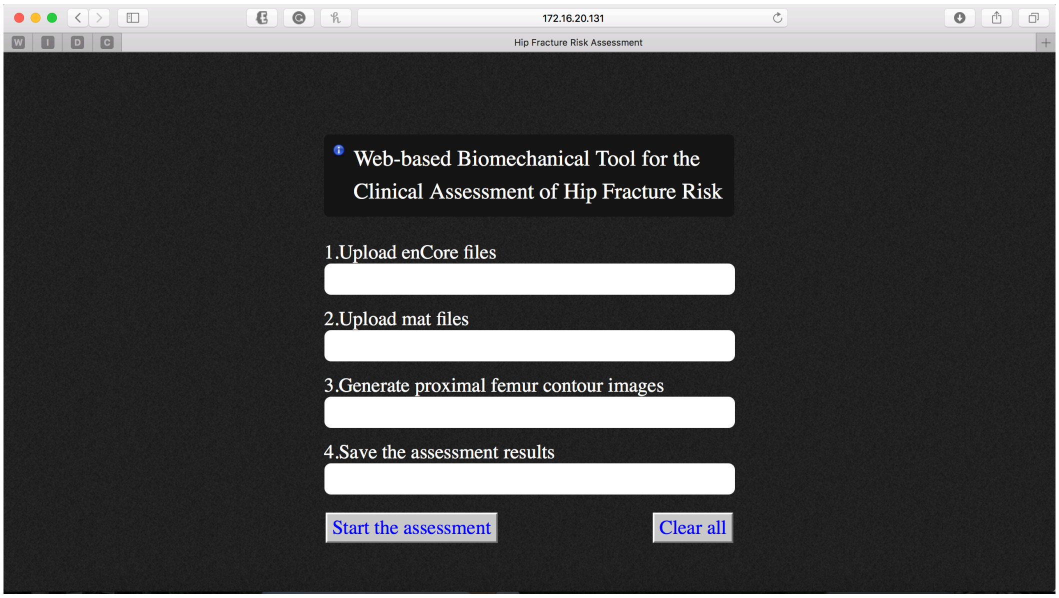
Subsequently, the Linux shell script is required for setting the environmental variables. By running on a shell, shell scripts can perform various operations, including environment setting-up, file manipulation, and program execution. In this study, the primary function of the Linux shell scripting is to set the environmental variables to realize the transmission of the parameters defined by the MATLAB program. The batch mode of the shell script is selected instead of the interactive mode, which means that the shell script has been written in advance to allow the commands in the script to be executed at one time. In the Linux shell script, the shell is declared at first. Afterwards, the environmental variables are defined and set up using the 'if…else…' structure. The four directories used for assessing hip fracture risks are then set as 'LIBRARY_PATH' in order to be transmitted as the inputs or outputs.

5.4 Construction of the front- and back-ends of the web-based tool

In this study, HTML and PHP are the programming languages of the front- and back-ends of this web-based tool, respectively. The constructed web-based tool can be hosted on the Apache HTTP server.

HTML describes the structure of web pages using markup. It is the standard markup language for creating web pages and web applications [71]. Web browsers can receive HTML documents from the web server and render the documents into web pages. HTML allows users to create multiple types of contents on a web page, such as headings, paragraphs, images or input fields. To further specify the appearance of the created contents on the web page, Cascading Style Sheets (CSS) need to be added. The CSS style sheets of a web page are usually separate from the HTML file. The initial release of HTML was in 1993 [72]. The latest version, HTML 5.3, was published on April 26, 2018 [73]. In the past 25 years, HTML has been continuously updated, except for the early and mid-2000s, as the development of Extensible Hypertext Markup Language (XHTML) occupied the W3C's HTML Working Group [74].

One of the most critical parts of the HTML code in this study is the HTML form, which is used to collect user’s inputs on the developed web page[71]. In this case, the inputs are the four directories of input and output files that are required for running the MATLAB program for hip fracture risk assessment. HTML form is a section containing form elements, or various types of input elements, including text fields, checkboxes and submit buttons [75]. In this study, input elements are used for defining the one-line text input fields that can allow users to input the directories. Figure 5-3 presents the full screenshot of the web-based tool’s interface, with the four white input fields being created with HTML forms. As seen in Figure 5-4, titles of web page and input fields are developed with HTML headings. A start button is then added to the interface for initiating the assessment process.



Web-based Biomechanical Tool for the Clinical Assessment of Hip Fracture Risk

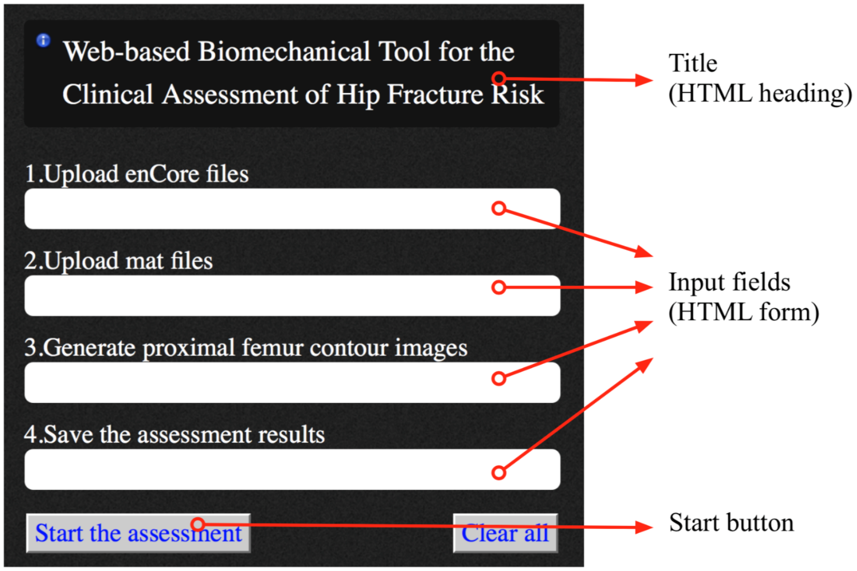
1.Upload enCore files

2.Upload mat files

3.Generate proximal femur contour images

4.Save the assessment results

Figure 5-3: Interface of the developed web-based tool



Web-based Biomechanical Tool for the Clinical Assessment of Hip Fracture Risk

1.Upload enCore files

2.Upload mat files

3.Generate proximal femur contour images

4.Save the assessment results

Figure 5-4: Layout of the web-based tool's interface

During the process of hip fracture risk assessment, or data requesting, a waiting interface is added by Asynchronous JavaScript And XML (AJAX) [76], as shown in Figure 5-6. AJAX is not a programming language, but a new technology for the improvement of web applications. When the assessment using Linux shell script finishes, there will be a notification message, which is shown in Figure 5-7.

Back-end of the web-based tool is developed with PHP, where CGI is used for invoking the Linux shell script. CGI refers to the interfacing standards between web servers and external programs (CGI programs), which can offer a standard protocol for information transmission between the server and the CGI program [77]. CGI was initially developed by the National Center for Supercomputing Applications (NCSA) team for the NCSA HTTPd Web servers in 1993 [78], and it has been a standard for web servers since then. The basic process of using CGI in this study is described in Figure 5-5, where CGI can be considered as a remote execution of the shell script. Additionally, the related parameters are obtained through the Shell environment variables.

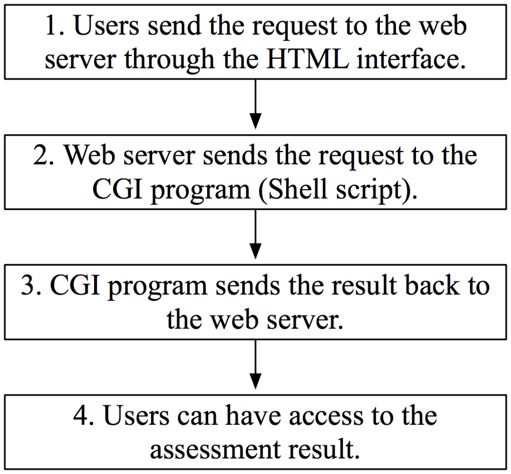

1. Users send the request to the web server through the HTML interface.
2. Web server sends the request to the CGI program (Shell script).
3. CGI program sends the result back to the web server.
4. Users can have access to the assessment result.

Figure 5-5: The process of using CGI in this project

5.5 Accessing the assessment results through Samba

Samba is a free software that can realize many networking protocols in Unix/Linux systems. For example, one of the most common protocols is Server Messages Block (SMB), which is a communication protocol for sharing files or printers between computers in the Local Area Network (LAN) [79]. In December 1991 and January 1992, the first version of Samba was developed [80]. Since then, Samba has been continuously improved for supporting new systems or features.

Samba can set up network shares for selected directories in Unix/Linux systems. By using Samba, which can be regarded as a file sharing scheme, users can have access to the shared files, printers or other resources located on a server that is in the same LAN. In this study, assessment results are generated in the Linux system by the web-based tool hosted on the

Apache server. Samba is used for accessing these results from the main computer system, which is Mac OS.

