2013

Novel Techniques for Automated Dental Identification

Nourdin Al-sherif
West Virginia University

Novel Techniques for Automated Dental Identification

Nourdin Al-sherif

Dissertation submitted to the

Benjamin M. Statler College of Engineering and Mineral Resources
at
West Virginia University
in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy
in
Computer Engineering

Hany Ammar, Ph.D., Chair
Guodong Guo, Ph.D., Co-Chair
Xin Li, Ph.D.
Donald A. Adjeroh, Ph.D.
Matthew C. Valenti, Ph.D.
Richard Jurevic, DDS, Ph.D.

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2013

Keywords: Dental Identification, ADIS, Film Segmentation, Teeth Classification, Dental Chart, Potential Match Search, Teeth Image Comparison

ABSTRACT

Novel Techniques for Automated Dental Identification

Nourdin Al-sherif

Automated dental identification is one of the best candidates for postmortem identification. With the large number of victims encountered in mass disasters, automating the process of postmortem identification is receiving an increased attention. This dissertation introduces new approaches for different stages of Automated Dental Identification system: These stages include segmentations, classification, labeling, and matching:

We modified the seam carving technique to adapt the problem of segmenting dental image records into individual teeth. We propose a two-stage teeth segmentation approach for segmenting the dental images. In the first stage, the teeth images are preprocessed by a two-step thresholding technique, which starts with an iterative thresholding followed by an adaptive thresholding to binarize the teeth images. In the second stage, we adapt the seam carving technique on the binary images, using both horizontal and vertical seams, to separate each individual tooth. We have obtained an optimality rate of 54.02% for the bitewing type images, which is superior to all existing fully automated dental segmentation algorithms in the literature, and a failure rate of 1.05%. For the periapical type images, we have obtained a high optimality rate of 58.13% and a low failure rate of 0.74 which also surpasses the performance of existing techniques.

An important problem in automated dental identification is automatic classification of teeth into four classes (molars, premolars, canines, and incisors). A dental chart is a key to avoiding illogical comparisons that inefficiently consume the limited computational resources, and may mislead decision-making. We tackle this composite problem using a two-stage approach. The first stage, utilizes low computational-cost, appearance-based features, using Orthogonal Locality Preserving Projections (OLPP) for assigning an initial class. The second stage applies a string matching technique, based on teeth neighborhood rules, to validate initial teeth-classes and hence to assign each tooth a number corresponding to its location in the dental chart, even in the presence of a missed tooth. The experimental results of teeth classification show that on a large dataset of bitewing and periapical films, the proposed approach achieves overall classification accuracy of 77% and teeth class validation enhances the overall teeth classification accuracy to 87% which is slightly better than the performance obtained from previous methods based on EigenTeeth the performance of which is 75% and 86%, respectively.

We present a new technique that searches the dental database to find a candidate list. We use dental records of the FBI's Criminal Justice Service (CJIC) ADIS database, that contains 104 records (about 500 bitewing and periapical films) involving more than 2000 teeth, 47 Antemortem (AM) records and 57 Postmortem (PM) records with 20 matched records.

The proposed approach consists of two main stages, the first stage is to preprocess the dental records (segmentation and teeth labeling classification) in order to get a reliable, appearance-based, low computational-cost feature. In the second stage, we developed a technique based on LaplacianTeeth using OLPP algorithm to produce a candidate list. The proposed technique can correctly retrieve the dental records 65% in the 5 top ranks while the method based on

EigenTeeth remains at 60%. The proposed approach takes about 0.17 seconds to make record to record comparison while the other method based on EigenTeeth takes about 0.09 seconds.

Finally, we address the teeth matching problem by presenting a new technique for dental record retrieval. The technique is based on the matching of the Scale Invariant feature Transform (SIFT) descriptors guided by the teeth contour between the subject and reference dental records. Our fundamental objective is to accomplish a relatively short match list, with a high probability of having the correct match reference. The proposed technique correctly retrieves the dental records with performance rates of 35% and 75% in the 1 and 5 top ranks respectively, and takes only an average time of 4.18 minutes to retrieve a match list. This compares favorably with the existing technique shape-based (edge direction histogram) method which has the performance rates of 29% and 46% in the 1 and 5 top ranks respectively.

In summary, the proposed ADIS system accurately retrieves the dental record with an overall rate of 80% in top 5 ranks when a candidate list of 20 is used (from potential match search) whereas a candidate size of 10 yields an overall rate of 84% in top 5 ranks and takes only a few minutes to search the database, which compares favorably against most of the existing methods in the literature, when both accuracy and computational complexity are considered.

Dedication

To my mother's soul, and my father,

To my wife and kids with love,

To all my teachers and to all those who helped me with this work.

Acknowledgements

To begin with, I have been very fortunate to work in the ADIS project, being guided by Dr. Hany Ammar as my research advisor. I would like to express my special thanks to him. His valuable comments and feedback were very helpful to me.

I was privileged to have Dr. Guodong Guo co-advise my Ph.D. research, it has been a wonderful experience to get exposed to the great research qualities Dr. Guo possess. I express my heartfelt thanks to Dr. Guo, who has been guiding me right from the beginning of my research work. His continuous support, sound advices and valuable time were very vital to the quality of this research. I would not have reached this stage in research without his support. I highly appreciate his patience, humbleness and friendliness. Dr. Guo's enthusiasm has always inspired me.

I am also very grateful to Dr. Xin Li not only for being a valuable member of my examination committee, but also for being skillful teacher of some courses such as Image Processing and video digital processing. I would like to extend my vote of thanks to Dr. Don Adjeroh ,Dr.Mathew Valenti for serving on my committee. Besides I would like to extend my gratitude towards Dr. Richard Gurevic from the WVU School of dentistry for accepting to serve on my committee. I would also like to convey my special thanks to Dr.Arun Ross who not only taught me some of the important concepts in Pattern Recognition and Biometrics, but also gave valuable suggestions for my research.

I would also like to thank my colleague Dr. Aymen Abaza. Dr. Abaza helped me a lot to establish my studies during the first two years of my PhD.

Last, but by no means least, I find it an opportunity to express my love and deep appreciation to my dear family. My family has always been a source of endless support and love; my wife (Nawal Al-sherif) and my kids (Muayed and Mahad) for their love, care, support and sacrifices. Also I would like to thank all my lovely family members, my parents, brothers, and sisters.

Supporting Publications

[1] **Al-sherif Nourdin**; Guodong Guo; and Hany Ammar; "A New Approach to Teeth Segmentation," (ISM), 2012 IEEE International Symposium on Multimedia, vol., no., pp.145-148, 10-12 Dec. 2012.

[2] **Al-sherif Nourdin**; Guodong Guo; and Hany Ammar; "Automated Classification of Teeth in Bitewing Dental Images Using OLPP", (ISM), 2012 IEEE International Symposium on Multimedia, vol., no., pp.92-95, 10-12 Dec. 2012.

[3] **Nourdin Al-sherif**; Ayman Abaza; and Hany Ammar; "Human Identification Based on Tooth Contour Guided SIFT Descriptor", Proceedings of the IASTED International Conference December 14 – 16, 2011 Dallas, USA Signal and Image Processing (SIP 2011).

[4] **Al-sherif Nourdin**; Guodong Guo; and Hany Ammar; "Retrieving Dental Radiographs for Post-Mortem Identification Using Orthogonal Locality Preserving Projections (OLPP)", December 26 - 28, 2012 Cairo, Egypt International Computing Conference in Arabic: ICCA 2012.

[5] **Nourdin Al-sherif**; Ayman Abaza; and Hany Ammar; "Dental Record Retrieval Based on Guided SIFT Descriptors", Proc. Of the 1st Taibah University International Conference on Computing and Information Technology (ICCIT 2012), Al-Madinah – Saudi Arabia, March 2012.

[6] **Nourdin Al-sherif**; Ayman Abaza; and Hany Ammar; "Human Identification Based on Tooth Contour Feature", Proceedings of the ICCA International Computing Conference In Arabic, 2011 Riyadh, Riyadh – Saudi Arabia, May 2011.

[7] **Al-sherif Nourdin**; Guodong Guo; and Hany Ammar; "A New Approach to Retrieve Dental X-ray Images for Post-Mortem Identification based on LapacianTeet", to appear at 2nd Conference on Computer Science: Innovation & Technology-2-4 July 2013.

[8] **Al-sherif Nourdin**; Guodong Guo; and Hany Ammar; "Dental Biometrics Human Identification Using Dental Radiographs", to be submitted to IEEE Transactions on Information Forensics and Security.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Overview

Humans have used body characteristics such as face, and voice for thousands of years to recognize each other. Reliable recognition is important for many applications, such as law enforcement, border control, or access control. Law enforcement agencies charged with the responsibility of investigating the evidence from cases involving violent crime, missing persons and mass disaster scenarios, have exploited biometrics for decades as key forensic recognition tools. With the evolution in information technology and the huge volume of cases that need to be investigated by forensic specialists, automation of forensic identification became inevitable.

Post-Mortem (PM) identification i.e. identification after death, is a more difficult problem than Ante-Mortem (AM) identification, since few biometric characteristics can resist early decay of body tissues as well as withstand severe conditions usually encountered in mass disasters. Dental features are the best candidates for PM identification.

In 1997, the Criminal Justice Information Services Division (CJIS) of the Federal Bureau of Investigation (FBI) created a dental task force (DTF) whose goal is to improve the utilization and effectiveness of the National Crime Information Center's (NCIC) Missing and Unidentified Persons (MUP) files. The DTF recommended the creation of a Digital Image Repository (DIR) and an Automated Dental Identification System (ADIS). The proclaimed ADIS is an automated biometric system that when fed with raw subject dental record or records, will find a minimum set of candidate (or reference) records that have high similarities to the subject. Then, a forensic expert can examine the radiographs of the few candidates instead of the entire DIR.

We view ADIS as a collection of the following components:

(i) Record Preprocessing component which handles dental records cropping into dental films, grayscale contrast enhancement of films, classification of films into bitewing, periapical, or panoramic views, segmentation of teeth region from films, teeth contour extraction and finally annotating teeth with labels according to their location.

(ii) Potential Matches Search component which manages archiving and retrieval of dental records based on high-level dental features (e.g. number of teeth and their shape properties) or dimensionality reduction and produces a candidate list.

(iii) Image Comparison component which mounts for low-level tooth-to-tooth comparison between subject teeth -after alignment- and the corresponding teeth of each candidate, thus producing a short match list.

1.2 What is new in this thesis?

This thesis introduces new approaches for different stages of ADIS system:

- We adapt the seam carving technique to the problem of segmenting dental image records into individual teeth. Our approach is heavily based on concepts of seam carving technique. We propose a teeth segmentation approach for segmenting the dental images. The First stage the teeth images are preprocessed by a two-step thresholding technique, which starts with an iterative thresholding followed by an adaptive thresholding to binarize the teeth images. The second stage we adapt the seam carving technique on the binary images, using both horizontal and vertical seams, to separate each individual tooth.
- An important problem in automated dental identification is automatic classification of teeth into four classes (molars, premolars, canines, and incisors). An equally important problem is the construction of a dental chart, which is a data structure that guides tooth-to-tooth matching. Dental chart is a key to avoid illogical comparisons that inefficiently consume the limited computational resources and may mislead decision-making. We tackle this composite problem using a two-stage approach. The first stage utilizes low computational-cost, appearance-based features, using Orthogonal Locality Preserving Projections (OLPP) for assigning an initial class. The second stage applies a string matching technique, based on teeth neighborhood rules, to validate initial teeth-classes and hence to assign each tooth a number corresponding to its location in the dental chart, even in the presence of missed tooth.
- We present a new technique that searches the dental database in a fast way to find a candidate list. This list must have a very high probability of containing the target record, and its size is very short compared to the original database size. Therefore, it will facilate comparison image component (comparsion on low level teeth features) in the automated system in order to retrieve the mach list in a rapid manner and the expert forensic is to be making a final decision about the identity of the undefined person. Our approach consists of two main

stages, in the first stage is to preprocessing of the dental records (segmentation and teeth labeling classification) in order to get a reliable appearance-based, low computational-cost features. In the second stage we used technique based on LaplacianTeeth using OLPP algorithm to produce candidate list.

- We address the teeth matching problem by presenting a new technique for dental record retrieving. The technique is based on the matching of the Scale Invariant feature Transform (SIFT) descriptors guided by the teeth contour between the subject and reference dental records. Our fundamental objective is to accomplish a relatively short match list, with a high probability of having the correct match reference. Matching these SIFT descriptors between subject and reference results in matching score. To move from the tooth-to-tooth matching level to record-to-record matching level, we use the mean rule to obtain record-to-record matching score. Finally, the matching score between a subject record and a reference record will be sent to the match list.

The projected deliverables as new contributions of this thesis are:

- We adapting the seam carving technique for teeth segmentation that was used for resizing digital images by using both horizontal and vertical seams, to separate each individual tooth (region of interest) from their corresponding dental film.
- We employ new teeth classification criteria based on appearance features, Orthogonal Locality Preserving Projection (OLPP) to construct dental chart that provide high teeth labeling accuracy to improve the identification performance.
- We present a new technique for dental records retrieval (candidate list) based on orthogonal LaplacianTeeth guided by the appearance teeth features with reducing searching time and maintaining reasonable accuracy. The normalized subject and reference teeth are subjected into teeth space spanned by orthogonal LaplacianTeeth space that produced two vectors, one for the subject tooth and another for reference tooth which entered to Least Square Error (LSE) which gives the score of matching between the subject and reference tooth. In some cases, a tooth may occur in multiple films resulting in multiple distance values corresponding to different representations of the same tooth. In such cases, we choose the mean of all the distances as the match score. To move from tooth-to-tooth comparison toward record-to-record comparison, we combine the match scores generated by all component teeth via the mean rule.
- We present a new technique for dental records retrieval based on SIFT descriptors guided by the tooth contour to search the dental database in a fast way to find a match list. This match list must have a very high probability of containing the target record, and it is very short compared to the original database size.

1.3 Problem Statement

In pursue of our research effort on developing a research prototype of an automated dental identification system (ADIS), we identify the following challenging problems:

- **Problem one (Teeth Segmentation):** The extraction of the teeth from their corresponding dental radiographs is called local segmentation. In other words, the goal of radiograph segmentation is to localize the region of each tooth in a dental X-ray image.

- Given:

- ✓ A dental record *D*, with *f* cropped radiographic films.
- ✓ Dental film *f,* may suffer from poor quality, low contrast and uneven exposure.

- The Problem:

- ➢ To separate region of interest, tooth segment, from dental film, in the presence in variations and intensity variations within the same X-ray image and from one image to another.
- ➢ To use a fast method, so as not to affect the overall system timeline performance.

- **Problem two (Teeth Labeling):** This involves automatic classification of teeth into four classes (molars, premolars, canines, and incisors) as part of creating a data structure that guides tooth-to-tooth matching, thus avoiding illogical comparisons that inefficiently consume the limited computational resources and may mislead decision-making.

- Given:

- ✓ A training database of labeled dental Regions of Interest (ROIs) each assigned a class label $\omega_k \in$ {'*I*', '*C*', '*P*', '*M*'}, where '*I*' stands for Incisor, '*C*' stands for Canine, '*P*' stands for Premolar, and '*M*' stands for Molar.
- ✓ Adult Dental Atlas with specific teeth arrangement.
- ✓ A dental record *D*, with *f* radiographic films, and each film contains *Q* dental ROIs,
- ✓ Each of the *Q* dental ROIs corresponds to an input tooth (t_q),

- ✓ Presence of missed tooth.

- The Problem:

- ➢ To assign a tooth-class label ω_k to each t_q (Initial teeth classification).
- ➢ To assign a label to each tooth based on teeth neighborhood rules, even in the presence of missed tooth.
- ➢ To validate the initial teeth-classes and hence to assign each tooth a number corresponding to the tooth location in the dental chart.

- **Problem three (Teeth Matching):**

(i) **Potential Matches Search:** Is to search the dental database in a very fast manner to find a candidate list. This candidate that has high similarity with the subject dental record, and have a suitable length compared to the original database size.

(ii) **Image Comparisons:** Image Comparison component which mounts for low-level tooth-to-tooth comparison between subject teeth and the corresponding teeth of each candidate, thus producing a short match list.

- Given:

- ✓ A Subject dental record D_s, with f_s radiographic films, and each film contains Q_s dental ROIs,
- ✓ Each of the Q_s dental ROIs corresponds to a tooth view (t_s), and each tooth view is assigned a label corresponding to the tooth location in the dental chart.
- ✓ A data base of references, each reference record D_r with f_r radiographic films, and each film contains Q_r dental ROIs,
- ✓ Each of the Q_r dental ROIs corresponds to a tooth view (t_r), and each tooth view is assigned a label corresponding to the tooth location in the dental chart.
- ✓ There exist multiple views for some teeth either in the subject or the reference records.

- The Problem:

- To determine the similarity of match $p_M(t_s, t_r)$ between subject tooth (t_s) and reference tooth (t_r).
- To determine the probability similarity of match $p_M(S, R)$ between the subject record S and each of the reference records R.
- To form a candidate list based that contains those reference records of high matching probability.
- This match list must have a very high probability of containing the target record, and have a suitable length compared to the original database size.

- Problem four (ADIS Performance): The present ADIS system is not fast and accurate due to some of its components in the algorithm. Some of the components in the algorithm consume more time and as well the entire algorithm has non-acceptable accuracy. One needs to concentrate on the individual components to improve the performance in terms of speed and accuracy. Improving each individual module in the ADIS system will result in the overall performance of the system both in speed and accuracy. The components which perform better in terms of speed and accuracy are replaced with the existing one to improve the overall performance of the ADIS system.

  Segmentation, labeling and matching are some of the components which are responsible for retrieving a less accurate matching list, as well responsible for slow ADIS system. We propose new techniques which will replace these components and increase the overall performance of the ADIS system.

2 Literature Review

2.1 Biometric Identification

Biometric identification has been receiving growing interest over the past few years especially after the wide spread use of the Internet, which called for reliable means of personal identification and verification. Identification is a “one-to-many” matching process in which a database of known identities, with specific classes of characteristics, is searched for the best match to a set of subject identity characteristics. On the other hand, verification is a “one-to-one” matching process in which a claimant’s characteristics are compared against the characteristics of the claimed identity. In general, identification should answer the question “Who am I?” while verification should answer the question “Am I who I claim I am?”. The terms “identification” and “recognition” are used interchangeably whereas the term “authentication” is an alternative term for “verification” [1].

Biometrics is the science of automatically identifying individuals based on their physiological or behavioral biological characteristics. Hence, Biometric characteristics may be classified as physiological identifiers and behavioral identifiers. Physiological identifiers are characterized by physical attributes of an individual such as fingerprints, iris patterns, facial images, and hand geometry. Whereas behavioral identifiers are characterized by behavioral traits that are learned or acquired over time, signature, speech, and keyboard stroke patterns are examples of behavioral identifiers. In our everyday life we use some biometric characteristics for personal identification; the most common examples are facial images and speech.

For a biometric characteristic to qualify as a reliable identifier it should possess universality, uniqueness, permanence, and collectability [2]. Universality means that each individual has the characteristic, while uniqueness implies that different persons have no resemblance to each other in the characteristic. Permanence requires that the characteristic should neither change nor can be alterable; whereas collectability implies that the characteristic can be captured or represented.

Traditional personal identification techniques, which are either “Token-based” or “Knowledge-based”, are not as reliable and “foolproof” as biometric techniques. The flaws of traditional techniques originate from the fact that they rely on “non-inherent” attributes to identify a person. Examples of traditional techniques are drivers’ licenses and door keys as tokens, and passwords and PINs as knowledge. False identification is likely to occur in cases when tokens are lost or stolen and when knowledge is forgotten or guessed. Biometric identification is exploited in a diverse number of modern and classical applications, ranging from physical and logical access control in security applications, to criminal and victim identification in forensic applications.

2.2 Postmortem Identification

Antemortem (AM) identification, that is identification prior to death, is usually possible through comparison of many biometric identifiers. While postmortem (PM) identification, that is identification after death, is impossible using behavioral biometrics. Moreover, under severe circumstances, such as those encountered in mass disasters (e.g. airplane crashes, wars) or if identification is being attempted more than a couple of weeks postmortem (e.g. late discovery of dead bodies), under such circumstances, most physiological biometrics may not be employed for identification, because the soft tissues of the body decay to unidentifiable states. Therefore, in addition to the properties of qualified biometric identifiers listed in section 2.1, a postmortem biometric identifier has to resist the early decay that affects body tissues [3][4].

Because of their survivability and diversity, the best candidates for postmortem biometric identification are the teeth, more specifically, the dental features. Forensic odontology is the branch of forensics concerned with identifying human individuals based on their dental features [5][6]. Traditionally, forensic odontologists relied on the morphology of dental restorations (fillings, crowns, .. etc.) to identify victims. However, contemporary generations have less dental decay than their predecessors do. Hence, it is becoming important to make identification decisions based on inherent dental features like root and crown morphology, tooth size, rotations, spacing between teeth and sinus patterns [3][4].

Employment of dental records for postmortem identification finds old roots in forensic identification history; however, more attention has been given to this topic after several mass disasters in which forensic odontology was the prime tool for positive identification of victims. One of these famous disasters is the explosion of the TWA flight 800 over the Atlantic Ocean on July 1996 when all 230 passengers were killed. By May 1997, 216 corpses were positively identified, 137 out of which were identified by dental means [3][4]. In the Spring, 1997, The Criminal Justice Information Services Division (CJIS) created a dental task force (DTF) whose goal is to improve the utilization and effectiveness of the National Crime Information Center's (NCIC) Missing and Unidentified Persons (MUP) files. It recommended the creation of an Automated Dental Identification System (ADIS) with goals and objectives similar to the

Automated Fingerprint Identification System (AFIS) but using dental characteristics instead of fingerprints. The proclaimed dental identification system is a near-fully automated system that when fed with raw PM dental records, will find a minimum set of candidate Antemortem (AM) records, ideally one, for each of the input Postmortem (PM) records. A victim identity is discovered once his/her PM record matches an AM record.

2.3 Dental Identification Systems

2.3.1 Semi-Automated Dental Identification Systems

There are many trials to evolve computer- aided postmortem identification systems, but we can consider that CAPMI [7] and WinID [8] are the most popular among these systems. These systems are very developed but none of them can provide us with the desired level of automation, where they need an important role for human interference. As a result, in both CAPMI and WinID feature extraction, coding, and image comparison are passed by hand.

2.3.1.1 Assisted Post Mortem Identification system CAPMI

Assisted Post Mortem Identification system (CAPMI) in order to enhance the effectiveness of forensic scientists by automatically comparing the reference files and subject files. CAPMI is considered as a sorting tool, not an identification system, and only works effectively when it deals with large numbers of cases [9]. CAPMI generates ranked list which starts with the most likely matches. This list leads forensic odontologists to get the best possible matches of antemortem records to postmortem records. Data entry in CAPMI system can be done either by using a keyboard input or by using an optical mark reader, the dental information associated with each tooth includes:

For dental data representation, up to 16 bits of information are used per tooth, for example, if a tooth is present, bit ‘0’ is set, but if a tooth is absent, then it is reset. To show if a crown is present and the tooth is root filled, the other bits can be used to present the 5 surfaces of the tooth. The following notations are used to depict the features of each tooth:

mesial cavity is entered as (m), a crown is entered as (jmodbp) to refer that all five surfaces are restored (modbp) and a crown is present (j). (0) refers to the presence of a tooth while (x) to its absence. UL6.mod refers that the Upper Left first permanent molar is present and has restorations in the mesial, occlusal and distal surfaces; while LL3x refers the Lower Left canine is lost. (b: bucal, p: palatal).

The dental information of a subject is compared to the dental information of the candidates by entering the postmortem data for all teeth. The result will be arranged from the most likely match to the least likely match. The comparison product is either: Match, Mismatch, or a possible match. A Match can be found when antemortem and postmortem dental are exactly the same. A Mismatch can be found when antemortem dental condition is different from the postmortem dental condition. A Possible match can be found when the antemortem dental condition is different from the postmortem dental condition but could have evolved into the postmortem condition.

In order to achieve a positive identification, forensic scientists depend on many facts such as: teeth orientation, type of restorative materials, and radiographic appearance. While in CAPMI system these properties are not included because the experiment has proved that these additional data would only increase the processing time and decrease the power of the system and this due to mismatches induced by the subjectivity inherent in the recognition and identification of these entities. According to study made by the US Army on 7030 Army males, the average person has seven features. For persons with four or more features, 93% had no other person with the same features. For persons with 7 dental features, the correct match was the peak of the most likely identities list in 95% of the tests. A correct match was found within the first ten records 100% of the time. CAPMI is able of comparing and sorting 1200_5000 records per second on most personal computers [10].

2.3.1.2 WINID

WinID is a dental computer system that matches missing individuals to unidentified individuals by using dental and anthropometric characteristics in order to class probable matches [8]. There are another information can be inserted into the WinID database, like restored dental surfaces, physical descriptors, pathological, and anthropologic findings.

The dental codes utilized in WinID are extensions of the CAPMI codes. In the WinID system we have two kinds of codes, a primary codes and secondary codes, where a dash (-) is placed between the primary and secondary codes. A primary codes and up to five secondary codes can be specified. We can notice that the most investigations in WinID employ only the primary codes because if we are uncertain of a secondary code we can leave the code out.

The following tables are listing the primary and secondary codes:

Primary Codes

- M - *mesial* surface of tooth is restored.
- O - *occlusal* surface of posterior tooth is restored.
- D - *distal* surface of tooth is restored.
- F - *facial* surface of tooth is restored.
- L - *lingual* surface of tooth is restored.
- I - *incisal* edge of anterior tooth is restored.
- U - tooth is *unerupted*
- V - non-restored tooth - *virgin*
- X - tooth is missing- *extracted*
- J - tooth is *missing postmortem* or the clinical crown of the tooth is not present for examination. Also used for avulsed tooth. The root or an open socket is present, but no other information is available.
- N - *no information* about tooth is available

Secondary Codes

- A - an *anomaly* is associated with this tooth. Specifics of the anomaly may be detailed in the comments section.
- B - tooth is *deciduous.*
- C – *crown.*
- E - *resin* filling material.
- G - *gold* restoration.
- H - *porcelain.*
- N - *non-precious* filling or crown material. Includes stainless steel.
- P - *pontic.* Primary code must be X to indicate missing tooth.
- R - *root canal* filled.
- S - *silver* amalgam.
- T - *denture* tooth. Primary code must be X to indicate missing tooth.

- *Z - temporary* filling material. Also indicates gross *caries* (used sparingly).

Examples

- MODFL-S mesial occlusal distal facial lingual silver amalgam restoration.
- DL tooth has distal lingual restoration.
- MODFL-CG gold crown.
- MODFL-CHR endodontically treated tooth with porcelain crown.
- MI-E mesial incisal resin.
- X tooth missing.
- V-B virgin deciduous tooth.
- MO-SB mesial occlusal silver amalgam in deciduous tooth.
- X-PN missing tooth replaced with non-precious pontic.
- X-T missing tooth replaced with denture tooth.
- J- missing postmortem
- MO-AZ mesial occlusal temporary filling (or caries) on tooth with an anomaly

2.3.1.3 CAPMI VS. WINID

In [11] McGivney emphasized the similarities and differences between CAPMI and WinID. The similarities of the both systems are 1) using the same comparison algorithm, and 2) having the same capability of ranking possible identifications by least number of dental mismatches or most dental hits. On the other hand, only WinID system is able of producing a ranked list of non-dental identifier matches in addition to a list of most restoration hits. The other important advantages of WinID system over CAPMI is its ability to generate the odontograms and to displaying images.

In [12], Lewis and Leventhal proposed a Locator System algorithm that differs from CAPMI and WinID because the Locator System depends on initial screening and classification of AM and PM files into 6 categories based on dental work features, such as, the kinds of restored surfaces, attendance of crown and roots, missing teeth. They also stated that the three systems have poor performance with few or no restorations.

In [13], Bowers and CornwellIn compared PM records of two cases to their candidate AM records using operator controlled digital image processing and analysis techniques. For the first

AM/PM pair, they set up a common baseline between a tooth- pair by adjusting the orientation of films such that the cemento-enamel junction (CEJ) of the tooth-pair emerges horizontal. In the second AM/PM couple, they chart a smoothed restoration area from the AM tooth to the PM tooth employing an alignment change and conclude that the PM record matches the AM record based on their visual comparison of figures of aligned restorations.

Bowers and CornwellIn refuse the hypothesis that the first PM state matches its candidate AM state by noticing major difference in these measures. They resize the PM tooth such that its (CEJ) measurement agrees with that of the AM tooth, so they measure two age-independent characteristics: the root furcation heights and the distal root divergence angle.

2.3.2 Automated Dental identification System (ADIS)

ADIS is required to speed up the postmortem identification process. It is estimated that at any point in time there are over 100,000 unsolved MUP cases in the National Crime Information Center (NCIC), 60 % of which have remained unsolved for 90 days or longer. Technically speaking, this large number of unsolved cases hinders the capabilities of search techniques currently employed. The benefit of ADIS to the society will surpass saving millions of dollars to also include psychological benefits of families of missing persons when knowing the disposition of their loved ones without having to wait extended period of time [14].

In 1997, the Criminal Justice Information Services (CJIS) division of the Federal Bureau of Investigation (FBI) formed a Dental Task Force (DTF), which recommended the creation of:

- Digital Image Repository (DIR): later in May of 2005, (CJIS) Management approved the creation of the National Dental Image/Information Repository (NDIR) to facilitate the identification of Missing, Unidentified, and Wanted persons. The NDIR became available in February 15, 2006 and is housed on Law Enforcement Online (LEO) and is permitting law enforcement agencies to store, access and supplement dental records which were housed in the Missing, Unidentified, and Wanted Persons (MUP) files in the National Crime Information Center (NCIC) system. The NCIC system is a nationwide computerized database of documented criminal justice information maintained and supported by CJIS. Data in NCIC is exchanged in a shared-management partnership with criminal justice officials of local, state and federal governments in the United States, its possessions, and Canada [15].

- Automated Dental Identification System (ADIS) with goals and objectives similar to the Automated Fingerprint Identification System (AFIS) (AFIS later on turned to Integrated AFIS, more commonly known as IAFIS, which is a national fingerprint and criminal history system maintained by the (CJIS) Division. The IAFIS provides automated fingerprint search capabilities, latent searching capability, electronic image storage, and electronic exchange of fingerprints and responses. As a result of submitting fingerprints electronically, agencies receive electronic responses to criminal ten-print fingerprint submissions within two hours and within 24 hours for civil fingerprint submissions [16]).

2.3.3 ADIS Overview

The proclaimed dental identification system is fully automated system that when fed with raw subject dental records will find a minimum set of match reference records, ideally one, for each of the subject records. The forensic dental expert can examine the dental records of those few candidates to make a final decision on the identity of the missing or unidentified person [17].

Automated Dental Identification System (ADIS) is comprised of two stages of (1) search and retrieval stage based on potential similarities to produce a candidate list and (2) verification stage for matching based on low level comparison of dental images to produce a short match list (as shown in Figure 1). Some of the challenges behind developing ADIS were described in [18][19].

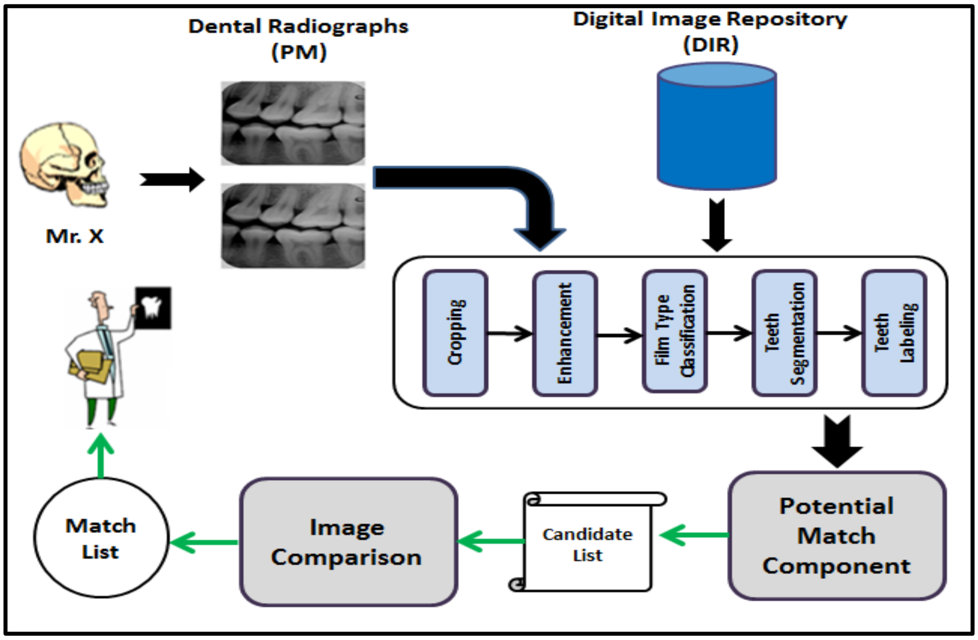


Figure 1: ADIS Overview

2.3.4 ADIS Architecture

Nassar and Ammar [20] proposed a prelude research prototype of ADIS, which Fahmy and et al [14] later extended and refined - as depicted in Figure 2 to become the conceptual framework of the ADIS prototype.

At a high level of abstraction, they view ADIS as a collection of the following components: (i) the Record Preprocessing component, (ii) the Potential Matches Search component, (iii) the Image Comparison component [21][10][22][23].

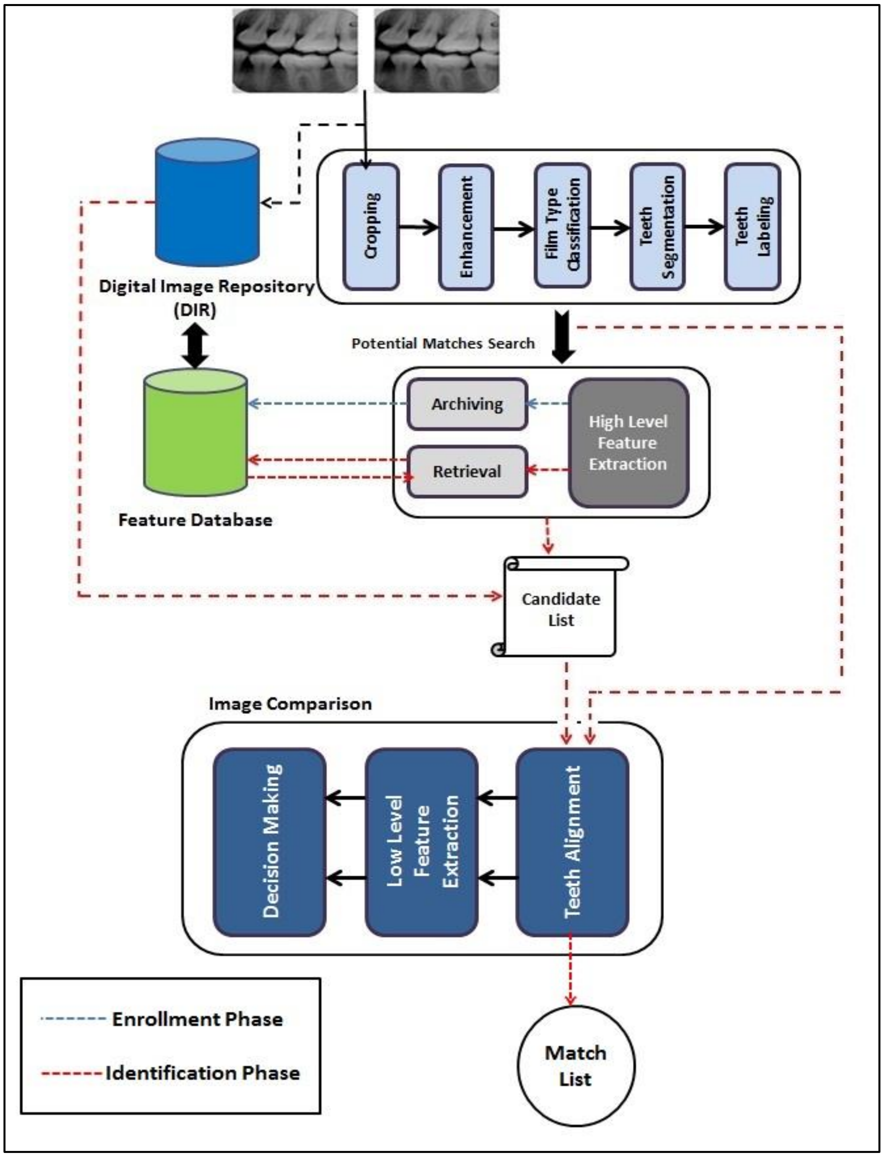


Figure 2: Block Diagram of ADIS.

2.3.4.1 Record Preprocessing

(a) Records Cropping: which is the first segmentation level, where the dental films are segments from the input records. Dental radiographic films of an individual are usually composed into a digital image record. In order to achieve high level of automation in postmortem identification it is necessary to decompose dental image records into their constituent radiographic films, which are in turn segmented to localize dental regions of interest.

Li et al. [24] presented an approach for this problem, started by using the rectangular film property and separating the background that have various color and textures. Then they classify the connected components according to the corners being right or round. Cutting the round corner components depends on whether they are 90° or 180° V-corners; and cutting the right corner ones by viewing the boundary films under a graph theoretic framework, where it is always possible to reduce the graph by successively removing unit vertexes without affecting its connectivity.

In order to develop an abstraction for varying characteristics of dental X-ray records, they proposed to classify the ambiguity sources of digitization process into the following three categories:

A) Intensity-related: Dental X-ray films are usually placed on the top of a uniform background; but it occasionally observes textured background. Dental fillings, which exist in a moderate portion of X-ray films, could have identical intensity values to the background color.

B) Geometry-related: Dental films have rectangular shape and two types of corners: round or right. However, the width, height and corner type of dental films change from record to record and sometimes vary within the same record. In some records, some portion of the border of dental films could get smeared out and become indistinguishable from the background.

C) Topology-related: Ideally it would like all dental films to be separated from each other during the digitization phase (more convenient from segmentation perspective). However, in typical databases used in forensic odontology, significant portion of the records contain films that are connected to each other

.

These three classes of ambiguities jointly contribute to the difficulty of cropping dental X-ray records. Two aspects motivate their approach: First, archness of round-corner films provides valuable clues for segmentation. Even if several round-corner films are placed side-by-side in parallel, the arc segments still remain distinguishable. Second, the straight orientation of dental X-ray films significantly reduces the solution space - i.e., it is sufficient to consider only horizontal and vertical lines for cropping purpose. They propose a three-stage approach for cropping as depicted in Figure 3: First a preprocessing stage whereby they extract the background layer of the image record, extract connected components and classify them as either round-corner or right-corner connected components. The second stage performs arch detection and dimension analysis, realization of this stage differs according to the outcome of the preprocessing stage. The third stage is a post-processing stage that performs topological assessment of the cropping results in order to eliminate spurious objects.

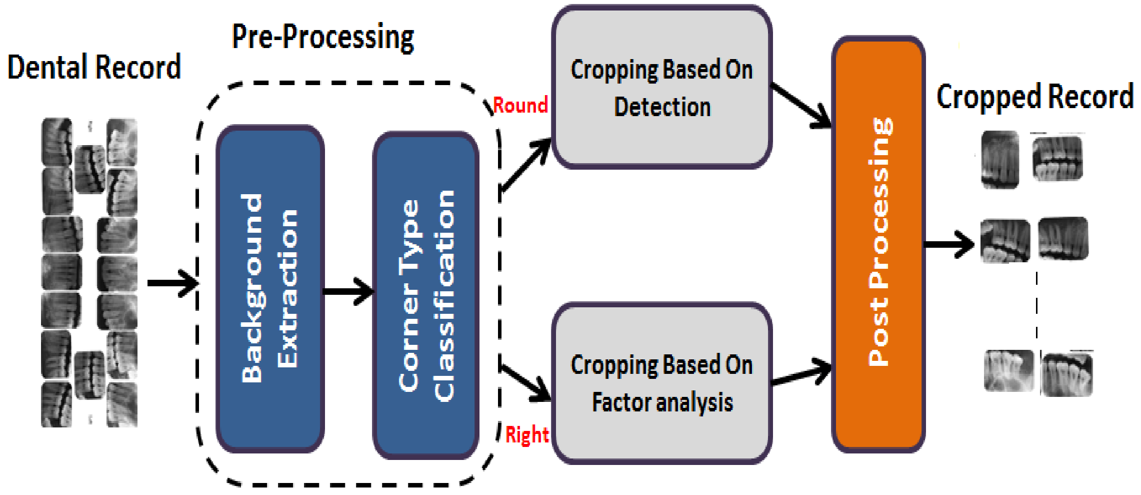


Figure 3: A block diagram illustrating the three-stage algorithm for dental record cropping.

Haj et al. [25], the proposed algorithm was heavily based on concepts of mathematical morphology and shape analysis. Characteristics of dental X-ray records vary not only from database to database but also from image to image. Figure 4 illustrates the heterogeneity among records of a typical dental record database [26][27].

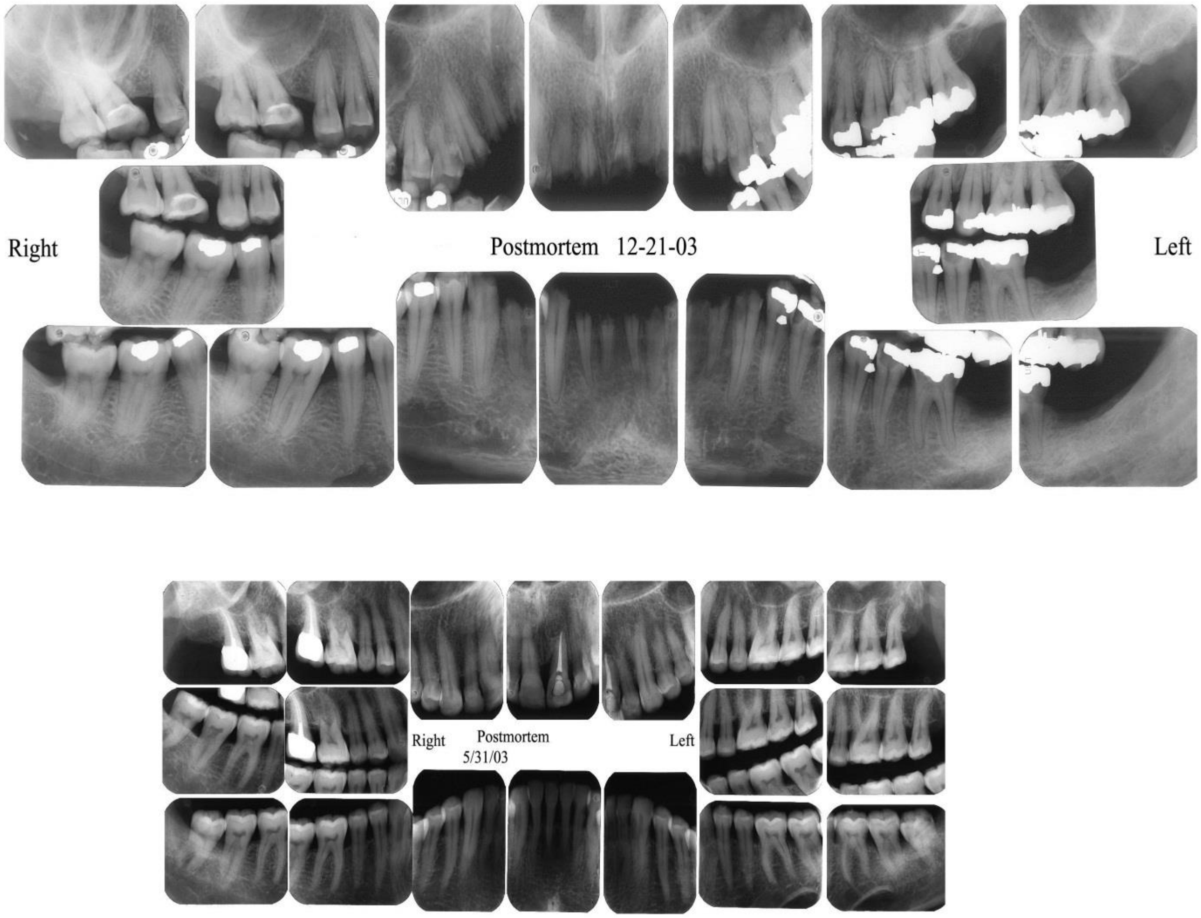


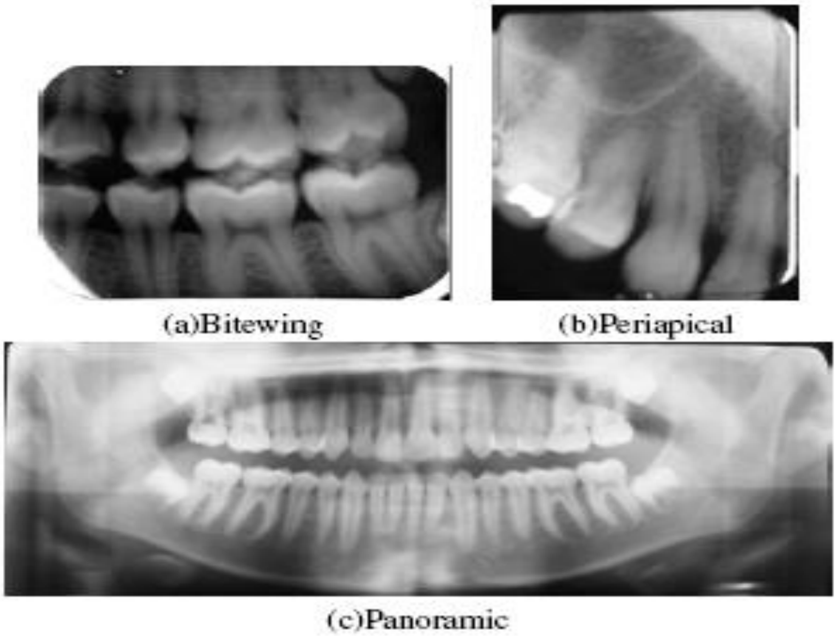
Figure 4: Sample dental records that show the heterogeneity among records of a typical dental record database. Note the varying background intensity, orientation and connectedness from record to record.

