System Designs for Diabetic Foot Ulcer Image Assessment

by

Lei Wang

Submitted to the Department of Electrical and Computer Engineering in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical and Computer Engineering

at

WORCESTER POLYTECHNIC INSTITUTE

February, 2016

Author.....

Lei Wang

Department of Electrical and Computer Engineering
February, 2016

Certified by.....

Peder C. Pedersen
Professor of Electrical and Computer Engineering
Dissertation Supervisor

This doctoral dissertation has been examined by a committee as follows:

Dr. David Cyganski.....

Professor, Electrical and Computer Engineering

Dr. Michael A. Gennert.....

Director, Robotics Engineering Program

Dr. Emmanuel O. Agu.....

Associate Professor, Computer Science

Abstract

For individuals with type 2 diabetes, diabetic foot ulcers represent a significant health issue and the wound care cost is quite high. Currently, clinicians and nurses mainly base their wound assessment on visual examination of wound size and the status of the wound tissue. This method is potentially inaccurate for wound assessment and requires extra clinical workload. In view of the prevalence of smartphones with high resolution digital camera, assessing wound healing by analyzing of real-time images using the significant computational power of today's mobile devices is an attractive approach for managing foot ulcers. Alternatively, the smartphone may be used just for image capture and wireless transfer to a PC or laptop for image processing.

To achieve accurate foot ulcer image assessment, we have developed and tested a novel automatic wound image analysis system which accomplishes the following conditions: 1) design of an easy-to-use image capture system which makes the image capture process comfortable for the patient and provides well-controlled image capture conditions; 2) synthesis of efficient and accurate algorithms for real-time wound boundary determination to measure the wound area size; 3) development of a quantitative method to assess the wound healing status based on a foot ulcer image sequence for a given patient and 4) design of a wound image assessment and management system that can be used both in the patient's home and clinical environment in a tele-medicine fashion.

In our work, the wound image is captured by the camera on the smartphone while the patient's foot is held in place by an image capture box, which is specially design to aid patients in photographing ulcers occurring on the sole of their feet. The experimental results prove that our image capture system guarantees consistent illumination and a fixed distance between the foot and camera. These properties greatly reduce the complexity of the subsequent wound recognition and assessment.

The most significant contribution of our work is the development of five different wound boundary determination approaches based on different computer vision algorithms. The first approach employs the level set algorithm to determine the wound boundary directly based on a manually set initial curve. The second and third approaches are the mean-shift segmentation based methods augmented by foot outline detection and analysis. These two approaches have been shown to be efficient to implement (especially on smartphones), prior-knowledge independent and able to provide reasonably accurate wound segmentation results given a set of well-tuned parameters. However, this method suffers from the lack of self-adaptivity due to the fact that it is not based on machine learning. Consequently, a two-stage Support Vector Machine (SVM) binary classifier based wound recognition approach is developed and implemented. This approach consists of three major steps 1) unsupervised super-pixel segmentation, 2) feature descriptor extraction for each super-pixel and 3) supervised classifier based wound boundary determination. The experimental results show that this approach provides promising performance (sensitivity: 73.3%, specificity: 95.6%) when dealing with foot ulcer images captured with our image capture box. In the third approach, we further relax the image capture constraints and generalize the application of our wound recognition system by applying the conditional random field (CRF) based model to solve the wound boundary determination. The key modules in this approach are the TextonBoost based potential learning at different scales and efficient CRF model inference to find the optimal labeling. Finally, the standard K-means clustering algorithm is applied to the determined wound area for color based wound tissue classification.

To train the models used in the last two approaches, as well as to evaluate all three methods, we have collected about 100 wound images at the wound clinic in UMass Medical School by tracking 15 patients for a 2-year period, following an IRB approved protocol. The wound recognition results were compared with the ground truth generated by combining clinical labeling from three experienced clinicians. Specificity and sensitivity based measures indicate that the CRF based approach is the most reliable method despite its implementation complexity and computational demands. In addition, sample images of Moulage wound simulations are also used to increase the evaluation flexibility. The advantages and disadvantages of three approaches are described.

Another important contribution of this work has been development of a healing score based mechanism for quantitative wound healing status assessment. The wound size and color composition measurements were converted to a score number ranging from 0-10, which indicates the healing trend based on comparisons of subsequent images to an initial foot ulcer image. By comparing the result of the healing score algorithm to the healing scores determined by experienced clinicians, we assess the clinical validity of our healing score algorithm. The level of agreement of our healing score with the three assessing clinicians was quantified by using the Kripendorff's Alpha Coefficient (KAC). Finally, a collaborative wound image management system between the PC and smartphone was designed and successfully applied in the wound clinic for patients' wound tracking purpose. This system is proven to be applicable in clinical environment and capable of providing interactive foot ulcer care in a telemedicine fashion.

Acknowledgements

First, I would like to express my greatest gratitude to my advisor, Prof. Peder C. Pedersen, for providing me this opportunity to work as a research assistant for the foot ulcer image assessment project since the beginning of 2012. We have been working together to overcome a lot of difficulties beyond our initial imagination. During these four years, Professor Pedersen has always been giving me tremendous support, both academically and mentally. Without his help, I could never achieve what I have accomplished in this dissertation. He is smart, respected, graceful and self-demanding. These impressed me a lot and I certainly will benefit from the working experience with him for my entire life.

I would like to thank all members in our SUGAR research team for their consistent encouragement. Professor Diane Strong, our project manager, and Professor Bengisu Tulu from business school made everything so organized that we can comfortably conduct our research. Professor Emmanuel Agu, a computer vision expert, who has given me so many inspiring advices, helped me to move forward when I had no idea for the next step. Besides, I sincerely appreciate all efforts from our collaborators at the University of Massachusetts Medical School during the two user studies. Dr. Ronald Ignotz, a former research scientist in Plastic Surgery department, and Heather Tessier, the lab manager, handled all the subject recruiting and IRB applications, and help us to test our system on clinical patients and collect valuable feedback from them.

I would like to thank the National Science Foundation for their funding during the past four years. It gave us the luxury to focus on the research and travel to conferences making our work publicly known. Without their funding, I can never finish my PhD study.

My good friend, Steve (Qian) He, another PhD student working in our Sugar app from computer science department, has spent a lot of time to help me build up the collaborative system. His outstanding coding expertise provided us the ability to design a user-friendly

software interface on the smartphone side. Furthermore, I thank my friends, Joyce Cheng, Xiaoqiang Fu, Ying Lang and my cousin Zhenhao Lei, for their comforting when I felt frustrated and hopeless. Their help made my life much easier when the road was dark.

Finally and most importantly, I would love to thank my parents for their enormous support, selfless sacrifices and ever-lasting love. They were my best safety net, and never let me down when I needed help. I cannot imagine that I could make anything happen without them.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Challenges

The World Health Organization (WHO) estimates that more than 180 million people worldwide have diabetes mellitus and this number will possibly double by the year 2030 [1]. According to [1], one frequently occurring complication of diabetes (especially type 2 diabetes) is Diabetic Foot Syndrome (DFS), which may increase the risk of developing foot ulcers and wounds as a result of neuropathy. For individuals with type 2 diabetes, foot ulcers constitute a significant health issue affecting 5-6 million individuals in the US [2], [3]. Such these wounds are painful, susceptible to infection and very slow to heal [4], [5]. According to published statistics, diabetes-related wounds are the primary cause of non-traumatic lower limb amputations, with approximately 73,000 such amputations in the US in 2010 [6]. The foot ulcer is a typical chronic wound, whose care and treatment involve high costs for health services and imply important consequences for the health of the population [7]. An estimated 6.5 million patients in the United States are affected by chronic wounds, and it is estimated that in excess of $25 billion is spent on the treatment of chronic wounds every year. The burden is growing rapidly due to increasing health care costs, an aging population and a sharp rise in the incidence rate of diabetes and obesity worldwide [8]. The cost associated with treating diabetic foot ulcers is high and is estimated to be $15,000 per year per individual. Overall diabetes healthcare cost in the US was estimated at $245 billion in 2012 [6].

Accurate measurement of wound dimension and assessment of ulcer healing becomes an important task in clinical wound treatment, as the care for each individual patient largely

is based on this assessment. Today, clinicians usually measure wound dimensions (not area) using standardized scale and often record the results in paper format. The status of the wound tissue is based mainly on visual inspection, to which is added other relevant information about the patient's state of health [7]. One example of the most widely used tools for assessment of the healing status of a pressure ulcer is called the PUSH (Pressure Ulcer Scale for Healing) tool [9]. This tool assess the ulcer by visual inspection of four types of wound tissues, i.e., epithelial, granulation, slough, or necrosis, along with the manual measurement of the wound dimensions and the subjective assessment of oozing from the ulcer. According to [7], such a visual inspection based approach is a subjective and potentially inaccurate method, which lacks of the capability to perform consistent wound area determination or tissue classification in an objective way.

Besides, patients must go to their wound clinic on a regular basis to have their wounds checked by their clinicians. Bus et al [1] found that regular foot care for patients with a high risk for ulceration is recommended once at least every three months. Furthermore, the patients with conditions including visual impairment, living alone, cognitive problems, or lack of knowledge about the disease should have more frequent proper examination of their feet and such patients may benefit from screening on a weekly basis in order to identify ulceration or pre-signs of ulceration at an early stage [10]. This need for frequent clinical evaluation not only causes inconvenience for patients and an additional workload for clinicians, but also adds to the health care cost since patients with limited mobility require special transportation, e.g., ambulances [10].

Computer-aided measurements of the size and characterization of chronic wounds is a new approach to standardize the accuracy of chronic wound assessment. Such a technology can potentially reduce clinicians' workload considerably, make the treatment and care more consistent and accurate, increase the quality of documentation in the medical record and enable clinicians to achieve quality benchmarks for wound care, as determined by the Center for Medicare Services. Also, computer-aided measurements will permit the definition of "standard" wound healing rates, and minimize inter- and intra- observer variations [11].

The current state of the art approach in measuring wound size using digital images known as digital planimetry [12], requires the clinician to identify wound borders and tissue type within the image. This is a time-intensive process and is a barrier to achieve quality clinical benchmarks. Other attempts to use image processing techniques for such tasks include measurement of wound area, or alternatively using a volume instrument system (MAVIS) [13] or a medical digital photogrammetric system, which is able to automatically measure the dimensions of skin wounds. These wound image processing methods were based on color segmentation algorithms and were able to segment an image into one of these three tissue types: healthy skin, wound tissue and epithelial tissue. These approaches suffer from drawbacks including high cost, complexity, and lack of tissue classification [14]. Hence, for more accurate and flexible wound monitoring, PC or even mobile computing device based wound assessment methods are needed for an automatic wound area determination, tissue type classification and further overall evaluation of the wound healing status. Using this method, real-time wound monitoring in both the clinical and home environment context becomes possible given a set of image capture and computing devices configured with carefully designed wound analysis software.

1.2 Image Capture for Diabetic Foot Ulcers

Based on clinical literature [1], [15], foot ulcers usually occur on the sole of the feet. It is particularly challenging for diabetic patients, especially elderly patients or patients with limited flexibility, to photograph such ulcers due to mobility limitations. Hence, for wound care at the patients' home, it is necessary to design a device which enables individuals with advanced diabetes to capture quality foot ulcer images. From our literature review, we found two most related works as follows.

Bus et al [1] described a device that uses digital photography as a telemedicine monitoring tool to assess the high risk diabetic foot in the home environment for prevention purposes. This device, shown in part (a) in Figure 1.1, provides high-quality digital photographs of the plantar surfaces of the feet under standardized conditions with respect to foot positioning camera orientation and light conditions. Moreover, this device

is fully automatic, and the acquired photographs are sent automatically via the Internet to a central database server from where they can be downloaded and assessed by trained healthcare professionals. In [1], all wound photographs were assessed independently by four clinically trained observers and compared with live and repeated photographic assessments of the feet by the same observers. The agreement between assessments is reported as 74% to 100%. The major shortcoming of this device is its dimensions. The foot soles are imaged from an optical distance of about 1 meter in order to minimize geometric image distortions. Even with a high-quality mirror to fold the imaging light path, the outer dimensions of the device, which can be indirectly determined by the dimension of the light path, makes it not portable. There is no specific discussion about the cost for this device.

(a)

(b)

Figure 1.1 Two devices for foot ulcer image capture, and both images are screenshots from [1] *and* [15]*, respectively*

Foltynski et al [15] described another novel device called “Patient’s Module (PM)” , as shown in part (b) in Figure 1.1, is presented for tele-monitoring of patients with diabetic foot ulcer. The PM consists of an optical scanner as image capture device, a PC motherboard with processor, RAM memory, hard disk drive, RS232-IR converter, AD converter with I/O ports, wireless GSM modem, wireless controller board for remote control and custom-made case. The study revealed that the PM can be used as a home tele-monitoring device, and the analysis of the data sent from patient’s home enables the assessment of wound healing progress, giving the physician the possibility for earlier change of the treatment if the wound healing is not satisfactory.

1.3 Computer Based Wound Image Analysis Method Overview

Reliable wound assessment is best achieved by performing accurate measurements of wound area size, colors and size of different wound tissues, including proliferation, infected area, slough or necrosis. Due to the lack of consistency, routine visual inspection based wound care, even with the assistance of tools such as PUSH, cannot ensure an objective assessment of wound healing rate. Hence, PC/laptop based evaluation of foot ulcers based on computer vision and image processing techniques is an ideal approach to accurate chronic assessment. Automatic detection of foot ulcer size and tissue composition are particularly desirable for both clinicians and diabetic patients to monitor the wound healing status and to provide more effective wound care.

Prior work on ulcer image analysis: Current research on chronic ulcer image analysis mainly focuses on the following tasks [10]: 1) wound boundary determination and size measurement, 2) wound tissue classification within the determined wound area, and 3) wound healing rate assessment. In earlier investigations, active contour models or and level set based methods have commonly been applied to directly measure the area of wounds semi-automatically [16] [17] [18]. Although good segmentation results were obtained under well-controlled conditions, these approaches still suffer from several problems including dependence on the initial contour setting, sensitivity to illumination conditions and instability when dealing with the poorly defined wound boundary or false edges. Moreover, it was reported in [18] that these method failed to provide correct results when the wound boundary has an irregular shape. In other works, the

investigators decomposed the wound boundary determination into two sub-tasks: 1) skin area segmentation from the background and 2) wound area segmentation [19] [20]. As in [19], the fuzzy C-Mean (FCM) clustering method was applied to both two sub-tasks in different color spaces for burn wounds segmentation. The Positive Predicative Value (PPV) and the sensitivity (S) were used to measure the segmentation performance. It was reported that the proposed method was able to provide good results based on the ground truth generated by experienced clinicians, but the results were limited to images of burn wounds (PPV > 0.87 and S > 0.72).

Wound area determination approaches by applying machine learning based classifiers have also been widely used [7], [21]–[24]. In [22] [21], the authors utilized a color feature (composed of the multi-dimensional histogram intersection and sampling, green and blue channels from RGB model) and a Salient Covariance (SCOV) texture feature to differentiate the wound and healthy skin. Then a regular support vector machine (SVM) classifier trained on 6 images was applied in a pixel-wise fashion for segmentation of the entire wound region. The average error rate of classification was reported as 9% on 2 test images. However, the validity of this method has not been substantiated due to their small image dataset.

In some of the above publications [7], [23], [24], a bottom-up object recognition scheme was applied for wound tissue classification and indirect wound area determination (by grouping all the regions classified as one type of the wound tissues). Generally, these classification approaches consist of following three modules: 1) image segmentation, 2) feature extraction for each segment and 3) classifier training on large numbers of features from either the wound or non-wound segments. In module 1, different image segmentation algorithms were applied, including mean shift [23] [7], JSEG (an unsupervised segmentation of color-texture regions in images) [23], region growing [10] [7] and integrated segmentation approach [24]. In [23], the segmentation results were evaluated by calculating the matching degree (overlapping score) with the medical reference constructed from clinicians' tracings. However, this approach was discarded in [7] because of the large intra-observer variability, which undermined the reliability of using manually delineated ground-truth reference images. Instead, they evaluated the segmentation results by measuring the region uniformity and contrast in segmented

images. In module 2, different feature descriptors were used to capture the most distinguishable characteristics of the wound tissue. In [23], several color and texture descriptors that had been already used in dermatological imaging system were compared. The authors in [23] concluded that the feature descriptor that combined 23 attributes provided the best results in terms of the specificity (92%) and sensitivity (77%) measurements. The attributes included Mean Color Descriptor (MCD) and Dominant Color Descriptor (DCD) as color descriptor and Gray Level Co-Occurrence Matrix (GLCM) as texture descriptor. Veredas *et al* [7] utilized a combination of color features (extracted on image pixels and histograms) and texture features (extracted by two-level wavelet filters). The features were used as the descriptor for each region in the segmented image. In [20], a combination of geometric indicators (area, perimeter, centroid and axes lengths) and min/max/mean intensities of each color channel were employed to describe each region. In module 3, support vector machine (SVM) [19] [22] [23] and Artificial Neural Network (ANN) [19] [20] were applied to train the wound tissue classifier. A cascade classifier based on ANN and Bayesian committee machine was formed in [19], which was reported with a global performance (sensitivity = 78.7%, specificity = 94.7%) in wound tissue classification.

Prior work on wound healing assessment: In addition to wound boundary determination and tissue recognition, wound healing and monitoring have been the subject of several studies on wound image analysis. In [25], the authors described how the wound healing rate should be defined to enable appropriate description of wound healing dynamics. They suggested that wound area measurements should be described as a percentage of initial wound area and fitted to a delayed exponential model. In the proposed model, the wound healing rate is described by the slope of the time curve. The slope as a function of time is fitted to the normalized wound area measurements after initialization delay. Loizou et al [26] established and standardized an objective technique to assess the progress of healing of foot wounds. In this study, the authors perform automated segmentation of the wounds based on snakes while the initial contour is automatically positioned to the area of interest. They followed up the rate of wound healing by means of texture features and geometric measures extracted from the automated segmented areas. The overall conclusion was that while none of the geometrical features (area,

perimeter, x-, y-coordinate) showed significant changes between visits, several texture features (mean, contrast, entropy, sum variance, sum average) did change accordingly, indicating these features might provide a better wound healing rate indication. Finally, in [27] the authors evaluated several methods for quantitative wound assessment on diabetic foot ulcers, namely wound volume, wound area, and wound coloration.

1.4 Wound Image Analysis Software Overview

Quite a few computer device based tools are currently available for wound analysis and monitoring. However, most of these software packages have only incorporated semi-automated wound area detection or segmentation to minimize the clinician's initial involvement.

PictZar Digital Planimetry Software [28] is a commercial software for wound analysis, which provides measurements such as length, width, surface area, circumference, and estimated volume to the users. The software requires the user to draw and calibrate in order for the above measurements to be computed. In [29], a color image processing software application called WITA was developed for digital wound image analyzing. Based on the tissue samples used for training, the program classifies the tissue and monitors wound healing. The wound tissue types are divided into black necrotic eschar, yellow fibrin or slough, red granulation tissue and unclassified parts of the image; however, no evaluation against the known ground truth was presented for the image analysis part of the software. To obtain wound dimensions, users must mark the distance on the photograph that is equivalent to 1 cm (or 1 inch). A different approach to wound monitoring software and hardware was proposed in [30]. The authors developed a new "wound mapping" device, which is based on electrical impedance spectroscopy and involves the multi-frequency characterization of the electrical properties of wound tissue under an electrode array. This approach, however, requires major changes to the clinical routine in wound care (nurses or care-givers needed to use large-scale medical imaging devices to take sample wound images on daily basis at the wound clinic). In [31], the authors developed a tool called "MelaFind" including an imaging device and automatic melanoma detection system. Based on my research, no further detailed information about the evaluation of the system performance was revealed for MelaFind.

Another evidence-based wound assessment tool, called “Silhouette TM”, was developed by Aranz Medical [32]. The system includes a laser imaging device that precisely and consistently measures the wound depth and volume of wounds and their healing progress. The system software allows users to store and manage wound information that is then available instantly to key stakeholders. It is claimed that any single Silhouette-derived measurement is likely to be within a 2% error for area, 1% error for perimeter, 5% error for average depth and 5% error for volume (95% confidence interval). Inter and intra-rater variability is extremely low (below 1% for area and perimeter and below 2% for average depth and volume). Based on [28], this product is now available for clinical practice. The main disadvantage of this system is that it relies on laser imaging techniques. Hence the cost is high and it is only applicable in clinical environment.

In [33], a smartphone based application (this app has different versions for both Android and iOS based smartphones), called “MOWA”, for advanced management of pressure ulcers was developed. In this application, the user first opens the pre-captured wound image in the smartphone Gallery, and then draws the wound boundary using a “pen tool” with a pre-designed mask. The wound area is calculated either automatically if a blue marker with a fixed size is found on the image or manually by setting the size of the surrounding rectangle accordingly. Afterwards, this app will automatically perform the color segmentation within the wound area and categorize wound tissues into several types including necrosis, fibrin, granulation and unknown. It is claimed that this application is a practical software solution for analyzing wound image, but no performance or efficiency evaluation has been presented.

1.5 General Wound Image Analysis System

In this work, we propose a general wound image analysis system which consists of several functional modules including wound image capture, wound image storage, wound image preprocessing, wound boundary determination, wound analysis by color segmentation and wound trend analysis based on a time sequence of wound images for a given patient. The functional diagram of our quantitative wound assessment system is shown as in Figure 1.2 and explained below.

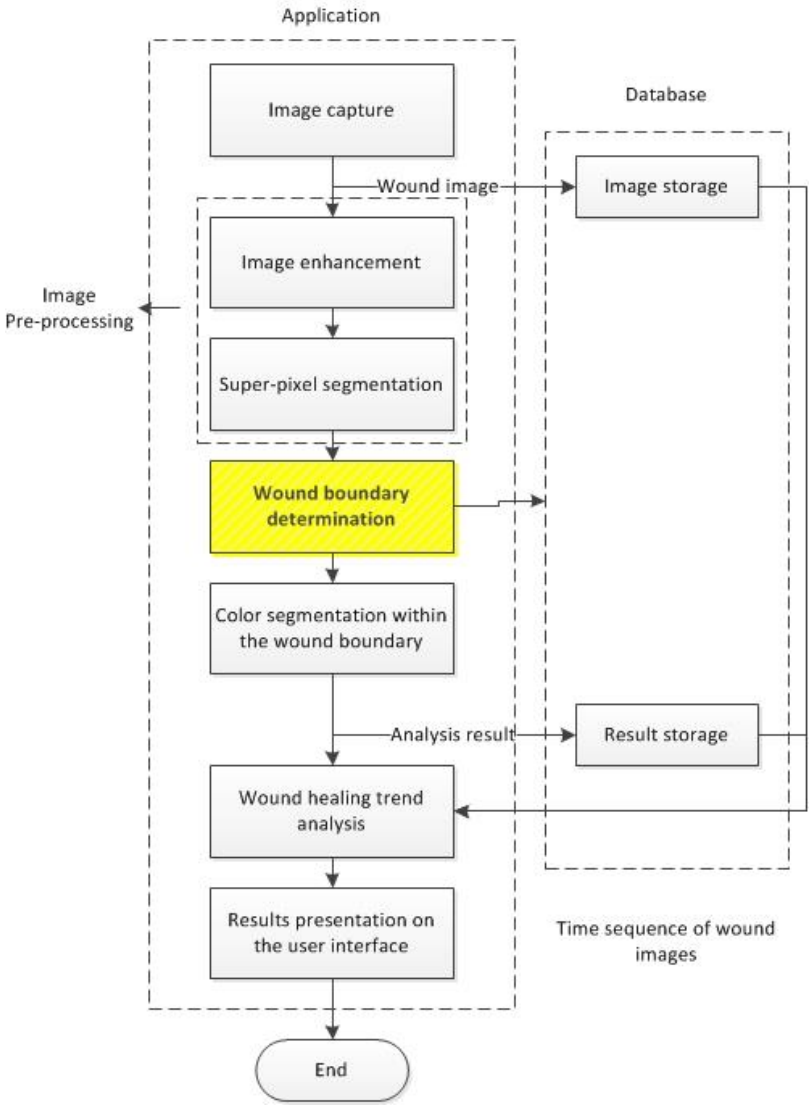


Figure 1.2 Wound Image Analysis Software System [20]

All these processing steps can be carried out solely by the computational resources of the smartphone or on the collaborative system of PCs and smartphones (In the collaborative system, the smartphone will be used to capture image and the PC will be used to analyze wound images). Note that the words highlighted in bold in the text correspond to specific blocks in Figure 1.2 with block diagrams.

A Nexus 5 smartphone was chosen due to its excellent CPU+GPU performance and high resolution camera. Although there are likely performance variations across the cameras of modern smartphones, such a study was considered beyond the scope of this research. After the wound image is captured, the JPEG file path of this image is added into a

wound image database. This compressed image file, which cannot be processed directly with our main image processing algorithms, therefore needs to be decompressed into a 24-bit bitmap file based on the standard RGB color model. In our system, we use the built-in APIs of the Android smartphone platform to accomplish the JPEG compression and decompression task. The "image quality" parameter was used to control the JPEG compression rate. Setting "image quality" to 80 was shown empirically to provide the best balance between quality and storage space. For an efficient implementation on the smartphone alone, no method was used to further remove the artifacts introduced by JPEG lossy compression.

In the **Image preprocessing** step, we first down-sample the high resolution bitmap image to speed up the subsequent image analysis and to eliminate excessive details that may complicate wound image segmentation. In our case, we down-sample the original image (pixel dimensions 3264 x 2448) by a factor 4 in both the horizontal and vertical directions to pixel dimensions of 816 x 612, which has proven to provide a good balance between the wound resolution and the processing efficiency. In practice, we use the standard API for image resizing on the Android smartphone platform to ensure high efficiency. Second, we smooth the images to remove noise (assumed mainly to be Gaussian noise produced by image acquisition process) by using the Gaussian blur method whose standard deviation $\sigma = 0.5$ was empirically judged to be optimal based on multiple experiments. Afterwards, for most of bottom-up object recognition systems mentioned in Section 1.3, the super-pixel segmentation can also be viewed as an important pre-processing step to further down-sample the image grid by focusing on single homogeneous region, instead of single pixel, during the classification process.

The wound boundary determination module, which is marked in yellow in Figure 1.2, is the most important part in the entire wound image analysis system. To determine the boundary of the wound area, different computer based object detection approaches mentioned in Section 1.3 will be applied. These approaches fall into two categories: the non-machine learning based or machine learning based methods. For non-machine learning based methods, the implementation usually is easy, efficient and does not depend on any priori-knowledge. This makes it quite suitable for the wound recognition task based on smartphone-alone platforms. One disadvantage is the fact that the wound

recognition approach is not adaptive and suffers from dealing with foot ulcers locating at foot boundary or toes, due to the lack of help from clinical input. For machine learning methods, the complexity of the model training makes the full implementation impossible for smartphone-alone system. Hence, we need to implement the model based wound boundary determination method on the smartphone or PC, and the model training on a sever or remote PC separately in an offline mode. This tends to increase the system complexity and the platform dependency. However, given a sufficient number of training images, the model can accurately determine boundaries for various types of wounds regardless of theirs shape, scale and location.

When the wound boundary has been successfully determined and the wound area has been calculated, we next evaluate the healing state of the wound by performing **Color segmentation**, with the goal of categorizing each pixel in the wound boundary into certain classes labeled as granulation, slough and necrosis [34][35]. A classical self-organized clustering method with high computational efficiency is used [36]. After the color segmentation, a feature vector including the wound area size and dimensions for different types of wound tissues is formed to describe the wound quantitatively. This feature vector, along with both the original and later analyzed images, is saved in the result database.

The ***Wound healing trend analysis*** is performed on a time sequence of images belonging to a given patient to monitor the wound healing status. The current trend is obtained by comparing the wound feature vectors between the current wound record and the one that is just one standard time interval earlier (typically one or two weeks). Alternatively, a longer term healing trend is obtained by comparing the feature vectors quantitatively between the current wound and the base record which is the earliest record for this patient.

1.6 Organization of Dissertation

The material presented in Chapter 2 to 6 is roughly organized based on the flowchart shown in Figure 1.2. Chapter 2 describes the image capture device design to facilitate foot ulcer photographing, and also compare three widely used super-pixel segmentation algorithms using our wound image dataset. In Chapter 3 to 5, the most important module

in Figure 1.2, the wound boundary determination, is researched and implemented. Chapter 3 presents two non-machine learning based wound boundary determination approaches and supplemented with several optimization methods including the GPU based parallel implementation and the "turning points detection" method to better recognize the wounds located at the foot edges or toes. In Chapter 4, a SVM based two-stage cascade wound classifier is developed by applying different super-pixel based wound descriptors. Following this, Chapter 5 describes and implements a more robust but also computationally more intensive machine learning model for wound boundary determination based on the conditional random field model. To achieve the best results, three CRF formulations have been applied and the performance is compared. To train the models and evaluate the wound boundary determination approaches proposed in Chapter 3-5, the ground truth wound labeling is generated on 100 wound images collected given an IRB approval at the wound clinic in UMass Medical School, with help from three experienced wound clinicians. Chapter 6 describes our approach for color based wound tissue classification and wound healing status assessment. Finally, conclusions and future directions are presented in Chapter 7.

Chapter 2

Wound Image Capture and Pre-processing

The ultimate goal for our wound image analysis algorithm is to determine the wound boundary and assess the wound healing status from a time sequence of wound images. To perform an objective wound analysis which may require comparison of visual indicators between different images, there are two key requirements: 1) the wound images should be captured using a special device to ensure the consistency of imaging conditions including illumination, viewpoint and scale, and 2) captured raw wound images may need to be refined by applying image pre-processing techniques. In addition, current computer vision tasks are usually based on super-pixels (perceptually homogeneous regions or patches) [37], instead of the individual pixels, to extract more visually meaningful features and reduce the complexity of following tasks. Therefore, we also consider the super-pixel segmentation as part of the wound image pre-processing flow. In the chapter, we will first introduce a novel design of image capture box to facilitate the image taking process for diabetic patients. Then, three widely used super-pixel segmentation algorithms will be reviewed and applied to our wound image dataset. The segmentation performance of these algorithms is evaluated based on uniformity and contrast measures.

2.1 Image Capture Box Design

As mentioned earlier, two devices for the image capture of diabetic foot ulcers have been reported in the literature [1] [15]. However, drawbacks to these designs are either large dimensions or high cost. Moreover, both devices require Wi-Fi connectivity and a laptop or PC for image processing. Our image capture box is specifically designed to aid patients with type 2 diabetes in photographing ulcers occurring on the sole of their feet.

This task can be particularly challenging due to mobility limitations, common for individuals with advanced diabetes. To this end, we designed and built an image capture box with an optical system containing a dual set of front surface mirrors, integrated LED lighting and a comfortable, slanted surface for the patients to place their foot. The design ensures consistent illumination and a fixed optical path length between the sole of the foot and the camera, so that pictures captured at different times would be taken from the same camera angles and under the same lighting conditions.

2.1.1 Optical Design

After some initial designs that did not accomplish the desired goals, we formulated a new design concept using two mirrors, which are in straight angle. The ray diagram for this design is shown in Figure 2.1.

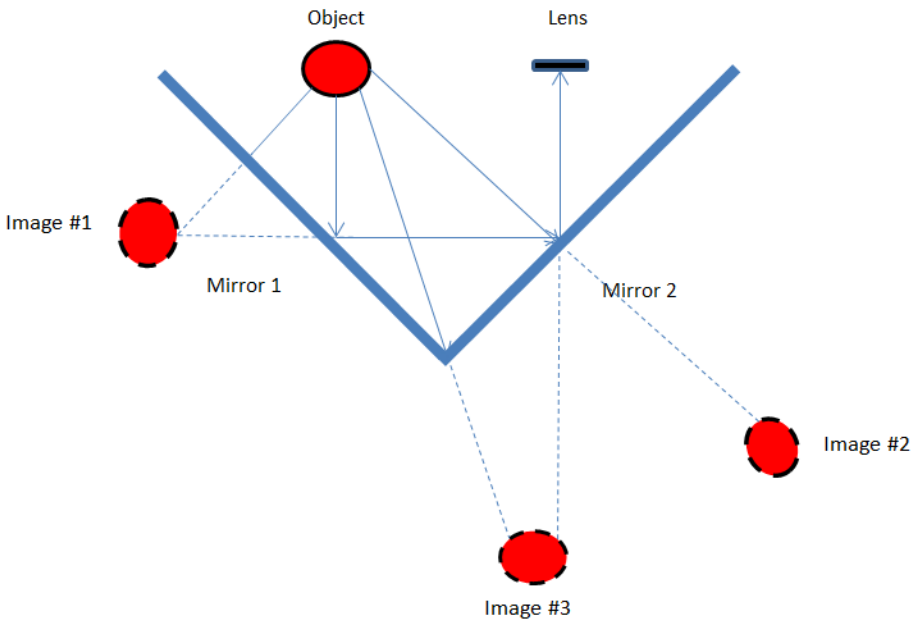


Figure 2.1 Ray diagram for two mirrors in straight angle

An inspection of the diagram in Figure 2.1 shows that there are three images in this optical system. Image #1 is located directly across the face of the mirror and at the same distance behind the mirror as the object is in front of the mirror. Image #2 is located directly across the other face of the mirror and the same distance behind the mirror as the

object is in front of the mirror (still nothing surprising). Both image #1 and #2 are primary images. On the other hand, image #3 is located directly through the joint edge of the mirror and the same distance from the joint edge as the object is from the joint edge. In fact, a diagonal line drawn from the object location through the joint edge of the mirror will pass through the third image which is a secondary image. So the image locations for the primary images are found in the usual way; but the image location for the secondary image must be found by measuring the object's distance to the joint edge of the mirror and then measuring along the diagonal line the same distance behind the mirror. Actually, the final image captured by the smartphone using the new foot box is image #3.

2.1.2 Detailed Design of Image Capture Box

In this section, we present the mechanical design based on 3D drawing using SolidWorks, which is a 3D mechanical CAD (computer-aided design) program. The drawings from different angles are shown in Figure 2.2.

As seen in the figure above, the entire box is in a shape of rectangular trapezoid. The openings for placing the foot and smartphone are on the slanted surface which is placed at a 45 degree angle with respect to the horizontal base. This design allows the patient to rest his/her foot comfortably and view the wound through the smartphone camera. When using the box, the patient is able to ensure that the wound is completely located in the opening by simply observing the camera view on the smartphone.

(a)

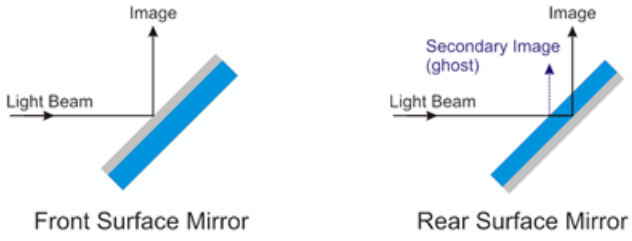
(b)

(c) *(d)*

Figure 2.2 3D drawing of the mechanical structure of the foot box. (a) from the back; (b) from the front; (c) internal structure from the front; (d) internal structure from left side

Inside the box there are two mirrors placed in a 90° angle relative to each other. The connecting line of the two mirrors is placed in a plane which is parallel to the slanted surface. Since we use a pair of straight-angled mirrors to perform imaging which contains more than one reflection, the ghosting effect will happen as shown in part (b) of Figure 2.3 if normal mirrors are used. In order to avoid this interference, we use front surface mirrors. This type of mirror has the reflective surface on the front, as opposed to the conventional, back surface mirror with the reflective surface on the back side of the glass. The optical path for both the front surface mirror and the normal mirror is shown in part (a) of Figure 2.3. By comparing part (b) and part (c) in Figure 2.3, it is obvious to see the improvement achieved by using the front surface mirror to eliminate the ghost effect.

In order to provide enough light for illuminating wound, the light emitting diode (LED) lights are located at the back side of the box (we have tried to put the LED lights at different locations, from which we found that the light condition is optimal for foot imaging when the LEDs are placed at the back side). For the actual image capture box, we use white acrylic material to manufacture the box to further enhance the brightness of interior. The actual product and the sample image taken with this box are shown in part (a) of Figure 2.4.



(a)

(b) *(c)*

Figure 2.3 Ghost image caused by the normal second surface mirror. (a) ghost image optical path; (b) ghost image using the normal mirrors for the box; (c) improvement by using the front surface mirrors

(a) *(b)*

Figure 2.4 Image capture box illustration; (a) actual product of the image capture box; (b) wound image captured using the warm LED light

In the new version of design, we use warm white LED lights which provide light that essentially resembles sunshine. The difference when compared with using the cool white LEDs can be clearly seen in part (b) of Figure 2.4. It is estimated that the manufacturing price for each box is within 100 US dollars.

2.1.3 Geometric Distortion Measurements

Since we used a pair of straight angled mirrors to image the entire foot region, we need to verify that there is no obvious geometric distortion across the entire image plane. For this test, we prepared a grid of black and white squares on the white paper as shown in Figure 2.5, where the physical side dimension of each square was 15 mm. The grid paper was placed over the opening of the Image Capture Box, and an image was captured, as shown as in part (a) of Figure 2.6. Then we selected 9 black squares at different locations on the image plane, marked these squares in red using Photoshop and numbered them 1 - 9, as shown in part (b) of Figure 2.6,. After calculating the area size for each square (the detailed area calculation method will be introduced in the next paragraph), we can determine the geometric distortion by comparing the areas of squares at the edge of the image field with the area of the square in the middle. Because we use the image capture box which provides fixed object-to-image distance, we can simply convert the area size in terms of pixel to actual unit by multiplier a constant scale factor to the former one.

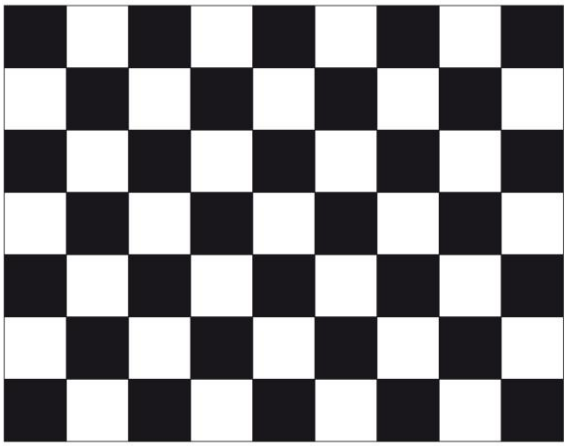
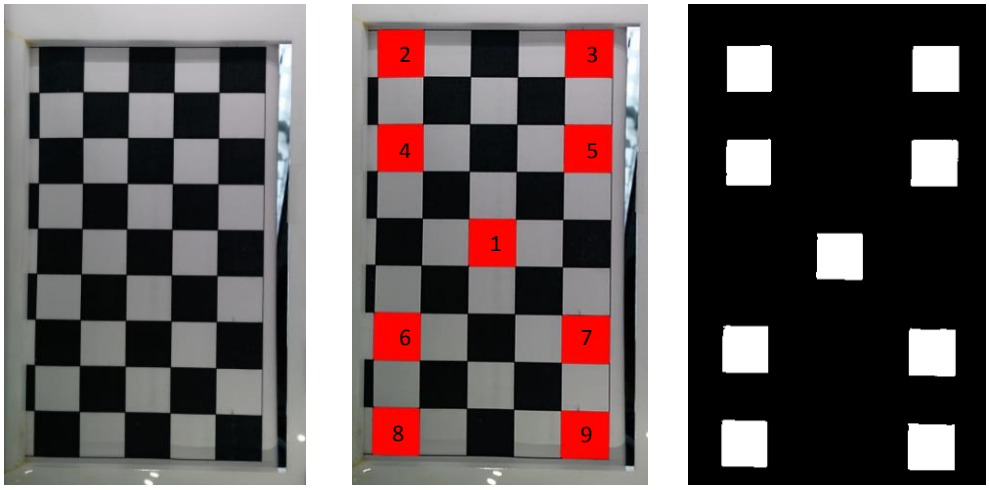


Figure 2.5 A grid of black and white squares

To calculate the square area, we generate a binary image with the 9 selected rectangles highlighted in white. Then we apply the mean shift based segmentation algorithm, which will be introduced in details in Chapter 3, to segment the binary image into 10 different regions. 9 of them represent the 9 rectangles and the other one stands for the black background region. In this case, the area size for each rectangle can be easily calculated by counting the number of pixels in each region as shown in part (c) of Figure 2.6. The

results are shown in Table 2.1, and we can see from these results that largest deviation from each rectangle area to the middle rectangle is less than 3% percent of the mean area size. Even when considering the possible inaccuracy introduced by the square patch delineation using Photoshop, we may conclude that the geometric distortion is small enough to neglect.



(a) *(b)* *(c)*

Figure 2.6 Geometric distortion estimation for images captured with the image capture box: (a) original image of a black-white grid sheet; (b) five selected rectangle areas marked in red for geometric distortion detection; (c) region area determination results for all rectangle areas.

Table 2.1 Geometric distortion estimation results

Patch ID	*Physical dimension*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
Area in pixels	*N/A*	*2622*	*2590*	*2648*	*2638*	*2676*	*2639*	*2692*	*2622*	*2650*
Area in mm^2	*225*	*219*	*216*	*220*	*220*	*222*	*220*	*224*	*219*	*220*
Percen-tage error relative to Patch 1	*+2.7*	*0*	*-1.4*	*+0.5*	*+0.5*	*+1.4*	*+0.5*	*+2.3*	*0*	*+0.5*

2.2 Super-pixel Segmentation Algorithms

2.2.1 Segmentation Algorithm Categories and Overview

As introduced to [38], image segmentation is an important low level computer vision operation, which clusters an image into a number of homogeneous regions in terms of selected features. The wound image segmentation is a significant step which provides the foundation of the following foot outline and wound boundary determination. According to [38], the existing segmentation algorithms can be divided into the following categories: *region based segmentation*, *edge based segmentation*, *feature space clustering* and *energy minimization* (graph based).

For region based segmentation algorithms, it is usually assumed that the values for pixels in a small patch always are quite similar. During the segmentation process, a given pixel is said to belong to a cluster if a pre-defined similarity criterion is satisfied between the cluster center and the pixel. The number of cluster centers needs to be specified as well. The most widely used region based segmentation algorithms are as follows: seeded region growing algorithm [39], unseeded region growing algorithm [40], JSEG algorithm [41] and fast scanning algorithm [42]. According to [38], the segmentation performance of all region based methods is greatly inferenced by noise and by similarity criterion selection.

Feature space analysis based algorithms are in an iterative fashion, the cluster is represented by using a centroid, and in each iteration each pixel is assigned to one centroid based on a certain similarity measure. According to [43], the feature space analysis based algorithms can be further divided into two categories: hierarchical (this kind of algorithm can be further categorized into agglomerative and divisive algorithms) and partitional segmentation. The hierarchical algorithms suffer from high computational cost while the partitional algorithms, in spite of the relatively light computation requirement, suffer from their parametric nature and the assumption about the cluster shape (the assumed cluster model is described by a set of parameters, such as the Gaussian distribution model). As argued in [44], "*arbitrarily structured feature spaces can be analyzed only by nonparametric methods since these methods do not have embedded assumptions"*. Hence, the mean shift and quick shift algorithm [44], [45], one

of the density estimation based non-parametric approaches, is widely applied to low-level image segmentation tasks, due to its high efficiency and freedom from any *a-priori* knowledge based assumptions.

For edge based segmentation algorithms, the main basis is the gradient information. One widely used algorithm, called watershed algorithm [46], tries to find the "watershed lines" which are viewed as the "separating lines" between adjacent regions. The watershed algorithm suffers from over-segmentation and high computational cost. Recently, a new type of image segmentation algorithms has been introduced which is based on energy functional optimization. According to [38], this segmentation category can be further divided into two groups based on the energy functional they are using: 1) functional defined on a continuous contour or surface (the level set based algorithm is a typical example) and 2) functional defined on a set of discrete variables (the graph cut based algorithm is a typical example, which is also called the labeling optimization method) [47]. According to [48], the labeling optimization method ensures a globally optimal solution and can be efficiently addressed by classical graph cut based algorithms. However, when dealing with a high resolution image, the graph based algorithm can become very computational demanding if the number of nodes grows rapidly [47], [48].

In the following several sections, we will introduce three widely used image segmentation algorithms in detail and compare their performance by applying them on our wound image dataset. First, the mean shift algorithm which is the most popularly applied feature space analysis based segmentation approach will be introduced. Then, we will describe how JSEG algorithm works, which is one of the most typical region-based segmentation algorithms. Finally, a detailed introduction of the SLIC algorithm, a state-of-art segmentation algorithm which can be treated as the combination edge-based and region-based approaches will be given.

2.2.2 Mean Shift Algorithm

The mean shift algorithm belongs to the density estimation based non-parametric clustering methods, in which the feature space can be considered as the empirical probability density function of the represented parameter. This type of algorithms adequately analyzes the image feature space (color space, spatial space or the

combination of the two spaces) to cluster and can provide a reliable solution for many vision tasks [44]. In general, the mean shift algorithm models the feature vectors associated with each pixel (e.g., color and position in the image grid) as samples from an unknown probability density function $f(\mathrm{x})$ and then finds clusters in this distribution. The center for each cluster is called the mode [49]. Given n data points x_i, $i = 1,...,n$ in the d-dimensional space R^d, the multivariate kernel density estimator is shown as below [44].

$$f_{h,K}(x) = \frac{c_{k,d}}{nh^d} \sum_{i=1}^{n} k(\left\| \frac{x - x_i}{h} \right\|^2) \qquad \textit{Equation 2.1}$$

where h is a bandwidth parameter satisfying $h > 0$ and $c_{k,d}$ is a normalization constant [44]. The function $k(\mathrm{x})$ is the profile of the kernel defined only for $x \geq 0$ and $\|.\|$ represents the vector norm. In applying the mean shift algorithm we use a variant of what is known in the optimization literature as multiple restart gradient descent. Starting at some guess at a local maximum y_k, which can be a random input data point x_i, the mean shift computes the density estimate $f(\mathrm{x})$ at y_k and takes an uphill step using the gradient descent method. The gradient of $f(\mathrm{x})$ is given as follows [44].

$$\nabla f(x) = \frac{2c_{k,d}}{nh^{d+2}} [\sum_{i=1}^{n} g(\left\| \frac{x - x_i}{h} \right\|^2)] \cdot m(x) \qquad \textit{Equation 2.2}$$

$$m(x) = \frac{\sum_{i=1}^{n} x_i g(\left\| \frac{x - x_i}{h} \right\|^2)}{\sum_{i=1}^{n} g(\left\| \frac{x - x_i}{h} \right\|^2)} - x \qquad \textit{Equation 2.3}$$

where $g(r) = -k'(r)$ and n is the number of neighbors taken into account in the 5 dimension sample domain. In our case, we use the Epanechinikov kernel [50], which makes the derivative of this kernel a unit sphere. Based on [44], we use the combined kernel function shown in eq. (2.5) where h_s and h_r are different bandwidth values for spatial domain and range domain, respectively. In [44], the two bandwidth values are referred to as spatial and range resolutions. The vector $m(x)$ defined in eq. (2.3) is called

the mean shift vector, since it is the difference between the current value x and the weighted mean of the neighbors x_i around x. In the mean-shift procedure, the current estimate of the mode y_k at iteration k is replaced by its locally weighted mean as shown below in eq. (2.4) [44].

$$y_{k+1} = y_k + m(y_k) \qquad \textit{Equation 2.4}$$

$$K_{hs,hr}(x) = \frac{C}{h_s^{\,2} h_r^{\,3}} k\left(\left\|\frac{x^s}{h_s}\right\|^2\right) k\left(\left\|\frac{x^r}{h_r}\right\|^2\right) \qquad \textit{Equation 2.5}$$

This iterative update of the local maxima estimation will be continued until the convergence condition is met. In our case, the convergence condition is specified as the Euclidean length of the mean shift vector that is smaller than a preset threshold. The threshold value for the mean shift iteration is the same for the task of locating the foot in the full image and for locating the wound within the foot boundary.