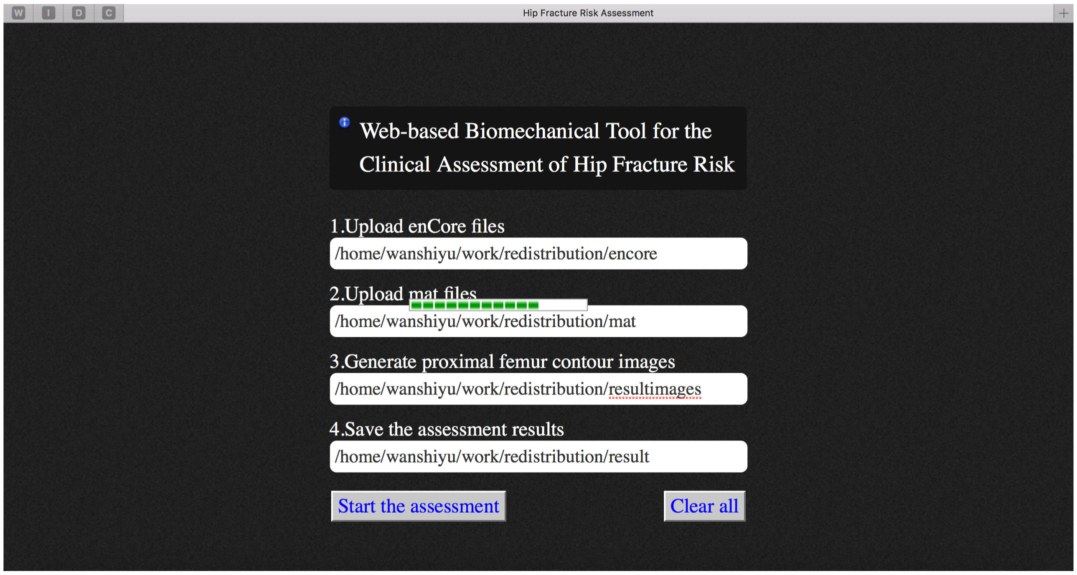
To realize the file sharing between Linux and Mac OS, Samba was installed in the Linux system through the terminal. A shared path was also created for saving the results of hip fracture risk assessment. The permission of this shared path was then set by executing the 'chmod' command. Afterwards, the configuration file of Samba ('samba.conf') needed to be modified based on the requirement of this study. In addition, the Samba account was required to be added in Linux. With the username and password of the Samba account, the files under the shared path are accessible and modifiable from the main system by inputting the server address into a web browser.

5.6 Operation process of the web-based tool

As mentioned before, the framework of this web-based tool is hosted on a local server in Linux system. Users can have access to the web page and the assessment results through Samba. Different from the standalone program, there is no need for installing any software or environment to assess hip fracture risks using this web-based tool.

The web page can be started by inputting its IP address into any web browser in the main system. As shown in Figure 5-6, the input directories of DXA images and output directories of contour images and assessment results need to be entered on the web page manually. Afterwards, users can click on the 'Start assessment' button to initiate the operation for the DXA images under the input paths. The 'Clear' button is used to empty all the input boxes on this web page. There is a progress bar during the assessment, as shown in Figure 5-6.

Once the assessment is finished, there will be a notification message to inform the user, with the assessment results saved under the designated directory, as shown in Figure 5-7. Subsequently, users can have access to the results through Samba from any computer that is in the same LAN. In the web-based tool developed in this study, the assessment results are saved as a text file, which can be transformed into an Excel file later.



Hip Fracture Risk Assessment

Web-based Biomechanical Tool for the Clinical Assessment of Hip Fracture Risk

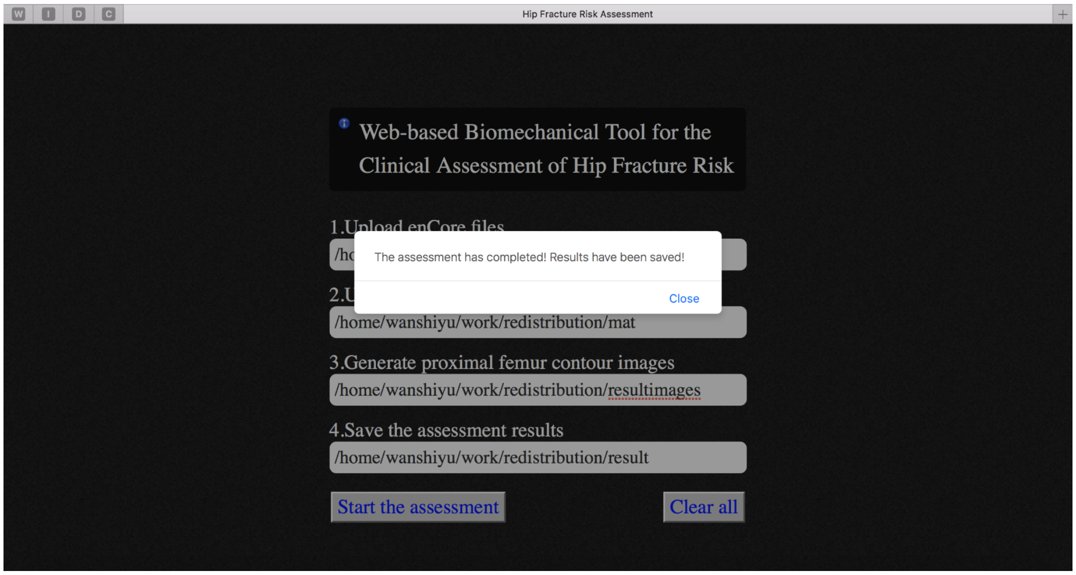
1.Upload enCore files
/home/wanshiyu/work/redistribution/encore

2.Upload mat files
/home/wanshiyu/work/redistribution/mat

3.Generate proximal femur contour images
/home/wanshiyu/work/redistribution/resultimages

4.Save the assessment results
/home/wanshiyu/work/redistribution/result

Figure 5-6: Inputting the directories and the waiting interface



Hip Fracture Risk Assessment

Web-based Biomechanical Tool for the Clinical Assessment of Hip Fracture Risk

1.Upload enCore files
/hc

2.U
/home/wanshiyu/work/redistribution/mat

3.Generate proximal femur contour images
/home/wanshiyu/work/redistribution/resultimages

4.Save the assessment results
/home/wanshiyu/work/redistribution/result

The assessment has completed! Results have been saved!

Figure 5-7: The notification message indicating that the assessment is finished

Chapter 6

Results and discussion

6.1 Validation of quality examination algorithms of DXA images

In this section, there will be some discussion on the division of datasets and convergence test of *k*-NN algorithm. Then, the quality examination algorithms of DXA images proposed in this study will be validated and compared using different performance measures.

6.1.1 Division of datasets in the *k*-NN algorithm

To develop a *k*-NN algorithm, the first step is always the division or selection of datasets. The division of datasets in this study is summarized in Figure 6-1. The provided data (DXA images) was first divided into two datasets, the train set and the test set, which were used for constructing and validating. The test set was independent of the train set but with the same or similar probability distribution. In other words, the ratio between the quantities of qualified (or positive) and unqualified (or negative) samples should be the same or similar in these two datasets. Afterwards, the train set in the first division was further divided into a train set and a validation set. This train set was used for the actual training, while the validation set was used for tuning the hyper-parameters during the model-constructing stage. It should be noted that the divided datasets cannot have any intersection.

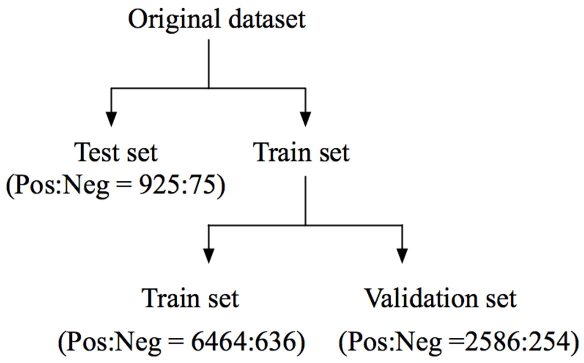


Figure 6-1: A visualization of the division of datasets

Typically, if a model does not include any hyper-parameter or is easy to be tuned, there is no need for a validation set. For the *k*-NN algorithm in this study, the setting of parameters was based on the testing accuracy using the validation set. The performance of a model using the validation set can be considered as a signal to inform whether it can fulfill the requirement of accuracy, while the test set is only used for assessing this model.

There are two critical ratios in the division of datasets, which are the ratio between the train set (the one at the second level) and validation set, and the ratio between positive and negative samples. According to Figure 6-1, those two ratios are approximately 2.5:1 and 10:1. Generally, the division of datasets mainly depends on the total number of provided samples and the actual model that need to be developed. The ratio of positive to negative samples is dependent on the condition of provided dataset, such as the size or the proportion. There is no fixed rule on how to divide the datasets. Hence, these ratios can be regarded as model- and original dataset-specific.

In the original dataset provided for this study, the ratio between qualified and unqualified DXA images was around 12:1, which meant that there was a limited quantity of negative samples. To improve the performance of developed algorithm, most of the provided DXA images were put into the train and validation sets. Moreover, the ratio between positive and negative samples in train and validation sets was 10:1 instead of 12:1 to avoid imbalance. For the test set, this ratio remained to be around 12:1 to simulate the actual distribution of DXA images in the clinical practice.

6.1.2 Convergence test of the *k*-NN algorithm

Before validation, convergence test was conducted to find out the optimal size of train set that can meet the requirements for classification accuracy and computational cost. The convergence rate of classification accuracy was the main subject, along with observation of testing time.

In the convergence test, train sets with different sizes, consisting of 20, 30, 50, 100, 150, 200, 300, 400, and 500 samples, were selected. In these train sets, the ratio between qualified and unqualified contour images was 9:1. After training, the *k*-NN algorithm was tested using the same test set of 508 samples (positive: negative=1:1). After each time of training and testing, the testing accuracy and testing time were recorded and plotted. It should be clarified that the test set in this section was specifically selected for the convergence test, which was different from the test set used for validation.