In [25], to crop the dental films from the digitized dental X-ray records. First, extracted the background that varies in its gray intensity and textures, and then they used iterative steps of basic thresholding and analyzing of the connected component to extract the dental films based on their geometrical properties. Finally, they applied series of mathematical morphology operations and horizontal and vertical edge detection to extract the films not obtained from the previous stages.

In [25], used same dental records of database as used in [26][27], each of which may contain 1 to 20 films, and has variability in the gray intensity and texture of its background. In addition, the position of films and their dimensions in a dental record vary from the other dental records in the database. They obtained 92% of the films which were perfectly cropped, 5.5% were under segmented, and only 2.5% films developed into erroneous segments. While the algorithm in [24], obtained 60% of dental records that were perfectly cropped, 33% partially cropped, and 7% failed to produce any film.

(b) Film Enhancement: A tooth maps to an area with mostly “bright” gray scales (except for the pulp tissue) while the gum maps to areas with “mid-range” gray scales, and air maps to “dark” gray scales. Thus, a “significant” contrast in gray scales within a “small” area of a dental radiograph indicates a transition from one object to another. In order to assist the coming preprocessing steps, Hajsaid et al [28][29][30] introduces a gray scale contrast stretching to transform poor quality dental film in a way that insures an appreciable degree of contrast between the dominant gray scales used in capturing the different classes of objects; in other word to compensate for possible poor contrast.

(c) Film Type Classification: In the context of ADIS, there are three types of dental images according to the way they capture the dental features. When taking radiographs of the teeth, the dental arches are divided into different regions to get a clear view of individual teeth. Each film placement is corresponding to a film type, and is designed to record specific teeth. Some commonly used film types are [31]:


(a)Bitewing (b)Periapical

(c)Panoramic

Figure 5: Different Radiographic Dental Films.

Bitewing Film: records two-thirds (the crown portion) of opposing Molar and premolar teeth, as shown in Figure 5(a). Periapical Film: either records teeth of the lower or upper jaw. It records the whole tooth (crown and root). It has different placement to record all teeth. Figure 5-b shows an upper periapical for the incisors and canine region. Periapical films are further classified into upper and lower. Panoramic Film: records all the teeth at the same time, as shown in Figure 5(c).

Zhou and Abdel-Mottaleb [32][33] classify those types based on the number, location and relationship between extracted vertical and horizontal edges. Panoramic images have more teeth than perapical and bitewing images, and therefore have more vertical edges which correspond to the teeth's boundaries. The number of vertical edge pixels can be used as a feature to distinguish between the panoramic images and the other two types of dental images. In order to distinguish between periapical and bitewing images, feature related to the orientation of horizontal and near horizontal edges can be used. There are two types of horizontal (or near horizontal) edges: one with upward gradient and the other with downward gradient. The periapical image has more of one type of these edges than the other, while a bitewing image or a panoramic image has nearly equal horizontal (and near horizontal) edges with both upward and downward gradients. The second feature that is used by[32] [33] was either the ratio of the number of horizontal edges with upward gradient to the total number of horizontal edges, or the ratio of the number of horizontal edges with downward gradient to the total number of horizontal edges. The total of two features to classify the three types of images are: (1) the number of vertical edge pixels in the image normalized by the number of rows in the image; and (2) ratio of horizontal edges with upward gradient to the total amount of horizontal edges. These two types of features are only related to the content of the dental images and are independent of the size of the image.

(e) Teeth Segmentation: After the dental films are cropped from dental records, they passed through the enhancement step followed by a film type classification step to classify the dental film view (Bitewing, Upper Periapical, or Lower Periapical) in order to preparing it for teeth segmentation. The extraction of the teeth from their corresponding dental radiographs is called teeth segmentation. Segmentation is a step required to identify the extent of teeth comprised in a digital image of a dental radiographic film. The segmentation of dental images is difficult due to the shape variations and intensity variations within the same X-ray image and from one image to another. There are several techniques proposed for teeth segmentation. These techniques are devoted to develop fully automated segmentation: In [34], Jain and Chen separate the upper jaw and the lower jaw in the bitewing and panoramic dental images by detecting the gap valley between them using the Y-axis projection histogram. Afterwards, the technique isolates each tooth from its neighbors in each jaw by detecting the gaps between them using intensity integral projection. This approach is semi-automated since it needs user interaction to choose an initial valley gap point is required to detect the gap valley between the upper and lower jaw. The withdraw of this method is, the performance of the segmentation output depends on the initial selection of valley gap point, Figure 6 shows two different segmentation results produced by choosing two different initialization points both of which are on the valley gap; the choice in Figure 6(a) leads to perfect segmentation while the choice in Figure 6(b) leads to total segmentation failure.

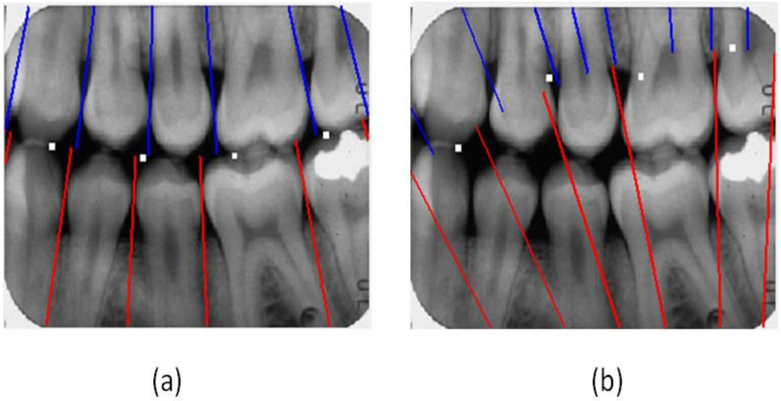


Figure 6: Success and failure cases for an image. (a) Completely segmented image. (b) Failed segmented image.

Nomir and Abdel-Mottaleb [35] proposed a fully automated approach for bitewing dental X-ray images only. The technique depends on applying the following stages: i) iterative threshold to divide the image into two parts teeth and background; ii) adaptive threshold in order to increase the accuracy and remove teeth interfering; iii) horizontal integral projection in order to separate the upper jaw from the lower jaw; iv) and, finally, vertical integral projection in order to separate each individual tooth as shown in Figure 7.

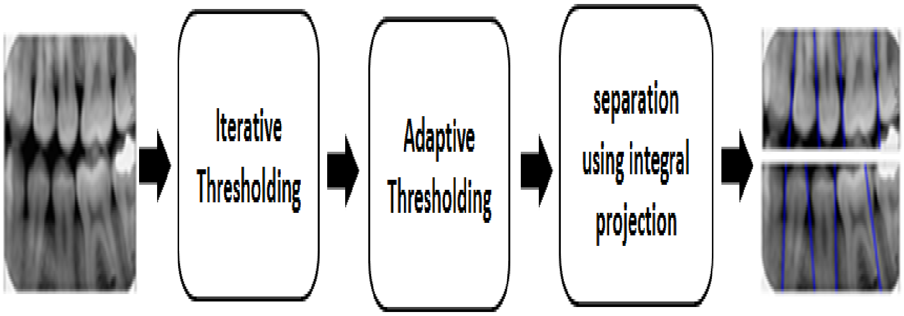


Figure 7: Block diagram of teeth segmentation (Two step thresholding).

Another fully automated approach for dental X-ray images is introduced by Zhou and Abdel-Mottaleb [32]. As shown in Figure 8, the technique depends on improving the image contrast by applying morphological transformation (top-hat filter and bottom-hat filter), and then using the window-based adaptive threshold and integral projection to segment the teeth and separate the upper and lower jaw.

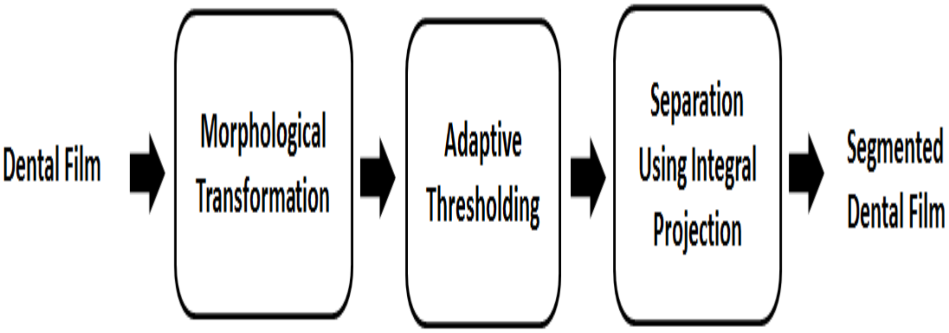

Dental Film
Morphological Transformation
Adaptive Thresholding
Separation Using Integral Projection
Segmented Dental Film

Figure 8: Block diagram of teeth segmentation (Morphological Transformation).

Hajsaid et al [28] proposed automated method based on mathematical Morphology and can be used with both Bitewing and Periapical views. Their algorithm includes as shown in the Figure 9: 1) noise filtering, 2) thresholding to isolate the teeth from the background, and 3) analyzing connect components labeling to determine the qualified ROIs based on geometrical properties. They employ the mathematical morphology to segment teeth from dental radiographs images. They use mathematical morphology operator to highlight the desired objects and suppress the others, and then threshold the resulting image to separate the desired objects from the background. Next, analyze the connected components obtained from the thresholded image based on their geometric properties in order to isolate the desired objects. The Figure 10 illustrates the grayscale line profiles of the input image.

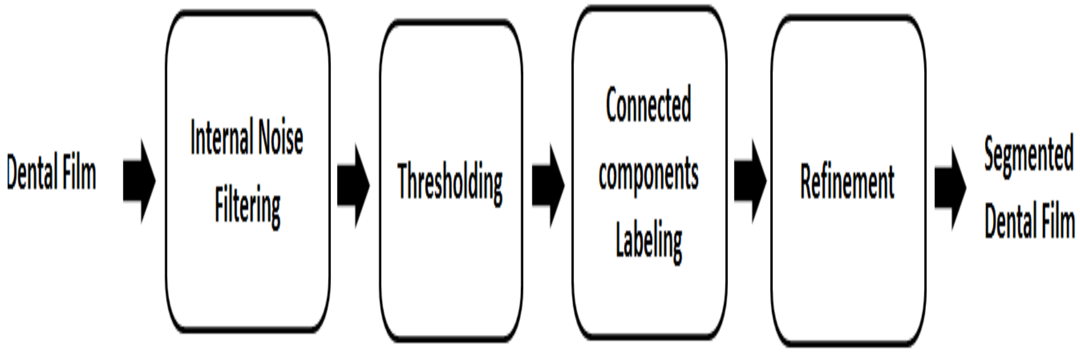


Figure 9: Block diagram of teeth segmentation (Mathematical Morphology).

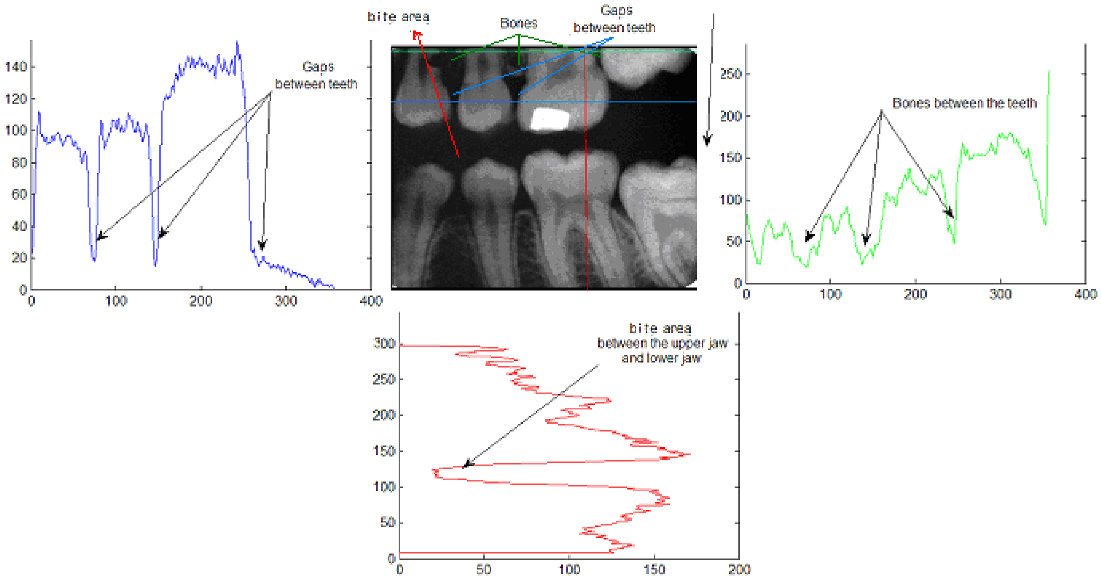


Figure 10: Grayscale line profiles of the input image.

Nassar et al. in [36] present a metrics-based object counting approach for the empirical assessment of image segmentation. To evaluate the performance of the segmentation algorithm, reference images are used to record the outcome of the experiment in a tabular form as shown in Figure 11. Each cell P_{ij} of the results table contains the number of instances where the segmentation algorithm correctly detects objects j out of i objects that are present in reference image, with $\sum_{j=0}^{i} P_{ij} = F_i$, where F_i is the number of reference images that contain exactly i objects. The results table is used in determining metrics of optimality, sub optimality, and failure based on the relative weights of the main diagonal, the sub diagonals, and the base row, respectively, the performance metrics are defined as follows [36]:

$$\text{Optimality} = 100 * \frac{\sum_{i=0}^{N} P_{ii}\, F_i}{\sum_{i=1}^{N} F_i^2} \ \%$$

$$\text{Failure} = 100 * \frac{\sum_{i=0}^{N} P_{0i}\, F_i}{\sum_{i=1}^{N} F_i^2} \ \%$$

$$m^{th} \text{ Order Sub-optimality } = 100 * \frac{\sum_{i=0}^{N-m} P_{i(i+m)}\ F_{(i+m)}}{\sum_{i=1}^{N} F_i^2} \ \%$$

While optimality and failure percentages capture instances of extreme performance of a segmentation algorithm, suboptimality measures capture the performance of algorithms in between the two extremes. For example, in teeth segmentation, first-order suboptimality is a measure of the tendency of the algorithm to miss the detection of exactly one tooth and detect all of the others, but without failure. In practical cases, it is difficult to achieve optimal performance with 100% images, and when comparing segmentation algorithms, one should favor those whose failure rates are the lowest and their optimality and low-order measures of sub optimality predominate the testing results [36].

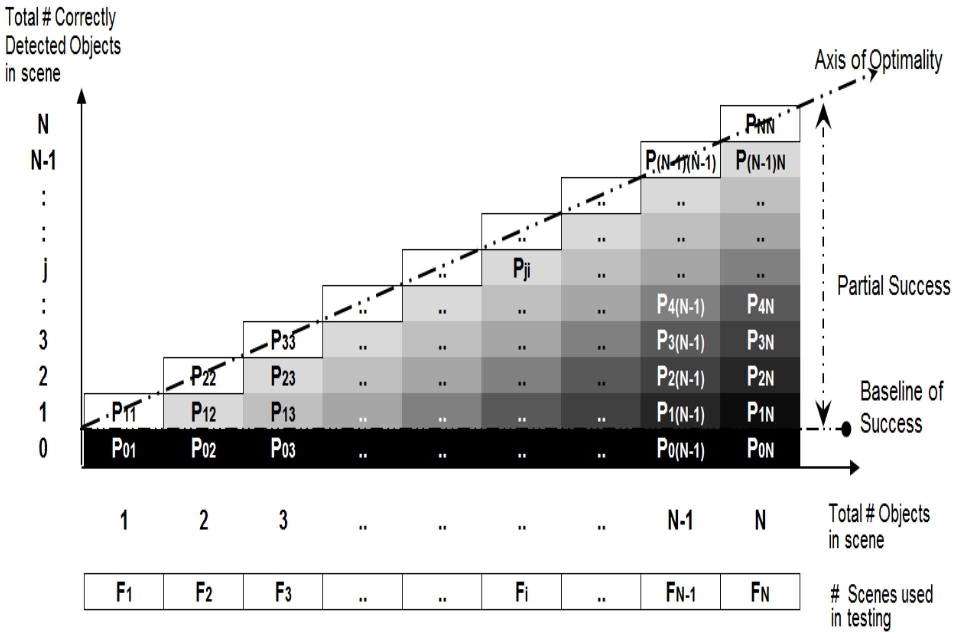


Figure 11: Tabular presentation of segmentation results.

The failure rate is especially important when assessing teeth segmentation algorithms, since those films where no teeth can be properly segmented cannot be used in the identification process.

Table 1 shows a brief comparison between the three algorithms based on underlying principles, type of dental film views they are reported to operate on, and the level of automation they achieve.

Table 1: Summary of comparison between three teeth segmentation.

Algorithm	Principles	Types of views	Is it automated?
Jain and *Hong* [34]	Integral projection	Bitewing and Panoramic	No, semi-automated
Nomair and *Abdel-Mottaleb* [35]	Iterative and adaptive thresholding, integral projection	Bitewing only	Yes
Zhou and *Abdel-Mottaleb* [32]	Morphology, adaptive threshold, integral projection	Bitewing only	Yes
Hajsaid et al [28]	Proposed a grayscale contrast stretching transformation to improve the performance of teeth segmentation	Bitewing and Periapical	Yes

This is our first problem and more details will be given in chapter 3 section 3.1.

(d) Teeth contour extraction: Shah and et al [37], presented an algorithm where the target is to segment each tooth from the isolated region, this is recognized as teeth contour extraction. They proposed an algorithm using active contour without edges, this technique is based on the intensity of the overall region of the tooth image and, therefore, does not necessitate the presence of a sharp boundary between teeth. Further, this technique can extract the region contour in the presence of additive noise and in the absence of well-defined image gradients.

(d) Teeth Labeling: Annotating teeth with labels that is corresponding their location which create dental charts capture the position of the radiographed teeth in the human mouth, thus they help avoiding illogical teeth comparisons that inefficiently consume the limited computational resources and may also mislead the decision-making process. The problem of automatic construction of dental charts was addressed in [38][39], they presented an automated algorithm (as shown in Figure 12) to classify teeth in bitewing dental film images, using Bayesian classification, and assign an absolute number to each tooth based on common numbering system used in dentistry. Fourier descriptors of the contours of the molar and the premolar teeth in bitewing images are used in the Bayesian classification of these two types of the teeth. Then, the spatial relation between the two types of the teeth is considered to number each tooth and correct the misclassification of some teeth in order to obtain high precision results. Deal only with Bitewing films (2 class problem) to classify molar and premolars.

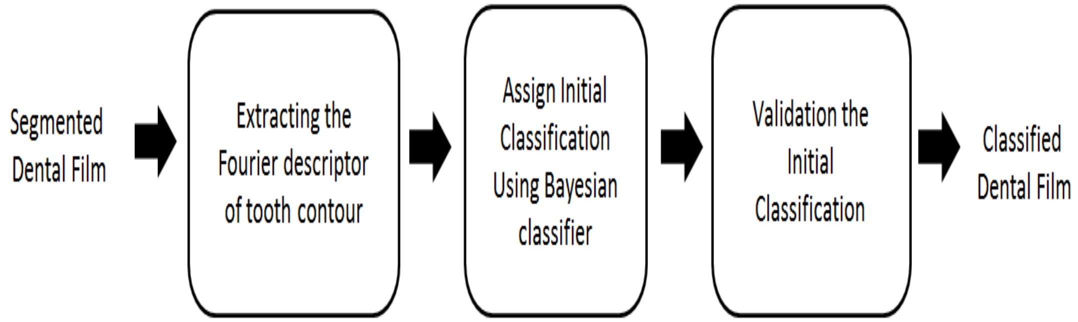

Segmented Dental Film
Extracting the Fourier descriptor of tooth contour
Assign Initial Classification Using Bayesian classifier
Validation the Initial Classification
Classified Dental Film

Figure 12: Block diagram of teeth classification (Fourier descriptor of tooth contour).

Jain and Chen [40] used a hidden Markov model (HMM) as an underlying representation of the dental atlas, where the states representing the available teeth have discrete observations, namely the class of each tooth, and the states representing the missing teeth have continuous observations (They can assign one or two missed teeth according to the distance between neighboring teeth). All these techniques use teeth contour features (shape based); while automatic extraction of teeth contour is computationally intensive, and to the best of our knowledge, none of the techniques exploited in teeth contour extraction.

Nasser and et al [41] presented a dual-stage technique as shown in the Figure 13 for automatic construction of dental charts based on appearance based features and string matching. Initially each segmented tooth (of a Bitewing/periapical film) is independently classified based on teeth reconstruction in two/four image subspaces established using Principal Component Analysis (PCA). Using teeth neighborhood rules the initial teeth class assignments is validated and possibly corrected. Finally, if the resulting teeth class sequence is unique a number for each tooth is assigned corresponding to its position in the dental quadrant it belongs to. Otherwise, if the resulting teeth class sequence is not unique it calls a reject option.

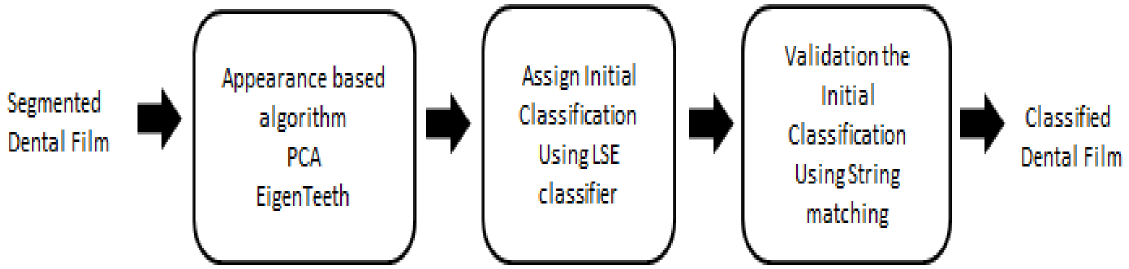

Segmented Dental Film
Appearance based algorithm PCA EigenTeeth
Assign Initial Classification Using LSE classifier
Validation the Initial Classification Using String matching
Classified Dental Film

Figure 13: Block diagram of teeth classification (Eigen teeth).

This is our second problem and more details will be given in chapter 3 section 3.2.

2.3.4.2 Potential Match Search and Image Comparison

Potential Match Search component manages archiving and retrieval of dental records based on high-level dental features (e.g. teeth intensity and shape descriptors) and produces a candidate list. The discriminative feature of this search engine is being very fast so as to be able to search a large database, with optimum accuracy. The forensic experts used to use dental and non-dental feature to reduce their search domain; in other words, they are implementing a similar technique to potential match search component, by using these features to reduce their search space. They used non-dental features as (sex, race, age, etc.) and dental features as presence/ absence of dental work and missed teeth.

In ADIS implementation, dental experts use the same set of non-dental feature as (sex, race, age, etc.) to reduce the database, and then apply the potential match search component to search the database to produce a relatively short candidate list. The Image Comparison component then handles another level of dental image retrieval against this candidate list to produce a match list. The image comparison component takes about 20 minutes to make record to record comparison. To search a database of 50 records directly using the image comparison, the system will run for about 8 hours on most personal computers.

The problem of teeth matching was addressed in [34] Jain and Chen propose a semi-automated system for human identification based on matching of teeth contours extracted from dental x-ray images. The system follows three main steps for identification: radiograph segmentation, teeth contour extraction, and shape matching. For each radiograph, a human user initializes segmentation by specifying a pixel that belongs to the gap valley (an artificial curve that best separates the maxilla and mandible), then detection of the entire gap valley as well as teeth isolation are carried out using integral projection. In [34], extraction of an isolated tooth contour is achieved in two steps, crown extraction and root extraction. Crown extraction is treated in a pixel classification approach along radial rays, while root extraction is considered an iterative neighborhood context analysis problem. Identification is completed by searching a database of contours for records with relatively small Euclidean matching distance to subject contours using a quasi-affine transformation model for contour alignment. In [42], Chen and Jain, propose a directional SNAKE approach for contour extraction of isolated teeth to mitigate excessive contour extraction errors of their approach in [41]. In [43], Chen and Jain present an automatic method for matching dental radiographs. The matching is performed in three steps. First, a shape registration method is proposed to align and compute the distance between two teeth on the basis of tooth contours. If the shapes of the dental work are available, they can assist in the matching. They propose an area-based metric for matching the dental work. The two matching distances are then combined to obtain the tooth correspondence and to measure the similarity between images. In [44], Nomair and Abdel-Mottaleb present combining two matching algorithms of a system for identifying individuals based on their dental X-ray records. The system archives the AM images into the database by segmenting the teeth, extracting their contours, numbering the teeth, and representing each tooth by a set of features. The first matching technique relies on selecting a set of salient feature points with high curvature from the contour of a tooth and generates a signature vector for each salient point. The second matching technique uses Hierarchical Chamfer distance transformation to reduce the search space. The search space is reduced significantly as well as the computational load. Given a PM image, the system segments the teeth, extracts their contours and represents each tooth a set of feature vectors. The matching modules search the database for the best matching candidates. Matching scores are calculated based on the distances between the feature vectors of AM and PM teeth.

In [32], Zhou and Abdel-Mottaleb present a content-based archiving and retrieval system of dental images for use in human identification. It contains three major stages: dental image classification, bitewing image segmentation and retrieval based on teeth shapes using bidirectional Hausdorff distance. In the classification stage, two features are proposed for dental X-ray image classification. The classified bitewing images are segmented to extract the contours of molars and premolars, which are then used to archive the images in an AM database. During retrieval, a PM bitewing image is segmented to extract teeth contours, which are used to find the most similar images in the AM database using Hausdorff distance measure.

To the best of my knowledge, the problem of implementing a potential match search was not handled in a proper way in the context of ADIS [34][42][43][44][32] as all the present trial are based on contour alignment and matching to find the distance between two dental images. All these techniques use teeth contour features (shape based); they made a tooth to tooth comparison; in other word they are implementing the image comparison stage, while using the teeth contour as a feature space. In [45], Nassar was addresses the dental image matching in two stages: first extracting high-level features thus expediting retrieval of potential matches, low-level image comparison between subject and candidate records thus achieving high matching accuracy. He focused on the tooth-to-tooth matching stage (or micro-decision-making stage) of the image comparison system and he proposed probabilistic models for class-conditional densities that he uses in micro-decision making. He also proposed an adaptive strategic searching technique and used it in conjunction with back propagation in order to estimate parameters of the micro-decision-making stage. In [46], Abaza present an algorithm that is based on dimensionality reduction using principle component analysis (PCA) that initially each segmented tooth is compared to the corresponding reference tooth based on a lower dimensional image subspace established using (PCA). By fusing multiple tooth matching scores, it calculates the similarity between the subject record and all the reference records in the database. Then it retrieves records based on the minimum Euclidean distance. In [47], Abaza is used a method Shape-Based Approach based on edge direction histogram. A histogram of the edge directions is used to represent the shape attribute. The edge information contained in the database images is generated in the preprocessing stage using edge operator such as Canny edge or Sobel edge operators. In [47] compute the similarity and distance between the subject image edge histogram and a stored reference image edge histogram.

This is our third and fourth problems and more details will be given in chapter 4 and 5 respectively.

2.3.5 ADIS web Interface

In developing an environment that integrates the various ADIS components, Chekuri et al. [48][49][50] presented a web application that is delivered to users from a web server over a network such as World Wide Web or an intranet. Web applications are popular due to the ubiquity of the web browser and have the ability to update and maintain web applications without disturbing and installing software on potentially thousands of client computers. Though many variations are possible, a web application is commonly structured as a three-tiered application. Given a subject record, the web application should allow the user to upload the dental record onto the server after the user is authenticated. The server should invoke Matlab to do preprocessing steps like cropping, enhancement, segmentation; film classification on the dental record uploaded and then should run the potential match algorithm and should give back a candidate list of reference records from the database. The candidate list should be fed into the image comparison module to get back a final match list. This application should also enable uploading of reference records on to the database. When the reference record is uploaded the preprocessing steps should be run on the record submitted and the post preprocessing data should be automatically uploaded onto the database.

2.3.5.1 Components in Web-ADIS

The design of Web-ADIS can be classified mainly into three main components: (i) Web-ADIS client. (ii) Web-ADIS server (iii) Database server. Figure 14 shows the tree diagram for Web-ADIS is as given below:

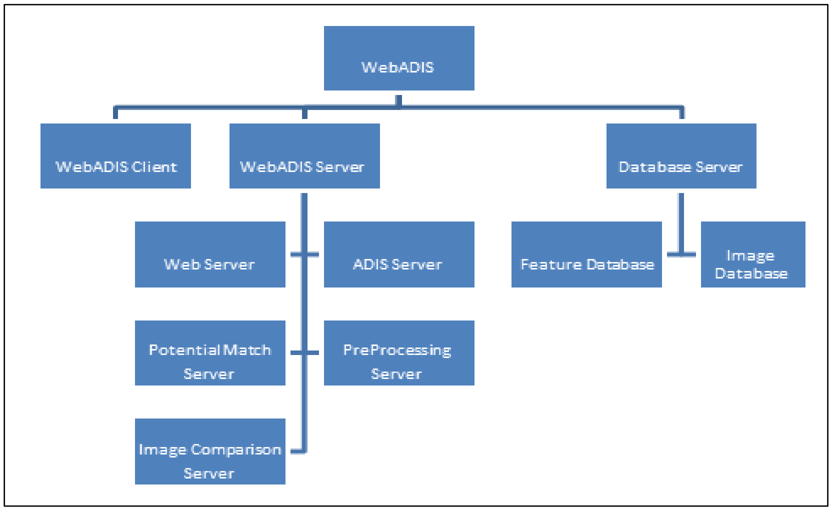


Figure 14: Tree Diagram of Web-ADIS

Web-ADIS Server and Database Server are a combination of five and two sub-components respectively. The subcomponents of Web-ADIS Server are Web Server, ADIS server, Potential Match Server, Preprocessing Server and Image Comparison Server. The sub-components of Database Server are Feature Database and Image Database.

- **Web ADIS Client**

The Web-ADIS client can be technically defined as a user using valid browser software and is trying to connect to Web-ADIS server. An authenticated user can upload a subject record into Web-ADIS server and can get back the match list. An authenticated client if authorized can also update the Web-ADIS server with new algorithms which will help the system to be more effective in terms of speed of response and accuracy. It also helps to keep the system upto date with newly developed algorithms which are proven to give better results than the presently used ones.

- **Web ADIS Server**

This component plays the role of a controller for all the work being done by Web-ADIS. This commands the flow of signals in the system. When a request is sent from Web-client, the processing of the request is done by Web-ADIS server by communicating with the database server. The results are returned to the client when the processing of request is complete.

A brief description of sub-components of Web-ADIS server is given below:

- **Web Server**

The main job of Web server is to securely connect Web-ADIS server to outside world through internet. Web server can be defined as software responsible for serving Web pages via the http protocol to clients. Web server has a unique address so that other computers connected to internet can identify it on the vast network.

- **ADIS Server**

ADIS Server acts as a controller for Web-ADIS and controls the flow of signals to Preprocessing Server, Potential Match Server, Image Comparison Server and Database Server. When a request is made by the user the ADIS Server first evaluates the type of request depending on the mode of operation, prompts the other servers to act on the request and when the result is ready to be send back to the user, it prompts the Web Server to display the result to the user.

- **Database Server**

The subject record submitted during identification mode should be compared to the dental/non-dental features of the reference images to find the final match. These features of reference images and the dental images of reference records are stored in the database so that they can be accessed while caring out identification process on the subject record. For the ease of accessing and storage of information the database server is divided into two parts: (i) Image Database: All the reference dental images are stored in image database also called Digital Image Repository (DIR). This is also the computerized repository to all users authorized for maintenance mode usually users from the Criminal Justice Information Services (CJIS) to update the references images to the database. The users trying to identify a subject in the identification mode can access the images, which are used by the Potential Match Server and the Image comparison Server to compare with the image of the subject record in process to get the match. (ii) Feature Database: This is the store house for the dental and non-dental features of all the reference dental images. Dental features are the features extracted from the reference images after the preprocessing steps are run on the reference dental images. Non-dental features comprises of features that are not related to dental images, these comprise of details like name, age, gender etc. The Feature Database can be updated by authorized users and can be accessed by the users trying to identify a subject record.

3 Proposed Preprocessing Techniques

3.1 Teeth Segmentation

Image segmentation is essential in the biometric and forensic identification process, and it plays a critical role in the most subsequent identification process. The segmentation process depends on modeling the similarity characteristic and common features of the desired object while dealing with variation in intensity, scale, pose, and shape. However, no single image segmentation technique performs well for all kinds of images; in addition, the performance of various segmentation techniques is not the same and may vary from image to image.

Teeth segmentation is one of the major pre-processing steps in building an Automated Dental Identification System (ADIS). The extraction of the teeth from their corresponding dental radiographs is called teeth segmentation. Dental radiographs may suffer from poor teeth image quality, low contrast and uneven exposure that complicate the task of teeth segmentation. To achieve a good performance in segmentation, the teeth images are preprocessed by a two-step thresholding technique, which starts with an iterative thresholding followed by an adaptive thresholding to binarize the teeth images. Then, we adapt the seam carving technique on the binary images, using both horizontal and vertical seams, to separate each individual tooth. Experimental results prove that the seam carving technique approach for teeth segmentation achieves higher accuracy and requires significantly less time as compared to the other approaches currently being used within the context of ADIS.

3.1.1 Introduction

The segmentation of dental images is difficult due to the shape variations and intensity variations within the same X-ray image and from one image to another. Some of dental images are suffering from some difficulties that are related to intrinsic factors such as the condition of the human teeth and bone density, or to extrinsic factors such as age of the film and the quality of the scanning machine. Therefore, we define the factors that introduce difficulties in dental image segmentation as follows (i) image blurring (Figure 15 (a) and Figure 15(e)), (ii) fillings (Figure 15(c) and Figure 15(f), (iii) teeth interfering (Figure 15(c) and Figure 15(e), (iv) image scan quality (Figure 15(h)), (v) the imperceptible difference in intensity between and the teeth (Figure 15(c) and Figure 15(e) (vi) dental work (Figure 15(g)).

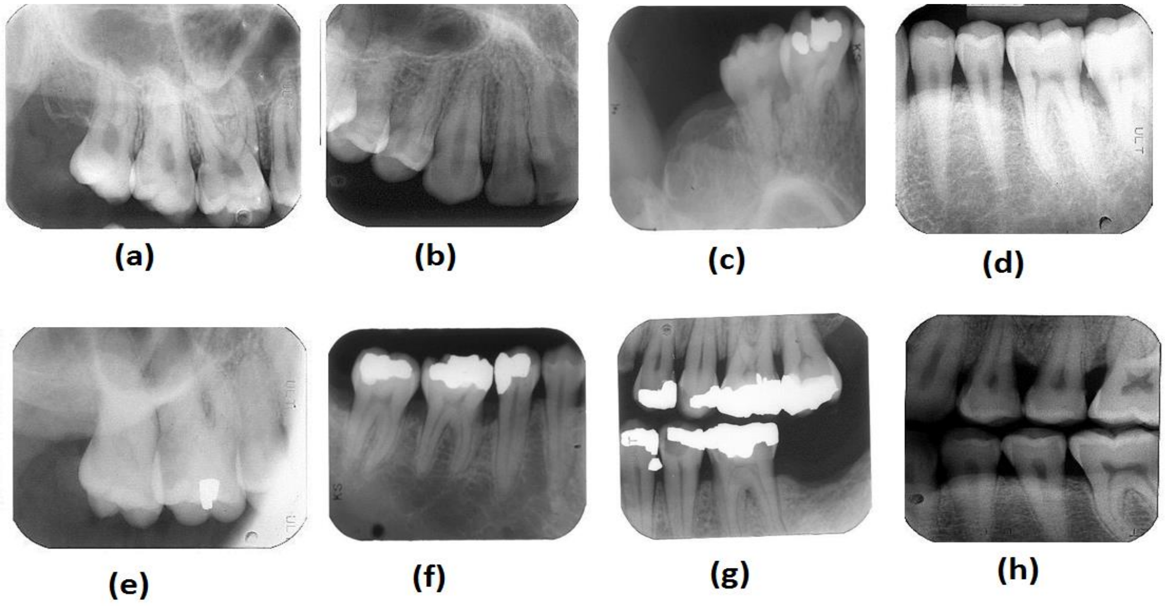


Figure 15: Examples of difficulties that could face dental image segmentation.

Most techniques are devoted to develop a fully automated segmentation. Jain and Chen [34] proposed a semi-automated method based on integral projection and can be used with bitewing and panoramic views. Nomir and Abdel-Mottaleb [35] proposed an automated method based on iterative adaptive thresholding followed by an integral projection. It can be used for bitewing views. Zhou and Abdel-Mottaleb [32] proposed an automated method based on mathematical morphology followed by an integral projection which can only be used for bitewing views. Hajsaid et al [32] proposed another automated method based on mathematical morphology and can be used for both bitewing and periapical views. Nassar et al. [36] introduced evaluation criteria to measure the performance of different segmentation methods.

Our approach starts by preprocessing the original teeth images by two step thresholding technique in order to obtain a good performance of teeth segmentation. Intially, iterative thresholding is applied on the teeth image to obtain binary image .This binary image is masked with the original image and adaptive thresholding is applied on the resultant masked image. Then, to separate each individual tooth we adapt the seam carving technique on the binarized images, using both horizontal and vertical seams.

The remainder of the section is organized as follows: Radiograph Preprocessing and Teeth Segmentation in sub-section 3.1.2. The experimental results are presented in sub-section 3.1.3.

Notations and Terminology

Some notations and definitions are introduced in this section.

Teeth segmentation: Teeth segmentation means partitioning an image into its constituent regions and extracting the objects of interest. In the context of ADIS, segmentation is a step required to identify the extent of teeth comprised in a digital image of a dental radiographic film.

Tooth segment: A tooth segment represents a Region of Interest (ROI) that contains distinctive features used in the subsequent steps of identification. A qualified ROI is defined as a rectangular area in the image of a dental film that bounds one tooth as shown in Figure 16.

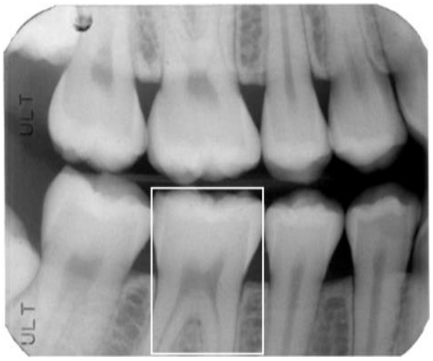


Figure 16: Local Segmentation

3.1.2 Radiograph Preprocessing and Teeth Segmentation

Our teeth segmentation algorithm consists of three main components: preprocessing, horizontal and vertical seam carving, as shown in Figure 17. In the preprocessing step we use the technique introduced in [19] which was adapted from [35]. This technique starts by applying an iterative thresholding followed by an adaptive thresholding to binarize the teeth images. Then, we propose to use a horizontal seam to separate the upper jaw from the lower jaw. Finally, we apply vertical seams to extract each tooth from their corresponding jaw (both the upper and lower).

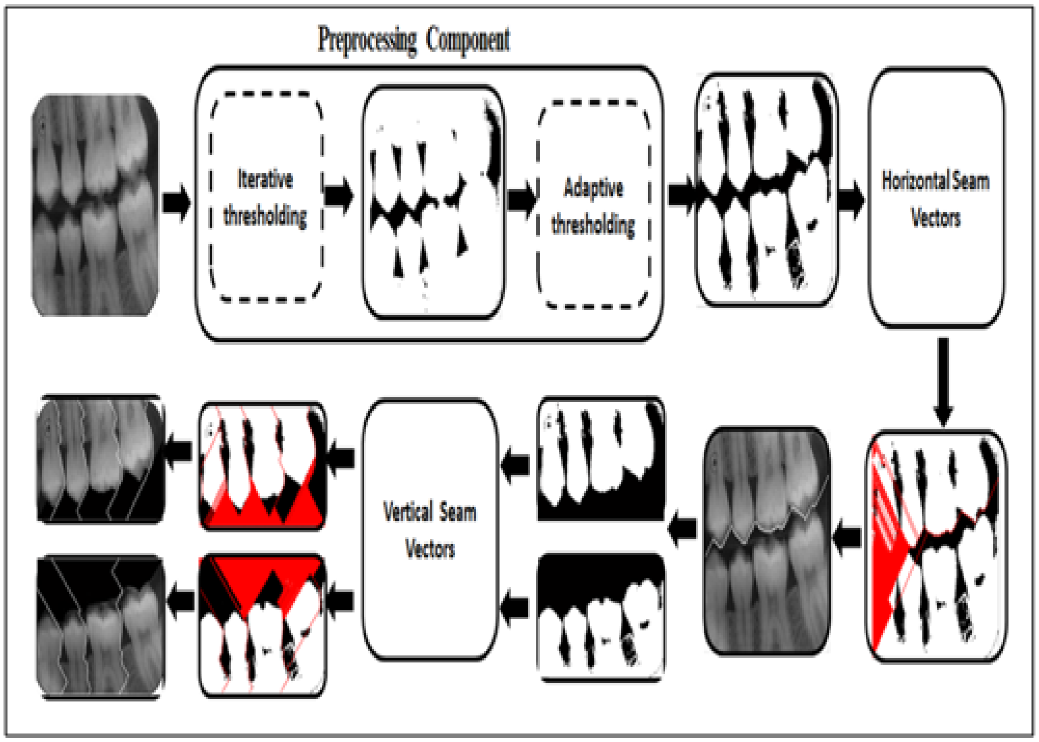


Figure 17: Block diagram of the teeth segmentation algorithm.