After the filtering (also referred to as the mode seeking) procedure above, the image is usually over-segmented, which means that there are more regions in the segmentation result than necessary for wound boundary determination [51]. To solve this problem, we have to merge the over-segmented image into a smaller number of regions which are more object-representative based on some rules. In the fusion step, extensive use was made of region adjacency graphs (RAG) [52], [53]. The initial RAG was built from the initially over-segmented image, where the modes are the vertices of the graph and where the edges are defined based on 4-connectivity on the lattice. The fusion was performed as a transitive closure operation [54] on the graph, under the condition that the color difference between two adjacent nodes should not exceed h_f , which is regarded as the region fusion resolution.

2.2.3 Simple Linear Iterative Clustering Algorithm

Similar to the mean shift algorithm, the SLIC clustering takes the spatial continuity into account by expanding the original 3D color range space to 5D space, by including two spatial components. By default, we only need to specify one parameter *k*, the desired number of approximately equally sized super-pixels. Unlike the random initialization in typical K-Mean clustering method, the SLIC algorithm places *k* initial cluster centers

uniformly on the image grid with *S* pixels apart between each pair. It is easy to verify that $S = \sqrt{N/k}$. In this case, we can expect the segmented super-pixels are in approximate equal size. In the assignment step, each pixel is assigned to the closest cluster center, which is similar to the traditional K-Mean process. However, instead of computing the distance between the target pixel with all cluster centers, SLIC only computes distances from each cluster centers to pixels within $2S \times 2S$ neighborhood region. This narrowing down of the search region will largely reduce the number of distance calculation and then accelerate the algorithm. However, it was pointed out in [37] that this speeding up is allowed only when a special distance measure *D* is introduced, as in eq. (2.1). The novel combination of the distances from color space and spatial space can effectively prevent the variance of boundary adherence (the boundary of the resulting super-pixels are more consistent with the edge information presented in the image) due to the change of super-pixel size [37].

$$d_c = \sqrt{(L_i - L_j)^2 + (a_i - a_j)^2 + (b_i - b_j)^2}$$
$$d_s = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \qquad \textit{Equation 2.6}$$
$$D = \sqrt{d_c^2 + (\frac{d_s}{S})^2 m^2}$$

where $(L_i, a_i, b_i, x_i, y_i)$ and $(L_j, a_j, b_j, x_j, y_j)$ represent two points in the 5D color and spatial space. d_c and d_s represent the distance in color subspace and spatial subspace, respectively. As mentioned before, S stands for the approximate super-pixel size. The parameter *m* is used to weight the relative importance between color similarity and spatial proximity. If *m* is relatively large, the resulting super-pixels have good boundary adherence, but the size and shape is somehow irregular. It is recommended in [55] that the *m* should be chosen from the range $[1, 40]$ if the algorithm is performed in CIE Lab color space. In our case, we empirically set *m* as 16 and *S* as 30. Then, the following process is much similar to K-mean algorithm: 1) replace the cluster center with the mean of all pixels in that cluster; 2) calculate the cluster center change and decide whether it converges; and 3) reallocate isolate pixels to nearby cluster center to enforce connectivity. According to [37], this iterative segmentation is guaranteed to converge after a certain

number of iterations. In practice, to improve efficiency, we can terminate the iteration when the cluster center change is smaller than a pre-specified threshold.

2.2.4 JSEG Algorithm

According to [41], the most prominent advantage of the JSEG algorithm is that there is no need for manual parameter specification. There are three stages for the segmentation process: 1) "*color space quantization*" to reduce the number of distinct colors, 2) "*J-Image Calculation*" and 3) "*region growing*". Stage 2 and 3 can be viewed as the spatial segmentation.

In Stage 1, an unsupervised color quantization algorithm using peer group filtering , a non-linear algorithm for image smoothing and noise removal [56], with good perceptual information preserving property is applied. The key finding for this algorithm is that the human visual perception has higher sensitivity to intensity variation in homogeneous regions than in textured patches. In this stage, the spatial information for each pixel is not considered. Finally, we will replace the original color value for each pixel with a class number, similarly to the K-Means algorithm [36]. The total color class number C is usually set as an integer in the range from 10-20. Hence, a class-map of the image is formed.

At the beginning of spatial segmentation stage, a novel notion of "J-Images" has been introduced. We assume Z is the set of all N pixels in the image. Each $z \in Z$ is a 2D vector (x, y) representing the spatial position for the current pixel variable z. Let C be the total number of classes and we calculate the J value for the entire image by following formulas.

$$J = \frac{S_T - S_W}{S_W} \quad \textit{Equation 2.7}$$

$$S_T = \sum_{z \in Z} \|z - m\|^2 \quad \textit{Equation 2.8}$$

$$S_W = \sum_{i=1}^{C} \sum_{z \in Z} \|z - m_i\|^2 \quad \textit{Equation 2.9}$$

According to [41], S_W is defined as the total variance of the position vectors in the same color class S_T is defined as the total variance of all data points m and m_i are defined as

the mean of all vectors in set Z and vectors belonging to each class i, respectively. The concept of J values is based on the Fisher's linear discrimination approach [57]. Furthermore, the average $\bar{J}$ is defined as below.

$$\bar{J} = \frac{1}{N}\sum_{k} M_k J_k \qquad \textit{Equation 2.10}$$

where J_k is the J value computed on region k, M_k is the pixel-wise size of the region k. N represents the total number of data points. According to [41], given a specified region number, the lower the value of $\bar{J}$, the better the segmentation. Hence, we can find the optimal segmentation by minimize the $\bar{J}$ value. However, due to 10^6 different possible segmentations, the direct minimization of $\bar{J}$ is not tractable. Therefore, it is preferred to use local $\bar{J}$ value to measure the color uniformity of certain local regions, such as a neighborhood centered a candidate pixel. In this way, we can replace the pixel value with the local $\bar{J}$ calculated over the neighborhood region in difference size (different scales). The resulting image is called J-image, where the valleys represent the uniform region and the hills stand for boundaries. In practice, we will calculate several J-images at different scales.

Finally, making use of the J-image based measure value, an iterative region growing approach can be employed to find the approximately optimal image, in a coarse-to-fine fashion. As stated in [41], the region growing process has two steps: 1) the seed location determination and 2) region growing starting from the seed points. This process will be repeated until no finer segmentation has been archived.

2.2.5 Evaluation of Wound Image Segmentation of Different Algorithms

It is argued in [7] that experimental results deny the feasibility of using manually generated ground truth to measure the performance of image segmentation algorithms. Furthermore, for segmentation algorithms with high complexity, it makes more sense to apply goodness evaluation methods, which assess the performance on a number of test images by using some desirable properties of segmented images as goodness measures. Hence, we use the following evaluation method used in [7] to assess the SLIC algorithm [37], mean-shift algorithm [44] and JSEG algorithm [41] mentioned earlier. The

evaluation is based on two measures: region uniformity measurement U_α and region contrast measurement C_α [7], [58]. They are defined as in eq. 2.11 and eq. 2.12. In both equations, f_i is the color vector in CIE lab space for pixel i in region R_j, $\overline{f}_j$ is the average value of all f_i in R_j. A_j .is the area of the region R_j, $f_{\min}$ .and $f_{\max}$ are the maximum and minimum values in this region. Based on the goal of image segmentation (to achieve maximum uniformity in each resulting super-pixel), we can see that the larger the value of U_α, the better the uniformity is achieved in the current segmentation result, α is the significant area where the measure is evaluated. In eq. 2.12, c_j is the contrast measure in region R_j, v_j is the weight assigned to region. R_j. Based on [7], v_j is modeled as the simple linear contribution of region R_j to the total area of image. c_{ij} is the contrast between region R_j and R_i. Finally, p_{ij} is the adjacency parameter as the ratio of the common perimeter of R_j to R_i. More details about the definition and derivation of these formulas can be found in [7], [58].

$$U_\alpha = 1 - \frac{2\sum_{R_j \in \alpha} \sum_{i \in R_j} \left\| f_i - \overline{f_j} \right\|}{\sum_{R_j \in \alpha} A_j \left\| f_{\max} - f_{\min} \right\|} \qquad \textit{Equation 2.11}$$

$$C_\alpha = \frac{\sum_{R_j \in \alpha} v_j c_j}{\sum_{R_j \in \alpha} v_j}$$

$$c_j = \sum_{AdjR_i} p_{ij} c_{ij} \qquad \textit{Equation 2.12}$$

$$c_{ij} = \frac{\left\| \overline{f_i} - \overline{f_j} \right\|}{\left\| \overline{f_i} \right\| + \left\| \overline{f_j} \right\|}$$

We applied the super-pixel segmentation evaluation scheme proposed in [7]. As shown in Table 2.2, we calculated the mean and standard variance of U_α and C_α measures for three super-pixel segmentation algorithms (with appropriate parameter tuning) applying on 100 sample images (significant area α in eq. 2.11 and 2.12 is set as the entire image area). In Table 2.3, the same indicators are calculated with α containing the entire wound area

manually delineated by three experienced clinicians (the details about these ground truth generation process will be discussed in details in Section V). In Table 2.4, the efficiency of these three segmentation approaches is evaluated based on the Nexus 5 smartphone platform with Quad core CPU, 2.3 GHz Krait 400, 2GB RAM. We can see from Table 2.1 and Table 2.2 that the mean value of region uniformity is generally large, and the mean value of region contrast is generally small. The results fulfill our expectation since we deliberately tuned the parameters to obtain a relatively large number of super-pixels. In addition, the standard deviation is generally small for all cases. Comparing these two tables, we can see that the uniformity in the wound area is smaller and region contrast is higher than the entire image. It tells us that the nature of wound area is more complicated than the background and healthy skin. Comparing the corresponding columns in the two tables, we find that SLIC provides the best performance balance on both the uniformity and contrast measures. However, the difference in the performance of different algorithms is not large. The average region uniformity U_α is larger than 0.86 and the average region contrast C_α is smaller than 0.2 for all three methods. Hence, we may conclude based on these findings that all three super-pixel segmentation algorithms can achieve good performance given carefully parameter tuning. As mentioned earlier in Section 2.2.3, we only need to tune one parameter for SLIC algorithm. In contrast, at least 2 and 3 parameters need to be tuned in JSEG shift and mean shift algorithms, respectively [41], [44]. Furthermore, SLIC has the best performance in terms of efficiency as shown in Table 2.4.

Table 2.2 Statistics of region uniformity and contrast measures on the CIE Lab space with the significant area containing the entire image for three different super-pixel segmentation methods

Method	*Mean shift*		*JSEG*		*SLIC*	
	μ	σ	μ	σ	μ	σ
U_α	*0.867*	*0.022*	*0.878*	*0.051*	*0.873*	*0.032*
C_α	*0.147*	*0.008*	*0.166*	*0.019*	*0.099*	*0.027*

Table 2.3 Statistics of region uniformity and contrast measures on the CIE Lab space with the significant area containing the manually delineated wound area for three different super-pixel segmentation methods

Method	*Mean shift*		*JSEG*		*SLIC*	
	μ	σ	μ	σ	μ	σ
U_α	*0.692*	*0.109*	*0.769*	*0.162*	*0.748*	*0.174*
C_α	*0.217*	*0.058*	*0.233*	*0.084*	*0.184*	*0.061*

Table 2.4 Computation time results evaluated on Nexus 5 smartphone platform for three different super-pixel segmentation algorithms

Method	*Mean shift*	*JSEG*	*SLIC*
Computation time	*12.7s*	*9.7s*	*7.2s*

2.3 Conclusion

In this chapter, we first designed an image capture box to facilitate the foot ulcer photographing process. Then, we briefly introduced the categorization of super-pixel segmentation algorithms and finally focused on three most widely used approaches. Finally, we applied the region uniformity and contrast based measures to evaluate the three segmentation algorithms on 100 sample wound images.

Based on the evaluation results, we can conclude that all three segmentation approaches described in this chapter provide promising super-pixel segmentation with only slight differences on performance. The SLIC algorithm gives the best tradeoff on region uniformity and contrast, as well as the best efficiency. However, it seems easier to solve the over-segmentation problem by region-merge with the mean-shift based structure since there are plenty of previous works focusing on this topic. Hence, in Chapter 3, we choose the mean shift algorithm since the purpose is to find the foot area based on the region-merging result of super-pixels grid. In Chapter 4 and 5, we need to calculate the feature vector for each super-pixel and generate final wound area by grouping all super-pixels classified as “wound” together. Therefore, we applied the SLIC algorithm for super-pixel segmentation due to its good boundary adherence property and high time performance.

Chapter 3

Wound Boundary Determination (Non-Machine Learning Based)

As mentioned in Chapter 1, wound boundary determination is the most important step in the entire wound analysis flow, since the wound area, the most significant indicator for wound healing assessment, can be directly calculated based on the determined boundary. Besides, the wound boundary provides the foundation for the subsequent color based tissue segmentation and healing status evaluation. As discussed earlier, most of the previous works mainly dealt with the wound images containing the wound area and limited amount of surrounding normal tissue. One of the main goals of our application is patients' self-management by our automatic wound analysis system, with assistance of the image capture box. This approach is to acquire the wound image results in images where the wound area only occupies a small portion of the entire image and where the image may contain background regions outside the foot image. These characteristics make the wound boundary determination more complicated.

In this chapter, three non-machine learning based wound boundary determination approaches, based on widely used image segmentation algorithm, will be proposed and evaluated. First, it is nature to come up with a method by putting an initial curve which encloses the wound area and driving the curve to contract until it reaches the wound boundary. The level set algorithm introduced in [59] provides us a partial differential equation based mathematical tool to implement the method mentioned above. Instead of employing the standard level set model, we applied the distance regularized level set evolution (DRLSE) and a narrow-band implementation [60] to accelerate the curve evolution. The second approach is based on mean shift image segmentation algorithm [44]

and then applies the largest connect component detection method on the binary foot image to determine the wound boundary within the foot outline. In the third approach, we augment the mean shift image segmentation algorithm by determining the wound boundary by analyzing the internal and external boundaries of the foot outline. Based on the wound location, one of two different algorithmic paths is selected. In Section 3.1-3.3, we will introduce these three approaches, respectively. Finally, in Section 3.4, the comparison of these three approaches on sample wound images will be presented.

3.1 Level Set Based Wound Boundary Determination Approach

3.1.1 Theoretical Foundation

The wound boundary determination approach is required to perform well for a wide range of wound images. We chose the level set based algorithm to determine the wound boundary due to following reasons: 1) the level set based algorithm can achieve relatively fine resolution, which can result in smooth wound boundary; 2) the level set based algorithm is capable of handling of slightly tilted lines and corners, which is a desirable characteristic since the shape of wounds is often irregular; 3) the level set based algorithm allows precise and easy calculation of surface normal vectors, which can improve the curve evolution efficiency [61]–[63]. Furthermore, since joining surfaces is handled implicitly, the wound contour will be self-adaptive to the topological change during the evolution. This means that the implicit contour will accurately follow shapes that change topology, for example when a shape splits into two, develops holes, or the reverse of these operations [62], [63]. Finally the property of the level set algorithm allows direct numerical computation on image grids. Because of these features, level set based algorithms have been widely used in image segmentation.

The original level set methods for capturing dynamic contours was introduced by Osher and Sethian [64]. The basic idea is to use the zero level set of a higher dimensional function to represent the target contour. This higher dimensional function is called the *level set function* (LSF), $\phi(x, y, t)$. In this case, finding the contour is accomplished by the evolution of the LSF from a random initial state to convergence on the target boundary. Caselles and Malladi [61], [62] first brought the level set idea to image

processing and computer vision areas. The most basic curve evolution equation is presented in a partial differential equation (PDE) as given in eq. (3.1) [61]–[64].

$$\frac{\partial \phi}{\partial t} = -F\left|\nabla \phi\right| \quad \textit{Equation 3.1}$$

where F is the speed function that controls the motion of the contour and ∇ is the gradient operator. We assume that the curve movement is always proportional to F in a direction normal to the curve itself. In the image segmentation case, the speed function F is formulated as in eq. (3.2) [62].

$$F = -g_I(F_A + F_G) \qquad \textit{Equation 3.2}$$

where F_A causes the curve to uniformly expand or contract depending on the local geometry information. F_G is a function of local curvature, which keeps the propagating curve smooth and conforms to the edge distribution of the original image. The term g_I is a stopping term defined in eq. (3.3) [62].

$$g_I = \frac{1}{1+\left|\nabla(G_o \otimes I(x,y))\right|^2} \qquad \textit{Equation 3.3}$$

where $G_o \otimes I(x,y)$ filters the original image by a Gaussian smoothing filter to minimize noise. g_I is maximized when the gradient value is the smallest, which means the F is completely performed on the curve when no edge is detected. The gradient value increases greatly when the current contour is located in the vicinity of the boundaries, in which case g_I is approximating to 0. This cancels the influence of F and then the evolution of the contour will be stopped [62].

3.1.2 Distance Regularization Level Set Evolution (DRLSE)

The conventional level set methods suffer from the irregular boundary determination (*i.e.*, the detected boundary may be irregular and misshapen [60], and the LSF may lose the signed distance property ($\left|\nabla \phi\right|$=1), which is the basis for the successful evolution) [59], [60]. Hence, after each iteration we have to re-initialize the LSF. This re-initialization becomes a large computation burden and can reduce the accuracy of the zero level set evolution as well.

Li et al proposed a novel evolution function [60], which includes a distance-regularization term. This forces the LSF to maintain a desired shape and implies that the signed distance property of the LSF is preserved during the entire evolution process [60]. The evolution equation is modified from the gradient operation in eq. (3.1) to the expression in eq. (3.4).

$$\frac{\partial \phi}{\partial t} = \mu \Re(p, \phi) + \lambda L(g, \phi) + \alpha A(g, \phi) \qquad \textit{Equation 3.4}$$

where the term $\Re(p, \phi)$ is the distance-regularization term mentioned above, $L(g, \phi)$ is minimized when the zero level contour is located at the object boundary, in other words, $L(g, \phi)$ is the stopping term. The term $A(g, \phi)$ is used to speed up the motion of the zero level set contour in the evolution process, while p is the potential function and g is defined as in eq.(3.3).The controlling parameters α, λ and μ are set by users. Details regarding the function definitions are given in [60].

Finally, we formulate the numerical calculation of eq. (3.4) with a simple finite difference scheme [60], as shown in eq. (3.5).

$$\begin{aligned} L(\phi) &= \mu \Re(p, \phi) + \lambda L(g, \phi) + \alpha A(g, \phi) \\ \phi^{k+1}{}_{i,j} &= \phi^{k}{}_{i,j} + \Delta t L(\phi^{k}{}_{i,j}) \end{aligned} \qquad \textit{Equation 3.5}$$

where k represents the iteration times, Δt is the time step, $L(\phi^{k}{}_{i,j})$ represents the right side of eq. (3.4) while replacing ϕ with $\phi^{k}{}_{i,j}$, which denotes the LSF value at iteration time k on the position (i, j) in the image grid. As is commonly known, it is justified to apply the finite difference scheme to the discretization of the PDE like eq. (3.4). In order to reduce the computational demands, a narrow-band level set algorithm has been proposed to restrict the computation of the level set evolution to a narrow neighborhood area around the zero level set (usually 8-neighbor area surrounding the zero crossing points). According to [60], the narrow-band implementation of the model expressed in eq. (3.5) allows the use of a large time step, which will further speed up the evolution process.

Based on the discussion provided by Li et al. in [60], the algorithm structure can be summarized as follows:

- **Step 1: Initialization**: initialize curve and LSF as a binary step function, which has been set according to different locations (c for the pixels within the initial curve, -c outside the curve and 0 for the pixels on the curve, where c is a random positive integer and is set to 2 in our case). Then, construct the initial narrowband region;
- **Step 2**: **Update the LSF** as $\phi^{k+1}{}_{i,j} = \phi^{k}{}_{i,j} + \Delta t L(\phi^{k}{}_{i,j})$ just over the narrow band area;
- **Step 3: Update the narrow-band** based on the new zero-crossing points on the previous defined narrow band area;
- **Step 4: Assign values to new pixels in narrowband**, if $\phi^{k}{}_{i,j} > 0$, set $\phi^{k}{}_{i,j}$ to h, otherwise set it to –h, where h is a constant with a default value 3 (the same value of the width of the neighborhood area);
- **Step 5: Check the termination condition**. If it is satisfied, quit and return the curve. Otherwise, return to step 2.

3.1.3 Level Set Based Wound Boundary Determination

The narrow-band based DRLSE algorithm was introduced in Section 3.1.2. Based on the algorithmic structure presented in the previous section, the implementation details of the narrow-band DRLSE algorithm for wound boundary determination will be discussed here. The flowchart of the wound boundary determination process is shown in Figure 3.1.

The DRLSE can be implemented with a simple finite difference scheme on the image grid [60]. The evolution equation is given in as eq. (3.5). By iteratively updating the level set function, the implicit contour of the level set function will converge towards the actual wound boundary from its initial position.

Considering the complicated nature of a typical wound, the level set evolution process might likely be prematurely stopped by the internal boundary within the wound area if we were to place the initial curve inside the wound area and expanded it gradually. To avoid this problem, we choose to set the initial curve on the healthy skin area and make sure that the curve encloses the entire wound area. As the level set evolution continues, the initial curve will keep contracting and approach the actual wound boundary with little resistance, since the healthy skin color is much more uniform than the wound area. In our

case, a simple skin detection method based on color will be employed to locate the skin area. Afterwards, the initial curve will be set as the boundary of the skin area.

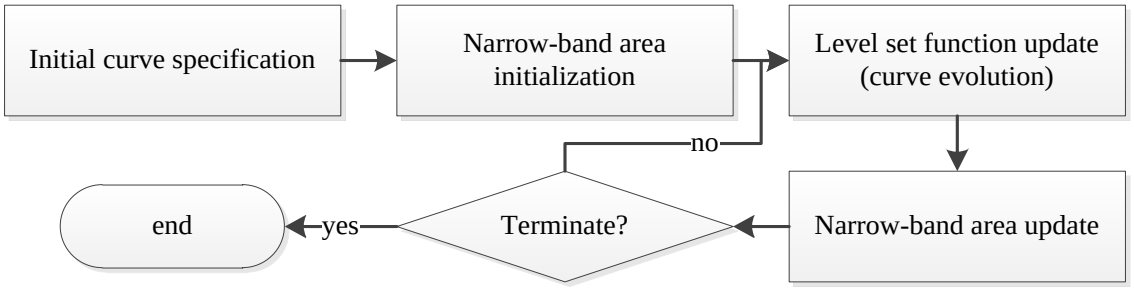


Figure 3.1 Flowchart of the wound boundary determination based on DRLSE algorithm

As discussion in [60], the narrow band area is defined as the union of the 3x3 neighborhoods of the zero crossing points of the level set function. We call a grid point (*i,j)* a zero crossing point if either $\phi_{i-1,j}$ and $\phi_{i+1,j}$ or $\phi_{i,j-1}$ and $\phi_{i,j+1}$ are of opposite signs [60]. After each update of the level set function, the narrow band area will be re-calculated accordingly.

After each iteration, we have to decide whether to stop the evolution process based on some rules. In our case, we will make this decision by checking the evolution rate R of the level set function defined as in eq. (3.6).

$$R = \frac{\left|N_k - N_{k-1}\right|}{N_{k-1}+1} \quad \textit{Equation 3.6}$$

where N_k represents the number of grid points (pixels) locating on the implicit contour of the current level set function, N_{k-1} represent the same number in the previous iteration. Hence, the numerator represents the absolute change of the level set function. Denominating it by N_{k-1}, we can obtain the changing rate of the level set function after the current update. The reason to add 1 to N_{k-1} is to ensure that the denominator never equates to zero. In our case, we consider a changing rate smaller than 0.001 as a sign of convergence, and we will use a changing rate of 0.001 as the criterion for stopping the evolution process. In order to control the overall processing time below certain limit, the evolution will be stopped if the iteration time exceeds a preset maximum number of iterations (this number will be set as 200 in our case) even the condition defined in eq. (3.6) is not satisfied. The actual wound boundary determination results using level set

based approach, together with the results from the two mean-shift based approaches introduced in Section 3.2 and 3.3, will be presented in Section 3.4.

3.1.4 GPU&CPU Collaborative Implementation of the Level Set Algorithm

In spite of the narrow-band optimization, the level set method is still too computationally demanding for direct implementation on a smartphone. Hence, we need strategies that permit the boundary detection of the wound image to be carried out in an acceptably short time. In our system, we implement the iterative LSF updating, which is the most computationally intensive part of the level set algorithm, on a GPU (Graphic Processing Unit), which is available on most of today's PCs and laptops. The major difference between CPU and GPU implementation is that the latter supports parallel computation to a much greater extent. Unlike the traditional, sequential approach to programming on CPUs, GPUs programs deal with complex parallel reductions to best utilize the available parallel resources [65].

The traditional narrow band method avoids unnecessary computation by only updating field elements near the level set curve. However, according to [65], even computations near the level set curve can be eliminated in regions where the level set field has locally converged. This observation motivates the method of tracking the active computational domain according to both the temporal and spatial derivatives of the level set field. Afterwards, the level set updating only happens on the pixels belonging to the specified active computational domain [65]. The active domain is updated before each iteration. The original version of this algorithm is based on the 3D image situation [65], but we have transformed it to operate in the 2D wound image situation.

In our system, the GPU programming is implemented with Open Computing Language (OpenCL), which is a framework for writing programs that execute across heterogeneous platforms, such as CPUs and GPUs. In our case, we view each pixel in the current active computational domain as the basic processing unit (called 'work item' in OpenCL). Then we maximize the usage of all the streaming cores at the same time by running a kernel (similar to the concept of function in C) on as many units as possible at the same time. The programming flow chart for modified level set algorithm based on OpenCL is shown in Figure 3.2. We can see from this figure that the host program is running on the CPU

including the functions of target image reading, edge calculation, platforms and memory initialization. In contrast, the actual level set algorithm is encapsulated in kernels running on GPU. All the kernels process only one pixel at a time. Besides, the intermediate processing results of kernels are stored on global memory on GPU and will not be read back to the host until the end of the algorithm. In this way, the memory access latency from different platforms can be reduced to the greatest extend.

3.2 Mean Shift Based Wound Boundary Determination Approach

3.2.1 Mean Shift Based Segmentation Algorithm

In this approach, we chose the mean shift algorithm proposed in [44] for several reasons. First, the mean shift algorithm takes into consideration the spatial continuity inside the image by expanding the original 3D color range space to 5D space, including two spatial components, since direct classification in 3D color range space only has been proved to be inefficient [44]. Second, a number of acceleration algorithms are available [52] . Third, for both mean shift filtering and region merge methods, the quality of the segmentation is easily controlled by the spatial and color range resolution parameters [44], [52]. Hence, the segmentation algorithm can be adjusted to different degrees of skin color smoothness by changing the resolution parameters. Finally, the mean shift filtering algorithm is suitable for parallel implementation since the basic processing unit is the pixel. In this case, the high computational efficiency of GPUs can be exploited.

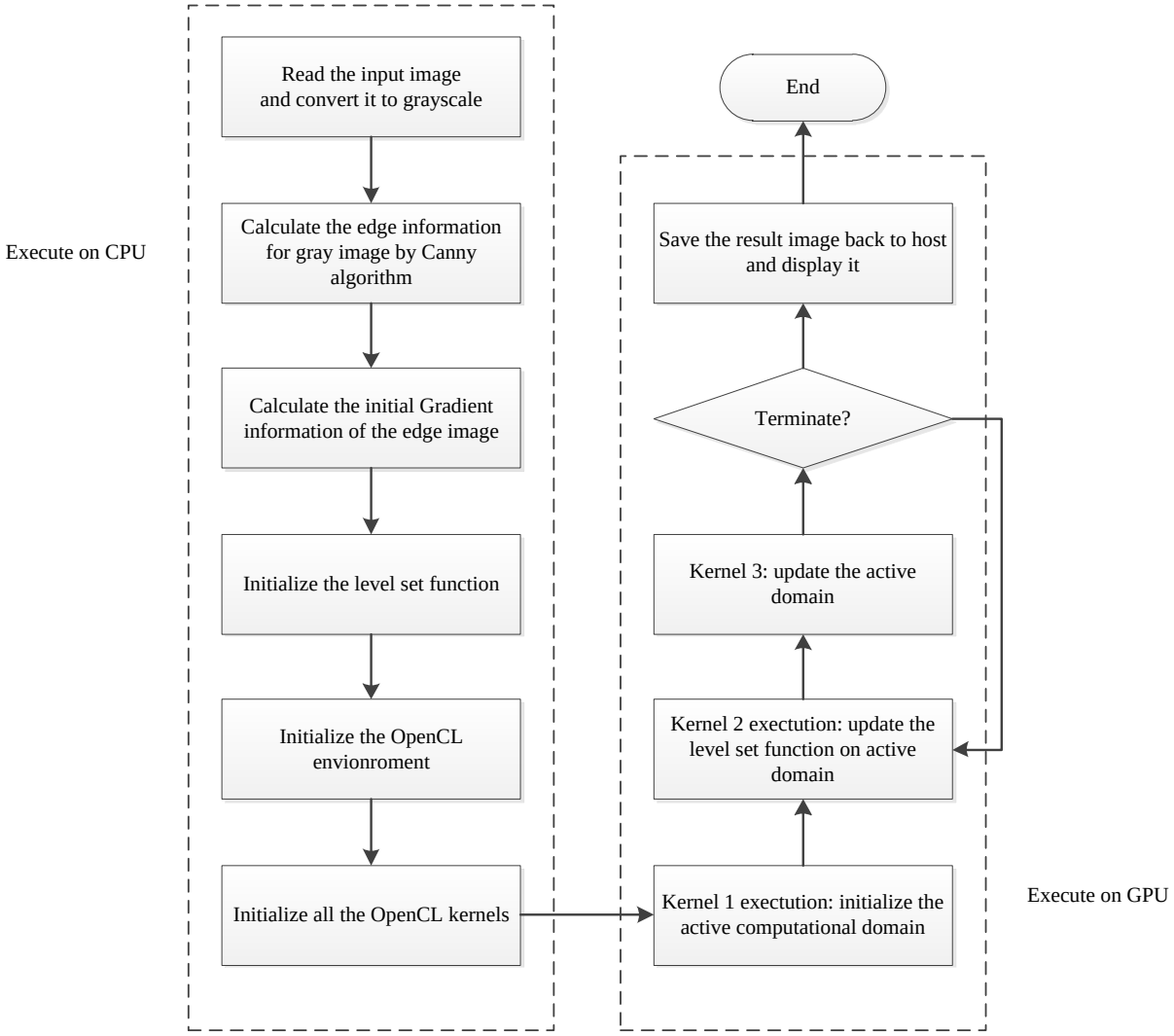
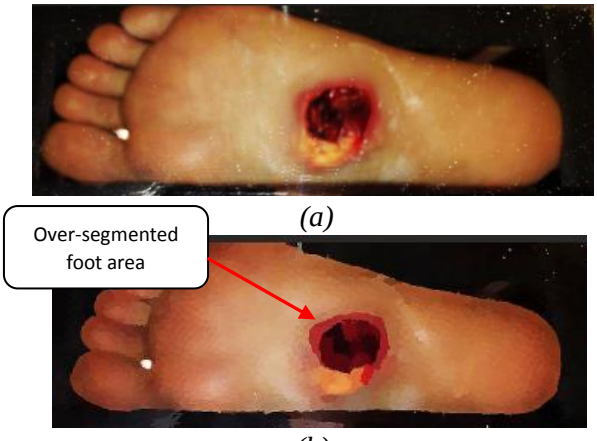


Figure 3.2 Programming flowchart of level set algorithm on GPU&CPU platform

The mean shift algorithm belongs to the category of density estimation based non-parametric clustering methods, in which the feature space can be considered the empirical probability density function of the represented parameter. This type of algorithms adequately analyzes the image feature space (color space, spatial space or the combination of the two spaces) and can provide a reliable solution for many vision tasks [44]. In general, the mean shift algorithm models the feature vectors associated with each pixel (e.g., color and position in the image grid) as samples from an unknown probability density function $f(x)$ and then finds clusters in this distribution. Details about the mean shift based image segmentation algorithm were introduced in Chapter 2. This iterative

update of the local maxima estimation as shown in eq. (2.10) will be continued until the convergence condition is met. In our case, the convergence condition is specified as the Euclidean length of the mean shift vector that is smaller than a preset threshold. The threshold value for the mean shift iteration is the same for the task of locating the foot in the full image and for locating the wound within the foot boundary.

After the filtering (also referred to as the mode seeking) procedure above, the image is usually over-segmented, which means that there are more regions in the segmentation result than necessary for wound boundary determination [51]. To solve this problem, we have to merge the over-segmented image into a smaller number of regions which are more object-representative based on some rules. In the fusion step, extensive use was made of Region Adjacency Graphs (RAG) [52], [53]. The initial RAG was built from the initial over-segmented image, the modes being the vertices of the graph, where the edges were defined based on 4-connectivity on the lattice. The fusion was performed as a transitive closure operation [54] on the graph, under the condition that the color difference between two adjacent nodes should not exceed h_f , which is regarded as the region fusion resolution. The mean shift filtering and region fusion results of a sample foot wound image (part (a) in Figure 3.3) are shown in part (b) and (c) in Figure 3.3, respectively. We can see that the over-segmentation problem in part (b) is effectively solved by region fusion procedure. From the region fusion result in part (c), the foot boundary is readily determined by a largest connected component detection algorithm, which will be introduced in the next Section. A C++ based implementation method of the mean shift algorithm can be found in [52].



(a)

(b)

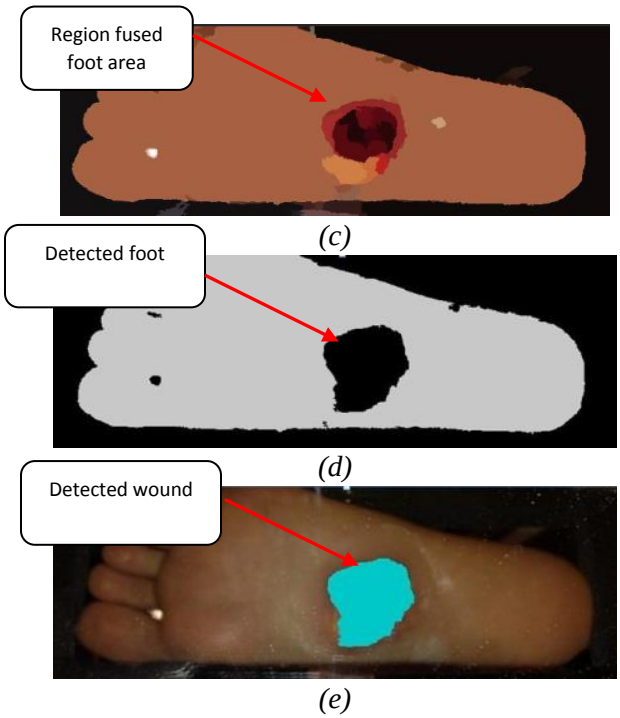


Figure 3.3 Mean shift based image segmentation sample result. (a) Original image. (b) Mean shift filtered image. (c) Region fused image. (d) The foot boundary detection result. (e) Wound boundary determination result. Note that we artificially increased the brightness and contrast of the images in this figure to highlight the over-segmentation in part (b) and to better observe the region fusion result in part (c).

3.2.2 Wound Boundary Determination

Because the mean shift algorithm only manages to segment the original image into homogeneous regions with similar color features, an object recognition method is needed to interpret the segmentation result into a meaningful wound boundary determination that can be easily understood by the users of the wound analysis system. A standard recognition method relies on known model information to develop a hypothesis, based on which a decision is made whether a region should be regarded as a candidate object, *i.e.*, a wound. A verification step is also needed for further confirmation. Because our wound determination algorithm is designed for real time implementation on the smartphones with limited computational resources, we simplify the object recognition process while ensuring that recognition accuracy is acceptable.

Our wound boundary determination method is based on three assumptions. First, the foot image contains little irrelevant background information. In reality, it is not a critical

problem as we assume that the patients and/or caregivers will observe the foot image with the wound on the smartphone screen before the image is captured and ensure that the wound is clearly visible. Second, we assume that the healthy skin on the sole of the foot is a nearly uniform color feature. Finally, we assume that the foot ulcer is not located at the edge of the foot outline. These are reasonable assumptions for our initial system development and appear consistent with observations made initially from a small sampling of foot images. Later in this chapter, ways to relax these assumptions will be investigated. Based on these assumptions, the proposed wound boundary determination method is illustrated as in Figure 3.4, and explained below.

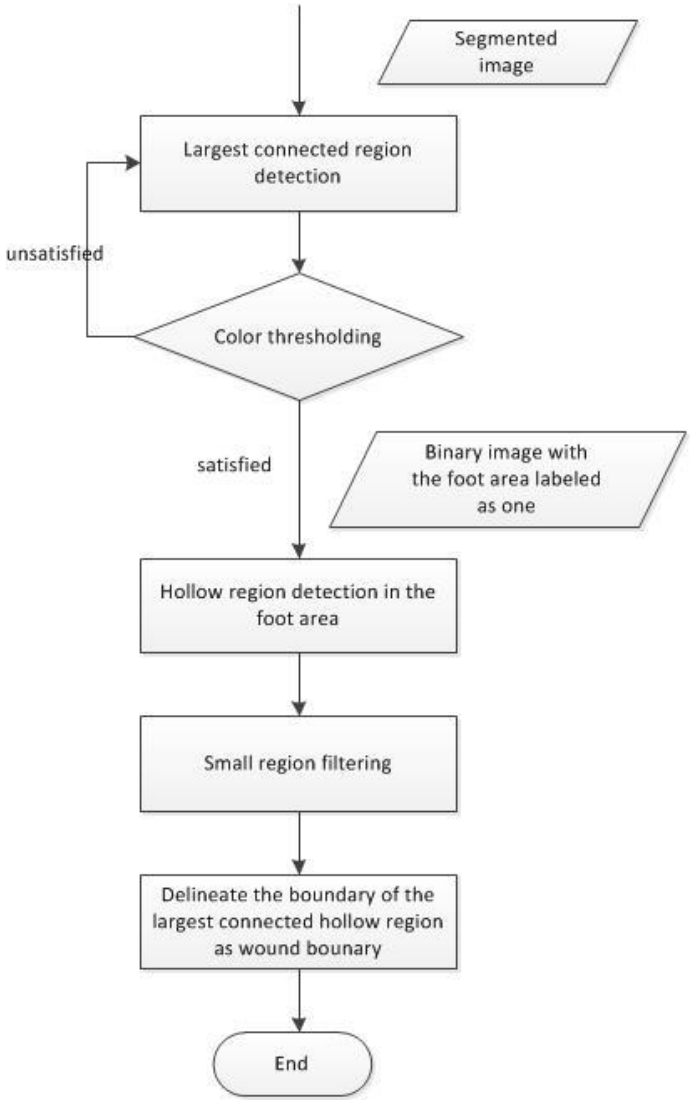


Figure 3.4 Largest connected component detection based wound boundary determination method flowchart

The ***Largest connected component detection*** is first performed on the segmented image, using the fast largest connected component detection method introduced in [66] including two passes. In this first pass, temporary labels are assigned to each pixel and a class equivalence map is constructed. In the second pass, each temporary label is replaced by the smallest label of its equivalent class. In foot ***Color thresholding***, the color feature extracted in the mean shift segmentation algorithm of this component is compared with an empirical skin color feature by calculating the Euclidean distance between the color vector for the current component and the standard skin color vector from the Macbeth color checker [67]. If the distance is smaller than a pre-specified and empirically determined threshold value, we claim that the foot area has been located. Otherwise, we iteratively repeat the largest component detection algorithm on the remaining part of the image while excluding the previously detected components until the color threshold condition is satisfied.

After the foot area is located, we generate a binary image with pixels that are part of the foot labeled "1" (white) and the rest part of the image labeled "0" (black). The result of the foot area determination executed on the region fusion image shown in part (c) in Figure 3.3 is presented in part (d). To determine the actual wound boundary, the system locates the black part labeled as "0" within the white foot area (***Hollow region detection in the foot area***). Here we use the simple line-scanning based algorithm illustrated in Figure 3.5 and explained below.

In this wound boundary determination algorithm, each row in the binary image matrix is regarded as the basic scanning unit. In each row, the parts labeled as "0" in the detected foot region are regarded as the wound part. After every row is scanned, the wound boundary is determined accordingly. Because some small outlier regions may also be generated due to the local color variation of the skin, a ***Small region filtering*** procedure is needed to identify only the largest black region as the wound. A sample of the wound boundary determination result is shown in part (e) in Figure 3.3.

After the best estimate of the wound boundary is obtained, we analyze the wound area within the boundary using a wound description model. Many methods for assessing and

classifying open wounds require advanced clinical expertise and experience, and specialized criteria have been developed for diabetic foot ulcers [21] [32].

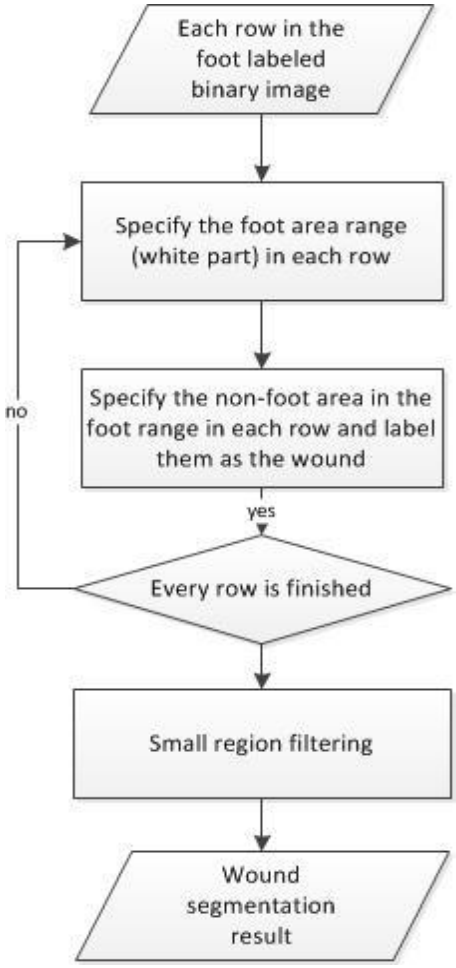


Figure 3.5. Wound part detection algorithm flowchart

3.2.3 GPU based Implementation of the Mean-Shift Algorithm

Because the CPUs on smartphones are not nearly as powerful as those on PCs or laptops, an optimized parallel implementation based on GPUs is critical for the most computationally demanding module in the algorithm structure. For current Android based smartphones, such as Nexus 4 from Google, the GPUs (Adreno 320) have good computational power (up to 51.2 G Floating Point Operation per Second (FLOPS)) [68]. As the experimental results in Section V shows, the hybrid implementation on both CPUs and GPUs can significantly improve the time performance for algorithms that are suitable for parallel implementation.

Since our wound analysis is implemented on Android smartphones, we take advantage of the Android APIs for GPU implementations. In our case, we use the *Renderscript*, which offers a high performance computation API at the native level written in C (C99 standard) [69] and which gives the smartphone apps the ability to run operations with automatic parallelization across all available processor cores. It also supports different types of processors such as the CPU, GPU or DSP. In addition, a program may access to all of these features without having to write code to support different architectures or a different number of processing cores [70].

On the Nexus 5 Android smartphone, we implemented the mean shift based segmentation algorithm on both the Adreno 320 GPU and the Quad-core Krait CPU using Renderscript. The algorithm implementation flow is shown as in Figure 3.6 and is explained below. Our implementation scheme is similar to the ones used in [71].

The processing steps ***Color space transformation***, ***Color histogram generation and discretization***, and ***Weight-map generation*** are all implemented on the CPU (these steps belongs to the mean shift filtering module introduced in Section 3.2.2). Afterwards, all the needed data are moved to the global memory on the GPU. This data includes the original image data in CIE Lab color space, the discretized color histogram for all three channels in this color space and the weight-map. The weight-map combines the edge information into the image segmentation to further improve the accuracy [52]. Because the mean shift based segmentation algorithm operates on each pixel of an image, and the computation, which takes place at each pixel, is independent of its distant surroundings, it is a good candidate for implementation on a parallel architecture.