The variation of testing accuracy and testing time are shown in Figure 6-2 and Figure 6-3. It can be observed that the classification accuracy reached its highest value (0.9862) and converged when the size of train set was 200. Besides, the increase of testing time was approximately linear with increasing size of train set.

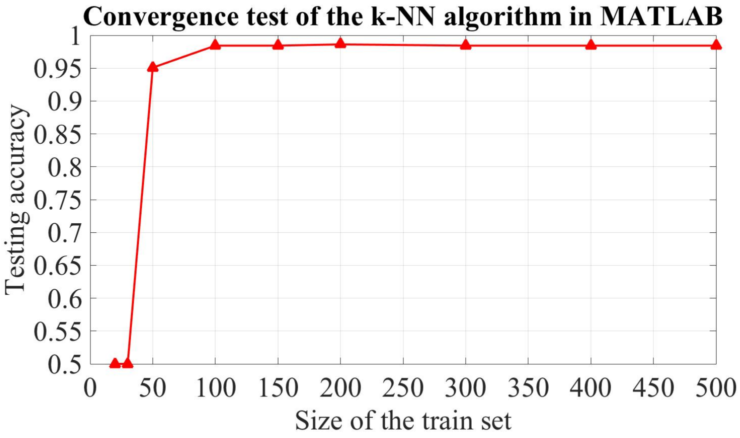


Figure 6-2: Convergence of testing accuracy of the *k*-NN algorithm

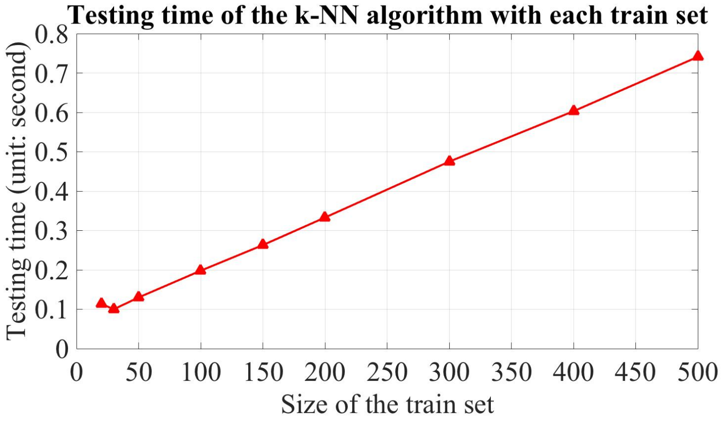


Figure 6-3: Testing time of the *k*-NN algorithm

6.1.3 Performance measures used for the validation

After development process, the two quality examination algorithms need to be validated or evaluated based on their performance. In these algorithms, DXA images are classified as either qualified or unqualified cases. Therefore, DXA quality examination can be considered as a binary classification problem, similar to a ‘pass or fail’ test.

For a more straightforward view, a binary classification problem can be represented by the contingency table [81] shown in Table 6-1. It has two dimensions- the actual value and classified value. The accuracy of a binary classification is measured by the number of correct predictions divided by the total number of predictions [82], and it is computed by the sum of True Positives and True Negatives divided by the total predictions. Accuracy is an important metric, but it is not enough for evaluating a binary classifier.

Based on Table 6-1, the main problem in this model is the existence of False Positives since they will cause errors in the FEA procedures. Therefore, if the quantity of False Positives is relatively high, even the accuracy is acceptable, this model cannot be applied to a more general application. The second problem is accuracy paradox [83], which can occur when there is a large class imbalance between positive and negative classes [84]. For example, if 98% of the provided cases are positive, then predicting every case is positive will achieve an accuracy of 98%. In this case, even a high accuracy is achieved, every unqualified case would have been missed and brought errors in FEA. Hence, other performance measures are required for validating the proposed quality examination algorithms in addition to the classification accuracy.

Table 6-1: Contingency table of a binary classification problem

		Actual value	
		Positive	Negative
Classified value	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Under these circumstances, precision and recall were also selected for the validation. The precision, or Positive Predictive Value (PPV), is defined by the number of True Positives divided by the total of True Positives and False Positives. The recall is the number of True Positives divided by the total of True Positives and False Negatives [85]. Compared with other popular performance measures, such as the Receiver Operating Characteristics (ROC) plots, precision and recall can provide a more accurate evaluation of the binary classifiers with imbalanced datasets [86]. The precision represents the ratio between positive predictions and the classified positive ones, while the recall can be regarded as the ratio between positive predictions and the actual positive samples in the test dataset. A low precision or recall can indicate a large number of False Positives or False Negatives, separately.

As a result, the performance measures used for validating quality examination algorithms of DXA images are shown in Table 6-2. Compared with using a single measure, combining all the measures can better evaluate the classification ability of the proposed algorithms [87]. Additionally, the contingency tables of the two algorithms were also used for comparison.

Table 6-2: Performance measures used for the validation

	Definition	Meaning
Classification accuracy	$\frac{\text{Correct identifications}}{\text{Total identifications}}$	Accuracy of the classification given a random sample.
Precision	$\frac{\text{True Positives}}{\text{True Positives + False Positives}}$	Exactness of the model. Lower precision means more False Positives.
Recall	$\frac{\text{True Positives}}{\text{True Positives + False Negatives}}$	Sensitivity or completeness of the model.

6.1.4 Validation and comparison of the proposed algorithms

A test set of 1000 samples (see Figure 6-1) was used to validate the two quality examination algorithms and obtained the results given in Table 6-3 and Table 6-4. From the contingency tables, it is obvious that the second algorithm has shown a high value of False Positives, which will cause errors in the following FEA procedures.

Table 6-3: Contingency table of the algorithm based on geometric features

	Actual Positive	Actual Negative
Predicted Positive	920	5
Predicted Negative	5	70

Table 6-4: Contingency table of the algorithm based on distance between neighboring points

	Actual Positive	Actual Negative
Predicted Positive	923	69
Predicted Negative	2	6

The two algorithms were compared based on consumed time and different performance measures (see Figure 6-4 and Figure 6-5). It should be noted that the consumed time refers to the total time for feature extraction and quality examination. In terms of accuracy, precision and consumed time, the algorithm based on geometric features showed a better performance. However, the recall value of the algorithm based on distance between neighboring points was slightly higher.

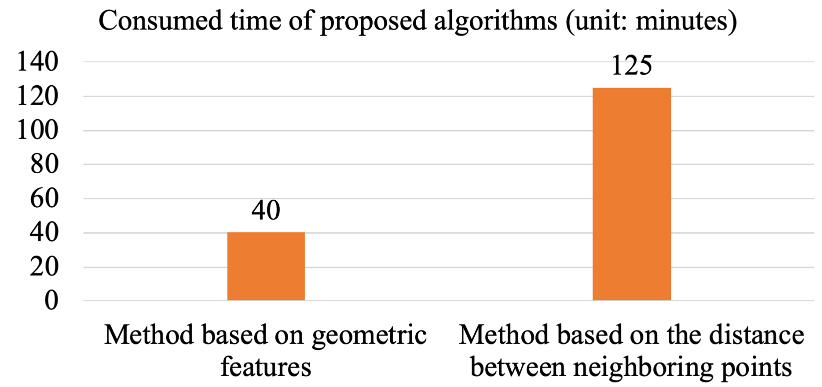


Figure 6-4: Consumed time of proposed quality examination algorithms

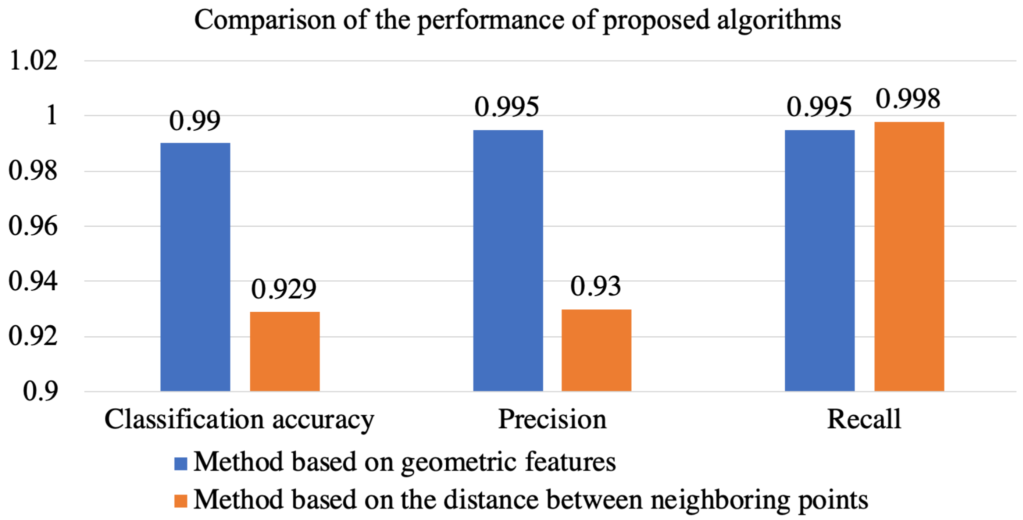


Figure 6-5: Detailed comparison based on performance measures

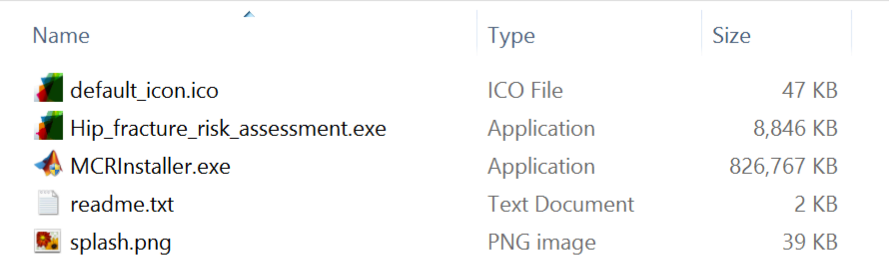
6.2 Validation of the standalone program for hip fracture risk assessment

For validation purpose, this program has been tested on several computers with different versions of Windows operating system and obtained good results. The versions of system did not show any influence on the installation or operation of this standalone program.