3.1.2.1 Radiograph preprocessing

Segmentation in dental X-Ray images is a very challenging task because of the variations from image to image which are caused by discrepancy in the exposure. Moreover, deviations from one region to another within the same image increase the difficulty in teeth segmentation. Typically, Dental X-Ray images have three different regions. The background, bone areas, and the teeth. Among these, background region corresponds to the soft tissues which has the lowest intensities, whereas bone areas and the teeth have the middle and highest intensities, respectively. It is difficult to use a single threshold for the whole image because the intensities of bone areas are very close to the intensities of the teeth [28].

We used the technique introduced in [35] to pre-process the dental images which starts with an iterative thresholding followed by an adaptive thresholding introduced in [51] [52]. To segment the teeth from both the background and the bone areas, our segmentation algorithm involves two steps, iterative thresholding succeeded by the adaptive thresholding:

- **Iterative thresholding**

Iterative thresholding in our work is refinement of the approach used in [53]. In [53], in order to account for the small changes that are expected over a range of subjects, iterative thresholding was employed to automatically segment CT lung images. In their work, it was presumed that image volume essentially consists of two types of voxels, body and non-body voxels. In order to start their algorithm, an initial threshold is chosen based on the properties of two volumes i.e., pure air and chest wall/body. On the other hand, in the proposed work, the initial threshold is selected by first detecting the edges in the original teeth images using canny edge detection [54]. Then morphological dilation is applied to get the pixels around the location of edges [55]. This step aids in selecting the areas of higher contrast in the original image. Let f (i, j) denotes the gray value at location (i, j). In order to compute the first threshold (i.e.; $\text{threshold} = \mathrm{T}_1$), initially an arbitrary threshold is selected to segment the resulting dental image that is obtained from the masking the original image with the dilated image. Then compute μ_0^1 and μ_B^1 that denotes the mean gray values for the teeth and non-teeth regions respectively, based on which T1 can be calculated by using the following equations in which i is set to be 1.

$$\mu_0^i = \frac{\sum_{(i,j)\ \varepsilon\ dental} f(i,j)}{\#\ dental}$$

$$\mu_B^i = \frac{\sum_{(i,j)\ \varepsilon\ background} f(i,j)}{\#\ background\_pixels}$$

$$T_i = \frac{\mu_0^i + \mu_B^i}{2}$$

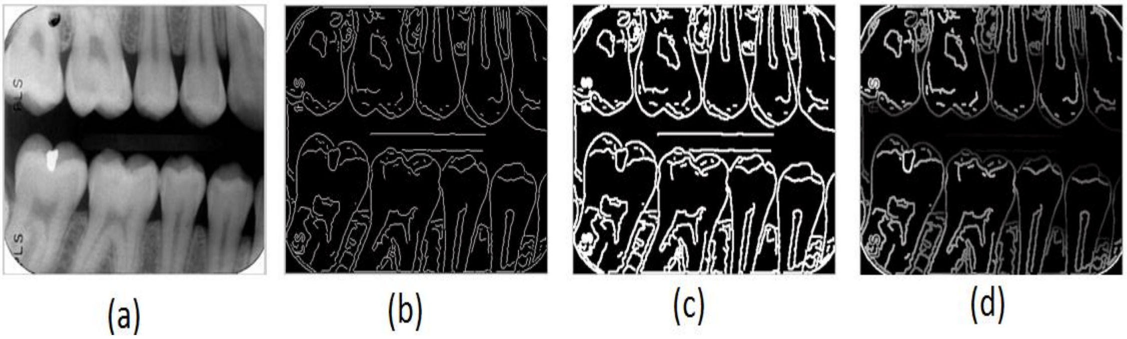


Figure 18: (a) Original image; (b) binary edge image; (c) dilated edge image; (d) resulting masking image.

Figure 18(a) shows the original teeth image and Figure 18(b), Figure 18(c), and Figure 18(d) shows the results of applying canny edge detection, morphological dilation operations, and the masking the original image with the dilated image respectively. Then, by using the threshold value T_1 repeat the above steps to obtain the threshold T2 (setting i=2 in the above equations).

Typically, half of the pixels in the original teeth image belong to the teeth and other half includes background and bones. Now, T_2 is used as an initial threshold to start the iterative thresholding scheme. The iterative threshold step is continued until the threshold remains unchanged i.e., $\mathrm{T}_\mathrm{i} = \mathrm{T}_{\mathrm{i}+1}$. The final threshold obtained after convergence segments the image in to teeth and background regions. It is observed that four to seven steps are enough to obtain convergence. The resulting binary image obtained from original teeth image is depicted in the Figure 19(b). Figure 19(c) shows the result of masking the original image with the binary image obtained from iterative thresholding. i.e., the ones in the binary image have been replaced with the original image gray values, and the zeros are kept. This is the image to which adaptive thresholding is applied.

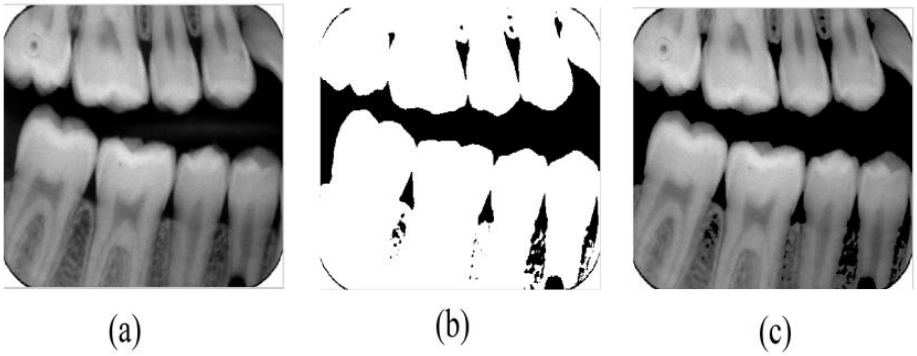


Figure 19: (a) The original image; (b) result of iterative thresholding; (c) the original image after masking by the binary image.

From the experimental results, applying the iterative thresholding technique to segment the teeth from the background sometimes did not lead to good results, e.g., see Figure 20(b) show some of the bad images that might be obtained by employing iterative segmentation scheme on the original image to separate the teeth from the background while Figure 20(d) shows more accurate result that can be obtained by employing iterative segmentation scheme on the dilated image. Some of the teeth parts are missing in the Figure 20(b), since the estimated average threshold value is biased to the gray level of the background. It is clearly evident from the Figures that the results obtained in Figure 20(d) are more accurate.

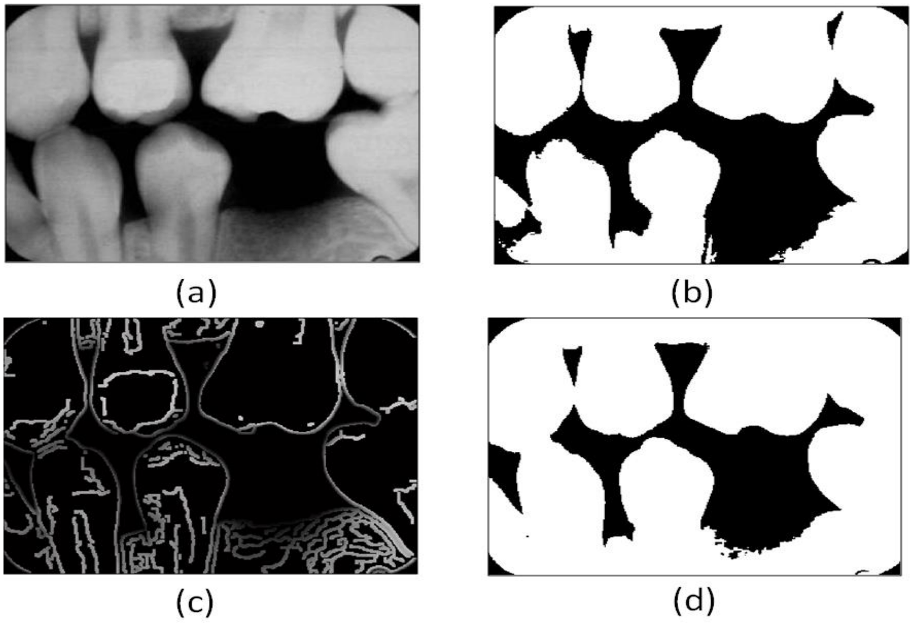


Figure 20: (a) Original image; (b) the result from the original image after iterative thresholding; (c) dilated image; (d) the result from the dilated image after iterative thresholding.

- **Adaptive thresholding**

Applying iterative thresholding to segment teeth from the background sometime does not provide a good result, e.g., see Figure 21(c). Applying adaptive thresholding to the result of masking the original image Figure 21(d) always produces more accurate results. We can notice that in Figure 21(e) the teeth are separated better than in Figure 21(c). As aforementioned this is due to the closeness in the intensities of the pixels in the bone areas and teeth regions.

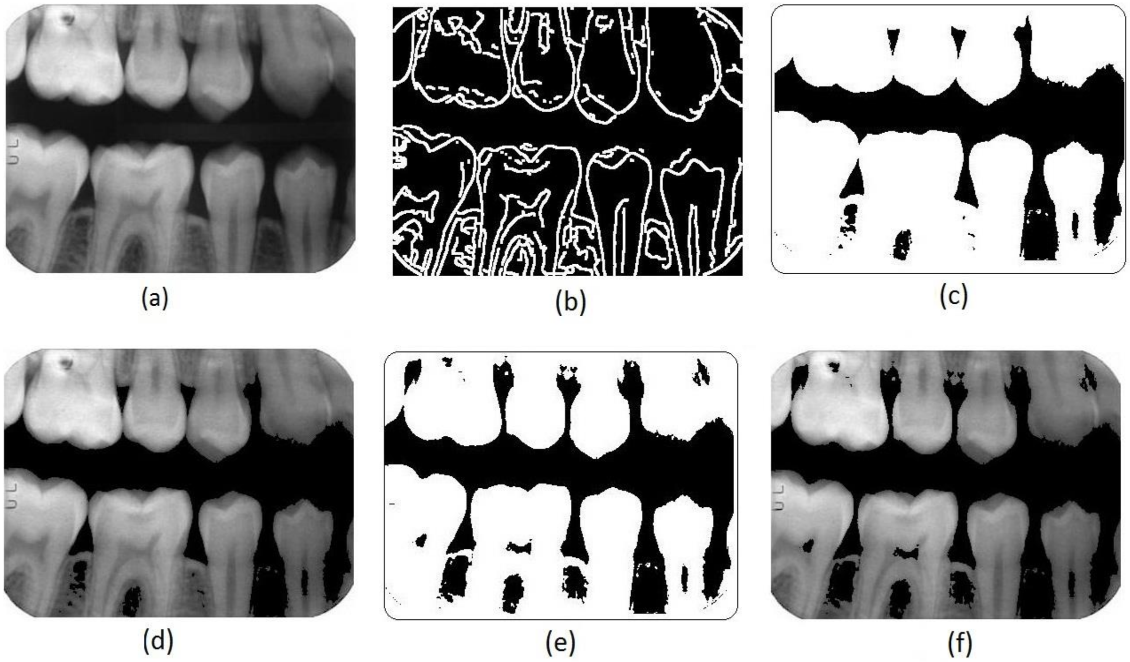


Figure 21: (a) Original image; (b) dilated edge image; (c) the result from the dilated image after iterative thresholding; (d) the original image after masking by the binary image; (e) the result after adaptive thresholding; the original image after masking by the binarized image.

Adaptive thresholding actually threshold’s the pixel at the center of a window according to the average gray value of the non-zero pixels inside this window. The following is applied to each pixel of the image that is acquired from masking the original image with the binary image obtained from the iterative thresholding step. A window of size i ×j pixels is used here.

$$C(i,j) = \frac{\sum_{S=-\frac{I}{2}}^{I/2} \sum_{t=-\frac{J}{2}}^{J/2} \quad f(i+s, j+t)}{\#\, non\ zero\ pixels}$$

If the center pixel value of the window is $C(i,j)$, then

$$C'(i,j) = \begin{cases} 1, & C(i,j) \geq \ T'(i,j) \\ 0, & otherwise \end{cases}$$

30 images from the database are randomly selected to be used as our training set and T'(i, j) = 0.9 × T (i, j) was observed to be appropriate threshold value.

3.1.2.2 Teeth segmentation

Our proposed approach for teeth segmentation is based on the technique that was used for resizing digital images [56]. In this technique the energy of each pixel in the image is computed before resizing. A seam is a continuous series of pixels that is a connected path of low energy pixels in an image that can be either vertical or horizontal; a vertical seam is a path of pixels connected from top to bottom in an image with one pixel in each row; a horizontal seam is similar, except that the connection is from left to right. The energy indicates the importance of pixels. There are several energy functions, while the simplest is the image gradient magnitude.

After defining the energy of each pixel, lower energy pixels are removed or duplicated in order to decrease or increase the image size. In this proposed approach, we compute the seam carving based on the energy functions of the pixel intensity value of the dental image that is a binary image obtained from the preprocessing step as described in Section 3.1.2.1. So the energy function can take either 0 for the pixels in background region or 1 for the pixels in teeth region instead of the gradient as in [56]. We found that the seam carving technique cannot work well when applied to the original teeth image. The preprocessing step is essential for the seam carving method to work properly for our problem.

3.1.2.2.1 Separating the Upper and the Lower Jaws

The energy of a seam is defined as the sum of all pixels' energy in the seam. We use the dynamic programming technique that is also employed in [56] to look for the seam with minimal energy. In order to separate the upper and the lower jaws from their corresponding binary dental image, we first compute the horizontal seam by establishing the cumulative energy function matrix using the following equation:

$$M(x,y) = E(x,y) + \min\{M(x-1,y-1), M(x,y-1), M(x+1,y-1)\}$$

Where E(x,y) denotes the energy of the pixel located in the image at (x,y). This function considers the three possible paths to reach cell (x,y), i.e., horizontal, vertical, and diagonal. When (x,y) is on the edge, some of these directions may not be possible and they are omitted from the formula.

These cumulative energy functions are used for the dynamic programming algorithm to find the minimum cumulative energy path. In the case of a horizontal seam, the first step is to traverse the image from left to right and compute the cumulative minimum energy M for all possible connected seams for each entry (x, y), as shown in Figure 22(a).

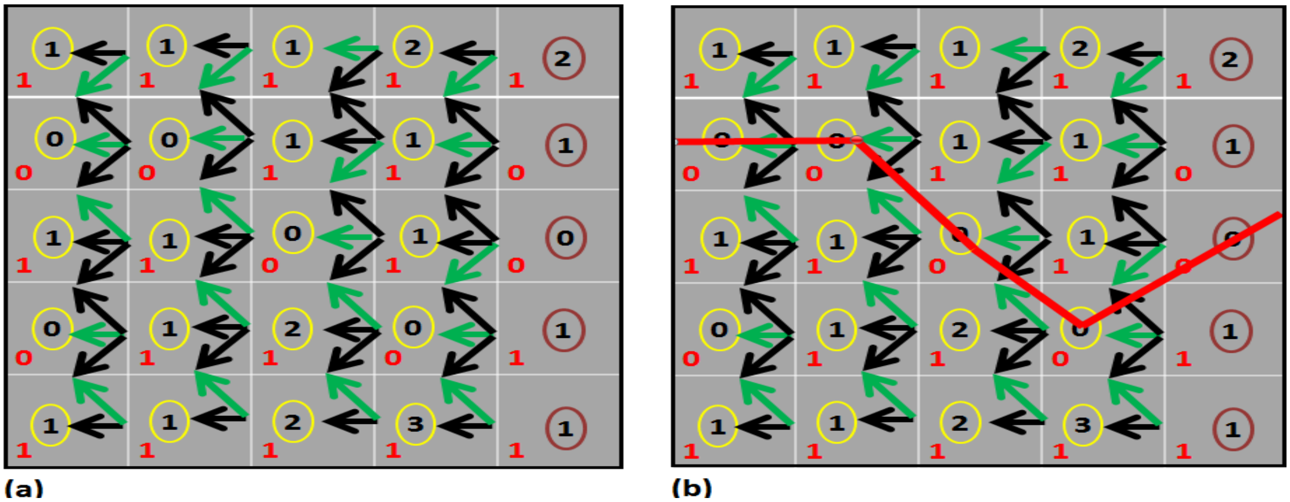


Figure 22: (a) Each square represents a pixel, with the bottom-left value in red representing the energy value for that pixel. The value in black represents the cumulative sum of energies leading up that pixel. (b) Trace the seam from the last column by following the green arrows.

Then the pixel at last column is found in the cumulative matrix whose cumulative energy is minimum. Following this, we backtrack from this pixel to obtain an optimal horizontal seam that separates the upper and the lower jaws, as displayed in Figure 23(b) and Figure 23(b).

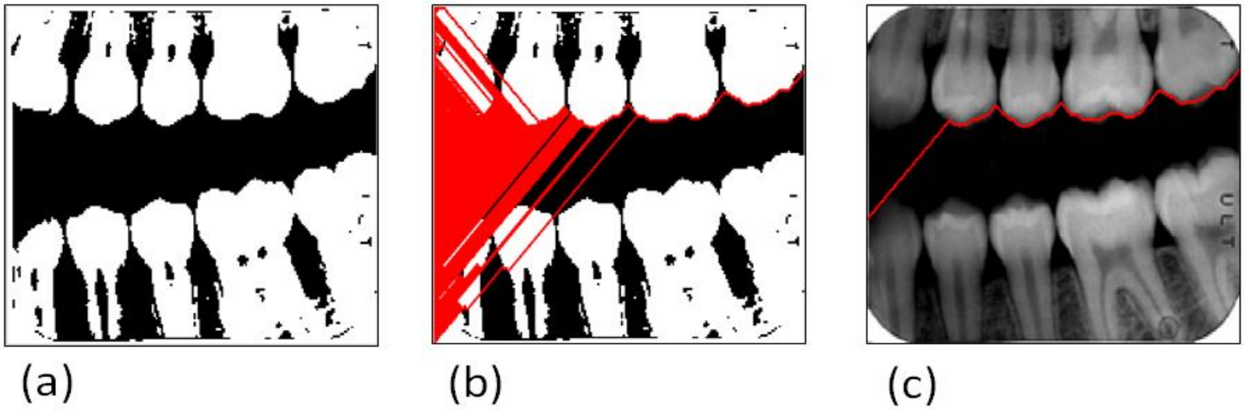


Figure 23: (a) Result of adaptive thresholding; (b) Horizontal seams carving; (c) Separating the upper and the lower jaws.

3.1.2.2.2 Segmentation of Each Individual Tooth

To separate each tooth from either an upper or lower jaw, we use vertical seams which are similar to the horizontal seam used for separating the jaws. The goal is to find the seams that separate adjacent teeth in to individual ones. The energy of a seam is defined as the sum of all binarized pixels values in the seam. We use the dynamic programming technique as we have done with horizontal seam to find the seam with minimum energy. In order to separate each tooth from its corresponding jaw (upper or lower), we compute the vertical seams by computing the cumulative energy matrix.

The pixels in the cumulative matrix at last row, whose cumulative energy are minimal, are located. After performing this, we backtrack from these pixels to obtain optimal vertical seams (i.e. more than one seam that depends on the number of teeth per jaw) that separates adjacent teeth. Figure 24 and Figure 25 show the extraction of individual teeth from its upper jaw and the lower jaw, respectively.

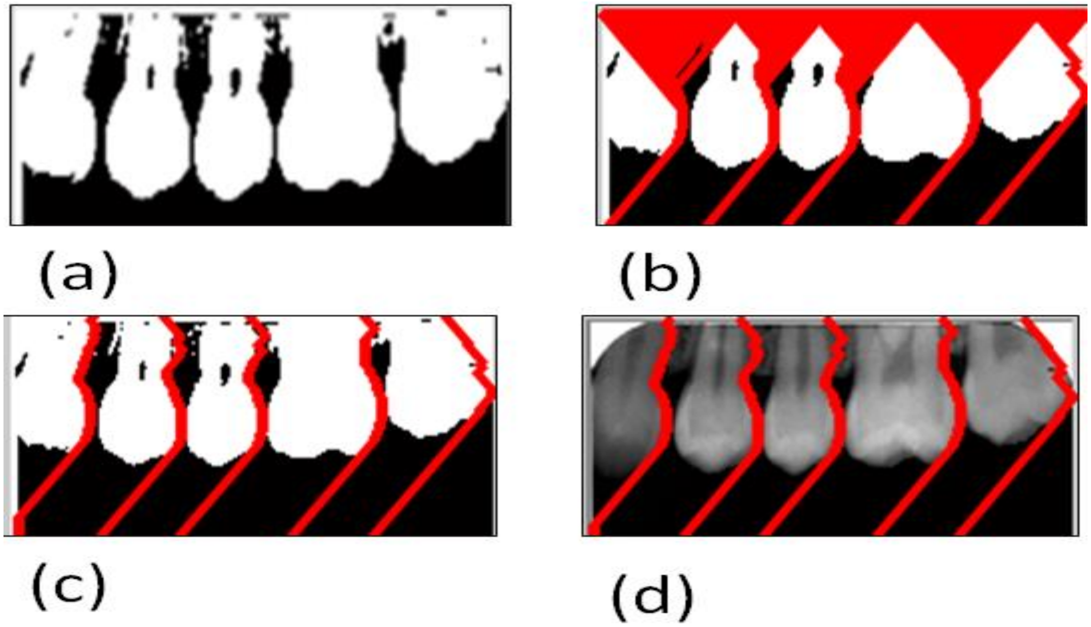


Figure 24: Upper jaw: (a) A result of image pre-processing; (b) Vertical seams carving; (c) Detected separating lines; (d) Detected separating lines overlaid over the original image.

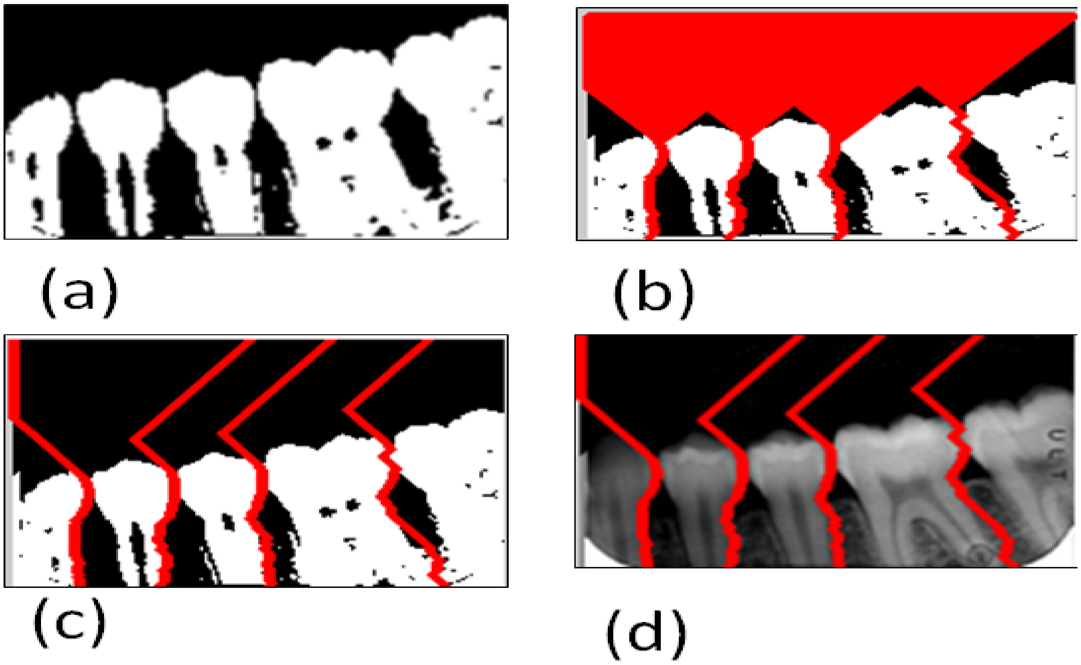


Figure 25: Lower jaw: (a) A result of image pre-processing; (b) Vertical seams carving; (c) Detected separating lines overlaid over the preprocessing image; (d) Detected separating lines overlaid over the original image.

3.1.3 Experimental results

Our proposed segmentation technique was applied to two test sets of 500 bitewing and 130 periapical dental radiographic films selected from large dental radiographic databases [26] [27]. For the bitewing and periapical radiographic films, we obtained accurate segmented results in most images. Table 2 gives the number of films/teeth we used in our test. Some teeth segmentation/separation results are shown in Figure 26.

Table 2: The segmentation results

	Bitewing		Upper Periapical	Lower Periapical
Total number of dental films	500		66	64
Total number of teeth	**Upper jaw**	**Lower jaw**	250	220
	1833	1692		

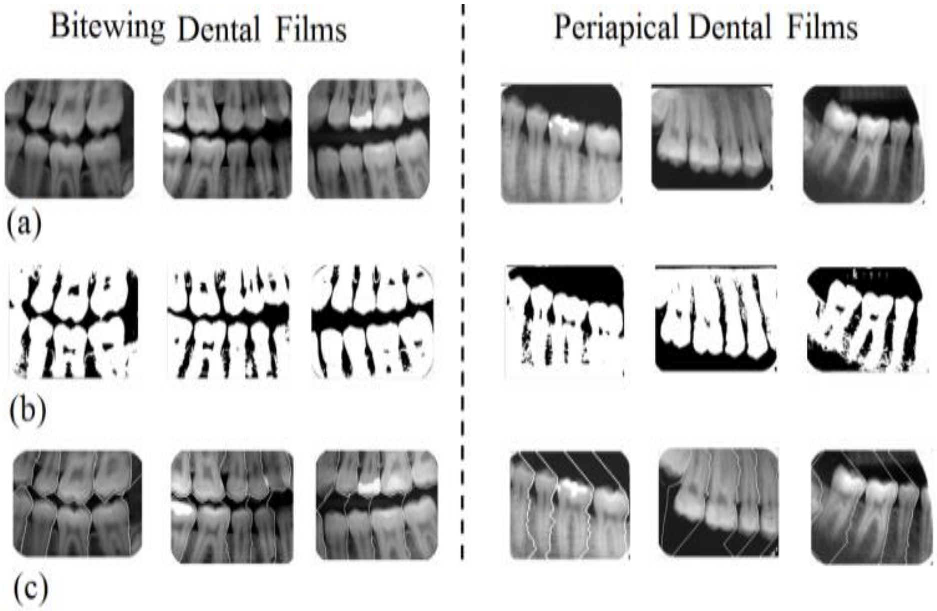


Figure 26: Teeth segmentation and separation results; (a) the original images; (b) the images after pre-processing; (c) the detected separating lines overlaid over the original image.

A few teeth segmentation/separation results are demonstrated in Figure 27. The cases where teeth were not correctly or partially separated are due to the poor quality of the images, when the teeth at the borders of the images, and when the bones were considered as part of the teeth..

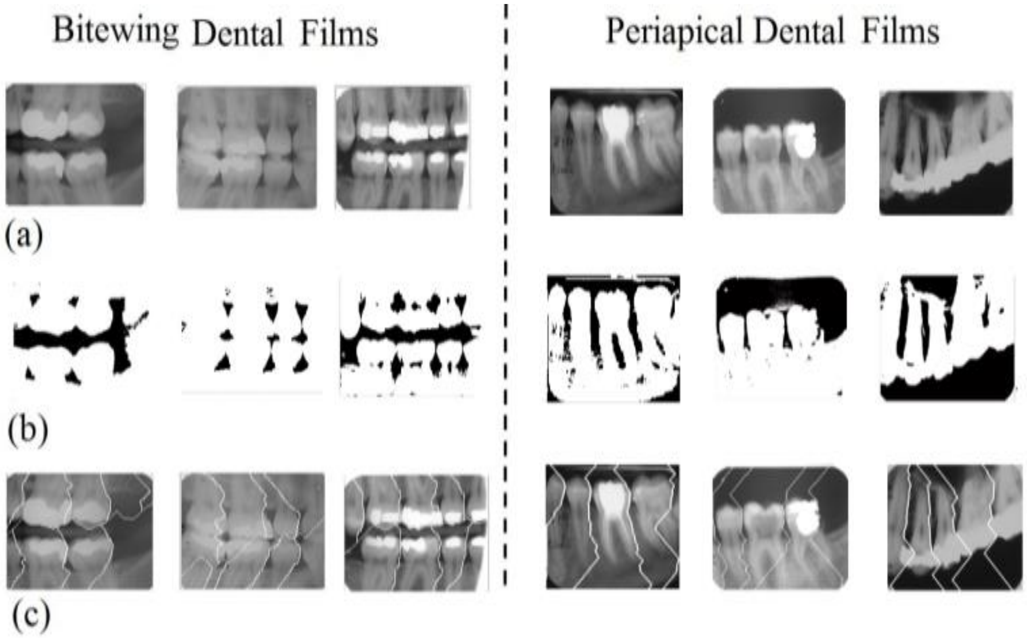


Figure 27: Teeth segmentation and separation results; (a) the original images; (b) the images after pre-processing; (c) the detected separating lines overlaid over the original image.

Figure 28 shows the comparison of teeth segmentation output from the two dental film (a, b) using state of the art algorithms [35,34] and our proposed method. It is clear that the segmentation results, (c, d), obtained with integral projection in [35], or the results, (e, f), obtained with a semi-automated method in [34], are not able to separate the teeth well. And also, the tooth separation does not fit tightly around the actual boundary of the teeth. However, the results, (g, h), obtained by our proposed method are significantly better and the regions of interest for most teeth are bounded tightly.

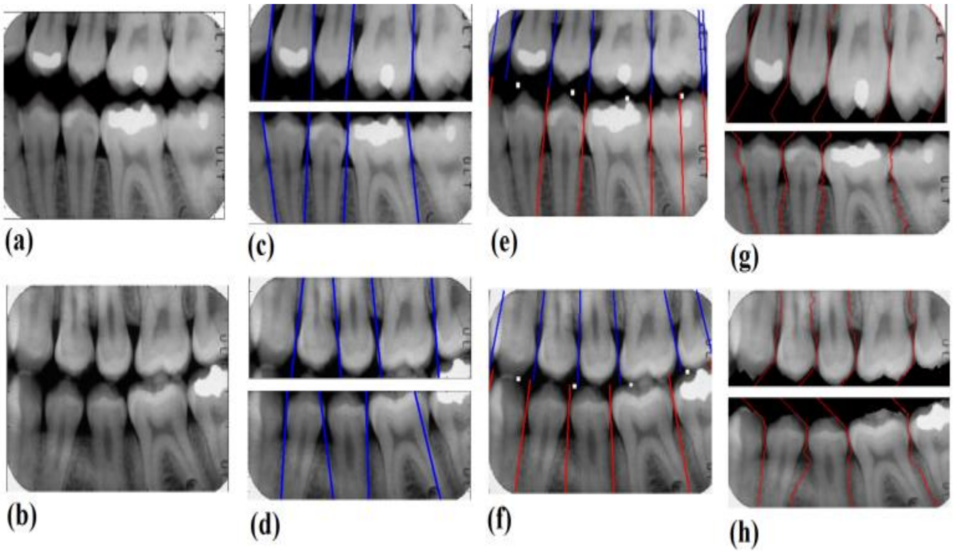


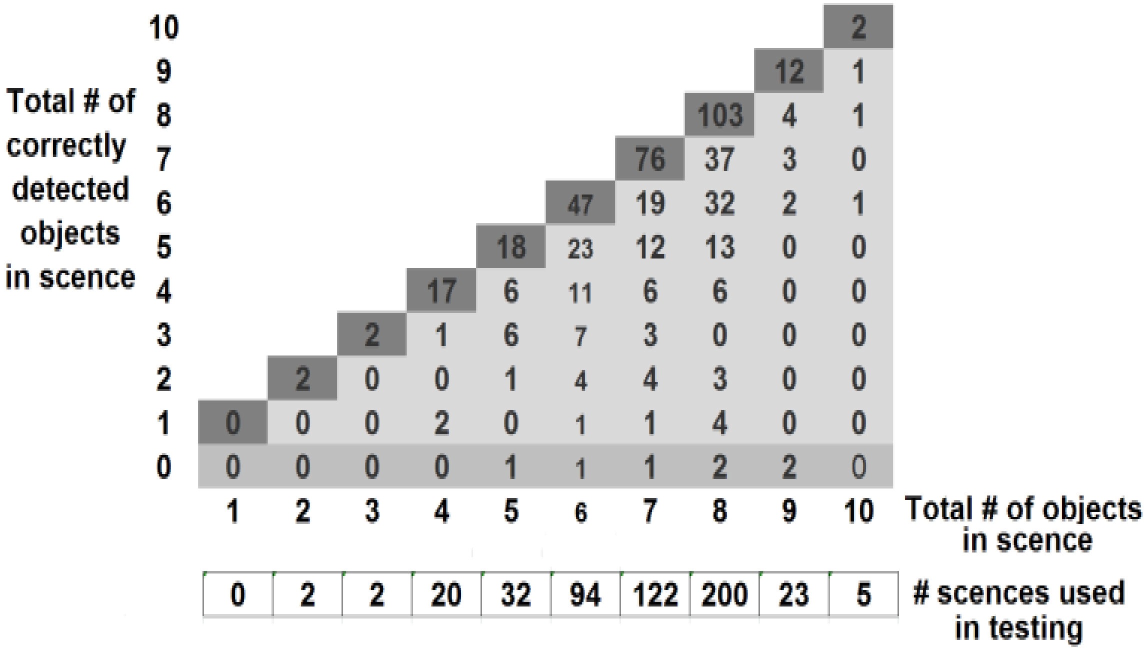
Figure 28: Comparison of teeth separation results; (a,b) input images; (c,d) the detected separating lines overlaid over the original images by using algorithm [35]. (e,f) the detected separating lines by using algorithm [34]. (g,h) the detected separating lines by using our proposed algorithm. One can see that our method gives better teeth separations than those in [35,34].

3.1.3.1 Teeth Count

To have a quantitative comparison, we use the performance evaluation methodology in [36], and compare the performance of our proposed method with the methods in [34] [35] [32] [28], using the same radiographic databases for the empirical assessment. Optimality rate and failure rate [36] are the two measures used to quantify the performance of the segmentation. The optimality rate is computed by the summation of product of the main diagonal with the corresponding total number of dental films in the scene, divided by the summation of the square of the total number of dental films used in testing. The failure rate is computed by the summation of product of the total number of incorrectly segmented of dental films with the corresponding total number of dental films in the scene, divided by the summation of the square of the total number of dental films used in testing.

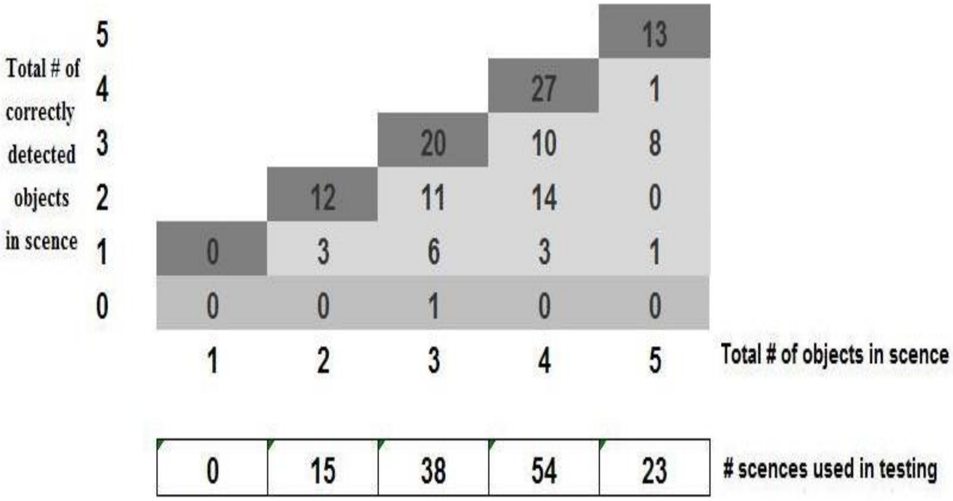
All films in the bitewing radiographic set contain up to 10 teeth per film, and films in the periapical radiographic set contain up to 5 teeth per film. In counting the number of correctly detected teeth in a film, we visually inspect the outcome of segmentation for each film using a simple rule of tooth containment within each segment in a given film. It is worth noting that, the bitewing set of dental images used in [34] [35] [32] [28] for testing is the same as the one used for testing our segmentation approach, and the periapical set of dental images used in [28] is the same as the one used for testing our segmentation method.

Testing results for the algorithm that we proposed for bitewing radiographic set and periapical set are summarized in Figure. 29 and 30, respectively.



Total # of correctly detected objects in scence										
10										2
9									12	1
8								103	4	1
7							76	37	3	0
6						47	19	32	2	1
5					18	23	12	13	0	0
4				17	6	11	6	6	0	0
3			2	1	6	7	3	0	0	0
2		2	0	0	1	4	4	3	0	0
1	0	0	0	2	0	1	1	4	0	0
0	0	0	0	0	1	1	1	2	2	0
Total # of objects in scence	1	2	3	4	5	6	7	8	9	10
# scences used in testing	0	2	2	20	32	94	122	200	23	5

Figure 29: Testing results for our teeth segmentation algorithm on bitewing views.



Total # of correctly detected objects in scence					
5					13
4				27	1
3			20	10	8
2		12	11	14	0
1	0	3	6	3	1
0	0	0	1	0	0
Total # of objects in scence	1	2	3	4	5
# scences used in testing	0	15	38	54	23

Figure 30: Testing results for our teeth segmentation algorithm on periapical views.

We have obtained an optimality rate of 54.02%, which is superior to all existing fully automated dental segmentation algorithms in the literature, and a failure rate of 1.05%, which is the lower than any previous teeth segmentation algorithms in bitewing views. See Table 3 for details.

Table 3
Comparison between the failure rates and optimality rates of the variant teeth segmentation algorithms with bitewing views.

Algorithm	Our Algorithm	Hajsaid et al [28]	Jain and Chen* [34]	Nomir and Abdel-Mottaleb [35]	Zhou and Abdel-Mottaleb [32]
Failure rate	1.05%	1.27%	2.61%	11.18%	3.47%
Optimality rate	54.02%	14.96%	61.85%	19.24%	28.84%

* denotes a semi-automated dental segmentation algorithm.

For the periapical view, we have obtained a high optimality rate of 58.13% and a low failure rate of 0.74%, better than the 27.18% and 8.88%, respectively, obtained by Hajsaid et al. [28], which is the state-of-the-art automated teeth segmentation algorithm on periapical view.

3.1.3.2 Time Complexity

To compare between the time complexities of the proposed algorithm and those proposed in [35] and [34], we used 40 bitewing films with different dimensions. We invoked MATLAB® implementations of each of the algorithms on MATLAB® implementation running on a 2.66 GHz 4 GB RAM Intel® Core(TM) Quad CPU PC platform. Table 4 summarizes the outcome of the time complexity comparison between the three teeth segmentation algorithms; h is the image height, w is the image width, and n is the size of window used in adaptive thresholding. Figure 30 records the execution times of the three teeth segmentation algorithms for each of the 40 bitewing films. Based on Figure 31 and Table 4, we observe that: (i) the proposed algorithm is significantly faster than the algorithm [35] (ii) the difference in the average execution time between the proposed algorithm, algorithm [34], is small, (iii) the time complexity variation of teeth segmentation of proposed approach from one dental film to another is very small.

Table 4: Comparison between the time complexities of the three teeth segmentation algorithms.

Algorithm	**Our Algorithm**	**Jain and Chen* [34]**	**Nomir and Abdel-Mottaleb [35]**
Average time complexity	2.59 sec	0.94 sec	9.39 sec
time complexity order	O(nhw)	O(hw)	O(nhw)

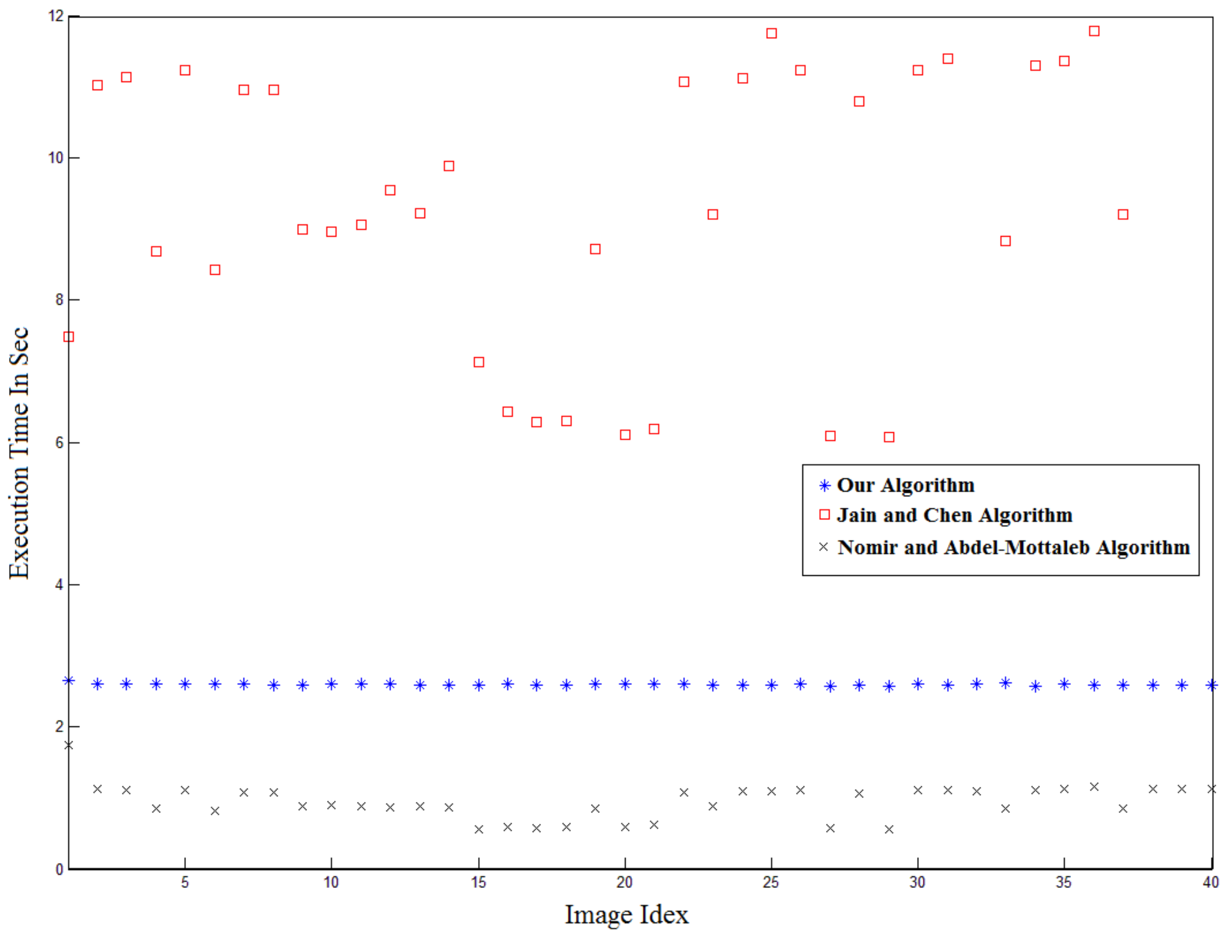


Figure 31: Computational complexity comparison for various algorithms.