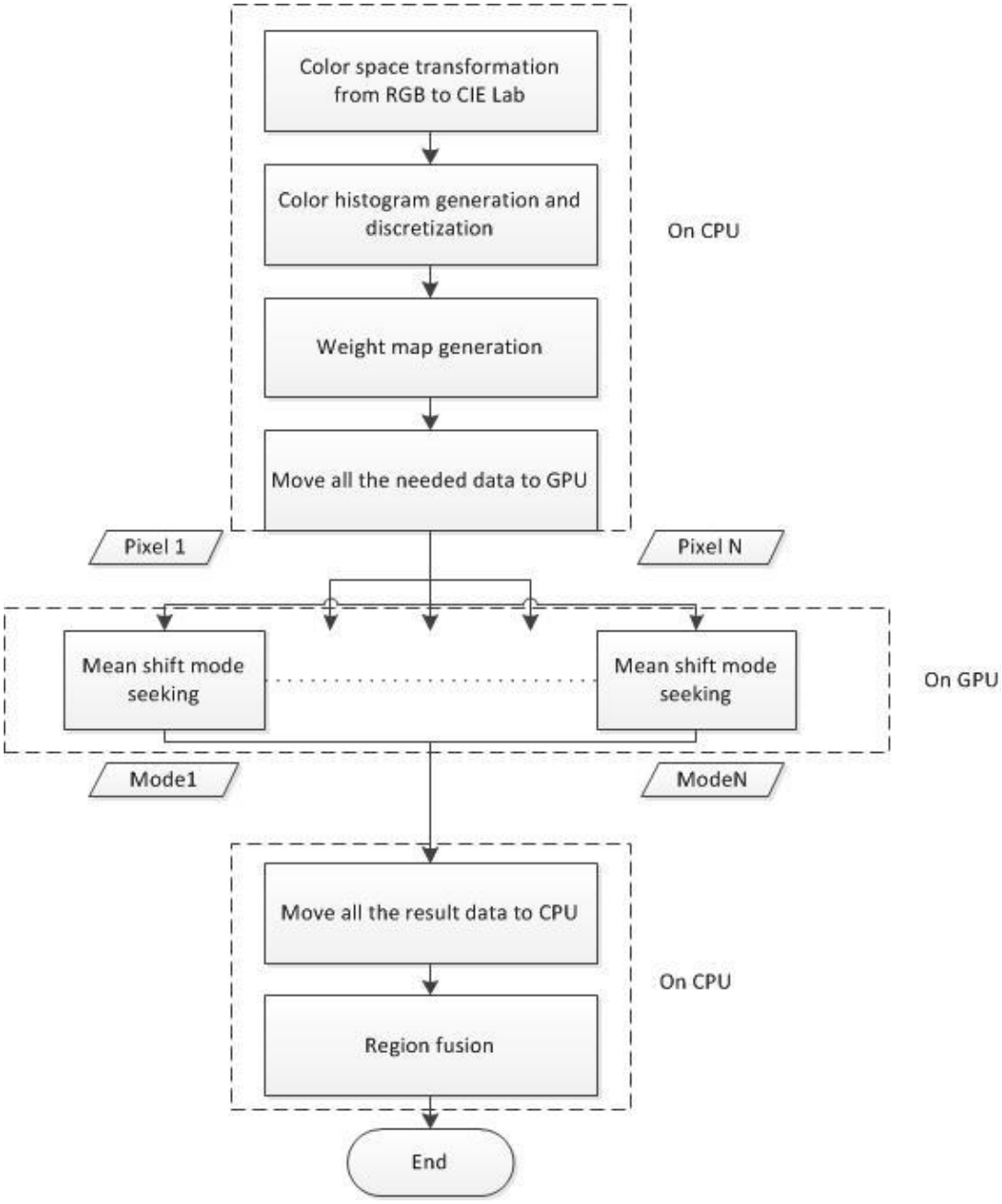


Figure 3.6 Implementation flow of the mean shift algorithm on both CPUs and GPUs

Hence, we developed a parallel implementation of ***Mean shift mode seeking***, which simply copies the image to the device and breaks the computation of the mode seeking into single pixels and their surrounding intermediate neighboring region. An independent thread is spawned for the mean shift mode seeking for each pixel. Multi-threads are running at the same time on the GPU to realize the parallel computation. The number of threads running in parallel is determined by the computational capability of the GPU. In

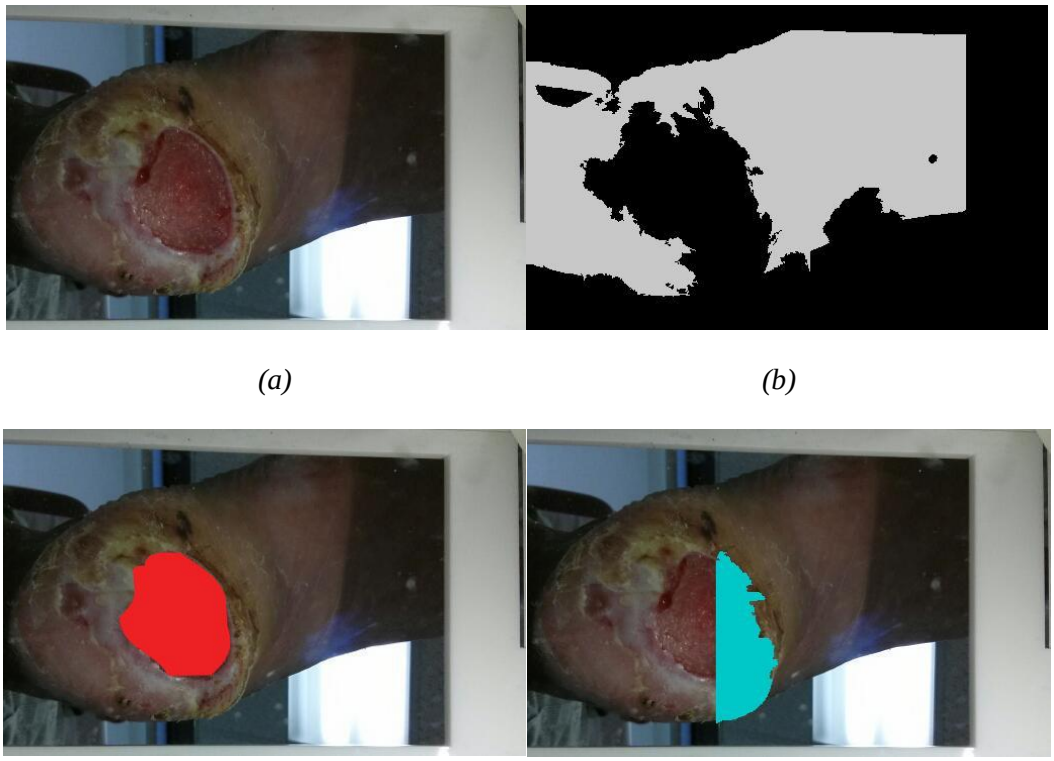
the Renderscript programming, this number is optimized automatically and does not need to be specified. After the mean shift mode seeking, all the result modes for each pixel are moved back to the local memory of the CPU. The ***Region fusion*** step, as discussed in detail in [17], is performed on the CPU.

3.3 Improvement to the Mean Shift Based Wound Boundary Determination Approach

In Section 3.2, we applied the largest connect component detection method on the binary foot image to determine the wound boundary within the foot outline. The main shortcoming of this algorithm is the requirement that a closing (complete) foot outline, which leads to detection failure if a wound are located at the foot boundary (in this case, the wound boundary will be regarded as part of the foot boundary, as shown in the part (b) of Figure 3.7). Figure 3.7 is an example of the binary foot image on which our current algorithm cannot provide an accurate wound boundary determination. For example, it can be clearly seen that the result from the largest connect component detection method, used in the implementation of the mean shift method in Section 3.2 and described in [10] is differ greatly from the wound labeling ground truth from an expertized wound clinician (comparison between the part (c) and (d) in Figure 3.7).

Hence, instead of detecting the connected "black" component within the "white" foot region from the foot binary image, we might need to detect the largest "almost enclosed" black region, which can also be understood as the most concave section on the foot outline. Based on carefully observation, this task may be done by locating the most irregularly changing point on the foot outline. We call these points as *turning points* on the foot outline (in our case, the number of turning points needed to be detected is 3).

In this section, we propose novel wound determination method, which will first determine whether the wound is located at the edge of the foot or not and then determine the wound boundary by applying different methods depending on the location of the wound.



(a) *(b)*

(c) *(d)*

Figure 3.7 An example of the incomplete foot outline. (a) The original image, (b) foot binary image , (c) wound area manually labeled in red by a wound clinician, (d) boundary determination results using the initial method, i.e., the largest connected component detection method.

3.3.1 Boundary Determination for the Wounds Enclosed in the Foot Outlines

First of all, we have to classify the wound boundary determination tasks into one out of two categories: 1) the wound is fully enclosed within the foot outline; 2) the wound is located at (or very near to) the boundary of the foot outline. One idea may be to use the foot boundary smoothness to distinguish between these two situations. However, the problem is that we may need a gold standard for the ordinary smooth foot curve, i.e. the boundary of the healthy foot, and quantitatively compare the actually detected foot outline to it in some way. The search for such a ground truth healthy foot curve is never an easy task. Moreover, we have to ensure that the patient's entire foot is imaged completely, which is a difficult-to-meet expectation for a patient-based wound analysis system considering the low mobility and lack of smartphone experience using of many

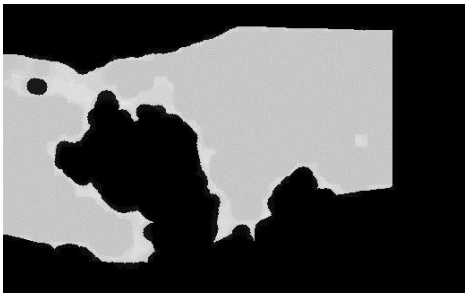
type 2 diabetic patients. Therefore, we use the following method to realize the task classification.

First we apply the image morphology operations called a closing operation to remove all the holes in the foot region (white part in the binary image) and smooth the external foot boundary which will help us to eliminate the possible interference for accurate wound boundary determination. The result after closing operation with a 9 x 9 circle structure element is shown in the part (b) of Figure 3.8, to be compared with the original foot binary image shown in the part (a). Second, we apply the combined region and boundary algorithm [72] to trace the external foot boundary along the edge of the white part in the foot binary image, as well as all the internal boundaries if there are any. The concepts of the external boundary and the internal boundary are illustrated in the parts (c) and (d) of Figure 3.8, respectively. For all the internal boundaries in a foot region, we only keep the ones with the perimeter larger than a preset threshold (in our implementation, it is set as 50 pixel lengths). In this case, the internal boundary in the part (c) is going to be filtered out, while the internal boundary in the part (d) will be regarded as the target wound boundary.

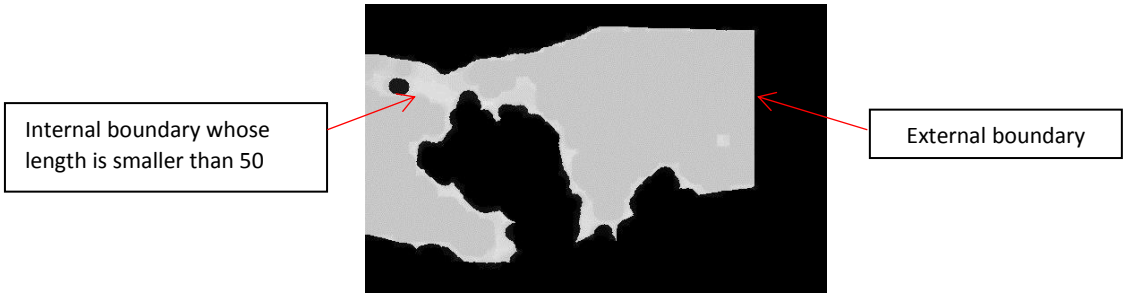
This simple thresholding method may not be a perfect algorithm but it works for most of the wound images in our experiments. In other words, if there is at least one internal boundary exceeding the preset threshold within the foot region, we regard it as the wound boundary and return it as the final boundary determination result (as shown in the part (e) of Figure 3.8). On the other hand, if there are not any internal boundaries whose length are beyond the threshold, we may need the boundary determination algorithm introduced in the next section. Note that here we assume there is at least one wound area on the photographed foot. Unfortunately, our method will assume the existence of at least one wound in the processing image. Even if there is no wound at all, an area which is recognized to be the most similar to a wound will still be identified. This false determination is certainly undesirable. This shortcoming might be overcome by design a wound filtering algorithm that can indicate the absence of a wound. For example, for each detected wound area, we will extract one feature vector from it and input to a well-trained classifier. The output of this classifier will be a Boolean variable with the decision

about whether this area is actually a wound or not. This operation needs machine learning to be evolved and will discuss further in Chapter 4 and 5.

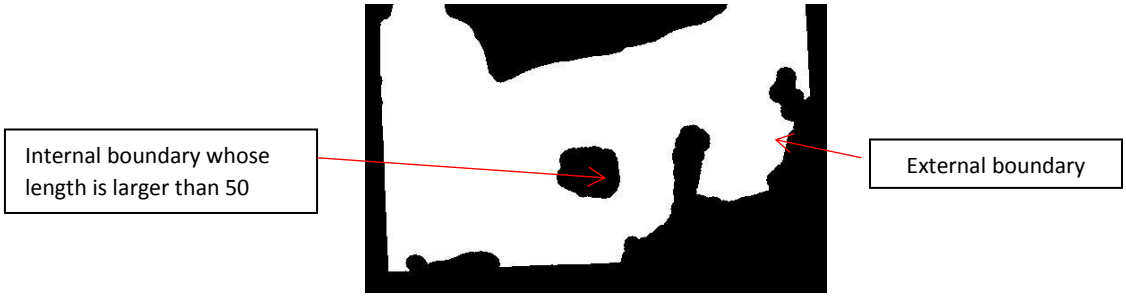
(a)



(b)



(c)



(d)

(e)

Figure 3.8 Classification of the wound boundary determination tasks. (a): The same foot region detection result as shown in the part (b) of Figure 1 (this part is placed here again for more clear comparison), (b): foot region after closing operation, (c) and (d): illustrations for the internal and external boundary of the binary foot region, (e) example of wound boundary determination by internal boundary detection of the foot region

3.3.2 Boundary Determination for the Wounds Near the Edge of the Foot Outline

- *Algorithm Overview*

After a careful study and observation, we propose a wound boundary determination algorithm structured as shown in Figure 3.9, which is applicable for a wound located at or near to the foot outline. As was mentioned in Section 3.3.1, we already have the external boundary of the non-enclosed foot outline.

As illustrated in the block diagram in Figure 3.9, the input to this algorithm is the external boundary of the foot region shown in part (c) of Figure 3.8. Instead of keeping all the points on the foot boundary, we down-sample the edge points (all the points on the external boundary of the foot region) by applying the Harris Corner Detection method [73] to a number of corner points. The corner points, also called junctions of edges, are prominent structural elements in an image and are therefore useful in a wide variety of computer vision applications, including serving as reference points for precise geometrical measurement [74]. A corner point can also be defined as a point, for which there are two dominant and different edge directions in a local neighborhood of the point. And corner points are more robust features for geometric shape detection than are the regular edge points. The corner detection result is shown in the part (a) of Figure 3.10, where the corner points are marked by black crosses. In our experiments, the perimeter of the foot outline usually is made up of over 2000 pixels. After down-sampling, the number of corner points is reduced to around 60-80 (also in terms of pixels). This down-sampling greatly improves the time performance of our algorithm. In addition, the down-sampling procedure will also be useful when detecting the turning points (this will be discussed in detail in the next bullet item).

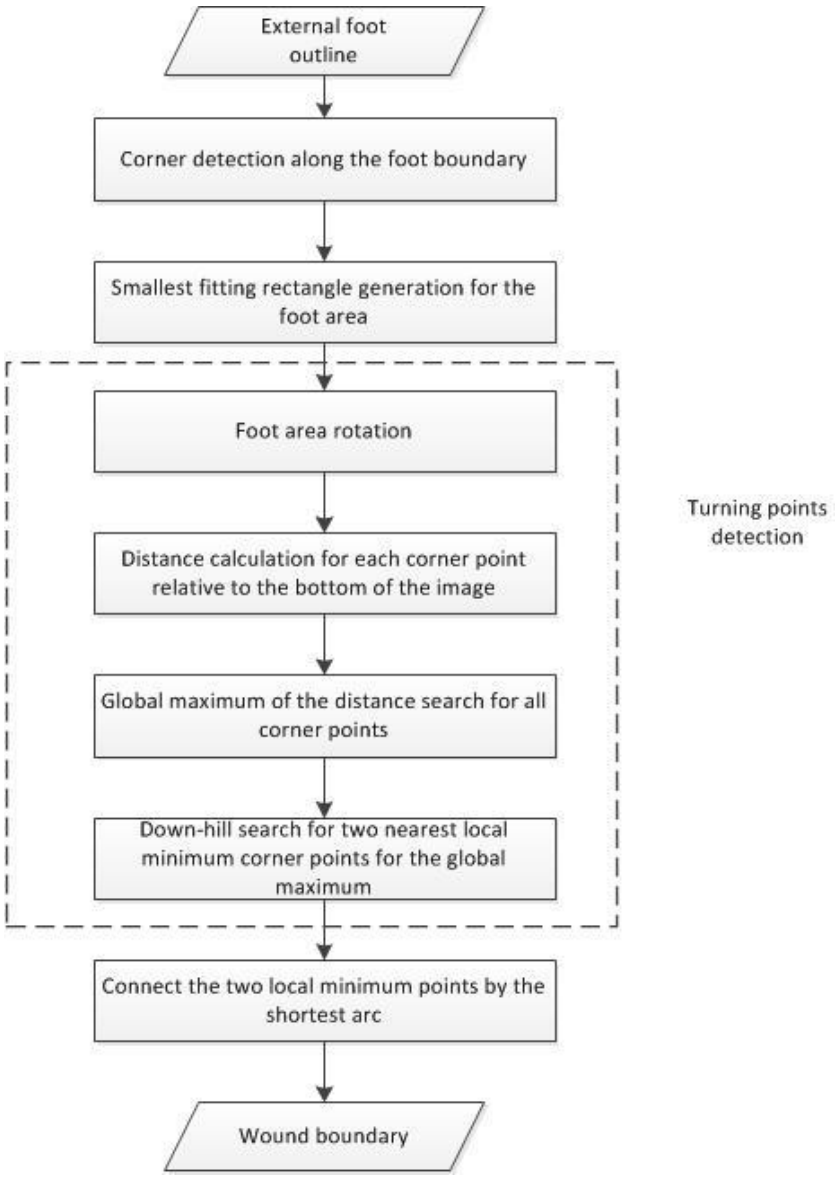


Figure 3.9 Main structure of the new wound boundary determination algorithm

The 3rd to the 8th block in Figure 3.9 shows the main concept, which is to detect the three turning points on the foot boundary. These turning points will be used to determine the wound section on the foot outline (the three turning points are marked with small black crosses in part (c) of Figure 3.10). The turning points can be defined as the points that define the largest direction changes in the contour along the foot boundary. In the next bullet item, the method for detecting the turning points will be discussed in details. After the three points are determined, we can move along from the global maximum point (which is the most concave point in the middle) to the two local minimum points

along the foot boundary in the two opposite directions. Then we will connect the two local minimum points by an arc which is the optimal approximation to the non-closed part of the wound boundary. In our case, we draw an arc from either local minimum to the other one with a radius equal to half of the diagonal length for the wound image. The arc drawing algorithm is as described in [23].

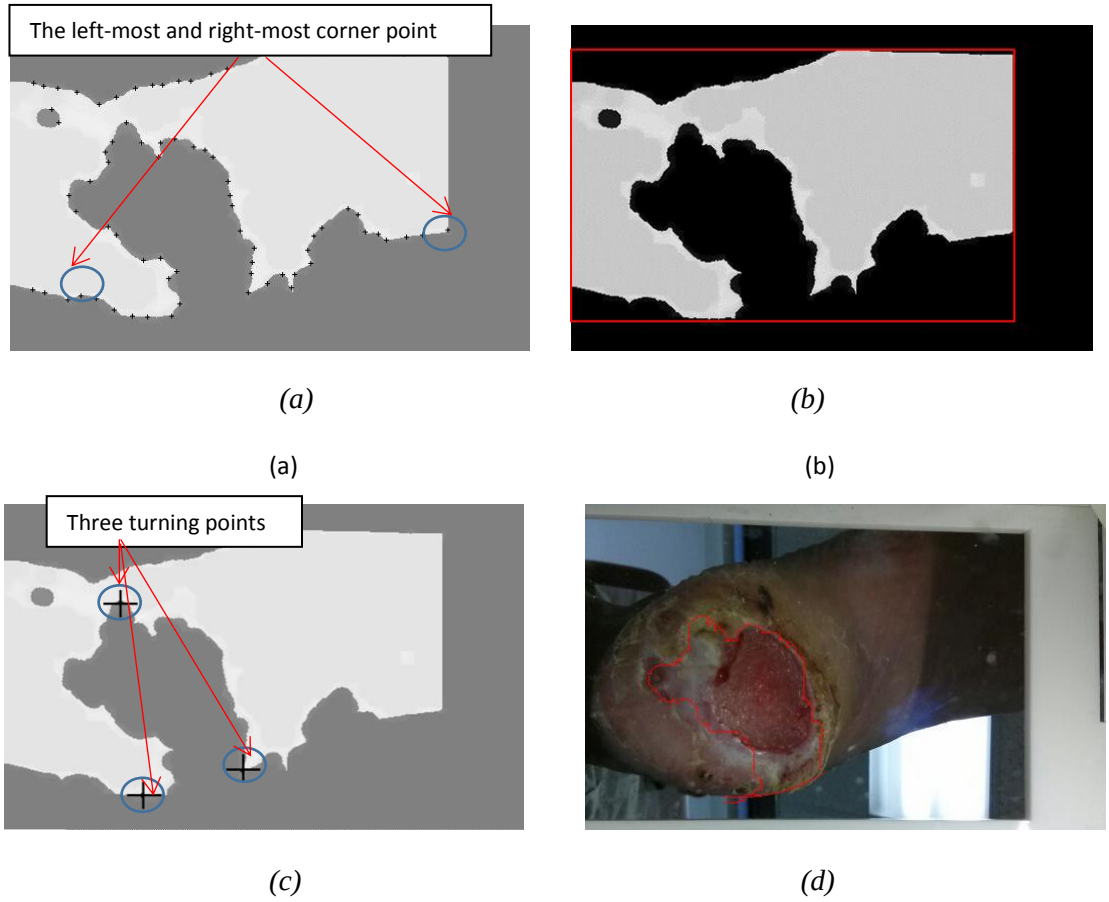


Figure 3.10 The result of the opening operation (a) the corner point detection result, (b) the smallest fitting rectangle, (c) three turning points detection result on the foot outline, (d) the wound boundary determination result based on these three turning points

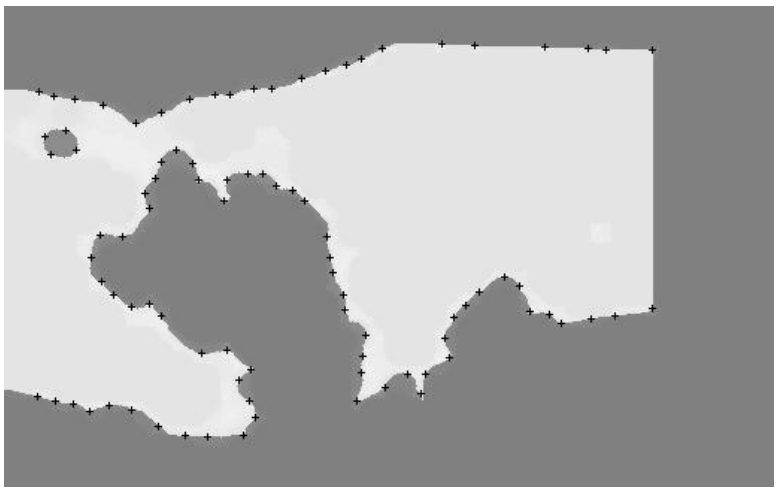
- *Turning Points Detection*

In this section, we propose a maximum-minimum search approach to detect the turning points.

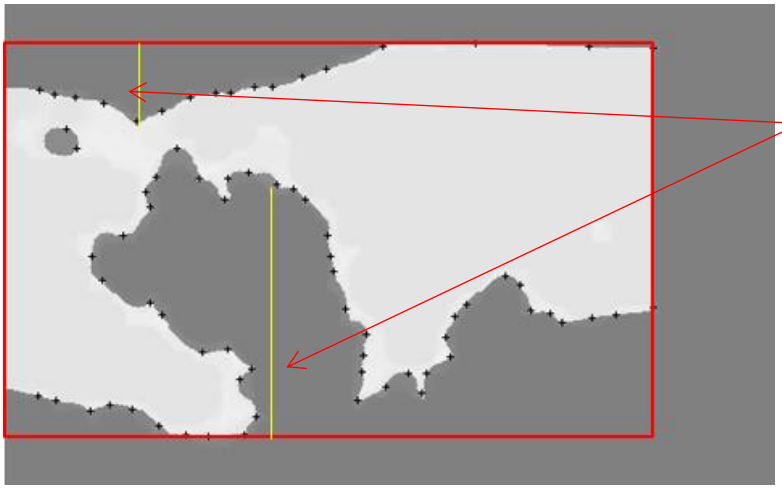
First, we sort all the corner points into a list based on their position on the foot boundary (from the top-right to top–left, in a clock-wise direction), then locate the two special

extreme corner points on the foot boundary: the leftmost and the rightmost (as indicated in part (a) of Figure 3.11).

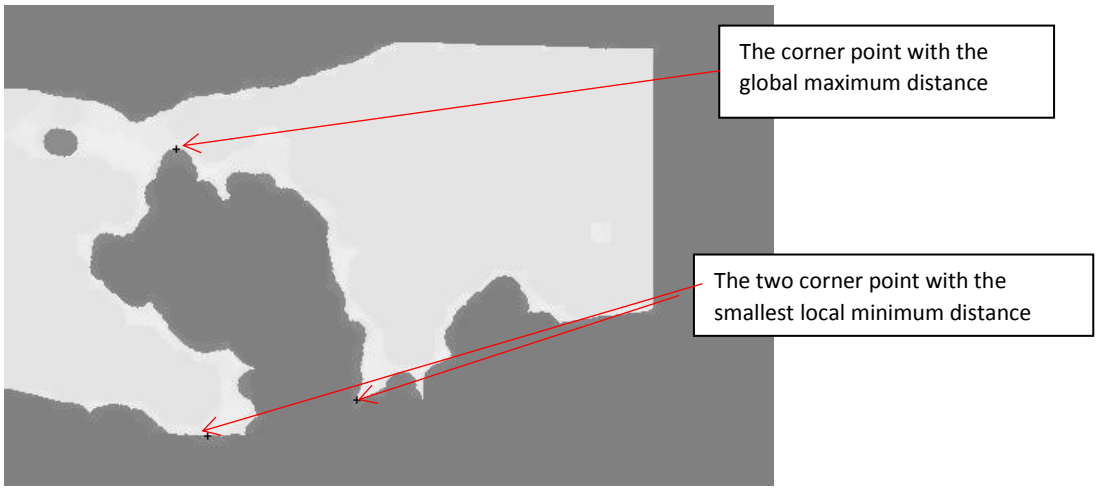
As shown in the part (a) in Figure 3.11, we divide the corner points into two groups: 1) the corners points which located between the two extreme points and 2) the corner points located outside this range. Note that this categorization is based on the clock-wise sorted corner points list. In the part (a), we mark the first group of corner points by blue circles and the second group by red circles. For the first group, we calculate the vertical distance of each corner point to the top side of the smallest fitting rectangle (SFR) of the foot region. The smallest fitting rectangle is supposed to be tangent to the foot area at four boundary points: the top-most, bottom-most, left-most and right-most point of the foot area, as shown by the red frame in the part (b) of Figure 3.11.



(a)



(b)



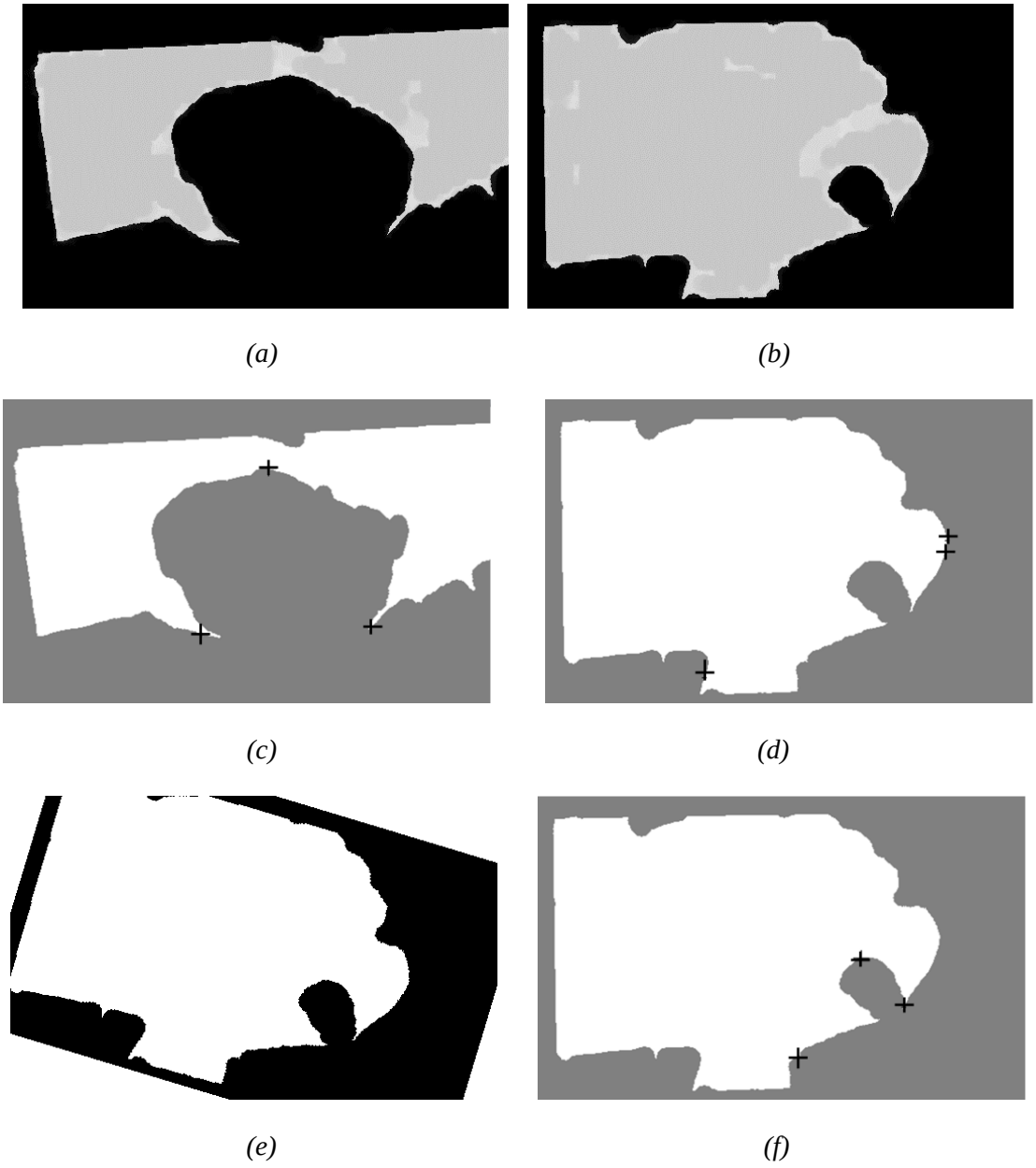
(c)

Figure 3.11distance calculation from corner points to the sides of the smallest rectangle, (a): corner points grouping, (b): vertical distance calculation from each corner point to the corresponding side of the SFR, (c): three turning points determination result

Similarly, we calculate the vertical distance of each corner point in the second group to the bottom side of the SFR (as shown in the part (b) in Figure 3.11). Afterwards, we locate the turning points by searching for the corner point with the global maximum vertical distance to the corresponding rectangle side (top or bottom based on its group number: first or second) and also the two corner points on each side of the maximum point with the smallest local minimum vertical distance (as shown in part (c) of Figure 3.11). The only concern is the search for target turning points may be accidentally stopped by interfering local extrema, which is a common problem of most local search algorithms. As mentioned earlier, we only keep a certain number of corner points on the foot outline. Based on the experimental results, we find that this down-sampling procedure can eliminate most of the interfering local extrema that may impede the search for the optimal turning points.

For the foot image shown in the part (a) of Figure 3.12, the turning points can be accurately located by this maximum-minimum search approach (as shown in the part (c) of Figure 3.12, where the black cross in the middle represents the global maximum corner point and the two crosses on sides represent the two local minimum points). However, for

the image in the part (b), the wound is located at the corner of the foot outline. In this case, the approach stated above may fail since the point with the global maximum is no longer the desired turning point for wound boundary determination (as shown in the part (d) of Figure 3.12). To solve this problem, we rotate the foot image clock-wise by the angle which is determined by the formula in eq. (3.7).



(a) *(b)*

(c) *(d)*

(e) *(f)*

Figure 3.12 Turning Points detection results, (a) and (b): foot binary images, (c) and (d): three turning points detection results, (e): rotated foot binary image from (b), (f) three turning points detection result on the rotated image

$$\tan\theta = \frac{y_{mostright} - y_{mostleft}}{x_{mostright} - x_{mostleft}} \qquad \textit{Equation 3.7}$$

where $y_{mostright}, y_{mostleft}, x_{mostright}, x_{mostleft}$ are the x-y coordinates of the two extreme points as shown in the part (a) of Figure 3.10 (we assume the origin is located at the top-left corner). The rotated image is shown in the part (e) of Figure 3.12. We can see from part (f) of Figure 3.12 that the detected global maximum corner point after rotation is the desired point.

3.4 Wound Boundary Determination Results

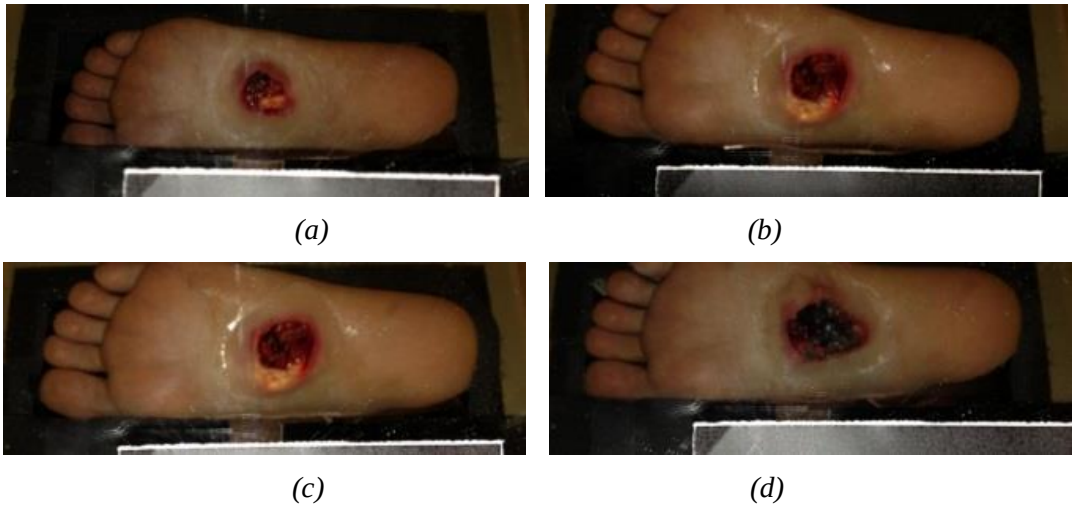
3.4.1 Experimental Set-up

To evaluate accuracy, we tested the three approaches introduced in Section 3.1 – 3.3 on two categories of wound images. For the first category, we used 60 images of simulated wounds, typically referred to as moulage wounds. The moulage wounds permitted us to evaluate our method under relatively consistent skin conditions and on wounds with distinct boundaries. Moulage is the art of applying mock injuries for the purpose of training emergency response teams and other medical and military personnel. In our case, we use the moulage wounds that include typical granulation, slough and necrotic tissues; the wounds were provided by *Image Perspectives Corporation* (Carson City, Nevada) and applied to the this writer's foot. The selected four sample images (not all 60 images were presented in this work of the Moulage wounds are shown in Figure 3.13 and were captured with the smartphone camera placed on the image capture box.

For the second category, we evaluated our wound image analysis method on 100 images of actual patient wounds collected at the UMass-Memorial Health Center Wound Clinic (Worcester, MA), following an IRB approved protocol in accordance with Federal Regulations. The purpose of selecting these typical wound images from type 2 diabetic patients was to provide a more realistic test of our wound boundary determination and color segmentation algorithms. Six selected sample images are shown in Figure 3.14. Compared with the images in Figure 3.13, the real wound images are more complex: they may have uneven illumination over the image plane, complex surrounding skin texture and wounds appearing in a variety of shapes, sometimes lacking distinct boundaries. Note

that all six of the wound images in Figure 3.14 were captured by our smartphone and image capture box.

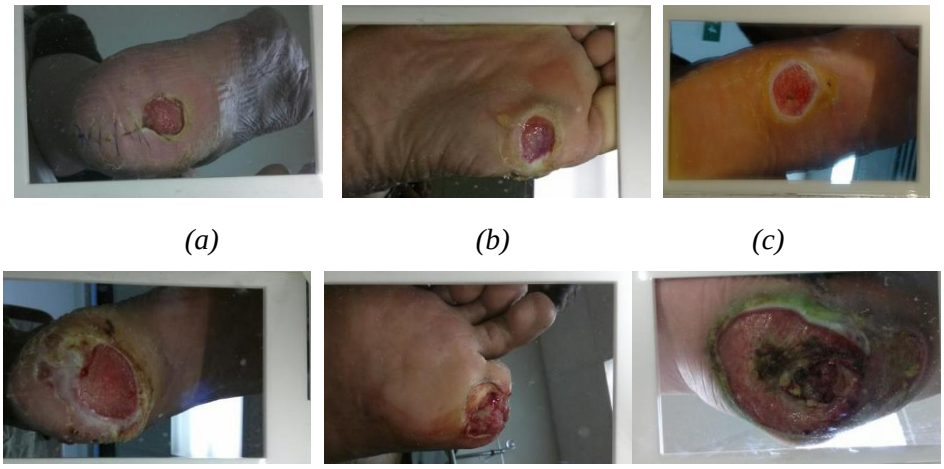
To test algorithm efficiency, all three methods were implemented on the following PC: Intel quad-core CPU, 4GB RAM, ATI Radeon HD 6350 GPU. In addition, the two mean shift based wound analysis algorithms discussed in Section II was further implemented on the Nexus 5 Android smartphone: CPU (Quad core, 1500MHz, Krait, 2048 MB system RAM) and the GPU (Adreno 320, Qualcomm). All the programming was done in Java in the Eclipse IDE (Integrated Development Environment).



(a) *(b)*

(c) *(d)*

Figure 3.13 Wound images of the moulage simulation applied on the author's feet



(a) *(b)* *(c)*

(d) *(e)* *(f)*

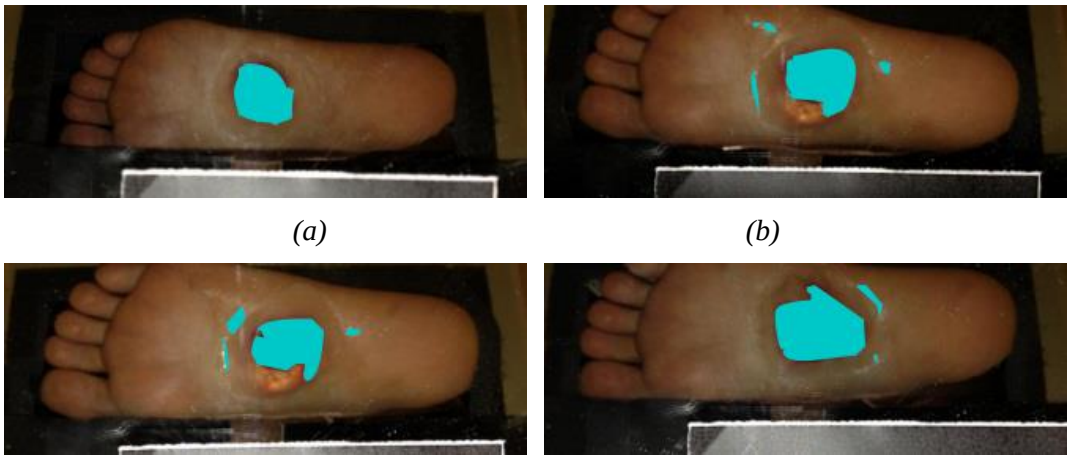
Figure 3.14 Clinical image samples of actual patients

As mentioned in Section 2.1, the captured wound images were down-sampled by a scaling factor 4 in either spatial dimension. Furthermore, irrelevant background information was cropped off to better facilitate the following boundary determination. As a result, the image dimensions of the Moulage wounds are 816×300 pixels. The image dimensions of the real wounds are 516×322 pixels.

3.4.2 Wound Boundary Determination Results on Images of Moulage Wounds

- *Evaluation of the Level Set Based Algorithm*

After applying the level set based algorithm, the wound boundary determination results on images of Moulage wounds are shown in Figure 3.15, where the determined wound areas are marked in light blue color. As mentioned in Section 3.1, the level set based algorithm is semi-automatic and we need to manually set the initial curve to start the contour evolution. Because of the nature of the level set based algorithm, the evolution has been accidentally stopped on false edge information in the foot skin area (it can be easily observed on part (b) – (d) in Figure 3.15). Actually, we can increase the number of iterations to diminished the false wound area, however, this will be further contract the target wound boundary and cause leakage when the actual boundary is weak. This can be viewed in part (c), where the yellow wound tissue is not included in the determined wound area. Another possible solution to remove the outlier areas is to filter the resulting closing areas by size, i.e. only the largest area is kept.



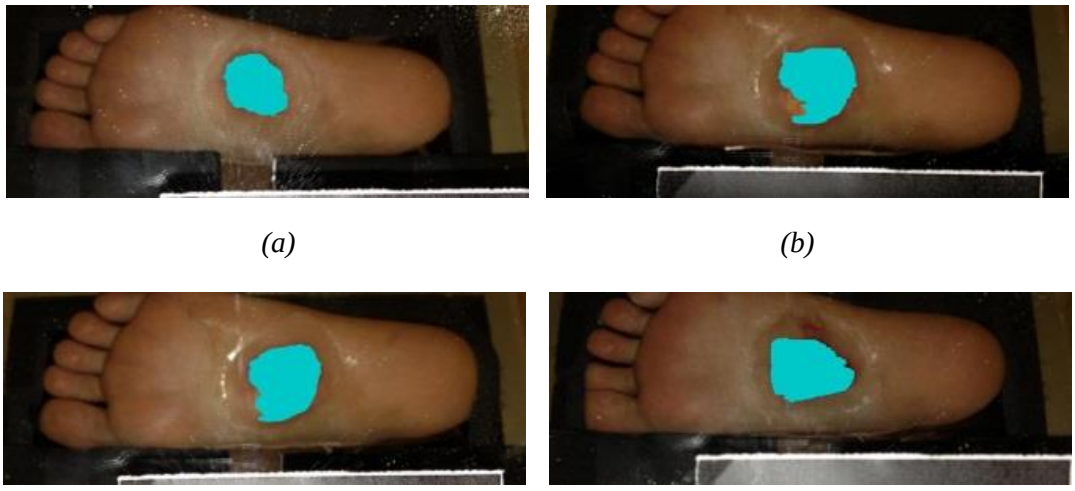
(a) *(b)*

(c) *(d)*

Figure 3.15 Wound boundary determination results on images of Moulage wound simulation by level set based approach

- *Evaluation of the Mean Shift Based Algorithm*

The wound boundary determination results on Moulage wounds images are shown in Figure 3.16. As seen from the part (a)-(d), the mean shift segmentation algorithm described in Section II provides promising boundary detection results. However, there are still areas near the wound boundary with visibly imperfect detection as shown in part (c) where the yellow wound tissue at the boundary has similar color to the healthy skin surrounding it.



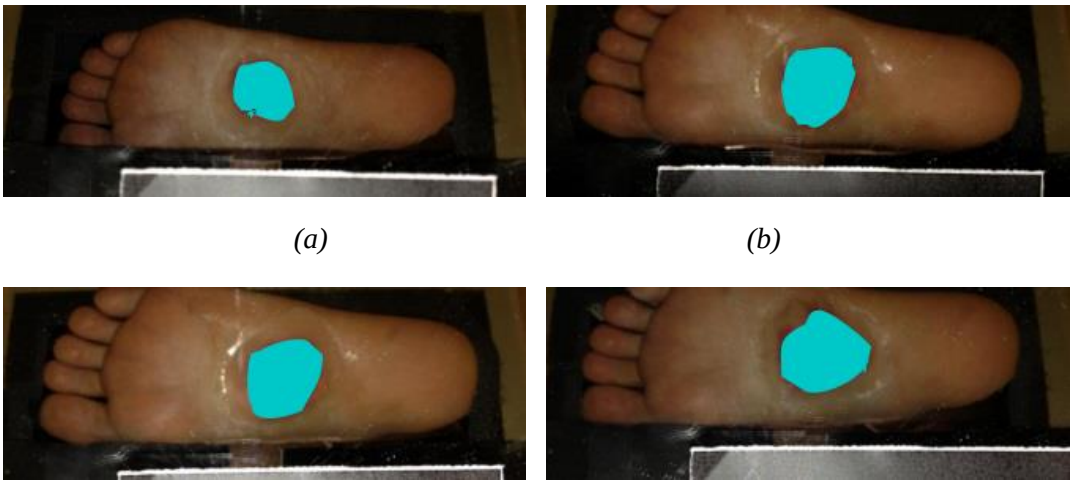
(a) *(b)*

(c) *(d)*

Figure 3.16 Wound boundary determination results on images of Moulage wound simulation by mean shift based approach

- *Evaluation of the Improved Mean Shift Based Algorithm*

The wound boundary determination results on Moulage wounds images are shown in Figure 3.17. All the wound simulations are located within the foot outline. Therefore, according to the description in Section 3.3, the longest internal boundary will be viewed as the wound boundary. Compared with the results shown in Figure 3.15 and 3.16, we can see the combined region and boundary algorithm provides promising internal boundary determination. Finally, the morphological based boundary refinement procedure introduced in Section 3.3 also contributes significantly to smooth the determined wound boundary.



(a) *(b)*

(c) *(d)*

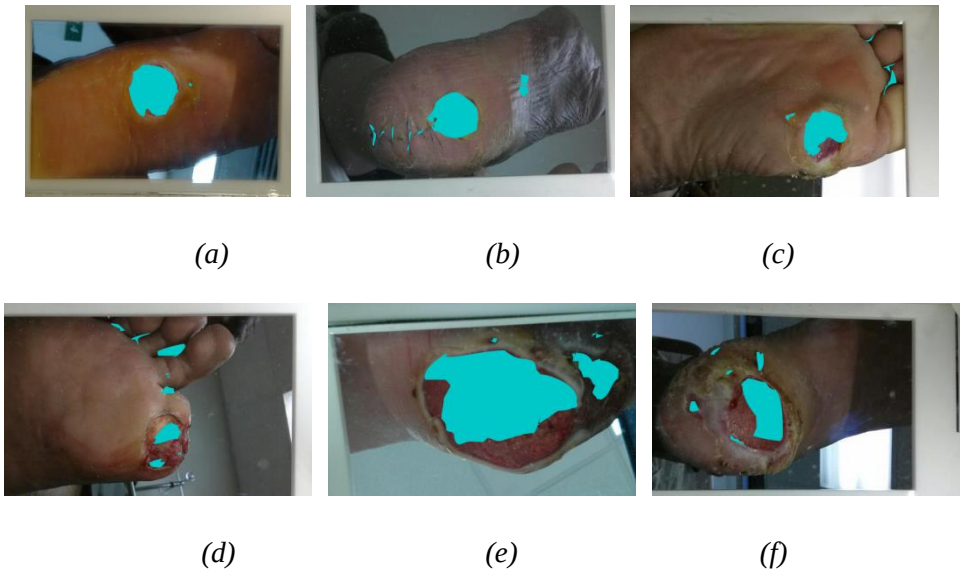
Figure 3.17 Wound boundary determination results on images of Moulage wound simulation by improved mean shift based approach

3.4.3 Wound Boundary Determination Results on Real Wound Images

- *Evaluation of the Level Set Based Algorithm*

The wound boundary determination results of the level set based algorithm on real wound images are shown in Figure 3.18. These images are more complex than is the case for the images of Moulage wounds. Some of them are not illuminated correctly because the position of the foot, such as part (b), and some have uneven skin texture such as part (e) and (f). Moreover, some wounds are located at the foot boundary (perhaps with the toe amputated), or at the toe.