However, this program needs to be run as administrator, since it moves files and generates new files during the operation.

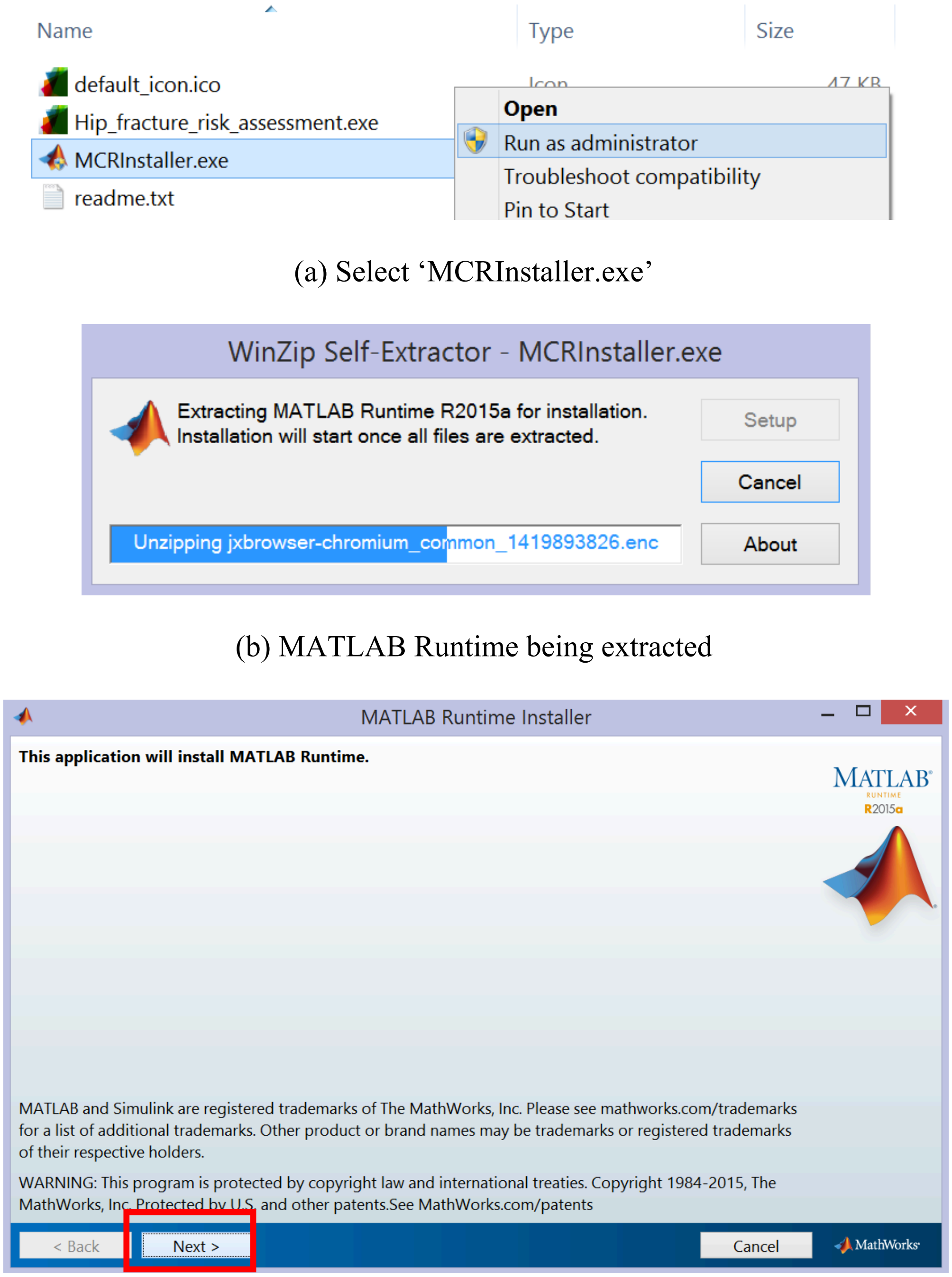
Figure 6-6 presents the files in the output folder for redistribution, including the standalone program (‘Hip_fracture_risk_assessment.exe’) and ‘MCRInstaller.exe’. Firstly, ‘MCRInstaller.exe’ was initiated for installing MATLAB Runtime, as shown in Figure 6-7. Same as MATLAB software, MATLAB Runtime has different versions. To ensure a proper operation environment, MATLAB Runtime has to be compatible with the original MATLAB software used for packaging. In this study, the standalone program was packaged with MATLAB 2015a, hence it needs MATLAB Runtime 2015a as its operation environment. Therefore, ‘MCRInstaller.exe’ was manually included in the output folder for a more convenient installation.

After installing MATLAB Runtime, the standalone program could be started by clicking on ‘Hip_fracture_risk_assessment.exe’. The GUI of this program is the same as the design in Figure 4-2. It should be noted that this program does not need to be installed, it can be directly opened from the executable file.



Name	Type	Size
default_icon.ico	ICO File	47 KB
Hip_fracture_risk_assessment.exe	Application	8,846 KB
MCRInstaller.exe	Application	826,767 KB
readme.txt	Text Document	2 KB
splash.png	PNG image	39 KB

Figure 6-6: Output folder for redistribution



(a) Select 'MCRInstaller.exe'

Extracting MATLAB Runtime R2015a for installation.
Installation will start once all files are extracted.

Unzipping jxbrowser-chromium_common_1419893826.enc

(b) MATLAB Runtime being extracted

This application will install MATLAB Runtime.

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. Please see mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

WARNING: This program is protected by copyright law and international treaties. Copyright 1984-2015, The MathWorks, Inc. Protected by U.S. and other patents.See MathWorks.com/patents

(c) Interface of MATLAB Runtime installer

Figure 6-7: Installation of MATLAB Runtime

In this section, 100 DXA images (100 pairs of enCore and MATLAB files) were randomly selected to validate the feasibility of applying this program to clinical assessment of hip fracture risks. With the required directories ('Input 1', 'Input 2' and 'Output'), femur contour images were generated under the specified directory path by clicking the 'Start' button in the first part of this program. The quality examination consisted of two steps, as mentioned in Chapter 3.4. In each step, the segmented contour (Figure 6-8) was displayed on the computer screen once it had been generated. Afterwards, the quality examination would start automatically based on the coordinating contour images. During this process, contour images were classified into two sub-folders ('GoodImages' and 'BadImages') under the designated output directory. When the quality examination was finished, a message would be prompted to inform users that they could save the results, as shown in Figure 6-9. If the output data exceeds the storage limit of Excel, the results will be saved as a text file.

With classified contour images, the second part of this program (Hip Fracture Risk Assessment) could only process the DXA images that had corresponding contour images in the 'GoodImages' folder. As shown in Figure 6-10, a progress bar and a notification message were also included in this part.

Subsequently, the assessment results were exported into an Excel file (see Figure 6-11) under the selected directory. In the output file, the information and different results of the qualified DXA images are listed following the storage order.

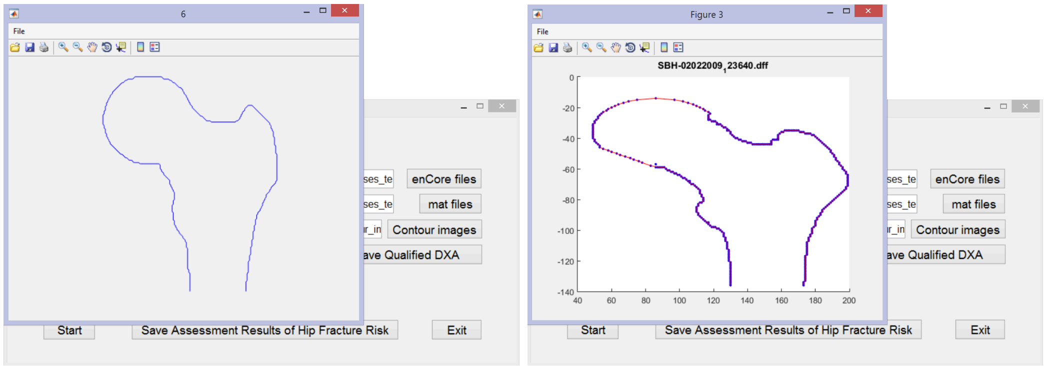
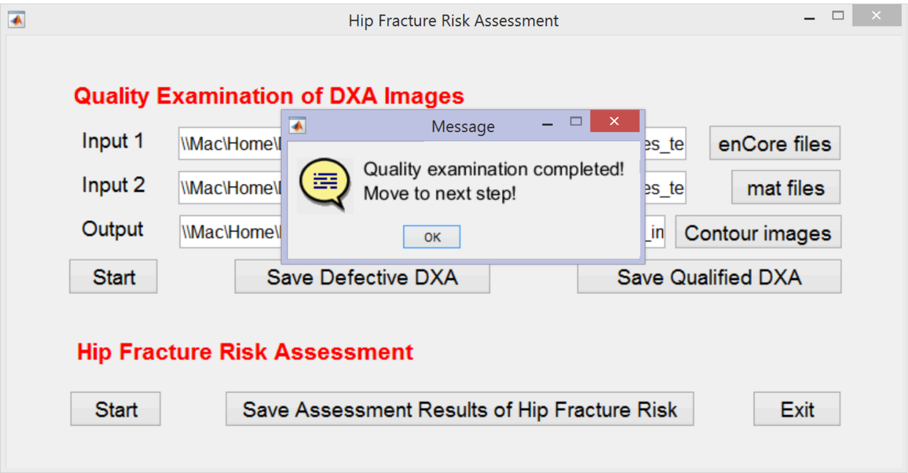
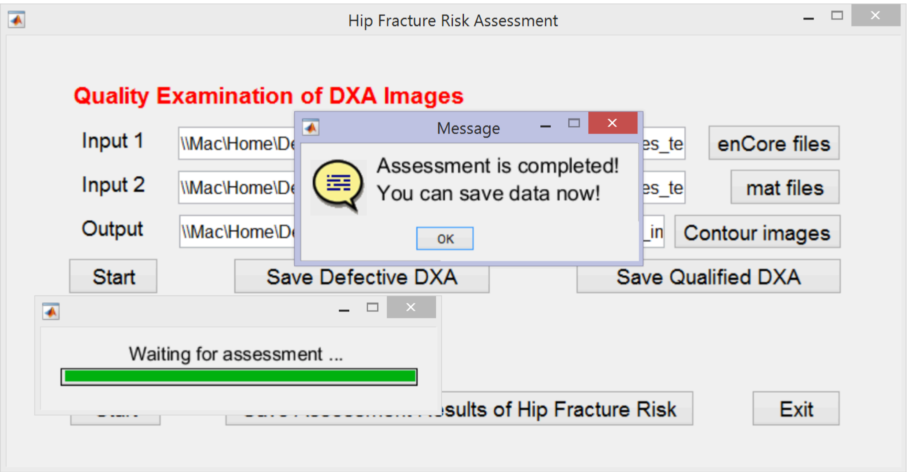