3.2 Teeth Labeling

Forensic radiography is part of forensic medicine, which is concerned with identifying people using the postmortem radiological images of different parts of the body including skeleton, skull and teeth. Radiological images of the decedent's body are compared with antemortem records of a missing person to evaluate similarities between them. Traditionally, dental-based identification relies on information such as missing teeth and dental works [57]. Nowadays with the advancements in dentistry and care of teeth by people, these methods might not be reliable; hence, developing new methods that use inherent dental features for identification is important [2]. In order to build an automated dental identification system (ADIS), we need to extract features from the teeth in the dental images of missing people and archive them in a database. During retrieval, the features for each tooth in the query image need to be extracted and compared to those stored in the database. If we limit the comparison of the teeth to the ones that have the same index number, this will help limit the search space and increase the robustness of the system. In this chapter, we present an algorithm for the classification and numbering of teeth to be used during archiving and retrieval in/from the database. We propose an appearance-based approach using Orthogonal Locality Preserving projection Algorithm (OLPP) starts by classifying each tooth in dental film (bitewing and periapical film) then it considers the relationship between the neighboring teeth to correct any initial misclassification.

3.2.1 Introduction

An important problem in automated dental identification, is automatic classification of teeth into four classes (molars, premolars, canines, and incisors) as part of creating a data structure that guides tooth-to-tooth matching, thus avoiding illogical comparisons that inefficiently consume the limited computational resources and may mislead decision-making. We tackle this problem by first using low computational-cost, appearance-based features to assign an initial class, and second a string matching technique, based on teeth neighborhood rules, is applied to validate initial teeth-classes and hence assign each tooth a number corresponding to its location in the dental chart. The proposed approach works very fast, as it does not need any complex feature extraction steps, and achieves accurate teeth labeling even in the absent of one tooth. The remainder of the section is organized as follows: Section 3.2.2 elaborate on the input segment normalization steps, followed by teeth Numbering systems for adults in section 3.2.3. The proposed technique is in section 3.3.3. The experimental results are presented in section 3.2.4.

3.2.2 View Normalization

The View Normalization and resizing step comes after the teeth segmentation stage, whose outcomes do not have standard view in terms of scale and rotation. The view normalization is an essential step to improve and overcome variations in orientation and scale. To normalize the view of the dental segmentation outcome is to ensure that the surface of tooth crown appears horizontally as much as possible. This can be done by finding the appropriate rotation angle that is corresponding to strongest mean of the vertical orientations. The view normalization task proceeds as follows [58]:

- An edge image is calculated from the input image using Canny edge detection, as shown in Figure 32(b). The choice of the width of the smoothing kernel is such that to avoid any excessive number of edge points due to intensity variations not related to the tooth surfaces.
- Using the gradients of the input image, the orientation at each of the remaining edge points is estimated, and accordingly classified as either horizontal or vertical. Hence a histogram of orientations of the vertical edge points is computed, as shown in Figure 32(c).
- The mean of the vertical orientations is computed, and its negative is used as the rotation angle for the input image such that the lateral surfaces of the tooth appear predominantly vertical, as shown in Figure 32(d).
- By analyzing the vertical and horizontal span of the tooth, the tooth ROI is confined to a square area, whose edge is just above the predominantly horizontal line segment belonging to the crown and such that the lateral sides of the tooth are also contained in the square area as shown in Figure 32(e).
- Do not a like other some of teeth segmentation algorithms, they can’ not discriminate between upper and lower teeth, the proposed approach is able to annotate segmented teeth as either upper or lower during the teeth segmentation stage.

- The last step is to resize the tooth ROI image to a standard size, as shown in Figure 32(f). This step of rough alignment is performed once, and experimentally it was proven to enhance the performance, we used nxn = 32x32.

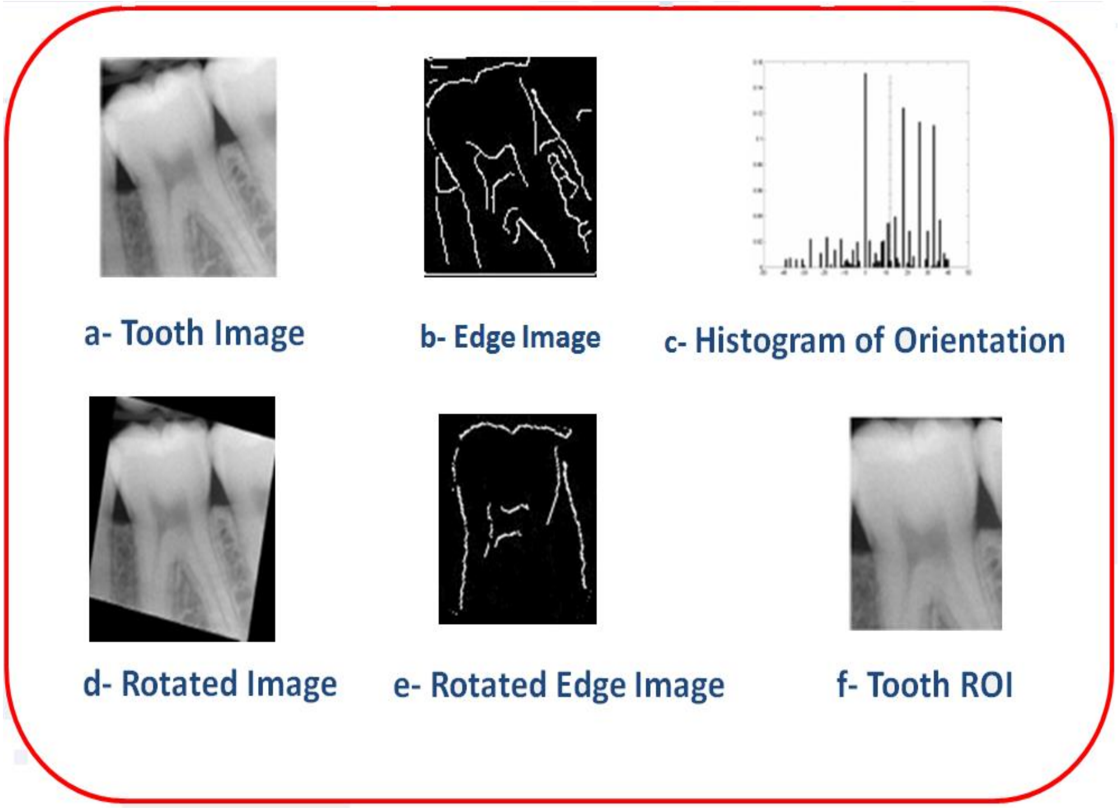


Figure 32: Illustration of the orientation adjustment and ROI confinement steps of a segmented tooth.

3.2.3 Teeth numbering systems for adult:

The adult dentition contains 32 teeth, 16 teeth in each jaw. We divide the jaws into four equal quadrants which each quadrant contains eight teeth, three molars, two premolars (bicuspid), one canine (cuspid), and two incisors. Various numbering systems for adults have been developed in order to have a standard way of referring to particular teeth (there are more than 32 different systems). Among these, two are commonly in use in the US today: 1) Universal Numbering System, which has been adopted by the American Dental Association, and 2) Palmer Notation Method, which is used by some orthodontists, pedodontists (child dental specialist), and oral surgeons. While internationally the two-digit FDI World Dental Federation notation is widely used [59]. Table 5 shows the difference between those systems. In the proposed approach, we used the Palmer Notation Method.

Table 5: Different teeth numbering systems

Universal Numbering System															
Upper (Maxillary) Left								**Upper (Maxillary) Right**							
16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
M	M	M	P	P	C	I	I	I	I	C	P	P	M	M	M
M	M	M	P	P	C	I	I	I	I	C	P	P	M	M	M
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Lower (Mandibular) Left								**Lower (Mandibular) Right**							

Palmer Notation															
8	7	6	5	4	3	2	1	1	2	3	4	5	6	7	8
M8	M7	M6	P5	P4	C3	I2	I1	I1	I2	C3	P4	P5	M6	M7	M8

FDI Two-Digit Notation															
28	27	26	25	24	23	22	21	11	12	13	14	15	16	17	18
38	37	36	35	34	33	32	31	41	42	43	44	45	46	47	48

3.2.4 Proposed technique

The proposed approach that is used for teeth classification is appearance based using Orthogonal Locality Preserving Projections (OLPP) algorithm [60].

3.2.4.1 Theoretical conception of OLPP stages

To overcome the problem of singularity that occurs if the number of the teeth images in the training set is much smaller than the dimension of the tooth image vector, we apply PCA to project the teeth into the sub-space without losing any information, so that the number of images in the training set will exceed the PCA coefficients of the image and it becomes nonsingular. In addition the PCA projected image is more robust to noise than the preprocessed one. In brief, the algorithm of OLPP to obtain LaplacianTeeth is started as follows:

- **Preprocessing PCA projection:** The size of the tooth segment is (n x m), which might cause problem especially when n is very large while the intrinsic dimensionality of the data is much lower. To overcome this one has to consider methods of dimensionality reduction that allows to represent the data in a lower dimensional space. Therefore dimension of the image should be reduced using one of the techniques exiting in the literature such as principle component analysis (PCA). Then throw away the eigenvectors values whose corresponding eigenvalues are too small in order to remain the eigenvector that are statically uncorrelated. The training set of teeth images t_i is projected into the PCA subspace by throwing away the components corresponding to zero eigenvalue. The transformation matrix of PCA is denoted by W_{PCA}. The features extracted with PCA projection are statistically uncorrelated.
- **Constructing the adjacency graph:** Let $X=[t_1,t_2,...,t_N]$ be a set of training set of teeth images. Let G denote a graph with N nodes where the ith node corresponds to the tooth image t_i. If nodes t_i and t_j are connected then the (i, j) and (j, i) elements of the nearest-neighbor matrix get the values $S_{ji} = S_{ij} = e^{-(||t_i - t_j||)2} \,/\, t$ where t is a suitable constant. Otherwise they are zero. This is called the Heat kernel approach. The weight matrix S of graph G models the local structure of the teeth manifold.
- **Computing the Orthogonal Basis Functions:** We define D as a diagonal matrix whose entries are column (or row, since S is symmetric) sums of S, $D_{ii} = \sum_j S_{ji}$ We also define L = D - S, which is called Laplacian matrix in spectral graph theory. Let $\{a_1, a_2, . . . , a_{k-1}\}$ be the orthogonal basis vectors, we define:

  $A_{k-1} = [a_1, a_2, . . . , a_{k-1}]$, $B_{k-1} = [A_{k-1}]^T \ (XDX^T)^{-1} \ A_{k-1}$. The orthogonal basis vectors $\{a_1, a_2, .. , a_{k-1}\}$ can be computed as follows: compute a_1 as the eigenvector of $(XDX^T)^{-1}$ XLX^T associated with the smallest eigenvalue. Compute a_k as the Eigenvector of:

  $$M^{(K)} = \{I - (XDT^T)^{-1} A^{(K-1)}. [B^{(K-1)}]^{-1} [A^{(K-1)}]^T\}. (XDT^T)^{-1}\ XLX^T$$

  associated with the smallest eigenvalue of $M^{(K)}$.

- **OLPP Embedding:**

  Let W_{OLPP} = [a_1, a_2, . . . a_L], the embedding is as follows: $x \rightarrow y = W^T x$ where $W = W_{PCA} W_{OLPP}$. Where y is a l-dimensional representation of the teeth image x, and W is the transformation matrix.

- **Classification Process:**

  The one that is closer to the projected subject image is considered the most likely candidate.

3.2.4.2 Initial Teeth Classification

We will tackle this problem by first using low computational-cost, appearance-based features (OLPP) to assign an initial class, and second a string matching technique, based on teeth neighborhood rules, is applied to validate initial teeth-classes and hence assign each tooth a number corresponding to its location in the dental chart as described in Figure 33. We propose Orthogonal Locality Preserving Projections (OLPP) algorithm approach to the problem of classifying a tooth into: Incisor, Canine, Premolar, or Molar. We seek to establish a LaplacianTeeth Space for each of the four teeth classes, namely the LaplacianMolar Space, the LaplacianPremolar, the LaplacianCanine, and the LaplacianIncisor Space. To construct these spaces we use sets of exemplar teeth images from each of the four classes.

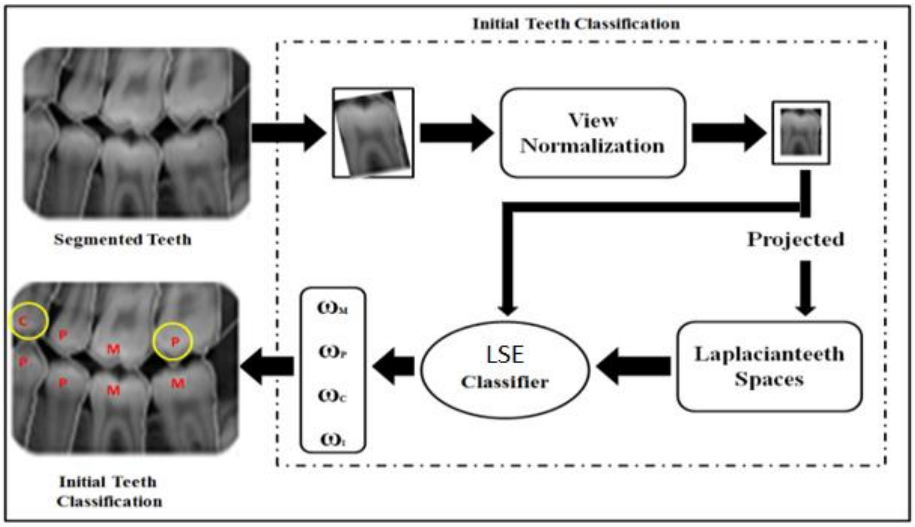


Figure 33: Block Diagram of Initial Teeth Classification.

In order to reduce the complexity of the classifier we intentionally flip teeth horizontally – if necessary – such that their crowns point upwards (see Figure 34 (a)). In choosing exemplar sets we seek to capture the intra-class variability; we achieve this by selecting exemplar teeth with variable sizes and orientations. Moreover, in the learning phase not only do we use the exemplars but also we use their conjugates produced by horizontal flipping (see Figure 34 (b)).

To construct these spaces we use sets of exemplar teeth images from each of the four classes. To setup the image subspaces from training set for the four teeth classes a data set of exemplars was prepared using the dental image database provided by the Missing and Unidentified Person Unit of Washington State Patrol [26]. 100 Molars, 100 Premolars, 20 Canines and 40 Incisors from records were segmented and preprocessed with total number of 260 teeth. In selecting these teeth, we tried our best to avoid unintentional bias towards the datasets. So we tried to select teeth evenly from upper and lower jaws, from the right and left sides of the mouth, and the teeth images of different intensity contrast.

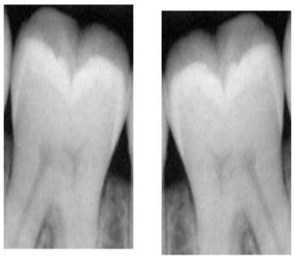


Figure 34: (a) Upward pointing crown (b) Conjugate of (a) under horizontal flipping.

Then the subject tooth (t_q) which has to be classified first undergoes view normalization to: (i) adjust its dimension to comply with that of the LaplacianTeeth, and (iii) compensate for possible poor contrast.

Figure 35, depicts a sample of exemplars from the teeth classes.

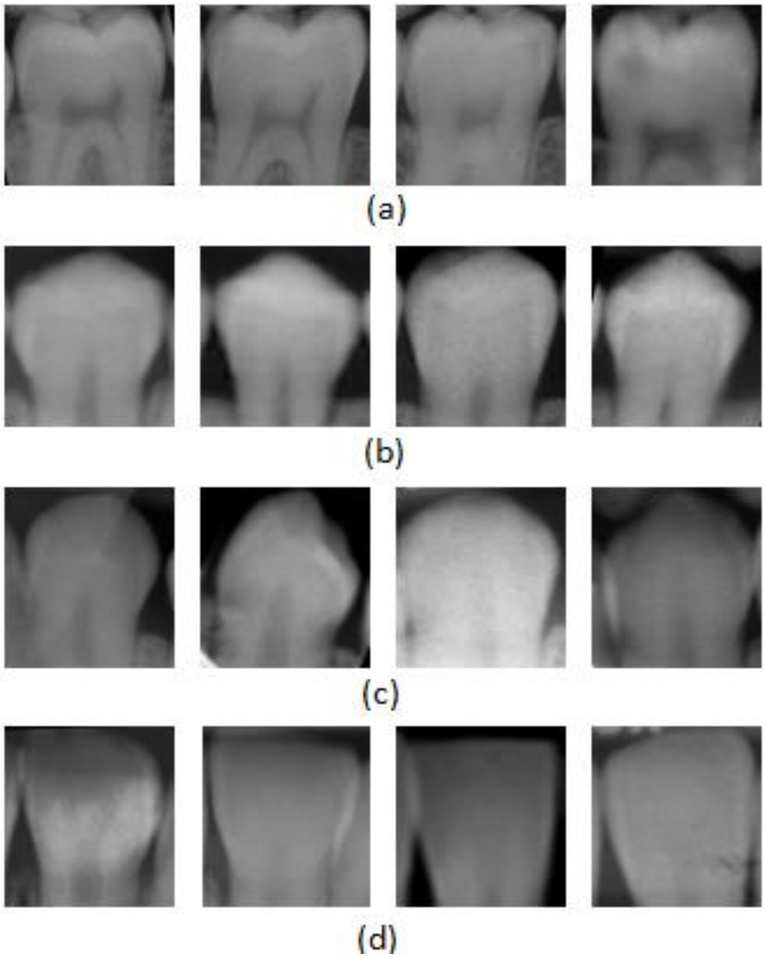


Figure 35: Sample of teeth used in constructing the image subspaces of the four teeth classes. (a) Molar. (b) Premolar. (c) Canine. (d) Incisor.

Figure 36, shows image representations of the first 11 eigenvectors of each class are namely the LaplacianMolar space, LaplacianPremolar space, LaplacianCanine space and the LaplacianIncisor space. The image on the upper left corner of each set represents the sample average.

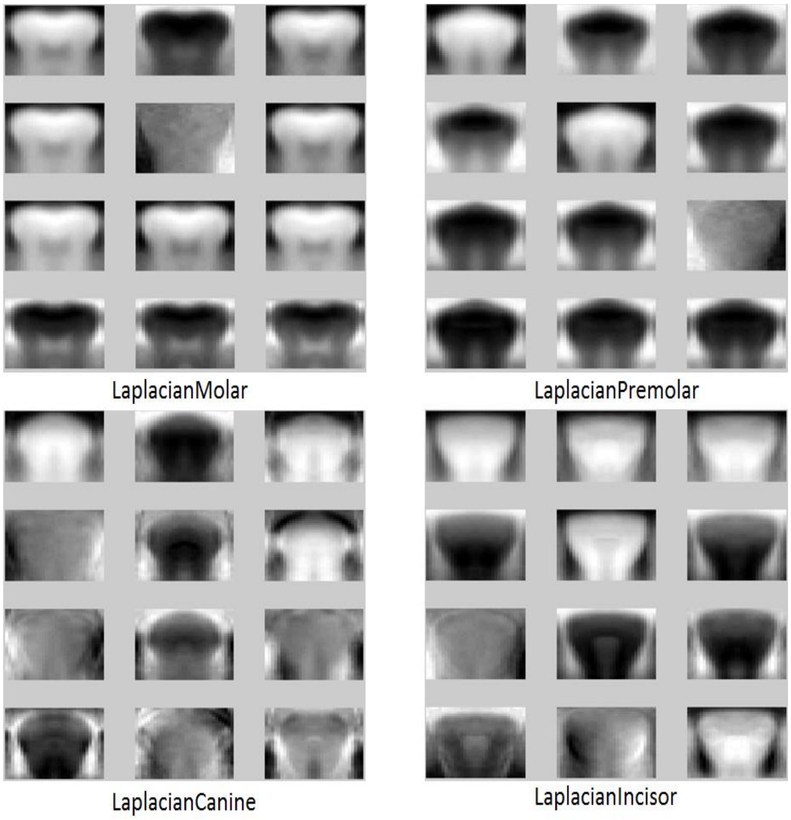


Figure 36: LaplacianTeeth representation for the four classes of teeth

Then the view-normalized tooth (t_{qN}) is projected on each of the four LaplacianTeeth spaces, hence, we obtain four coefficient sets w_M, w_P, w_C, w_I corresponding to the projections of subject tooth ($t_{qN)}$ onto the the molars subspace, the premolars subspace, the canines subspace, and the incisors subspace respectively. The obtained weight sets are used in conjunction with the sample mean of each of the four teeth classes $\mu_M, \mu_P, \mu_C, \mu_I$ to reconstruct the view-normalized tooth t_{qN} in the four image subspaces, thus obtaining the approximations ($\hat{T}_M, \hat{T}_P, \hat{T}_C, \hat{T}_I$). Thereafter we compute the sum of square error between the view-normalized tooth t_{qN} and its approximations, using this equation:

$$e_\omega = \sum(\check{T}_\omega - t_{qN})^2 \text{ , } \omega \in \{\text{‘}M\text{’}, \text{‘}P\text{’}, \text{‘}C\text{’}, \text{‘}I\text{’}\}$$

Where (x, y) are image coordinates. The subject tooth is classified based on the minimum euclidean distance rule given as follows:

$$t_{qN} \rightarrow \omega \mid \arg\min_{<\omega>} \{e_\omega\}, \omega \in \{'M','P','C','I'\}$$

3.2.4.3 Class Validation and Number Assignment

In section 3.2.4.2, we have discussed initial classification of teeth using OLPP algorithm. However, similar to other classification problems, the initial class assigned to the tooth based on least energy discrepancy rule is susceptible to errors. Hence validating the class of the teeth assigned through this initial classification is desired. Since the array of teeth in humans follows a particular pattern, teeth neighborhood rules could be exploited to validate and correct the initial classification of tooth. By doing this, we also intend to correct the initial class labels assigned to the tooth, when there is discrepancy between the initial tooth classification and reference pattern. If we find the detected or corrected classification sequence to be unique, each tooth would be assigned a number based on the position of it in its dental quadrant. For instance, consider as in the Figure 37, shown below, the lower sequence was correctly classified, so in the teeth labeling stage it was validated and labeled as (P4 – P5 – M6 – M7). The upper sequence wasn't correct, in this case, so it was corrected to (PPMM) and then labeled (P4 – P5 – M6 – M7).

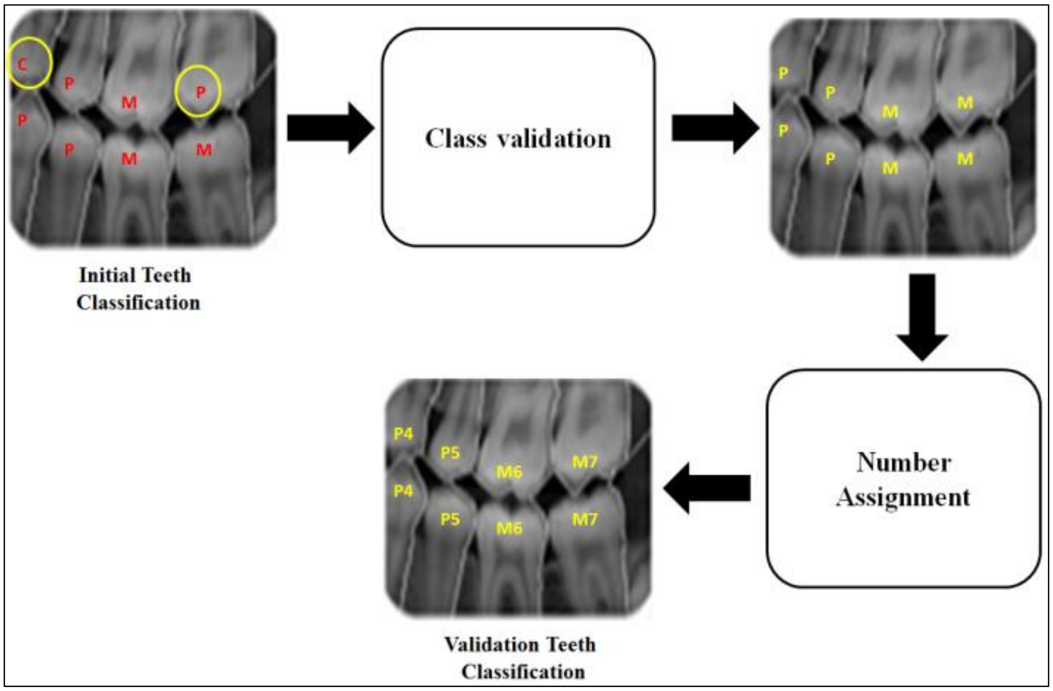


Figure 37: Block diagram showing the process of Validation and correction of Initial Teeth Classification

Our proposed approach for class validation and correction is based on the concept of string matching using edit distances. A number of string matching algorithms have been reported in the literature [61][62]. Generally, string matching can be thought of as the maximization/ minimization of a certain cost function such as the edit distance. Based on the approach outlined in [58] which actually is simplified form of the edit distance algorithm proposed in [63], string matching can be used to compare the reference pattern of teeth and initial classification of segmented dental film. The process is briefly discussed below. Consider that:

- **X** denotes the 16 character, reference pattern of the teeth '*MMMPPCIIIICPPMMM*'.
- $S_F = \{s_1, s_2, s_3, s_4, s_5, s_6, \ldots s_j \ldots s_n, 1 \leq n \leq 16 \mid s_j \in \{$'M,'P','C','I'$\}$ denote the sequence of the initially assigned.

Hence class validation problem can be posed as string matching problem with possible error. Here we intend to match S_F with input reference pattern X and by doing this we want to find the possible error and correct it. To accomplish this, we define the simplified edit distance as the cost function as in [58], based on the energy discrepancies between the tooth and its approximations: $e_\omega = \sum(\check{T}_\omega - t_{qN})^2$. Let $\{e_\omega, \omega = \mathrm{M}, \mathrm{P}, \mathrm{C}, \mathrm{I}\}$ denote the energy discrepancies between view normalized tooth image and the four approximations in the laplacian space. Whenever there is an error between the reference pattern X and the string S_F the cost $\psi^{[i]}_{\omega_0 \to \omega}$ of changing initially assigned class label ω_0 to ω where i denotes the number of the tooth and $\omega_0, \omega \in$ {'M','P','C','I'} be defined as [58]:

$$\psi^{[i]}_{\omega_0 \to \omega} = \frac{\mathrm{e}^{[i]}_{\omega} - \mathrm{e}^{[i]}_{\omega_0}}{\max\left(\mathrm{e}^{[i]}_{j}\right) - \mathrm{e}^{[i]}_{\omega_0}}$$

Hence the total cost of changing the string S_F to S'_F is defined as:

$$\psi_{S_F \to S_F^{''}} = \sum_{<i>} \psi^{[i]}_{\omega_o(i) \to \omega(i)}$$

If there is discrepancy between the string S_F (or S'_F) and the reference pattern X, string S_F is modified as S'_F by minimizing the cost function defined above. String edit distance technique can also be used when there is missing tooth in the sequence. In such cases, there are some assumptions or cues to determine whether there is a missing segment (tooth) such as existence of more than two segmented tooth in the sequence, the missing tooth should be in the middle of the sequence, and the gap of missing tooth must exceed a certain threshold value that is determined empirically from the training stage. In the case we have a missing tooth in the sequence, it is assigned do not care character “d” and still the same method for computing the cost as discussed above can be used where there is no change in the cost associated for changing “d” to any of the four teeth classes.

3.2.5 Experimental Results

To test the proposed approach for teeth classification and numbering, we prepared a disjoint test set with the data set of exemplars. One thousand and five hundred and ninety seven teeth were segmented and used for the teeth labeling experiment (776 molars, 771 premolars, 27 canines and 23 incisors). In preparing the test set, we used the CJIS ADIS project dental image database [27]. Table 6 gives the number of films/teeth we used in our test. In our experiment, we considered some of instances that have been addressed in [58], where there is a match between S_F or (S'_F) and X at more than unique shift. In the Bitewing films, we made use of the coexistence between the upper jaw and lower jaw segments to detect and possibly correct instances where the resulting sequences of the opposite quadrants are inconsistent with one another. This enhances the accuracy of overall labeling:

Table 6
NUMBER OF FILMS AND TEETH USED IN TESTING

	No. of films	**No. of teeth**
Bitewing films	256	1259
Periapical films	127	338
Total	383	1597

(Table 7) capture teeth classification confusion matrices for all films, (Table 8) bitewing films, (Table 9) periapical films.

Table 7
CONFUSION MATRIX FOR TEETH CLASSIFICATION IN ALL FILMS

INITIAL CLASSIFICATION PERFORMANCE 77.33%					
		ASSIGN CLASS			
		I	C	P	M
TRUE CLASS	I	12	3	8	0
	C	4	2	17	4
	P	24	8	726	13
	M	4	2	275	495
AFTER VALIDATION PERFORMANCE 87.66%					
		ASSIGN CLASS			
		I	C	P	M
TRUE CLASS	I	15	2	6	0
	C	4	6	14	3
	P	16	7	729	19
	M	4	1	122	649

Table 8
CONFUSION MATRIX FOR TEETH CLASSIFICATION IN BITEWING FILMS ONLY.

INITIAL CLASSIFICATION PERFORMANCE 81.10%					
		ASSIGN CLASS			
		I	C	P	M
TRUE CLASS	I	0	0	0	0
	C	0	0	9	0
	P	0	2	625	8
	M	0	0	219	396
AFTER VALIDATION PERFORMANCE 91.32%					
		ASSIGN CLASS			
		I	C	P	M
TRUE CLASS	I	0	0	0	0
	C	0	3	6	0
	P	0	2	617	16
	M	0	0	85	530

Table 9
CONFUSION MATRIX FOR TEETH CLASSIFICATION IN PERIAPICAL (Lower & Upper) FILMS ONLY.

INITIAL CLASSIFICATION PERFORMANCE 63.31%					
		ASSIGN CLASS			
		I	C	P	M
TRUE CLASS	I	12	3	8	0
	C	4	2	8	4
	P	24	6	101	5
	M	4	2	56	99
AFTER VALIDATION PERFORMANCE 74.07%					
		ASSIGN CLASS			
		I	C	P	M
TRUE CLASS	I	15	2	6	0
	C	4	3	8	3
	P	16	5	112	3
	M	4	1	37	119

We observe the following from the results obtained through our experiments:

- Overall, the initial classification is 77% accurate and teeth class validation enhances the overall teeth classification accuracy to 87%.
- In bitewing films, initial classification is 81% accurate and teeth class validation enhances the overall teeth classification accuracy to 91%.
- In periapical films, initial classification is 63% accurate and teeth class validation enhances the overall teeth classification accuracy to 74%.

Nasser et al [58] presented a dual-stage technique for automatic construction of dental charts using Principal Component Analysis (EigenTeeth) and also they used string matching, they used the same database as one we used in the our testing. They got an initial classification accurate for all dental films of 75% and their result was enhanced to 86% after applied the class validation. For bitewing dental film view they got an initial classification accurate of 79% teeth class validation raised the overall teeth classification accuracy to 90%. For periapical dental film view they got an initial classification accurate of 61.5%, teeth class validation raised the overall teeth classification accuracy to 74%.

Figure 38(a,b,c) show examples of successful teeth numbering of bitewing and periapical films, respectively. Figure 38(d) shows an example of incorrectly numbered periapical films.

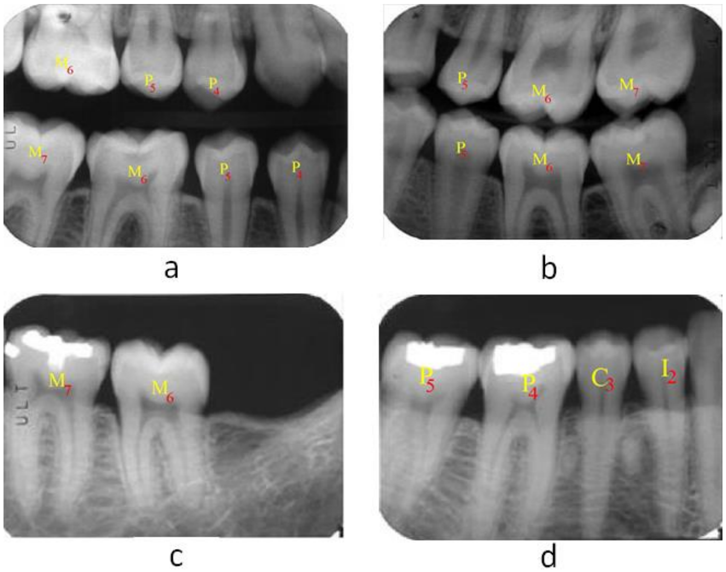


Figure 38: (a) & (b) Examples of bitewing films with the correct number of teeth. (c) Example of periapical film with the correct number of teeth. (d) Example of bitewing films with the incorrect number of teeth.

Receiver Operating Characteristic (ROC) curve is a useful technique to organize classifiers and visualize their performance [64]. We conducted experiments and plotted the ROC curve to compare the two classifiers performance (Eigen teeth [58] and LaplacianTeeth) on the same data set as depicted in the table 10.

Table 10
DATA SETS USED TO GENERATE ROC CURVES

	Molar	**Premolar**	**Subtotal**
Training set	160	160	320
Data set	40	40	80
Total	200	200	400

Figure 39 shows the ROC curves of two different features, one based on EigenTeeth and the other based on LaplacianTeeth which was employed in this section. From the plotted ROC curve, it can be observed that the LaplacianTeeth has lower error equal rate (EER), and therefore has better performance over the classifier based on EigenTeeth.

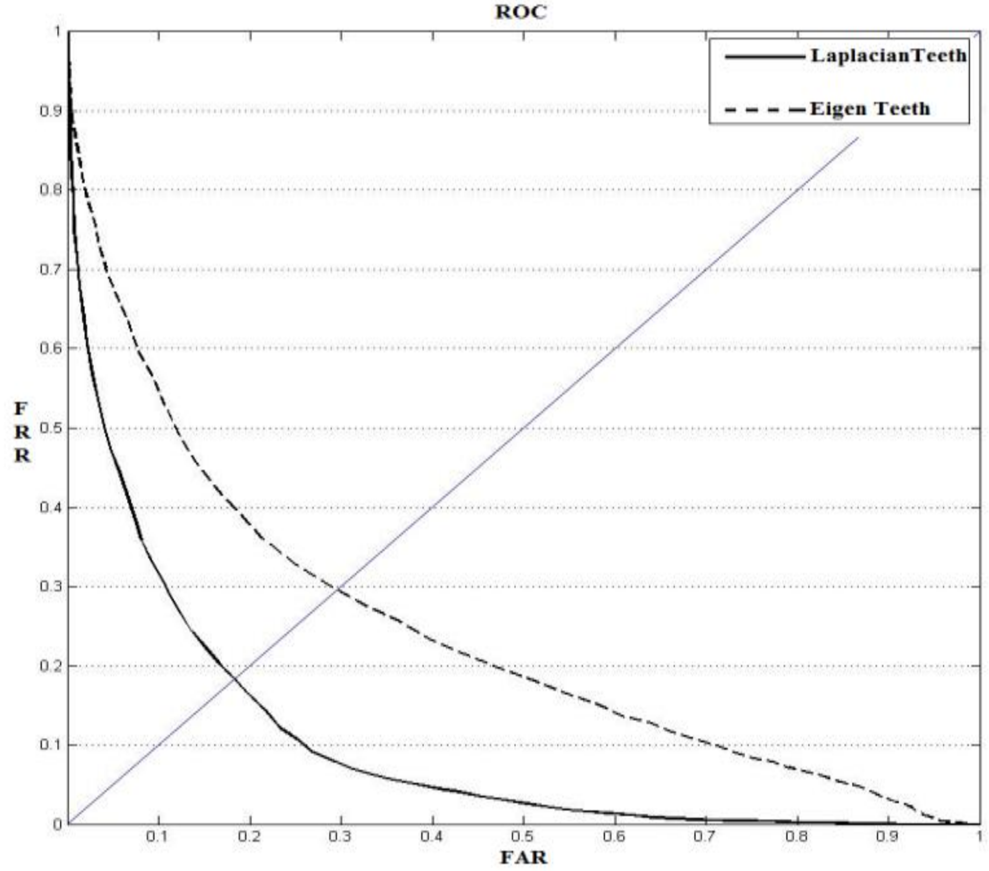


Figure 39: ROC curve of EigenTeeth and LaplacianTeeth based classifier.

Table 11 shows the computational time required for the tooth classification between the proposed algorithm, and the one based on EigenTeeth [58], where both algorithms were implemented on MATLAB® implementation running on a 2.66 GHz 4 GB RAM Intel® Core(TM) Quad CPU PC platform. The proposed approach has higher classification time than the other method, while yielding better performance of teeth classification as mentioned above.

Table 11
Classification performance comparison.

Method	**Avergae Labeling Time / Bitewing Dental Film**
Eigen Teeth	11.50 sec
Laplacian Teeth	15.30 sec

4 Potential Match Search

Recent disasters have reiterated the significance of automated dental identification systems. For example in the 9/11 attacks, many victims were identifiable only from pieces of their jaw bones. This is the main motivation for building the Automated Dental Identification System (ADIS), and for the system to have a very powerful searching engine. However, archiving and retrieving dental records from large databases is a challenging task and has received inadequate attention in literature. This chapter presents a new technique that searches the dental database in a fast way to find a candidate list (references dental records) that has high similarity with the subject dental record, and its size is very short compared to the original database size. Therefore, it will facilitate comparison image component (comparison on low level teeth features) in the automated system in order to retrieve the candidate list in a rapid manner and reduce the database to be searched by a more time expensive image comparison.

4.1 Introduction

The identification of the deceased individuals in huge disasters is difficult. In such cases, most physiological biometrics such as iris, fingerprint, and face may not be viable options for identification, because of the decay of the body soft tissues. The dental feature is one of the best candidates for the postmortem biometric identification due to the following reasons [3]: resist early decay of body tissues, withstand severe conditions in mass disasters, and unique (Identification can sometimes be made from one tooth).

The main objective of this chapter present a new technique that reduces the time it takes to retrieve the candidates list while maintaining the accuracy and this is done by comparing a subject dental image record (postmortem PM) with a set of reference dental image records (antmortem AM) that stored into database. This candidate list must have a very high probability of containing the target record, and its size is very short compared to the original database size. The proposed approach consists of two main stages, in the first stage preprocessing of the dental records (segmentation and teeth labeling) is performed in order to get a reliable appearance-based, low computational-cost features. In the second stage, we used a technique based on LaplacianTeeth using Orthogonal Locality Preserving projection (OLPP) algorithm to produce candidate list. The aim of this approach is to present a new technique to search and retrieve the candidate list of the reference dental images from within the database using the appearance feature of the teeth.

The chapter is organized as follows: In Section 4.2, brief introduction of preprocessing operation of the dental images is given. The proposed technique that is based on Orthogonal Locality Preserving projection (OLPP) is described in Section 4.3. In Section 4.4 the experimental results are presented.

4.2 Teeth preprocessing

To design a fast search engine that retrieves dental record images after comparing reference dental record images with subject dental record to be searched, it takes more than one factor in determining the selection teeth feature, including the computational complexity and reliability of these features in the matching process. Methods based on only teeth contour take a relatively long time to extract the tooth contour as well as less accurate [65]. These reasons have led us to find an alternative method that depends on reliable appearance-based, low computational-cost features. To achieve that goal, we used the following preprocessing steps.

4.2.1 Teeth Segmentation

In this approach we used teeth segmentation technique introduced in section 3.1, which starts with an iterative thresholding followed by an adaptive thresholding to binarize the teeth images. Then, we adapt the seam carving technique on the binary images, using both horizontal and vertical seams, to separate each individual tooth.

4.2.2 View Normalization

The view normalization and resizing step comes after the teeth segmentation stage, whose outcomes do not have standard view in terms of scale and rotation. The view normalization is an essential step to improve and overcome variations in orientation and scale. An input tooth is geometrically normalized. To perform geometric normalization of the segmented tooth, the most important step is to ensure that its lateral surface appears predominantly vertical [58].For further details of this method, refer to section 3.2.2.

4.2.3 Teeth labeling

We tackled this problem using a two stage approach. The first stage utilizes low computational cost, appearance-based features for assigning an initial class based on Orthogonal Locality

Preserving projection Algorithm (OLPP) introduced in section 3.2. The second stage applies a string matching technique [58], based on teeth neighborhood rules, to validate the initial class and hence assign a number corresponding to the tooth location in the dental chart .

4.3 Proposed Approach

The proposed approach for retrieving dental radiographs for post-mortem identification is appearance based using Orthogonal Locality Preserving Projection algorithm (OLPP). It differs from Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA), which preserves the euclidean structure of tooth space [60]. PCA and LDA are the two prominent methods widely used for face recognition. Recently, a number of research efforts have shown that face images possibly lie on a non-linear submanifold [66]. However PCA and LDA see only the global euclidean structure. They fail to discover the underlying structure if the face images lie on a non-linear submanifold. So the manifold structure needs to be modeled by preserving the local structure. Locality Preserving Projection (LPP) is a manifold learning technique which preserves the local structure. However, LPP is nonorthogonal. The Orthogonal Locality Preserving Projections (OLPP) on the other hand produces orthogonal basis functions and can have more locality preserving power than LPP.

Section 4.3.1, presents a brief idea about the OLPP algorithm. We used a technique based on LaplacianTeeth using OLPP algorithm in order to retrieve dental radiographs for post-mortem identification and produce candidate list, described in section 4.3.2.

4.3.1 OLPP Algorithm [67]

To overcome the problem of singularity that is present if the number of the teeth training images set is much smaller than the dimension of the tooth image vector, we apply PCA to project the teeth into the sub-space without losing any information, so that the number of images in the training set will exceed the PCA coefficients of the image and it becomes nonsingular. In addition the PCA projected image is more robust to noise than the preprocessed one. In brief, the algorithm of OLPP to get LaplacianTeeth is started as follows: (i) PCA Projection: the training set of teeth images t_i is projected into the PCA subspace by throwing away the components corresponding to zero eigenvalue. The transformation matrix of PCA is denoted by W_{PCA}. The features are extracted with PCA projection are statistically uncorrelated. (ii) Constructing the adjacency graph: Let X=[t_1,t_2,...,t_N] be a set of training set of teeth images. Let G denote a graph with N nodes where the i^{th} node corresponds to the tooth image t_i. If nodes t_i and t_j are connected then the (i, j) and (j, i) elements of the nearest-neighbor matrix get the values $S_{ji} = S_{ij} = e^{-(\|t_i - t_j\|)2} / t$ where t is a suitable constant. Otherwise they are zero. This is called the Heat kernel approach. The weight matrix S of graph G models the local structure of the teeth manifold.

(iii) Computing the Orthogonal Basis Functions: We define D as a diagonal matrix whose entries are column (or row, since S is symmetric) sums of S, $D_{ii} = \sum_j S_{ji}$ we also define L = D - S, which is called Laplacian matrix in spectral graph theory. Let {a_1, a_2, . . . , a_{k-1}} be the orthogonal basis vectors, we define:

A_{k-1} = [a_1, a_2, . . . , a_{k-1}] , B_{k-1} = $[A_{k-1}]^T$ $(XDX^T)^{-1}$ A_{k-1}. The orthogonal basis vectors {a_1, a_2, .. , a_{k-1}} can be computed as follows: compute a_1 as the eigenvector of $(XDX^T)^{-1}$ XLX^T associated with the smallest eigenvalue. Compute a_k as the Eigenvector of:

$$M^{(K)} = \{I - (XDT^T)^{-1}A^{(K-1)}.\left[B^{(K-1)}\right]^{-1}[A^{(K-1)}]^T\}.(XDT^T)^{-1}\,XLX^T$$

associated with the smallest eigenvalue of $M^{(K)}$.