We can see from these results that the DRLSE algorithm performs satisfactorily only when the wound is well located in the middle of the foot area and the skin texture is uniform (as shown in part (a) in Figure 3.18). Otherwise, this approach often suffers from the false edge information on the textured foot skin, which is common to patients with diabetic foot ulcers (the results contain some small false wound area as shown in part (b) – (f)). Furthermore, when wound is located at the edge of the foot outline, the contour evolution always leaks at the weak boundary since we set the foot outline as the initial boundary.



(a) *(b)* *(c)*

(d) *(e)* *(f)*

Figure 3.18 Wound Boundary determination results on images of real wound simulation by level set based algorithm

- *Evaluation of the Mean Shift Based Algorithm*

To adapt to different conditions in real wound images, we had to adjust the parameters of the algorithm for each wound image. There are three adjustable parameters in the mean shift based wound boundary determination algorithm for each wound image: the spatial resolution h_s, the range resolution h_r and the fusion resolution h_f.

Initially, we tried the parameter settings $h_s = 7$, $h_r = 6$ and $h_f = 3$ because a published work [52] showed good segmentation results with these values. However, our experiments with these default settings applied to the real wound images did not provide good wound boundary results for all six images. As mentioned in [44], only features with large spatial support are represented in the mean shift filtered image when h_s increased, and only features with high color contrast survive when h_r is large. In [52], it is also stated that a larger number of regions will remain in the region-fused image when employing a smaller h_f. In Section 3.2, we discussed how the wound boundary determination is strongly dependent on whether a complete foot boundary can be detected. Hence, we need a better spatial resolution as well as a better region fusion resolution value to group the small regions in the foot area to a connected component when the skin texture is complex (as shown in part (f) in Figure 3.18). On the other hand, if the wound is located near to the foot boundary (as shown in part (d), (e) and (f)), better spatial and

fusion resolution is also needed to avoid the disconnected foot boundary detection, which will cause error in the wound boundary determination.

Unfortunately, after trying different parameter settings over specified ranges $(4 \le h_s \le 10,\ 4 \le h_r \le 7, 3 \le h_f \le 15)$ and using 0.5 as the adjustment step to customize the parameters for each wound image, the results shown in Figure 3.19 are still not promising. Especially when dealing with the wounds located not completely within the foot outline, the algorithm only provides partially correct wound recognition results, as shown in part (c), (e) and (f) of Figure 3.19.

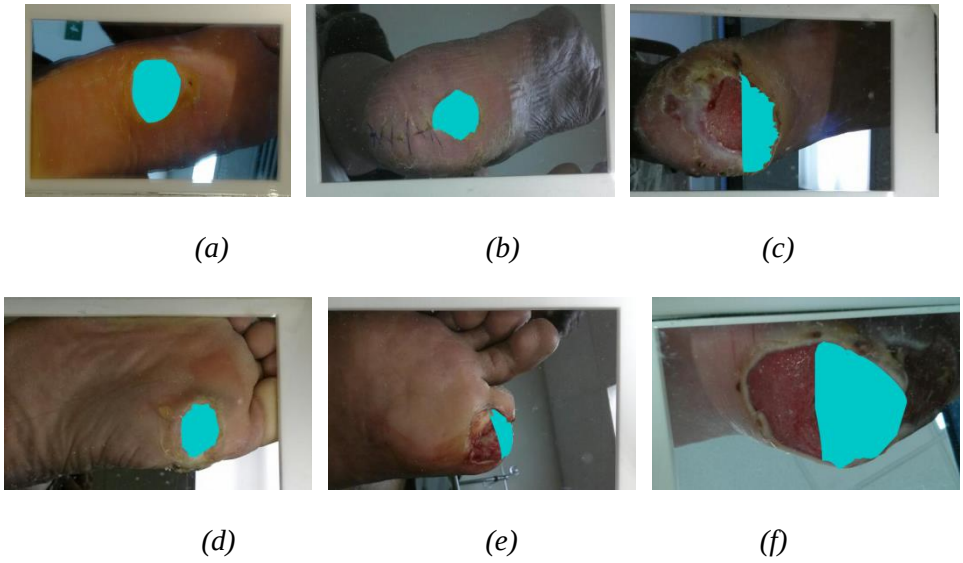


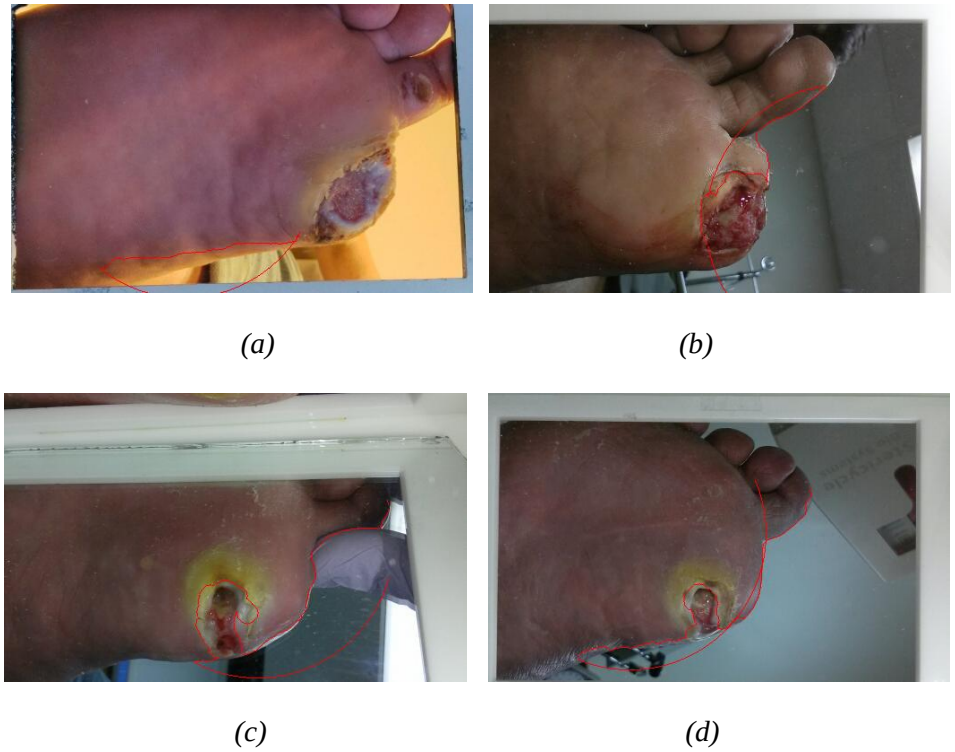
Figure 3.19 Wound Boundary determination results on images of real wound simulation by mean shift based approach

- *Evaluation of the Improved Mean Shift Based Algorithm*

When applying the improved mean shift based algorithm to real wound images, promising results were obtained on most of the wound images. However errors occurred in particular on the images with toe-amputations. Examples of such errors can be seen in Figure 3.20. Since this algorithm is boundary based, we visualize the result by drawing determined boundary, instead of wound area, in both Figure 3.20 and Figure 3.21.

For toe-amputated feet, the foot outline has a quite different shape compared to that of a healthy foot. Therefore, the corner point with the global maximum distance to the side of the SFR may not be located on the wound section in such cases, and the detected three turning points might not correspond to the target wound region. An approach to solve this

problem is formulated as follows. Instead of only seeking one global maximum, we locate all the corner points with local maximum vertical distance to the rectangle side and sort them in a descending order. Afterwards, we choose the first two corner points which have the two largest distances to the bottom side of the SFR. The one near to the right-most corner point represents the toe-amputated area. And the other corner point is assumed to represent the wound area. We have implemented this approach, and the experimental results on all foot images with toe amputation are shown in Figure 3.21. It can be clearly seen that this modified wound boundary determination is much more accurate. However, this also brings to us another question: how to decide whether the foot image is that of a toe-amputated foot or not. One way to resolve this question is to include toe-amputation information (amputated or not) in the patient's medical record.



(a) *(b)*

(c) *(d)*

Figure 3.20 Wound boundary determination errors on images with toe-amputated feet

Finally, we applied this approach to 6 wound images in Figure 3.14. Comparing to results provided by other two approaches, we can see an obvious improvement. However, the determined boundaries are still not completely consistent with the actual boundary, due to approximation nature of the contour generation method based on three turning points.

(a) *(b)*

(c) *(d)*

Figure 3.20 Improved wound boundary determination results on foot images with toe amputation

(a) *(b)* *(c)*

(d) *(e)* *(f)*

Figure 3.21 Wound Boundary determination results on images of real wound simulation by improved mean shift based approach

3.4.4 Quantitative Evaluation and Comparison of the Three Wound Boundary Determination Algorithms

To quantitatively evaluate the three wound boundary determination algorithms, described in this chapter, we asked three experienced wound clinicians from University of Massachusetts Medical School to independently trace the wound boundaries for all 100 real wound images. Then we generated ground truth wound label from these boundary determinations. The details of the ground truth generation process are described in Chapter 4. Finally, we use the Matthew Correlation Coefficient (MCC) [75] to measure the wound boundary determination accuracy for the three algorithms. The mathematical definition of MCC is shown in eq. (3.8).

$$MCC = \frac{tp \times tn - fp \times fn}{\sqrt{(tp + fp)(tp + fn)(tn + fp)(tn + fn)}} \quad \textit{Equation 3.8}$$

where *tp*, *tn*, *fp* and *fn* represent true positive, true negative, false positive and false negative results, respectively [75]. The MCC returns a value between -1 (total disagreement) and +1 (perfect prediction). As mentioned in Section 3.1, 3.2 and 3.3, all three boundary determination algorithms have several parameters, which need to be set manually. For a fairer comparison, we find the optimal parameters empirically for each algorithm beforehand. Then we run these three algorithms on the same wound image dataset. The MCC based performance evaluation results are shown as in Table 3.1. Moreover, we perform the computation time analysis of these three algorithms on the same dataset. Especially, for each algorithm, the average running time for both CPU based implementation and the GPU optimized implementation are evaluated separately (Note that all the implementations of three algorithms are based the PC). The computation time analysis results are shown in Table 3.2.

For the level set based algorithm (Approach 1), the boundary determination results are promising on images with normal and uniform skin color. However, false edges may interfere with the evolution when the skin color is not uniform enough and when missing boundaries, as frequently occurring in wound images, results in evolution leakage (the level set evolution does not stop properly on the actual wound boundary). In addition, the

level set evolution completely depends on the initial curve which has to be pre-delineated either manually or by a well-designed algorithm.

For the mean shift and connected component detection based algorithm (Approach 2), the experimental results show that this method efficiently provides accurate wound boundary determination results when dealing with the wounds located within the foot outline. However, when the wounds are located on the image edge, such as on toes, this approach cannot find the actual wound boundary. For the improved algorithm presented in Section 3.3 (Approach 3), the boundary determination results are much better than those obtained with Approach 2 when wounds occur on the image edge or on toes. Moreover, the toe-amputation status needs to be an input parameter to Approach 3, which makes the approach not completely automatic.

As seen from Table 3.2, the computation time for GPU+CPU implementation of the accelerated DRLSE algorithm has been enhanced by about a factor 5 compared with the CPU alone implementation. However, the parallel implementations of Approach 2 and 3 do not provide such a significant improvement on time performance. The reasons are: 1) based on the description in Section 3.2 and [71], various CPU based optimization procedures have been applied to the mean-shift segmentation algorithm, and 2) the data transfer from RAM on PC to memory space on GPU chip is quite time-consuming and this extra time overhead undermines the efficiency enhancement from parallel implementation.

Finally, we also perform the efficiency analysis for Approach 2 and 3 based on smartphone-alone platform. We haven't implemented Approach 1 on the smartphone since it is only semi-automatic (the level set evolution needs to start with a manually set initial curve). While GPU+CPU implementation of the mean shift algorithm on PCs only provides minimal improvements in computation time over a CPU implementation, the GPU + CPU implementation on the smartphone does improve the time performance by about a factor of two, which can be seen from Table 3.3. This is partly because the CPU on the smartphone is not as powerful as the one on the laptop. Moreover, the Renderscript implementation utilizes both the smartphone GPU as well as the CPU and even available DSP devices on chip to provide the effective optimization.

Table 3.1MCC based wound boundary determination accuracy evaluation results on 100 real wound images. Approach 1, 2 and 3 represent the boundary determination approach introduced in Section 3.1, 3.2 and 3.3 respectively

Approach ID	*1*	*2*	*3*
MCC score	*0.337*	*0.436*	*0.622*

Table 3.2 Computation time (unit: seconds) analysis results for three wound boundary determination accuracy approaches on 100 real wound images. Approach 1, 2 and 3 represent the boundary determination approach introduced in Section 3.1, 3.2 and 3.3 respectively

Approach ID	*1*		*2*		*3*	
Implementation	*CPU*	*CPU+GPU*	*CPU*	*CPU+GPU*	*CPU*	*CPU+GPU*
Computation time	*13.8*	*4.9*	*4.7*	*3.9*	*6.8*	*5.3*

Table 3.3 Computation time (unit: seconds) analysis results for Approach 2 and 3 on smartphone-alone platform

Approach ID	*2*		*3*	
Implementation	*CPU*	*CPU+GPU*	*CPU*	*CPU+GPU*
Computation time	*31.2*	*15.8*	*34.3*	*17.3*

3.5 Conclusion

As has been described in the preceding sections, we have designed three different non-machine learning approaches to determine wound boundary. Level set and mean shift based algorithms (referred as Approaches 1 and 2) suffer from their own disadvantages and cannot provide accurate results when the wound conditions are complicated. As an augmentation of Approach 2, we improved the mean shift based algorithm so that we can now detect the wounds, at almost any places and with various skin textures, with a much higher degree of accuracy (this approach is referred as Approach 3), with only a slight

reduction in efficiency. However, Approach 3 is still far from perfect. In this Conclusion section, we will give a quick overview of the algorithm flow an also provide some critical evaluation of this algorithm. Note that the improvement idea proposed for Approach 3 cannot be applied to Approach 1 since the foot outline is not determined before level set evolution starts.

Approach 3 classifies the wound locations into three categories: 1) wound in the middle of the foot, 2) wound at the edge of the foot without toe-amputation and 3) wound at the edge of the foot with toe-amputation. For the first category, the wound is expected to be surrounded by healthy skin and can easily be detected by tracing the internal boundary within the foot outline. For the second and third categories, we apply the three turning points detection method to locate the wound boundary which is assumed to be the most irregularly changed section on the foot outline. In practice, the method dealing with these two situations (with or without toe-amputation) is slightly different as described. Hence, we may require that the toe-amputation information is an input to the algorithm and obtained as part of the patient's medical record.

Generally speaking, this algorithm is fully automatic with an appropriate parameter setting. We need to set the parameters empirically, which means we have to try different parameter settings on different wound images and choose the setting which provides the most consistent wound boundary determination results on all sample images. A disadvantage of this algorithm is that there are quite a number of parameters that need to be set up. These parameters are fully described as in Table 3.4.

As seen from the table above, we have 7 parameters which need to be chosen manually. Fortunately, the experimental results show that most of these parameters can easily be set to a fixed value, chosen empirically, and still provide robust results. The most difficult cases are the setting of region fusion bandwidth in the mean shift segmentation and the region area threshold value to filter out the small holes on the detected foot area.

The region fusion bandwidth strongly impacts the accurate foot outline detection from the irrelevant background information, which serves as the basis for the subsequent wound boundary determination. Unfortunately, the experimental results show that there is no

single fixed region fusion bandwidth parameter that can provide us consistently promising foot outline detection results on all sample wound images.

Finally, all three approaches heavily rely on the parameter setting and cannot be automatically refined based on new clinical inputs. In next two chapters, we will try to overcome these limitations by explore machine learning based wound recognition methods.

Table 3.4 Parameters used in the improved mean shift based wound boundary determination algorithm

Function Module	*Parameters*	*Parameter Number*
Mean Shift Segmentation	*Spatial bandwidth, Color range bandwidth, Region fusion bandwidth*	*3*
Morphology Operation	*Structure element radius*	*1*
Contour detection	*Region area threshold value to filter the small holes*	*1*
Corner Detection	*Corner quality threshold value to filter the less strong corners*	*1*
Boundary Connection	*Radius for the arc that connects the two local minimum corner points*	*1*

Chapter 4

Binary SVM Classifier Based Wound Boundary Determination

4.1 Wound Recognition System Overview

As mentioned at the end of Chapter 3, the mean shift algorithm, even the improved version with the three turning points detection method, still has difficulty in providing a consistently accurate result when the wound structure is complicated. This includes situations such as 1) the healthy skin area in the image is not smooth enough to be fused into an entire area, 2) the wound boundary is not clearly delineated and part of the wound may be fused as the healthy foot area and 3) the toe is amputated which causes an unexpected indentation on the foot outline, which may confuse the maximum-minimum searching based turning points detection method. To overcome these problems, we may need to adjust the region fusion bandwidth sometimes, and record the toe-amputation status as one item of patients' medical history. Another drawback of the mean shift based algorithm is that we cannot update the algorithm based on new wound images and related clinical inputs. Besides, this algorithm does not exploit the clinical nature of wounds and therefore cannot provide consistently accurate wound boundary results. An example of the boundary determination error based on the improved mean shift based approach in Chapter 3 is shown in Figure 4.1. We can see some sutures attached to the actual wound area in part (a). However, Approach 3 erroneously recognizes the sutures as part of the wound as shown in part (b). Hence, we need find a smarter approach which can exploit the nature of wound texture for recognition. In part (c) of Figure 4.1, the wound determination result by the approach proposed in Section 4.3 is shown and we can see the

suture is successfully excluded from the determined wound area since we teach the model how to distinguish wounds from other objects, such as sutures, on the skin by ground labels from clinicians.

(a) *(b)* *(c)*

Figure 4.1 Example of the detection error, (a) the original image with sutures attached to the actual wound area, (b) wound boundary determination results by improved mean shift based algorithm introduced in Chapter 3, (c) wound boundary determination results by SVM based method introduced in Section 4.3

As seen from the discussion above, it is a great challenge to develop accurate algorithms for the wound boundary determination, tissue classification and healing status analysis. The goal is for such algorithms to be automatically run in a real-time fashion and to provide meaningful feedback to the direct users or the remote observers through a telemedicine scheme. Since our system may not necessarily be used in a clinical environment (for example, it may be used in the patients' homes), it is nearly impossible for us to fully control the image capture conditions and focus only on patients' foot. Hence, we should assume the wound image will not only contain the necessary skin and wound information but also some irrelevant background scene, which will complicate the wound boundary determination. For this reason, some methods reviewed in Chapter 1 are semi-automatic and require some manual input to pre-locate the wound. In our work, our goal is to develop a fully automatic wound boundary determination method which is able to recognize the wound from the wound image under relatively relaxed image capture conditions. We categorize our method as an object recognition scheme whose output is an accurate object boundary rather than a bounding box enclosing the target as required by most object recognition tasks. In this case, we start using the term "wound recognition" as the equivalent expression of "wound boundary determination", since both have the same goal. In later sections, we may interchangeably use either one of these two terms to

refer to the same task. In computer vision, the object recognition is a task that identifies objects and further determines the object boundaries in an image or video sequence. As concluded in [76], machine learning (ML) is currently the "*only known way to develop computer vision systems that are robust and easily reusable in different environments*". Hence, we plan to base our wound recognition method on a machine learning model. Based on the literature review in Chapter 1, the bottom-up machine learning based scheme is most widely used in current wound tissue recognition tasks and the corresponding experimental results prove its effectiveness.

Therefore, in this section, we will develop a foot ulcer recognition system, by applying the bottom-up object recognition scheme, as illustrated in Figure 4.2. An image, which may or may not contain the target object, is first segmented into a group of homogenous regions (so-called super-pixels). Then a feature descriptor is generated for each region. This super-pixel based descriptor can also be referred as the "*local feature representation*" of the wound image. Then these descriptors will constitute the input into a binary classifier whose output is a Boolean value: "true" represents that the current super-pixel is classified as "wound", or "false" represents that the super-pixel is classified as "non-wound". Usually, if there is an object and we need to accurately measure or analyze it (such as the wound area measurement and analysis), some methods will be utilized to further delineate the object boundary. In the bottom-up scheme, the object boundary can be easily determined by grouping all super-pixels classified as "wound" together. The binary classifier is the core of the object system and it is implemented by using a well-designed machine learning method. Inside the dash frame on the right-hand side of Figure 4.2, a typical flow of the supervised machine learning scheme (which is based on the training dataset with each sample labeled manually) is illustrated. Actually, we can build our wound recognition system on the basis of this general workflow. The key techniques used in this system consist of image segmentation (for local representation image), object representation (including feature detection and description) and classifier training and testing (based on a selected machine learning scheme).

4.2 Wound Descriptors

In this section, we will explore the technical details about three types of super-pixel based feature descriptors and compare their performance by applying a SVM based binary wound classifier to these descriptors. Two of them have previously been applied to wound tissue classification in [23] and [7], respectively. The simplified version of the third descriptor has been used for object recognition from natural scene images [77]. For better performance on our wound recognition task, we augment it by Dominant Color information.

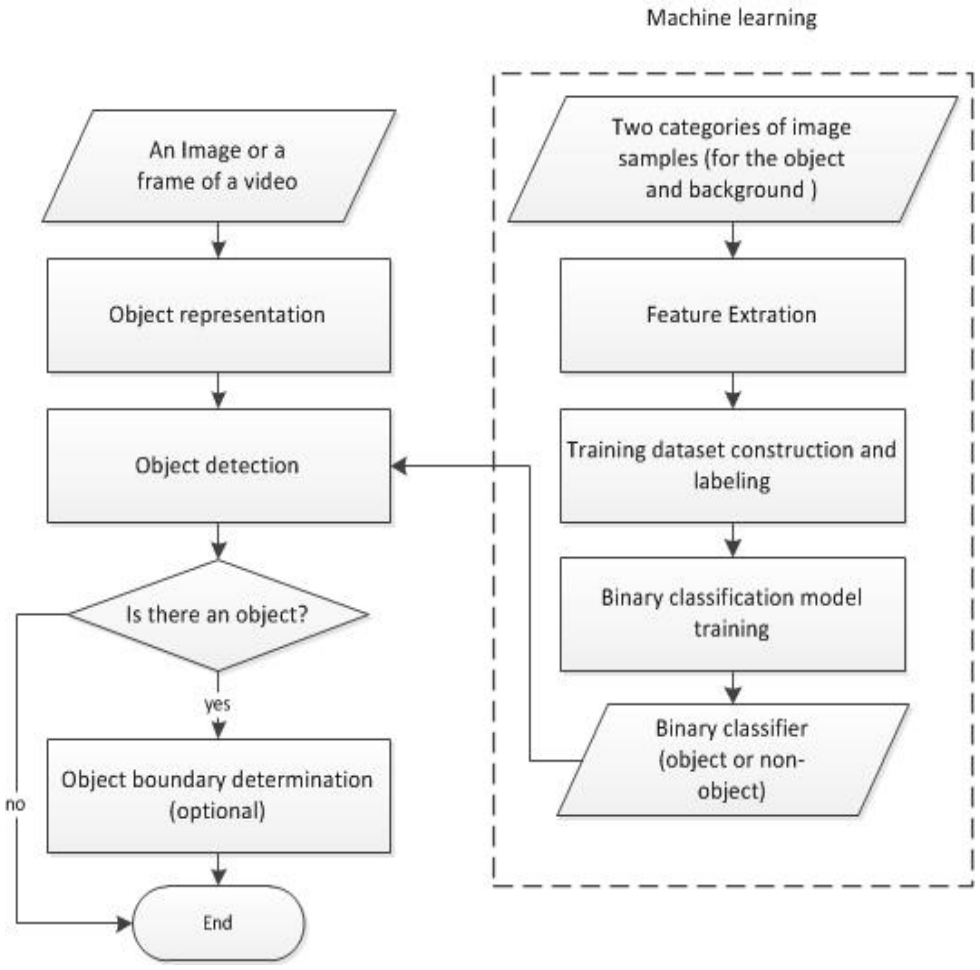


Figure 4.2 A basic flow for machine learning based object recognition system

4.2.1 Wound Descriptor Based on Color and Gray-Level Co-occurrence Matrix Texture

In [23], the authors used color and texture descriptors which have already been applied in dermatological imaging systems for the wound tissue classification task. Color is claimed to be most important visual cue as demonstrated by the red-yellow-black healing visual assessment of chronic wounds [35] during clinical visits. The color descriptors that the authors extracted were: the Mean Color Descriptor (MCD) [78], the locally adapted Dominant Color Descriptors (DCD) based on the mean-shift clustering algorithm [23], [78], [44], and 2-D and 3-D color histograms [21] [79] [80] tested in different color spaces and sizes. In addition, it is argued that texture is also a helpful discriminative feature for tissue classification if it is computed in a neighborhood in a properly specified shape and scale. According to [23], practical experience indicates that texture, although not as dominant as the color information, still provides complementary data for wound recognition when contrasted with healthy skin tissues. In [23], a number of widely used texture descriptors of different color channels (RGB for example) or gray level image (GL) were extracted from super-pixels in wound images. These descriptors are: Gabor based features (GAB) calculated on five scales, local binary pattern (LBP) histograms [21], Haralick Gray Level Co-occurrence Matrix features (GLCM) [81] and the normalized texture contrast and anisotropy (CA) [23]. The performance of these feature descriptors were applied to the wound tissue classification task reported in [23], and the results are shown in Table 4.1. The performance measures used in [23] are sensitivity (also called true positive rate or recall) and specificity [82] (also called true negative rate). The sensitivity indicates the proportion of wound regions (in ground truth images) which is correctly classified by the trained classifier, and the specificity measures the proportion of non-wound super-pixels which are correctly identified.

We can read from the results in table 4.1 that the sensitivity is always lower than specificity for any descriptor, which may indicate that the binary classification task of the wound and non-wound region suffers mainly from the errors of mistaking the wound regions as non-wound ones. Another finding is that the combination of the MCD, DCD and GLCM descriptors provides the best wound tissue classification performance [23].

Hence, we will apply this combined set of descriptors as one candidate for our foot ulcer recognition task.

Table 4.1 wound tissue classification performance of several color and texture descriptors [23]

Feature	*Descritpor*	*Sensitivity (%)*	*Specificity (%)*
Color	*Histogram-RGB*	*58*	*87*
	Histogram-LAB	*66*	*87*
	Histogram-HSV	*62*	*86*
	Histogram-nRGB	*57*	*89*
	MCD-DCD	*67*	*89*
Texture	*LBP*	*30*	*78*
	GLCM	*54*	*82*
	GAB	*47*	*81*
	CA	*32*	*79*
Color + Texture	*MCD+DCD+GLCM*	*77*	*92*

For the color descriptor part, we first use 3 color components (the L, a and b components in CIE LAB color space), which are obtained from the mean shift segmentation algorithm introduced in Chapter 2, and this 3-element color vector is called the dominant color for the current region [83]. In order to compute the dominant color descriptor, the colors present in a given region are first clustered. This results in a small number of colors, and the percentages of these colors are calculated [83]. Optionally, the variance of a given dominant color may also be computed. The percentages of the colors present in the region should add up to 1. Finally, a spatial coherence value, which is used to differentiate large color blobs from colors that are distributed widely on the entire image, is generated. The mathematical definition of this descriptor is shown as below.

$$F = \{(c_i, p_i, v_i), s\}, (i = 1, 2, ..., N) \qquad \textit{Equation 4.1}$$

where c_i is the i^{th} dominant color and p_i is its percentage value and v_i is its color variance, and where N represents the number of dominant color cluster centers. The spatial coherence s is a single number that represents the overall spatial homogeneity of the dominant colors in the image. Moreover, we add the mean color descriptor, which is also a 3-element color vector in CIE LAB color space, into the wound descriptor. This mean color descriptor is the mean vector of the three channel mean values of all pixels

for this super-pixel region in the original image. Finally, we calculate the occurrence probability P_o, defined as below, of the dominate color in the current region.

P_o = # of pixels with the dominate color in the current region /# of pixels in the current region *Equation 4.2*

In conclusion, a 7-element color feature descriptor has been generated.

The texture features are extracted based on the GLCM [81]. According to [81], the angular second moment, contrast, correlation of the GLCM provide a good representation of the significant texture properties for certain image regions. A co-occurrence matrix is a matrix that is defined over a image or a sub-region (in most cases, a rectangle region) of a image to be the distribution of co-occurring values at a given offset (in this disserttation, we set the offset as 1 pixel). Mathematically, a co-occurrence matrix *C* is defined over a *n x m* image *I*, with the offset vector $(\Delta x, \Delta y)$ as parameters. The exact formulation is given as below.

$$C_{\Delta x,\Delta y}(i,j) = \sum_{x=1}^{n}\sum_{y=1}^{m} if\,((I(x,y) == i)\,\&\,(I(x+\Delta x, y+\Delta y) == j)) \qquad \textit{Equation 4.3}$$

where *i* and *j* are the image intensity values of the image, *x* and *y* are the spatial coordinates in the image grid and the offset vector consists of values on different directions. As suggested in [81], four directions will be applied: horizontal, vertical, left-diagonal and right diagnoal. The corresponding four offset vectors are $(0,\pm 1)$, $(\pm 1,0)$, $(\pm 1,\pm 1)$ and $(\pm 1,\mp 1)$. Generally, the co-occurrence matrix will be normalized by dividing each element by the total number of pairs of pixels involved in the calculation of eq. (4.3). Finally, four independent co-ocurrence matrices are generated (one in each direction out of four) and all elements are in the range of 0-1. Hence, these matrixes can also be viewed as co-occurrence probability matrices [84].

Based on the GLCMs, several texture properties can be calculated. As recommended in [81] and [84], we compute 5 texture features for each GLMC:

(1) ngular second-moment, defined by eq. (4.4), is an indicator of the homogeneity of a super-pixel region;

(2) Contrast, defined by eq. (4.5), is an indicator of the contrast or amount of local variation presented in a super-pixel region;

(3) Correlation, defined by eq. (4.6), is an indicator of gray-tone linear dependencies in a super-pixel region;

(4) Entropy, defined in eq. (4.7), is an indicator of the degree of randomness in a super-pixel region;

(5) Inverse difference momment, defined in eq.(4.8), which is another indicator of the homogeneity.

$$f_{angular-second-moment} = \sum_{n}\sum_{m} p_{nm} \qquad \textit{Equation 4.4}$$

$$f_{contrast} = \sum_{k=0}^{N_g} k^2 \sum_{n=1}^{N_g} \sum_{\substack{m=1 \\ |n-m|=k}}^{N_g} p_{nm} \qquad \textit{Equation 4.5}$$

$$f_{correlation} = \frac{\sum_{n}\sum_{m}(nm)P_{mn} - \mu_x \mu_y}{\sigma_x \sigma_y} \qquad \textit{Equation 4.6}$$

$$f_{inverse-difference-moment} = \sum_{n}\sum_{m} \frac{1}{1+(n-m)^2} p_{nm} \qquad \textit{Equation 4.7}$$

$$f_{entropy} = -\sum_{n}\sum_{m} p_{mn} \log(p_{mn}) \qquad \textit{Equation 4.8}$$

where *n* and *m* are the indices of the all gray level ranging from 0-255, μ_x, μ_y, σ_x and σ_y are the means and standard deviations of the marginal probability p_x and p_y for each GLCM, respectively. p_{nm} is the element of the GLCM at spatial position *(n,m)* [81]. N_g is the total number of gray levels, which should be equal to 255 in our case.

In conclusion, incorporating the five features for each GLCM means we will obtain a 20-element texture vector of all GLCMs for each region. When adding to that the 16-element color descriptor acquired previously, a 36-element feature descriptor for each region will be generated.

4.2.2 Wound Descriptor Based on Color Histogram and Wavelet Texture

In [7], the color features are extracted using three different color models: CIE Lab, RGB and normalized-RGB. the normalized-RGB feature is extracted from the original RGB values by a straightforward normalization method $r = R/(R+G+B)$; $g = G/(R+G+B)$; $b = B/(R+G+B)$ [7]. It is obvious that the sum of the three normalized elements should be equal to 1, i.e. $r+g+b=1$. Hence, the third component is redundant and can be skipped for reducing the dimensionality of feature descriptors. According to [7], the normalization procedure can eliminate the influence on color features due to the brightness of the lighting resource; specifically, using the normalized RGB color features enhances the classification robustness against the illumination change. A significant characteristic of this normalized color features is as follow: for matte surfaces, if ambient light is not taken into consideration, normalized-RGB features are robust (under certain conditions) to changes in the angle between the surface normal and the direction of the illumination resource [85]. This invariance to ambient light orientation serves well for extracting intrinsic characteristic of the original image under different illumination conditions. As introduced in [7], three types of color based features in various color spaces (CIE Lab, RGB and normalized RGB) can be extracted from the super-pixels in each wound image: 1) pixel based color features, 2) histogram based color features, and 3) wavelet based texture features for each channel isolated image in RGB space, i.e. three 8-bit image for R, G, B channel, respectively.

For pixel based color features, the mean and the variance of each color component in both CIE Lab and normalized-RGB color models are extracted. Alternatively, the following features can be extracted from the normalized color histogram in RGB and normalized-RGB color space ($\sum_i h_i = 1$, where h_i is the i^{th} bin): the two highest peaks of the results from moving average filter [86]; the intensity levels for whose bin value larger than the 50% of maximum amplitude in ascending-ordered histogram; the variance; the skewness $\sum_{i=1}^{n}(h_i-\mu)^3/((n-1)\sigma^3)$, where μ is the mean and σ is the deviation; the energy ($\sum_{i=1}^{n} h_i^2$); and the entropy ($-\sum_{i=1}^{n} h_i \log(h_i)$) [7].

Wavelets make it possible to analyze a signal in different levels of details [83]. In the case of 2D image which can be viewed as 2D digital signal, the wavelet theory gives us the ability to analyze the images at different resolutions or scales. For example, if the original image is in size of 512 x 512, we can study the frequency responses in resolutions of 256 x 256, 128 x 128, 64 x 64 and so on. In particular, for texture analysis of images, the capability of describing the structure of the image at different scales is important since a texture can vary significantly if the distance between the object and the camera has been changed.

The function shown in eq. (4.9) represents the general formulation of wavelets.

$$\psi(\mathrm{t}) = \frac{1}{\sqrt{a}}\psi(\frac{t-b}{a}) \qquad \textit{Equation 4.9}$$

where *a* stands for the frequency scaling factor and *b* represents the time displacement. If we define *a*, such that $a = 2^m$, the wavelet transform is referred to as the Dyadic Wavelet Transform (DWT), and its specific formulation can be derived from a smoothing or scaling function denoted as $\theta(t)$. According to [83], from the viewpoint of both the scaling function and the wavelet, two FIR filters can be formulated to compute the DWT. These filters are referred as *h* and *g* and are a low-pass and a high-pass filter, respectively. The 2D DWT transform can be illustrated in Figure 4.3 [83], where * is the convolution operator and the subscript for each filter indicates the direction in which the convolution is applied. Finally, the down-sample operation here means scaling down the rows or columns by a factor 2. As seen from Figure 4.3, the 2D DWT results in four output images. In each filtering step, a low frequency image (similar to the intensity image representing the brightness) L_{j+1} and three detail images are produced. The detail images $D_j^{1...3}$ contain the details (high frequency components) extracted from D_j that are not presented in D_{j+1}. This procedure will be repeated recursively until a specified iteration count is reached.

All four output images obtained by applying 2D DWT can be viewed as basis for extracting texture features. Each of these images reveals important gradient information at a given direction (horizontal, vertical and diagonal) and scale. According to [7], two

levels of DWT have been applied for wavelet based texture feature extraction. The first level wavelet filters were applied to each color channel of the image derived with the Daubechies fourth order function [83]. Then, the second level wavelet filters were applied to the low frequency image (LL) [83]. On each output channel from wavelet filtering (LH, HL and HH) the energy per super-pixel ($\frac{1}{area}\sum_{p\in \mathrm{Re}gion} p^2$, where p indicates the pixel value on each position in the output channel and "area" represents the super-pixel size in terms of super-pixel) was computed on each segmented region.

Finally, a total of 63-element color texture descriptor has been obtained [7].

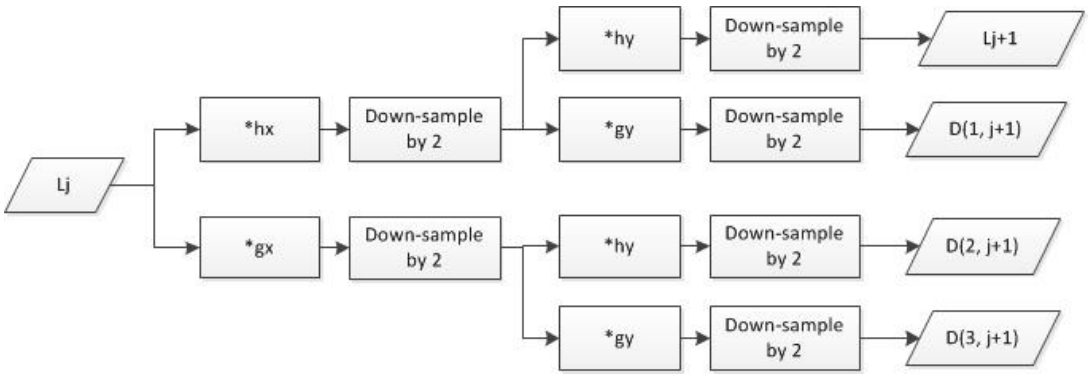


Figure 4.3 2D DWT using FIR filters

4.2.3 Wound Descriptor Based on Bag-of-word Representation of DSIFT and Dominant Color Descriptor

In this section, we create a new wound descriptor for each super-pixel by combining the Bag of Words (BoW) histogram based on local Dense SIFT (DSIFT) descriptors and the dominant color descriptor mentioned in Section 4.2.1.

In [77] [87] [88], the bag-of-words representations for global or regional image features are widely used for object recognition tasks in natural scene images and the detection of humans in the image. In [77], a bag-of-words based classifier which uses the super-pixels as basic input unit was designed. To generate the bag-of-words based descriptor for each super-pixel, standard SIFT descriptors were first extracted "at each pixel of the image with a fixed scale and orientation" using the accelerating SIFT extraction approach. This is called the *dense SIFT* descriptor extraction [77] since the SIFT descriptor was extracted at each pixel of the image, instead of selected locations in the standard

procedure. According to [77], when applying the SIFT descriptor to tasks such as bag-of-words based object category classification, experimental evaluations show that better classification results are often obtained by computing the SIFT descriptors over dense grids in the image domain, as opposed to the extraction at sparse interest points proposed in [89]. A basic explanation for this is that a larger set of local image descriptors computed over a dense grid usually provide more information than corresponding descriptors evaluated at a much sparse set of image points.

This direction of development was initiated in [90] and has now been established as a state-of-the-art approach for visual object category classification. As mentioned in [91], the computation of dense SIFT descriptors is usually accompanied with a clustering stage, where the individual SIFT descriptors are reduced to a small vocabulary of visual words, which can then be combined with a bag-of-words model methods [91]. The extracted descriptors are clustered to generate K (a predefined number) centers (so-called a dictionary of visual words) using a K-means algorithm. Then each descriptor is classified based on its nearest center as determined by the minimum square distance rule. The descriptors in each super-pixel s_i are aggregated into one normalized histogram $h_i \in R_+^K$ based on the classification results, where K is the number of words predefined in the *codebook* (the set of clusters resulted from the K-means algorithm).

According to [77], there are two major shortcomings of the dense SIFT descriptor. The first shortcoming for the Bag-of-word based descriptor is that the resulting classifier is very "specific" [77]. It identifies the similarity between super-pixels in the training data without examining the neighborhood region structure. In the wound recognition case, this shortcoming means the some outlier super-pixels far from the actual wound region may be incorrectly classified as wound due to its similarity in color with the actual wound. The second shortcoming of training a local classifier for each single super-pixel is that the associating histogram with each super-pixel contains too many zero elements (sparseness). This finding is easy to understand since the pixel values in each super-pixel are very homogeneous in terms both color and texture by the definition. Second, DSIFT features are extracted in a fixed-scale format and on an image grid with finer resolution than standard SIFT feature. Hence, most super-pixels possibly have many DSIFT

descriptors that are clustered to the same visual word in the generated visual dictionary [77]. Based on [77], the sparse feature vectors will heavily undermine the discriminative ability of the trained classifier using modern machine learning techniques, such as support vector machine (SVM) and artificial neural network (ANN).

In [77], these two shortcomings have been corrected by applying a histogram merging techniques in the super-pixel neighborhood. In [77], an adjacency graph $G(S,E)$ is defined on super-pixels s_i in an given image, where h_i^0 is the non-normalized histogram calculated for all dense SIFT descriptors extracted from this region. E is the set of edges connecting pairs of adjacent super-pixels (s_i, s_j) in the image, and $D(s_i, s_j)$ is the length of "the shortest path" between two super-pixels. Then, h_i^N is the histogram obtained by merging the histograms of the super-pixel s_i and its neighbors, i.e. we update the value for each bin in the histogram by the sum of the given bin value in all histograms in a defined neighborhood. The histogram merging process is described in formula defined as below [77].

$$h_i^N = \sum_{S_j | D(s_i, s_j) \le N} h_j^0 \qquad \textit{Equation 4.10}$$

where N represents the radius of the neighborhood region in center of the current super-pixel. In another word, each super-pixel in this neighborhood should be less than N nodes away from its center in the adjacency graph.

According to [77], this solution can provide an effect similar to regularization (a well-known procedure to avoid overfitting i.e., the learned model works promisingly for classification on the training dataset but shows poor generalization for the testing set) for the SVM or neural network based classifier [77]. It also provides spatial consistency in our classification because, as we increase N, histograms of adjacent super-pixels have more features in common.

Second, since more super-pixel based neighborhood information has been considered using the histogram merge technique, long-range connection between object pixels has been better described. Therefore, by adjusting the parameter N, we can easily handle target objects at different scales [77]. This is important for our wound recognition task

since the size of the wound may vary a great deal, and also the object-to-observer distance will possibly cause the change of wound scale. However, as stated in [77], increasing *N* will blur object boundary and further result in recognition errors near the actual object boundary. According to [77], the conditional random field (CRF) based technique can be applied to refine the resulting boundary. More details of CRF can be found in Chapter 5.

After being combined with the dominant color descriptor (DCD) introduced earlier, the final descriptor contains 162 elements.

4.2.4 Principal Component Analysis Based Dimensionality Reduction

After a set of color and texture features is extracted from each resulting region, a Principal Component Analysis (PCA) allows the reduction in the dimensionality of the initial color and texture feature space [92]. The dimensionality reduction can enhance the time performance of model training and object classification. PCA is mathematically defined as an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by some projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on.

Consider a data matrix *X* in dimension of $m \times n$, whose columns have zero mean (the sample mean of each column has been shifted to zero). Each row represents a zero-mean shifted instance feature vector of length of *n*. Mathematically, the transformation is defined by a set of vectors of weights (transformation matrix) of *p*-dimensional (where *p* is supposed to be much more smaller than *n*, which will result the dimension reduction) . The weights are defined as $w_{(k)} = (w_1, w_2, ..., w_p)_{(k)}$ that maps each row vector $x_{(i)}$ of *X* to a new vector of principal components $t_{(i)} = (t_1, t_2, ...t_p)_{(i)}$ given by

$$t_{(i)} = x_{(i)} \cdot w_{(k)} \qquad \textit{Equation 4.11}$$

in such a way that the individual variables of *t* considered over the data set successively inherit the maximum possible variance from *x*, with each weight vector constrained to be a unit vector. The transformation matrix can be calculated from the *p* eigen-vectors

corresponding to the *p* largest eigen-values of the covariance matrix $X^T X$. Note that the dimension of each feature vector is reduced from *n* to *p*.

4.2.5 Wound Descriptor Evaluation

In our experiments, we evaluate 4 combined feature descriptors by applying them to the single-stage binary SVM based wound classifier on 100 sample images of real wounds using leave-one-out cross validation scheme [1]. More details about the machine learning techniques can be found in the next section (Section 4.3). Descriptor 1 and 2 come from the contribution in Section 4.2.2 and 4.2.3, respectively. In Descriptor 3, we use the color features in Descriptor 1 and 2 (mean and variance of color in CIE Lab and normalized RGB space, two highest peaks, dominant levels, variance, skewness, energy and entropy of the color histogram in RGB and normalized RGB space), and we append the color features with the BoW histogram representation of local DSIFT features as mentioned in Section 4.2.4. In another word, Descriptor 4 is like a combination of descriptor 1-3. After applying the PCA dimention reduction method by maintaining 99% total invariance, the resulting descriptor length is 12, 19, 33, 39 for descriptor 1-4, respectively.

Based on the results in Table 4.2, the general sensitivity and specificity measures when the evaluation is carried out on the entire image show that the Descriptor 3 provide the best performance (71.4% for sensitivity and 97.2% for specificity). However, when the significant area is confined to the wound and small number of surrounding skin super-pixels only, Descriptor 1 performs better than the others (72.2% for sensitivity and 96.1% for specificity). These results may indicate that the BoW features are useful for ruling out the irrelevant background information, but the BoW features also demonstrate some inability to distinguish between the wound and healthy skin when compared to the wavelet and GLCM based texture features. This observation directly inspire us for the cascade two-stage classifier design introduced in the next section. Morever, the longest feature descriptor 4, while consuming more computing time as shown Table 4.3, provides no significant performance improvement. Based on the discussion above, we applied the Descriptor 3 and 2 to the first and second stage classification, respectively. The cascade two-stage classification system will be introduced in the next section (Section 4.3).

4.3 A Novel Two-Stage Cascade Binary Wound Classifier

The major purpose of this chapter is to design a smartphone based wound image assessment system (bottom-up object recognition scheme based) for healing status tracking of diabetic foot ulcers. We have utilized several image processing techniques and proposed a cascade two-stage approach based on Support Vector Machine to determine the wound area. Support Vector Machine based methods have proved to be effective when applying to similar clinical problems [23], [93]. Our proposed methodology consists of the following steps: 1) use the Simple Linear Iterative Clustering (SLIC) method to segment the images into a number of super-pixels as mentioned in Chapter 2; 2) extract significant color and texture features from these super-pixel and use Principle Component Analysis (PCA) technique to reduce the dimensionality of the feature space as discussed in Section 4.2; 3) in the first training stage, train *k* SVM binary classifier based on partially different training images; 4) in the second training stage, train one SVM binary classifier based on the incorrectly classified test instances from the first stage, and 5) refine the determined wound boundary using Conditional Random Field techniques.