Figure 6-8: Generated contour images being displayed on the computer screen


Hip Fracture Risk Assessment
Quality Examination of DXA Images
Input 1
Input 2
Output
\\Mac\Home\
es_te
_in
enCore files
mat files
Contour images
Start
Save Defective DXA
Save Qualified DXA
Hip Fracture Risk Assessment
Start
Save Assessment Results of Hip Fracture Risk
Exit
Message
OK

Quality examination completed!
Move to next step!

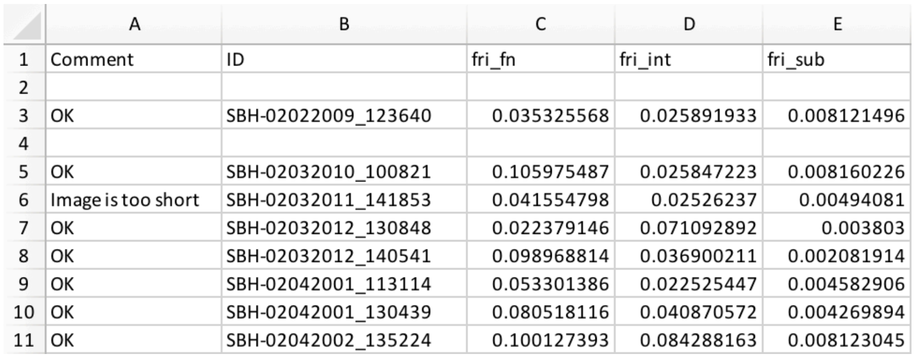
Figure 6-9: Notification message for quality examination of DXA images



Assessment is completed!
You can save data now!

Waiting for assessment ...

Figure 6-10: Notification message for hip fracture risk assessment



	A	B	C	D	E
1	Comment	ID	fri_fn	fri_int	fri_sub
2					
3	OK	SBH-02022009_123640	0.035325568	0.025891933	0.008121496
4					
5	OK	SBH-02032010_100821	0.105975487	0.025847223	0.008160226
6	Image is too short	SBH-02032011_141853	0.041554798	0.02526237	0.00494081
7	OK	SBH-02032012_130848	0.022379146	0.071092892	0.003803
8	OK	SBH-02032012_140541	0.098968814	0.036900211	0.002081914
9	OK	SBH-02042001_113114	0.053301386	0.022525447	0.004582906
10	OK	SBH-02042001_130439	0.080518116	0.040870572	0.004269894
11	OK	SBH-02042002_135224	0.100127393	0.084288163	0.008123045

Figure 6-11: Hip fracture risk assessment results of qualified DXA images

In conclusion, the standalone program developed in this study can automatically classify DXA images based on the quality of proximal femur contours and assess hip fracture risks for qualified DXA images. Since there is no requirement for installation and knowledge of MATLAB, it is effective and convenient for the redistribution of the in-house MATLAB code for hip fracture risk assessment.

6.3 Discussion on the standalone program and the framework of web-based biomechanical tool

In this study, the standalone program is ready for a large-scale assessment, while the web-based biomechanical tool only has a preliminary framework and cannot be used in any clinical application. Therefore, this section will mainly focus on their differences and future directions of routine clinical assessment, instead of comparing their performance.

As mentioned earlier, assessing hip fracture risks using the standalone program requires installation of specific version of MATLAB Runtime. If the framework of web-based tool can be further improved, there will be no need for installing any environment or software, except for being connected to the Internet. In addition, any modification or upgrade of the standalone program requires repackaging and redistribution. Comparatively, the web-based tool will be easier to be maintained and upgraded, since the MATLAB code is stored on the server and can be modified as required. It can be concluded that a fully developed web-based biomechanical tool will have more potential for clinical application.

However, the web-based tool developed in this study is just a preliminary framework, and has certain limitations that need to be resolved (see Chapter 7.2), which make it far from being officially released. Under this circumstance, the standalone program can be considered as a temporary alternative to redistribute the in-house MATLAB code developed by our research group for hip fracture risk assessment.

In the future, the web-based biomechanical tool can be further improved and gradually replace the standalone program as the redistribution method and be applied to the routine clinical assessment of hip fracture risks.

Chapter 7

Conclusions and future work

7.1 Conclusions and contributions

The previous DXA-based finite element model for hip fracture risk assessment was not fully-automated because defective DXA images could not be automatically excluded. Before conducting the assessment with finite element modeling, DXA images were required to be classified manually based on their qualities. Apart from that, the previous model was fully built with in-house MATLAB code, which was difficult to be applied to the clinical assessment of hip fracture risks. This thesis addressed these limitations, the major conclusions and contributions of the present study are summarized as follows:

- Two automatic quality examination algorithms of DXA images have been proposed in this study based on neighboring pixels or points located on the femur outline. Unqualified DXA images can be identified and excluded before the finite element modeling to make sure a fully automatic and subject-specific assessment of hip fracture risks. After that, these two quality examination algorithms have been integrated with the previous DXA-based finite element model to be redistributed for a more general clinical application.

- With the utilization of *k*-NN algorithm in this study, the application of machine learning algorithms in the field of medical image classification has been explored.
- A fully standalone and automatic program for the clinical assessment of hip fracture risks has been built based on the two quality examination algorithms of DXA images and the DXA-based finite element model. With this computer program, clinicians or physicians can conduct subject-specific assessment of hip fracture risks, without any requirement on installation of MATLAB or knowledge on MATLAB programming.
- A preliminary framework of web-based biomechanical tool has been developed in this study as another redistribution method of the in-house MATLAB code and a direction for future studies. The operation of this web-based tool does not require any installation of environment or software. Although there exist some limitations, it provides a more convenient and efficient way for the routine clinical assessment of hip fracture risks using DXA images.

7.2 Limitations of the present study

The limitations of the present study are listed below:

- The DXA images used in this study are all produced by DXA scanners of the same model. Due to the differences in DXA images that produced by different scanners, there is a possibility that the quality examination algorithms developed in this study cannot be applied to some DXA images. Therefore, the applicability of the quality examination algorithms proposed in this study has not been verified conclusively.

- The safety issues of the web-based biomechanical tool have not been resolved. In this study, the HTML and PHP files, or the front- and back-ends of the web-based tool are stored on the same server, which can result in safety issues. If the front-end is under attack, the back-end, including the in-house MATLAB code, will also be influenced. To improve the performance of this web-based tool, these safety issues need to be eliminated.
- In the framework of the web-based tool, the file transmission between Linux system and main system has not been implemented. In this study, DXA images and results of hip fracture risk assessment are saved in the shared folder in Linux system, which can be accessed through Samba from the main system. In other words, users cannot upload or download files from the server directly. File transmission using the shared folder is acceptable for demonstration, but not for production.

7.3 Future work

To further improve the routine clinical assessment of hip fracture risks and the performance of the web-based tool, future works may include:

- DXA images can be retrieved from more DXA scanners to test the applicability of quality examination algorithms developed in this study.
- To eliminate the potential safety issues induced by using one server, the front-end (the webpage) of the web-based tool and the in-house MATLAB code can be hosted on two servers, as shown in Figure 7-1. In this scheme, clients send the request for hip fracture risk assessment and upload DXA images into Server 1, followed by the data transmission between two servers. When the assessment is completed, clients

will receive a notification message from Server 1 and download the results. By this way, the MATLAB code cannot be accessed by users directly. Thus, the web-based biomechanical tool can be better protected from the external attack.