(iv) OLPP Embedding: Let W_{OLPP} = [a_1, a_2, . . . a_L], the embedding is as follows: $x \rightarrow y = W^T x$ where $W = W_{PCA} W_{OLPP}$. Where y is a l-dimensional representation of the teeth image x, and W is the transformation matrix.

(v) Matching Process: The Matching process computes similarity score between subject and the reference tooth images from a specific class (‘M’,’P’,’C’,’I’) based on their projection in the laplacian space.

4.3.2 Retrieving Dental Radiographs for Post-mortem

4.3.2.1 LaplacianTeeth Construction

To setup the image subspaces from training set for each of the four teeth classes, a data set of exemplars was prepared using the dental image database provided by the Missing and Unidentified Person Unit of Washington State Patrol [26]. 100 Molars, 100 Premolars, 20 Canines and 40 Incisors from records were segmented and preprocessed with total number of

260 teeth. In selecting these teeth, we tried our best to avoid unintentional bias towards the datasets. So we tried to select teeth evenly from upper and lower jaws, from the right and left sides of the mouth, and the teeth images of different intensity contrast. Figure 40, depicts a sample of exemplars from the teeth classes.

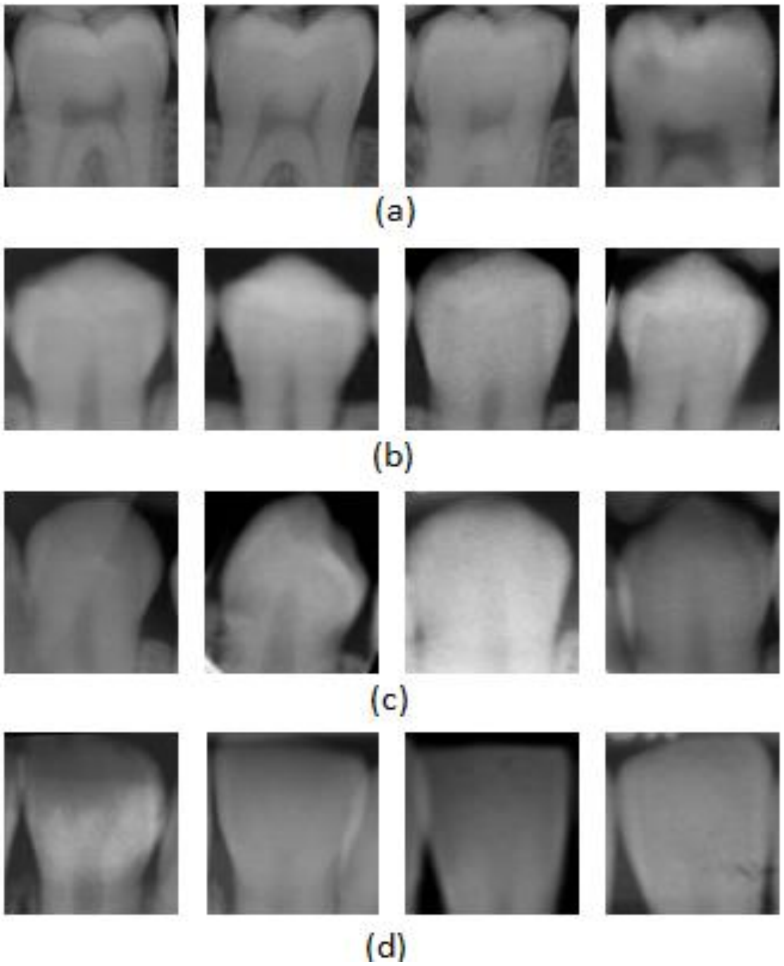


Figure 40: Sample of teeth used in constructing the image subspaces of the four teeth classes. (a) Molar. (b) Premolar. (c) Canine. (d) Incisor.

As described in sub-section 4.3.2, we used the Orthogonal Locality Preserving Projections (OLPP) algorithm to establishing a LaplacianTeeth space for each of the four teeth classes from the data training set of teeth images. Figure 41 shows image representations of the first 11 eigenvectors of each class are namely the LaplacianMolar space, LaplacianPremolar space, LaplacianCanine space and the LaplacianIncisor space. The image on the upper left corner of each set represents the sample average.

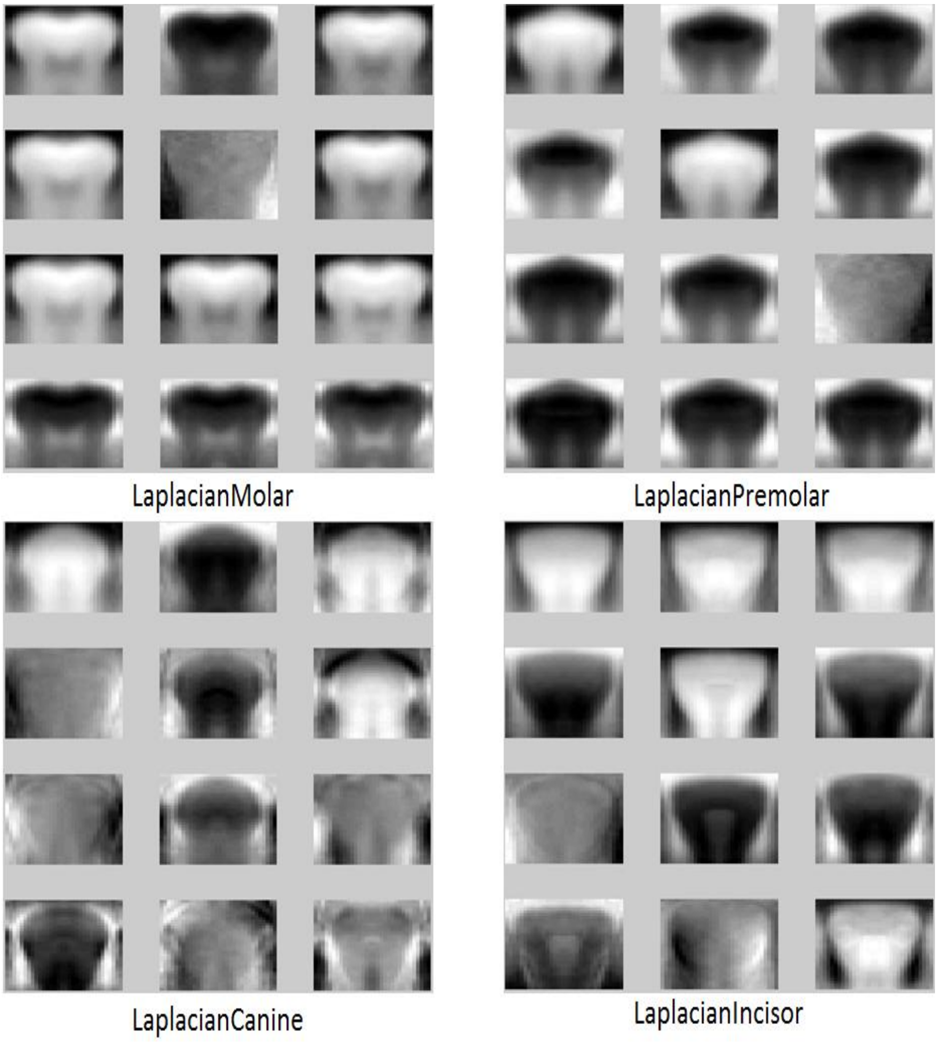


Figure 41: The LaplacianTeeth representation for the four classes of teeth.

4.3.2.2 Using LaplacianTeeth for teeth matching

The proposed approach for potential match search is illustrated in Figure 42. The subject tooth (t_S) and the reference tooth (t_R) which have to be matched, first undergoes view normalization to: (i) adjust its dimension to comply with that of the LaplacianTeeth, and (ii) compensate for possible poor contrast.

Second, the normalized subject and reference tooth input images are projected onto their corresponding LaplacianTeeth Space in order to get two vectors, one for the subject tooth and another for reference tooth. The two vectors (subject and reference) are used to measure Least Square Error (LSE) which gives the score of matching between the subject and reference tooth.

In some cases, a tooth may occur in multiple films resulting in multiple distance values corresponding to different representations of the same tooth. In such cases, we choose the mean of all the distances as the match score. To move from tooth-to-tooth comparison toward record-to-record comparison, we combine the match scores generated by all component teeth via the mean rule.

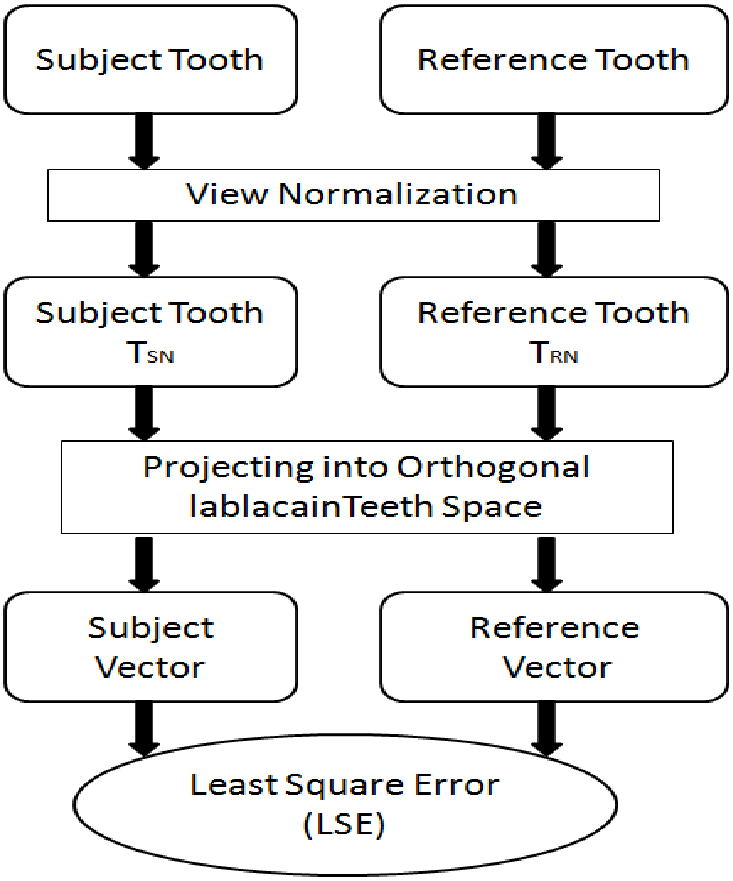


Figure 42: Block diagram of proposed method

4.4 Experimental results

We conducted an experiment to test proposed approach in searching and retrieving a candidates list (references dental records) that has high similarity with the subject dental record. We use dental records of the two types of films: (bitewing and periapical dental views) of the FBI's Criminal Justice Service (CJIC) ADIS database [27], which includes dental radiographs of ante-mortem (AM) and postmortem (PM). Table 12 shows the number of records (each record may have one or more of dental films) used in the experiment, which includes 57 records for dental X-rays of deceased persons (postmortem dental records) and 47 records for dental X-rays to living persons (ant-mortem dental records), with 20 matched records, with a total over 2000 teeth.

Table 12
NUMBER OF DENTAL RECORDS and DENTAL OF FILMS USED IN EXPERIEMNT.

	No. of Dental Records	**No. of Dental Films**
Subject (PM)	75	312
Reference (AM)	75	191

Figure 43 shows the results of the experiment, the proposed technique correctly retrieve the dental record 65% in the 5 top ranks while the other method based on the EigenTeeth remains at 60% [46]. It also shows the difference in accuracy even with the increase in size of the first position of the ranked list; our approach remains ahead in precision to the method based on EigenTeeth.

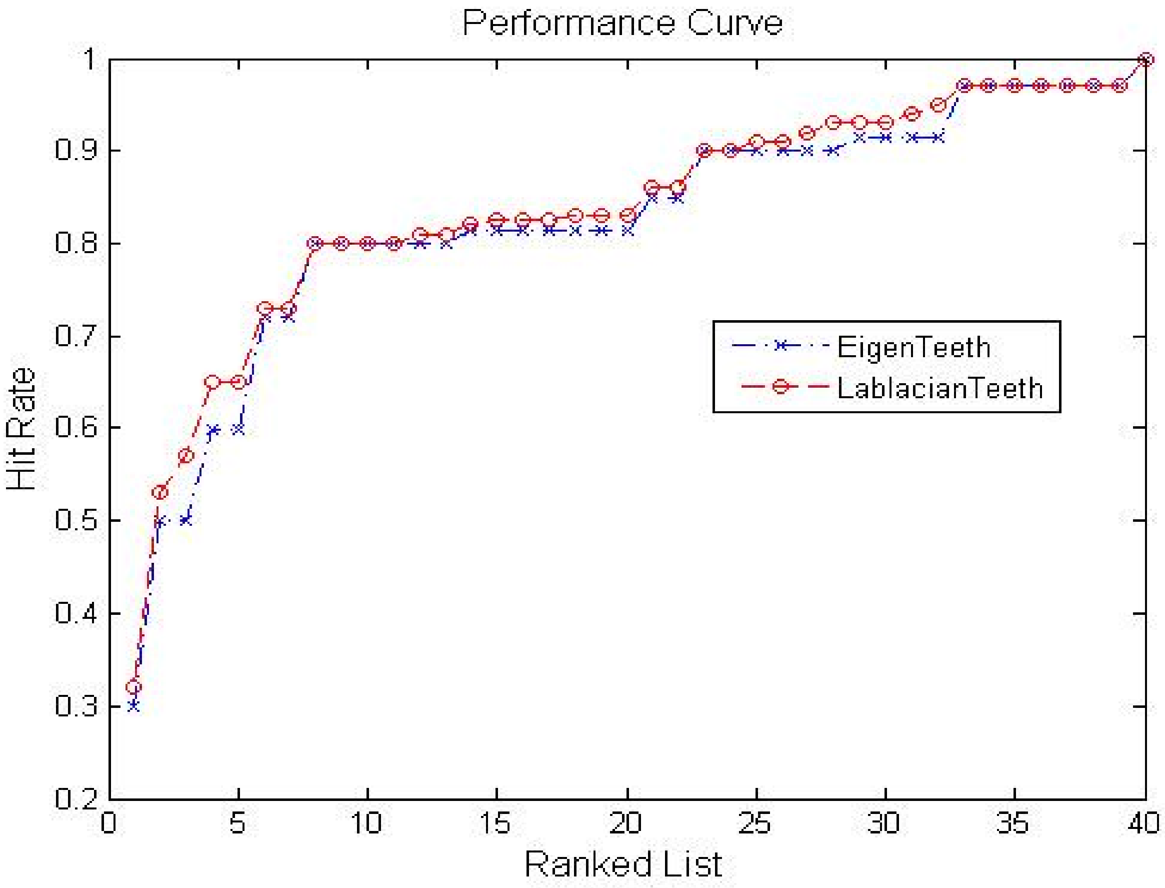


Figure 43: Matching performance comparison between the proposed method and the EigenTooth based approach.

To select the optimal normalized teeth image size, we found that moving from image size 16x16 to 32x32 enhances the matching performance. Moving to 64x64 and 128x128 didn’t enhance the matching performance any more, while it takes more processing time. The tooth image size of 32x32 yields the best performance compared with 64x64 and 128x128 image sizes.

Table 13 shows the computational time required for the comparison between the proposed algorithm, and the one based on EigenTeeth [46], where both algorithms were implemented in MATLAB platform under the same settings. The average time in proposed approach to match record to record comparison takes less than 0.17 seconds and the average time to retrieve the best matched AM reference dental record for given PM query dental record is about 0.17n seconds, where n is the number of records in the database. On the other hand, method based on EigenTeeth takes less than 0.09n, which is less when compared to the proposed approach.

However our approach is capable of achieving better performance than the method based on EigenTeeth in terms of hit rate.

Table 13

COMPUTATIONAL TIME COMPARISON BETWEEN THE PROPOSED METHOD (LAPLACIANTEETH) AND THE PCA BASED APPROACH (EIGENTEETH)

	EigenTeeth	LaplacianTeeth
Average Record to Record	0.09 sec	0.17 sec
Average record retrieval	4.23 sec	7.99 sec

5 Image Comparison

Automated dental identification is one of the best candidates for postmortem identification. With the large number of victims encountered in mass disasters (e.g., September 11, and Asian tsunami), automating the dental identification process would enhance the scalability of this biometric. However, archiving and retrieving dental records from large databases is a very challenging task and has received inadequate attention in the literature. In this chapter presents a new approach based on Scale Invariant Feature Transform (SIFT) descriptors of tooth contour to compare between subject and the corresponding teeth of each candidate, thus producing a short match list. The basic idea of this approach is to compare SIFT descriptors of the subject tooth and the reference tooth so as to calculate tooth-to-tooth matching score. Then the various tooth-to-tooth matching scores are integrated into record-to-record matching score.

5.1 Introduction

Dental biometrics automatically analyzes dental radiographs to identify the deceased individuals. The radiographs acquired after a person is deceased are called the Post-mortem (PM) radiographs, and the radiographs acquired while the person is alive are called the Ante-mortem (AM) radiographs. The AM radiographs, collected from the dentists, are labeled with the patient names. The approach for human identification is to match the PM radiographs of the unidentified person against the candidate list of AM radiographs. If the set of teeth in the PM images matches the teeth in an AM radiograph, then the identity of that person is obtained.

The approach that depends on the contents of the tooth image that is sometimes incomplete and imperfect; thus reflecting a major impact in the results. This does not happen with the proposed approach that is based on SIFT descriptors, which in many cases does not require a full image of the tooth. In other words, any part of the tooth image is sufficient to extract the tooth contour that guides the SIFT descriptors [68]. The proposed approach for searching the candidate list based on the matching of the SIFT descriptors guided by the teeth contour between the subject and reference dental records.

The problem of teeth matching was addressed in [34], where Jain and Chen propose a semi-automated system for human identification based on matching of teeth contours extracted from dental x-ray images. Their system follows three main steps for identification: radiograph segmentation, teeth contour extraction, and shape matching. For each radiograph, a human user initializes segmentation by specifying a pixel that belongs to the gap valley (an artificial curve that best separates the maxilla and mandible), then detection of the entire gap valley as well as teeth isolation are carried out using integral projection. In [34], extraction of an isolated tooth contour is achieved in two steps, crown extraction and root extraction. Crown extraction is treated in a pixel classification approach along radial rays, while root extraction is considered an iterative neighborhood context analysis problem. Identification is completed by searching a database of contours for records with relatively small euclidean matching distance to subject contours using a quasi-affine transformation model for contour alignment. In [42], Chen and Jain, propose a directional SNAKE approach for contour extraction of isolated teeth to mitigate excessive contour extraction errors of their approach in [34]. In [43], Chen and Jain present an automatic method for matching dental radiographs. The matching is performed in three steps. First, a shape registration method is proposed to align and compute the distance between two teeth on the basis of tooth contours. If the shapes of the dental work are available, they can assist in the matching. They proposed an area-based metric for matching the dental work. The two matching distances are then combined to obtain the tooth correspondence and to measure the similarity between images. In [44], Nomair and Abdel-Mottaleb present combining two matching algorithms of a system for identifying individuals based on their dental X-ray records. The system archives the AM images into the database by segmenting the teeth, extracting their contours, numbering the teeth, and representing each tooth by a set of features. The first matching technique relies on selecting a set of salient feature points with high curvature from the contour of a tooth and generates a signature vector for each salient point. The second matching technique uses Hierarchical Chamfer distance transformation to reduce the search space as well as the computational load significantly. Given a PM image, the system segments the teeth, extracts their contours and represents each tooth as a set of feature vectors. The matching modules search the database for the best matching candidates. Matching scores are calculated based on the distances between the feature vectors of AM and PM teeth. In [32], Zhou and Abdel-Mottaleb present a content-based archiving and retrieval system of dental images for use in human identification. It contains three major stages: dental image classification, bitewing image segmentation and retrieval based on teeth shapes using bidirectional Hausdorff distance.

In the classification stage, two features are proposed for dental X-ray image classification. The classified bitewing images are segmented to extract the contours of molars and premolars, which are then used to archive the images in an AM database. During retrieval, a PM bitewing image is segmented to extract teeth contours, which are used to find the most similar images in the AM database using Hausdorff distance measure.

The remainder of the chapter is organized as follows: Section 5.2 introduces the preprocessing techniques, followed by the proposed retrieving technique, section 5.3 presents the experiment results.

5.2 Material and Methods

This section illustrates the required preprocessing steps followed by a description of the proposed SIFT-based dental retrieving technique. Figure 44, shows the logical block diagram of the proposed technique. The view normalization will be explained in section 5.2.1; the contour extraction in section 5.2.2 and matching of subject and reference tooth using SIFT descriptors in section 5.2.3.

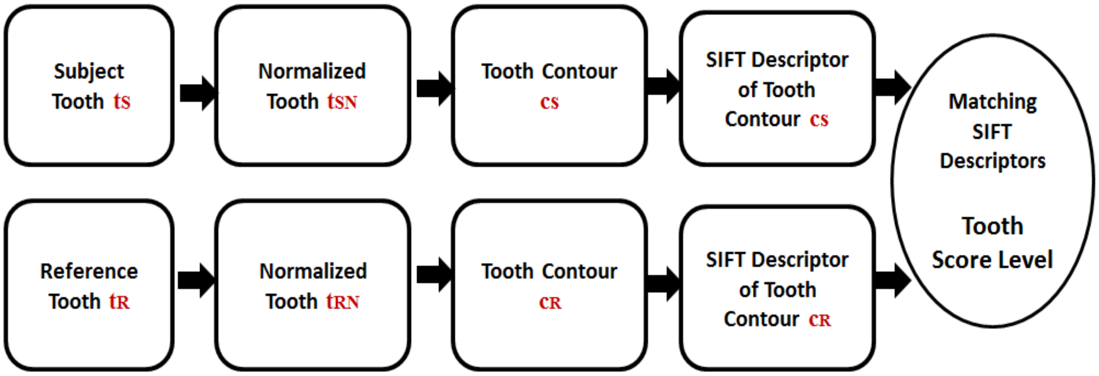


Figure 44: The logical diagram of the Dental matching component.

5.2.1 View Normalization

The view normalization is an essential step to improve and overcome variations in orientation and scale. To normalize the view of the dental segmentation outcome is to ensure that the surface of tooth crown appears horizontally as much as possible. This can be done by finding the appropriate rotation angle that is corresponding to strongest mean of the vertical orientations. See chapter 3 of teeth labeling, section 3.2.2.

5.2.2 Contour Extraction

Tooth contour extraction is to isolate each tooth in order to facilitate the extraction of features like crown contour (crown is part of the tooth above the gum line) and root contour (root is part of the tooth below the gum line "inside the bone") for identification use. The contour is one of the important features that can discriminate between teeth. The task of teeth contour extraction was accomplished using active contour without edges [37]. Active contour without edges [69] is based on the intensity of the overall region of the tooth image and, therefore does not necessitate the presence of a sharp boundary between the teeth as in the case of snakes [70]. Also, the starting contour can be in any part of the image and does not have to be around the boundary of the tooth. Further this technique can extract the region contour in the presence of additive noise and in the absence of well – defined image gradient. Figure 45, shows some result of contour extraction using the technique introduced in [37]. As a result of poor quality of some images, the resulting contours may also have poor quality such as Figure 45(g,h) or partial poor quality as in (c,d). To overcome this drawback, SIFT descriptors were used to represent the tooth. The tooth contour is used to guide the selection of SIFT points, which was experimentally proven to reduce the processing time and enhance the performance accuracy.

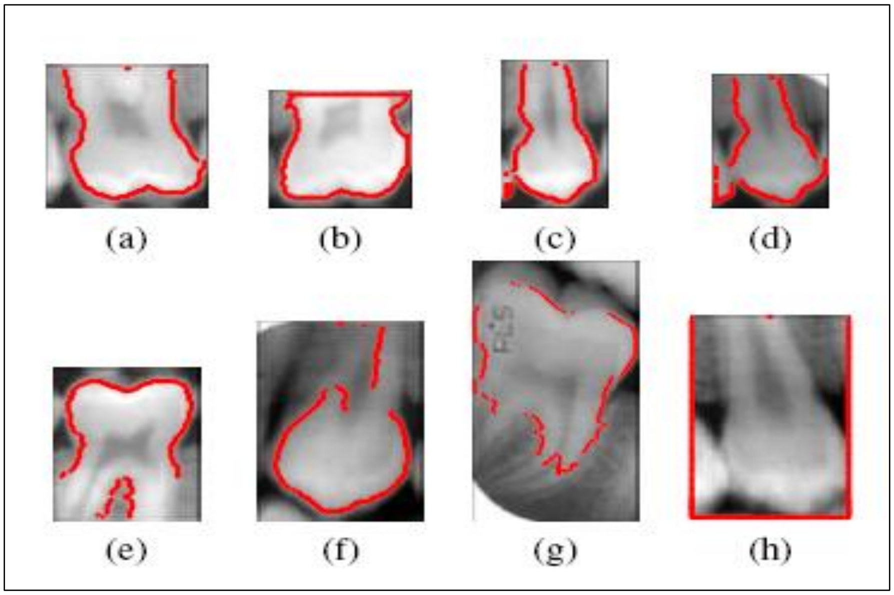


Figure 45: Contour extracted using active contour without edges. (a, b) Perfectly segmented (P), (c, d) Perfect crown (PC), (e, f) Correctable Error (C),(g,h) Error in contour (E).

5.2.3 SIFT Descriptors and Matching

The scale invariant feature transform (SIFT) [71] algorithm is a method for extracting highly distinctive invariant features from images, that can be used to perform efficient matching between different views of an object. SIFT algorithm [68] consists of four major stages: scale-space extrema detection, key point localization, orientation assignment and key point descriptor. The first stage uses difference-of-Gaussian function to identify potential interest points, which are invariant to scale and orientation. In the key point localization step, the low contrast points are rejected and the edges response is eliminated. Hessian matrix is then used to compute the principal curvatures and to eliminate the key points based on the ratio between their principal curvatures. An orientation histogram is formed from the gradient orientations of sample points within a region around the key point in order to get an orientation assignment. According to [68], the best results were achieved with a 4 x 4 array of histograms with 8 orientation bins in each; hence the descriptor of SIFT that we used is 4 x 4 x 8 = 128 dimensions.

SIFT feature points have already demonstrated their efficiency in many other applications including biometrics [72]. Hence, we adopted the SIFT for tooth representation. The Lowe’s implementation of SIFT (Lowe’s parameters) has been employed. The tooth contour is considered as one of the important features that can discriminate between teeth. In our work, we only consider SIFT descriptors that lay across the tooth contour points as shown in Figure 46. The figure shown is the result of computer simulation for clear demonstration purpose.

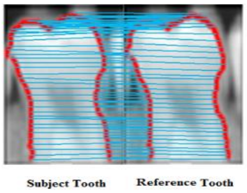


Figure 46: SIFT descriptors of subject and reference tooth contour guided points.

Then matching these SIFT descriptors between subject and reference teeth up in order to get the tooth level score as follows:

$$\boldsymbol{SC_{tooth(i)}} = \text{Match\_descriptor}\left\{\boldsymbol{T_{sub(x,y)}}, \boldsymbol{T_{ref(x,y)}}\right\}$$

Where $T_{sub(r,c)}$ denotes the SIFT descriptor points of the subject tooth and $T_{ref(r,c)}$ denotes those of the reference tooth.

5.2.4 Teeth score Fusion

To move from level tooth-to-tooth level to record-to-record matching level, two problems are encountered (as shown in Figure 47):

- The same tooth may have multiple representations in either the reference record or the subject record or even both. At the tooth-level fusion, we want to calculate the matching score between the different views of the same tooth t_{si} of a subject record (t_{S1}, .., t_{S32}) and the available views of reference tooth t_{Ri} of a reference record (t_{R1}, .., t_{R32}), such that we produce a single distance representing the matching score between the subject tooth t_{Si} and reference tooth t_{Ri}, namely $d_i(t_{Si}, t_{Ri})$. We use mean fusion rule for multiple teeth representation fusion.

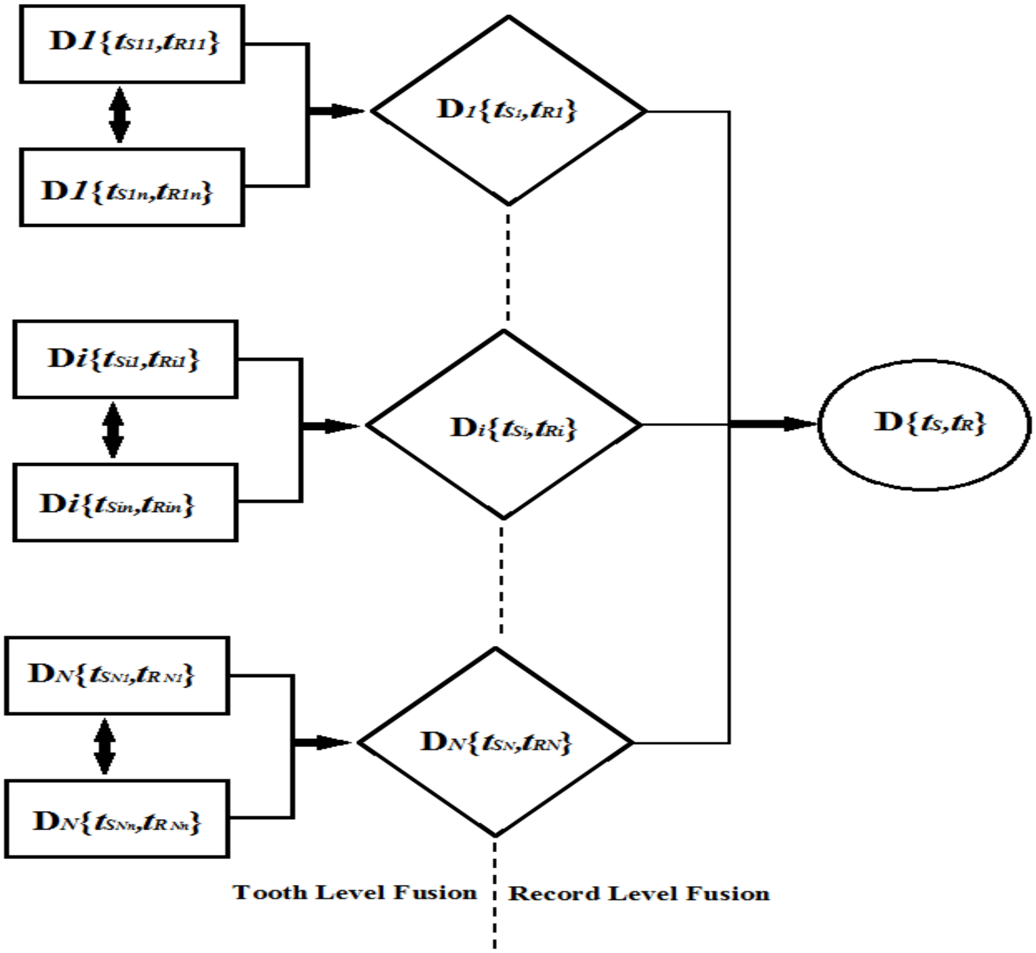


Figure 47: A hierarchical fusion scheme

- Integrating the various teeth matching scores to have one matching score between the subject and reference records. With N teeth in common between the dental charts of a subject record and a reference record, we obtain N ($N \leq 32$) tooth-to-tooth matching scores. At the record-level, we are looking for an integration scheme for combining these N $\{\mathrm{d}_i(t_{Si}, t_{Ri}), i = 1, .., N\}$ into a single record-to-record matching score $D(S, R)$. We study two alternative rules for fusing the match scores, namely the mean rule. We found the mean rule to yield better performance.

This result D(S, R) represents the matching score between a subject record and a reference record which will send it to candidate match list.

5.3 Experimental results

To test the proposed approach for dental image retrieval, a test set was prepared using the FBI's Criminal Justice Service (CJIC) ADIS database [27],which includes dental radiographs of ante-mortem (AM) and postmortem (PM). The matching method was evaluated using 57 PM query dental records. The AM database contains 47 dental records, as described in Table 14. Each segmented tooth in PM query image is classified and labeled. Then its SIFT descriptors are computed and matched with the AM tooth SIFT descriptors that has the same labeling; the best matched AM tooth is the tooth with mean matching number.

Table 14
NUMBER OF DENTAL RECORDS, DENTAL OF FILMS, AND TEETH USED IN EXPERIEMNT.

	No. of Dental Records	**No. of Dental Films**
Subject (PM)	75	312
Reference (AM)	75	191

Figure 48 shows the result of the experiment, the proposed technique correctly retrieve the dental record 35% and 75% in the 1 and 5 top ranks respectively. As stated in section 5.2.2 (View Normalization) the output of teeth segmentation techniques do not have standard view in terms of scale and rotation. We use the view normalization on the teeth outcomes from segmentation component to ensure that the surface of tooth crown appears horizontally as much as possible which involves the important features of the tooth. Figure 48, also depicts the matching performance does not improve, when we repeat the experiment of matching without view normalization, and in this case the proposed technique correctly retrieved the dental record 26% and 60% in the 1 and 5 top ranks respectively.

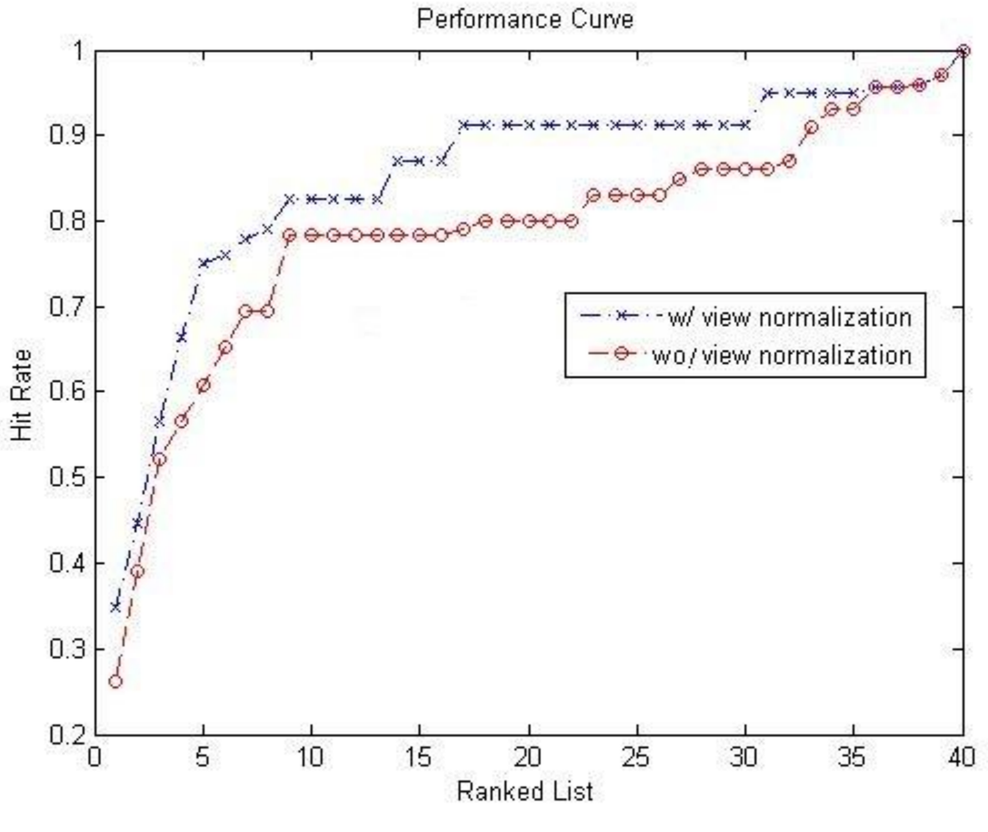


Figure 48: Matching performance of proposed method.

To select the optimal normalized teeth image size, we found that moving from image size 16x16 to 32x32 enhances the matching performance. Moving to 64x64 and 128x128 didn’t enhance the matching performance any more, while it takes more processing time. We returned this to the geometric properties of segmented teeth, as going to a higher dimension means that we will magnify most of the teeth images. Figure 49 shows the performance using various tooth image sizes, where 32x32 scaling yields the best performance.

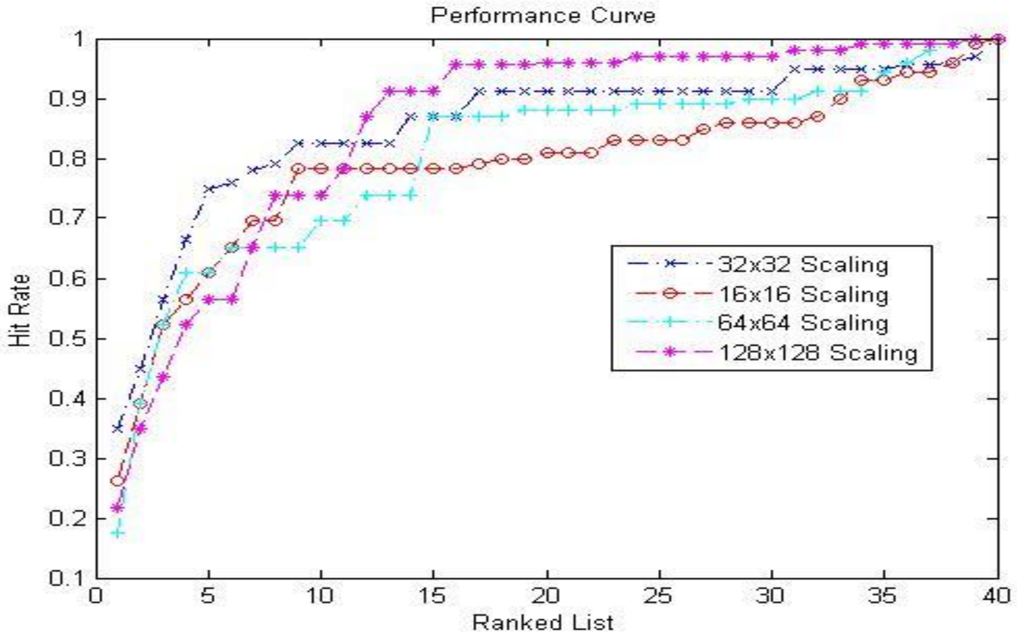


Figure 49: Performance comparison using four different scaling.

In [46], Abaza used a method Shape-Based Approach based on edge direction histogram. A histogram of the edge directions is used to represent the shape attribute. The edge information contained in the database images is generated in the preprocessing stage using edge operator such as Canny edge or Sobel edge operators. In [46] compute the similarity and distance between the subject image edge histogram and a stored reference image edge histogram. Their experiment results are based on same database which has the performance rates of 29% and 46% in the 1 and 5 top ranks respectively.

Chen et al. [72] proposed a method which uses the distance between the tooth contours and the distance between the dental works. Their experiment results are based on same database but with only 11 PM subjects that were matched to the 25 AM subjects. When compared to their approach we achieved slight increase in the accuracy (75% compared to 72%) with more number of subjects.

On the other hand, computational time comparison between the proposed method and the technique (Contour Alignment based) introduced in [46] , our proposed approach takes 6.28 seconds to compare record to record and the average search time is 6.28n seconds, where n is the number of records in the database (we measured the processing time for running the image comparison component using MATLAB® implementation running on a 2.66 GHz 4.0 GB RAM Intel® Core(TM)^2 Quad CPU PC platform), while the comparison technique introduced in [46] takes about 20 minutes to do a single record-to-record matching. To search a database of 50 records directly using the image comparison, the system will run for about 8 hours. The proposed method significantly reduces the search time while still maintaining a comparable accuracy. They measured the processing time for running the image comparison component using MATLAB® implementation running on a 2.66 GHz 1.0 GB RAM Intel® Pentium IV PC platform.

To compare the matching final result of SIFT descriptor extracted from the whole tooth segment and those guided by the tooth contour, we repeat the experiment as shown in Figure 50. We return performance of using contour guided SIFT to the fact that the contour is the most distinguishable region between teeth. Table 15 shows the significant difference in the time between the naïve SIFT descriptors and the contour guided ones. The contour guided SIFT significantly reduce the matching time, while yielding better performance.

Table 15
COMPUTIONAL TIME COMPARISION BETWEEN SIFT POINTS GUIDED BY TOOTH CONTOUR AND SIFT FROM THE WHOLE TOOTH SEGMENT.

	SIFT of tooth segment	**SIFT of tooth contour**
Average Tooth to Tooth	1.26 sec	0.94 sec
Average Record to Record	9.85 sec	6.28 sec
Average Record Retrieval	6.56 min	4.18 min

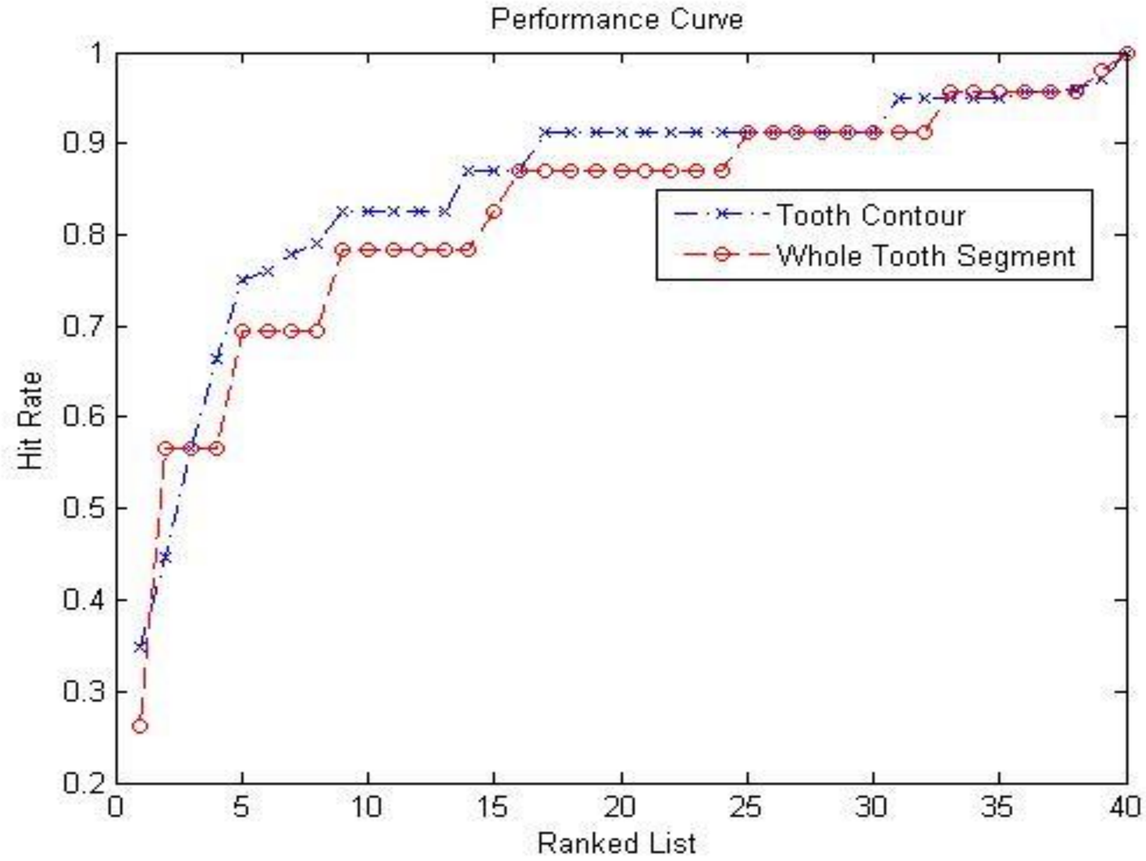


Figure 50: Matching performance using SIFT guided by the tooth contour and the SIFT points from the whole tooth segment.