Table 4.2 Wound recognition performance evaluation for different combined descriptors. 1) descriptor1: MCD + DCD + GLCM, descriptor 2: MCD + color histogram + wavelet, descriptor 3: MCD + color histogram + BoW, 4: MCD + DCD + color histogram + wavelet + GLCM + BoW; 2) descriptors are evaluated on two different types of image area: the entire image area containing wound bed, healthy skin and irrelevant background and the area contianing only wound bed and small amount of surrounding healthy skin

Descriptor ID	*Classification results (α = entire image)*		*Classification results (α = wound + healthy skin only)*	
	Sensitivity (%)	*Specificity (%)*	*Sensitivity (%)*	*Specificity (%)*
1	*68.7*	*86.2*	*72.2*	*86.1*
2	*68.3*	*86.9*	*74.5*	*88.3*
3	*71.4*	*87.2*	*69.4*	*86.6*
4	*70.4*	*87.0*	*72.3*	*85.8*

Table 4.3 Computation time evaluation for different combined descriptor generation (include the PCA dimension reduction) based on Nexus 5 smartphone

Descriptor ID	*1*	*2*	*3*	*4*
Computation time	*4.5*	*4.2*	*5.6*	*8.8*

4.3.1 Methodology Overview

Figure 4.4 describes our overall process for the wound boundary determination on diabetic foot ulcer images. We captured the sample foot ulcer images in the Wound Clinic at University of Massachusetts Medical School by tracking a number of patients over a period of two years, based on an approved IRB protocol. In most cases, the foot ulcers were located at the sole of foot. As mentioned in Section 2.1, to facilitate the image capture process, especially for elder patients with limited mobility, and to maintain fixed range between camera and wound and consistent illumination conditions, we used the image capture box described in Section 2.1. As mentioned in Section 3.4, we use the Nexus 5 smartphone whose F2.0 auto-focus lens can ensure well-focused, high resolution images within a certain distance. Then, the SLIC segmentation algorithm introduced in Section 2.2 is applied to accomplish the super-pixel segmentation task. For each super-pixel, a color and texture based descriptor is extracted using one of the methods introduced in Section 4.2. The PCA based dimension-reduction method can be applied optionally to improve the training efficiency. The modules presented in the second and third columns in Figure 4.4 will be described in Section 4.3.3, except for the color segmentation block which will be presented in Chapter 6.

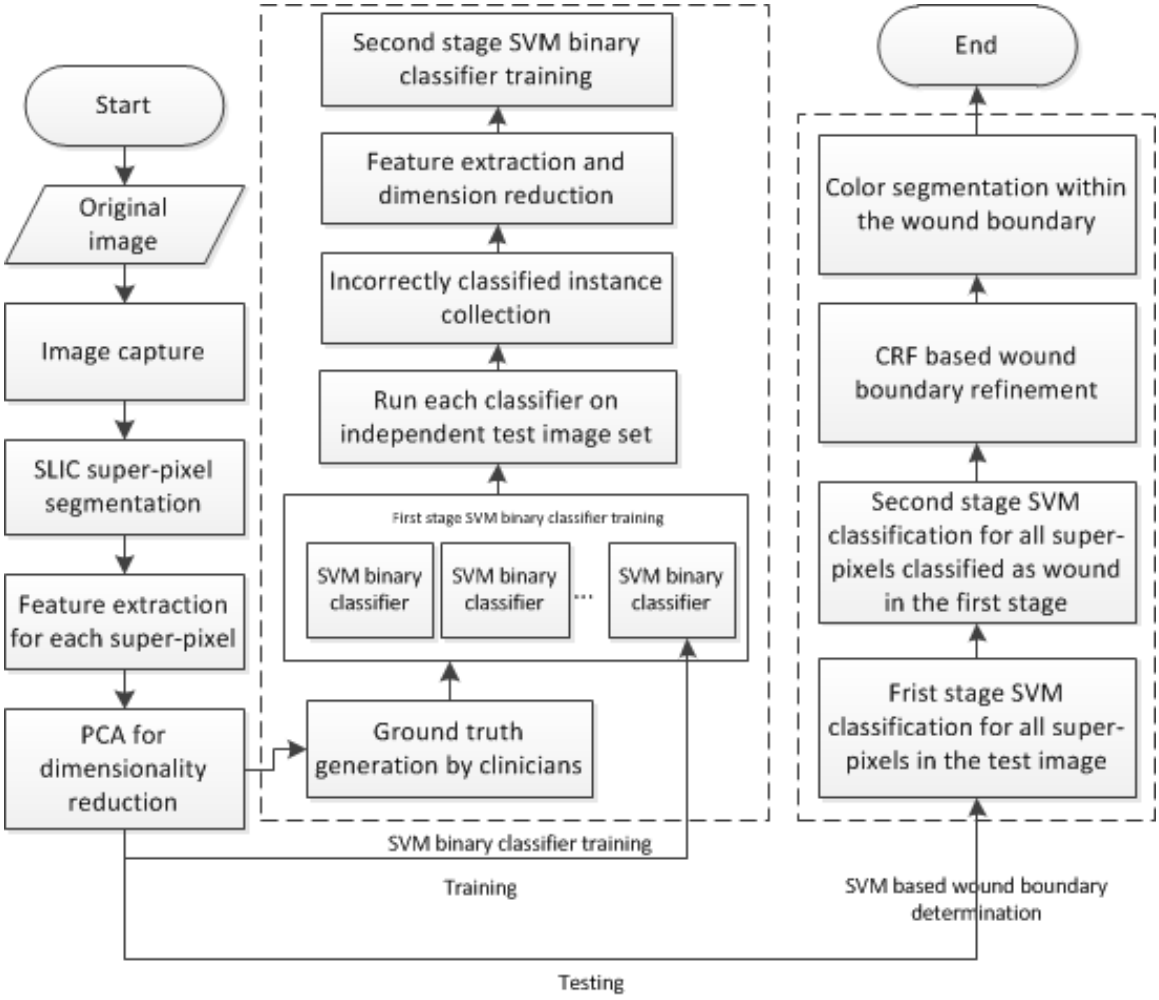
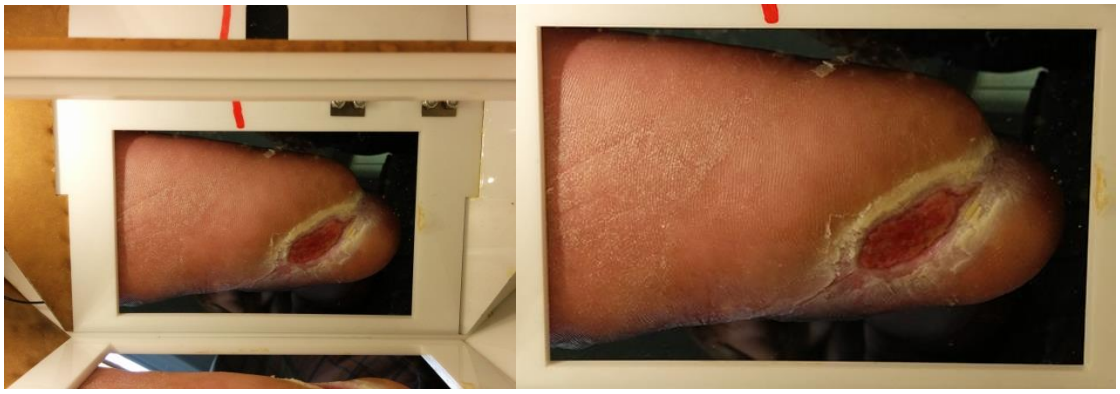


Figure 4.4 Complete flowchart for the wound boundary determination system (the details are introduced in the Methodology Section)

We selected 100 photographs which were considered to be an appropriate data set for wound classifier training. As mentioned in Section 3.4, for an efficient smartphone based implementation, we down-sampled the original image by a factor 4 in both the horizontal and vertical directions. Furthermore, we frame the wound images to remove irrelevant background information (Thanks to the image capture box, we can safely rule out the irrelevant background by simply framing the original image with a rectangle with fixed size, center and orientation). The final dimensions of the image are 560×320. The image cropping process can be illustrated in Figure 4.5.



(a) *(b)*

Figure 4.5 wound image framing, the irrelevant background information in (a) is removed and the resulting image is shown in (b)

Afterwards, a cascade two-stage SVM based wound boundary determination method is proposed based on the extracted feature descriptors. The generation process of ground-truth wound labeling will be introduced in Section 4.3.3. In the first stage, k-fold cross-validation, which will described in details in Section 4.3.3, is performed to train *k* SVM binary classifiers. After applying these *k* classifiers to different testing datasets, we collect all the incorrectly classified super-pixels to form a new training dataset for the next stage. In the second stage, we extract new feature descriptors for these super-pixels and train a new SVM binary classifier. When analyzing a new wound image, first we apply the first stage SVM classifier to all super-pixels. Then, for all the super-pixels classified as "wound", we applied the second stage SVM classifier to remove false positives. Finally, a Conditional Random Field (CRF) based algorithm is applied to further refine the determined wound boundary. In following sections, each module of the flowchart will be discussed in details.

4.3.2 SVM Based Binary Classifier

- *SVM Basics*

As mentioned in the introduction section, the SVM based classifier is the most widely used supervised machine learning technique because of its capability of dealing multi-dimensional features and convenience of over-fitting protection [76]. The major idea is as follows: the non-linear separable training data in low dimensional feature space becomes linear separable in higher dimension space, by mapping the feature vector from lower

dimensional space to higher dimensional space using an appropriate kernel function [94], [95]. The SVM classifier selected here is a soft-margin algorithm (so-called C-SVM) [95]. In the next several paragraphs, we will give a brief introduction of the basic SVM and C-SVM algorithm.

SVMs are built upon the concept of a "margin" – "the distance between the two sides of a hyperplane that separates two data classes" [94]. The class separation task can be converted to maximizing the margin, i.e. producing the largest distance between the separating hyperplane and the training samples on either side of it has been proven to reduce an upper bound on the expected generalization error [76] [96].

If the training data is from classes which are linearly separable, then a pair (w,b) exists such that the following equations are satisfied.

$$w^T x_i + b \geq 1, \textit{ for all } x_i \in P \qquad \textit{Equation 4.12}$$

$$w^T x_i + b \leq -1, \textit{ for all } x_i \in N \qquad \textit{Equation 4.13}$$

The decision rule is given by the following equation.

$$f_{w,b}(x) = \mathrm{sgn}(w^T x + b) \qquad \textit{Equation 4.14}$$

where w is termed the weight vector and b the bias (or $-b$ is termed the threshold).

It is easy to show that, when it is possible to linearly separate two classes, an optimum separating hyperplane can be found by minimizing the squared norm of the separating hyperplane. The minimization can be formulated as a convex quadratic programming (QP) problem [94]:

$$\min_{w,b} \phi(w) = \frac{1}{2}\|w\|^2 \qquad \textit{Equation 4.15}$$
$$\textit{subject to } y(wx + b) \geq 1, i = 1, \ldots l$$

In the case of linearly separable data, once the optimum separating hyperplane is found, data points that lie on its margin are known as support vector points and the solution is represented as a linear combination of only these points. Other data points can be ignored. This scheme can be illustrated in Figure 4.6. Therefore, the model complexity of an SVM is unaffected by the number of features encountered in the training data (the number of

support vectors selected by the SVM learning algorithm is usually small). For this reason, SVMs are well suited to deal with learning tasks where the number of features is large with respect to the number of training instances. A general set of algorithm steps for SVMs is formulated below.

- ✓ Step 1: introduce non-negative Lagrange multiplier α_i, one for each of the inequality constraints in eq. (18). This gives Lagrange equation as below:

$$L_P = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i y_i (x_i \cdot w - b) + \sum_{i=1}^{N} \alpha_i \quad \textit{Equation 4.16}$$

- ✓ Step 2: minimize L_P with respect to w and b, which is a convex quadratic programming problem.
- ✓ Step 3: in the solution, those points for which $\alpha_i > 0$ are called "support vectors".

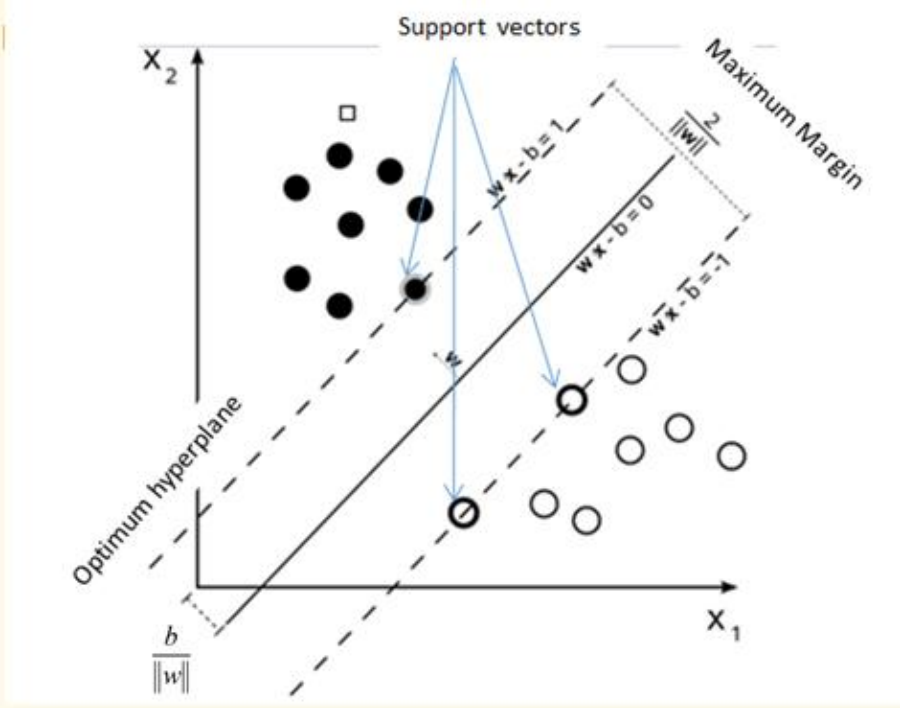


Figure 4.6 Maximum margin separator [76]

- *Kernel based soft margin SVM (C-SVM algorithm)*

Even through the maximum margin allows the SVM to select among multiple candidate hyperplanes, for many datasets, the SVM may not be able to find any separating hyperplane at all because the data contains instances of misclassification. The problem

can be addressed by using a soft margin that allows some misclassifications of the training instances to be present [95]. The implementation of a soft margin is carried out by introducing positive slack variables ξ_i , $i = 1,...N$ in the constraints, which then become as below.

$$\begin{aligned} & w^T x_i - b \geq 1 - \xi_i \quad for\ y_i = 1 \\ & w^T x_i - b \geq -1 + \xi_i \quad for\ y_i = -1 \\ & \xi_i \geq 0,\ i = 1....N \end{aligned} \qquad \textit{Equation 4.17}$$

Thus, for an error to occur, the corresponding ξ_i must exceed unity, so $\sum_i \xi_i$ is an upper bound on the number of training errors. In this case the Lagrangian is as below.

$$L_P = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i \{y_i (x_i \cdot w - b) - 1 + \xi_i\} + C \sum_{i=1}^{N} \xi_i - \sum_i \mu_i \xi_i \qquad \textit{Equation 4.18}$$

where μ_i are the Lagrange multipliers introduced to enforce positivity of the ξ_i .

Although the C-SVM algorithm can provide good classification performance for simple and ideal cases, most real-world problems involve non-separable data for which no hyperplane exists that successfully separates the positive from negative instances in the training set. One solution to the inseparability problem is to map the data into a higher dimensional space and define a separating hyperplane there. This higher-dimensional space is called the transformed feature space, as opposed to the input space occupied by the training instances [94].

With an appropriately chosen transformed feature space of sufficient dimensionality, any consistent training set can be made separable. A linear separation in transformed feature space corresponds to a nonlinear separation in the original input space. Mapping the data to some other (possibly infinite dimensional) Hilbert space *H* as $\Phi : R^d \rightarrow H$. Then the training algorithm would only depend on the data through dot products in *H*, i.e. on functions of the form $\Phi(x_i) \cdot \Phi(x_j)$. If there were a kernel function *K* such that $k(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$, we could only need to use *K* in the training algorithm, and wound never need to explicitly determine Φ . Thus, kernels are a special class of functions that allow inner products to be calculated directly in feature space, without

performing the mapping described above [76] [94]. Once a hyperplane has been created, the kernel function is used to map new points into the feature space for classification [94].

4.3.3 Supervised Machine Learning Based Method for Wound Boundary Determination

- *Expert Labeling for Supervised Machine Learning*

In [7], [23], both Wannous and Verdedas developed a graphical user interfaces to deal with the time consuming task of having experts label a large set of more than 10000 regions (super-pixels) from the segmented wound images. It was reported in [7] that labeling one image needed 6 minutes. To save clinicians' time, we have implemented a simpler labeling method. For this work, a team of 3 expert wound clinicians from the Plastic Surgery Department at University of Massachusetts (UMASS) Medical School delineated the wound boundary for all 100 wound images, and each expert did the delineation independently. They used a laptop, installed with Photoshop and USB connected with an electronic pen and drawing panel. In this way, it took only 30 seconds on average for the clinicians to label wound image. To generate the ground truth binary label for each super-pixel (wound or non-wound), we first assigned each pixel a label by applying a majority vote scheme: if the current pixel is located inside the wound boundary delineations by more than two clinicians, we label it as wound, otherwise it is labeled as non-wound. The entire process is shown as in Figure 4.7. Then for each super-pixel, another majority vote scheme is applied: if more than half of all pixels in this super-pixel were labeled as wound, we assigned it as wound region, otherwise we assigned it as non-wound region. To validate our labeling method, we regenerated the wound boundary delineation based on super-pixels' labels in each image and compared it with the original delineation from the clinicians. The resulting similarity in pixel-level between these two set of delineations is higher than 96%. Hence, we can conclude that the novel method provides labeling results with good accuracy, although the second majority vote may cause some inaccuracy for regions near the wound boundary.

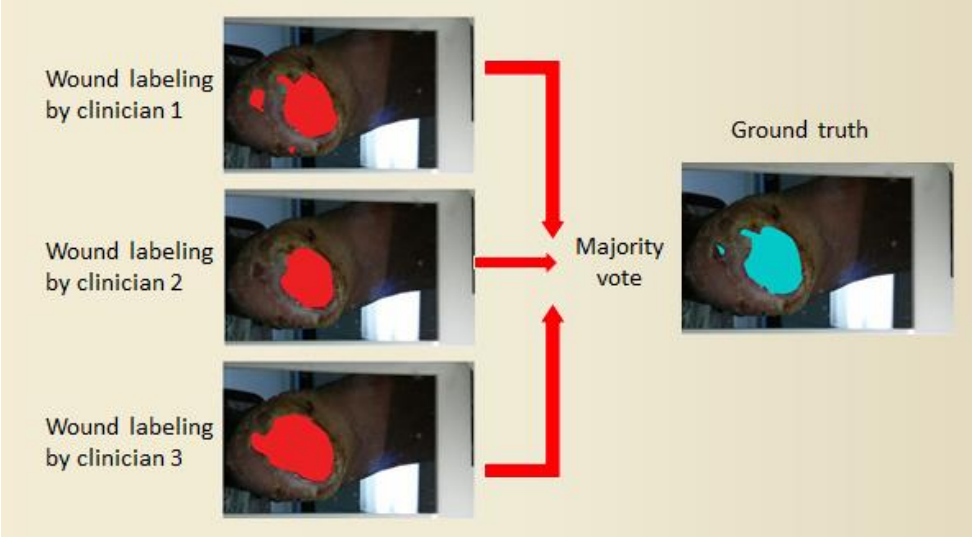


Figure 4.7 Majority vote scheme for ground truth labeling generation from three experts' labeling

- *Two-Stage Cascade SVM Based Machine Learning Architecture*

In [7], Verdedas et al created a classifier based on a stable neural network from a set of sub-classifiers trained by following a cross-validation strategy. The results has demonstrated that this approach provide promising wound tissue classification performance when dealing with images that only contains wound and surrounding healthy skins. In this work, we have expanded Verdedas's proposal to a two-stage cascade machine learning architecture based on Support Vector Machines (SVMs), as shown in following steps. The entire system has been implemented in JAVA by using the LibSVM library [97].

- ✓ Step 1: split the entire training image dataset into *k* subsets (folds) of equal size. Since we can control the segmented super-pixel number by using SLIC algorithm and since the dimensions of most training images are approximately the same, we have approximately an equal number of super-pixels for each training image. Hence, in each subset, we have both approximately equal number of images and super-pixels.
- ✓ Step 2: make the number of wound regions and non-wound regions from *k-1* subsets equal. Since we use the image capture box to photograph the foot, the number of wound regions is typically small compared to number of non-wound regions. This skewed distribution of the number of instances in the different categories undermines the performance of the trained classifier. The simple method for obtaining an equal

number of wound and non-wound regions is as follows:, if we have *m* wound super-pixels and *n* non-wound super-pixels ($n > m$) in the subset 1 to k-1, we will randomly select *m* non-wound regions from *n* in total. Consequently, we have *2m* training patterns.

- ✓ Step 3: follow a classical 10-fold cross-validation scheme to train a binary SVM based classifiers on these *2m* training patterns. Based on each subset, we further split this subset into 10 equal size folds. Then, we run the standard RBF-$\chi 2$ kernel based soft margin SVM (*C-SVM*) [95] 10 times, where each time one fold will be the validation set and the remaining 9 folds will be training set.
- ✓ Step 4: use the trained classifier to classify regions in the subset other than *k-1* subsets for training.
- ✓ Step 5: collect the incorrectly classified instances from the results in Step 4 into the training set for the second stage.
- ✓ Step 6: repeat first stage training (Step 2 to 5) *k* times and let each subset be the validation set exactly once.
- ✓ Step 7: train the second stage SVM binary classifier in the same way mentioned in Step 3.

Based on the findings in Section IV, we extract the descriptors consisting of the color features and the BoW histogram representation of DSIFT features for super-pixels for the first stage training set. For the second stage training, we extract the descriptors consisting of the color features and the wavelet based features for the second stage training set. In kernel based SVM training, the non-linear separable classes in low dimension feature space may become linear separable when mapping the features to higher dimensional space using a suitable kernel function [94]. Hence, the key point of the SVM classifier design remains the choice of the kernel function. Since no universal kernel will fit all applications, we evaluated different classical kernels, including linear, polynomial, perceptron and RBF-$\chi 2$, by computing ROC curves as shown in Figure. 4.8.

It can be seen that the RBF-$\chi 2$ gives the best performance. After the selection of a particular kernel, its parameters must be tuned. In our case, we utilized the soft-margin SVM with the RBF kernel. Hence, we have two parameters to be tuned: (C, Υ) where C is the regulation parameter and Υ is the kernel parameter [94]. The best parameter

combination is often selected by a grid search with exponentially growing sequences of the two parameters [94]. Each point in this grid will be applied to the SVM training process and the General Testing Error (GTE) on the validation sets will be evaluated. Then the optimal parameter combination is the one that provides the least GTE. After running the grid search for both stages, we found the optimal parameters for the two stages are ($C = 100, \Upsilon = 0.05$) and ($C = 85, \Upsilon = 0.12$), respectively.

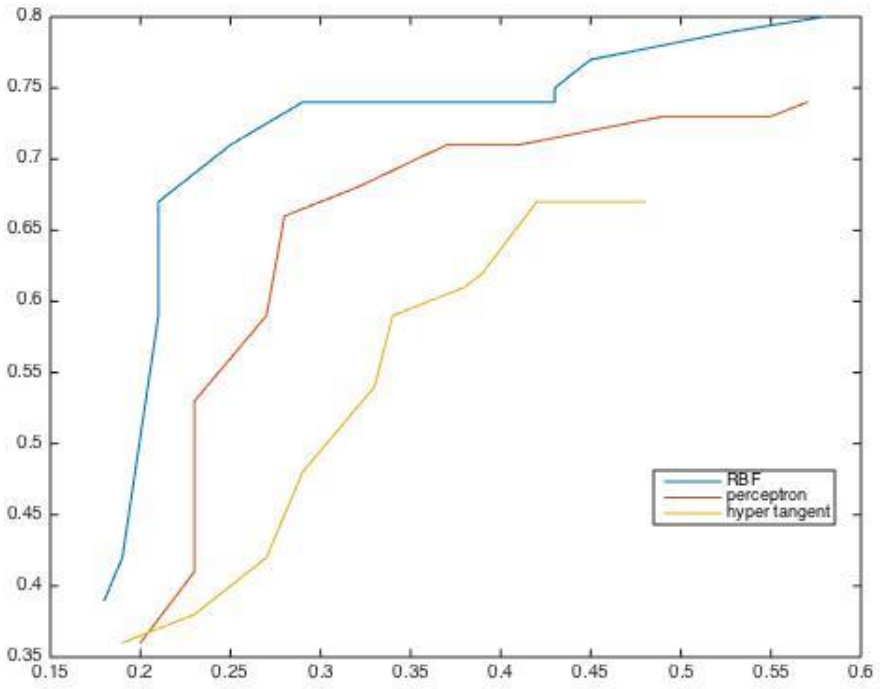


Figure 4.8 ROC curves by using different kernels for SVM training

To evaluate the performance of the two-stage SVM based approach more completely, we also compare it to other machine learning strategies, with the same descriptor as input. The reference strategies include the 1) single stage SVM based classifier with the same configuration as the first stage classifier in the two-stage approach, and 2) single stage classifier based on Artificial Neural Network (ANN)[1] with one hidden layer of 40 neurons [7]. As mentioned earlier, we use the specificity and sensitivity as the

[1] Artificial neural networks (ANN) is a machine learning strategy, inspired by human-brain activity, which is used to estimate classification model depending on high dimensional input. ANN is commonly design as a multi-layer perceptron structure, where each node is usually a simple logistic regression classifier taking the output of the previous layer as its input.

performance measures of the wound boundary determination methods. Table 4.4 shows that the two single stage methods, either based on SVM or ANN, provide approximately equal performance. More importantly, in spite of a little cost increase on time performance (as shown), our two-stage SVM based classifier outperforms both single-stage methods, achieving 5% higher in sensitivity and 8% higher in specificity. The significantly improvement in sensitivity shows that some incorrectly classification on original non-wound regions have been corrected in the second stage. Three examples of the wound classification results are shown in Figure 4.9.

Table 4.4 Wound boundary determination performance evaluation of different machine learning strategies. Method 1: single-stage SVM based method, method 2: single-stage ANN based method, method 3: novel two-stage SVM based method, method 4: two-stage SVM + CRF refinement. The classification efficiency measures the average time consumed by the entire wound boundary determination process for 100 images including super-pixel segmentation, descriptor generation and classification, and only method 4 includes the boundary refinement process

Method ID	*1*	*2*	*3*	*4*
Sensitivity (%)	*68.3*	*66.4*	*71.4*	*73.3*
Specificity (%)	*86.9*	*83.7*	*92.8*	*94.6*
Computation time (second)	*15.4*	*16.1*	*18.8*	*20.5*

- *Wound Boundary Refinement*

Even though the two-stage SVM based classifier significantly improves the wound boundary determination performance, there are still some misclassified regions. Most of the error regions are located near the wound boundary and connected to the target wound area. In addition, some misclassified regions may also be found inside the wound bed (holes), healthy skin or in the irrelevant background (outliers). To refine the wound boundary determination, we first generate a binary result image based on the wound boundary determination result, by assigning the value 255 to the pixels classified as wound and 0 to the pixels classified as non-wound.

Therefore, we have a three step approach for wound boundary refinement. The first step is to apply the Conditional Random Field technique to re-label the incorrectly classified

regions near the actual wound boundary. Second, the one precautionary step is to break narrow false bridges (connections between two pixels which should be classified into different categories) and fill the possible remaining holes inside the identified wound area by applying the closing image morphological operation (we used the circular structure element with radius = 3). At the last step, we remove the outlier regions if there are any, and this is easy because outliers are always disconnected from the main wound area and in small size. We can simply run a connected region detection method [20] and only keep the largest connected region. An example of the boundary refinement results after the two precautious steps are shown in Figure 4.9, Column 4. Especially when comparing last two images on the third row, we see significant refinement on the determined boundary by last two steps when CRF fails to provide accurate refinement.

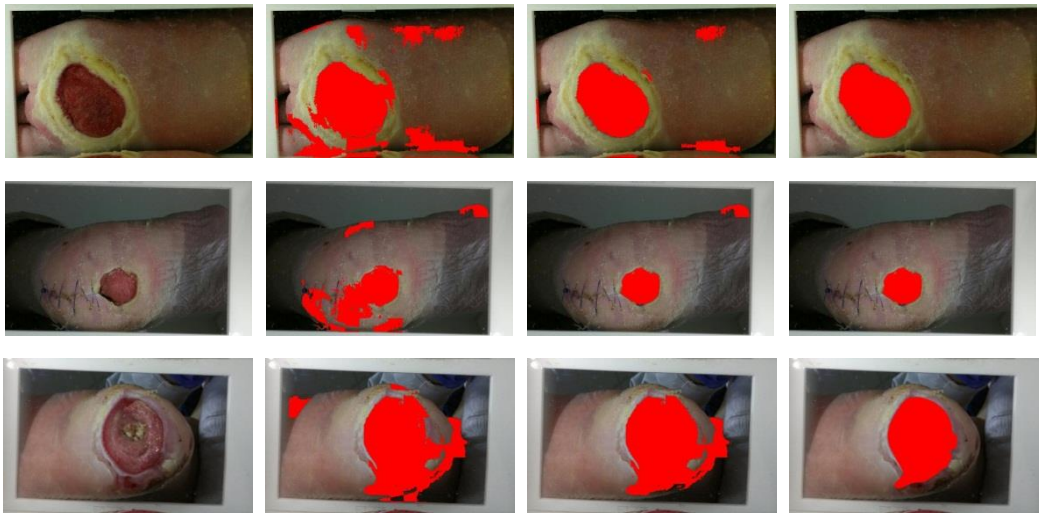


Figure 4.9 Examples of wound boundary determination results (the determined wound areas are covered with red color), column 1: the original image, column 2: the boundary determination results by applying our two stage classifier, column 3: the results after applying the CRF refinement technique, column 4: the results after the outlier removal or hole filling up.

Since the last two steps are straightforward and only involve basic image processing techniques, in following parts of this section, we will provide additional details about the first refinement step. Conditional random field (CRF) and level set are two candidate techniques, which were used as post-boundary refinement approaches in [7]. As mentioned in Section 3.1, the level set based algorithm is unstable especially when the actual wound boundary is vague. Hence, we decide to apply the CRF to recover more

precise boundaries by further reducing the misclassification occurring near the edges of wound. CRF provides a natural way to incorporate the contextual constraints by including them in a pairwise edge potential term, as shown in eq. 4.20. Actually, in Chapter 5, we will develop another novel wound boundary determination method based on more complicated CRF model, which is constructed on the entire wound image, rather than only focusing on the wound area determined by the SVM based classifier. More details about the CRF basics is found in Section 5.2. However, for a better demonstration of the boundary refinement method, we will provide a brief introduction about the pair-wise CRF model applied here.

The refinement problem can be formulated as the task of finding the most probable labels for all pixels in a given image. It includes the use of both unary pixel properties and pairwise relations between pixel labels. In a CRF, the labeling problem is transformed to an energy minimization problem. The energy is usually defined as the sum of a series of unary terms and pairwise terms, which indicate individual label preferences (unary term) and spatial coherence respectively (pairwise term).

Let $P(c \mid G; w)$ be the conditional probability of the set of class label assignments c given the adjacency graph $G(S, E)$ and scalar weight w. The energy function in [98] is shown as below.

$$-\log(P(c \mid G, w)) = \sum_{s_i \in S} \psi(c_i \mid s_i) + w \cdot \sum_{(s_i, s_j) \in E} \phi(c_i, c_j \mid s_i, s_j) \quad \textit{Equation 4.19}$$

Our unary potentials ψ are defined directly by the probability outputs provided by libSVM library [97] for each super-pixel as below.

$$\psi(c_i \mid s_i) = -\log(P(c_i \mid s_i)) \quad \textit{Equation 4.20}$$

Our pairwise edge potential φ is similar to those introduced in [77], defined as below.

$$\phi(c_i, c_j \mid s_i, s_j) = \left(\frac{L(s_i, s_j)}{1 + \left\| s_i - s_j \right\|}\right)[c_i \neq c_j] \quad \textit{Equation 4.21}$$

where [.] is the zero-one indicator function and $\left\| s_i - s_j \right\|$ is the norm of color difference between super-pixels in the LUV (or CIE Lab) color space where the perceptible color

difference can be directly calculated. $L(s_i, s_j)$ is the shared boundary length between super-pixels and acts here as a regularizing term which discourages small isolated regions.

In many CRF applications for this domain, the unary and pairwise potentials are represented by a weighted summation of many simple features, and so the parameters of the model are trained by maximizing their conditional log-likelihood. In the formulation in eq. (4.18), it simply has one weight *w* which represents the trade-off between spatial regularization and our confidence in the classification. We estimate *w* by cross validation on the training data [94]. Once our model has been trained, we wish to find the most probable labeling c^*; i.e. the labeling that maximizes the conditional probability in eq. (4.18). The optimal labeling is found by carrying out inference with multi-label graph optimization library of [77] using α-expansion. Since the CRF is defined on the super-pixel graph, inference is very efficient, taking less than half a second per image on a typical modern PC [77]. Based on our experimental results, the CRF refinement consumes 2.2 second in average on the Nexus 5 smartphone, which is still efficient enough for real time processing. Example of CRF refinement is shown in Figure 4.7, Col 3. To demonstrate the refinement effectiveness, we re-evaluate the specificity and sensitivity measures based on the CRF refined boundary determination results on the same testing images in Section IV. By comparing column 2 and 5, we can see 1.9% improvement on sensitivity and 1.8% improvement on specificity.

4.4 Conclusion

A novel system for automatic wound boundary determination on diabetic foot ulcer images has been proposed in this paper. We tracked 15 patients in the Wound Clinic at UMASS over a 2-year period and 100 well-focus, high-resolution foot ulcer images were capture, with assistance of the image capture box to benefit the patients with limited mobility. The images were first segmented into super-pixels by SLIC algorithm, which outperformed other widely used segmentation algorithms in our evaluation.

The inputs of the classifiers were color and texture descriptors of super-pixels. To train the supervised classifiers, we asked 3 experienced clinicians to delineate the wound boundary on 100 images and designed a novel way to generate the ground truth label for

each super-pixel. Experimental evaluation showed that our method largely reduced the workload for clinicians and maintained high reliability at the same time. In the first stage, *k* C-SVM based classifiers were trained by *k*-fold cross-validation strategy on the entire training dataset. In the second stage, only the incorrectly classified instances when applying the k classifiers trained in the first stage on their own testing set were used as the training set to train another C-SVM based classifier. To achieve better performance, we used different combinations of color and texture descriptors for the two training stages. Finally, to refine the determined boundary, we utilized a number of image processing techniques, including morphological operations, connected region detection and CRF based relabeling method, to either remove the outlier non-wound regions or fill the undetected wound regions.

To evaluate our two-stage binary classification system, we compared its performance with two other machine learning strategies: single-stage SVM or ANN based classifier. Moreover, the classification performance on different color and texture descriptors were also assessed based on single-stage SVM machine learning strategy. During the evaluation, we found that the sensitivity rate became the most variable performance measurement across the different machine-learning approaches analyzed in this paper. On the other hand, almost every machine learning strategy used here can achieve promising specificity rate (higher than 92%) and there was only small variance between methods (< 3%). According to our results, the mean color + color histogram + wavelet texture feature (applied in our second stage training) showed best distinguishability when dealing with the classification between wound regions and surrounding skin regions. However, the dominant color + color histogram + BoW histogram of DSIFT features (applied in our first stage training) provided the best sensitivity when classifying irrelevant background and foot skins. Besides, our two-stage SVM based classifier provided the best sensitivity rate to determine wound boundary from foot ulcer images containing wound ,foot skin and irrelevant background, which were supposed to be images easy to acquire in real world.

We implemented the entire wound classification system based on Nexus 5 android smartphone and measured the computation time for each module. We found the average consuming time for our two stage classification method is 17.5 seconds. Compared with

the method proposed in Section 3.4, our two-stage approach only increased the consuming time in a small extent (2 seconds), however, the boundary determination performance has been improved a lot (12 % improvement on sensitivity and 8% on specificity). We only implemented the off-line classification system on the smartphone, not including the classifier training process. Moreover, as stated in [76], SVM based training strategy is good at dealing with long descriptor on relatively small training set. Hence the PCA dimension reduction wasn't really helpful and we didn't use it in the actual smartphone implementation to further increase the time performance.

To obtain a more precise estimation of the appropriateness and extend the feasibility of our machine learning approach, a few tasks need to be addressed in future. 1) We need to recruit more clinicians to delineate wound boundaries and try to minimize the impact of inter- and intra-observer variability, which was reported as an unavoidable problem to undermine the classification efficiency when using manual delineation as the ground truth; 2) we need to extend the current wound image database to further enhance the classifier training. Hence, not only more type 2 diabetic patients need to be recruited, but also we considered about training our supervised wound classifier on other type of wound images (for example burning wound) and also wound images taken under variant lighting conditions. In this case, our methodology would become easy to apply in other similar scenarios; and 3) we could exploit the cloud computing techniques to update the wound classification model using an online machine learning strategy to add more flexibility to our method, rather than pre-store the trained model files on the smartphone.

Chapter 5

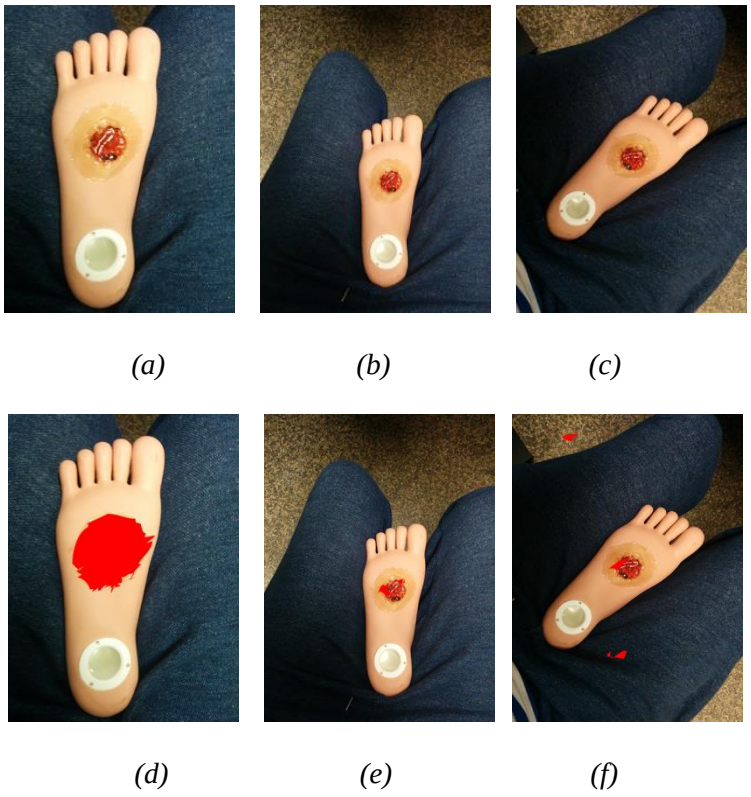
CRF Based Approach for Unconstrained Wound Boundary Determination under Various Imaging Conditions

5.1 Overview

The cascade SVM classifier based method developed and tested in Chapter 4 can only provide accurate wound boundary determination results on wound images captured using the image capture box. In other words, this approach cannot handle the wound recognition task under ordinary imaging condition, i.e. different illumination level and ranges. We tried to train the model by using image dataset with more diversity, but were still not able to achieve robust wound boundary determination results. An example of the recognition failure is shown in Figure 5.1. Images of the same Moulage wound under different illumination levels and ranges are shown in images part (a) – (c). The identified wound area from our method is marked in red in part (d) – (f). We can see clearly false recognition on the second and third images. Especially, the cascade SVM classifier method totally missed the target when the wound scale is small and the background information is complicated. Hence, a more robust method is needed if we wish to broaden the applicability of the wound recognition algorithm by relaxing the image capture constraints, i.e. being able to capture images without the use of the image capture box .

In modern computer vision research, two significant findings have been made [98]–[101]: 1) it is better to model the object classification problem based on optimization principles, either explicitly or implicitly, and 2) contextual constraints (the similarity and mutual relationship between the current pixel and its neighborhood pixels) are important for the interpreting of visual information in images. Naturally, the following two questions have

been presented: 1) how to define an optimality criterion under contextual constraints, and 2) how to find the optimal solution. Note that the object classification problem can also be referred as image multi-labeling problem (we assume that there are more than two object categories in images. This assumption should be valid for wound images used in our work) and the two terms "object classification" and "image multi-labeling" will be treated completely equivalent.



(a) *(b)* *(c)*

(d) *(e)* *(f)*

Figure 5.1 An example of recognition failure by the SVM based approach on Moulage wound images of different scales (ranges) and illumination levels, (a) – (c): original wound images, (d) – (f): wound boundary determination results

One response to these two questions is offered by a common approach called Markov Random Field (MRF), which is a branch of probability theory. MRF model is characterized by following two properties [102]: 1) it enables a formulation for the mathematical description of contextual constraints and the probability distribution of mutually related features; and 2) it provides an approach for defining optimality criteria based on Maximum A Posteriori (MAP) concept. This MAP-MRF framework provides us with the ability to develop algorithms for object classification using relational

principles instead of ad hoc heuristics [102], [103]. Technically, MRF model formulates in a probabilistic generative framework modeling the joint probability of the image pixel values and their corresponding labels. Especially important is the fact that a theoretical result about the equivalence between MRF and Gibbs distribution, a probability distribution of particles in a system over a set of possible states as in eq. (5.1), provides a mathematically tractable means of calculating the joint probability of an MRF [102].

However, in spite of its recent popularity, MRF based approaches suffer from the following two limitations with respect to object classification problem [101]. First, due to the complexity of inference and parameter estimation, only local relationships between neighboring nodes are incorporated into the model. This property enables the model to maintain local consistency of the assigned labels, based on very local regularities, but makes it highly inefficient at representing multiple interacting features or long-range correlations of the observations. However, as the conditional probability of a labeling will likely depend on structure at different levels of granularity in the image. To accomplish this, we need to formulate a model that can capture both local and global relationships. Secondly, due to its generative nature [2], two steps are needed to estimate the conditional probability of a labeling given the observed data: 1) estimation of the likelihood and 2) estimation of the distribution of the observed image data [102]. Actually, we are only interested in the distribution of labels given image data, as it is a waste of resource to estimate the entire generative image model.

A very different non-generative (discriminative) approach is to directly model the conditional probability of labels given a set of images: fewer labeled images will be required, and the resources will be directly relevant to the task of inferring labels. This is the key idea underlying the conditional random field (CRF) [104]. Originally proposed for segmenting and labeling 1D text sequences, CRFs directly model the posterior distribution as a Gibbs distribution [101]. This conditional probability model can depend on arbitrary non-independent characteristics of the observation, unlike a generative image model which is forced to account for dependencies in the image and therefore requires

[2] The major difference between the generative and discriminative models lies in the way to calculate the conditional probability of a label given the observed data ($p(y|x)$). The generative model specifies the joint probability of the observed data and the label and then calculate $p(y|x)$ using Bayes rules. In contrast, the discriminative model calculates the posterior $p(y|x)$ directly.

strict independence assumptions to make inference tractable. Besides, the CRF model does not model the distribution for observations, which can be regarded as constant values when performing classification. For these reasons, there has been increasing interests in recent years in modeling image labeling problems using CRF model.

In this chapter, we will focus on applying different CRF based models to our wound recognition task and then evaluate the performance. The material will be organized as follows. First basic knowledge about CRF model will be introduced. Second, we will describe the hierarchical CRF models and apply them to our wound recognition task. Then we will discuss how to utilize the efficient model inference approach by modifying the original CRF model into a densely connected model. Finally, the wound recognition performance will be evaluated on both images of Moulage wound simulation and real wounds from recruited subjects.

5.2 Conditional Random Field Basics in Context of Image Labeling

5.2.1 Basic Pairwise Conditional Random Field Model

Consider an ordered set of variables $X = [X_1, X_2, ...X_n]$, where each variable X_i will be annotated by a label from a set L corresponding to the object classes. We write $y \in L^n$ for a labeling of *X*, and use y_i to refer to the labeling of the variable X_i. The random variables X and y are jointly distributed, but in the discriminative framework we construct a probabilistic model $P(y \mid X)$ to be estimated from *M* paired training instances $\{X^{(i)}, y^{(i)}\}_{i=1}^{M}$ and don't need to model the marginal $p(X)$.

The neighborhood system N is defined by sets N_i, $\forall i \in V$, where N_i denotes the set of all neighbors of the variable X_i. A clique *c* is a set of random variables X_c which are conditionally dependent on each other [102]. Any possible assignment of labels to the random variables will be called a labeling (denoted by *y*), which take its possible values from L^n. The labeling on the clique c is referred as y_c.We use $V = \{1, 2, ...n\}$ to refer to the set of valid vertices (or indexes) of X. As mentioned before, the posterior

distribution $P(y \mid X)$ over the labeling of the CRF is a Gibbs distribution [102] and can be written as below.

$$P(y \mid X) = \frac{1}{Z} \exp(-\sum_{c \in C} \varphi_c(y_c)) \qquad \textit{Equation 5.1}$$

where Z is a normalizing constant called the partition function, and C is the set of all cliques. The term $\varphi_c(y_c)$ is known as the potential function of the clique $c \subset V$ where $y_c = \{y_i : i \in c\}$. The corresponding Gibbs energy is given by the formula below [98].

$$E(y) = -\log(P(y \mid X)) - \log Z = \sum_{c \in C} \varphi_c(y_c) \qquad \textit{Equation 5.2}$$

Finding the most probable labeling is equivalent to solving the maximum a posterior (MAP) problem. This optimal labeling x^* of the CRF is defined as below [98].

$$y^* = \arg\max_{y \in L} P(y \mid X) = \arg\min_{y \in L} E(y) \qquad \textit{Equation 5.3}$$

According to [98], [100], [101], labeling problems in computer vision area are typically formulated as a pairwise conditional random field whose energy can be written as the sum of unary and pairwise potentials [99].

$$E(y) = \sum_{i \in V} \varphi_i(y_i, \theta_u) + \sum_{i \in V, j \in N_i} \varphi_{ij}^P(y_i, y_j, \theta_p) \qquad \textit{Equation 5.4}$$

where N_i is the set of neighbors of vertex i. The unary potential $\varphi_i(y_i, \theta_u)$ is computed independently for each pixel by a classifier that produces a distribution over the label assignment y_i given image features. The pairwise potential $\varphi_{ij}^P(y_i, y_j, \theta_p)$ is computed between each pair of two adjacent pixels in the image domain to penalize the adjacent pixels taking different labels. θ_u and θ_p are sets of parameters for the unary potential and pairwise potential, respectively. The details of the parameter estimation, also referred as the model learning, will be discussed in Section 5.2.2. The actual formulations to calculate the unary and pairwise potentials are task dependent. This flexibility allows us to incorporate different factors (color, edge, texture or spatial position) into the CRF energy formulations [98], [99].