- Web Authentication [88] can be applied to improve the performance and security of the web-based biomechanical tool, while reducing cost at the same time. To prevent any violation to the privacy rights, only clinicians or physicians should have access to patients' DXA scans. In this case, these clinicians or physicians can apply for a pair of assigned username and password to have the log-in entry to send the request for hip fracture risk assessment and upload DXA images.
- The requests from clients can be set as scheduled jobs using Web authentication to reduce the cost of real-time data transmission between two servers. After sending the request, the client can close the webpage and does not need to wait for the result. In the meantime, all the requests in a time period can be sent to Server 2 as scheduled jobs waiting to be processed. When the assessment for a request is completed on Server 2, the client will receive an email or a notification that includes a hyperlink for downloading the results.
- The application of machine learning algorithms in the fields of medical image processing and assessing the risk of bone disease needs to be further studied in the future. The possibility of directly assessing fracture risks using machine learning algorithms can be investigated.

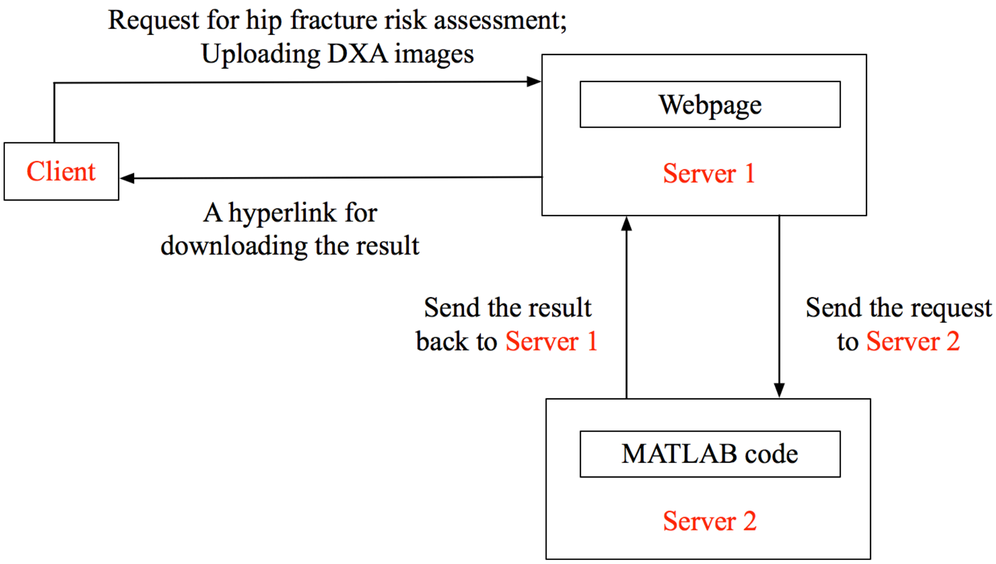


Figure 7-1: Developing the web-based biomechanical tool with two servers

Bibliography

[1] E. Barrett-Connor, “The economic and human costs ofosteoporotic fracture,” *The American Journal of Medicine*, vol. 98, no. 2, pp. 3S-8S, 1995.

[2] Osteoporosis Canada, “Osteoporosis Fast facts,” 2017. [Online]. Available: https://osteoporosis.ca/about-the-disease/fast-facts/. [Accessed: 30-Oct-2018].

[3] O. Johnell and J. Kanis, “An estimate of the worldwide prevalence and disability associated with osteoporotic fractures,” *Osteoporosis International*, vol. 17, no. 12, pp. 1726–1733, 2006.

[4] B. Gullberg, O. Johnell, and J. A. Kanis, “World-wide projections for hip fracture,” *Osteoporos International*, vol. 7, no. 5, pp. 407–413, 1997.

[5] J. A. Kanis, E. V. McCloskey, H. Johansson, A. Oden, L. J. Melton, and N. Khaltaev, “A reference standard for the description of osteoporosis,” *Bone*, vol. 42, no. 3, pp. 467–475, 2008.

[6] D. Testi, M. Viceconti, F. Baruffaldi, and A. Cappello, “Risk of fracture in elderly patients: a new predictive index based on bone mineral density and finite element analysis,” *Computer Methods and Programs in Biomedicine*, vol. 60, no. 1, pp. 23–33, 1999.

[7] Y. Luo, “A biomechanical sorting of clinical risk factors affecting osteoporotic hip fracture,” *Osteoporosis International*, vol. 27, no. 2, pp. 423–439, 2016.

[8] D. Marshall, O. Johnell, and H. Wedel, “Meta-analysis of how well measures of bone mineral density predict occurrence of osteoporotic fractures,” *British Medical Journal*, vol. 312, no. 7041, p. 1254(6), 1996.

[9] S. A. Wainwright *et al.*, “Hip fracture in women without osteoporosis,” *The Journal of Clinical Endocrinology & Metabolism*, vol. 90, no. 5, pp. 2787–2793, 2005.

[10] J. Kanis, E. McCloskey, H. Johansson, A. Oden, O. Ström, and F. Borgström, “Development and use of FRAX® in osteoporosis,” *Osteoporosis International*, vol. 21, no. Supplement 2, pp. 407–413, 2010.

[11] G. Iolascon, A. Moretti, G. Cannaviello, G. Resmini, and F. Gimigliano, “Proximal femur geometry assessed by hip structural analysis in hip fracture in women,” *Aging Clinical and Experimental Research*, vol. 27, no. 1, pp. 17–21, 2015.

[12] J. A. Kanis, A. Oden, H. Johansson, F. Borgström, O. Ström, and E. Mccloskey, “FRAX® and its applications to clinical practice,” *Bone*, vol. 44, no. 5, pp. 734–743, 2009.

[13] F. A. Mourtada, T. J. Beck, D. L. Hauser, C. B. Ruff, and G. Bao, “Curved beam model of the proximal femur for estimating stress using dual-energy x-ray absorptiometry derived structural geometry,” *Journal of Orthopaedic Research*, vol. 14, no. 3, pp. 483–492, 1996.

[14] L. Yang, N. Peel, J. A. Clowes, E. V McCloskey, and R. Eastell, “Use of DXA-based structural engineering models of the proximal femur to discriminate hip fracture,” *Journal of Bone and Mineral Research*, vol. 24, no. 1, pp. 33–42, 2009.

[15] W. Leslie, P. Pahlavan, J. Tsang, and L. Lix, “Prediction of hip and other osteoporotic fractures from hip geometry in a large clinical cohort,” *Osteoporosis International*, vol. 20, no. 10, pp. 1767–1774, 2009.

[16] C. M. Langton, S. Pisharody, and J. H. Keyak, “Comparison of 3D finite element analysis derived stiffness and BMD to determine the failure load of the excised proximal femur,” *Medical Engineering and Physics*, vol. 31, no. 6, pp. 668–672, 2009.

[17] S. Ahmed, “Automation of a DXA-based finite-element tool for clinical assessment of hip fracture risk,” University of Manitoba, 2016.

[18] D. D. Cody, G. J. Gross, F. J. Hou, H. J. Spencer, S. A. Goldstein, and D. P. Fyhrie, “Femoral strength is better predicted by finite element models than QCT and DXA,” *Journal of Biomechanics*, vol. 32, no. 10, pp. 1013–1020, 1999.

[19] K. E. Naylor, E. V Mccloskey, R. Eastell, and L. Yang, “Use of DXA-based finite element analysis of the proximal femur in a longitudinal study of hip fracture,” *Journal of Bone and Mineral Research*, vol. 28, no. 5, pp. 1014–1021, 2013.

[20] Y. Luo, S. Ahmed, and W. D. Leslie, “Automation of a DXA-based finite element tool for clinical assessment of hip fracture risk,” *Computer Methods and Programs in Biomedicine*, vol. 155, pp. 75–83, 2018.

[21] Y. Luo, Z. Ferdous, and W. D. Leslie, “Precision study of DXA-based patient-specific finite element modeling for assessing hip fracture risk,” *International Journal for Numerical Methods in Biomedical Engineering*, vol. 29, no. 5, pp. 615–629, 2013.

[22] S. Pisharody, R. Phillips, and C. M. Langton, “Sensitivity of proximal femoral

stiffness and areal bone mineral density to changes in bone geometry and density," *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, vol. 222, no. 3, pp. 367–375, 2008.

[23] J. Thevenot, P. Pulkkinen, J. Koivumaki, V. Kuhn, F. Eckstein, and J. Jamsa, Timo Thevenot, "Discrimination of cervical and trochanteric hip fractures using radiography-based two-dimensional finite element models," *Open Bone Journal*, vol. 1, no. 1, pp. 16–22, 2009.

[24] D. Testi, M. Viceconti, A. Cappello, and S. Gnudi, "Prediction of hip fracture can be significantly improved by a single biomedical indicator," *Annals of Biomedical Engineering*, vol. 30, no. 6, pp. 801–807, 2002.

[25] J. Op Den Buijs and D. Dragomir-Daescu, "Validated finite element models of the proximal femur using two-dimensional projected geometry and bone density," *Computer Methods and Programs in Biomedicine*, vol. 104, no. 2, pp. 168–174, 2011.

[26] M. Danielson *et al.*, "A comparison of DXA and CT based methods for estimating the strength of the femoral neck in post-menopausal women," *Osteoporosis International*, vol. 24, no. 4, pp. 1379–1388, 2013.

[27] Y. Luo, Z. Ferdous, and W. D. Leslie, "A preliminary dual-energy X-ray absorptiometry-based finite element model for assessing osteoporotic hip fracture risk," *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, vol. 225, no. 12, pp. 1188–1195, 2011.

[28] Z. Ferdous, "Development of a DXA-based patient-specific finite element model for assessing osteoporotic fracture risk," University of Manitoba, 2012.

[29] Y. Long, “Study of DXA-derived cortical bone thickness in assessing hip fracture risk,” University of Manitoba, 2014.