6 Overall performance rate of ADIS

The present ADIS system is not fast and accurate due to some of its components in the algorithm. Some of the components in the algorithm consume more time and the entire algorithm does not have an acceptable level of accuracy. One needs to concentrate on the individual components to improve the performance in terms of speed and accuracy. Improving each individual module in the ADIS system will result in the increase of overall performance of the system both in speed and accuracy. The components which perform better in terms of speed and accuracy are replaced with the existing ones to improve the overall performance of the ADIS system. We view ADIS as a collection of the following components as shown in Figure 51:

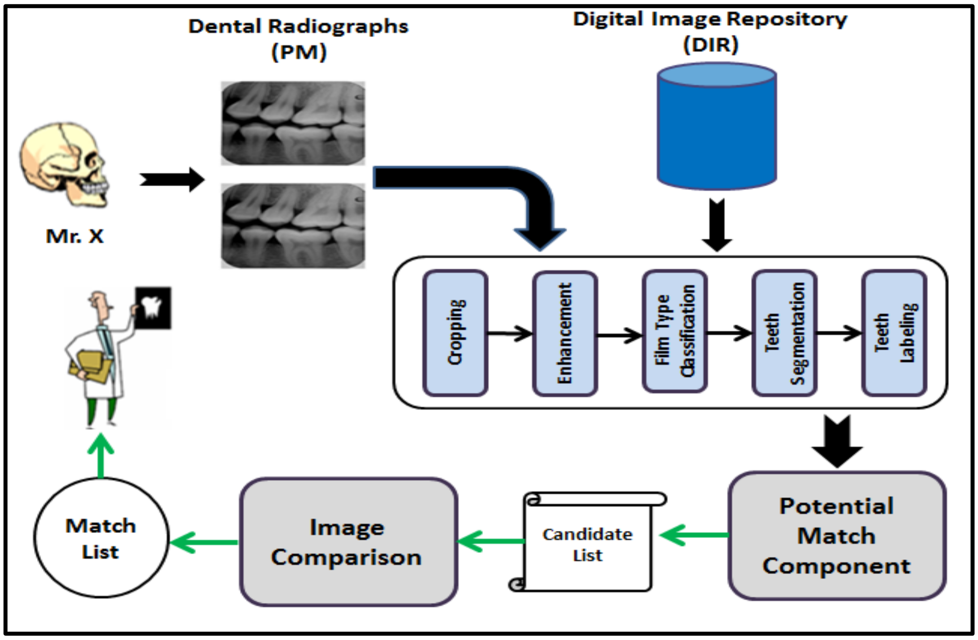


Figure 51: Block diagram of ADIS system.

(i) Record Preprocessing component which handles dental records cropping into dental films, grayscale contrast enhancement of films, classification of films into bitewing, periapical, or panoramic views, segmentation of teeth region from films, teeth contour extraction and finally annotating teeth with labels according to their location.

6. Overall performance rate of ADIS

(ii) Potential Matches Search component which manages archiving and retrieval of dental records based on high-level dental features (e.g. number of teeth and their shape properties) or dimensionality reduction and produces a candidate list.

(iii) Image Comparison component which mounts for low-level tooth-to-tooth comparison between subject teeth -after alignment- and the corresponding teeth of each candidate, thus producing a short match list.

Segmentation, labeling and matching are some of the components which are responsible for retrieving a less accurate matching list, as well responsible for a slow ADIS system. We proposed new techniques which will replace these components and increase the overall performance of the ADIS system.

For teeth segmentation subcomponent in the preprocessing component, we adapted the seam carving technique that was used for image resizing to the problem of segmenting dental image records into individual teeth. We have obtained optimality rate of 54.02% and a failure rate of 1.05% for bitewing view. And optimality rate of 58.13% and a low failure rate of 0.74 for periapical view. The proposed approach takes about averaging time of 2.56 seconds to segment bitewing dental film and approximately half of this time that require to segment periapical dental film that depends on the number of teeth per dental film.

For teeth classification subcomponent in the preprocessing component, developed a dual-stage approach for automatic construction of dental charts based on appearance-based features (LaplacianTeeth) and string matching technique. Enhanced the teeth classification performance by 10%. The average computational time required for the teeth classification of bitewing dental film is 15.30 seconds also that depends on the number of segmented teeth per dental film.

For potential match search, the main task of the of this component technique, that searches the dental database in a fast way to find a candidate list (references dental records) has high similarity with the subject dental record, and its size is very short compared to the original database size. Therefore, it will facilitate comparison image component (comparison on low level teeth features). We presented fast search engine techniques for potential matching problem based on LaplacianTeeth Images. By fusing multiple teeth matching scores, we calculate the

Euclidean distance between the subject record and all the reference records of the database. Then we retrieve records based on minimum distance. Testing results based on a sizable data-set suggest that appearance-based features are fast (average record retrieving time of 1.99 n seconds, where n is the number of records in the database) and search database of size 47 the system will run for averaging time of 91.65 seconds and achieve rank 5 accuracy of 65%.

Finally, we presented a new technique for dental record retrieving. The technique is based on the matching of the Scale Invariant feature Transform (SIFT) descriptors guided by the teeth contour between the subject and reference dental records. The proposed technique correctly retrieve the dental record 35% and 75% of the times in the 1 and 5 top ranks respectively, and takes only an average time of 4.18 minutes to retrieve a match list of size 47 records.

We conduct two experiments to measure the overall performance rate of the automated dental identification. In both the experiments, we used the same database of FBI’s Criminal Justice Service (CJIC) ADIS database [27], that contains 104 records (about 500 bitewing and periapical films) involving more than 2000 teeth, 47 Antemortem (AM) records and 57 Postmortem (PM) records with 20 matched records. First, we used the components of ADIS system presented in [47]. Then, we repeated the experiment by replacing the four components as shown in Figure 51, viz. teeth segmentation, teeth labeling, potential match component, and image comparison with proposed novel subcomponent algorithms that are stated in chapter 3.1, 3.2, 4, and 5 of this thesis.

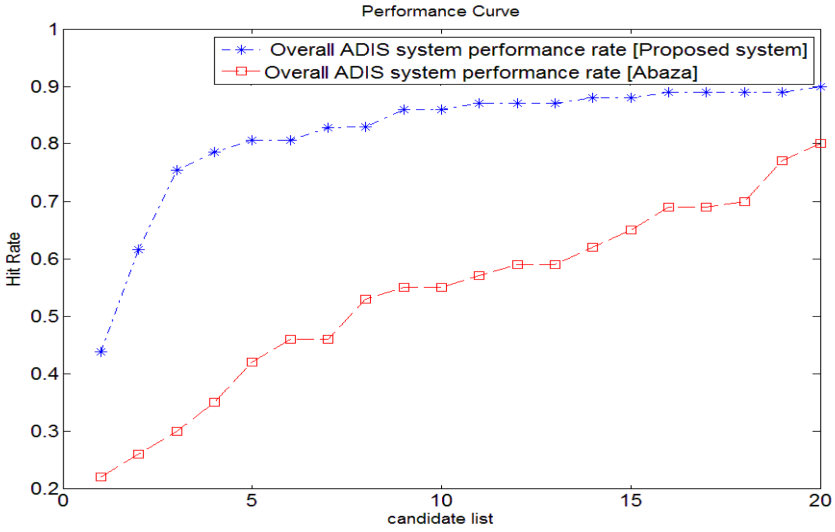


Figure 52: Performance comparison using the top 20 dental records from candidate list.

6. Overall performance rate of ADIS

In each experiment only the top 20 dental records that are acquired as an output from potential match search component were used (Instead of searching the whole database, only make a search and match on a small portion of candidate list) and fed into image comparison component. Figure 52, shows the comparison of overall performance rate of the of the automated dental identification system between the method presented in [47] and the proposed system. The proposed system accurately retrieves the dental record with an overall performance rate of 80% in the top 5 ranks while the other method in [47] remains at 42% in the top 5 ranks. Also, even with the increase in size of the first position of the candidate list, the proposed system remains ahead in precision to the system presented in [47].

Then, we repeated the above mentioned experiments by using smaller size of candidate list i.e. only the top 10 dental records (output of potential match component).Now, the proposed system as shown in the Figure 53 accurately retrieves the dental record with an overall performance rate of 84% in the top 5 ranks while the other method presented in [47] remains at 55% in the top 5 ranks. Also, as before even with the increase in size of the first position of the candidate list; the proposed system remains ahead in comparison to [47].

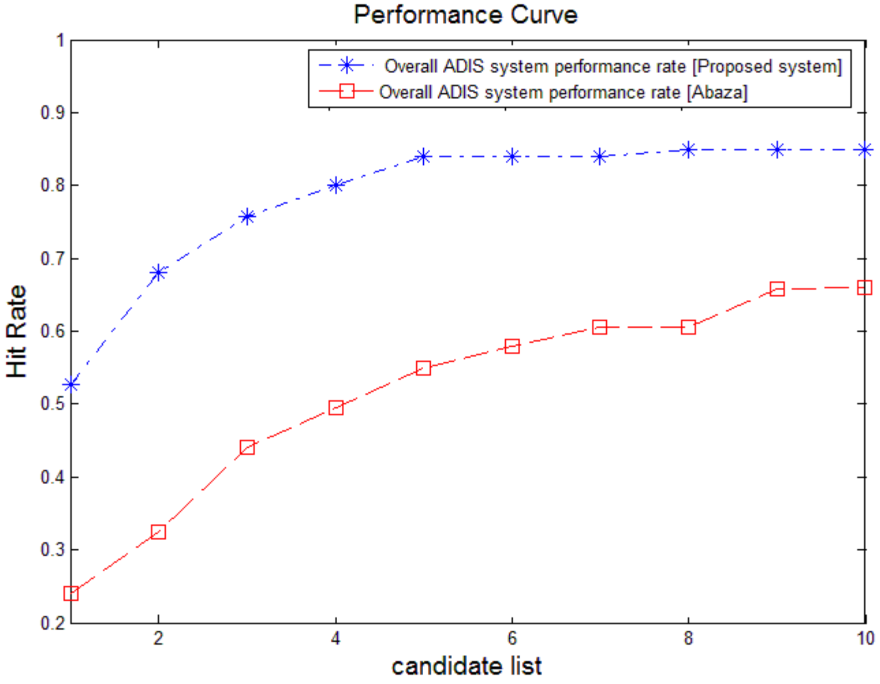


Figure 53: Performance comparison using the top 10 dental records from candidate list.

6. Overall performance rate of ADIS

In general, we have observed in both systems by reducing the size of the candidate list that was obtained from the potential match component the performance rate has been improved (i.e.; in the top 5 ranks) at the expense of decreasing the number of subject dental records which might not be included in the reduced candidate list. Furthermore, the proposed system achieved better overall performance rate compared with the ADIS system presented in [47].

7 Conclusion and Future Work

ADIS can be used by law enforcement agencies for resolving cases of Missing and Unidentified Person (MUP). The main contribution of this is to propose ADIS architecture that can handle a large database of images. ADIS architecture is split in to two main stages, where the first stage search the database for the candidate images, and the other stage reduces the number of candidate images to a match list of a few cases to be given to the forensic expert for a final decision. Samples of the performance results of this architecture were presented along with the description of the methodologies proposed. More extensive results of each component can be found in the list of ADIS publications. With a given dental record of a subject whose identity is to be determined, the purpose of an automated dental identification system (ADIS) is to speed-up the process of retrieving a candidate list of images from a set of reference records available in the database. The final process of matching the Post Mortem (PM) and Ante Mortem (AM) dental records is preceded by a set of pre-processing layers to enhance the performance of the system in terms of accuracy, speed and robustness. We presented different problems regarding building a prototype for an Automated Dental Identification System:

- We adapt the seam carving technique that was previously used for image resizing to the problem of segmenting dental image records into individual teeth. Our approach is heavily based on concepts of seam carving technique. We propose a teeth segmentation approach for segmenting the dental images. In the first stage, the teeth images are preprocessed by a two-step thresholding technique, which starts with an iterative thresholding followed by an adaptive thresholding to binarize the teeth images. In the second stage we adapt the seam carving technique on the binary images, using both horizontal and vertical seams, to separate each individual tooth. We have obtained an optimality rate of 54.02%, which is superior to all existing fully automated dental segmentation algorithms in the literature, and a failure rate of 1.05%. For the periapical view, we have obtained a high optimality rate of 58.13% and a low failure rate of 0.74. We measured the teeth segmentation time required for segmenting a bitewing record using an MATLAB® implementation running on a 2.66 GHz 4 GB RAM Intel® Core(TM) Quad CPU PC platform. The average teeth segmentation time of one bitewing record is 2.59 sec and it varies depending on the number of teeth in the dental film.

- For the teeth labeling problem, we adapted a dual-stage approach for automatic construction of dental charts based on appearance-based features and string matching with don't care possibility. Initially, each segmented tooth (of a bitewing/periapical film) is independently classified based on teeth reconstruction in to two/four image subspaces. Using teeth-neighborhood rules the initial teeth class assignments are validated and possibly corrected even in the presence of missed tooth. Finally, if the resulting teeth class sequence is unique, a number for each tooth is assigned corresponding to its position in the dental quadrant it belongs to. Otherwise, if the resulting teeth class sequence is non-unique we consider it as a reject. Including this previous knowledge about class of the dental view, enhances the initial classification by 10%. Experimental results based on a sizable dataset suggest that appearance-based features achieve fast classification with an average film classification time of 15.30 seconds with competitive accuracy compared to methods proposed in the literature. Overall around 77% of the teeth are initially assigned to their proper classes, 87% of the teeth are assigned proper class labels after validation.
- For the potential match problem, we presented fast searching engine techniques based on LaplacianTeeth Images. By fusing multiple teeth matching scores, we calculate the Euclidean distance between the subject record and all the reference records of the database. Then we retrieve records based on minimum distance. Testing results based on a sizable data-set suggest that appearance-based features are fast (average record retrieving time of 1.99 n seconds, where n is the number of records in the database) and achieve rank 5 accuracy of 65%.
- We address the teeth matching problem by presenting a new technique for dental record retrieving. The technique is based on the matching of the Scale Invariant Feature Transform (SIFT) descriptors guided by the teeth contour between the subject and reference dental records. Our fundamental objective is to accomplish a relatively short match list, with a high probability of having the correct match reference. Matching these SIFT descriptors between subject and reference results in matching score. To move from the tooth-to-tooth matching level to record-to-record matching level, we use the mean rule to obtain record-to-record matching score. Finally, the matching score between a

subject record and a reference record will be sent to the match list. The proposed technique correctly retrieve the dental record 35% and 75% in the 1 and 5 top ranks respectively, and takes only an average time of 4.18 minutes to retrieve a match list of size 47 records.

- The overall performance rate of the proposed ADIS (with using potential match component for reduction the candidate list). We obtained an overall rate of 80% in the top 5 ranks using a candidate list of size 20, and an overall rate of 84% in the top 5 ranks using a candidate list of size 10.

Challenges:

- The dental radiographs obtained from CJIS are not consistent as there are variations in the use of sensors, few portions of the images are brighter than the rest due to the differences in lighting conditions and also the dental X-rays have been taken from different viewpoints and at different instances of time.
- Some dental images may contain cuspid teeth, which are mostly like premolar teeth, Wisdom teeth, which are members of the molar teeth, but are in different shape compared to other molar teeth. Those cases may confuse the classifier leading to misclassification.

Future Work:

- **For teeth segmentation:**
  - i. Develop a fully automated segmentation evaluator of segmentation outcomes in order to utilize properly segmented images, because the improper segments may affect the accuracy of the identification process and increase the time of the process.
  - ii. Developing more efficient energy functions which do not depend on the intensity of the raw pixels in the binary image to further improve dental segmentation using seam carving technique has to be studied.
- **For teeth labeling:**
  - i. Explore an alternate cost functions which are more robust to initial teeth classification results.

ii. Explore the use of weighted costs for changing the class labels, where weights could be derived based on the specific domain knowledge of human teeth.

iii. Investigating whether filling the detected dental works using image-in-painting techniques will help.

iv. Learning different geometric properties of teeth shape.

v. Fusion between the appearance based and the feature based techniques for better labeling.

- **For matching problem:**

i. Fusion between techniques based on local features and global features.

ii. Research methods of adding weights to teeth matching score to improve identification performance.

iii. Till now, research in automated dental identification has dealt with two dimensional radiographic films. Using 3 dimensional views for possible enhancement in the dental identification has to be investigated. On the other hand, using multiple 2D views of tooth acquired from different angles to reconstruct corresponding 3D tooth is worth considering.

I. Glossary

Radiograph: An x-ray picture.

Bitewing: A kind of dental x-ray which is taken with the teeth bite together. The main function of this kind of x-ray is to detect cavity in between teeth and height of bone support.

Periapical: The surrounding of the bottom of the root of a tooth.

Panoramic Radiograph: An x-ray film to obtain the wide view of upper and lower jaw and their associated structures.

Permanent teeth: Adult's teeth. The first permanent tooth usually comes in around 6 years old.

Filling: A restoration places on a tooth to restore its function and appearance.

Primary teeth: Baby teeth.

Molar: The last 3 upper and lower teeth on both side of the mouth.

Wisdom tooth: The eighth (also the last tooth) tooth from the middle of the jaw.

Premolar: The two teeth located in front of the molar.

Incisor: The four upper and lower front teeth.

Canine: The third tooth from the middle of the jaw. There are totally 4 of them. They are the longest teeth in human.

Crown: A crown is almost like a "cap" on a tooth. It covers the tooth partially or totally above the gum to restore its function and outlook.

Root: The bottom part of tooth. It anchors the tooth to its supporting units.

Seam: a continuous series of pixels that is a connected path of low energy pixels in an image that can be either vertical or horizontal.

Dentistry: A branch of medicine that involves diagnosis, prevention, and treatment of any disease concern about teeth, oral cavity, and associated structures.

Forensic dentistry: or forensic odontology is the proper handling, examination and evaluation of dental evidence, which will be then presented in the interest of justice.

II. Bibliography

[1] Jain, A.K., Bolle, R. and Pankanti S. (eds.) *Biometrics: Personal Identification in Networked Society*. Kluwer, New York, 1999.

[2] Anil Jain, Lin Hong and Sharath Pankanti, "Biometric Identification", *Communication of The ACM,* vol. 43, no. 2, pp. 91-98, Feb. 2000.

[3] "American society of forensic odontology," Forensic Odontology News, vol. 16, no. 2, Summer 1997.

[4] The Canadian Dental Association, *Communique*. May/June 1997.

[5] Gust, and Ghosta, "*Forensic Odontology*", American Elsevier Pub. Co. 1996.

[6] P. Stimson, and C. Mertz, "Forensic Dentistry", CRC Press 1997.

[7] United States United States Army Institute of Dental Research Walter Reed Army Medical Center,Computer Assisted Post Mortem Identification via Dental and other Characteristics-
USAIDR Information Bulletin. Vol. 5, No. 1, Autumn 1990.

[8] Dr. J. McGivney et al. WinID2 software available (www.winid.com.).

[9] L. Lorton, M. Rethman, and R. Friedman, "The Computer-Assisted Postmortem Identification (CAPMI) System: A Computer-Based Identification Program," Journal of Forensic Sciences (JFSCA), vol. 33, no. 4, pp. 977-984, July 1988.

[10] G. Fahmy, D. Nassar, E. Haj-Said, H. Chen, O. Nomir, J. Zhou, R. Howell, H. H. Ammar, M. Abdel-Mottaleb and A. K. Jain, "Towards an Automated Dental Identification System (ADIS)", Proc. the International Conference on Biometric Authentication (ICBA), pp. 789- 796, Hong Kong, July 2004.

[11] J. McGivney. Commentary on: Lewis C. WinID2 veerses CAPMI4: Two Computer- Assisted Dental Identification Systems. J Forensic Sci 2002;47(3) pp. 536-538.

[12] C. Lewis, and L. Leventhal, Locator system versus WinID3 versus CAPM14: Identifying Vicimsfrom Dental Remains in a Large Disaster. J of Forensic Identification 2004; 52(4) pp. 185-202.

[13] CM. Bowers ,RJ. Johansen . Digital Imaging Methods As an Aid in Dental Identification of Human Remains. J Forensic Sci 2002;47(2): pp. 354-359.

[14] G. Fahmy, D. Nassar, E. Haj Said, H. Ammar, M. Abdel-Mottaleb, H. Chen, A. Jain, "Toward an Automated Dental Identification System", Journal of Electronic Imaging, vol. 14, no. 4, 2005.

[15] National Dental Image Repository website, http://foia.fbi.gov/ndirpia.htm.

[16] Integrated Automated Finger Identification System, http://www.fbi.gov/hq/cjisd/iafis.htm.

[17] A. Jain and H. Chen, “Dental Biometrics: Matching Dental X-rays for Human Identification”, Proc. of the Biometric Consortium Conference (BCC), Crystal City, Arlington – VA, 2003.

[18] D. Nassar, E. Haj Said, G. Fahmy, R. Howel, and H. Ammar, “Challenges of developing an Automated Dental Identification Systems”, Proc. of the Biometric Consortium Conference (BCC), Crystal City, Arlington – VA, 2003.

[19] M. Abdel-Mottaleb, O. Nomir, D. Nassar, G. Fahmy, and H. Ammar, “Challenges of Developing an Automated Dental Identification System", Proc. of the IEEE mid-west symposium for circuits and systems, Cairo - Egypt, Dec 2003.

[20] D. Nassar, and H. Ammar, “A Prototype Automatic Dental Identification System”, Proc. of the 4th National Conference on Digital Government Research (dg.o.), Boston – MA, 2003.

[21] G. Fahmy, D. Nassar, E. Haj Said, H. Chen, O. Nomir, J. Zhou, R. Howell, H. Ammar, M. Abdel-Mottaleb and A. Jain, “Automated Dental Identification System (ADIS)”, Proc. of the 5th National Conference on Digital Government Research (dg.o.), May 2004.

[22] H. Chen and A. Jain, “Dental Biometrics: Matching of Dental Radiographs for Human Identification”, Proc. of the Biometric Consortium Conference (BCC), Crystal City, Arlington – VA, 2004.

[23] D. Nassar, E. Haj Said, A. Abaza, S. Chekuri, J. Zhou, M. Mahoor, O. Nomir, H. Chen, G. Fahmy, H. Ammar, M. Abdel-Mottaled and A. Jain, “Automated Dental Identification System (ADIS)” Proc. of the 6 th National Conference on Digital Government Research (dg.o.), Atlanta - GA, May 2005.

[24] X. Li, A. Abaza, D. Nassar, and H. Ammar, “Fast and Accurate Segmentation of Dental X-ray Records”, Proc. of the 2nd international Conference on Biometric Authentication (ICBA), Hong Kong, Jan 2006.

[25] Eyad Haj Said, ACCURATE SEGMENTATION OF DIGITIZED DENTAL X-RAY RECORDS

[26] CJIS Division – Washington State Patrol Missing and Unidentified Person Unit, Digitized Dental Images (Database), May 2000.

[27] CJIS Division – ADIS project, Digitized Radiographic Images (Database), August 2002.

[28] E. Haj Said, D. Nassar, G. Fahmy, and H. Ammar, “Teeth Segmentation in Digitized Dental X-Ray Films Using Mathematical Morphology” IEEE Transactions on Information Forensics and Security, vol. 1, no. 2, June 2006.

[29] E. Haj Said, D. Nassar, G. Fahmy, and H. Ammar, “Dental X-ray Image Segmentation”, Proc. of SPIE Technologies for Homeland Security and Law Enforcement conference, Orlando - FL, Apr 2004.

[30] E. HajSaid, D. Nassar, and H. Ammar, "Image segmentation for automated dental identification", Proc. of IS&T/SPIE Electronic Imaging, San Jose - CA, Jan 2006.

[31] S. White, and M. Pharoah, *Oral Radiology Principles and Interpretation*. Mosby, Inc. Fourth Edition 2000.

[32] J. Zhou, and M. Abdel-Mottaleb, "A Content-based System for Human Identification based on Bitewing Dental X-Ray Images", Pattern Recognition, vol. 38, pp. 2132-2142, Nov 2005.

[33] J. Zhou, and M. Abdel-Mottaleb, "Automatic Human Identification Based on Dental X-ray Images", Proc. of SPIE Technologies for Homeland Security and Law Enforcement conference, Orlando - FL, Apr 2004.

[34] A. K Jain and H. Chen, "Matching of Dental X-ray Images for Human Identification", Pattern Recognition, vol. 37, pp. 1519-1532, 2004.

[35] O. Nomir, M. Abdel-Mottaleb, A system for human identification from X-ray dental radiographs, J. Pattern Recognition 38 (2005) 1295–1305.

[36] D. E. M. Nassar, F. U. Chaudhry, and H. H. Ammar, "On Performance Evaluation of Image Segmentation Algorithms: Success is Not All or None", Proc. the 1st International Computer Engineering Conference (ICENCO' 2004), pp. 354-359, Egypt, Dec. 2004.

[37] S. Shah, A. Abaza, A. Ross and H. Ammar, "Automatic Teeth Segmentation Using Active Contour Without Edges" Proc. of the Biometric Consortium Conference (BCC), Baltimore - MD, Sep 2006.

[38] M. Mahoor, and M. Abdel-Mottaleb, "Classification and Numbering of Teeth in Dental Bitewing Images", Pattern Recognition, vol. 38, pp.577-586, April 2005.

[39] M. Mahoor, M. Abdel-Mottaleb, "Automatic Classification of Teeth in Bitewing Dental Images", Proc. of the international Conference on Image Processing (ICIP), Singapore, Oct 2004.

[40] A. Jain and H. Chen, "Registration of dental atlas to radiographs for human identification", *Proc. of SPIE Conference on Biometric Technology for Human Identification II*, vol. 5779, pp. 292-298, Orlando, Florida, Mar 2005.

[41] D. Nassar, "Automated Dental Identification: A Micro-Macro Decision-Making Approach", Ph.D. Dissertation, Lane Department of Computer Science and Electrical Engineering, West Virginia University, Dec. 2005.

[42] Hong Chen and Anil K. Jain, .Tooth Contour Extraction for Matching Dental Radiographs., Proc. ICPR 2004, vol. III, pp. 522-525, Aug. 2004, Cambridge, UK.

[43] H. Chen and A. Jain, “Dental Biometrics: Alignment and Matching of Dental Radiographs”, Proc. of Workshop on the applications of Computer Vision (WACV), pp. 316-321, Breckenridge, Colorado, Jan 2005.

[44] Nomir and Abdel-Mottaleb, “Combining Matching Algorithms for Human Identification using Dental X-Ray Radiographs”, Processing, 2007. ICIP 2007. IEEE International Conference on , vol.2, no., pp.II-409-II-412, Sept. 16 2007-Oct. 19 2007.

[45] D. Nassar, and Hany Ammar, “A Neural Network System for Matching Dental Radiographs”, Pattern Recognition, vol. 40, pp. 65-79, Jan 2007.

[46] A. Abaza, A. Ross, and H.H. Ammar, "Retrieving dental radiographs for post-mortem identification", in Proc. ICIP, 2009, pp.2537-2540.

[47] A Abaza, “High Performance Image Processing Techniques In Automated Identification System”, Ph.D. Dissertation, Lane Department of Computer Science and Electrical Engineering, West Virginia University, April 2008.

[48] S. Chekuri, D. Nassar, A. Abaza, A. Bahu, H. Ammar, and G. Fahmy, “ A web-based Automated Dental Identification System (webADIS)”, Proc. of the 5th IBIMA International Conference on Internet & Information Technology in Modern Organizations, Cairo - Egypt, Dec 2005.

[49] S. Chekuri, D. Nassar, A. Abaza, E. HajSaid, A. Bahu, U. Qurashi, G. Fahmy, and H. Ammar, “webADIS: A Flexible web-based Enviroment for the Automated Dental Identification System” Proc. of the 7th Annual International Conference on Digital Government Research (dg.o.), San Deigo – CA, May 2006.

[50] S. Chekuri, D. Nassar, A. Abaza, A. Bahu, U. Qureshi, and H. Ammar, “webADIS: A Flexible web-based Environment for the Automated Dental Identification System”, Proc. of the 2nd International Computer Engineering Conference (ICENCO), Cairo - Egypt, Dec 2006.

[51] R. Gonzalez, R. Wood, Digital Image Processing, Addison Wesley, Reading, MA, 1993.

[52] E. Davies, Machine Vision, Academic Press, New York, 1990 pp. 91–96.

[53] S. Hu, E.A. Huffman, M. Reinhardt, Automatic lung segmentation for accurate quantization of volumetric X-Ray CT images, IEEE Trans. Med. Imaging 20 (6) (2001) 490–498.

[54] R. Jain, R. Kasturi, B.G. Schnck, Machine Vision, McGraw-Hill Inc., New York, 1995.

[55] C. Giardina, E. Doughetr, Morphological Methods in Image and Signal Processing, Prentice-Hall, Englewood Cliffs, NJ, 1988.

[56] Avidan S, Shamir A. Seam carving for content-aware image resizing, ACM Trans Graph (SIGGRAPH), 2007, 26(3): 10–18.

[57] B.G. Brogdon, M.D., “Forensic Radiology”, CRC Press, Boca Raton, 1998.

[58] D. Nassar, A. Abaza,Xin Li, and H. Ammar, “Automatic Construction of Dental Charts for Postmortem Identification” IEEE Transactions on Information Forensics and Security, vol. 3, no. 2, pp. 234–246, 2008.

[59] S. Bunn, http: http:// www.drbunn:com=number:htm.

[60] S. Vasuhi, and V. Vaidehi, "Identification of Human Faces Using Orthogonal Locality Preserving Projections," 2009 International Conference on Signal Processing Systems, vol., no., pp.718-722, 15-17 May 2009.

[61] T. Cormen, C. Leiserson, and R. Rivest, Introduction to Algorithms, New York: McGraw-Hill, 1990.

[62] R. Baeza-Yates, Algorithms for string searching: A survey, ACM Special Interest Group on Information Retrieval (SIGIR) Forum, vol. 23, no. 3-4, pp. 34-58, 1989.

[63] R. Wagner and M. Fischer, “The String-to-String Correction Problem”, Journal of the Association for Computing Machinery (JACM), vol. 21, no. 1, p.168-173, 1974.

[64] Slaby, A."ROC Analysis with Matlab," Information Technology Interfaces, 2007. ITI 2007. 29th International Conference on , vol., no., pp.191-196, 25-28 June 2007.

[65] Samir Shah, Ayman Abaza, Arun Ross, and Hany Ammar, “Automatic tooth segmentation using active contour without edges,” in Proc. of Biometric Consortium Conference, USA, 2006.

[66] Kumar, B.G.V.; Aravind, R.; "Face hallucination using OLPP and Kernel Ridge Regression," Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on , vol., no., pp.353-356, 12-15 Oct.

[67] Deng Cai, Xiaofei He, Jiawei Han, and Hong-Jiang Zhang, "Orthogonal Laplacianfaces for Face Recognition," Image Processing, IEEE Transactions on , vol.15, no.11, pp.3608-3614, Nov. 2006.

[68] Luo Juan and Oubong Gwun. A Comparison of SIFT, PCA-SIFT and SURF. International Journal of Image Processing (IJIP), Volume (3): Issue (4) 2009.

[69] T. Chan and L. Vese, “Active contours without edges”, IEEE Trans. on Image Processing, vol. 10, no. 2, pp. 266–277, 2001.

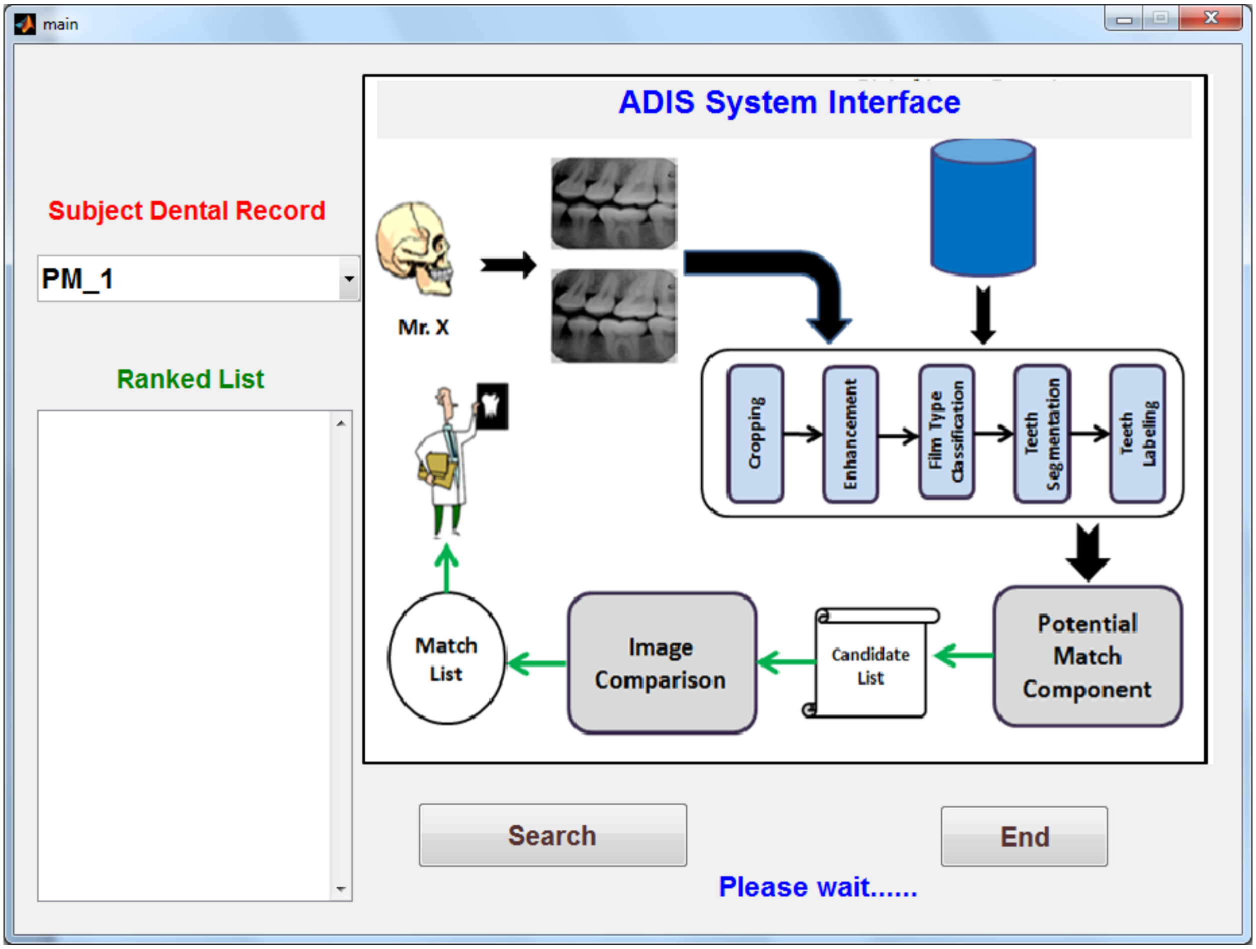
[70] M. Kass, A. Witkin, and D.Terzopoulos, "Snakes: Active contour models", International Journal of Computer Vision, vol. 1, no. 4, pp. 321-331, 1987.

[71] D. G. Lowe, “Object recognition from local scale-invariant features,” 7th, International Conference on Computer Vision, vol.2. pp.1150, 1999.

[72] U. Park, S. Pankanti and A. K. Jain, "Fingerprint Verification Using SIFT Features", Proceedings of SPIE Defense and Security Symposium, Orlando, Florida, 2008.

[73] H. Chen and A. K. Jain, “Dental biometrics: Alignments and matching of dental radiographs,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, no. 8, pp. 1319–1326, Aug. 2005.