5.2.2 Parameter Learning

Parameter estimation is the problem of finding a set of parameters θ so that the resulting conditional distribution $P(y \mid X)$ best fits a set of training examples $D = \{x^{(i)}, y^{(i)}\}_{i=1}^{M}$ which were labeled manually as the ground truth. Intuitively, the goal, to be accomplish during parameter estimation, can be described as follows: if we look at any of the training data instance $x^{(i)}$, the model distribution over outputs $p(y \mid x_i, \theta)$ should match the ground truth label $y^{(i)}$ as closely as possible.

Ideally, the parameters in this model would be determined using maximum a-posteriori (MAP) method [99], [105]. In this case, we are supposed to maximize the conditional likelihood of the labels given the training data. The exact formula is shown as below [99].

$$\theta^{*} = \arg\max_{\theta}(L(\theta))$$
$$L(\theta) = \sum_{i} \log P(y_i \mid x_i, \theta) + \log P(\theta) \qquad \textit{Equation 5.5}$$

According to [99], the purpose of the prior term $\log P(\theta)$ is to regularize the learning process and avoid overfitting. According to optimization theory, the maximization of $L(\theta)$ can be accomplished by applying gradient descent algorithm [99]. However, this algorithm requires the evaluation of marginal probability over the class labels at each pixel for all training images, and the exact computation of these marginal is intractable [106]. The practical evaluation is carried out using probabilistic inference [106], and the label marginal distribution can be approximated by the most probable labeling given by the probabilistic inference method [99]. These modes can be found using standard graph cut based inference approach, such as alpha-expansion [48] (more details about this algorithm can be found in the next section about model inference). Alternatively, belief propagation [3] [107] or variational methods [107] may provide more accurate approximation, but make the inference less efficient [99].

According to [99], the number of parameters is large if multiple unary potential terms are used (each potential term is supposed to have more than one parameter), the modal

[3] Belief propagation is one inference algorithm on graphical model based on message passing scheme. It computes the marginal distribution for each unobserved node, conditional on any observed nodes. This algorithm has been proved to be quite effective for optimal label inference on general graph.

approximation is not sufficient. Hence, dealing with the difficulties for maximizing $L(\theta)$ directly, a more practical piecewise training solution has been proposed in [99], [108]. In the piecewise training method, the parameters in each potential term in CRF formula is learned independently by maximizing the conditional likelihood, as if it was the only term in the model [99]. However, if the terms in the CRF model are correlated, performing piecewise training brings about the over-counting problem during the inference using the simply combined model, i.e. the learned parameter value is larger than the desired optimal one. Hence, these trained terms need to be combined with weighting functions to overcome the over-counting problems [99].

According to [99], there is no analytical method to accurately evaluate the degree of over-counting caused by correlation between different terms. Therefore, a scalar weight is introduced to each potential term, and these weights can be optimized on the validation dataset.

5.2.3 Model Inference

Given the CRF formula and the learned parameters, the optimal labeling y^* which maximizes the conditional probability in eq. (5.1) can be found using appropriate inference method. There are two most widely used inference approaches: 1) graph-cut based moving making algorithm [48], and 2) traditional message passing based algorithm [107]. A brief introduction about these two types of approaches will be given in the rest of this section.

The two most typical graph-cut based move making algorithms are alpha-expansion and alpha-beta-swap algorithm, both breaking down the original multi-labeling problem into a sequence of binary labeling problem. These move making algorithms begin with an arbitrary initial solution of the labeling problem and proceed by making a number of sequential changes, where each of these changes is supposed to result a solution of the same or lower energy calculated by eq. (5.4) [109]. According to [48], the "near global optimal" solution found by alpha-expansion algorithm is ensured to be within a known factor of the actual optimal solution. For a pair of labels α and β, a swap move takes some subset of the pixels currently given the label α and reassign the label β to them. The swap move algorithms keep updating the local minimum until there is no swap move can result

a lower energy. Similarly, the expansion move algorithms finds a local minimum such that no expansion move, for any label α, yields a labeling with lower energy.

According to [48], the swap move algorithms can be applied to any CRF models with a semi-metric pairwise term. But the expansion move algorithms are only applicable to CRFs with metric pairwise term. Definition for metric pairwise term φ_{ij}^{P} in the CRF model requires that the term should satisfy three constraints shown in eq. (5.6) [48]. If only the first two equations are satisfied in eq. (5.6), we called this term as semi-metric.

$$\begin{aligned} &\varphi_{ij}^{P}(y_i, y_j) = 0 \Leftrightarrow y_i = y_j \\ &\varphi_{ij}^{P}(y_i, y_j) = \varphi_{ji}^{P}(y_j, y_i) \\ &\varphi_{ij}^{P}(y_i, y_j) \le \varphi_{ik}^{P}(y_i, y_k) + \varphi_{kj}^{P}(y_k, y_j) \end{aligned} \qquad \textit{Equation 5.6}$$

If the energy does not meet these constraints, graph cut based swap or expansion algorithms can still be applied by "truncating" the violating terms [48]. However, in this case we are no longer guaranteed to find the optimal labeling with respect to corresponding moves. According to [48], the performance of this version seems to work well when only relatively few terms need to be truncated.

The message passing based algorithm can solve a wide variety of inference problems. The algorithms operate on factor graphs that visually represent the problems [107]. The factor graph is a bipartite graph that expresses how a global function of several variables factors into a product of local functions. The most well-known message passing algorithm is called belief propagation, also known as sum-product message passing [107]. It calculates the marginal distribution for each unobserved node, conditional on any observed nodes. The details about how to apply the message passing based algorithm in the CRF inference problem will be shown in Section 5.3.4.

5.3 Tree Conditional Random Field Based Models for Object Classification

As mentioned before, the conditional random field (CRF) based models have been widely applied to object classification (image labeling) tasks due to its generative nature and flexibility to incorporate various features in a single unified formulation. A lot of models

have been proposed in recent years [100], [101], [109]. The major difference between models lies with the potential granularity (pixel-wise or super-pixel wise) and the features used to generate the potential term. Consequently, the inference algorithm for each model varies accordingly. In this section, we will introduce three CRF based models, each of which has been claimed to provide strong performance in object classification on natural scene images.

5.3.1 TextonBoost

Practically, the unary potential term is a vector for each pixel (or super-pixel), and each element in this vector represents the corresponding probability of the current pixel taking a certain label. In other words, it can be viewed as a local statistical classifier, which produces distribution $P(y_i \mid x_i, \theta_u)$ over a label y_i given the local pixel or region feature x_i, and θ_u is the classifier parameters that need to be estimated during the learning process.

The multi-layer perceptron (also referred as ANN), as introduced in Section 4.3, might be an intuitive choice for the local classifier generation. However, according to [101], the performance of this classifier is limited by class overlap and even slight image noise. Hence, we need to search for more robust approaches. In most of recent CRF models for image labeling, the unary potentials for pixel-wise features are derived from TextonBoost, which estimates the probability distribution of labels on current pixel by boosting weak classifiers based on a set of shape filter responses [99]. Before we actually discuss each CRF model, we will introduce the TextonBoost concept at first. A general TextonBoost process is shown in Figure 5.2. The Each individual functional block in Figure 5.2 will be discussed in the following text.

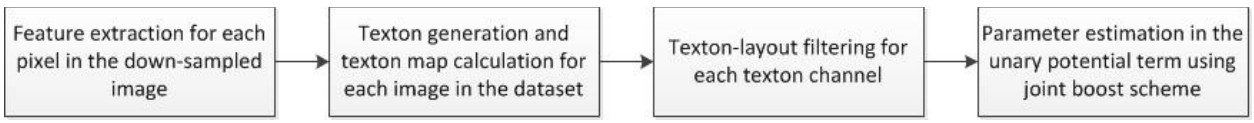


Figure 5.2 TextonBoost working flow

- *Textons*

This section describe the first two functional blocks in Figure 5.2. Efficiency demands a compact representation for the range of different appearance of an object. Textons [110]

have been used since they have been proven effective in categorizing materials as well as generic object classes [99]. The term texton was utilized for describing human textural perception, and is somewhat analogous to phonemes used in speech recognition [99], [110].

An example of a standard textonization process has been shown in Figure 5.3. For the first step, the training images are convolved with a filter bank at different scales. There are actually quite a number of different options for the filter bank, where the only requirement is that the filter bank should be sufficiently representative. In this work, we apply the same filter-bank as in [111], which consists of Gaussians at three different scales ($\sigma = 1, 2, 4$), Laplacian of Gaussians (LoG) [112] at four different scales ($\sigma = 1, 2, 4, 8$) and 4 first order derivatives of Gaussians (DoG). The Gaussians are applied to all three color channels, and then produce 9 filter responses. The LoGs are applied only to the luminance channel in CIE Lab space and provide 4 filter responses. Finally, DoGs, with two different scales and two different directions, are also only applied to the luminance channel and give 4 filter responses. Hence, 17 dimensional filter responses are produced in total for each pixel.

The original RGB color space needs to be converted to CIE Lab space for perceptual uniformity. This filter-bank was determined to have full rank in a singular-value decomposition and therefore contains no redundant element [99]. The 17 dimensional responses for all training pixels are then normalized (to give zero mean and unit covariance), and an unsupervised clustering is performed to generate the texton dictionary. As recommended by previous works [99], we apply the Euclidean –distance K-means clustering algorithm. The one most obvious shortcoming of the original K-means algorithm is its computational expense. Fortunately, its time performance can be greatly improved by employing the triangle inequality techniques for acceleration [113]. Finally the texton maps are generated by assigning each pixel in each image to the nearest cluster center. We will denote the texton map as *T* where pixel *i* has value $T_i \in \{1, 2, ...\mathrm{K}\}$ and where *K* represents the cluster number set in the K-mean algorithm. In practice, we set the cluster number as 400 empirically. According to [99], to improve the

time performance, we can apply the *kd*-tree [4] [114] to perform the nearest neighbor search. It is claimed in [99] that "*textonization using kd-tree with leaf node bins containing 30 cluster centers gave a speed up of about 5 times over simple linear search* " .

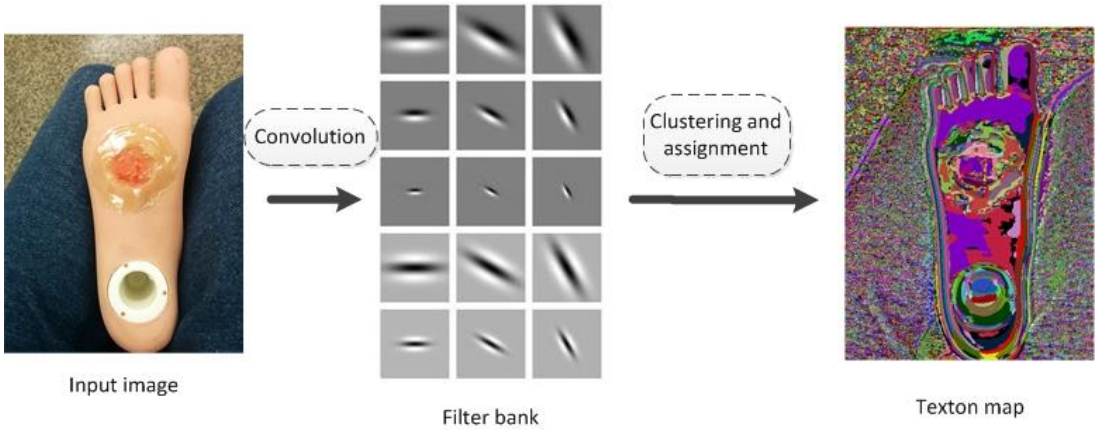


Figure 5.3 The process of image textonization

In practice, we can use other dense features, such as the location feature [99] and dense SIFT feature [77] which has been introduced in Section 4.2, instead of the filter bank output introduced above. Actually, we extract multiple features at the same time for each pixel and generate an independent texton map based on each feature. More details about our implementation will be presented in Section 5.4.

- *Texture-Layout Filters*

Each texture-layout filter is a pair (r,t) of an image region r (according to [99], it is better to use the rectangular region) and a texton t. Region r is defined in coordinates relative to the pixel i being classified [99]. For simplicity, a set R of candidate rectangles are chosen at random, such that their top-left and bottom right corner lie within a fixed bounding box covering about half the image area. According to [99], the bounding box was ± 100 pixels in x and y direction. This enables the model to involve long-range contextual information, in addition to the original CRF model which only contains pixel-wise connections between adjacent pixels in the second order clique [99]. The feature

[4] kd tree is a space-partitioning data structure to store k-dimensional data elements (each node contains one data point). The tree structure is organized in a way such that each non-leaf node can be viewed as a splitting hyperplane which separate the entire space into two half-spaces. A key advantage of the kd tree structure is that it can greatly accelerate the nearest neighbor search in the space by using binary search idea.

response at location i is the proportion of pixels having the texton index t in the offset region $r+i$. The exact formula is shown below.

$$v_{[r,t]}(i) = \frac{1}{area(r)} \sum_{j \in (r+i)} [T_j = t] \qquad \textit{Equation 5.7}$$

Outside the image boundary there is zero contribution to the feature response. According to [99], the filter responses can be efficiently computed over the entire image by applying integral images which is used for efficiently calculating the sum of pixel values in a rectangular region. The texton map is separated into K different channels (one for each texton). Afterwards, a separate integral image is calculated for each channel. This can be regarded as an integral histogram, where each bin represents a single texton. The integral image can be used to compute the texture-layout filter responses in constant time [99]: if $\hat{T}^{(t)}$ is the integral image of T for texton channel t, then we can calculate the feature response as given below [99].

$$v_{[r,t]}(i) = \hat{T}^{(t)}_{r_{br}} - \hat{T}^{(t)}_{r_{bl}} - \hat{T}^{(t)}_{r_{tr}} + \hat{T}^{(t)}_{r_{tl}} \qquad \textit{Equation 5.8}$$

where r_{br}, r_{bl}, r_{tr} and r_{tl} denote the bottom right, bottom left, top right and top left corners of rectangle r. Texture-layout filters which is in form of pairs of rectangular regions and textons [48]. Similar features were proposed in [13], although textons were not used, and responses were not aggregated over a spatial region. Region shapes other than rectangles were also investigated in [99]. In particular rectangles rotated by 45 degrees were evaluated, and pairs of rectangles with the texton responses either added or subtracted. However, despite considerable extra computational expense (since these new combinations of features must be tested at each round of boosting, as demonstrated in following sections), no greatly improved performance was produced by these complicated features. It was stated that this should be attributed to overfitting. Besides, [99] tried to use the bag of visual words in [88] to model contextual appearance, and the final classification performance were worse than the K-means algorithms. The analysis in [99] argued that this is due to the invariance to spatial layout of the textons of the clustering algorithm used in [36].

- *Unary Potential trained by using Joint Boost Scheme*

In [99], the unary potential term based on texture-layout filter output was trained using the adapted version of the Joint Boost algorithm [115], which combined a number of "weak classifiers" (iteratively selected discriminative texture-layout filters) into a strong classifier $P(\mathrm{c} \mid \mathrm{x}, \mathrm{i})$. Each weak classifier would be shared by a number of classes (class set *C*). In this case, each weak classifier is capable of dealing with the multi-classification task between the classes in *C*. According to [99], this also gives us the possibility of more efficient classification and better generalization.

In simple binary classification case, boosting provides a simple way to sequentially fit additive models of the form below.

$$H(\mathrm{v}) = \sum_{m=1}^{M} h_m(\mathrm{v}) \qquad \textit{Equation 5.9}$$

where *v* is the input feature vector and *M* is the number of boosting rounds. $H(\mathrm{v})$ is called the strong classifier and $h_m(\mathrm{v})$ is called the weak classifier. In the multi-class case, the cost function was modified as given in eq. (5.10) (the same as in the Adaboost[5] [116]).

$$H(\mathrm{v}, \mathrm{c}) = \sum_{m=1}^{M} h_m(\mathrm{v}, \mathrm{c}) \qquad \textit{Equation 5.10}$$

where $H(\mathrm{v}, \mathrm{c})$ can be viewed as a probability distribution over *c* and defined as below.

$$H(\mathrm{v}, \mathrm{c}) = \log \frac{P(\mathrm{z}^c = 1 \mid \mathrm{v})}{P(\mathrm{z}^c = -1 \mid \mathrm{v})} \qquad \textit{Equation 5.11}$$

where z^c is the membership label (± 1) for class *c*.

Boosting then optimizes the following cost function shown as below by updating one term of the additive model at one iteration [115].

[5] AdaBoost (Adaptive Boosting), is a machine learning algorithm whose final classification output is a weighted sum of output from a number of weak classifier. Different from traditional machine learning algorithms (SVMs and neural networks), the AdaBoost training process gradually favors those instances misclassified by previous weak classifier in subsequent boosting rounds. Besides, only those features capable of improving the classification accuracy of the model will be selected. Hence, the "curse of dimensionality" (lack of capability to deal with instances represented by high dimensional features) can be effectively avoided.

$$J = \sum_{c=1}^{C} E(\mathrm{e}^{-z^c H(\mathrm{v},\mathrm{c})}) \qquad \textit{Equation 5.12}$$

where C is total number of classes. According to [115], the term $-z^c H(\mathrm{v},\mathrm{c})$ is called the "margin" and related to the generalization error [115]. The cost function was viewed as an approximation to the likelihood of the training data under a logistic noise model [115].

In multi-class case, the joint boost algorithm proposed in [115] differs from the traditional Adaboost scheme in terms of the structure of the weak classifiers h_m. The key contribution of this algorithm is that a subset of classes $S(\mathrm{m})$ will be chosen at each round *m*, and this subset was supposed to share the feature that their classification error will be reduced.

We now describe in more details how joint boost scheme works. At each iteration, this scheme will solve a weighted least squares problem as given below.

$$J_{wse} = \sum_{c=1}^{C} \sum_{i=1}^{N} w_i^c (\mathrm{z}_i^c - \mathrm{h}_m(\mathrm{v}_i, \mathrm{c}))^2 \qquad \textit{Equation 5.13}$$

where $w_i^c = e^{-z_i^c H(\mathrm{v}_i,\mathrm{c})}$ are the weights for training example i and the classifier for class c. Each training example i has C weights, one for each binary problem. Again, z^c is the membership label (± 1) for class *c*. For each binary classification problem, we can consider as negative examples all the other classes and the background, or just the background class (in such a case, we can set the weights to $w_i^c > 0$ for samples in the class *c* ($z_i^c = 1$), and we set $w_i^c = 0$ for samples in any other classes).

Then, for classes in the chosen subset, $c \in S(\mathrm{n})$, it is common to define the weak classifier to be simple functions of the form as below.

$$h_m(\mathrm{v},\mathrm{c}) = \mathrm{a}_S \, \delta(\mathrm{v}^f > \theta) + \mathrm{b}_S \, \delta(\mathrm{v}^f \le \theta) \qquad \textit{Equation 5.14}$$

where v^f denotes the *f'th* component of the feature vector *v*, θ is a threshold, $\delta(.)$ is the indicator function, and a_S, b_S are regression parameters, which will be defined in eq. (5.15).

For classes not in the chosen subset, we define the weak classifier to be a class-dependent constant k^c . The definition of the entire shared stump is as below. Note that this definition includes eq. (5.14).

$$h_m^n(\mathrm{v},\mathrm{c}) = \begin{cases} a_S & \text{if } v^f > \theta \text{ and } c \in S(\mathrm{n}) \\ b_S & \text{if } v^f \le \theta \text{ and } c \in S(\mathrm{n}) \\ k_S^c & \text{if } c \notin S(\mathrm{n}) \end{cases} \qquad \textit{Equation 5.15}$$

The class-specific constant k_S^c is used to prevent a class being chosen for sharing just to balance the numbers of negative and positive training examples. According to [115], the constant k_S^c encoded a prior bias for each class, without using features from other classes and changes how features are shared especially in the first round of boosting. Hence, to ensure that the shared features indeed improve the discriminative ability of the entire model, we only add a class into the shared subset when there is a decrease of the classification error which is larger than using a constant as weak classifier. At iteration *n*, the algorithm will select the best stump and a subset of classes. For the class subset $S(\mathrm{n})$ for iteration *n*, we try to find the parameters to minimize eq. (5.13). According to [115], minimizing this equation provides us the following results.

$$a_S(\mathrm{f},\theta) = \frac{\sum_{c\in S(\mathrm{n})}\sum_i w_i^c z_i^c \delta(\mathrm{v}_i^f > \theta)}{\sum_{c\in S(\mathrm{n})}\sum_i w_i^c \delta(\mathrm{v}_i^f > \theta)} \qquad \textit{Equation 5.16}$$

$$\mathrm{b}_S(\mathrm{f},\theta) = \frac{\sum_{c\in S(\mathrm{n})}\sum_i w_i^c z_i^c \delta(\mathrm{v}_i^f \le \theta)}{\sum_{c\in S(\mathrm{n})}\sum_i w_i^c \delta(\mathrm{v}_i^f \le \theta)} \qquad \textit{Equation 5.17}$$

$$k^c = \frac{\sum_i w_i^c z_i^c}{\sum w_i^c} \quad c \notin S(\mathrm{n}) \qquad \textit{Equation 5.18}$$

It is easy to see that each weak classifier has 4 parameters $(\mathrm{a}_S, \mathrm{b}_S, \mathrm{f}, \theta)$ for the positive class, $C - |S(\mathrm{n})|$ parameters for the negative class and 1 parameter to decide which subset $S(\mathrm{n})$ should be chosen.

The entire boosting training process with the shared regression stumps are summarized as below.

STEP 1: initialize all the weights $w_i^c = 1$ and set $H(\mathrm{v}_i, \mathrm{c}) = 0$ for all $i = 1, 2, .. \mathrm{N}$ and $c = 1, 2, ..C$.

STEP2: Repeat for $m = 1, 2...M$

- Repeat for $n = 1, 2...2^C - 1$
    a) Fit shared stump in eq.(14) by specifying the parameters by eq. (5.16)-(5.18)
    b) Evaluate error as in eq.(5.13)
- Find the optimal subset: $n^* = \arg\min_n J_{wse}(\mathrm{n})$
- Update the class estimates as below.

$$H(\mathrm{v}_i, \mathrm{c}) := H(\mathrm{v}_i, \mathrm{c}) + h_m^{n^*}(\mathrm{v}_i, \mathrm{c}) \qquad \textit{Equation 5.19}$$

- Update the weights as below.

$$w_i^c := w_i^c e^{-z_i^c h_m^{n^*}(\mathrm{v}_i, \mathrm{c})} \qquad \textit{Equation 5.20}$$

Finally, we will discuss about the weak classifier optimization. We must find the optimal regression stump, a slow computation, since it needs to span all features and all *N* thresholds (*N* is the number of training examples). According to [99], several optimizations are available to accelerate the search for the optimal weak classifier h_i^m. Since the set of all possible sharing sets is exponentially large, it is suggested that the greedy approximation introduced in [115] should be used. This approximation method can ensure that the search can be accomplished in quadratic time complexity (meaning that the search time varies quadratically with the number of data points).

To accelerate the minimization over features, a random feature selection procedure can be applied [115]. According to [115], optimization over $\theta \in \Theta$ for a discrete set Θ will be more efficient if an appropriate histogram representation of weighted feature responses is applied. As we mentioned earlier, the feature that we used for boosting is the texture layout features $v_{[\mathrm{r,t}]}(\mathrm{i})$. Hence, by treating Θ as an ordered set, histograms of values $v_{[\mathrm{r,t}]}(\mathrm{i})$, which are weighted appropriately, are built over bins corresponding to the

thresholds in Θ; these histogram are accumulated to give the threshold sums necessary for the direct calculation of a_S and b_S in eq. (5.16) and (5.17).

Exhaustive search over all features *(r,t)* at each round of boosting is practically impossible. To improve the efficiency, it is suggested in [99], [115] that the search algorithm should only run over a random selected subset of all the possible features. In this case, it is highly likely that all features are utilized at least once to ensure that good features will be selected eventually, given an appropriate number of iteration rounds (usually set as 1000 - 5000).

Furthermore, the considerable memory and processing requirements of this procedure make training using all pixels from all sample images impractical. Hence we take training example *i* only at pixels lying on a down-sampled grid. However, the texture-layout filter responses are still calculated at full resolution to allow for per-pixel accurate classification at evaluation time; we simply train from fewer examples. One possible consequence of this down-sampling is that a small degree shift-invariance is learned [115], which might lead to inaccuracy at the target object boundary. Fortunately, this inaccuracy can be reduced by the pairwise potential term in the CRF model.

5.3.2 CRF Model 1

The first CRF model we will apply to our wound recognition task was proposed in [99]. The CRF energy formulation is shown as below.

$$E(y) = \sum_i \overbrace{(\varphi_i(y_i, x_i, \theta_\varphi) + \pi(y_i, x_i, \theta_\pi) + \lambda(y_i, x_i, \theta_\lambda))}^{unary\ potential\ terms} + \sum_{i, j \in N_i} \overbrace{\phi(y_i, y_j, g_{ij}(x_i, x_j), \theta_\phi)}^{pairwise\ potential} - \log Z(x, \theta)$$

Equation 5.21

$\varphi_i(y_i, x_i, \theta_\varphi) = p(y_i \mid x_i, \theta_\varphi)$ is the unary potential term derived from the texture-layout boosted classifier trained by TextonBoost method. According to [99], this term incorporates the texture, layout, and textural context information of the object classes. It was proven in [99] that this unary term is the most powerful term in this CRF model.

$\pi(y_i, x_i, \theta_\pi)$ is the unary color potential term which derived from the Gaussian Mixture Models (GMMs) in CIE Lab color space, and the mixture coefficients depend on the

class label. According to [99], the conditional distribution of the color *x* given a pixel depending on a class label *c* is shown in the formula below.

$$p(x \mid y) = \sum_k p(x \mid k)\, p(k \mid y)$$
$$p(x \mid k) = N(x \mid \mu_k, \Sigma_k) \qquad \textit{Equation 5.22}$$

where $p(x \mid k)$ is the component Gaussian density for each cluster. Each component density is a multi-dimensional Gaussian function, where μ_k and Σ_k are the mean and covariance matrix for cluster center *k*. $p(k \mid y)$ can be viewed as the mixture weight (coefficient) for cluster center *k*. The color potential was defined in [99] as below.

$$\pi(y_i, x_i, \theta_\pi) = \log \sum_k \theta_\pi(\mathrm{y}_i, k)\, p(k \mid x_i) \qquad \textit{Equation 5.23}$$

Comparing eq. (5.22) with eq. (5.23), we can find that the parameter term $\theta_\pi(\mathrm{y}_i, k)$ represent the distribution term $p(k \mid y)$ in eq. (5.22) for i^{th} label in the label set. The term $p(k \mid x_i) \propto p(x_i \mid k)$ based on Bayesian rule given the known prior probability of the cluster center $p(k)$. Now the only task left for us is to learn the parameter. First, the color clusters are learned in an unsupervised manner using K-means. The iterative algorithm, called Expectation and Maximization (EM), then alternates between inferring the optimal labeling (expectation step) and computing the parameters for potential term (maximization step) [99]. The details about the EM algorithm can be found in [117].

The location potential term used in [99] is shown as below.

$$\theta_\lambda(c, \hat{i}) = \left(\frac{N_{c,\hat{i}} + \alpha_\lambda}{N_{\hat{i}} + \alpha_\lambda}\right)^{w^\lambda} \qquad \textit{Equation 5.24}$$

where $N_{c,\hat{i}}$ is the number of pixels of class *c* at normalized location $\hat{i}$ for the entire training set. $N_{\hat{i}}$ is the total number of pixels at location $\hat{i}$. α_λ and ω_λ are model constant parameters which are specified as $\alpha_\lambda = 1$ and $\omega_\lambda = 2$ empirically [99].

The pairwise edge potential has the form of a contrast sensitive Potts model which will be presented in details in Section 5.3.3 and where the parameters in this model are estimated empirically.

5.3.3 CRF Model 2

Basic CRF models consist of unary potentials, defined on individual pixels, and pair-wise potential terms defined on pairs of adjacent pixels. By incorporating smoothness term in the CRF potentials, similar pixels are encouraged to have the same label, and the contextual relationship between different object classes can be modeled. According to [109], the nature of the adjacency basic CRF structure results in the inability to include long range connections within image. Consequently, the inaccurate classification is likely to happen at the object boundary due to excessive smoothness.

To solve this problem, higher order potentials, defined on super-pixels or between pair of super-pixels, were incorporated into the basic CRF models to better describe the hierarchical connectivity [109]. This method gives us a integration of the "top-down" and "bottom-up" approaches that are common to many problems in computer vision [109]. To achieve this unification, a smart model, called the associative hierarchical random field (AHRF) was proposed in [109]. More importantly, it is shown that this model can be solved efficiently using graph-cut based move making algorithms mentioned earlier. It was also proved that a new model generated by summing up two AHRFs is also an AHRF which can be solved effectively. It enables different potentials based on different features to be incorporated within the CRF model and the model inference is still practical [109]. In this section, we will introduce the AHRF model in details.

- *Framework of Associate Hierarchical Conditional Random Field*

The pairwise conditional CRF formulation in eq. (5.4) was improved in [109] by incorporating higher-order potentials defined over super-pixels. This extension is valid since pixels in the same super-pixel are supposed to be assigned to the same label with high probability. The energy of the higher order random field proposed in [109] is of the form as shown below.

$$E(x) = \sum_{i \in V} \varphi_i(x_i) + \sum_{i \in V, j \in N_i} \varphi_{ij}^{P}(x_i, x_j) + \sum_{c \in S} \varphi_c^{h}(x_c) \qquad \textit{Equation 5.25}$$

where S is a set of cliques (or super-pixels), given by one or more super-pixel segmentation algorithms [37], and $\varphi_c^h(x_c)$ are higher-order potentials defined over the

cliques. The higher order potentials can be described as robust P^N model as given below [109].

$$\varphi_c^h(x_c) = \min_{l \in L}(\gamma_c^{\max}, \gamma_c^l + \sum_{i \in c} w_i k_c^l \Delta(x_i \neq l)) \qquad \textit{Equation 5.26}$$

where w_i is the weight of the variable x_i, and γ satisfy $\gamma_c^l \leq \gamma_c^{\max}, \forall l \in L$. The potential has a cost of γ_c^l if all pixels in the segment are assigned with label *l*. The pixels not being assigned with the same label are penalized with a cost expressed as $w_i k_c^l$, and the maximum cost of the potential is truncated to $\gamma_c^{\max}$. This framework supports the integration of higher order potential based on super-pixels in multiple scales of the image grid.

It was proven in [109] that the higher-order P^N of eq. (5.25) is equivalent to " *the cost of minimal labeling of a set of pairwise potentials defined over the same clique variables* x_c *and a single auxiliary variable* $x_c^{(1)}$ *that takes the values from an extended label set* $L^E = L \cup \{\mathrm{L}_F\}$". L_F is the free label meaning that there is no dominant label in this clique. Then this segment is said to be unassigned. Finally, we can formulate the framework to incorporate pairwise dependencies between auxiliary variables as shown in eq. 5.27 [109].

$$E(x) = \sum_{i \in V} \varphi_i(x_i) + \sum_{i \in V, j \in N_i} \varphi_{ij}^P(x_i, x_j) + \min_{x^{(1)}}(\sum_{c \in S} \varphi_c(\mathrm{x}_c, \mathrm{x}_c^{(1)}) + \sum_{c,d \in S} \varphi_{cd}^P(\mathrm{x}_c^{(1)}, \mathrm{x}_d^{(1)})) \quad \textit{Equation 5.27}$$

These pairwise terms defined over higher-order clique grid forces consistency between adjacent cliques. Then, the model in eq. (5.27) can be generalized to a hierarchical framework in which the relationship between layers takes the form as below [109].

$$\varphi_c(\mathrm{x}_c, \mathrm{x}_c^{(1)}) = \phi_c(\mathrm{x}_c^{(1)}) + \sum_{i \in c} \phi_c(\mathrm{x}_i, \mathrm{x}_c^{(1)}) \qquad \textit{Equation 5.28}$$

The weights for each node in the higher layer in $\phi_c(.)$ are proportional to the sum of the weights in the "base layer" belonging to the clique c. More generally speaking, the energy of the new hierarchical model is of the form as below [109].

$$E(x)=\sum_{i\in V}\varphi_i(x_i)+\sum_{i\in V,\,j\in N_i}\varphi_{ij}^{P}(x_i,x_j)+\sum_{x^{(1)}}E^{(1)}(\mathrm{x},\mathrm{x}^{(1)}) \qquad \textit{Equation 5.29}$$

where the third term in this energy expression could be recursively defined as below [109].

$$E^{(n)}(\mathrm{x}^{(n-1)},\mathrm{x}^{(n)})=\sum_{c\in S}\varphi_c^{P}(\mathrm{x}^{(n-1)},\mathrm{x}_c^{(\mathrm{n})})+\sum_{c,d\in S}\varphi_{cd}^{P}(\mathrm{x}_c^{(\mathrm{n})},\mathrm{x}_d^{(\mathrm{n})}))+\min_{x^{(\mathrm{n}+1)}}E^{(\mathrm{n}+1)}(\mathrm{x}^{(\mathrm{n})},\mathrm{x}^{(\mathrm{n}+1)})$$

Equation 5.30

where $x^{(0)}=x$ represent the state of the base level, and $x^{(\mathrm{n})}$ for $n\geq 1$ describes the state of auxiliary variables. The structure of the graph is defined previously. The inter-layer relationship between two layers of auxiliary variables can be described using a " *weighted robust P^N potential with the unary term $\varphi_c(\mathrm{x}_c^{(\mathrm{n})})$ and pairwise term* " as below [109].

$$\varphi_c(\mathrm{x}_d^{n-1},\mathrm{x}_c^{(\mathrm{n})})=\begin{cases}0 & \textit{if } x_c^{(\mathrm{n})}=L_F\cup x_c^{(\mathrm{n})}=x_d^{(\mathrm{n}-1)}\\ w_d k^{x_c^{(\mathrm{n})}} & \textit{otherwise}\end{cases} \qquad \textit{Equation 5.31}$$

where the weights are summed up over the base layer as $w_d=\sum_{j\in d}w_j$. As formulated in eq. (5.31), $x_c^{(\mathrm{n})}$ is encouraged to take label L_F if either most of its directly connected nodes from the lower layer (we can call these nodes as the children of $x_c^{(\mathrm{n})}$) are assigned with label L_F, or if these children are assigned with different labels.

- *Application of AHRF framework to object classification task*

Based on the definition of AHRF framework described above, a set of potentials used in the object-class segmentation problem have also been presented in [109]. This set consists of unary potentials defined on both pixels and super-pixels, pairwise potentials between pixels and between super-pixels, and connective potentials between different layers in the hierarchical graph structure. In the first model introduced in Section 5.3.2, the unary term in the CRF model has already been decomposed into a set of potentials $\varphi_c(\mathrm{x}_c)$. In the model proposed in [109], the unary potential term is decomposed similarly, writing as $\varphi_c(\mathrm{x}_c)=\lambda_c\xi_c(\mathrm{x}_c)$, where ξ_c is a feature based potential over c and λ_c is its

weight. First the parameter estimation of potentials $\xi_c(\mathrm{x}_c)$ will be introduced, and later discuss the learning of the weights λ_c.

According to [109], the potentials defined over a three-level hierarchy provide sufficiently promising performance, and the performance has not been improved noticeably by increasing the hierarchical level indefinitely. The nodes of each layer are pixels, segments and super-segments, respectively. Here, we apply mean shift based algorithm [49], which has been introduced in Section 3.2, to perform the super-pixel segmentation. The reason for choosing mean shift algorithm is that the spatial and range resolution parameters [49] allow us to adjust the segmentation scales easily. In this case, we can have the segments for the second layer with finer scale segmentation and super-segments for the third layer with coarser scale segmentation.

Features: In related works [99], [109], several features are used and evaluated, including textons based shape filters [99], local binary patterns [118], multi-scale dense SIFT [77], opponent SIFT [119], color distribution features [99] and histogram of oriented gradients (HOG) [120]. As mentioned earlier in this chapter, textons are defined as clustered 17-dimensional responses to 17 different filters (Gaussian, Gaussian derivative and Laplacian filters at different scales). Local binary pattern is an 8-dimensional binary feature, of which each element represents a comparison of the pixel value of the center pixel with its 8 neighbors. The SIFT feature contains the histogram of gradients of 4×4 cells quantized into 8 bins. The resulting 128 dimensional vector is normalized to the range from 0 to 1. Opponent SIFT is another version of the traditional SIFT and based on histograms of gradients for 3 channels in the chosen color space. According to [109], we will generate dictionary which contains 400 words for each descriptor, except the local binary feature, using K-mean clustering method. Then we quantize all descriptors based their own dictionary.

Unary Potentials from Pixel-wise Features: unary potentials from pixel-wise features are derived from TextonBoost as mentioned earlier, and allow us to perform texture based segmentation, at the pixel level, within the same framework. The features used for constructing these potentials are computed on every pixels of the image, and are also called dense features. TextonBoost estimates the probability of a pixel taking a certain

label by booting weak classifiers based on a set of texture layout responses. We observed that textons were unable to discriminate between some classes of similar textures. This motivated us to extend the TexonBoost framework by boosting classifiers defined on multiple dense features (such as color, textons, histograms of oriented gradients, and pixel location) together. The further details of the texton boost procedure have already been described in Section 5.3.1. The results in [109] show that the boosting of multiple features together results in a significant improvement of the scene classification performance (note the improvement from the 72% in [99] to 82% in [109]). The potentials are incorporated into the framework in the standard way as a negative log-likelihood as below.

$$\varphi(\mathrm{x}_i = l) = -\log \frac{e^{H_l(\mathrm{i})}}{\sum_{l' \in L} e^{H_l(\mathrm{i})}} = -H_l(\mathrm{i}) + \mathrm{K}_i \qquad \textit{Equation 5.32}$$

where $H_l(\mathrm{i})$ is the Ada-Boost classifier response for a label l and a pixel i and $\mathrm{K}_i = \log \sum_{l' \in L} e^{H_l(\mathrm{i})}$ is a normalizing constant.

Histogram-based segment unary potentials: in [109], unary potentials are also defined over segments and super-segments. For many object recognition problems, "*the distributions of pixel-wise feature responses are more discriminative than any feature alone*" [109]. The unary potentials of auxiliary segment variables are estimated using multi-class JointBoost mentioned in Section 5.3.1 over the normalized histograms of multiple clustered pixel-wise features. The learning process is the same to the pixel-wise unary potential introduced in Section 5.3.1. The unary potential defined on super-pixels is incorporated into the energy as below [109].

$$\begin{aligned} \varphi_c(\mathrm{x}^{(1)} = l) &= \lambda_s |c| \min(-\mathrm{H}_l(\mathrm{c}) + \mathrm{K}_c, \alpha^h) \\ \varphi_c(\mathrm{x}^{(1)} = L_F) &= \lambda_s |c| \alpha^h \end{aligned} \qquad \textit{Equation 5.33}$$

where $\mathrm{H}_l(\mathrm{c})$ is the response given by the Ada-boost classifier to clique c taking label l, α^h a truncation threshold and $\mathrm{K}_i = \log \sum_{l' \in L} e^{H_{l'}(\mathrm{c})}$ a normalizing constant [109]. In our case, the cost of pixel labels being different from an associated segment labels was set to $k_c^l = (\varphi_c(\mathrm{x}^{(1)} = L_F) - \varphi_c(\mathrm{x}^{(1)} = l)) / 0.1|c|$. It can be seen that 10% of the pixels at most is

allowed to take a label different to the segment label without changing the state of the segment to L_F .

Pairwise Potentials: The pairwise terms on the pixel level $\varphi_{ij}^{P}(.)$ take the form of the classical contrast sensitive Potts potentials as below [109].

$$\xi^{P}(\mathrm{x}_i, \mathrm{x}_j) = \begin{cases} 0 & if\ x_i = x_j \\ g(\mathrm{i}, \mathrm{j}) & \text{otherwise} \end{cases} \qquad \textit{Equation 5.34}$$

where the function g(i, j) describes the edge information based on the pixel value difference of neighboring pixels [98], as shown in eq. (5.35).

$$g(\mathrm{i}, \mathrm{j}) = \theta_p + \theta_v \exp(-\theta_\beta \left\| I_i - I_j \right\|) \qquad \textit{Equation 5.35}$$

where I_i and I_j are the color vector of pixel *i* and *j*, respectively. This pairwise constraint encourages neighboring pixels in the image (having a similar color) to have the same label. More details can be found in [109].

As stated in [109], we use the pairwise potential in the segment level defined as in eq. (5.36). This potential forces the super-pixels with the same texture and color features to be assigned with the same label. The term $g(\mathrm{c}, \mathrm{d}) = \min(|c|, |d|) \left| h(\mathrm{x}_c^{(1)}) - h(\mathrm{x}_d^{(1)}) \right|_2^2$, where $h(.)$ is the normalized histogram for color values of a given segment [109], is based on the Euclidean distance of histograms between given neighboring two segments, indexed as *c* and *d*.

$$\xi_{cd}^{P}(\mathrm{x}_c^{(1)}, \mathrm{x}_d^{(1)}) = g \begin{cases} 0 & if\ x_c^{(1)} = x_d^{(1)} \\ g(\mathrm{c}, \mathrm{d}) / 2 & if\ (x_c^{(1)} = L_F \ \&\ x_d^{(1)} \neq L_F) \cup (x_c^{(1)} \neq L_F \ \&\ x_d^{(1)} = L_F) \\ g(\mathrm{c}, \mathrm{d}) & \text{otherwise} \end{cases}$$

Equation 5.36

5.3.4 CRF Model 3

Although the hierarchical connectivity and high-order potentials defined on super-pixel are incorporated into the CRF framework, some researchers claim that these methods suffer from the instability of unsupervised super-pixel segmentation algorithms, especially with respect to the ability to recognize objects with complicated boundaries

[100]. In [100], a fully connected CRF model has been proposed to refine the image labeling results. In the fully connected model, each pair of pixels in the image is connected by an edge, which has been further associated with a pairwise potential. The main challenge for this model is the size of this model [100]: even for a low resolution image, there are in the order of 10^6 of nodes and 10^{10} of edges. To deal with this gigantic problem, a highly efficient inference algorithm has been proposed in [100]. In this approach, the pairwise edge potentials are defined by a linear combination of Gaussian kernels in an arbitrary feature space [100]. The CRF distribution was estimated by a mean field approximation [107]. Most importantly, it was proved that a mean field update of all variables in a fully connected CRF can be performed using Gaussian filtering in feature space [100]. This inference algorithm is sub-linear in the number of edges in the model. The details about the dense connected CRF model and the inference algorithm will be provided in following two sub-sections.

- *The fully connected CRF Model*

The basic conditional random field (CRF) model has been introduced earlier this section. In the fully connected pairwise CRF model, we consider all unary and pairwise cliques in the complete graph *G* defined on the entire image. As in eq. (5.4), the pixel-wise unary potential was computed independently for each pixel by a classifier that provides a probabilistic distribution of the assigned label given image features. The pairwise potentials in the fully connected CRF model are defined as below [100].

$$\begin{aligned} \varphi_{ij}^{p}(x_i, x_j) &= \mu(x_i, x_j) k(f_i, f_j) \\ k(f_i, f_j) &= \sum_{m=1}^{K} w^{(m)} k^{(m)}(f_i, f_j) \end{aligned} \qquad \textit{Equation 5.37}$$

where each $k^{(m)}$ is a Gaussian kernel defined as in eq. (5.38), the vectors f_i and f_j are feature vectors for pixel i and j in an arbitrary feature space, w^m are linear combination weights, and μ is a label compatibility function, which could be defined as simple as Potts model [121]: $\mu(x_i, x_j) = [x_i \neq x_j]$. It penalizes the adjacent similar pixels to be assigned different labels. Each kernel $k^{(m)}$ is characterized by a symmetric, positive-definite precision matrix $\Lambda^{(m)}$, which defines its shape.

$$k^{(m)}(f_i, f_j) = exp(-\frac{1}{2}(f_i - f_j)\Lambda^{(m)}(f_i - f_j)) \qquad \textit{Equation 5.38}$$

In [100], a contrast-sensitive two kernel potential, as shown in eq. (5.39) and defined on the color vectors (I_i and I_j) and positions (p_i and p_j), is used.

$$k(f_i, f_j) = w^{(1)} \exp(-\frac{\left|p_i - p_j\right|^2}{2\theta_\alpha^2} - \frac{\left|I_i - I_j\right|^2}{2\theta_\beta^2}) + w^{(2)} \exp(-\frac{\left|p_i - p_j\right|^2}{2\theta_\gamma^2}) \qquad \textit{Equation 5.39}$$

where the first term is called the appearance kernel which encourages the neighboring pixels with similar color to be assigned the same label. According to [100], the second term, called the smoothness term, is used to remove small, isolated regions.

- *Efficient Inference in Fully Connected CRFs*

Usually, the inference for grid connected random fields with dense structure makes the traditional message passing algorithm [122] converge slowly, resulting in high cost solutions. According to [109], the difficulties faced by these message passing algorithms can be attributed to the presence of frustrated cycles (where a cycle refers to some number of nodes in the graph are connected in a closed chain) that can be eliminated via the use of cycle inequalities, but only by significantly increasing run time. On the other hand, graph cut based move making algorithms don't suffer from this problem, but have been shown to be less powerful. For these reasons, [100] provides a filter-based approach for performing fast approximate maximum posterior marginal (MPM) inference (the optimal solution satisfies the formula in eq. (5.40)) in multi-label CRF models with fully connected pairwise terms. As mentioned earlier, a random field is a set of random variables $X = \{X_1, ...X_N\}$ conditioned on an image I , where *N* represents the number of variables (pixels). The random variables take values from a label set $L = \{l_1, ...l_L\}$. The fully connected pairwise CRF is shown as in eq. (5.41).

$$x_i^* = \arg\max_x p(x \mid I) \qquad \textit{Equation 5.40}$$

$$P(X \mid I) = \frac{1}{Z(I)} \exp(-E(X \mid I))$$
$$E(X \mid I) = \sum_{i \in N} \varphi_u(x_i) + \sum_{i<j \in N} \varphi_p(x_i, x_j) \qquad \textit{Equation 5.41}$$

where $E(X \mid I)$ is the energy functional that need to be minimized. As mentioned in Section 5.2.1, $Z(I)$ is the partition function. Both the unary and pairwise potential terms, as mentioned in Section 5.3.4, are implicitly conditional on the image I .

Given this form of CRF in eq. (5.41), the fast approximate MPM inference can be performed by cross bilateral filtering techniques with a mean-field approximation framework [107]. The details will be given below.