[30] E. M. Lewiecki, N. Binkley, and S. M. Petak, “DXA quality matters,” *Journal of Clinical Densitometry*, vol. 9, no. 4, pp. 388–392, 2006.

[31] H.-S. Kim and S.-O. Yang, “Quality control of DXA system and precision test of radio-technologists,” *Journal of Bone Metabolism*, vol. 21, no. 1, pp. 2–7, 2014.

[32] E. M. Lewiecki and N. E. Lane, “Common mistakes in the clinical use of bone mineral density testing,” *Nature Clinical Practice Rheumatology*, vol. 4, no. 12, p. 667, 2008.

[33] N. S. Altman, “An introduction to kernel and nearest-neighbor nonparametric regression,” *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[34] G. Bhattacharya, K. Ghosh, and A. S. Chowdhury, “An affinity-based new local distance function and similarity measure for kNN algorithm,” *Pattern Recognition Letters*, vol. 33, no. 3, pp. 356–363, 2011.

[35] Mathworks.com, “MATLAB Compiler,” 2017. [Online]. Available: https://www.mathworks.com/products/compiler.html. [Accessed: 06-Dec-2017].

[36] T. V. Nguyen, J. R. Center, and J. A. Eisman, “Osteoporosis: Underrated, underdiagnosed and undertreated,” *Medical Journal of Australia, 1 March 2004,* , vol. 180, no. 5, pp. S18–S22, 2004.

[37] E. M. Lewiecki *et al.*, “Best practices for dual-energy X-ray absorptiometry measurement and reporting: international society for clinical densitometry guidance,” *Journal of Clinical Densitometry*, vol. 19, no. 2, pp. 127–140, 2016.

[38] J. A. Kanis, “Diagnosis of osteoporosis and assessment of fracture risk,” *The Lancet*,

vol. 359, no. 9321, pp. 1929–1936, 2002.

[39] The University of Sheffield, “FRAX® Fracture Risk Assessment Tool.” [Online]. Available: https://www.sheffield.ac.uk/FRAX/. [Accessed: 02-Mar-2018].

[40] S. Silverman and A. Calderon, “The utility and limitations of FRAX: A US perspective,” *Current Osteoporosis Reports*, vol. 8, no. 4, pp. 192–197, 2010.

[41] N. B. Watts, E. M. Lewiecki, P. D. Miller, and S. Baim, “National Osteoporosis Foundation 2008 Clinician’s Guide to Prevention and Treatment of Osteoporosis and the World Health Organization Fracture Risk Assessment Tool (FRAX): What They Mean to the Bone Densitometrist and Bone Technologist,” *Journal of Clinical Densitometry*, vol. 11, no. 4, pp. 473–477, 2008.

[42] ClinRisk Ltd, “QFracture-2016,” 2018. [Online]. Available: http://www.qfracture.org/. [Accessed: 06-Mar-2018].

[43] Medscape, “Osteoporosis Self Assessment Tool for Men,” 2018. [Online]. Available: https://reference.medscape.com/calculator/osteoporosis-self-assessment-men. [Accessed: 06-Mar-2018].

[44] Medscape, “Osteoporosis Self Assessment Tool for Women,” 2018. [Online]. Available: https://reference.medscape.com/calculator/osteoporosis-self-assessment-women. [Accessed: 06-Mar-2018].

[45] K. H. Rubin *et al.*, “Comparison of different screening tools (FRAX®, OST, ORAI, OSIRIS, SCORE and age alone) to identify women with increased risk of fracture. A population-based prospective study,” *Bone*, vol. 56, no. 1, pp. 16–22, 2013.

[46] J. A. Kanis, N. C. Harvey, H. Johansson, A. Odén, E. V. McCloskey, and W. D. Leslie, “Overview of fracture prediction tools,” *Journal of Clinical Densitometry*,

vol. 20, no. 3, pp. 444–450, 2017.

[47] Garvan Institute of Medical Research, “Fracture Risk Calculator,” 2018. [Online]. Available: https://www.garvan.org.au/promotions/bone-fracture-risk/calculator/. [Accessed: 07-Mar-2018].

[48] J. Hippisley-Cox and C. Coupland, “Predicting risk of osteoporotic fracture in men and women in England and Wales: prospective derivation and validation of QFractureScores,” *BMJ*, vol. 339, no. 7733, pp. 1291–1295, 2009.

[49] M. J. Bolland *et al.*, “Evaluation of the FRAX and Garvan fracture risk calculators in older women,” *Journal of Bone and Mineral Research*, vol. 26, no. 2, pp. 420–427, 2011.

[50] M. S. Nanes and C. B. Kallen, “Clinical assessment of fracture risk and novel therapeutic strategies to combat osteoporosis,” *Fertility and Sterility*, vol. 92, no. 2, pp. 403–412, 2009.

[51] G. M. Blake and I. Fogelman, “The role of DXA bone density scans in the diagnosis and treatment of osteoporosis,” *Postgraduate Medical Journal*, vol. 83, no. 982, pp. 509–517, 2007.

[52] T. L. Kelly, D. M. Slovik, D. A. Schoenfeld, and R. M. Neer, “Quantitative digital radiography versus dual photon absorptiometry of the lumbar spine,” *Journal of Clinical Endocrinology and Metabolism*, vol. 67, no. 4, pp. 839–844, 1988.

[53] K. Engelke and C.-C. Glüer, “Quality and performance measures in bone densitometry,” *Osteoporosis International*, vol. 17, no. 9, pp. 1283–1292, 2006.

[54] Medicalexpo.com, “DEXA BONE DENSITOMETER / FAN BEAM,” 2017. [Online]. Available: http://www.medicalexpo.com/prod/hologic/product-70711-

428913.html. [Accessed: 06-Dec-2017].

[55] B. C. C. Khoo *et al.*, “In vivo short-term precision of hip structure analysis variables in comparison with bone mineral density using paired dual-energy X-ray absorptiometry scans from multi-center clinical trials,” *Bone*, vol. 37, no. 1, pp. 112–121, 2005.

[56] S. Van Cauter, M. De Beule, A. Van Haver, P. Verdonk, and B. Verhegghe, “Automated extraction of the femoral shaft axis and its distal entry point from full and reduced 3d models,” in *3rd ECCOMAS Thematic conference on Computational Vision and Medical Image Processing (VipIMAGE 2011)*, 2011, pp. 57–61.

[57] L. G. Shapiro and G. C. Stockman, “Binary Image Analysis,” in *Computer Vision*, 1st ed., Pearson, 2001, pp. 63–105.

[58] H. J. Vala and A. Baxi, “A review on Otsu image segmentation algorithm,” *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, vol. 2, no. 2, pp. 387–389, 2013.

[59] S. S. Skiena, “The algorithm design manual,” in *The Algorithm Design Manual: Second Edition*, 2nd ed., London: Springer London, 2008, pp. 166–167.

[60] L.-Y. Hu, M.-W. Huang, S.-W. Ke, and C.-F. Tsai, “The distance function effect on k-nearest neighbor classification for medical datasets,” *SpringerPlus*, vol. 5, no. 1, 2016.

[61] W.-J. Jiang, D. Xu, S.-L. Wu, and B. Meng, “Design of a flash digital watermarking application using MATLAB GUI,” in *2014 Tenth International Conference on Computational Intelligence and Security*, 2014, pp. 210–212.

[62] VMware, “VMware Virtualization for Desktop and Server, Application, Public and

Hybrid Clouds," 2018. [Online]. Available: https://www.vmware.com/ca.html. [Accessed: 27-Apr-2018].

[63] The Apache Software Foundation, "Welcome to The Apache Software Foundation," 2018. [Online]. Available: https://apache.org/. [Accessed: 26-Apr-2018].

[64] Security Space, "OS/Linux Distributions using Apache," 2018. [Online]. Available: https://secure1.securityspace.com/s_survey/data/man.201803/apacheos.html. [Accessed: 26-Apr-2018].

[65] R. B. Bloom, *Apache Server 2.0: The Complete Reference*. New York: Osborne/McGraw-Hill, 2002.

[66] The PHP Group, "PHP: Hypertext Preprocessor," 2018. [Online]. Available: http://www.php.net. [Accessed: 01-May-2018].

[67] D. R. Brooks, *Introduction to PHP for Scientists and Engineers: Beyond JavaScript*. London: Springer London, 2008.

[68] S. Parker, *Shell scripting expert recipes for Linux, Bash, and more*. Indianapolis, Ind.: Wrox/John Wiley & Sons, 2011.

[69] N. Bezroukov, "Introduction to the Unix shell history," 2017. [Online]. Available: http://www.softpanorama.org/People/Shell_giants/introduction.shtml. [Accessed: 02-May-2018].

[70] I. Milligan and J. Baker, "Introduction to the Bash Command Line," *Programming Historian*, 2014.

[71] D. R. Brooks, *An Introduction to HTML and JavaScript: for Scientists and Engineers*. London: Springer London, 2007.

[72] T. Berners-Lee and D. Connolly, "Hypertext Markup Language: A Representation

of Textual Information and MetaInformation for Retrieval and Interchange," 1993.

[73] W3.org, "HTML 5.3," 2018. [Online]. Available: https://www.w3.org/TR/html53/. [Accessed: 05-May-2018].

[74] W3.org, "XHTML^{TM} 1.1 - Module-based XHTML - Second Edition," 2018. [Online]. Available: https://www.w3.org/TR/xhtml11/. [Accessed: 05-May-2018].

[75] W3schools.com, "HTML Forms," 2018. [Online]. Available: https://www.w3schools.com/html/html_forms.asp. [Accessed: 07-May-2018].