III. Appendix



Subject Dental Record

PM_1

Ranked List

Please wait......

```
function varargout = main(varargin)
% ****************  Main Interface of ADIS System Program
*****************
% The user  must select subject record (PM)from subject record
list then, the main program calls the following functions, as
follows:
% MatADIS_RecordCrop Function: the main task of this function
extrcing the dental films from their corresponding denatl
record.
% inpunt: subject dental record.
% output: cropping films and their coordinates.
% MatADIS_FilmClassify Function: detect the film type: ' Bite-
wing',
%                'Periapical Upper', ' Periapical Lower', '
Panoramic'.
% MatADIS_TeethSegment Function: extracting teeth from their
corresponding dental film.
% inpunt:CroppingInfo, CroppedRecord, and Dental Film types
% output: SegmentationInfo and SegmentedFilm
% MatADIS_TeethLabel Function: Classify teeth into molar,
premolar, canine,
% and insorisr. then assign numbering them corresponding to
their position in the denatl chart.
% inpunt: SegmentedFilm and SegmentationInfo
% output: LabelInfo and  LabeledImg
%
*****************************************************************
*********
%      MAIN, by itself, creates a new MAIN or raises the
existing
%      singleton*.
%%      H = MAIN returns the handle to a new MAIN or the handle
to
%      the existing singleton*.
%
%      MAIN('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in MAIN.M with the given input
arguments.
%
%      MAIN('Property','Value',...) creates a new MAIN or raises
the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before main_OpeningFcn gets called.
An
%      unrecognized property name or invalid value makes
```

```
property application
%      stop.  All inputs are passed to main_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI
allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @main_OpeningFcn, ...
                   'gui_OutputFcn',  @main_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin
{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before main is made visible.
function main_OpeningFcn(hObject, eventdata, handles, varargin)

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

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

% Update handles structure
guidata(hObject, handles);

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


% --- Outputs from this function are returned to the command
line.
function varargout = main_OutputFcn(hObject, eventdata, handles)
clc
set(handles.listbox1,'string','')

subject_list = 'PM_1|PM_6|PM_32|PM_68|PM_72|PM_77|PM_79|PM_84
|PM_87|PM_88|PM_91|PM_92|PM_93|PM_94|PM_95|PM_96|PM_100|PM_102
|PM_103|PM_106|PM_107|PM_108';

handles.subject_list =subject_list;
set(handles.popupmenu1,'string',handles.subject_list);
handles.Avg = 1;
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
% varargout  cell array for returning output args (see
VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
%
****************************************************************
*****
pat1 = pwd;
subject_path = [pat1  '\im7.bmp'];
% image_directory = dir(FileName)
axes(handles.axes1);imshow(imread(subject_path),[]);
handles.subject_path = subject_path;
%
****************************************************************
*****
% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB handles structure with handles and user data (see
GUIDATA)
```

```
% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenu1 contents as cell array contents{get(hObject,'Value')}
returns selected item from popupmenu1

contents = get(hObject, 'String');
pop_val = get(hObject,'Value');

% --- Executes during object creation, after setting all
properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on selection change in listbox1.
function listbox1_Callback(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox1 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox1
% --- Executes during object creation, after setting all
properties.
function listbox1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.text4,'Visible','on')
%set(handles.pushbutton2,'enable','off')
index = get(handles.popupmenu1,'Value');
index = index +2;
%
*****************************************************************
*********
% Input:  ImgRecord
% Output: CroppingInfo
% Input Details:
%               - ImgRecord: an 8-bit grayscale image of dental
record to be
%               cropped.
%               - Parameters: A parameter structure with the
following
%               fields
%                             >mthd is an indicator of the
method used for
%                             cropping
% Output Details:
%               - CroppingInfo: An Array of N*4 integers where N
is the
%               number of cropped films, each row looks like [Xo
Yo W H],
%               where Xo,Yo are the row and column indices
(respectively)
%               of the upper left corner of a cropped film and
W,H are the width
%               and height (respectively) of that film.
%               - CroppedRecord: an image showing bounding
rectangles of films
%               overlaid on the input record.
%
=================================================================
============================
% Developed by: Diaa Eldin M. Nassar & Ayman Abaza
%                West Virginia University
%                The ADIS Research project
%                (c) 2005 Diaa Eldin M. Nassar & Ayman Abaza -
Feb. 2005
%
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
[CroppingInfo, CroppedRecord] = MatADIS_RecordCrop(ImgRecord);
%
===============================================================
===========================
%% Input:  InFilm
% Output: FilmClassInfo, OutClass
% Input Details:
%                - InFilm: an 8-bit grayscale image of dental
film whose
%                type is to be determined.

% Output Details:
%                - FilmClassInfo: A vector of film type
classification results.
%                - OutClass: a string specifying film type
classification
%                decision as one of the following: ' Bite-wing',
%                'Periapical Upper', ' Periapical Lower', '
Panoramic'.
%
===============================================================
===========================
%
===============================================================
===========================
% Developed by: Diaa Eldin M. Nassar & Ayman Abaza
%                West Virginia University
%                The ADIS Research project
%                (c) 2005 Diaa Eldin M. Nassar  & Ayman Abaza -
Feb. 2005
%
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
% Please contact the authors for permission.
%
===============================================================
===========================
 [FilmClassInfo, OutClass] = MatADIS_FilmClassify
```

```
(CroppingInfo,CroppedRecord);
 %
================================================================
============================
% ****************  Teeth Segmetation Program
*************************************************
% Output: SegmentationInfo, SegmentedFilm
% Input Details:
%               - InFilm: an 8-bit grayscale image of dental
film from which
%               teeth are to be segmented.
%
% Output Details:
%               - SegmentationInfo: An Array of N*4 integers
where N is the
%               number of segmented teeth, each row looks like
[xo yo h w],
%               where xo,yo are the row and column indices
(respectively)
%               of the upper left corner of the bounding
rectangle of a tooth
%               and h,w are the height and width (respectively)
of that rectangle.
%               - SegmentedFilm: an image showing bounding
rectangles of teeth
%               overlaid on the input film.
% Programmer Name: Nourdin Al-sherif
%                West Virginia University
%                The ADIS Research project
%                (c) 2013 Nourdin Al-sherif. 2010
%% No part of this code may be copied, replicated or used in any
form without the explicit consent of the authors.
[SegmentationInfo, SegmentedFilm] = MatADIS_TeethSegment
(CroppingInfo,CroppedRecord);
% Programmer Name: Nourdin Al-sherif
%                West Virginia University
%                The ADIS Research project
%                (c) 2013 Nourdin Al-sherif. 2010

% ****************  Teeth Labeling Program
*************************************************

% Input:  InFilm, SegmentationInfo, SegmentedFilm
% Output: LabelInfo,  LabeledImg
% Input Details:
%               - SegmentedFilm: an 8-bit grayscale image of
dental film for which
```

```
%               teeth labels are to be assigned.
%               - SegmentationInfo: An Array of N*4 integers
where N is the
%               number of segmented teeth, each row looks like
[xo yo w h],
%               where xo,yo are the row and column indices
(respectively)
%               of the upper left corner of the bounding
rectangle of a tooth
%               and w,h are the width and height (respectively)
of that rectangle.
% Output Details:
%               - LabelInfo: An N*3 array of strings where N is
the number of teeth
%               labels, which is the same as the number of
segmented teeth. Row i
%               contains the label assigned to tooth segment i
of Infilm. The first two
%               characters of the string specify the jaw segment
where tooth is located,
%               R or L are used in the first character to
indicate left of right, while
%               X or D are used in the second character to
indicate maxilla or mandible.
%               The third character of the string specifies the
tooth type,
%               1 and 2 are the first and second incisors
respectively, 3 indicates a canine,
%               4 and 5 are the first and second premolars
respectively, 6 and 7 are the first
%               and second molars, and 8 is the wisdom tooth.
%               - LabeledImg: an RGB image showing labels
overlaid on the film.
%               colors ar used for labels only.
%
================================================================
============================
% Programmer Name: Nourdin Al-sherif
%               West Virginia University
%               The ADIS Research project
%               (c) 2013 Nourdin Al-sherif. 2012
[LabelInfo,  LabeledImg] = MatADIS_TeethLabel(SegmentedFilm,
SegmentationInfo);
%
================================================================
============================
AM_OUT =in_gui5(index);
```

```
set(handles.listbox1,'string',AM_OUT);
%set(handles.text4,'string','RR');
set(handles.text4,'visible','off')
%set(handles.pushbutton2,'enable','on')
% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

close
% --- Executes during object creation, after setting all
properties.
function text4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to text4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called
```

```
function  AM_OUT =in_gui5(subj)
%
-----------------------------------------------------------------
--------
% --------- matching subject record with the all references
records.
% input: subject and all reference record images;
% output: retuen with list to main interface;
% Programmer Name: Nourdin Al-sherif
%               West Virginia University
%               The ADIS Research project
%               (c) 2013 Nourdin Al-sherif. 2012
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
%
-----------------------------------------------------------------
--------

% The subject Record of PM.
PM_F = subj;
% The result of matching will be written into ranked_list.txt
file.
fid1 = fopen('ranked_list.txt','w');
pat = pwd;
path=[pat '\test_im1\'];
% Read references records.
DAM= dir([path 'AM']);
[NumDirAM Dummy]= size(DAM);

% Reading the Reference Recods : AM.
RefList = [];
for k=3:49%NumDirAM,
%     clearvars -except  k RefList;
    if (DAM(k).isdir)
        D=dir([path 'AM\' DAM(k).name]);
        [NumDir Dummy]= size(D);
        for i=3:NumDir,
            if (D(i).isdir)
            else
                [PATHSTR,NAME,EXT] = fileparts(D(i).name);
                if (strcmp(lower(EXT),'.bmp')|strcmp(lower
(EXT),'.tif')|strcmp(lower(EXT),'.tiff')|strcmp(lower
(EXT),'.jpeg')|strcmp(lower(EXT),'.jpg')|strcmp(lower
(EXT),'.pgm'))
                    ImageName = [path 'AM\' DAM(k).name '\' NAME
EXT];
```

```
                    RefList = [RefList;ImageName];
                end
            end
        end
    end
end
% Reading the subject: PM.
for L=PM_F:PM_F %NumDirPM
DPM= dir([path 'PM']);
[NumDirPM Dummy]= size(DPM);
SubFile = [];
    if (DPM(L).isdir)
        DS=dir([path 'PM\' DPM(L).name]);
        [NumDir Dummy]= size(DS);
         j=3;
            if (DS(j).isdir)
            else
                [PATHSTR,SNAME,EXT] = fileparts(DS(j).name);
                if (strcmp(lower(EXT),'.bmp')|strcmp(lower
(EXT),'.tif')|strcmp(lower(EXT),'.tiff')|strcmp(lower
(EXT),'.jpeg')|strcmp(lower(EXT),'.jpg')|strcmp(lower
(EXT),'.pgm'))
                    ImageName = [path 'PM\' DPM(L).name '\'
SNAME EXT];
                   SubFile = [SubFile;ImageName];
                end
            end
    end

     % call Matching search based on sift descriptor of tooth
contour.
     all_img_scr = MatchSearch1(RefList,SubFile);
     % display result and write it into ranked_list.txt
    [trash idx] = sort([all_img_scr{:,2}], 'descend');
    pair_ref = pairs2(SNAME);
    fprintf(fid1,'\nSubject image : %s',SNAME);
    fprintf(fid1,'    <<======>>  %s\n ',pair_ref);

    fprintf('\nSubject image : %s',SNAME);
    fprintf('    <<======>>    %s\n ',pair_ref);

    out_idx =0;
    for no = 1 : size(idx,2)
        fprintf('\n%d',no);
        fprintf('    %s',all_img_scr{idx(no),1});

        out_idx = out_idx+1;
        s_pm = all_img_scr{idx(no)};
        AM_OUT{out_idx} =s_pm;

        fprintf('    %3.2f',all_img_scr{idx(no),2});

        fprintf(fid1,'\n%d',no);
        fprintf(fid1,'    %s',all_img_scr{idx(no),1});
        fprintf(fid1,'    %3.2f',all_img_scr{idx(no),2});

    end
      fprintf
(fid1,'\n----------------------------------------------');
      fprintf('\n');
end
 fclose(fid1);
```

```
function all_img_scr = MatchSearch1(RefList,SubFile)
%
-----------------------------------------------------------------
--------
% --------- Reading all the information about references and
subject records.
% input: subject and all reference record images;
% output: the all of score of matching on record level;
% Programmer Name: Nourdin Al-sherif
%              West Virginia University
%              The ADIS Research project
%              (c) 2013 Nourdin Al-sherif. 2012
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
%
-----------------------------------------------------------------
--------
% All_scr_recod =[];
all_img_scr ={};
% subject image information.
[path,NAME,EXT]=fileparts(SubFile);
% read subject image;
sub_Img=imread(SubFile);
S_PreProcessFileName= [path '\' NAME '_PreprocessingInfo
\ValidatedPreProcResults.txt'];
S_PreProcInfo=TestgetPreProcInfo(S_PreProcessFileName);
no_tooth =0;
[S_FilmNum Dummy]= size(S_PreProcInfo.Cropping);
for S_Findex=1:S_FilmNum,
    % cropping the subject_image(Findex).
    film_x = S_PreProcInfo.Cropping(S_Findex,1);
    film_y = S_PreProcInfo.Cropping(S_Findex,2);
%   InFilm = sub_Img(film_x:film_x+S_PreProcInfo.Cropping
(S_Findex,3)-1,film_y:film_y+S_PreProcInfo.Cropping(S_Findex,4)-
1);
    if(~isempty(S_PreProcInfo.Segmentation{S_Findex}))
        % number of segmentations
        [S_SegmentNum Dummy]=size(S_PreProcInfo.Segmentation
{S_Findex});
        for Sindex=1:S_SegmentNum,
            seg_x = S_PreProcInfo.Segmentation{S_Findex}
(Sindex,1)+film_x-1;
            seg_y = S_PreProcInfo.Segmentation{S_Findex}
(Sindex,2)+film_y-1;
            length = S_PreProcInfo.Segmentation{S_Findex}
(Sindex,3);
```

```
            width = S_PreProcInfo.Segmentation{S_Findex}
(Sindex,4);
            S_SegLabel = S_PreProcInfo.Labels{S_Findex}
(Sindex,:);
            % Extract tooth segment.
            S_segment = sub_Img(seg_x:seg_x+length-
1,seg_y:seg_y+width-1);
            % check: is upper in jaw or in lower jaw. D: is
lower jaw and X:is upper jaw
            if S_SegLabel(2)=='D'
            else
               % flip the tooth segment
               S_segment = flipud(S_segment);
            end
            % normalizing & Resizing the segment.
            S_segment = viewNormalize(S_segment);
            Img_sub = imresize(S_segment,[32 32]);
            S_lab = S_SegLabel;
            no_tooth = no_tooth+1;
            S_segment_inf{no_tooth} = struct
('im_name',NAME,'r_im',{Img_sub},'lab',{S_lab});
         end
     end
end

for i=1:size(RefList,1),
    clear R_segment_inf;
    no_tooth =0;
    [path,NAME,EXT]=fileparts(RefList(i,:));
    ref_Img=imread(RefList(i,:));
    R_PreProcessFileName= [path '\' NAME '_PreprocessingInfo
\ValidatedPreProcResults.txt'];
    R_PreProcInfo=TestgetPreProcInfo(R_PreProcessFileName);
%     ref_Name = [];
    [R_FilmNum Dummy]= size(R_PreProcInfo.Cropping);
    R_lindex=[];
    for R_Findex=1:R_FilmNum,
         if(~isempty(R_PreProcInfo.Segmentation{R_Findex}))
              R_lindex=[R_lindex R_Findex];
         end
    end
    if(~isempty(R_lindex))
         R_lindex=R_lindex(1);
    end
    for R_Findex=1:R_FilmNum,
        film_x = R_PreProcInfo.Cropping(R_Findex,1);
        film_y = R_PreProcInfo.Cropping(R_Findex,2);
```

```
%       InFilm = ref_Img(film_x:film_x+R_PreProcInfo.Cropping
(R_Findex,3)-1,film_y:film_y+R_PreProcInfo.Cropping(R_Findex,4)-
1);
        if(~isempty(R_PreProcInfo.Segmentation{R_Findex}))
              [R_SegmentNum Dummy]=size
(R_PreProcInfo.Segmentation{R_Findex});
              for Sindex=1:R_SegmentNum,
                   seg_x = R_PreProcInfo.Segmentation{R_Findex}
(Sindex,1)+film_x-1;
                   seg_y = R_PreProcInfo.Segmentation{R_Findex}
(Sindex,2)+film_y-1;
                   length = R_PreProcInfo.Segmentation{R_Findex}
(Sindex,3);
                   width = R_PreProcInfo.Segmentation{R_Findex}
(Sindex,4);
                   R_SegLabel = R_PreProcInfo.Labels{R_Findex}
(Sindex,:);
                   % Extract tooth segment.
                   R_segment = ref_Img(seg_x:seg_x+length-
1,seg_y:seg_y+width-1);
                  % check: is upper in jaw or in lower jaw. D:
is lower jaw and X:is upper jaw
                   if R_SegLabel(2)=='D'
                   else
                       % flip the tooth segment
                       R_segment = flipud(R_segment);
                   end
                   % normalizing & Resizing the segment.
                   R_segment = viewNormalize(R_segment);
                   Img_ref=imresize(R_segment,[32 32]);
                   R_lab = R_SegLabel;
                   no_tooth = no_tooth+1;
                   R_segment_inf{no_tooth} = struct
('im_name',NAME,'r_im',{Img_ref},'lab',{R_lab});
              end
        end
    end
      [R_im_name,scr_recod] = dist_point
(S_segment_inf,R_segment_inf,i);
      if isnan(scr_recod)
          scr_recod =0;
      end

      img_scr = {R_im_name scr_recod};
      all_img_scr = [all_img_scr ;img_scr];
end
```

III. Appendix

```
function   [R_im_name,scr_recod] = dist_point
(S_segment_inf,R_segment_inf,ref_indx)
%
----------------------------------------------------------------
--------
% --------- Extracting the contours of subject and refernce
teeth images.
% input: subject and all reference tooth images;
% output: score of matching on record level;

% Programmer Name: Nourdin Al-sherif
%               West Virginia University
%               The ADIS Research project
%               (c) 2013 Nourdin Al-sherif. 2012
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
%
----------------------------------------------------------------
--------
 t_comparing =[];
 T_num = [];
 num =0;
 count_teeth_match =0;
for i=1:size(S_segment_inf,2)
   for j=1:size(R_segment_inf,2)
       if  strcmp(S_segment_inf{i}.lab, R_segment_inf{j}.lab)
           count_teeth_match = count_teeth_match+1;
           t_comparing = [ t_comparing ; S_segment_inf{i}.lab '
' R_segment_inf{j}.lab];

            num = count_sift1(S_segment_inf
{i}.r_im,R_segment_inf{j}.r_im);
            T_num = [T_num;num];

       end
   end
end
   R_im_name = R_segment_inf{1}.im_name;
   scr_recod = mean(T_num);
```