The mean field approximation involves an alternative distribution $Q(X)$ over the random variables of the CRF, and the marginal on this distribution are defined to be independent, i.e. $Q(X) = \prod_i Q_i(x_i)$. The mean-field approximation makes attempt to minimize the KL-divergence $D(Q\|P)$ between Q and the true distribution P , where KL-divergence is a measure of the difference between two probabilities Q and P , and this difference reflects the amount of information lost when Q is used to approximate P [123]. By considering the fixed-point equations that must hold at the stationary points of $D(Q\|P)$, the following update can be derived for $Q_i(x_i = l)$ given the settings of $Q_j(x_j)$ for all $j \neq i$ as shown below [123].

$$Q_i(x_i = l) = \frac{1}{Z_i} \exp\{-\varphi_u(x_i) - \sum_{l' \in L} \sum_{j \neq i} Q_j(x_j = l') \varphi_p(x_i, x_j)\} \qquad \textit{Equation 5.42}$$

$$Z_i = \sum_{x_i = l \in L} \exp\{-\varphi_u(x_i) - \sum_{l' \in L} \sum_{j \neq i} Q_j(x_j = l') \varphi_p(x_i, x_j)\} \qquad \textit{Equation 5.43}$$

where Z_i is a constant which normalizes the marginal at pixel i . If the updates in eq. (5.42) are made in sequence across all pixels (updating and normalizing the L values $Q_i(x_i = l)$ where $l = 1...L$ at each step), the KL-divergence is guaranteed to decrease [123]. According to [123], parallel updates for eq. (5.42) can be evaluated by convolution with multi-dimensional Gaussian kernel using any efficient bilateral filter, such as a noise-reducing smoothing filter for images where the intensity value at each pixel in an image is replaced by a weighted average of intensity values from nearby pixels [123]. This can be achieved by the formulation shown as below.

$$\tilde{Q}^{(m)}(l) = \sum_{j \neq i} k^{(m)}(f_i, f_j) Q_j(l) = [G_m \otimes Q(l)](f_i) - Q_i(l) \qquad \textit{Equation 5.44}$$

where G_m is a Gaussian kernel corresponding to the m^{th} component of eq. (5.37). Since $Q_j(x_j = l')\varphi_p(x_i, x_j)$ can be written as $\sum_m w^{(m)} \tilde{Q}^{(m)}(l')$, and the complexity of approximate Gaussian convolution using the method in [123] is linear with the number of pixels. The algorithm is run for a fixed number of iterations. At the final iteration, the MPM solution is obtained by searching for the maximum x_i as formulated in eq. (5.45).

$$x_i^* = \arg\max_l Q_i(x_i = l) \qquad \textit{Equation 5.45}$$

5.4 Wound Recognition System Design Based on CRF Models

In this section, we design a wound classifier based on the CRF models described in Section 5.3. The entire model training and wound recognition process is illustrated in Figure 5.4. Different CRF models correspond to different unary and pairwise potential terms, as discussed in Section 5.3.

5.4.1 Classifier Training

The wound classifier training process is shown in the left column in Figure 5.4. Most modules in the system have been introduced in details in Section 5.3. As mentioned in earlier chapter, there are many different algorithms for super-pixel segmentation. In our system, we apply the parallel version of SLIC algorithm [37] due to its good boundary adherence and efficient implementation. Note that there is no need for super-pixel segmentation and super-pixel based potential training if CRF model 1 or 3 described in Section 5.3 has been applied.

According to [23], color based features in various color spaces have proven to be the most effective discriminative indicators for wound classification against the healthy skin. As mentioned in Section 4.2, other texture based features have also been adopted as auxiliary tools. In our classifier, we extract filter-bank based features [99], dense SIFT (DSIFT) feature [77], location features [99] and opponent SIFT [109] feature for boosting the unary potential term. According to Section 4.4, our wound recognition system is expected to be able to determine the accurate wound boundary from less controlled

images where the illumination and angles can vary. Especially, these images might contain wounds acquired at different ranges, and the images may also contain other background objects. Therefore, we apply these features which have already provided promising performance on object classification tasks in natural scene images [99], [109]. The features used for unary potential term training are all supposed to be extracted densely, i.e. the feature vector is extracted at every pixel location for all training images. However, this is not practical considering the large volume of dataset and high resolution of images. Instead, we extract features on a down-sampled version of the original image grid. The down sampled rate for each feature has been determined empirically.

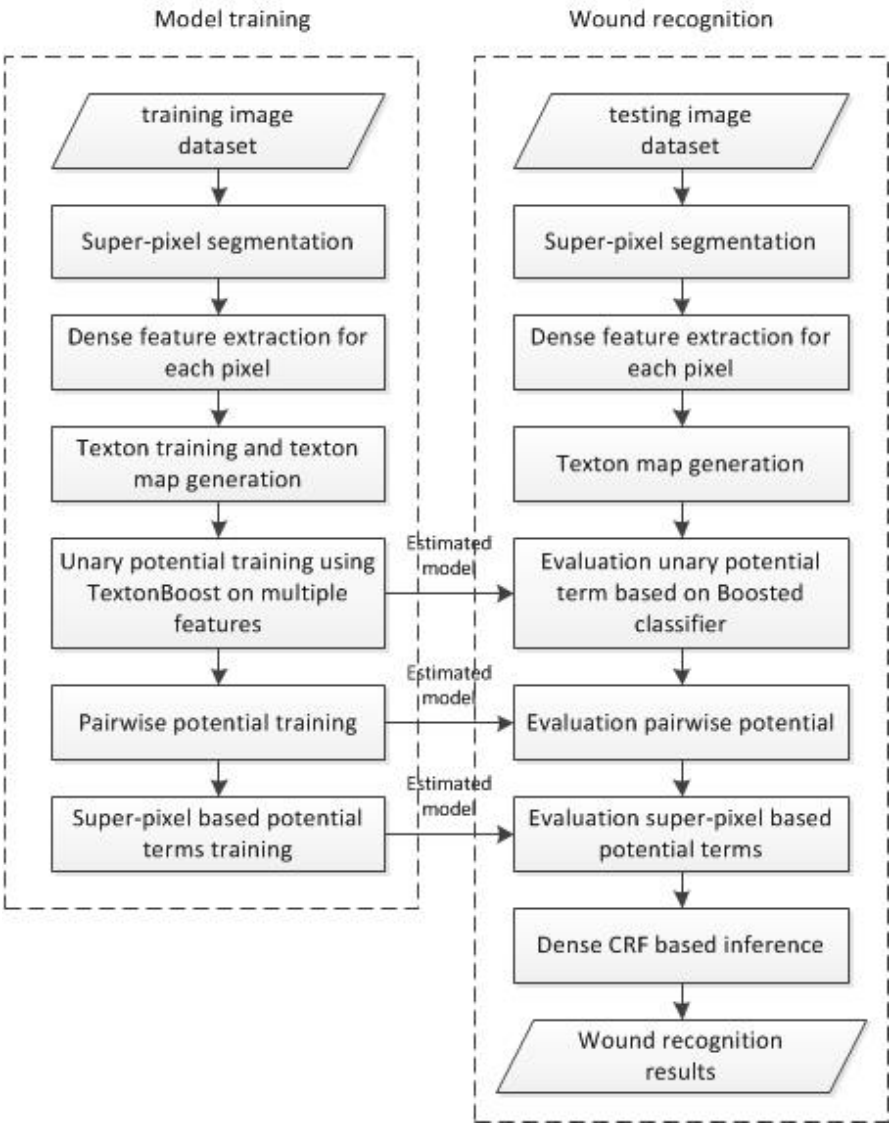


Figure 5.4 Wound recognition classifier based on CRF model

According to [99], [109], there are two different ways to incorporate these features into the CRF model. In the first method, we can further decompose the unary potential term as a weighted summation $\varphi(x) = \sum_c \lambda_c \xi_c(x)$, where $\xi_c(x)$ is a feature-based potential and λ_c is its weight. We need to perform the joint boost approach to learn each feature based potential, then we estimate the weights using local search scheme on a validation set [109]. This training method turns out to be robust, but time consuming as well.

The second method to learn a single unary potential term is implemented by combining multiple dense extracted features together. After extracting each feature over the image grid, we perform the texton generation. In this case, we have *NM* texton channels in total where *N* is the number of types of features in total and *M* is the cluster center number for texton generation. Before performing the texture-layout filtering, we calculate the integral image [99], as mentioned in Section 5.3.2, for each channel. Then we extract the texture layout based features based on these *NM* texton channels. Finally, we perform the joint-boost approach to learn the final unary term only for one time. Weighing the strengths and weaknesses of the first and the second method, we have chosen to apply the second method.

For the pairwise potential terms, no matter which formulation we are using, the parameters of the model are manually selected to minimize the error on the validation set using grid search approach [94], [97]. The details about the grid search method have been introduced in Section 4.2.

5.4.2 Wound Recognition Performance Evaluation

For evaluating the CRF methods' ability to recognize a wound and determine its boundary on a given set of images, the super-pixel segmentation and feature extraction are the same as is used in the training process. We applied the learned textons to generate the texton map for each feature channels. Afterwards, we evaluate the unary potential, pairwise potential and segment based potentials (if applicable) based on the model learned in the training process. Then we apply the CRF inference method to find the optimal labeling over the entire wound image.

5.4.3 Experimental Results

To evaluate the performance of the wound recognition system designed in Section 5.4.1 and 5.4.2, we apply the three CRF models, introduced in Section 5.3, to two different wound image datasets. The first image dataset is composed of images of Moulage wounds placed on an artificial foot. The second dataset consists of images of real diabetic foot ulcers from recruited subjects at the Wound Clinic in UMass Medical School. To better evaluate our system, the wounds in images of the first dataset were captured at different ranges, illumination levels and viewpoints. Specifically, we collected 162 images of 6 Moulage wound for the first dataset. 27 images for each wound, at 3 different ranges, 3 different viewpoints, and 3 different illumination conditions, were captured. In the second training dataset, 100 images were captured for 18 subjects and most of them were acquired using the image capture box designed in earlier chapter.

To evaluate the performance of the wound recognition over the entire dataset, we split both dataset equally into 10 folders. Then a ten-folder validation method is carried out as follows. We will perform the "train and test" operation for 10 rounds. In each round, we train the model on 9 folders and test the model on the remaining folder. The general specificity and sensitivity defined in Section 4.3 are evaluated by combining the testing results from 10 rounds. For the Moulage image dataset, we label the image using 4 different labels: the wound, gel which is the transparent material that surrounds the Moulage wound, the healthy skin and the background, as shown in Figure 5.5 (a) and (c). For the real image dataset, the image is labeled into 3 labels, which are the same as the 4 labels except the surrounding gel category, as shown in Figure 5.5 (b) and (d).

To compare the three CRF models, we apply these models one by one independently on the same two datasets in the ten-folder validation approach mentioned above. The two most important parameters are the cluster center N for the texton generation and the boosting iteration number M for joint boost training scheme. To obtain better parameter estimation, we perform a grid search method to select the best parameter pair (N, M). We perform the CRF model 3 on the moulage image dataset using the ten-fold validation method mentioned above. The Matthews Correlation Coefficient [75] results are shown as in Table 5.1. And the wound recognition computation time evaluation results are

shown in Table 5.2. We didn't evaluate the training efficiency evaluation since the model training is supposed to be performed "offline".

(a) *(b)* *(c)* *(d)*

Figure 5.5 Single sample of original image and ground truth from two dataset, (a) Original image of the dataset 1, (b) Original image of the dataset 2, (c) Ground truth labeling of the dataset 1: green for background, yellow for healthy skin, blue for artificial gel and red for wound, (d) Ground truth labeling of the dataset 2: the same labeling color fashion as in dataset 1 except for the absence of artificial gel category

Table 5.1 Matthew Correlation Coefficient results using different (N,M) parameter settings

	N = 100	*N = 200*	*N = 300*	*N = 400*	*N = 500*	*N = 600*
M = 1000	*0.393*	*0.438*	*0.471*	*0.523*	*0.532*	*0.538*
M = 2000	*0.469*	*0.498*	*0.547*	*0.596*	*0.602*	*0.606*
M = 3000	*0.550*	*0.582*	*0.617*	*0.648*	*0.651*	*0.655*
M = 4000	*0.598*	*0.632*	*0.668*	*0.699*	*0.694*	*0.703*
M = 5000	*0.707*	*0.738*	*0.769*	*0.813*	*0.816*	*0.821*

Table 5.2 Wound recognition time using different (N,M) parameter settings (Unit: seconds)

	N = 100	*N = 200*	*N = 300*	*N = 400*	*N = 500*	*N = 600*
M = 1000	*10.2*	*10.3*	*10.8*	*11.0*	*11.2*	*11.3*
M = 2000	*18.1*	*18.5*	*19.9*	*20.2*	*22.0*	*22.4*
M = 3000	*27.7*	*28.2*	*30.0*	*30.9*	*31.3*	*33.0*
M = 4000	*38.8*	*39.9*	*41.3*	*41.5*	*42.9*	*42.9*
M = 5000	*46.3*	*47.2*	*49.3*	*50.1*	*50.5*	*51.2*

Based on the results shown in Table 1 and 2, when $N = 600$ and $M = 5000$, the MCC result is the best. Moreover, we can see that the MCC value increases as we increase the boosting iteration number, but the time performance decreases obviously. On the other

hand, when the cluster center number N becomes larger than 400, there is no obvious improvement for the MCC result. However, increasing the cluster center will substantially increase the computation burden for the model training. In conclusion, we set $N= 400$ and $M = 3000$ empirically for the best tradeoff of accuracy and efficiency.

The sample wound recognition results are shown in Figure 5.6 and Figure 5.7 for dataset 1 and 2, respectively. The specificity and sensitivity evaluation results for the three CRF models on two dataset are shown in Table 5.3 and 5.4. Finally, the time performance results for wound recognition are shown in Table 5.5. We can see the Model 1 didn't perform the wound recognition very well on multi-scale situation, since it is a pairwise model where the pairwise potential terms have only been evaluated on pair of pixels in the same clique. Model 3 out-performed Model 1 on wound recognition accuracy since it generated the pairwise potentials on each pair of pixels in the image. In this case, the long-range connections were incorporated into the CRF formulation. The CRF Model 2 provided even better wound recognition performance than Model 3, i.e. the best of the three models introduced in this chapter, especially dealing with images of the same wound captured in different ranges (scales), viewpoints and illumination conditions, due to its hierarchical structure involving super-pixel based higher-order potential terms. According to [109], the potentials defined over a three-level hierarchy provide the best tradeoff between the time performance and recognition performance, although the hierarchy can be extended indefinitely. It is also found that performance saturated when the number of hierarchy level increases beyond three. However, the Model 2 required longer computing time than other two models due to the super-pixel segmentation and more potential terms to be evaluated. The comparison of these three models in terms of specificity and sensitivity is shown in Table 5.3.

The comparison between the results presented in Table 5.3 – 5.5 and Table 4.4 leads to the conclusion that the CRF model based methods are a better option than the super-pixel based SVM classifier for wound boundary determination tasks with relaxed image capture constraints. However, the SVM classifier based approach is far more computationally efficient than the CRF model based method, which can be easily proved by comparing the last two columns in Table 5.5. Hence, for smartphone alone system, the

method presented in Chapter 4 is a better candidate considering the tradeoff of performance and efficiency.

Figure 5.6 Samples of wound recognition results on dataset 1, Row 1: the original images, image 1-3 and 4-6 represent the two different wound simulation in different scales and viewpoints, respectively, Row 2-4: wound recognition results provided by CRF model 1-3, the wound areas are labeled in red color.

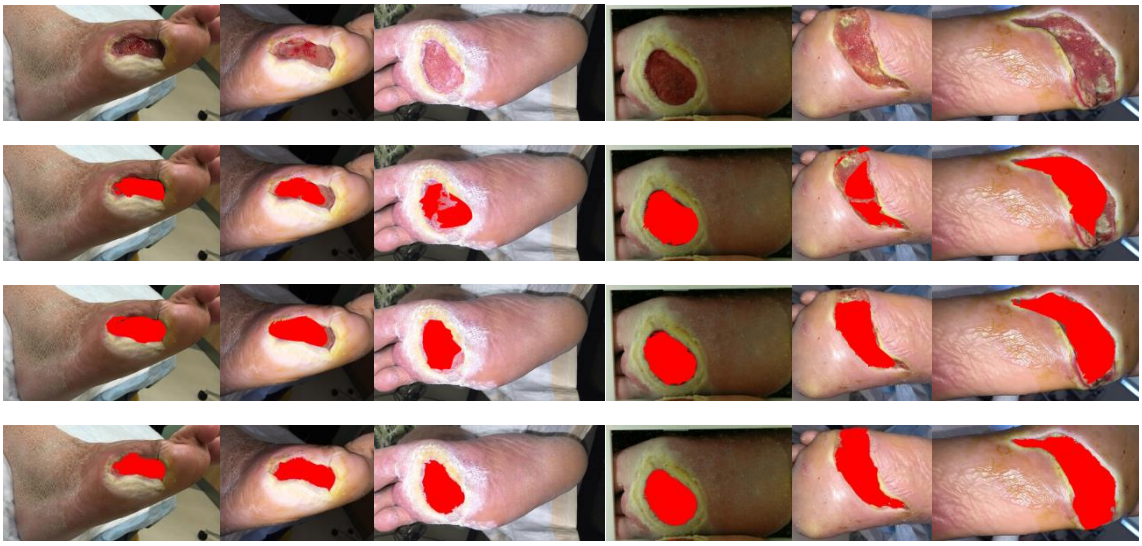


Figure 5.7 Samples of wound recognition results on dataset 2, Row 1: the original images, image 1-2, 3-4, and 5-6 represent three different wounds imaged in different scales, viewpoints and illumination, respectively. Row 2-4: wound recognition results provided by CRF model 1-3, the wound areas are labeled in red color

Table 5.3 Wound recognition specificity using different CRF models on two datasets

	CRF model 1	*CRF model 2*	*CRF model 3*
Dataset 1	*0.927*	*0.992*	*0.984*
Dataset 2	*0.898*	*0.955*	*0.911*

Table 5.4 Wound recognition sensitivity using different CRF models on two datasets

	CRF model 1	*CRF model 2*	*CRF model 3*
Dataset 1	*0.674*	*0.844*	*0.767*
Dataset 2	*0.618*	*0.769*	*0.703*

Table 5.5 Wound recognition computation time using different CRF models on two datasets (Unit: seconds)

	CRF model 1	*CRF model 2*	*CRF model 3*	*SVM based method*
Dataset 1	*36.7*	*57.4*	*30.9*	*7.3*
Dataset 2	*37.4*	*60.3*	*35.9*	*8.4*

Chapter 6

Wound Healing Status Assessment

With the wound boundaries determined by one of the wound recognition algorithms, described in Chapters 3 - 5 (the wound areas can be indirectly determined at the same time), we will further analyze the wound based on its color features. Then, we wish to quantitatively assess the wound healing status for a given wound in an image, and further carry out a comparison analysis of the healing status of the same wound in images captured at a later times. In this chapter, we will first discuss the color based wound tissue classification. Next, a quantitative measure, called the healing score, will be formulated, in order to evaluate the foot ulcer healing status based on a sequence of wound images. This will be tested on a sequence of wounds captured at the Wound Clinic in UMass Medical School. Finally, we will introduce the design of a collaborative wound image analysis system used at the Wound Clinic to track the wound status for visiting patients.

6.1 Color Based Wound Tissue Segmentation

6.1.1 Red-Yellow-Black (RYB) Model

The RYB (red-yellow-black) wound classification model, proposed in 1988 by Arnqvist, Hellgren and Vincent, is a consistent, simple assessment model to evaluate chronic wounds [35]. It classifies wound tissues within a wound boundary as red, yellow, black or mixed tissues, which represent the different phases on the continuum of the wound healing process. Specifically, red tissues are viewed as the inflammatory (reaction) phase, proliferation (regeneration), or maturation (remodeling) phase; yellow tissues imply

infection or tissue containing slough that are not ready to heal; and black tissues indicate necrotic tissue state, which is not ready to heal either [35]. Based on the RYB wound evaluation model, our wound analysis task is to classify all the pixels within the wound boundary into the RYB color categories and cluster them. Therefore, classical clustering methods will be applied to solve this task.

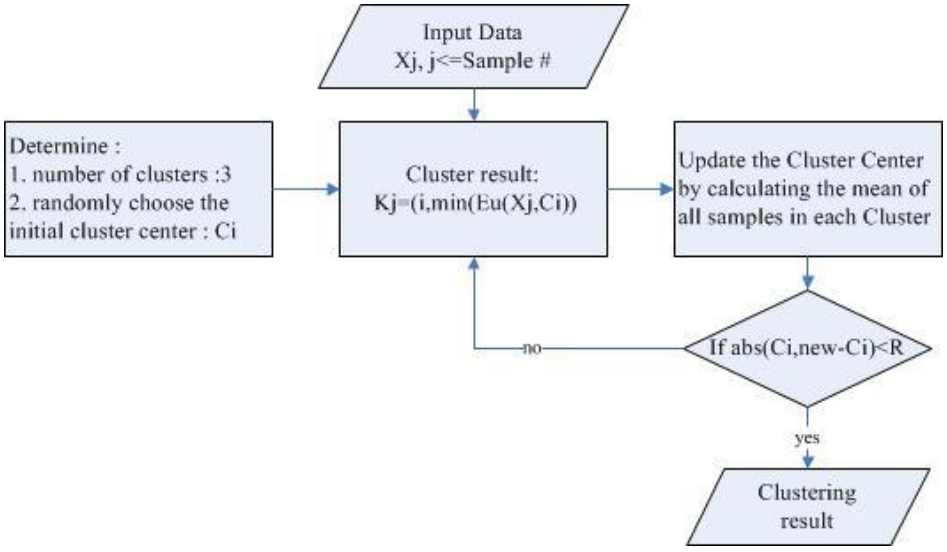
6.1.2 K-Mean Algorithm Basics

In unsupervised clustering methods, *k*-means clustering is "a method of cluster analysis which aims to partition *n* observations into *k* clusters in which each observation belongs to the cluster with the nearest mean" [36]. In our case, all pixels within the wound boundary can be viewed as observations. The three colors referred in RYB model are regarded as clusters. The algorithm is presented in block diagram form as well as graphically in Figure 6.1.

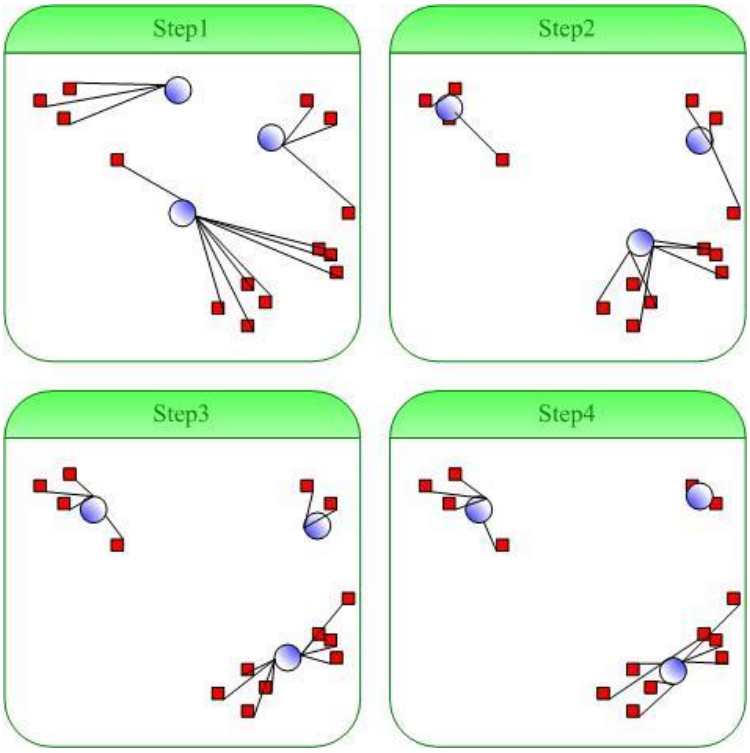
There are several issues that need to be further clarified.

- The color difference between the cluster center and the target pixel (expressed as Eu in the flowchart in part (a) in Figure 6.1) is calculated by the Euclidean distance in CIE Lab space. This measure has been shown to be consistent with the color difference perceived by human eyes.
- Strictly speaking, the K-mean algorithm is a NP-hard problem, which means that it is unable to converge to a solution within limited time when the image size is large. However we can terminate the iteration when the mean variance of each cluster is smaller than a specified threshold. This heuristic method is expressed as the decision block in part (a) of Figure 6.1. In part (a) of Figure 6.1, the initial centers are chosen randomly. However, in practice we may specify the initial centers according to some empirical values such as the Macbeth Color Checker table described in [67]. By this operation, the converging speed will increase and make the color clustering process more efficient.
- As shown in Figure 6.1, we preset the number of cluster center to be 3. However, the number could be smaller or bigger than 3. When the number is greater than 3, we have to perform some post-processing to the resulting clusters in order to combine such clusters that are close to each other. This post-processing can be performed by

determining whether the mean value difference between a pair of clusters is smaller than a preset threshold.



(a) K-mean algorithm flowchart



(b)K-mean algorithm description

Figure 6.1 K-mean algorithm

6.1.3 Wound Tissue Segmentation Results

Color segmentation results provided by K-mean algorithm based on wound boundary determination results on sample real wound images are shown in Figure 6.2. The results are promising despite a few misclassified pixels in part (c) and (f). In part (c), some dark red part is recognized as black tissue. In part (f), some "dark yellow" part is classified as the red tissues. In general, the wound analysis task is more complicated for the clinical images of real patients compared to those of moulage wounds due to the complicated skin color and texture of patients' feet and various wound locations.

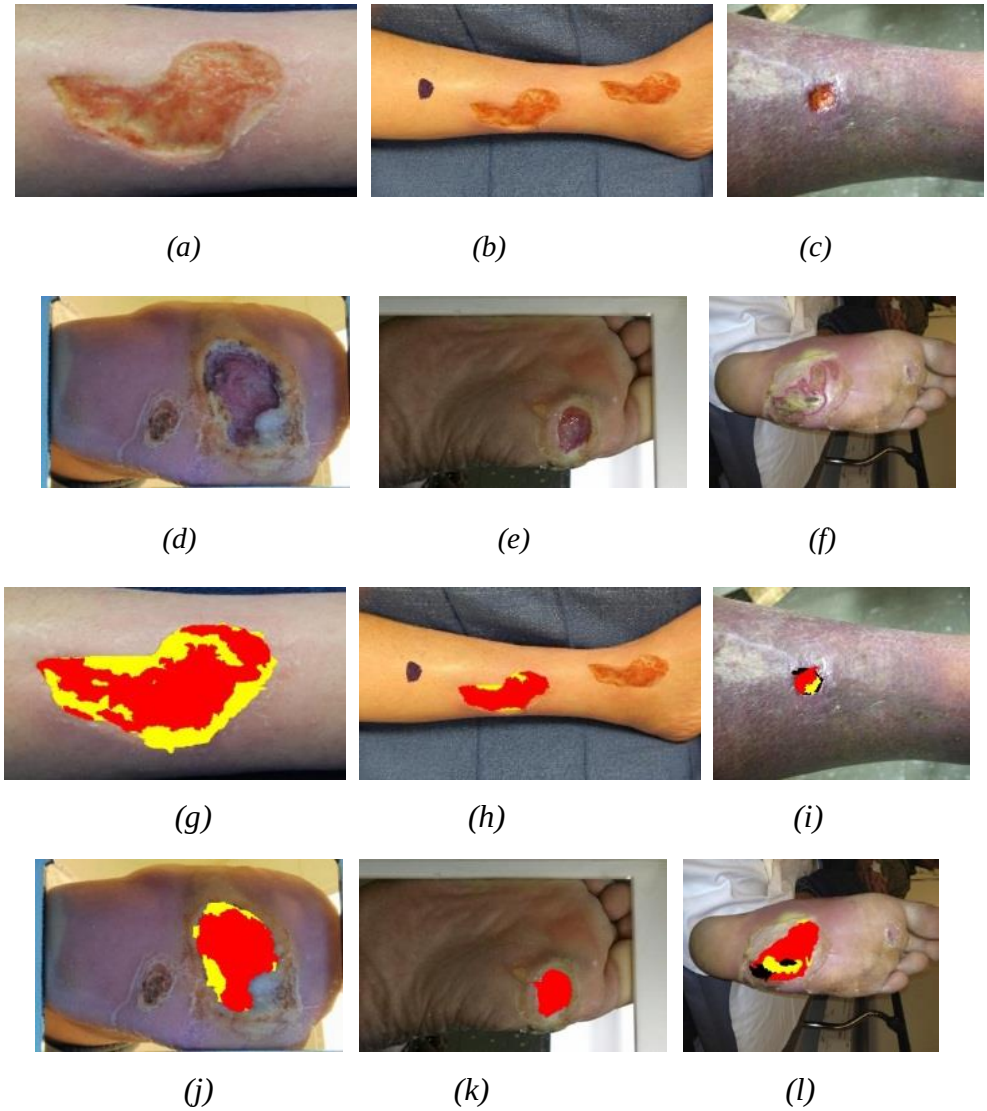


Figure 6.2 Wound tissue segmentation results based on color, (a) - (f): original wound images, (g) – (l): color segmentation results

6.2 Healing Score: A Quantitative Measure for Wound Healing Status

Our final goal is to provide more meaningful wound analysis results to the users, where the users might be either clinicians and/or diabetic patients. For clinicians, the wound area size and different color tissue composition may be sufficient. They can make their diagnosis based on these raw data. However, for patients assumed not to have clinical knowledge about wounds, only providing them some absolute numbers does not give them with much help in understanding of their actual wound status. Hence, there is a need to translate the raw data into a meaningful numerical value, like a score in the range of 0-10, where larger simply means better. In this section, we propose a numerical wound evaluation measure called healing score.

6.2.1 Healing Score Definition

To create a measure of wound healing status, we translate the raw wound assessment results into a numerical value called healing score (S_n) using eq. (6.1) – (6.3). Such a single-valued healing score will provide patients and caregivers with a simple measure of the wound healing or wound deterioration relative to the wound status at the initial visit. This score can range from 0-10. The larger the score is, the better the healing status is. The fundamental principle underlying the healing score design is the RYB evaluation model as mentioned in Section 6.1. The calculation of the healing score is described in the 3 steps below.

- Step1: For each patient, a reference score of 5 is assigned to the wound image at the first visit to the wound clinic;
- Step2: At each subsequent visit, the weighed area of the wound is calculated by applying eq. (6.1), where WA_n represents the weighted area of the wound at the nth visit. RA_n, YA_n and BA_n represent the red, yellow and black tissue areas, respectively. $[W_R, W_Y, W_B]$ is the vector of weights for red, yellow and black tissue areas, respectively. From clinical observations, changes in yellow and black tissue areas influence the healing status more than do changes in the red tissue area, which can be expressed as $W_R < W_Y < W_B$. In our case, we empirically determined an appropriate weight vector to be [1, 1.5, 2.5].

$$WA_n = W_R RA_n + W_Y YA_n + W_B BA_n \qquad \textit{Equation 6.1}$$

- Step 3: Compute a relative healing score using eq. (6.2) to compare WA_n with WA_0, the weighted area for the first wound image of the current patient. The parameter *G* is an empirically determined gain value, ranging from 0 – 1, to control the dynamic range of the healing score such that our assessment results match clinicians' judgments.

$$S_n = 1 - \frac{WA_n - WA_0}{WA_0} * G = (1+G) - G\frac{WA_n}{WA_0} \qquad \textit{Equation 6.2}$$

We find that the gain values of 0.5 and 0.4 provide similarly good results. Choosing G = 0.4, we verified that S_n ranges from 0 to 1.4 if we assume that WA_n is bound by $0 < WA_n < 3.5WA_0$.

To normalize S_n into the range [0, 10], we multiply the expression in eq. (6.2) by 10/1.4. This results in the final formulation of the healing score, as given in eq. (6.3). It is easily verified that the healing score increases from 0 (wound condition is seriously degraded) to 10 (wound is completely healed) as the weighted wound area decreases from its upper bound ($3.5WA_0$) to 0.

$$S_n = 10 - \frac{2.857WA_n}{WA_0} \qquad \textit{Equation 6.3}$$

The healing score is a simple, but useful mathematical construct, which may be applicable to other types of chronic wounds, such as venous ulcers, possibly requiring a parameter adjustment.

6.2.2 Clinical Assessment of Healing Score

- *Theoretical Basics*

To provide clinical validation for our wound healing score, which is meant as a tool for quantitative wound assessment, we need to compare the assessment results of our algorithm with the evaluation results provided by experienced wound clinicians, based on either wound images alone or wound images together with additional information. We presented the wound images by a laptop based custom App to each participating clinician independently. Then the clinician was asked to score the wound healing status multiple times, based on different amount of measurement data of the wound.

The wound image captured at the first visit for each patient was always given the score of 5. This wound image was taken as the reference image for the assessment of later images. For the wound images collected from patients' following visits, their wound healing status was scored based on the assumptions below:

- ✓ Assumption 1: relative to the wound observed at the first visit, in the subsequent visits the wound can begin to heal or get worse. Hence, the score can either increase or decrease from the initial score of 5.
- ✓ Assumption 2: the assigned healing score ranges from 0-10. The larger the score, the better the healing status. Specifically, a score ranged from 0-4 represents the wound healing status is becoming worse; a score ranged from 6-10 represents the wound healing status is becoming better.
- ✓ Assumption 3: score 10 represents the wound being almost healed, and score 0 represents a worsening of the wound conditions, to the point where a new treatment method is deemed necessary.

The before-mentioned computer-based app was used to present wound images to clinicians. Only the first image was shown initially, and each click of the 'Next Image' button brought up a new image for scoring, while retaining the previous images, as shown in Figure 6.3. For example, we first displayed the initial wound image; then we displayed the initial wound image with the next acquired wound image, and we asked the physician to score the latest image. The same is the case with the third wound image; it was displayed together with the initial image and the second image. The clinicians were only allowed to score the latest image presented.

In the first round of scoring, we showed original wound images for each patient to three clinicians independently and asked each one of them to score the healing status by comparing each image with the wound image collected during the first visit. In this evaluation, the scoring of the healing status was completely based on the clinicians' observation or measurement using the rulers, without any assistance from our wound assessment algorithm and based on no given rules.

In the second round of scoring, we showed original wound images for each patient, along with the wound area size and relative area ratio of the red, yellow and the black tissues within the wound boundary as determined by our algorithm, to three clinicians independently and asked each one of them to score the healing status. As a reminder, in the Red-Yellow-Black wound evaluation system, red, yellow and black represent the healing, infected and necrotic tissues, respectively. We compared the score results between these two evaluation scenarios to see how the accurate wound area measurement by our algorithm could help clinicians to better assess the wound healing status.

Figure 6.3 Software interface screenshot for presenting wound images to clinicians. Clinicians click the "Next Image" button to view the next image for current patient, click the "Next Patient" button to score the images for the next patient and click "Next Phase" button to score all images again with different given information.

- *Clinical Assessment Results*

To evaluate our wound assessment method, we involved 12 patients over a period of one year where each patient was seen over a period ranging from 1 month to 5 months in the Wound Clinic at UMass Medical School, based on an approved IRB protocol. Among the

12 patients, 9 of them were monitored over at least 2 consecutive visits (2 visits for 3 patients, 3 visits for 4 patients, 4 visits for 1 patient and 6 visits for 1 patient). In total, 32 foot ulcer images were collected (one patient, visiting only once, had foot ulcers on both feet) and 28 images were used for the clinical validation of the healing score algorithm. The two sets of scores mentioned in the previous section from the clinicians were compared to the scores, generated by the healing score algorithm, by calculating the Krippendorff's Alpha Coefficient (KAC) [124]. KAC is a statistical measure of the agreement of ratings given by two or more clinicians. The value of this coefficient ranges in $[-\infty, 1]$, where value 1 indicated perfect agreement and value 0 indicates the absence of agreement. A value less than 0 implied that the disagreement is systematic and exceeds what can be expected by chance.

First, we utilized the KAC to compare the consistency of healing score among the three clinicians (also referred to as 'raters'), both for the case where the clinicians are presented with only the wound image and for the case where wound assessment data are also available (Figure 6.3 illustrates the latter case). The calculated coefficients are referred to as the *inter-rater consistency coefficients*. The results are shown in the diagonally symmetrical Table 6.1 where the top number in each cell is the consistency coefficient for wound image only while the bottom number is the coefficient for wound image plus quantitative wound data. We can see that Clinician 1 and 2 have similar assessment about the wound healing status irrespective of whether the quantitative data is presented (KAC > 0.8 in Cell (1, 2)). Clinician 3's assessment differs somewhat from that of the other 2 clinicians (KAC < 0.5 in Cell (1, 3) and the top number in Cell (2, 3)). This lack of agreement is also evident in Figure 6.4, where the red curve (scores from Clinician 3) deviates from the green and black curves (scores from Clinicians 1 and 2) at several points. Another finding was that Clinician 2 and 3 agreed more when the wound quantitative data is also presented (KAC > 0.6 for the bottom number in Cell (2, 3)). Due to the limited number of clinicians and wound samples, our preliminary results indicating that adding wound data could have some influence should be tested with a larger group of clinicians and additional samples.

Next, the effect of providing quantitative wound data, in addition to the wound image itself, on the healing scoring of a given clinician (or 'rater') was evaluated by determining the KAC between the healing scores with and without the quantitative wound data presented. The evaluation results were reflected in the *intra-rater data impact coefficients*. The quantitative wound data consisted of healing score, total wound area and area components of red, green and black tissues. The results were given in Table 6.2 for the three clinicians, showing a modest, but detectable effect ($0.8 < KAC < 0.9$ for each cell); had there been no effect, KAC would be 1.0. We conclude that adding quantitative data to visual image appears to result in better and/or more consistent assessments, but with our limited set of observations, we cannot generalize as to whether these results would apply in a larger wound sample.

The agreement between the algorithm-based healing score and the clinician-based healing score was measured similarly, using the KAC. The results were given in Table 6.3, where the measured coefficients were called the *clinical validity coefficients*. As with the inter-rater consistency coefficients, the results were provided for both the case where the clinicians see only the wound image (top values) and the case where they see both the wound image and the quantitative wound data (bottom data). The values in Table 6.5 showed that our healing score algorithm agrees well with Clinician 2 ($KAC > 0.8$ especially when quantitative wound data is presented) and had an acceptably good agreement with Clinician 1 ($KAC > 0.6$). The KAC value for the scoring results from Clinician 3 and our algorithm was less than 0.5, possibly indicating differences in evaluation criteria.

The actual healing scores for 19 wounds (the wound image for each patient's initial visit is the reference image) given by 3 clinicians (for the case where both the wound images and quantitative wound data are presented) and by our algorithm were shown in Figure 6.4. From this figure, we can see that the scores given by our algorithm were a reliable quantitative indicator of the wound healing trend. Overall our algorithm provided a promising quantitative assessment that approximated well the average score from three clinicians. In Figure 6.5 and 6.6, we presented the wound area and tissue classification results for two patients, resulting in two time sequences of foot ulcer images. The corresponding actual wound area, areas of different wound tissues and the healing scores

for the two patients were shown in Table 6.4 and 6.5. We can see the basic trend is: the larger the weighted wound area, the smaller the healing score. However, there is one exception in Table 6.4. The Image 5 has more yellow tissues and less red tissues. However, it has a higher score. This is because we apply the *G* parameter tuning, as shown in eq. (6.2), to further fit the scores given by our healing score method with the results provided by wound clinicians.

Table 6.1 Krippendorff's Alpha Coefficients for the inter-rater consistency measurements, both for wound image only (top values) and for wound image plus quantitative wound data (bottom values)

	Clinician 1	*Clinician 2*	*Clinician 3*
Clinician 1		*0.85*	*0.42*
		0.80	*0.46*
Clinician 2	*0.85*		*0.46*
	0.80		*0.63*
Clinician 3	*0.42*	*0.46*	
	0.46	*0.63*	

Table 6.2 Krippendorff's Alpha Coefficients for the intra-rater data impact measurements

Clinician Number	*1*	*2*	*3*
Intra-rater data impact coefficients	*0.81*	*0.80*	*0.86*

Table 6.3 Krippendorff's alpha coefficients for the clinical validity measurements, both for wound image only (top values) and for wound image plus quantitative wound data (bottom values)

Clinician Number	*1*	*2*	*3*
Clinical validity coefficients	*0.73*	*0.68*	*0.42*
	0.66	*0.81*	*0.46*

Table 6.4 Wound assessment results for patient 1 (area unit: mm^2). For better accuracy, 2 significant digits for the healing scores are displayed.

	Image 1	*Image 2*	*Image 3*	*Image 4*	*Image 5*
Healing score	*5 (ref)*	*7.0*	*7.4*	*6.9*	*7.3*
Wound area	*928*	*403*	*293*	*279*	*332*
Red area	*751*	*353*	*283*	*215*	*126*
Yellow area	*158*	*39*	*10*	*43*	*184*
Black area	*19*	*11*	*0*	*21*	*22*

Table 6.5 Wound assessment results for patient 2 (area unit: mm^2)

	Image 1	*Image 2*	*Image 3*
Healing score	*5 (Ref)*	*3.9*	*5.6*
Wound area	*249*	*329*	*253*
Red area	*203*	*247*	*232*
Yellow area	*11*	*82*	*21*
Black area	*38*	*0*	*0*

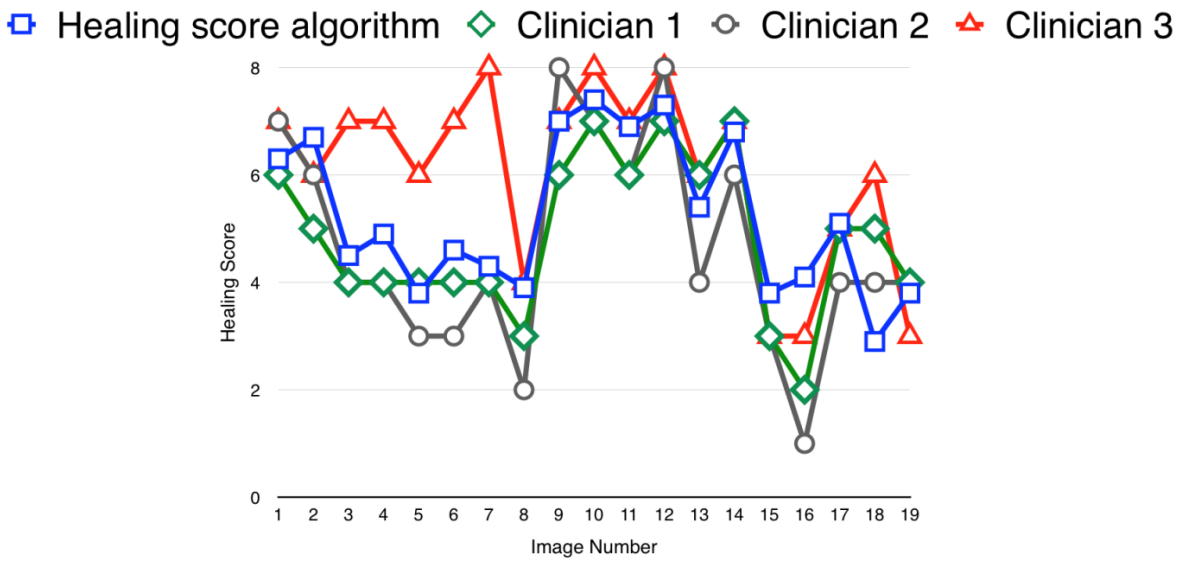


Figure 6.4 Healing scores by the three raters (green, black and red for Clinicians 1, 2 and 3, respectively), and by the healing score algorithm (blue), for the case where both wound images and wound data are presented.

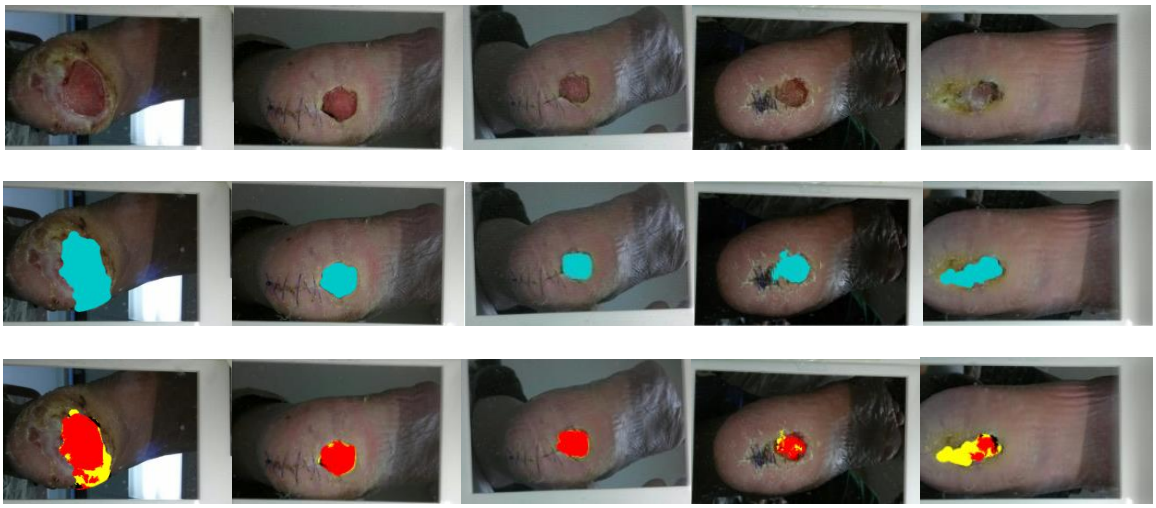


Figure 6.5 Wound area and tissue classification results for patient 1; Row 1: original foot ulcer images, Row 2: Wound boundary determination results, Row 3: Tissue classification results

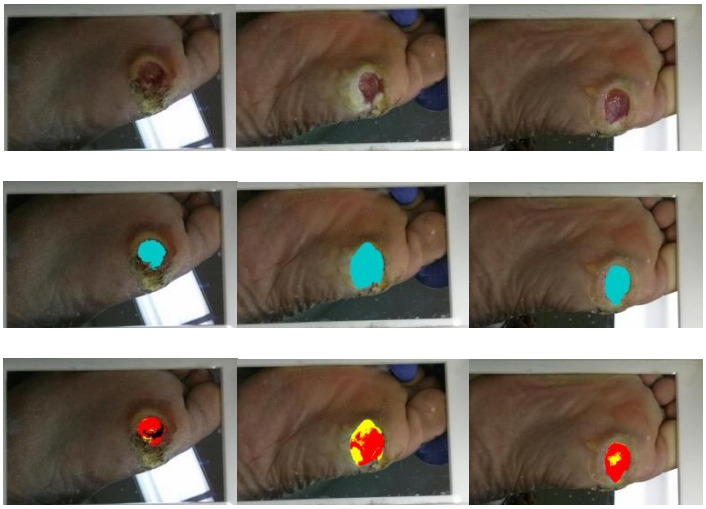


Figure 6.6 Wound area and tissue classification results for patient 2; Row 1: original foot ulcer images, Row 2: Wound boundary determination results, Row 3: Tissue classification results

6.3 Collaborative Wound Image Analysis System

The purpose of the collaborative wound image analysis system based on the smartphone and the laptop is to greatly increase the often insufficient computational resources of just the smartphones. In our work, we managed to make the less efficient wound analysis algorithm implementation work on Nexus 5, a powerful Android smartphones. However, a collaborative system is necessary for implementing the more advanced and robust wound recognition algorithms, such as the CRF based algorithms, which would be used by clinicians rather than patients. The reasons are stated as following. First, although the capability of wound image analysis on the smartphone is verified, we still want the wound assessment system can run as efficiently as possible given sufficient computer resources, and laptops or PCs are generally more powerful than smartphones. Second, the wound boundary determination algorithms introduced in Chapter 3 and 4 only work sufficiently well on wound images captured under strictly controlled acquisition conditions. Hence, we need the CRF based algorithm introduced in Chapter 5 to provide more robust performance on images taken under more relaxed imaging conditions. The efficiency analysis results in Chapter 5 show that the CRF based method can only be implemented efficiently based on PC due to its complexity. Finally, the wound analysis, especially the trend analysis results, can be illustrated in a more user-friendly way

graphically. Besides, we can manage the patients' demographics, wound images, image analysis results in a more intelligent database system.