[76] L. Babin, "Ajax Basics," in *Beginning Ajax with PHP From Novice to Professional*, Berkeley, CA: Apress, 2007, pp. 11–24.

[77] J. Lee, S. Cozens, and P. Wainwright, "Introduction to CGI," in *Beginning Perl*, 3rd ed., Berkeley, CA: Apress, 2010, pp. 317–348.

[78] W3.org, "CGI: Common Gateway Interface," 2011. [Online]. Available: https://www.w3.org/CGI/. [Accessed: 07-May-2018].

[79] M. Satran, "Microsoft SMB Protocol and CIFS Protocol Overview," *Msdn.microsoft.com*, 2018. [Online]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/aa365233(v=vs.85).aspx. [Accessed: 09-May-2018].

[80] C. Hertel, "Samba: An Introduction," *The Open Group*, 2001. [Online]. Available: https://www.samba.org/samba/docs/SambaIntro.html.

[81] C. L. Odoroff, "The information in contingency tables," *Technometrics*, vol. 22, no. 4, p. 632, 1980.

[82] C. E. Metz, "Basic Principles of ROC Analysis," *Seminars in Nuclear Medicine*, vol. 8, no. 4, pp. 283–298, 1978.

[83] F. Valverde-Albacete and C. Peláez-Moreno, "100% classification accuracy

considered harmful: The normalized information transfer factor explains the accuracy paradox," *PLoS ONE*, vol. 9, no. 1, p. e84217, 2014.

[84] F. J. Valverde-Albacete, J. Carrillo-de-Albornoz, and C. Peláez-Moreno, "A Proposal for New Evaluation Metrics and Result Visualization Technique for Sentiment Analysis Tasks," in *4th International Conference of the CLEF Initiative*, 2013, vol. 8138, pp. 41–52.

[85] C. Goutte and E. Gaussier, "A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation," in *Advances in Information Retrieval*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 345–359.

[86] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," *PLoS ONE*, vol. 10, no. 3, p. e0118432, 2015.

[87] H. He and E. A. Garcia, "Learning from imbalanced data," *EEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[88] W3.org, "Web Authentication: An API for accessing Public Key Credentials Level 1," 2018. [Online]. Available: https://www.w3.org/TR/webauthn/. [Accessed: 03-Jun-2018].

Appendix

Callback functions of objects on the GUI

Based on the style, there are three types of objects (text, edit and pushbutton) on the designed GUI, as shown in the following figure. Callback functions of pushbuttons need to be edited in the M-file generated along with the GUI.

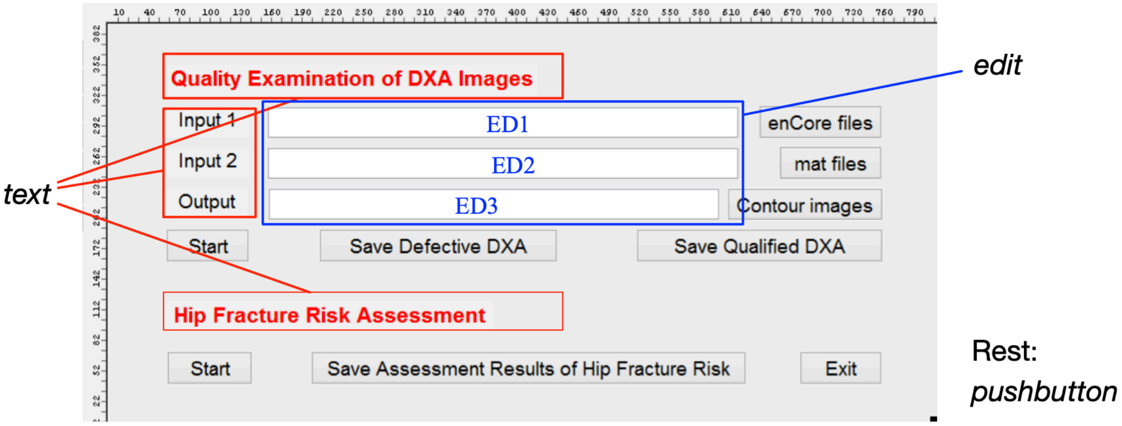


Figure: Layout of objects on the designed GUI

The M-files of this GUI are shown below, with edited callback functions highlighted in yellow.

```
function varargout = Hip_fracture_risk_assessment(varargin)
```

```
% IMAGE_QUALITY_IDENTIFICATION MATLAB code for Hip_fracture_risk_as-
sessment.fig
%      IMAGE_QUALITY_IDENTIFICATION, by itself, creates a new IM-
AGE_QUALITY_IDENTIFICATION or raises the existing
%      singleton*.
%
%      H = IMAGE_QUALITY_IDENTIFICATION returns the handle to a new IM-
AGE_QUALITY_IDENTIFICATION or the handle to
%      the existing singleton*.
%
%      IMAGE_QUALITY_IDENTIFICATION('CALLBACK',hObject,eventData,han-
dles,...) calls the local
%      function named CALLBACK in IMAGE_QUALITY_IDENTIFICATION.M with
the given input arguments.
%
%      IMAGE_QUALITY_IDENTIFICATION('Property','Value',...) creates a
new IMAGE_QUALITY_IDENTIFICATION or raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before Hip_fracture_risk_assessment_Open-
ingFcn gets called.  An
%      unrecognized property name or invalid value makes property ap-
plication
%      stop.  All inputs are passed to Hip_fracture_risk_assess-
ment_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Hip_frac-
ture_risk_assessment

% Last Modified by GUIDE v2.5 27-Jul-2019 21:41:22

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Hip_fracture_risk_assess-
ment_OpeningFcn, ...
                   'gui_OutputFcn',  @Hip_fracture_risk_assessment_Out-
putFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
```

```
% End initialization code - DO NOT EDIT


% --- Executes just before Hip_fracture_risk_assessment is made visi-
ble.
function Hip_fracture_risk_assessment_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Hip_fracture_risk_assessment
(see VARARGIN)

% Choose default command line output for Hip_fracture_risk_assessment
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Hip_fracture_risk_assessment wait for user response (see
UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Hip_fracture_risk_assessment_OutputFcn(hObject,
eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


function ED1_Callback(hObject, eventdata, handles)
% hObject    handle to ED1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ED1 as text
%        str2double(get(hObject,'String')) returns contents of ED1 as a
double


% --- Executes during object creation, after setting all properties.
function ED1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ED1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function ED2_Callback(hObject, eventdata, handles)
% hObject    handle to ED2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ED2 as text
%        str2double(get(hObject,'String')) returns contents of ED2 as a double


% --- Executes during object creation, after setting all properties.
function ED2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ED2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function ED3_Callback(hObject, eventdata, handles)
% hObject    handle to ED3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ED3 as text
%        str2double(get(hObject,'String')) returns contents of ED3 as a double


% --- Executes during object creation, after setting all properties.
function ED3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ED3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles) %% to get
input folder and path for ‘enCore’ files
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
folder_name = uigetdir;
if folder_name==0
    return
end

set( handles.ED1, 'String',folder_name );


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles) %% to get
input folder and path for ‘mat’ files
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
matlabfile_folder_name = uigetdir;
if matlabfile_folder_name==0
    return
end
set( handles.ED2, 'String',matlabfile_folder_name  );


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles) % to get
folder and path for output files
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
img_save_folder = uigetdir;
if img_save_folder==0
    return
end
set( handles.ED3, 'String',img_save_folder  );


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles) %% to start
quality check of DXA images
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global folder_name matlabfile_folder_name img_save_folder

folder_name = get( handles.ED1, 'String' );
matlabfile_folder_name = get( handles.ED2, 'String' );
img_save_folder= get( handles.ED3, 'String' );
```

```
bone_classify


% --- Executes during object creation, after setting all properties.
function pushbutton4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles) %% to save
the list of defective DXA images
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[ FileName, PathName ] = uiputfile(  '*.xls' );
if FileName==0
    return
end
global delete_list

P = [ PathName,FileName ];

results11=array2table(delete_list);
writetable(results11,P);


% --- Executes during object creation, after setting all properties.
function pushbutton5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called



% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles) %% to save
qualified DXA images
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global data_num;
global case_names;
global case_pred;
% %output to txt file
if data_num > 6900
[ FileName, PathName ] = uiputfile(  '*.txt' );
if FileName==0
    return
end
P = [ PathName,FileName ];
 f=fopen(P,'wt');
```

```
for i=1:data_num
    fprintf(f,'%s',case_names{i});
    fprintf(f,'%s','    ');
    fprintf(f,'%s\n',case_pred{i});
end
 fclose(f);
end

% %output to excel file (has storage limit <6960 cases)
if data_num <6900
[ FileName, PathName ] = uiputfile(  '*.xlsx' );
if FileName==0
    return
end

P = [ PathName,FileName ];
range = ['A1:A' num2str(data_num)];
xlswrite(P,case_names,range);
range = ['B1:B' num2str(data_num)];
xlswrite(P,case_pred,range);
end


% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles) %% to calcu-
late HFRI
% hObject    handle to pushbutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global folder_name matlabfile_folder_name

folder_name = get( handles.ED1, 'String' );
matlabfile_folder_name = get( handles.ED2, 'String' );

s_Main_code_in_house_ROI


% --- Executes on button press in pushbutton8.
function pushbutton8_Callback(hObject, eventdata, handles) %% to save
HFRI
% hObject    handle to pushbutton8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[ FileName, PathName ] = uiputfile(  '*.xls' );
if FileName==0
    return
end
global results

P = [ PathName,FileName ];

results11=struct2table(results);
writetable(results11,P);
```

```
% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles) %% to exit
the program
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close;
```