6.3.1 System Structure

The basic structure for this collaborative system is shown as Figure 6.7. Since there will be a client (the smartphone) and a server (the laptop) to work in a cooperative way, the communication scheme is rather significant. In this system, we chose the communication scheme between the smart phone and laptop in the application layer to be based on http protocol rather than the common method manipulating the transport layer by socket programming based on TCP/IP protocol. The reason is that the operation in the highest layer (application layer) in the TCP/IP network protocol model will be more generalized and can be applied to different settings in lower layers of this network model.

The smartphone will play the role of client in the communication scheme. It will accomplish the following task sequentially: 1) take the picture of the wound and save on the specified directory on the SD card; 2) make request to the server and send the JPEG file of wound image to the laptop by function "post()" in android http library and 3) receive the analyzed image file sent by laptop and display it on the screen. The laptop will be viewed as the server party, which will 1) listen to the request of the client and get the image file sent by the client by "dopost()" function, 2) accomplish the wound image process to he received wound image and 3) send the JPEG file of the processed image back to the smartphone (client) as a response.

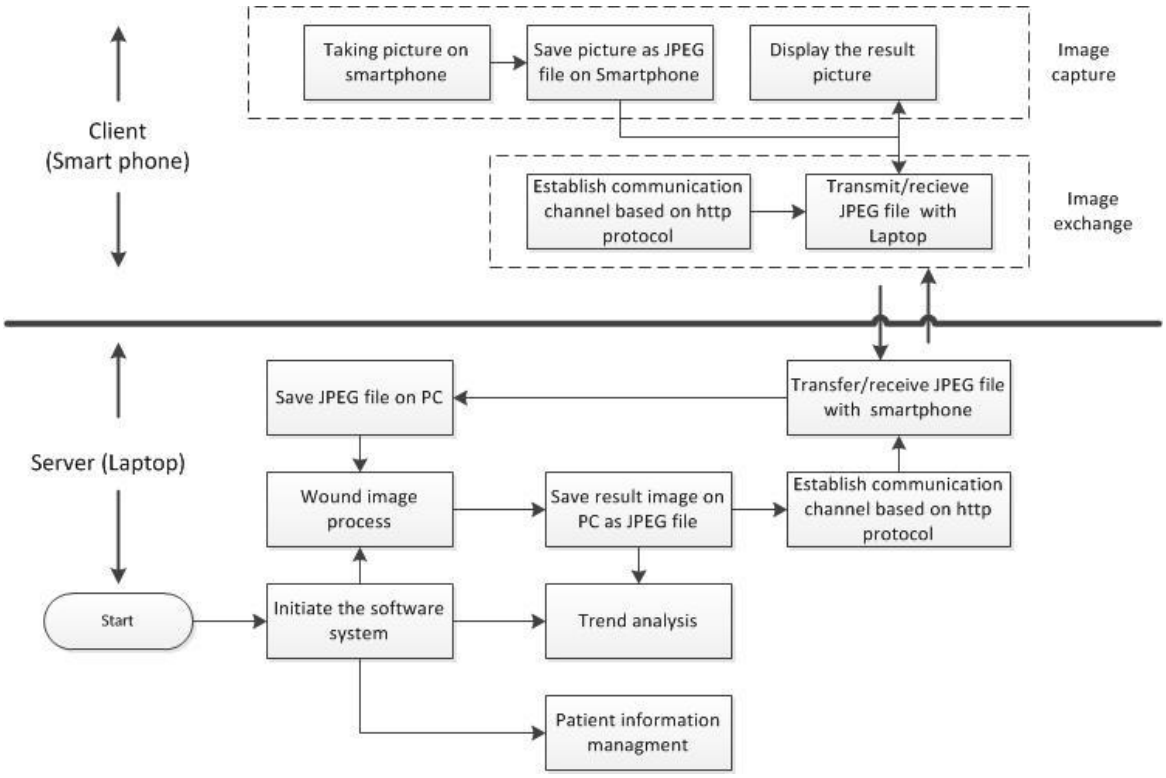


Figure 6.7 smartphone & laptop cooperation system

Based on the above description, one application should be designed to run on the client (smartphone) and another application should be designed to run on the server. On the smartphone, the task is easy to accomplish by using the apache http library included in the Android SDK. However, on the laptop, it is a little more difficult to establish a servlet application. We could run the servlet application in Tomcat, which is an open source servlet container developed by the Apache Software Foundation (ASF). The server program on the laptop will keep running and listening for the request from the client process on smartphones. In this case, any smartphone getting access to the Wi-Fi is able to send requests to the server. However, considering the integration of the servlet application into a clinical wound management system, we need a servlet container completely programmed in JAVA in order to pack the entire servlet application and container into an independent executable jar file. Hence, we use the Jetty which is a small, fast, embeddable web server and servlet container developed in JAVA. Moreover, the Jetty server can be started using API from the main thread in JAVA application.

As mentioned above, the servlet will be integrated into a wound image management system for clinical research purpose in wound clinics. Besides the wound image transmission and analysis, the software system also manages all registered patients' personal information and wound images in databases constructed by MySQL. This will allows clinicians to perform trend analysis for a certain patient over a time sequence of wound images. This system will be introduced in detail in the following section.

6.3.2 Work Flow and Main Functions

In this section, we will introduce our wound management system from the user's perspective. Once a user (usually a clinician in the wound clinic) double-clicks the application icon on the desktop of the laptop, the welcome page will be shown as in Figure 6.8. Three buttons are located on this page including "New Patients", "Existing Patients" and "Label Image". This system is going to be demonstrated begun from these three entries.

- *Registering a new patient*

When a new patient comes to the wound clinic center, the clinician needs to register this patient into our database and click the button "new patient" on the welcome page to initiate the new patient registration page shown in Figure 6.9. First, the clinician should click the "Assign" button and the system will automatically assigned an ID number to this new patient. Then the required patient's information must be input including the first name and last name. Other physical information, such as gender, age, weight, height and Body Mass Index (BMI), are optional. Note that the BMI is calculated by the system if the "Calculate" is clicked using the following formula [80].

$$BMI = \frac{mass(lb)}{(height(in))^2} \times 703 \qquad \textit{Equation 6.4}$$

After all needed information is entered (note that the sample information in Figure 6.9 is faked), the clinician will click the "Confirm" button to finish the registration process and save the patient's record. If the "Clean" button is clicked, all the input information will be erased and a new registration has begun.

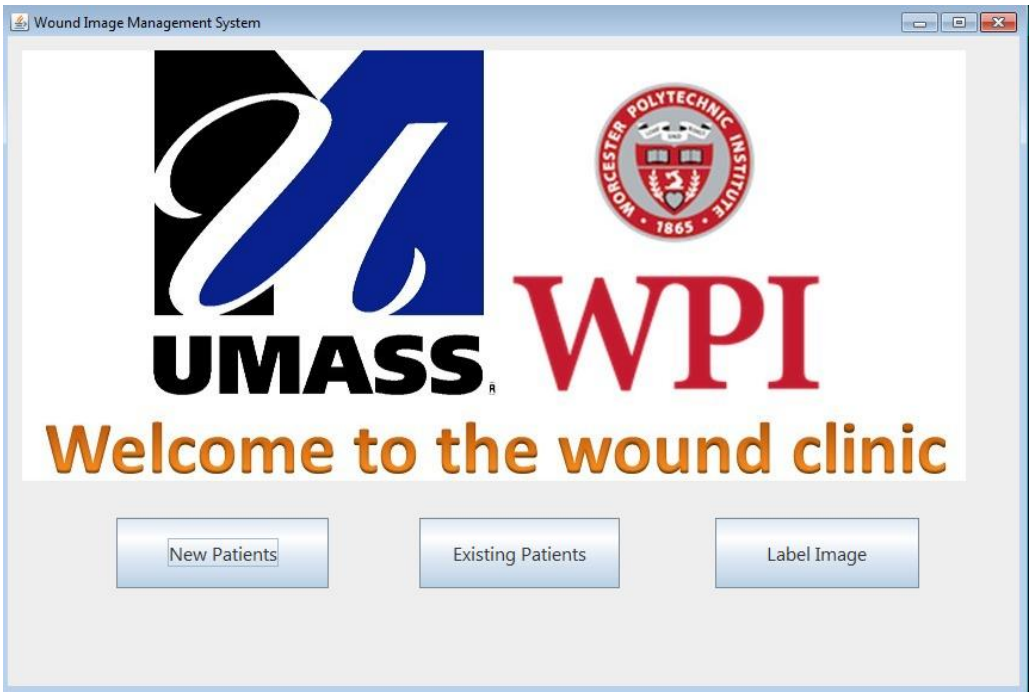


Figure 6.8 Welcome page

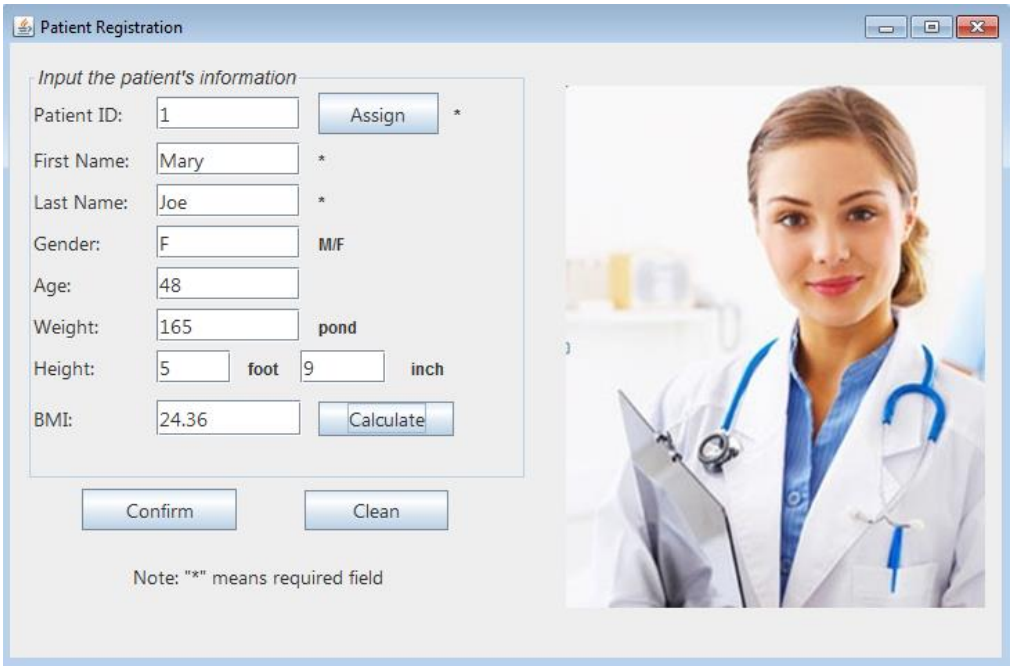


Figure 6.9 New patient registration page

- *If you want access a previously registered patient*

If a registered patient comes to the wound clinic to have his foot photographed, the clinician has to set up both the smartphone image capture app and the wound management system on the laptop. On the welcome page in Figure 6.8, the user will click

the “Existing patients” button which will takes him/her to the registered patient menu page for previously registered patients’ management as shown in Figure 6.10. On the menu page, the user inputs the patient’s ID number and which foot is going to be photographed (L or R) in the textboxes. Then the user clicks the “GO” button to enter the wound image management page as shown in Figure 6.5. On the smartphone side, the user runs the “phone client” app and follows the instructions to take a picture of the wound. Then the wound image will be sent to the laptop and automatically analyzed.

Once the image is analyzed, the captured wound image in its original and its analyzed form will be shown under the “current” tag. If there are any previous wound records associated with this patient, the five latest recorded wounds in their original form and in their analyzed form will be shown as well under the corresponding date tags. Moreover, the user can click any analyzed image to perform the trend analysis including comparing the current analysis result with the “one week earlier” and “five week earlier” records respectively. The trend analysis result is displayed in Figure 6.6. The comparison is performed in terms of the wound area size and size of different wound tissues.

If planning to search for a registered patient’s wound records on the wound image management page, the user will input the patient’ ID number in the textbox on the “Find” menu panel, which is on the right-top area of the page in Figure 6.11, and click the “GO” button. Then the six latest records about this patient will be shown under the corresponding date tags. Note that in order to protect the subjects’ privacy, the wound records will be stop being displayed from this page after 1 minute if there is no further operation.

Back to the menu page in Figure 6.10, if planning to view the information about a previously registered patient, the user should click the “View” button on the menu page. A table containing the personal information for all registered patients in our database will be shown in a new window as presented in Figure 6.12-6.13.

Figure 6.10Previously registered patients menu

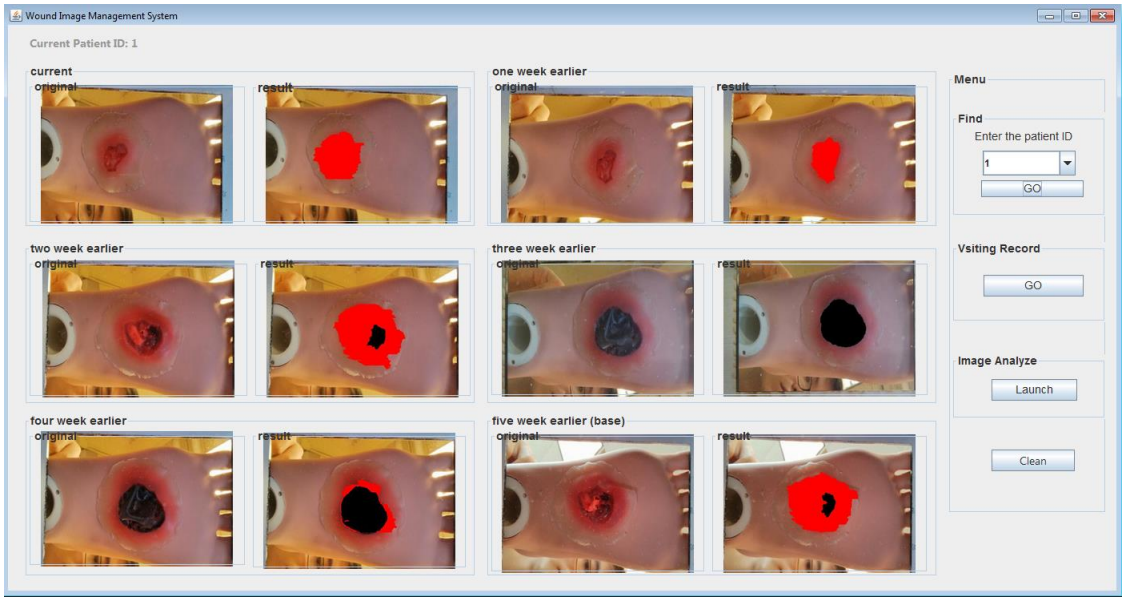


Figure 6.11 Wound image management page

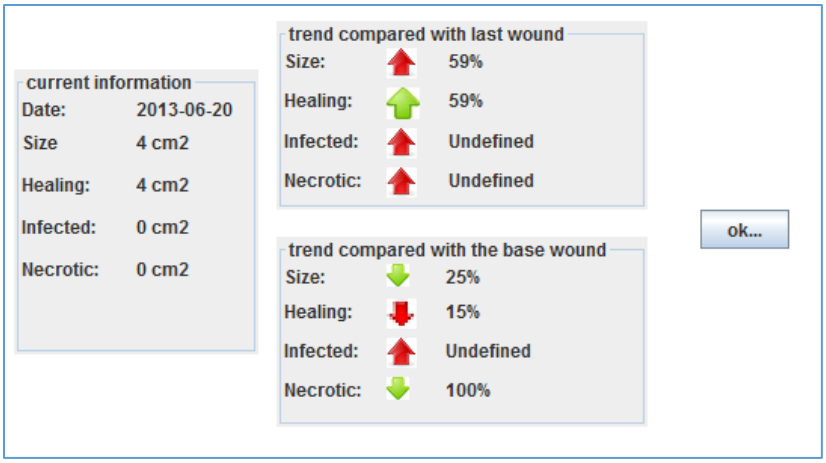


Figure 6.12 Trend analysis result

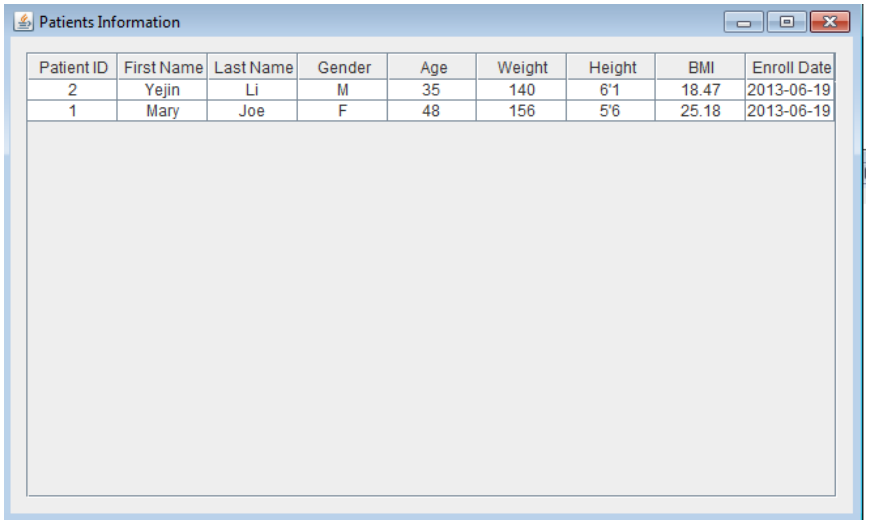


Figure 6.13 Patients' information view page

6.4 Conclusion

This chapter proposed a wound healing status assessment method, which consists of a color based tissue segmentation module and a trend analysis mechanism based on healing score. The healing score is a quantitative measure to evaluate the healing status by comparing the current wound measurements to the initial record. Moreover, to facilitate the patients' tacking at the wound clinic, a collaborative system between the smartphone and PC is designed. The clinical assessment results of our healing score algorithm shows very good agreement with clinicians' scores and thus indicates its strong potential for

automated quantitative wound healing assessment. However, more validation data is needed to further evaluate our algorithm.

The collaborative system has been practically used for wound tracking in the UMass-Memorial Health Center Wound Clinic in Worcester, MA. The evaluations have shown that our system did provide a convenient software platform to systematically manage the wound records for patients at wound clinics.

One major concern is the possibility of microbial contamination of the image capture box by the users or the environment. So far, we addressed this problem by wiping the surface of the box with an anti-microbial wipe after each use. A better solution may be a disposable contamination barrier, which will cover the slanted surface of the box except the openings. This will avoid the patient's foot directly touching the surface of the image capture box.

Finally, telehealth is an obvious extension to the collaborative system whereby clinicians can remotely access the wound image and the analysis results. Hence, a database will be constructed on a possibly cloud-based server to store the wound data for patients. The data privacy will be another concern. We plan to only allow the registered clinicians to have access to our database. In addition, the data will be encoded properly during the transmission.

Chapter 7

Conclusion

In this work we proposed a novel diabetic foot ulcer image assessment system to quantitatively measure the wound area size, tissue composition and healing status based on foot ulcer images captured in high quality. The major purpose of our system is to design and evaluate a convenient foot ulcer assessment system (either used by patients themselves or by clinicians at the wound clinic) based on computational devices (PC or mobile devices). Four major tasks have been carried out as follows: 1) a low-cost, easy-to-use image capture system consisting of an Android smartphone and an image capture box is designed to facilitate the high-quality foot ulcer imaging especially for elder patients with limited mobility; 2) five different wound boundary determination approaches have been proposed for applications in different context; 3) a wound healing assessment method including tissue classification based on RYB color model, healing status assessment based on well-designed healing score mechanism and 4) a collaborative wound management software for clinical use. The complete foot ulcer assessment pipeline was evaluated on two different image databases: 1) image database of Moulage wound simulations and 2) real wound images collected at wound clinic in UMass Medical School.

For better comparison, we summarize the quantitative measurement results of the performance and time performance (evaluated on PC) for all 5 wound boundary determination methods proposed in our work in the following table. According to the results presented in Table 7.1, we see that machine learning based approaches obviously provided better wound recognition performance than the non-machine learning based approaches. However, in terms of the time performance and implementation complexity,

the non-machine learning based wound boundary determination approaches performed better than machine learning based approaches. To be exact, with GPU based optimization, the run time for the entire wound assessment, including non-machine learning based wound boundary determination, color segmentation and healing status assessment, is about 6 and 16 seconds based on PC and smartphone, respectively. The two machine learning based approaches overcome the shortcomings of the first approach by enabling the self-adaptivity of wound recognition models when more and more clinical labeled data were available for training. The SVM classifier based method provided a good tradeoff: promising performance (specificity: > 94%, sensitivity: > 73%) and real-time performance (< 24 seconds in average on smartphones, < 11 seconds in average on PCs). Furthermore, the CRF based approach provided more robust wound determination performance than SVM based one when dealing with images captured in variant conditions (specificity: > 95%, sensitivity: > 77%). Especially, the CRF model involving with the unary potentials defined on segment based grid performed most consistently on images captured at different illuminations and ranges (or scales). However, the CRF based method is characterized by high computational requirements in its implementation and we are more inclined to apply it to the collaborative system where the wound assessment is running on PCs or servers with more powerful computational resources.

There are a number of directions for future research work. First, to further stabilize the wound boundary determination robustness, we might need to expand the diversity of the real wound image database in terms of wound type, shape, color composition, surrounding tissues (calculus or not), skin color and texture. Moreover, the image calibration for our system was accomplished by using the image capture box. To broaden the application context (without the assistance of image capture box when imaging the wounds), image calibration methods depending on standard color patches placed on the corner of patients' limbs need to be developed. In addition, although the clinical assessment results of our healing score algorithm shows good agreement with clinicians' scores and thus indicates its strong potential for automated quantitative wound healing assessment, more validation data is needed to further evaluate our algorithm.

Table 7.1 Summary of the wound recognition performance and time performance for five wound boundary determination methods proposed in our work; Approach 1: level set based approach; Approach 2: mean shift based approach; Approach 3: improved mean shift based approach; Approach 4: two-stage cascade SVM classifier based approach; Approach 5: hierarchical CRF based approach

Category	*Non-machine learning based*			*Machine learning based*	
Approach ID	*1*	*2*	*3*	*4*	*5*
Sensitivity	*48.7%*	*56.7%*	*63.6%*	*73.3%*	*76.9%*
Specificity	*83.1%*	*89.6%*	*92.3%*	*94.6%*	*95.5%*
Computation time (PC)	*6.4*	*5.8*	*7.0*	*10.4*	*62.4*
Computation time (smartphone)	*N/A*	*18.9*	*20.2*	*23.1*	*N/A*
Model training needed?	*No*	*No*	*No*	*Yes*	*Yes*

Another direction could be to improve the efficiency of the CRF based approach. The results presented in Chapter 5 show that the computational time of this approach is nearly 60 seconds even when implemented on powerful PCs. Due to the iterative nature, the potential evaluation is the most computationally expensive part. As described in Chapter 5, the iteration number has to be larger than 3000 to acquire near-optimal results. Hence, according to my best knowledge, the best option to reduce the computational burden is to remove the extraction of less effective features. This may require more detailed feature effectiveness evaluation.

Besides, instead of the measurements used in this work such as the area size and different wound tissue size, the depth is also a significant indicator for healing assessment. However, this measurement can only be extracted by using 3D imaging techniques or indirect 3D reconstructing based on multi-view model.

Finally, our wound assessment system can also be applied for monitoring of other types of wound (such as surgical wound) or skin disease (such as skin cancer, also known as melanoma) monitoring. All 5 boundary determination approaches might be used for these new tasks with only slight modification. However, the validity of the RYB model may need to be determined for such new applications. New models may also need to be constructed based on the specific nature of different skin problems.

Bibliography

[1] S. A. Bus, C. E. Hazengerg, M. Klein, and J. G. Van Baal, "Assessment of foot disease in the home environment of diabetic patients using a new photographic foot imaging device," *J. Med. Eng. Technol.*, vol. 34, no. 1, pp. 43–50, 2010.

[2] K. M. Buckley, L. K. Adelson, and J. G. Agazio, "Reducing the risks of wound consultation: adding digital images to verbal reports," *Wound Ostomy Cont. Nurs.*, vol. 36, no. 2, pp. 163–170, 2009.

[3] V. Falanga, "The chronic wound: Impaired healing and solutions in the context of wound bed preparation," *Blood Cells, Mol. Dis.*, vol. 32, no. 1, pp. 88–94, 2004.

[4] C. T. Hess and K. R.S., "Orchestrating wound healing: assessing and preparing the wound bed," *Adv. Ski. Wound Care*, vol. 16, no. 5, pp. 246–257, 2003.

[5] R. S. Rees and N. Bashshur, "The effects of TeleWound management on use of service and financial outcomes," *Telemed J E Heal.*, vol. 13, no. 6, pp. 663–674, 2007.

[6] N. I. of Health, "NIH's National Diabetes Infromation Clearing House," 2011. .

[7] F. Veredas, H. Mesa, and L. Morente, "Binary tissue classification on wound images with neural networks and bayesian classifiers," *IEEE Trans. Med. Imaging*, vol. 29, no. 2, pp. 410–427, 2010.

[8] C. K. Sen, G. M. Gordillo, S. Roy, R. Kirsner, L. Lambert, T. K. Hunt, F. Gottrup, G. C. Gurtner, and M. T. Longaker, "Human skin wounds: a major and snowballing threat to public health and the economy," *Wound Repair Regen.*, vol. 17, no. 6, pp. 763–771, 2009.

[9] C. S. J. Cuddigan, E. Ayello, "Pressure ulcers in America: prevalence, incidence,

and implications for the future. An executive summary of the National Pressure Ulcer Advisory Panel monograph," *Adv Ski. Wound Care*, vol. 14, no. 4, pp. 208–215, 2001.

[10] M. F. Ahmad Fauzi, I. Khansa, K. Catignani, G. Gordillo, C. K. Sen, and M. N. Gurcan, "Computerized segmentation and measurement of chronic wound images," *Comput. Biol. Med.*, vol. 60, pp. 74–85, 2015.

[11] D. Beeckman, L. Schoonhoven, J. Fletcher, K. Furtado, L. Gunningberg, H. Heyman, C. Lindholm, L. Paguay, J. Verdu, and T. Defloor, "EPUAP classification system for pressure ulcers: European reliability study," *Adv. Nurs.*, vol. 60, no. 6, pp. 682–691, 2007.

[12] M. E. Wendelken, W. T. Berg, P. Lichtenstein, L. Markowitz, C. Comfort, and O. M. Alvarez, "Wounds measured from digital photographs using photodigital planimetry software: validation and rater reliability," *Wounds*, vol. 23, no. 9, pp. 267–275, 2011.

[13] P. Plassmann and T. D. Jones, "MAVIS: a non-invasive instrument to measure area and volume of wounds. Measurement of Area and Volume Instrument System," *Med. Eng. Phys.*, vol. 20, no. 5, pp. 332–338, 1998.

[14] A. Malian, A. Azizi, F. Heyvel, and M. Zolfaghari, "Development of a robust photogrammetric metrology system for monitoring the healing of bedsores," *Photogramm. Rec.*, vol. 20, no. 111, pp. 241–273, 2005.

[15] P. Foltynski, P. Ladyzynski, K. Migalska-Musial, S. Sabalinska, A. Ciechanowska, and J. Wojcicki, "A new imaging and data transmitting device for telemonitoring of diabetic foot syndrome patients," *Diabetes Techonol Ther.*, vol. 13, no. 8, pp. 861–867, 2011.

[16] T. D. Jones and P. Plassmann, "An active contour model for measuring the area of leg ulcers.," *IEEE Trans. Med. Imaging*, vol. 19, no. 12, pp. 1202–1210, 2000.

[17] Plassmann, P, Jones, T.D., "Improved active contour models with application to measurement of leg ulcers," *SPIE Electron. Imaging*, vol. 12, no. 2, pp. 317–326, 2003.

[18] L. Wang, P. C. Pedersen, D. Strong, B. Tulu, and E. Agu, “Wound image analysis system for diabetics,” *Int. Soc. Opt. Photonics*, vol. 8669, p. 866924, 2013.

[19] K. Wantanajittikul, “Automatic Segmentation and Degree Identification in Burn Color Images,” pp. 169–173, 2011.

[20] L. Wang, P. Pedersen, D. Strong, B. Tulu, E. Agu, and Q. He, “Smartphone-Based Wound Assessment System for Diabetic Patients,” vol. 62, no. 2, pp. 477–488, 2013.

[21] M. Kolesnik and A. Fexa, “Multi Dimensional Color Histograms for Segmentation of Wounds in Images,” in *Int. Conf. Image Anal. Recognit*, 2005, pp. 1014–1022.

[22] M. Kolesnik and A. Fexa, “Segmentation of Wounds in the Combined Color-Texture Feature Space,” in *SPIE Medical imaging*, 2004, pp. 549–556.

[23] H. Wannous, Y. Lucas, and S. Treuillet, “Enhanced assessment of the wound-healing process by accurate multiview tissue classification,” *IEEE Trans. Med. Imaging*, vol. 30, no. 2, pp. 315–326, 2011.

[24] B. Song and A. Sacan, “Automated wound identification system based on image segmentation and Artificial Neural Networks,” *Bioinforma. Biomed. (BIBM), …*, 2012.

[25] D. Cukjati, R. Karba, S. Rebersek, and D. Miklavcic, “Measures of wound healing rate,” *Proc. 22nd Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (Cat. No.00CH37143)*, vol. 1, pp. 765–768, 2000.

[26] C. P. Loizou, T. Kasparis, O. Mitsi, and M. Polyviou, “Evaluation of Wound Healing Process Based on Texture Analysis,” in *IEEE Intel. Conf. on BIBE*, 2012, no. November, pp. 11–13.

[27] M. Burns, J. Enderle, E. Rosow, and Q. Z. Q. Zhu, “Development of a wound assessment system for quantitative chronic wound monitoring,” in *Proceedings of the IEEE 28th Annual Northeast Bioengineering Conference (IEEE Cat. No.02CH37342)*, 2002.

[28] “PictZar Digital Planimetry Software.” .

[29] D. Filko, D. Antonić, and D. Huljev, “Wita - Application for wound analysis and management,” in *12th IEEE International Conference on e-Health Networking, Application and Services, Healthcom 2010*, 2010, pp. 1–6.

[30] S. a. Weber, N. Watermann, J. Jossinet, J. A. Byrne, J. Chantrey, S. Alam, K. So, J. Bush, S. O’Kane, and E. T. McAdams, “Remote wound monitoring of chronic ulcers,” *IEEE Trans. Inf. Technol. Biomed.*, vol. 14, no. 2, pp. 371–377, 2010.

[31] “About MelaFind.” .

[32] “Silhouette: the Future of Wound Assessment.” .

[33] “MOWA: Mobile Wound Analyzer.” .

[34] J. Arnqvist, L. Hellgrenz, and J. Vincentz, “Semi-automatic classification of secondary healing ulcers in multispectral images,” in *Pattern Recognition Letters*, 1988, pp. 459–461.

[35] D. Kransner, “Wound care how to use the red-yellow-black system,” *Am. J. Nurs.*, vol. 95, no. 5, pp. 44–47, 2011.

[36] J. A. Hartigan and M. A. Wong, “A K-mean clustering algorithm,” *R. Stat. Soc.*, vol. 28, no. 1, pp. 100–108, 1979.

[37] R. Achanta, A. Shaji, and K. Smith, “SLIC Superpixels Compared to State-of-the-Art Superpixel Methods,” *Pattern Anal. …*, vol. 34, no. 11, pp. 2274–2281, 2012.

[38] Y. H. Wang, “Tutorial: Image Segmentation,” 2010.

[39] R. Adams and L. Bischof, “Seeded region growing,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 6, pp. 641–647, 1994.

[40] Z. Lin, J. Jin, and H. Talbot, “Unseeded region growing for 3D image segmentation,” in *Selected papers from the Pan-Sydney workshop on Visualisation-Volume 2*, 2000, pp. 31–37.

[41] Y. Deng and B. S. Manjunath, “Unsupervised segmentation of color-texture regions in images and video,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 8, pp. 800–810, 2001.

[42] J. Ding, C. Kuo, and W. Hong, “An Efficient image segmentation Technique by Fast Scanning and Adaptive Merging,” in *GVGIP*, 2009.

[43] Y. Zhao and G. Karypis, “Comparison of Agglomerative and Partitional Document Clustering Algorithms,” *Perform. Comput.*, pp. 1–11, 2002.

[44] D. Comaniciu and P. Meer, “Mean shift: a robust approach toward feature space analysis,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 1–37, 2002.

[45] A. Vedaldi and S. Soatto, “Quick shift and kernel methods for mode seeking,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5305 LNCS, no. PART 4, pp. 705–718, 2008.

[46] L. Vincent and P. Soille, “Watersheds in digital spaces: An efficient algorithm based on immersion simulations,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 6. pp. 583–598, 1991.

[47] Y. Boykov and G. Funka-Lea, “Graph cuts and efficient N-D image segmentation,” *Int. J. Comput. Vis.*, vol. 70, no. 2, pp. 109–131, 2006.

[48] Y. Boykov, Y. Boykov, O. Veksler, O. Veksler, R. Zabih, and R. Zabih, “Fast Approximate Energy Minimization via Graph Cuts,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222– 1239, 2001.

[49] Y. Cheng, “Mean shift, mode seeking, and clustering,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 8, pp. 790–799, 1995.

[50] W. Zucchini, “Applied smoothing techniques: part 1: Kernel density estimation,” 2003.

[51] J. C. Tilton, Y. Tarabalka, P. M. Montesano, and E. Gofman, “Best merge region-growing segmentation with integrated nonadjacent region object aggregation,” *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 11, pp. 4454–4467, 2012.

[52] C. M. Christoudias, B. Georgescu, and P. Meer, “Synergism in low level vision,” *Object Recognit. Support. by user Interact. Serv. Robot.*, vol. 4, no. 2, pp. 150–155, 2002.

[53] A. Duarte, Á. Sánchez, F. Fernández, and A. S. Montemayor, “Improving image

segmentation quality through effective region merging using a hierarchical social metaheuristic," *Pattern Recognit. Lett.*, vol. 27, no. 11, pp. 1239–1251, 2006.

[54] R. Lidl and G. Pilz, *Applied Abstract Algebra, 2nd Edition*. 1997.

[55] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC Superpixels," *... Fédéral Lausssanne ...*, no. June, p. 15, 2010.

[56] Y. D. Y. Deng, C. Kenney, M. S. Moore, and B. S. Manjunath, "Peer group filtering and perceptual color image quantization," *ISCAS99 Proceedings of the 1999 IEEE International Symposium on Circuits and Systems VLSI Cat No99CH36349*, vol. 4. pp. 21–24, 1999.

[57] M. Welling, "Fisher Linear Discriminant Analysis," *Science (80-.).*, vol. 1, no. 2, pp. 1–3, 2009.

[58] M. D. Levine and a M. Nazif, "Dynamic measurement of computer generated image segmentations.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 7, no. 2, pp. 155–164, 1985.

[59] J. A. Sethian, "Level Set Methods and Fast Marching Methods:," *Journal of Computing and Information Technology*, vol. 11. pp. 79–79, 2003.

[60] C. Li, C. Xu, C. Gui, and M. D. Fox, "Distance regularized level set evolution and its application to image segmentation," *IEEE Trans. Image Process.*, vol. 19, no. 12, pp. 3243–3254, 2010.

[61] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic Active Contours," *Int. J. Comput. Vis.*, vol. 22, no. 1, pp. 61–79, 1997.

[62] V. Caselles, F. Catté, T. Coll, and F. Dibos, "A geometric model for active contours in image processing," *Numedsche Math.*, vol. 66, pp. 1–31, 1993.

[63] R. Malladi, J. A. Sethian, and B. C. Vemuri, "Evolutionary fronts for topology-independent shape modeling and recovery," in *European Conference on Computer Vision*, vol. 800, 1994, pp. 1–13.

[64] S. Osher and J. a. Sethian, "Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations," *J. Comput. Phys.*, vol. 79, no.

1, pp. 12–49, 1988.

[65] T. Van Luong, N. Melab, and E. G. Talbi, “GPU computing for parallel local search metaheuristic algorithms,” *IEEE Trans. Comput.*, vol. 62, pp. 173–185, 2013.

[66] H. Samet and M. Tamminen, “Efficient Component Labeling of Images of Arbitrary Dimension Represented By Linear Bintrees.,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, no. 4, pp. 579–586, 1988.

[67] D. Pascale, “RGB Coordinates of the Macbeth Color Checker,” *Montr. Quebec, Canada, BabelColor Company. …*, 2006.

[68] Q. Inc, “Adreno graphics processing units,” *https://developer.qualcomm.com/mobile-development/maximize-hardware/mobile-gaming-graphics-adreno/adreno-gpu*. .

[69] Google, “Renderscript,” *http://developer.android.com/guide/topics/renderscript/compute.html#overview*. .

[70] B. Varga and K. Karacs, “High-resolution image segmentation using fully parallel mean shift,” *EURASIP J. Adv. Signal Process.*, vol. 2011, no. 1, p. 111, 2011.

[71] B. Fulkerson and S. Soatto, “Really quick shift: Image segmentation on a GPU,” in *Workshop on Computer Vision using GPUs*, 2010.

[72] F. Chang, C. J. Chen, and C. J. Lu, “A linear-time component-labeling algorithm using contour tracing technique,” *Comput. Vis. Image Underst.*, vol. 93, no. 2, pp. 206–220, 2004.

[73] C. Harris and M. Stephens, “A Combined Corner and Edge Detector,” *Procedings Alvey Vis. Conf. 1988*, pp. 147–151, 1988.

[74] K. Mikolajczyk and C. Schmid, “Scale & affine invariant interest point detectors,” *Int. J. Comput. Vis.*, vol. 60, no. 1, pp. 63–86, 2004.

[75] B. W. W. Matthews, “Comparison of the predicted and observed secondary structure of T4 phage lysozyme,” *Biochim. Biophys. Acta - Protein Struct.*, vol. 405, no. 2, pp. 442–451, 1975.

[76] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, “Machine learning: A review of classification and combining techniques,” *Artif. Intell. Rev.*, vol. 26, no. 3, pp. 159–190, 2006.

[77] B. Fulkerson, A. Vedaldi, and S. Soatto, “Class segmentation and object localization with superpixel neighborhoods,” in *Computer Vision, 2009 IEEE 12th International Conference on*, 2009.

[78] B. S. Manjunath, J. R. Ohm, V. V. Vasudevan, and A. Yamada, “Color and texture descriptors,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 6, pp. 703–715, 2001.

[79] P. Berris, “Acquisition of skin wound images and measurement of wound healing rate and status using color image processing,” 2000.

[80] M. Galushka, H. Zheng, D. Patterson, and B. L. B. L, “Case-based tissue classification for monitoring leg ulcer healing,” in *18th IEEE Symposium on Computer-Based Medical Systems (CBMS’05)*, 2005.

[81] R. Haralick, K. Shanmugan, and I. Dinstein, “Textural features for image classification,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 3. pp. 610–621, 1973.

[82] T. Fawcett, “An introduction to ROC analysis,” *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, 2006.

[83] R. Muñiz, “Novel Techniques for Color Texture Classification,” in *Procedings of the International Conference on Image Processing, Computer vision and Pattern Recognition*, 2006.

[84] W. Y. Ma, Y. Deng, and B. S. Manjunath, “Tools for texture / color based search of images,” in *SPIE Int. Conf. Human Vision and Electronic Imaging*, 1997, pp. 496–507.

[85] V. Vezhnevets, “A Survey on Pixel-Based Skin Color Detection Techniques,” *Cybernetics*, vol. 85, no. 0896–6273 SB - IM, pp. 85–92, 2003.

[86] M. A. Filters, “Moving Average Filters,” *Econ. Lett.*, vol. 37, pp. 277–284, 1991.

[87] R. Bahmanyar, S. Cui, and M. Datcu, “A Comparative Study of Bag-of-Words and Bag-of-Topics Models of EO Image Patches,” *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 6, pp. 1357–1361, 2015.

[88] T. Li, T. Mei, I. S. Kweon, and X. S. Hua, “Contextual bag-of-words for visual categorization,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 4, pp. 381–392, 2011.

[89] S. Keypoints and D. G. Lowe, “Distinctive Image Features from,” *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[90] A. Bosch, A. Zisserman, and X. Muñoz, “Scene classification via pLSA,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3954 LNCS, pp. 517–530, 2006.

[91] S. Lazebnik, C. Schmid, and J. Ponce, “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2, pp. 2169–2178, 2006.

[92] L. I. Smith, “A tutorial on Principal Components Analysis Introduction,” *Statistics (Ber).*, vol. 51, p. 52, 2002.

[93] D. Nouri, Y. Lucas, S. Treuillet, R. Jolivot, and F. Marzani, “Colour and multispectral imaging for wound healing evaluation in the context of a comparative preclinical study,” *Proc. SPIE, Med. Imaging Image Process.*, vol. 8669, no. 0033, pp. 866910–866923, 2013.

[94] C. J. C. Burges, “A Tutorial on Support Vector Machines for Pattern Recognition,” *Data Min. Knowl. Discov.*, vol. 2, pp. 121–167, 1998.

[95] D. Making, “Soft Margin SVM,” *Advances*, vol. 3, no. 4, pp. 1–4, 2003.

[96] V. Vapnik, *The Nature of Statistical Learning Theory*. 1995.

[97] and C.-J. L. Chih-Wei Hsu, Chih-Chung Chang, “A Practical Guide to Support Vector Classification,” *BJU Int.*, vol. 101, no. 1, pp. 1396–400, 2008.

[98] L. Ladický, C. Russell, P. Kohli, and P. H. S. Torr, “Associative hierarchical CRFs for object class image segmentation,” *Proc. IEEE Int. Conf. Comput. Vis.*, no. Iccv,

pp. 739–746, 2009.

[99] J. Shotton, J. Winn, C. Rother, and a Criminisi, "{TextonBoost} for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Appearance, Shape and Context," vol. 81, no. 1, pp. 2–23, 2009.

[100] P. Kr¨ahenb¨uhl, V. Koltun, and P. Krahenbuhl, "Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials," *Adv. Neural Inf. Process. Syst. 24 (Proceedings NIPS)*, no. 4, pp. 1–9, 2011.

[101] X. He, R. S. Zemel, and M. a. Carreira-Perpinan, "Multiscale conditional random fields for image labeling," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004, vol. 2, pp. 695 – 702.

[102] S. Z. Li, "Markov Random Field Modeling in Image Analysis," *Security*, vol. i, no. 4205, p. 371, 2009.

[103] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-6, no. 6, pp. 721–741, 1984.

[104] C. Sutton and A. McCallum, "An Introduction to Conditional Random Fields for Relational Learning," in *Introduction to Statistical Relational Learning*, 2007, pp. 93–126.

[105] S. K. S. Kumar and M. Hebert, "Discriminative random fields: a discriminative framework for contextual interaction in classification," *Proc. Ninth IEEE Int. Conf. Comput. Vis.*, pp. 1150–1157, 2003.

[106] D. Maxwell Chickering and D. Heckerman, "Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables," *Mach. Learn.*, vol. 29, no. 2, pp. 181–212, 1997.

[107] L. E. P. Xing, S. P. Schulam, and W. Wang, "13 : Variational Inference : Loopy Belief Propagation and Mean Field," no. 1, pp. 1–10, 2012.

[108] C. Sutton and a McCallum, "Piecewise training for undirected models," *Proc. 21*

Annu. Conf. Uncertain. Artif. Intellegence, pp. 568–575, 2005.

[109] L. Ladický, C. Russell, P. Kohli, and P. H. S. Torr, "Associative hierarchical random fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 6, pp. 1056–1077, 2014.

[110] J. Malik, S. Belongie, T. Leung, and J. Shi, "Contour and texture analysis for image segmentation," *Int. J. Comput. Vis.*, vol. 43, no. 1, pp. 7–27, 2001.

[111] J. Winn, A. Criminisi, and T. Minka, "Object categorization by learned universal visual dictionary," *Tenth IEEE Int. Conf. Comput. Vis. Vol. 1*, vol. 2, pp. 1800–1807, 2005.

[112] J. S. Chen, a Huertas, and G. Medioni, "Fast Convolution with Laplacian-of-Gaussian Masks.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 4, pp. 584–590, 1987.

[113] C. Elkan, "Using the Triangle Inequality to Accelerate k-Means," *Proc. Twent. Int. Conf. Mach. Learn.*, pp. 147–153, 2003.

[114] J. S. Beis and D. G. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997, pp. 1000–1006.

[115] A. Torralba, K. P. Murphy, and W. T. Freeman, "Sharing visual features for multiclass and multiview object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 5, pp. 854–869, 2007.

[116] R. E. Schapire and Y. Singer, "BoosTexter: A Boosting-based System for Text Categorization," *Mach. Learn.*, vol. 39, pp. 135–168, 2000.

[117] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. R. Stat. Soc. Ser. B*, vol. 39, no. 1, pp. 1–38, 1977.

[118] B. S. Manjunath, J. R. Ohm, V. V Vasudevan, and A. Yamada, "Color and texture descriptors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 6, pp. 703–715, 2001.

[119] H. Fujiyoshi, “Gradient-Based Feature Extraction -SIFT and HOG,” *Computer (Long. Beach. Calif).*, vol. 107, no. 206, pp. 211–224, 2007.

[120] D. Sen and S. K. Pal, “Gradient histogram: Thresholding in a region of interest for edge detection,” *Image Vis. Comput.*, vol. 28, no. 4, pp. 677–695, 2010.

[121] Y. Y. Boykov and M.-P. Jolly, “Interactive graph cuts for optimal boundary & region\nsegmentation of objects in N-D images,” *Proc. Eighth IEEE Int. Conf. Comput. Vision. ICCV 2001*, vol. 1, no. July 2001, pp. 105–112, 2001.

[122] Y. Weiss and W. T. Freeman, “On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs,” *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 736–744, 2001.

[123] V. Vineet, J. Warrell, and P. H. S. Torr, “Filter-based mean-field inference for random fields with higher-order terms and product label-spaces,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7576 LNCS, no. PART 5, pp. 31–44, 2012.

[124] A. F. Hayes and K. Krippendorff, “Answering the Call for a Standard Reliability Measure for Coding Data,” *Commun. Methods Meas.*, vol. 1, no. 1, pp. 77–89, 2007.