III. Appendix

```
function num = count_sift1(seg1,seg2)
%
----------------------------------------------------------------
--------
% --------- Extracting the contours of subject and refernce
teeth images.
% input: subject and reference tooth images;
% output: score of matching between subject and refernce tooth;

% Programmer Name: Nourdin Al-sherif
%               West Virginia University
%               The ADIS Research project
%               (c) 2013 Nourdin Al-sherif. 2012
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
%
----------------------------------------------------------------
--------
% tooth subject segment: PM.
synth1 = double(seg1);
% tooth refernce segment: AM.
synth2 = double(seg2);

% Extrract the contour of both subject(pm) and refernce(am)
tooth
% segements.
[C1]=TeethSegmentationcv_3(synth1);
[C2]= TeethSegmentationcv_3(synth2);

% coordites of contour of both subject(pm) and refernce(am)
r1 = C1(:,2);c1=C1(:,1);
r2 = C2(:,2);c2=C2(:,1);


% find sift of subject contour.
radius1 = 1;radius2 = 1;
I1 = synth1;
a1 =[1:1:size(r1,1)];
a1(:) = radius1;
rad1 = a1(:);
circles = [r1 c1 rad1 ];
enlarge_factor = 1.5;
sift_arr1 = find_sift(I1, circles, enlarge_factor);

% find sift of reference contour.
I2 = synth2;
```

```
a2 =[1:1:size(r2,1)];
a2(:) = radius2;
rad2 = a2(:);
circles = [r2 c2 rad2 ];
enlarge_factor = 1.5;
sift_arr2 = find_sift(I2, circles, enlarge_factor);

% matching their sift (subject & refernce contour)
[num, match1] =match(sift_arr1, sift_arr2,synth1,synth2,C1,C2);
```

```
function [B1]= TeethSegmentationcv_3(synth)
%
----------------------------------------------------------------
--------
% Extracting the contours of subject and refernce teeth images.
% Developed by: Diaa Eldin M. Nassar & Ayman Abaza
%               West Virginia University
%               The ADIS Research project
%               (c) 2005 Diaa Eldin M. Nassar  & Ayman Abaza -
Feb. 2005
%
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
% Please contact the authors for permission.
% input: image tooth;
% output: tooth contour (N*2);
%
----------------------------------------------------------------
--------
dt = 0.1; sigma = 3.75;
for z = 20:20
    % %Filtering the image with gaussian
    hsize = [3 3];
    h = fspecial('gaussian',hsize,sigma);
    synth = imfilter(synth,h);

    % forming the embedding function
    xc = round(size(synth,1)/2); yc = round(size(synth,2)/2); r
= round(size(synth,2)/2);
    Psi = zeros(size(synth)); Psi(yc,xc) = 1; [Psi,L] = bwdist
(Psi); Psi = Psi-r;
    for i = 1:1
        Psix = (Psi(3:end,2:end-1) - Psi(1:end-2,2:end-1))./2;
        Psiy = (Psi(2:end-1,3:end) - Psi(2:end-1,1:end-2))./2;
        Psixx = Psi(3:end,2:end-1) + Psi(1:end-2,2:end-1) - 2
*Psi(2:end-1,2:end-1);
        Psiyy = Psi(2:end-1,3:end) + Psi(2:end-1,1:end-2) - 2
*Psi(2:end-1,2:end-1);
        Psixy = (Psix(2:end-1,3:end) - Psix(2:end-1,1:end-
2))./2;
        Psixy = padarray(Psixy,[1 1],'replicate','both');

        PsiMod = sqrt(Psix.^2+Psiy.^2);
        PsiMod(find(PsiMod == 0)) = 0.001;
        kappa = -(Psixx.*(Psiy.^2) - 2.*Psix.*Psiy.*Psixy +
Psiyy.*(Psix.^2))./(PsiMod.^3);
```

```
        kappa = padarray(kappa,[1 1],'replicate','both');
        c1 = mean(synth(find(Psi<0))); c2 = mean(synth(find(Psi>
0)));
        ST = -(synth-c1).^2 + (synth-c2).^2;
        Psi = Psi - dt.*(2.*kappa+ST);
            %figure, imshow(synth,[]);hold on; contour(Psi, [0
0],'r');
    end

    BW = ~im2bw(Psi,0);
    BW1 = imfill(BW,'holes');
    B = bwboundaries(BW1);
    siz = [];
    for k = 1:size(B,1)
        siz = [siz;size(B{k},1)];
    end
    [j k] = max(siz);
    B1 = B{k};

end
```

```
function ROI = viewNormalize(ROI)

[h,w] = size(ROI);
H = fspecial('gaussian',[31 31], h/100);
CImg = conv2(double(ROI), H, 'same');

[Iy, Ix] = gradient(double(CImg));
Iphy = 180/pi*atan2(Iy, Ix);
Iphy(Iphy < 0) = Iphy(Iphy<0) + 180;
Iphy = floor(Iphy);
IG = edge(CImg, 'canny');

EROIQ = ones(h,w);
EROIQ(2:h-1,2:w-1) = 0;
EROIQ(IG>0 & EROIQ==1) = 0;

IGmsk = 0*IG;
IGmsk(floor(0.05*h):floor(0.95*h), floor(0.05*w):floor(0.95*w))
= 1;

IG = IG.*IGmsk;

mnIG = min(min(IG));
IG = IG - mnIG;
mxIG = max(max(IG));

HIphy = Iphy;
VIphy = Iphy;

VIphy(VIphy < 50 | VIphy > 130) = 0;
HIphy(HIphy > 40 & HIphy < 140) = 0;

conn_Img = GroupEdges  (IG, 8, 30, 8*(h+w));

Vconn_Img = conn_Img;

Vconn_Img (VIphy == 0) = 0;

VIphy(Vconn_Img == 0) = 0;

VOrnt = VIphy (VIphy > 0);
if(~isempty(VOrnt))
    VOrnt = VOrnt - 90;
    VOrntHst = hist(VOrnt, [-40:40])/length(VOrnt);

    mnVOrt = [-40:40]*VOrntHst';
```

```
        VRotImg = imrotate(ROI, -mnVOrt, 'bicubic');
        EROIQ = imrotate(EROIQ, -mnVOrt, 'bicubic');
        VIG = imrotate(IG, -mnVOrt, 'nearest');
        VIG(EROIQ > 0) = 0;
    else
        VRotImg = ROI;
        VIG = IG;
    end


    [Iy, Ix] = gradient(double(VRotImg));
    Iphy = 180/pi*atan2(Iy, Ix);
    Iphy(Iphy < 0) = Iphy(Iphy<0) + 180;
    Iphy = floor(Iphy);

    HIphy = Iphy;
    VIphy = Iphy;
    VIphy(VIphy < 50 | VIphy > 130) = 0;
    HIphy(HIphy > 40 & HIphy < 140) = 0;

    Iy(VIG < 1) = 0;
    Ix(VIG < 1) = 0;
    [x,y] = find(VIG);
    ndx = find(VIG);

    [h,w, z] = size(VRotImg);

    tstFlwH =  (-round(h/2) + repmat([h:-1:1]', [1 w]))/(h/2); %ones
    (h,w);
    tstFlwV =  (-round(w/2) + repmat([w:-1:1],  [h 1]))/(w/2); %
    tstFlwV;

    [conn_Img, RGB_conn, ord, Segments] = GroupEdges2  (VIG, 8, 5, 8
    *(h+w));
    IG2 = conn_Img;

    xflw = [];
    yflw = [];
    Ixflw = [];
    Iyflw = [];

    cntr_Img = VRotImg;
    flw_passed = [];
    for i = 1:size(ord,1)
        seg = Segments(i);
        seg = seg{1,1};
        ndxs = (seg(3,:))';
        uy = Iy(ndxs);
        ux = Ix(ndxs);
        ck_y = uy.*tstFlwV(ndxs);
        ck_x = ux.*tstFlwH(ndxs);
        ck_xy = ck_y + ck_x;

        ln_seg = length(ndxs);
        ln_ck = length(find(ck_xy >= 0));

        if (ln_ck >= 0.6*ln_seg)
            cntr_Img(ndxs) = 0;
            xflw = [xflw; (seg(1,:))'];
            yflw = [yflw; (seg(2,:))'];
            Ixflw = [Ixflw; ux];
            Iyflw = [Iyflw; uy];
            flw_passed = [flw_passed i];

        else
            IG2(IG2 == i) = 0;
        end

    end

    Rt_Hlf = IG2;
    Rt_Hlf(:,1:round(w/2)-1) = 0;
    ndx = find(Rt_Hlf);
    [xR,y] = find(Rt_Hlf);
    Rt_Hlf(ndx) = y;
    Rt_Hlf(Rt_Hlf==0) = w+1;
    y = hist(y, [1:w]);
    [dMx, mxY] = max(y);
    dMx = find(y >= 0.8*dMx);
    mxY = dMx(end);

    mnVec = min(Rt_Hlf');
    posmn = find(mnVec <= w);


    Lt_Hlf = IG2;
    Lt_Hlf(:,round(w/2):end) = 0;
    ndx = find(Lt_Hlf);
    [xL,y] = find(Lt_Hlf);
    Lt_Hlf(ndx) = y;
    y = hist(y, [1:w]);
    [dMn, mnY] = max(y);
    dMn = find(y >= 0.8*dMn);
```

```
mnY = dMn(1);

mnXY = min([xL; xR]);

mnY = max([1 mnY-round(0.1*w)]);
mxY = min([w mxY+round(0.1*w)]);

se = strel('diamond',1);
IG2 = imclose(IG2, se);
[conn_Img, RGB_conn, ord, Segments] = GroupEdges2  (IG2, 8, 5, 8
*(h+w));

HIphy(IG2 > 0 & HIphy == 0 & VIphy == 0) = 1;

H_ck = [];
for i = 1:size(ord,1)
    seg = Segments(i);
    seg = seg{1,1};
    ndxs = (seg(3,:))';

    horz_ck = length(find(HIphy(ndxs)));
    vert_ck = length(find(VIphy(ndxs)));
    if(vert_ck == 0)
        vert_ck  = 1;
    end
    H_ck = [H_ck (horz_ck/vert_ck)*length(ndxs)];
end
found_acc = 0;
while(~ found_acc & ~isempty(H_ck))
    [mx pos] = max(H_ck);
    if (mx == 0)
        break;
    else
        seg = Segments(pos);
        seg = seg{1,1};
        Xh = (seg(1,:))';
        Yh = (seg(2,:))';
        mnYh = min(Yh);
        mxYh = max(Yh);
        btwn = 0;
        if (~isempty(find(Yh >= mnY & Yh <= mxY)))
            btwn = 1;
        end

        if(mnYh < mnY & btwn)
            mnY = mnYh;
        end
        if(mxYh > mxY & btwn)
            mxY = mxYh;
        end
        mnX = min(Xh);
        if(abs(mnX-mnXY) <= 0.5*h)
            mxX = min([h mnX+mxY-mnY]);
            if(mxX-mnX >= 0.8*(mxY-mnY) & btwn)
                found_acc = 1;
            else
                H_ck(pos) = [];
                Segments(pos) = [];
            end
        else
            H_ck(pos) = [];
            Segments(pos) = [];
        end
    end
end

if(found_acc == 0)
    mnX = mnXY;
    mxX = min([h mnX+mxY-mnY]);
end

ROI = VRotImg(mnX:mxX, mnY:mxY);
```

```
%
================================================================
============================
% Image Cropping to Films
% Developed by: Diaa Eldin Nassar , Ayman Abaza
% West Virginia University
% EE 591B: Advanced Digital Image Processing - Final Project
% (c)2004 - Diaa Eldin Nassar , Ayman Abaza Dec. 2004
%
================================================================
=========
function [CroppingInfo, CroppedRecord] = CropRecord_WVU
(ImgRecord)
CroppingInfo=[];
unit_area=10000;
[c,r]=dental_classify(ImgRecord);
if (r==1)
    y=dental_round_cut((c/255));
else
    y=dental_right_cut((c/255));
end
c=bwlabel(y);
for index=1:max(max(c))
    b=(c==index);
    b1=dental_fill(b);
    c(find(b1==1))=index;
end

cmax=max(max(c));
seg_img = c;
seg_img(1,:) = 0;
seg_img(end,:) = 0;
seg_img(:,1) = 0;
seg_img(:,end) = 0;
for index=1:cmax
    y=(c==index);
    if sum(sum(y))<unit_area
        c(find(y==1))=0;
    else
        [R, C] = find (seg_img == index);
        mxX = max(R);
        mnX = min(R);
        mxY = max(C);
        mnY = min(C);
        CroppingInfo = [CroppingInfo; mnX mnY (mxX-mnX+1) (mxY-
mnY+1)];
    end
```

```
end
% c=label2rgb(c); imshow(c)
CroppedRecord = double(ImgRecord);
CroppedRecord = colorTeeth(CroppedRecord, CroppingInfo);
```

```
% Automatic Dental Image Segmentation
% Author: Xin Li
% Date: Oct. 15, 2004
% Input: x - dental image to be segmented
% Output: c - label image after segmentation

function [c,r]=dental_classify(x)
x=double(x);
[M,N]=size(x);
% Preprocessing stage
% Step 1: identify the background color
b=background_extract(x);
% Step 2: noise removal by two-stage morphological filtering
c=bwlabel(b);
T=3;
for index=1:max(max(c))
    [i,j]=find(c==index);
    if min(i)>T+1&max(i)<M-T&min(j)>T+1&max(j)<N-T
        b(find(c==index))=0;
    end
end
b1=bwmorph(1-b,'erode',3);
b2=bwareaopen(b1,1000);
b3=bwmorph(b2,'dilate',3);
c=bwlabel(b3);
% step 3: round-corner or right-corner classification
%     by=bwmorph(b,'remove');
%     s1=sum(sum(by));
%     s2=sum(sum(runlen_count(by,2)));
b4=corner_detect(b3);
b5=bwmorph(b4,'dilate',1);
b6=bwareaopen(b5,32);
ratio=sum(sum(b6))/sum(sum(b4));
    if ratio<3
%       disp('right corner');
        r=0;
    else
%       disp('round corner');
        r=1;
    end
```

```
function b=background_extract(x)
[M,N]=size(x);
h=imhist(x/255);
p(1)=find(h==max(h))-1;
h(p(1)+1)=0;
p(2)=find(h==max(h))-1;
h(p(2)+1)=0;
p(3)=find(h==max(h))-1;
L=10;
b=zeros(M,N);
for i=1:3
    layer=(x==p(i));
    bd=bwmorph(layer,'remove');
    s1=sum(sum(bd));
    s2=sum(sum(runlen_count(bd,L)));
    r(i)=s1-s2;
end
j=find(r==max(r));
%j=1;
if p(j)<=10
    I=find(x<=p(j));
elseif p(j)>=250
    I=find(x>=250);
else %modified by DMNassar on 09/20/2005
    I = [];
    for k = 1:length(j)
        I=[I; find(x<=p(j(k))+1 & x>=p(j(k))-1)];
    end
end
b(I)=1;
b=bwmorph(b,'close');
```

```
% Hit-or-Miss filtering for corner detection

function c=corner_detect(y)

% set up template
b1=[1 0 0;1 1 0;1 1 1];b2=[0 1 1;0 0 1;0 0 0];
c1=bwhitmiss(y,b1,b2);
c2=fliplr(bwhitmiss(fliplr(y),b1,b2));
c3=flipud(bwhitmiss(flipud(y),b1,b2));
c4=flipud(fliplr(bwhitmiss(flipud(fliplr(y)),b1,b2)));
cx=c1|c2|c3|c4;

b1=[0 0 0;1 1 0;1 1 0];b2=[1 1 1;0 0 1;0 0 1];
c1=bwhitmiss(y,b1,b2);
c2=fliplr(bwhitmiss(fliplr(y),b1,b2));
c3=flipud(bwhitmiss(flipud(y),b1,b2));
c4=flipud(fliplr(bwhitmiss(flipud(fliplr(y)),b1,b2)));
cy=c1|c2|c3|c4;

c=cx|cy;
```

```
function cy=Lcorner_detect(y,K)

% set up template
[M,N]=size(y);
b1=zeros(2*K+1);b1(K+1:2*K+1,1:K)=1;
b2=ones(2*K+1);b2(K+1:2*K+1,1:K)=0;
c1=bwhitmiss(y,b1,b2);
c2=fliplr(bwhitmiss(fliplr(y),b1,b2));
c3=flipud(bwhitmiss(flipud(y),b1,b2));
c4=flipud(fliplr(bwhitmiss(flipud(fliplr(y)),b1,b2)));
c=c1|c2|c3|c4;
cy=zeros(M,N);cy(2:M-1,2:N-1)=c(2:M-1,2:N-1);

function y=runlen_count(x,L)

[M,N]=size(x);
y=x;
for i=1:M
    for j=1:N
        if y(i,j)==1
            run=0;
            while j+run<=N&y(i,j+run)==1
                run=run+1;
            end
            if run>=L
                y(i,j:j+run-1)=0;
            end
        end
    end
end

for j=1:N
for i=1:M
        if y(i,j)==1
            run=0;
            while i+run<=M&y(i+run,j)==1
                run=run+1;
            end
            if run>=L
                y(i:i+run-1,j)=0;
            end
        end
    end
end

% [RES] = shift(MTX, OFFSET)
%
```

```
% Circular shift 2D matrix samples by OFFSET (a [Y,X] 2-vector),
% such that  RES(POS) = MTX(POS-OFFSET).

function res = shift(mtx, offset)

dims = size(mtx);

offset = mod(-offset,dims);

res = [ mtx(offset(1)+1:dims(1), offset(2)+1:dims(2)),   ...
          mtx(offset(1)+1:dims(1), 1:offset(2));         ...
        mtx(1:offset(1), offset(2)+1:dims(2)),           ...
      mtx(1:offset(1), 1:offset(2)) ];

% Hit-or-Miss filtering for corner detection

function c=Ucorner_detect(y)

[M,N]=size(y);
cx=[-1 -1 -1 0 1 1 1 0];
cy=[-1 0 1 1 1 0 -1 -1];
sy=zeros(M,N);
sy=sy+((shift(y,[cx(8),cy(8)])==1)&(shift(y,[cx(1),cy(1)])==1)&
(shift(y,[cx(2),cy(2)])==1));
sy=sy+((shift(y,[cx(8),cy(8)])==1)&(shift(y,[cx(1),cy(1)])==1)&
(shift(y,[cx(7),cy(7)])==1));
for i=2:7
    sy=sy+((shift(y,[cx(i-1),cy(i-1)])==1)&...
        (shift(y,[cx(i),cy(i)])==1)&(shift(y,[cx(i+1),cy(i+1)])
==1));
end
c=(y==0)&(sy>3);
```

```
% Automatic Dental Image Segmentation
% Author: Xin Li
% Date: Jan. 2005

function y=dental_round_cut(x)

[M,N]=size(x);
b=bwlabel(x);
y=zeros(M,N);
for index=1:max(max(b))
    x1=(b==index);
    c=Ucorner_detect(x1);
    y1=x1;
    T=2;
    [i,j]=find(c==1);
    for k=1:length(i)
        if i(k)>1&i(k)<M-1&j(k)>1&j(k)<N-1
            if y1(i(k),j(k))==0
                s1=sum(y1(max([1 i(k)-T]):i(k)+T,j(k))); %
modified by DM Nassar on 09/20/2005
                s2=sum(y1(i(k),max([1 j(k)-T]):j(k)+T)); %
modified by DM Nassar on 09/20/2005
                if s1>s2
                    % horizontal cut
                    if sum(sum(c(max([1 i(k)-T]):i(k)+T,:)))>1 %
modified by DM Nassar on 09/20/2005
                        [ii,jj]=find(c(max([1 i(k)-T]):i(k)+T,:)
==1); %modified by DM Nassar on 09/20/2005
                        if max(jj)-min(jj)>10
                            y1(i(k)+round(mean(ii)),:)=0;
                        end
                    end
                else
                    % vertical cut
                    if sum(sum(c(:,max([1 j(k)-T]):j(k)+T)))>1 %
modified by DM Nassar on 09/20/2005
                        [ii,jj]=find(c(:,max([1 j(k)-T]):j(k)+T)
==1); %modified by DM Nassar on 09/20/2005
                        if max(ii)-min(ii)>10
                            y1(:,j(k)+round(mean(jj)))=0;
                        end
                    end
                end
            end
        end
    end
    y=y+y1;
```

```
   end
```

```
% Automatic Dental Image Segmentation
% Author: Xin Li
% Date: Jan. 2005

function y=dental_right_cut(x)
[M,N]=size(x);
b=bwlabel(x);
y=zeros(M,N);
for index=1:max(max(b))
    x1=(b==index);
c=Lcorner_detect(x1,3);
y1=x1;
[i,j]=find(c==1);
for k=1:length(i)
    if i(k)>1&i(k)<M-1&j(k)>1&j(k)<N-1
    if sum(sum(c(i(k)-1:i(k)+1,:)))>1
        [ii,jj]=find(c(i(k)-1:i(k)+1,:)==1);
        if max(jj)-min(jj)>10
            y1(i(k)-1:i(k)+1,min(jj):max(jj))=0;
        end
    end
    if sum(sum(c(:,j(k)-1:j(k)+1)))>1
        [ii,jj]=find(c(:,j(k)-1:j(k)+1)==1);
        if max(ii)-min(ii)>10
            y1(min(ii):max(ii),j(k)-1:j(k)+1)=0;
        end
    end
end
end
y=y+y1;
end
```

```
% [RES] = shift(MTX, OFFSET)
%
% Circular shift 2D matrix samples by OFFSET (a [Y,X] 2-vector),
% such that  RES(POS) = MTX(POS-OFFSET).

function res = shift(mtx, offset)

dims = size(mtx);

offset = mod(-offset,dims);

res = [ mtx(offset(1)+1:dims(1), offset(2)+1:dims(2)),  ...
          mtx(offset(1)+1:dims(1), 1:offset(2));        ...
        mtx(1:offset(1), offset(2)+1:dims(2)),          ...
      mtx(1:offset(1), 1:offset(2)) ];



% Automatic Dental Image Segmentation
% Author: Xin Li
% Date: Jan. 2005

function y=dental_fill(x)

y=zeros(size(x));
[i,j]=find(x==1);
y(min(i):max(i),min(j):max(j))=1;

function y=runlen_count(x,L)

[M,N]=size(x);
y=x;
for i=1:M
    for j=1:N
        if y(i,j)==1
            run=0;
            while j+run<=N&y(i,j+run)==1
                run=run+1;
            end
            if run>=L
                y(i,j:j+run-1)=0;
            end
        end
    end
end

for j=1:N
for i=1:M
        if y(i,j)==1
            run=0;
            while i+run<=M&y(i+run,j)==1
                run=run+1;
            end
            if run>=L
                y(i:i+run-1,j)=0;
            end
        end
    end
end
```

```
function segmentedFilm = colorTeeth(Film, segments)

N = size(segments,1);

if (N > 0)
    thR = 0:2*pi/N:2*pi;
else
    thR = 0;
end
thG = 2*pi/3 + thR;
thB = 2*pi/3 + thG;
% else
%     thR = pi/2;
%     thG = 2*pi/3 + thR;
%     thB = 2*pi/3 + thG;
% end

ord = randperm(N);

if(size(Film,3) == 3)
    ColorFilm = double(Film);
    mx = max(max(Film));
    Film = double(rgb2gray(Film))/mx;
else
    mx = max(max(Film));
    Film = double(Film)/mx;
    ColorFilm = repmat(Film,[1 1 3]);
end
ColorFilm(:,:,:) = 0;

[H, W] = size(Film);

for i = 1:N

    xmn = segments(i,1);
    ymn = segments(i,2);
    xmx = xmn + segments(i,3);
    ymx = ymn + segments(i,4);

    xmn = max([1, xmn]);
    ymn = max([1, ymn]);

    xmx = min([H, xmx]);
    ymx = min([W, ymx]);

    ColorFilm(xmn:xmx, ymn:ymx, 1) = Film(xmn:xmx, ymn:ymx) *
(0.5 + 0.5 * sin(thR(ord(i))));
```

```
    ColorFilm(xmn:xmx, ymn:ymx, 2) = Film(xmn:xmx, ymn:ymx) *
(0.5 + 0.5 * sin(thG(ord(i))));
    ColorFilm(xmn:xmx, ymn:ymx, 3) = Film(xmn:xmx, ymn:ymx) *
(0.5 + 0.5 * sin(thB(ord(i))));

    %     ColorFilm(xmn:xmx, ymn:ymx, 1) = ColorFilm(xmn:xmx,
ymn:ymx, 1) + Film(xmn:xmx, ymn:ymx) * (0.5 + 0.5 * sin(thR(ord
(i))));
    %     ColorFilm(xmn:xmx, ymn:ymx, 2) = ColorFilm(xmn:xmx,
ymn:ymx, 2) + Film(xmn:xmx, ymn:ymx) * (0.5 + 0.5 * sin(thG(ord
(i))));
    %     ColorFilm(xmn:xmx, ymn:ymx, 3) = ColorFilm(xmn:xmx,
ymn:ymx, 3) + Film(xmn:xmx, ymn:ymx) * (0.5 + 0.5 * sin(thB(ord
(i))));

end

mx = max(max(max(ColorFilm)));
if(mx > 0)
    segmentedFilm = ColorFilm/mx;
else
    segmentedFilm = ColorFilm;
end
```

```
function [SegmentationInfo, SegmentedFilm] =
MatADIS_TeethSegment(CroppingInfo,CroppedRecord);
%
-----------------------------------------------------------------
--------
% --------- Teeth Segmentation
-----------------------------------  .
% input: dental record & CroppingInfo
% Output Details:
%                - SegmentationInfo: An Array of N*4 integers
where N is the
%                number of segmented teeth, each row looks like
[xo yo h w],
%                where xo,yo are the row and column indices
(respectively)
%                of the upper left corner of the bounding
rectangle of a tooth
%                and h,w are the height and width (respectively)
of that rectangle.
%                - SegmentedFilm: an image showing bounding
rectangles of teeth
%                overlaid on the input film.
% Programmer Name: Nourdin Al-sherif
%                West Virginia University
%                The ADIS Research project
%                (c) 2013 Nourdin Al-sherif. 2012
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
%
-----------------------------------------------------------------
--------


   % declaration varibles
    global im_t1 im_t2 im_t3 im_t4;
    global  im_name

  % Divide the dental film into parts (upper and lower jaw)
    set(handles.pushbutton1, 'Enable', 'off')
    set(handles.pushbutton2, 'Enable', 'off')
    set(handles.figure1,'Pcinter','watch');
    refresh(main_8) %redraws the GUI to reflect changes
    for no_film = 1 : size(CroppingInfo,2)
        handles.Im_org = CroppedRecord(no_film);
    end
   i2 =handles.Im_org;
```

```
    if size(i2,3)==3
       i2 = rgb2gray(i2);
    end
    [Adaptive] = ia_thresholding(i2);
    Adaptive = double(Adaptive);
    Y=permute(Adaptive,[2,1,3]);
    [handles.HL,handles.HR] = Seam_Gard(Y,handles.N);
    handles.HR = flipud(handles.HR);
    handles.H = handles.HR;
    % select the best candidate seam.
    handles.hor_vect = sel_hor_1(handles.H,handles.Im_org);
    % ...............  second horizontal
.................................
    Y1 =  fliplr(Y);
   [handles.HL1,handles.HR1] = Seam_Gard(Y1,handles.N);
    handles.HR1 = flipud(handles.HR1);
    handles.H1 = handles.HR1;
    % select the best candidate seam.
    handles.hor_vect1 = sel_hor_1(handles.H1,handles.Im_org);
    %
.................................................................
....
     % Draw the horizontal seam
     hold on
%      cc = 'r';
%      plot(handles.hor_vect,'.','LineWidth',1,...
%                          'MarkerEdgeColor',cc,...
%                          'MarkerFaceColor','r',...
%                          'MarkerSize',3)
%
%      cc = 'g';
%      plot(handles.N-handles.hor_vect1,'.','LineWidth',1,...
%                          'MarkerEdgeColor',cc,...
%                          'MarkerFaceColor','g',...
%                          'MarkerSize',3)

     cc = 'y';
     ave = round((handles.hor_vect + handles.N-
handles.hor_vect1)/2);
     plot(ave,'.','LineWidth',1,...
                       'MarkerEdgeColor',cc,...
                       'MarkerFaceColor','y',...
                       'MarkerSize',3)

     hold off

    [handles.upper_jaw,handles.lower_jaw]=Sepration_jaws
```

```
(ave,handles.Im_org);

  %[handles.upper_jaw,handles.lower_jaw]=Sepration_jaws
(handles.hor_vect,handles.Im_org);

  % detect mid segment
   D = 40;
  % ***************** Upper_Jaw ***************************
  get_on = 1;
  Y = handles.upper_jaw;
  height = size(Y,1);
  % compute vertical seam [HL and HR].
  [handles.HL,handles.HR] = Seam_Gard(Y,handles.M);
  handles.HR = flipud(handles.HR);
  axes(handles.axes2)
  cla;
  imshow(handles.upper_jaw/255);
      i = 1:size(handles.HR,1);
      [all_st_point,all_JU]   = capeture_1(handles.HL);
      cc = 'r';
      hold on
      s1=ceil(size(handles.HL,2))/2-D;
      s2=ceil(size(handles.HL,2))/2+D;
      for k = 1: size(all_JU,1)
           x_vect=handles.HL(:,all_JU(k));
           plot(handles.HL(:,all_JU(k)),i','.','LineWidth',1,...
                   'MarkerEdgeColor',cc,...
                   'MarkerFaceColor',cc,...
                   'MarkerSize',3)
                       cc = 'r';
      end
      hold off

%------------------------ croping upper
jaw--------------------------------
 for j = 1:size(Y,1)
   v =sum(Y(end-j,:));
    if v >= 8500
        break
    end
end
%j
cY = Y(1:end-j,:);
H = size(cY,1);
Wi = handles.M;
% figure(2)
% imshow(uint8(cY));
% hold on
all_cor_lowI =[];
for k = 1: size(all_JU,1)
    x_vect=handles.HL(:,all_JU(k));
    %set_po = [X2 Y2  X1 Y1 ];
    set_po = [x_vect(H) H  x_vect(1) 1 ];
    cor_lowI = [ 1   x_vect(1)   H   x_vect(H)];
    all_cor_lowI = [all_cor_lowI;cor_lowI];
   % plot([set_po(1),set_po(3)],[set_po(2),set_po
(4)],'Color','r','LineWidth',2)
end
%ordering
all_cor_lowI1 = sortrows(all_cor_lowI,2);
 M1 = all_cor_lowI1;
% % Change them to straight line wrt upper points
 M1(:,4)= all_cor_lowI1(:,2);
% % coordinates
 M2 =[];
 Y = 1; L= size(cY,1);
 X = 1; W = M1(1,2);
M2 = [Y X W L];
no_s = size(all_JU,1);
for id = 2: no_s
    X = M1(id-1,2);
    W = M1(id,2)- M1(id-1,2);
    M2 = [M2; Y X W L];
end
 X = M1(id,2);
W = handles.M- M1(id,2);
M2 = [M2; Y X W L];
% % Delete out
threshold = 50;
if M2(1,3)<threshold
     M2(1, :) = [];
end
if M2(end,3)<threshold
     M2(end, :) = [];
end
% % convert to image coordinates
Yn = L;
 M3 = M2; M3(:,1) = 1;
 M3U = M3;

% ****************** Lower_Jaw
********************************************
   get_on = 1;
   Y = handles.lower_jaw;
```

```
    % compute vertical seam [HL and HR].
    [handles.HL,handles.HR] = Seam_Gard(Y,handles.M);
    handles.HR = flipud(handles.HR);

   axes(handles.axes7)
   cla;
   imshow(handles.lower_jaw/255);
   i = 1:size(handles.HL,1);
   [all_st_point,all_JL]  = capeture_2(handles.HR);
   cc = 'r';
   hold on
   % detect mid segment

   s1=ceil(size(handles.HL,2))/2-D;
   s2=ceil(size(handles.HL,2))/2+D;
     for k = 1: size(all_JL,1)
           x_vect=handles.HL(:,all_JL(k));


            plot(handles.HR(:,all_JL(k)),i','.','LineWidth',1,...
                'MarkerEdgeColor',cc,...
                'MarkerFaceColor',cc,...
                'MarkerSize',3)

            cc = 'r';
    end
   toc
  hold off
%------------------------ croping lower
jaw-------------------------------
 for j = 1:size(Y,1)
   v =sum(Y(j+1,:));
    if v >= 8500
        break
    end
end
%j
cY = Y(j:end,:);
H = size(cY,1);
Wi = handles.M;
% figure(2)
% imshow(uint8(cY));
% hold on
all_cor_lowI =[];
for k = 1: size(all_JL,1)
    x_vect=handles.HR(:,all_JL(k));
    set_po = [x_vect(end) H   x_vect(j) 1]
    cor_lowI = [ 1 x_vect(j) H x_vect(end)];
    all_cor_lowI = [all_cor_lowI;cor_lowI];
    %plot([set_po(1),set_po(3)],[set_po(2),set_po
(4)],'Color','r','LineWidth',2)
end
%all_cor_lowI
% ordering
all_cor_lowI1 = sortrows(all_cor_lowI,2);
M1 = all_cor_lowI1;
% Change them to straight line wrt upper points
M1(:,4)= all_cor_lowI1(:,2);
% coordinates
M2 =[];
Y = 1; L= size(cY,1);
X = 1; W = M1(1,2);
M2 = [Y X W L];
no_s = size(all_JL,1);
for id = 2: no_s
    X = M1(id-1,2);
    W = M1(id,2)- M1(id-1,2);
    M2 = [M2; Y X W L];
end
X = M1(id,2);
W = handles.M- M1(id,2);
M2 = [M2; Y X W L];
% Delete out
threshold = 50;
if M2(1,3)<threshold
     M2(1, :) = [];
end
if M2(end,3)<threshold
     M2(end, :) = [];
end
% convert to image coordinates
Yn = handles.N - L;
M3 = M2; M3(:,1) = Yn;
M3L = M3;
% -------------  disply segmented teeth
-----------------------------------
axes(handles.axes11)
cla;
imshow(handles.Im_org);

for i = 1: size(M3U,1)
    %rectangle('Position',Y X   W L], 'LineWidth',2,
'EdgeColor','b');
    rectangle('Position',[M3U(i,2) M3U(i,1)   M3U(i,3) M3U(i,4)
], 'LineWidth',2, 'EdgeColor','r');
end
for i = 1: size(M3L,1)
    %rectangle('Position',Y X   W L], 'LineWidth',2,
'EdgeColor','b');
    rectangle('Position',[M3L(i,2) M3L(i,1)   M3L(i,3) M3L(i,4)
], 'LineWidth',2, 'EdgeColor','r');
end
  M = [M3U' M3L'];
  [feature, OutClass] = Filmclassify_UM(handles.Im_org);
%  dlmwrite('ValidatedPreProcResults.txt', M, ' ');
str1 = '[Teeth Segmentation]';
str2 = '>>Film #1: Bite-wing';
fid = fopen('ValidatedPreProcResults_2.txt','wt');
fprintf(fid, '%s\n', str1);
fprintf(fid, '%s\n', str2);
fclose(fid)
M = [M3U' M3L'];
% ----------------------- Labeling
-----------------------------------
A_class = [];
for i = 1: size(M,2)
    x = M(1,i);
    y = M(2,i);
    w = M(3,i);
    l = M(4,i);
    %imshow(im(x:x+l,y:y+w));
    S_segment=handles.Im_org(x:x+l,y:y+w);
    % flip the tooth segment
    if x == 1
       S_segment = flipud(S_segment);
    end
    S_segment = viewNormalize(S_segment);
    Img_sub = imresize(S_segment,[32 32]);
    class = labeling(Img_sub);
    A_class = [A_class class];
end

LABELING = A_class;
LABELING
```

```
function [Adaptive] = ia_thresholding(dental)
%
----------------------------------------------------------------
--------
% ----------------------- Preprocessing Operations
-----------------------
% apply two step thresholding techniques (iterative and
adaptive) to get
% binarized image
% segment rgb image into teeth and background region.
% input: dental film
% output:
% Programmer Name: Nourdin Al-sherif
%               West Virginia University
%               The ADIS Research project
%               (c) 2013 Nourdin Al-sherif. 2012
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
%
----------------------------------------------------------------
--------
[m n]=size(dental);
%Canny edge detection
%dental = rgb2gray(dental);
[canny_dental,thres]=edge(dental,'canny');

%Morphlogical dilation
se90 = strel('line', 3, 90);
se0 = strel('line', 3, 0);

dental_dil = imdilate(canny_dental, [se90 se0]);
%Get the image just for pixels of dilation edge
forthres_dental = immultiply(dental,dental_dil);

%Iterative threshold

%Get initial T1
sumobjectpixel=0;objectnumber=0;
sumbackgroundpixel=0;backgroundnumber=0;


objectnumber = sum(sum((forthres_dental>=100)));
backgroundnumber = sum(sum((forthres_dental<100)));
sumobjectpixel = double(sum(sum(immultiply((forthres_dental>=
100),forthres_dental))));
sumbackgroundpixel = double(sum(sum(immultiply((forthres_dental
<100),forthres_dental))));

uo0=sumobjectpixel/objectnumber;
ub0=sumbackgroundpixel/backgroundnumber;
T1=(ub0+uo0)/2;

%Get the second T2
sumobjectpixel=0;objectnumber=0;
sumbackgroundpixel=0;backgroundnumber=0;


objectnumber = sum(sum((forthres_dental>=T1)));
backgroundnumber = sum(sum((forthres_dental<T1)));
sumobjectpixel = double(sum(sum(immultiply((forthres_dental>
=T1),forthres_dental))));
sumbackgroundpixel = double(sum(sum(immultiply((forthres_dental
<T1),forthres_dental))));

uo1=sumobjectpixel/objectnumber;
ub1=sumbackgroundpixel/backgroundnumber;
T2=(uo1+ub1)/2;


%Iterative get the last Threshold
while abs(T2-T1)> 1
    T1=T2;
    objectnumber = sum(sum((forthres_dental>=T1)));
    backgroundnumber = sum(sum((forthres_dental<T1)));
    sumobjectpixel = double(sum(sum(immultiply((forthres_dental>
=T1),forthres_dental))));
    sumbackgroundpixel = double(sum(sum(immultiply
((forthres_dental<T1),forthres_dental))));

    uo1=sumobjectpixel/objectnumber;
    ub1=sumbackgroundpixel/backgroundnumber;
    T2=(uo1+ub1)/2;
    sumobjectpixel=0;objectnumber=0;
    sumbackgroundpixel=0;backgroundnumber=0;
end;

Threshold=num2str(T2,5);

%display two results
%result from edge

result_image = (forthres_dental>=T2);
```

```
%result from original

result_image1 = (dental>=T2);


Im = immultiply(result_image1,dental);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%
% adaptive method
km = 49; kn = 79;
kkm = 25; kkn = 40;
Mask = ones(km,kn);
Thresum = conv2(double(Im),Mask);
Threnum = conv2(double(result_image1),Mask);
Threshold1 = (Thresum./(Threnum+1));
Threshold = Threshold1(kkm:m+kkm-1,kkn:n+kkn-1);

gk = 0;
g2 = 0.85;
Tg = uint8(gk*ones(m,n));
Threshold = uint8(Threshold*g2);

Threshold = immultiply(result_image1,Threshold);

Adaptive = ( imsubtract(imadd(Im,Tg), Threshold)>0 );

bw = bwareaopen(Adaptive,200);
se = strel('rectangle',[10,5]);


se1 = strel('rectangle',[20,10]);
bw2 = imclose(bw,se);

bw3 = imopen(bw2,se);

Adaptive1 = ( imsubtract(imadd(Im,Tg), Threshold)>0 );
im = Adaptive1;

% adaptive method
km = 49; kn = 79;
kkm = 25; kkn = 40;
Mask = ones(km,kn)/(km*kn);
%Threshold = uint8(conv2(Im,Mask));

Thresum = conv2(double(Im),Mask);
```

```
Threshold = (Thresum(kkm:m+kkm-1,kkn:n+kkn-1));

Tg = uint8(gk*ones(m,n)) ;
Threshold = uint8(Threshold*g2);
Threshold = immultiply(result_image1,Threshold);


Adaptive = (imsubtract(imadd(Im,Tg), Threshold)>0 );
```

```
function [Vec_seam_L,Vec_seam_R,GL,GR]= Seam_Gard(im,tot_seams)
% compute the gradient of the given image
% and also compute the vertical seams.
% input: (im) dental film
% (tot_seams)number of horizontal seams.
% output: cordinates of horizonatl seams.
% % Programmer Name: Nourdin Al-sherif
%                West Virginia University
%                The ADIS Research project
%                (c) 2013 Nourdin Al-sherif. 2012
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.


global G_type;
[rows cols dim]=size(im);


if size(im,3) == 3
    G = rgb2gray(im);
else
    G = im;
end


G1 = G;
% take the minimum value.
tot_seam = min(tot_seams,cols-1);

tot_seam =1;

  GL = G;
  GR= flipud(G);


 tot_s_L = All_Seams(GL);
 tot_s_R = All_Seams(GR);

 %find Vec_seam.
 Vec_seam_L=Get_Seam_2(tot_s_L);
 Vec_seam_R=Get_Seam_2(tot_s_R);
```

```
function all_seam=All_Seams(g)
%
----------------------------------------------------------------
--------
% ----------- compute the seams picture of the given gardient
-----------
% input: gradient dental film
% output: coordinate seams of gradient dental film
% Programmer Name: Nourdin Al-sherif
%               West Virginia University
%               The ADIS Research project
%               (c) 2013 Nourdin Al-sherif. 2012
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
%
----------------------------------------------------------------
--------
[n m]=size(g);
all_seam=zeros(n,m);
% Replace the first line in the image with same gradient values.
all_seam(1,:)=g(1,:);
L = 2;
while L <= n
    K =1;
    while K <= m
        switch K
          case 1
              % first pixel.
            all_seam(L,K)= g(L,K)+min([all_seam(L-1,K),all_seam
(L-1,K+1)]);
          case m
              % last pixel.
            all_seam(L,K)= g(L,K)+min([all_seam(L-1,K-
1),all_seam(L-1,K)]);
            otherwise
              %pixels between
            all_seam(L,K)= g(L,K)+min([all_seam(L-1,K-
1),all_seam(L-1,K),all_seam(L-1,K+1)]);
       end
        K = K+1;
    end
    L = L+1;
end
```

```
function S_vec=Get_Seam_2(tot_s)
%
----------------------------------------------------------------
--------
% ----------- Compute the dimension of the image seams
-----------
% input: total seams
% output: get appropraite  seams
% Programmer Name: Nourdin Al-sherif
%               West Virginia University
%               The ADIS Research project
%               (c) 2013 Nourdin Al-sherif. 2012
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
%
----------------------------------------------------------------
--------


% Compute the dimension of the image seams
[rws cls]=size(tot_s);
% start from the bottom.
for j = 1:cls
        K = rws;
        while K >= 1
            if K==rws
                % the value of the last pixel.
                %[S_val, L]=min(tot_s(rws,:));
                [value ins]= sort(tot_s(rws,:));
                S_val = value(j);
                L = ins(j);
                S_vec(K,j)=ins(j);
            else
                % Three cases
                switch L
                     case 1
                        vec=[Inf tot_s(K,L) tot_s(K,L+1)];
                     case cls
                        vec=[tot_s(K,L-1) tot_s(K,L) Inf];
                    otherwise
                        vec=[tot_s(K,L-1) tot_s(K,L) tot_s(K,L+
1)];
                end
                %locate the smallest value and the index othe
three
                % neighboring pixels in row-1.
                [S_val idx]=min(vec);
                if idx == 1
                    L_idx = L-1;

                end
                if idx == 2
                    L_idx = L;
                end
                if idx == 3;
                    L_idx = L+1;
                end

                S_vec(K,j)=L_idx;
                L = L_idx;
            end

           K = K -1;
        end

end
```

```
function vect = sel_hor_1(hors_vect,image)
%
-----------------------------------------------------------------
--------
% ----------------------- select the optimal seam
-----------------------
% input: binarized inmage
% output: optimal seam that separated the upper and lower jaw
% Programmer Name: Nourdin Al-sherif
%               West Virginia University
%               The ADIS Research project
%               (c) 2013 Nourdin Al-sherif. 2012
% No part of this code may be copied, replicated or used in any
form without the explicit
% consent of the authors.
%
-----------------------------------------------------------------
--------
all_vect = [];
delta = 20;
[m,n]=size(image);
len_vect = size(hors_vect,2);
end_ind = size(hors_vect,1);

int_val1 = ceil(m/2-delta);
int_val2 = ceil(m/2+delta);


all_j = [];
all_sum_int =[];
for j = 1: len_vect
  vect = hors_vect(:,j);
  s_x1 = vect(1,1);
  e_x1 = vect(end_ind,1);

  if s_x1 >= int_val1 &&  e_x1 <= int_val2
      vect = hors_vect(:,j);

     sum_int =0;
     for m = 1:size(vect,1)
          sum_int = double(image(vect(m,1),m))+ sum_int ;
     end
      all_sum_int = [all_sum_int,sum_int];
      all_j = [all_j,j];
  end
end
[x,y]=min(all_sum_int);
```

```
   vect = hors_vect(:,all_j(y));
```

```
function [LabelInfo,  LabeledImg] = MatADIS_TeethLabel
(SegmentedFilm, SegmentationInfo)
%
=================================================================
============================
% Programmer Name: Nourdin Al-sherif
%                West Virginia University
%                The ADIS Research project
%                (c) 2013 Nourdin Al-sherif. 2012
%[LabelInfo,  LabeledImg] = MatADIS_TeethLabel(SegmentedFilm,
SegmentationInfo);
%
=================================================================
============================
load([pat '\parameters\TeethOLPP.mat'],
'irow','icol','ustd','um','m','u','omega');
for no_seg = 1:size(SegmentationInfo,2)
    pat = pwd;
    all_r =[];
    in_segment = SegmentationInfo(no_seg);
            InputImage = in_segment;
            InImage=reshape(double(InputImage)',irow*icol,1);
            temp=InImage;
            me=mean(temp);
            st=std(temp);
            temp=(temp-me)*ustd/st+um;
            NormImage = temp;
            Difference = temp-m;
            NormImage = Difference;

            p = [];
            aa=size(u,2);
            for i = 1:aa
                pare = dot(NormImage,u(:,i));
                p = [p; pare];
            end
            ReshapedImage = m + u(:,1:aa)*p;    %m is the mean
image, u is the eigenvector
            ReshapedImage = reshape(ReshapedImage,icol,irow);
            ReshapedImage = ReshapedImage';
            %show the reconstructed image.
            % subplot(1,2,2)
            % imagesc(ReshapedImage); colormap('gray');
            % title('Reconstructed image','fontsize',18)

            InImWeight = [];
            for i=1:size(u,2)
```

```
                t = u(:,i)';
                WeightOfInputImage = dot(t,Difference');
                InImWeight = [InImWeight; WeightOfInputImage];
            end

            % Find Euclidean distance
            e=[];
            for i=1:size(omega,2)
                q = omega(:,i);
                DiffWeight = InImWeight-q;
                mag = norm(DiffWeight);
                e = [e mag];
            end
           [values,order] = sort(e);
        all = [];
        x = order(1:80);
        molar  = size(find(x<=80),2);
        all = [all molar];
        x = order(1:80);
        pmolar  = size(find((x>=81 & x<=160)),2);
        all = [all pmolar];
        [r,idx] =sort(all, 'descend');

        all_r = [all_r idx(1)];
    class = all_r;
    all_class = [all_class;class]
end
LabelInfo = all_class;
```

```
function  reading_training_set()
%
==================================================================
=============================
% Programmer Name: Nourdin Al-sherif
%               West Virginia University
%               The ADIS Research project
%                (c) 2013 Nourdin Al-sherif. 2012
%                reading_training_set();
%
==================================================================
=============================
um=60;
ustd=30;

pat = pwd;
%read and show images(bmp);
S=[];   %img matrix
tag = [];
%path='J:\workspace\labeling\OLPP\olpp_1\teeth_set_3\';
pat = pwd;
pat1 = [pat '\teeth_set_4\'];
for i = 1:2
   if i == 1
     pat2 = [pat1 'Training\1'];
   end
   if i == 2
     pat2 = [pat1 'Training\2'];
   end
   if i == 3
     pat2 = [pat1 'Training\3'];
   end
   if i == 4
     pat2 = [pat1 'Training\4'];
   end

   pat3 = pat2;
   pat2 = [pat2 '\*.bmp'];
   %d=dir('*.bmp');
   d=dir(pat2);
   for j=1:length(d)
      im_name = [pat3 '\' d(j).name];
      img=imread(im_name);
      img = double(imresize(img, [32 32]));
      [irow icol]=size(img);    % get the number of rows (N1)
and columns (N2)
      temp=reshape(img',irow*icol,1);    %creates a (N1*N2)x1
```

```
matrix
       S=[S temp];           %X is a N1*N2xM matrix after finishing
the sequence
       tag = [tag,(i)];
   end
end
%Here we change the mean and std of all images. We normalize all
images.
%This is done to reduce the error due to lighting conditions.
for i=1:size(S,2)
     temp=double(S(:,i));
     m=mean(temp);
     st=std(temp);
     S(:,i)=(temp-m)*ustd/st+um;
end

%mean image;
m=mean(S,2);    %obtains the mean of each row instead of each
column
tmimg=uint8(m);    %converts to unsigned 8-bit integer. Values
range from 0 to 255
img=reshape(tmimg,icol,irow);    %takes the N1*N2x1 vector and
creates a N2xN1 matrix
img=img';        %creates a N1xN2 matrix by transposing the
image.
figure(3);
imshow(img);
title('Mean Image','fontsize',18)

% Change image for manipulation
dbx=[];   % A matrix
for i=1:size(S,2)
     temp=double(S(:,i));
     dbx=[dbx temp];
end
cd(pat);
options = [];
options.Metric = 'Euclidean';
options.NeighborMode = 'KNN';
options.k = 5;
options.WeightMode = 'HeatKernel';
options.t = 1;
W = constructW(dbx'./255,options);
options.PCARatio = 0.99;
options.ReducedDim = 25;
bSuccess = 0;
while ~bSuccess
```

```
    [eigvector, eigvalue, bSuccess] = OLPP(W, options,
dbx'./255);
end

dis_eigen(eigvector,icol,irow);

u =  eigvector;
% Find the weight of each face in the training set.
omega = [];
for h=1:size(dbx,2)
    WW=[];
    for i=1:size(u,2)
        t = u(:,i)';
        WeightOfImage = dot(t,dbx(:,h)');
        WW = [WW; WeightOfImage];
    end
    omega = [omega WW];
end



%% Testing
gen=[];imp=[];
for user = 1:2
    if user == 1
      pat2 = [pat1 'Testing\1'];
    end
    if user == 2
      pat2 = [pat1 'Testing\2'];
    end
    if user == 3
      pat2 = [pat1 'Testing\3'];
    end
    if user == 4
      pat2 = [pat1 'Testing\4'];
    end
    pat3 = pat2;
    pat2 = [pat2 '\*.bmp'];
    d=dir(pat2);
    for template=1:length(d)
        im_name = [pat3 '\' d(template).name];
        InputImage=imread(im_name);
        InputImage = double(imresize(InputImage, [32 32]));
        InImage=reshape(double(InputImage)',irow*icol,1);
        temp=InImage;
        me=mean(temp);
        st=std(temp);
        temp=(temp-me)*ustd/st+um;
        NormImage = temp;
        Difference = temp-m;
        NormImage = Difference;

        p = [];
        aa=size(u,2);
        for i = 1:aa
            pare = dot(NormImage,u(:,i));
            p = [p; pare];
        end
        ReshapedImage = m + u(:,1:aa)*p;    %m is the mean
image, u is the eigenvector
        ReshapedImage = reshape(ReshapedImage,icol,irow);
        ReshapedImage = ReshapedImage';
        %show the reconstructed image.
        % subplot(1,2,2)
        % imagesc(ReshapedImage); colormap('gray');
        % title('Reconstructed image','fontsize',18)

        InImWeight = [];
        for i=1:size(u,2)
            t = u(:,i)';
            WeightOfInputImage = dot(t,Difference');
            InImWeight = [InImWeight; WeightOfInputImage];
        end

        % Find Euclidean distance
        e=[];
        for i=1:size(omega,2)
            q = omega(:,i);
            DiffWeight = InImWeight-q;
            mag = norm(DiffWeight);
            e = [e mag];
        end

        for temp = 1: size(tag,2)
            if (tag(1,temp) == user)
                gen = [gen,e(1,temp)];
            else
                imp = [imp,e(1,temp)];
            end
        end

    end
end
cd(pat);
```

```
roc(gen,imp)

save([pat '\parameters\TeethOLPP.mat'], 'irow','icol',
'ustd','um','m','u','omega');
```

```
function [eigvector, eigvalue] = LaplacianTeeth()
%
================================================================
============================
% Programmer Name: Nourdin Al-sherif
%               West Virginia University
%               The ADIS Research project
%               (c) 2013 Nourdin Al-sherif. 2012
% [eigvector, eigvalue] = LaplacianTeeth();
%
================================================================
============================


% number of images on your training set.
M=20;
%Chosen std and mean.
%It can be any number that it is close to the std and mean of
most of the images.
um=60;
ustd=30;
pat = pwd;
pat1 = [pat '\teeth_set_3\Training\'];
pat2 = [pat1 '\*.bmp'];
%read and show images(bmp);
S=[];   %img matrix
tag = [];
d=dir(pat2);
  for j=1:length(d)
        im_name = [pat1 '\' d(j).name];
        img=imread(im_name);
        img = double(imresize(img, [32 32]));
        [irow icol]=size(img);    % get the number of rows (N1)
and columns (N2)
        temp=reshape(img',irow*icol,1);     %creates a (N1*N2)x1
matrix
        S=[S temp];         %X is a N1*N2xM matrix after finishing
the sequence
        tag = [tag,(i)];
   end
%Here we change the mean and std of all images. We normalize all
images.
%This is done to reduce the error due to lighting conditions.
for i=1:size(S,2)
    temp=double(S(:,i));
    m=mean(temp);
    st=std(temp);
    S(:,i)=(temp-m)*ustd/st+um;
end
%show normalized images
figure(2);
M = length(d);
for i=1:M
    str=strcat(int2str(i),'.bmp');
    img=reshape(S(:,i),icol,irow);
    img=img';
   % eval('imwrite(img,str)');
    subplot(ceil(sqrt(M)),ceil(sqrt(M)),i)
    imshow(img)
    drawnow;
    if i==3
        title('Normalized Training Set','fontsize',18)
    end
end
%mean image;
m=mean(S,2);   %obtains the mean of each row instead of each
column
tmimg=uint8(m);   %converts to unsigned 8-bit integer. Values
range from 0 to 255
img=reshape(tmimg,icol,irow);    %takes the N1*N2x1 vector and
creates a N2xN1 matrix
img=img';       %creates a N1xN2 matrix by transposing the
image.
figure(3);
imshow(img);
title('Mean Image','fontsize',18)

% Change image for manipulation
dbx=[];   % A matrix
for i=1:M
    temp=double(S(:,i));
    dbx=[dbx temp];
end
options = [];
options.Metric = 'Euclidean';
options.NeighborMode = 'KNN';
options.k = 5;
options.WeightMode = 'HeatKernel';
options.t = 1;
W = constructW(dbx'./255,options);
options.PCARatio = 0.99;
options.ReducedDim = 18;
bSuccess = 0;
while ~bSuccess
```

```
       [eigvector, eigvalue, bSuccess] = OLPP(W, options,
dbx'./255);
end
%eigvector
u = eigvector;
M=8;
% show eigenfaces
figure(4);
for i=1:size(u,2)
    img=reshape(u(:,i),icol,irow);
    img=img';
%     img=histeq(img,255);
%     subplot(ceil(sqrt(M)),ceil(sqrt(M)),i)
%     imshow(uint8(img))
    drawnow;
    ReshapedImage=img;
    imagesc(ReshapedImage); colormap('gray');

end
```

```
function [CorrSeq, CorrLbls, Rsk, RskVec, SqStrn] =
validateSequence(InLbls, InConf, subText)
 text = 'MMMPPCIIIICPPMMM';
Seqs = '8765432112345678';
ndx  = [4 4 4 3 3 2 1 1 1 1 2 3 3 4 4 4]';
CorrLbls = [];
CorrSeq = [];

if (~isempty(subText))
    delt = length(InLbls) - length(subText);
    if (delt < 0)
        delt = 1;
    end
    pos = findstr(subText, text);
    pos = max([1 pos-delt]);
    text = text(pos:end);
    Seqs = Seqs(pos:end);
    ndx = ndx(pos:end);

    pos = min([length(text) length(subText)+2*delt]);
    text = text(1:pos);
    Seqs = Seqs(1:pos);
    ndx = ndx(1:pos);
end

pos = findstr(InLbls, text);
n = length(InLbls);

if(~isempty(pos))
    for i = 1:length(pos)
        CorrSeq = [CorrSeq; Seqs(pos(i):pos(i)+n-1)];
        CorrLbls = [CorrLbls; InLbls];
    end
    Rsk = sum(min(InConf'));
    RskVec = min(InConf');
else
    mxRsk = sum(sum(InConf));
    N = length(text);
    t = 1;
    H = n;
    Vn = [1:n]';

    augStr = repmat('X', [1 N]);
    augStr(1:n) = InLbls;

    Rsk = mxRsk;
    for s = 1:N-n+1
```

```
        numChngs = InLbls - text(s:s+n-1);
        numChngs = length(find(numChngs));
        Cost = sum(InConf(sub2ind([n 4], Vn, ndx((s:s+n-1)))));%
+
        CostVec = InConf(sub2ind([n 4], Vn, ndx((s:s+n-1))));
        if(Cost < Rsk)
            Rsk = Cost;
            CorrSeq = Seqs(s:s+n-1);
            CorrLbls = text(s:s+n-1);
            RskVec = CostVec;
        elseif (Cost == Rsk)
            CorrSeq = [CorrSeq; Seqs(s:s+n-1)];
            CorrLbls = [CorrLbls; text(s:s+n-1)];
        end
        augStr =['X' augStr(1:end-1)];
        t = 1+t;
        H = 1+H;
    end
end
if (size(CorrLbls,1)>1)
    SqStrn = 0;
else
    SqStrn = 1;
end
```

```
function [feature, OutClass] = Filmclassify_UM(film)

%  Jindan Zhou @ University of Miami
%  Revision: 1.0 $   $Date: 03/22/2004 $
%  Image clssification based on Bayes rules

feature = FeatureVect(film);

if(feature(1)>1.5)
    cl=4;

else

    feature1(1)=feature(2);  feature1(2)=feature(3); feature1(3)
=feature(4); feature1(4)=feature(5);
    u1=[0.5,0.5,0.5,0.5];   u2=[0.07, 0.93, 0.32, 0.68]; u3=
[0.93, 0.07, 0.31, 0.69];
    f1 = 0.1; f2 = 1.49; f3 = 1.53;

    cp(1) = exp(-1/2*((feature1-u1)*(feature1-u1)')/f1)/((2*pi).
^(4/2)*sqrt(abs(f1)));
    cp(2) = exp(-1/2*((feature1-u2)*(feature1-u2)')/f2)/((2*pi).
^(4/2)*sqrt(abs(f2)));
    cp(3) = exp(-1/2*((feature1-u3)*(feature1-u3)')/f3)/((2*pi).
^(4/2)*sqrt(abs(f3)));

    [c,cl] = max(cp');
end

switch (cl)
    case 1
        OutClass=' Bite-wing';
    case 2
        OutClass=' Periapical Upper';
    case 3
        OutClass=' Periapical Lower';
    otherwise
        OutClass=' Panoramic';
end
```

```
function [feature] = FeatureVect(film)

%   Jindan Zhou @ University of Miami
%   Revision: 1.0 $   $Date: 05/04/2003 $
%   Extracting five fetures of dental image

    dental = film;
    if (length(size(dental))>2)
        dental = rgb2gray(dental);
    end
    [m,n] = size(dental);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

feature(1) = n/m;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ra = 10;

I = dental;
theta = 90-ra:90+ra;
[R,xp] = radon(I,theta);


[rm,rn] = size(R);
rm2 = floor(rm/2);
m2 = floor(m/2);
ma = floor(m/8);

horproj = R(rm2-m2+ma:rm2+m2-ma,:);
[R,xp] = radon(I,theta);
lxp = length(xp); lxp2 = floor(lxp/2);
xp = xp(lxp2-m2+ma:lxp2+m2-ma);

[C,I] = min(horproj);
[a,x] = min(C);
y1 = I(x);

ang = x-ra;
y = y1 +(- m2+ma);

cm = floor(m/2); cn = floor(n/2);

ycent = -y/(cos(ang/180*pi))+cm;

y1 = ycent+cm*tan(ang/180*pi);
```

```
y2 = ycent-cm*tan(ang/180*pi);

y1 = max([1 floor(y1)]); % Modified by DMNassar on Sep 19, 2005
y2 = max([1 floor(y2)]); % Modified by DMNassar on Sep 19, 2005

xc = 1:n;
yc = linspace(y1,y2,n);
yc = floor(yc);

dentalup = dental;
dentaldown = dental;

for k=1:n
    dentalup(yc(k):m,k) = 0;
    dentaldown(1:yc(k),k) = 0;
end

sumup = sum(sum(dentalup));
sumdown = sum(sum(dentaldown));

    feature(2) = sumdown/(sumup+sumdown);
    feature(3) = sumup/(sumup+sumdown);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%55

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


    edge1 = edgeH1(dental,'canny');
    edge1(1:10,:) = 0;
     edge1(m-10:m,:) = 0;

    edge2 = edgeH2(dental,'canny');
    edge2(1:10,:) = 0;
    edge2(m-10:m,:) = 0;

    sumedge1 = sum(sum(edge1));
    sumedge2 = sum(sum(edge2));

   feature(4) = sumedge1/(sumedge1+sumedge2);
   feature(5) = sumedge2/(sumedge1+sumedge2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

III. Appendix

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```
function [eout,thresh] = edgeH1(varargin)
%  Jindan Zhou @ University of Miami
%  Revision: 1.0 $   $Date: 05/04/2003 $
%  Edge detection using canny method to find type 1 horizontal
edges.

%EDGE Find edges in intensity image.
%   EDGE takes an intensity or a binary image I as its input,
and returns a
%   binary image BW of the same size as I, with 1's where the
function
%   finds edges in I and 0's elsewhere.
%
%   EDGE supports six different edge-finding methods:
%
%      The Sobel method finds edges using the Sobel
approximation to the
%      derivative. It returns edges at those points where the
gradient of
%      I is maximum.
%
%      The Prewitt method finds edges using the Prewitt
approximation to
%      the derivative. It returns edges at those points where
the gradient
%      of I is maximum.
%
%      The Roberts method finds edges using the Roberts
approximation to
%      the derivative. It returns edges at those points where
the gradient
%      of I is maximum.
%
%      The Laplacian of Gaussian method finds edges by looking
for zero
%      crossings after filtering I with a Laplacian of Gaussian
filter.
%
%      The zero-cross method finds edges by looking for zero
crossings
%      after filtering I with a filter you specify.
%
%      The Canny method finds edges by looking for local maxima
of the
%      gradient of I. The gradient is calculated using the
derivative of a
%      Gaussian filter. The method uses two thresholds, to
```

```
detect strong
%      and weak edges, and includes the weak edges in the output
only if
%      they are connected to strong edges. This method is
therefore less
%      likely than the others to be "fooled" by noise, and more
likely to
%      detect true weak edges.
%
%   The parameters you can supply differ depending on the method
you
%   specify. If you do not specify a method, EDGE uses the Sobel
method.
%
%   Sobel Method
%   ------------
%   BW = EDGE(I,'sobel') specifies the Sobel method.
%
%   BW = EDGE(I,'sobel',THRESH) specifies the sensitivity
threshold for
%   the Sobel method. EDGE ignores all edges that are not
stronger than
%   THRESH.  If you do not specify THRESH, or if THRESH is empty
([]),
%   EDGE chooses the value automatically.
%
%   BW = EDGE(I,'sobel',THRESH,DIRECTION) specifies
directionality for the
%   Sobel method. DIRECTION is a string specifying whether to
look for
%   'horizontal' or 'vertical' edges, or 'both' (the default).
%
%   [BW,thresh] = EDGE(I,'sobel',...) returns the threshold
value.
%
%   Prewitt Method
%   --------------
%   BW = EDGE(I,'prewitt') specifies the Prewitt method.
%
%   BW = EDGE(I,'prewitt',THRESH) specifies the sensitivity
threshold for
%   the Prewitt method. EDGE ignores all edges that are not
stronger than
%   THRESH. If you do not specify THRESH, or if THRESH is empty
([]),
%   EDGE chooses the value automatically.
%
```

```
%   BW = EDGE(I,'prewitt',THRESH,DIRECTION) specifies
directionality for
%   the Prewitt method. DIRECTION is a string specifying whether
to look
%   for 'horizontal' or 'vertical' edges, or 'both' (the
default).
%
%   [BW,thresh] = EDGE(I,'prewitt',...) returns the threshold
value.
%
%   Roberts Method
%   --------------
%   BW = EDGE(I,'roberts') specifies the Roberts method.
%
%   BW = EDGE(I,'roberts',THRESH) specifies the sensitivity
threshold for
%   the Roberts method. EDGE ignores all edges that are not
stronger than
%   THRESH. If you do not specify THRESH, or if THRESH is empty
([]),
%   EDGE chooses the value automatically.
%
%   [BW,thresh] = EDGE(I,'roberts',...) returns the threshold
value.
%
%   Laplacian of Gaussian Method
%   ----------------------------
%   BW = EDGE(I,'log') specifies the Laplacian of Gaussian
method.
%
%   BW = EDGE(I,'log',THRESH) specifies the sensitivity
threshold for the
%   Laplacian of Gaussian method. EDGE ignores all edges that
are not
%   stronger than THRESH. If you do not specify THRESH, or if
THRESH is
%   empty ([]), EDGE chooses the value automatically.
%
%   BW = EDGE(I,'log',THRESH,SIGMA) specifies the Laplacian of
Gaussian
%   method, using SIGMA as the standard deviation of the LoG
filter. The
%   default SIGMA is 2; the size of the filter is N-by-N, where
%   N=CEIL(SIGMA*3)*2+1.
%
%   [BW,thresh] = EDGE(I,'log',...) returns the threshold value.
%
```

```
%   Zero-cross Method
%   -----------------
%   BW = EDGE(I,'zerocross',THRESH,H) specifies the zero-cross
method,
%   using the specified filter H. If THRESH is empty ([]), EDGE
chooses
%   the sensitivity threshold automatically.
%
%   [BW,THRESH] = EDGE(I,'zerocross',...) returns the threshold
value.
%
%   Canny Method
%   ----------------------------
%   BW = EDGE(I,'canny') specifies the Canny method.
%
%   BW = EDGE(I,'canny',THRESH) specifies sensitivity thresholds
for the
%   Canny method. THRESH is a two-element vector in which the
first element
%   is the low threshold, and the second element is the high
threshold. If
%   you specify a scalar for THRESH, this value is used for the
high
%   threshold and 0.4*THRESH is used for the low threshold. If
you do not
%   specify THRESH, or if THRESH is empty ([]), EDGE chooses low
and high
%   values automatically.
%
%   BW = EDGE(I,'canny',THRESH,SIGMA) specifies the Canny
method, using
%   SIGMA as the standard deviation of the Gaussian filter. The
default
%   SIGMA is 1; the size of the filter is chosen automatically,
based
%   on SIGMA.
%
%   [BW,thresh] = EDGE(I,'canny',...) returns the threshold
values as a
%   two-element vector.
%
%   Class Support
%   -------------
%   I can be of class uint8, uint16, or double. BW is of class
uint8.
%
%   Remarks
%   -------
%   For the 'log' and 'zerocross' methods, if you specify a
%   threshold of 0, the output image has closed contours,
because
%   it includes all of the zero crossings in the input image.
%
%   Example
%   -------
%   Find the edges of the rice.tif image using the Prewitt and
Canny
%   methods:
%
%       I = imread('rice.tif');
%       BW1 = edge(I,'prewitt');
%       BW2 = edge(I,'canny');
%       imshow(BW1)
%       figure, imshow(BW2)
%
%   See also FSPECIAL.

%   OBSOLETE syntax
%   --------------------
%   BW = EDGE(... ,K) allows the specification of a
directionality
%   factor, K.  This only works for the 'sobel', 'prewitt', and
%   'roberts' methods.   K must be a 1-by-2 vector, K = [kx ky].
%   For Sobel and Prewitt, K=[1 0] looks for vertical edges,
%   K=[0 1] looks for horizontal edges, and K=[1 1], the
default,
%   looks for non-directional edges.   For the Roberts edge
detector,
%   K=[1 0] looks for 135 degree diagonal edges, K=[0 1] looks
%   for 45 degree diagonal edges, and K=[1 1], the default,
looks
%   for non-directional edges.
%
%   Copyright 1993-2002 The MathWorks, Inc.
%   $Revision: 5.26 $  $Date: 2002/03/26 16:39:10 $

[a,method,thresh,sigma,H,kx,ky] = parse_inputs(varargin{:});

% Transform to a double precision intensity image if necessary
if ~isa(a, 'double')
   a = im2double(a);
end

m = size(a,1);
```

```
n = size(a,2);
rr = 2:m-1; cc=2:n-1;

% The output edge map:
e = repmat(false, m, n);

if strcmp(method,'canny')
   % Magic numbers
   GaussianDieOff = .0001;
   PercentOfPixelsNotEdges = .7; % Used for selecting thresholds
   ThresholdRatio = .4;         % Low thresh is this fraction
of the high.

   % Design the filters - a gaussian and its derivative

   pw = 1:30; % possible widths
   ssq = sigma*sigma;
   width = max(find(exp(-(pw.*pw)/(2*sigma*sigma))>
GaussianDieOff));
   if isempty(width)
      width = 1;  % the user entered a really small sigma
   end

   t = (-width:width);
   gau = exp(-(t.*t)/(2*ssq))/(2*pi*ssq);     % the gaussian 1D
filter

   % Find the directional derivative of 2D Gaussian (along X-
axis)
   % Since the result is symmetric along X, we can get the
derivative along
   % Y-axis simply by transposing the result for X direction.
   [x,y]=meshgrid(-width:width,-width:width);
   dgau2D=-x.*exp(-(x.*x+y.*y)/(2*ssq))/(pi*ssq);

   % Convolve the filters with the image in each direction
   % The canny edge detector first requires convolution with
   % 2D gaussian, and then with the derivitave of a gaussian.
   % Since gaussian filter is separable, for smoothing, we can
use
   % two 1D convolutions in order to achieve the effect of
convolving
   % with 2D Gaussian.  We convolve along rows and then columns.

   %smooth the image out
   aSmooth=imfilter(a,gau,'conv','replicate');         % run the
filter accross rows
```

```
   aSmooth=imfilter(aSmooth,gau','conv','replicate');  % and
then accross columns

   %apply directional derivatives
   ax = imfilter(aSmooth, dgau2D, 'conv','replicate');
   ay = imfilter(aSmooth, dgau2D', 'conv','replicate');

   mag = sqrt((ax.*ax) + (ay.*ay));
   magmax = max(mag(:));
   if magmax>0
      mag = mag / magmax;   % normalize
   end

   % Select the thresholds
   if isempty(thresh)
      [counts,x]=imhist(mag, 64);
      highThresh = min(find(cumsum(counts) >
PercentOfPixelsNotEdges*m*n)) / 64;
      lowThresh = ThresholdRatio*highThresh;
      thresh = [lowThresh highThresh];
   elseif length(thresh)==1
      highThresh = thresh;
      if thresh>=1
         error('The threshold must be less than 1.');
      end
      lowThresh = ThresholdRatio*thresh;
      thresh = [lowThresh highThresh];
   elseif length(thresh)==2
      lowThresh = thresh(1);
      highThresh = thresh(2);
      if (lowThresh >= highThresh) | (highThresh >= 1)
         error('Thresh must be [low high], where low < high <
1.');
      end
   end

   % The next step is to do the non-maximum supression.
   % We will accrue indices which specify ON pixels in strong
edgemap
   % The array e will become the weak edge map.
   idxStrong = [];
   for dir = 2:3
      idxLocalMax = cannyFindLocalMaxima(dir,ax,ay,mag);
      idxWeak = idxLocalMax(mag(idxLocalMax) > lowThresh);
      e(idxWeak)=1;
      idxStrong = [idxStrong; idxWeak(mag(idxWeak) >
highThresh)];
```

```
   end
%    for dir = 1:4
%       idxLocalMax = cannyFindLocalMaxima(dir,ax,ay,mag);
%       idxWeak = idxLocalMax(mag(idxLocalMax) > lowThresh);
%       e(idxWeak)=1;
%       idxStrong = [idxStrong; idxWeak(mag(idxWeak) >
highThresh)];
%    end
   rstrong = rem(idxStrong-1, m)+1;
   cstrong = floor((idxStrong-1)/m)+1;
   e = bwselect(e, cstrong, rstrong, 8);
   e = bwmorph(e, 'thin', 1);  % Thin double (or triple) pixel
wide contours

elseif any(strcmp(method, {'log','marr-hildreth','zerocross'}))
   % We don't use image blocks here
   if isempty(H),
      fsize = ceil(sigma*3) * 2 + 1;  % choose an odd fsize > 6
*sigma;
      op = fspecial('log',fsize,sigma);
   else
      op = H;
   end

   op = op - sum(op(:))/prod(size(op)); % make the op to sum to
zero
   b = filter2(op,a);

   if isempty(thresh)
      thresh = .75*mean2(abs(b(rr,cc)));
   end

   % Look for the zero crossings:  +-, -+ and their transposes
   % We arbitrarily choose the edge to be the negative point
   [rx,cx] = find( b(rr,cc) < 0 & b(rr,cc+1) > 0 ...
      & abs( b(rr,cc)-b(rr,cc+1) ) > thresh );   % [- +]
   e((rx+1) + cx*m) = 1;
   [rx,cx] = find( b(rr,cc-1) > 0 & b(rr,cc) < 0 ...
      & abs( b(rr,cc-1)-b(rr,cc) ) > thresh );   % [+ -]
   e((rx+1) + cx*m) = 1;
   [rx,cx] = find( b(rr,cc) < 0 & b(rr+1,cc) > 0 ...
      & abs( b(rr,cc)-b(rr+1,cc) ) > thresh);   % [- +]'
   e((rx+1) + cx*m) = 1;
   [rx,cx] = find( b(rr-1,cc) > 0 & b(rr,cc) < 0 ...
      & abs( b(rr-1,cc)-b(rr,cc) ) > thresh);   % [+ -]'
   e((rx+1) + cx*m) = 1;
   % Most likely this covers all of the cases.   Just check to
see if there
   % are any points where the LoG was precisely zero:
   [rz,cz] = find( b(rr,cc)==0 );
   if ~isempty(rz)
      % Look for the zero crossings: +0-, -0+ and their
transposes
      % The edge lies on the Zero point
      zero = (rz+1) + cz*m;   % Linear index for zero points
      zz = find(b(zero-1) < 0 & b(zero+1) > 0 ...
         & abs( b(zero-1)-b(zero+1) ) > 2*thresh);     % [- 0
+]'
      e(zero(zz)) = 1;
      zz = find(b(zero-1) > 0 & b(zero+1) < 0 ...
         & abs( b(zero-1)-b(zero+1) ) > 2*thresh);     % [+ 0
-]'
      e(zero(zz)) = 1;
      zz = find(b(zero-m) < 0 & b(zero+m) > 0 ...
         & abs( b(zero-m)-b(zero+m) ) > 2*thresh);     % [- 0 +]
      e(zero(zz)) = 1;
      zz = find(b(zero-m) > 0 & b(zero+m) < 0 ...
         & abs( b(zero-m)-b(zero+m) ) > 2*thresh);     % [+ 0 -]
      e(zero(zz)) = 1;
   end

else  % one of the easy methods (roberts,sobel,prewitt)

   % Determine edges in blocks for easy methods
   nr = length(rr); nc = length(cc);

   blk = bestblk([nr nc]);
   nblks = floor([nr nc]./blk); nrem = [nr nc] - nblks.*blk;
   mblocks = nblks(1); nblocks = nblks(2);
   mb = blk(1); nb = blk(2);

   if strcmp(method,'sobel')
      op = [-1 -2 -1;0 0 0;1 2 1]/8; % Sobel approximation to
derivative
      bx = abs(filter2(op',a)); by = abs(filter2(op,a));
      b = kx*bx.*bx + ky*by.*by;
      if isempty(thresh), % Determine cutoff based on RMS
estimate of noise
         cutoff = 4*sum(sum(b(rr,cc)))/prod(size(b(rr,cc)));
thresh = sqrt(cutoff);
      else                     % Use relative tolerance specified
by the user
         cutoff = (thresh).^2;
      end
      rows = 1:blk(1);
      for i=0:mblocks,
         if i==mblocks, rows = (1:nrem(1)); end
         for j=0:nblocks,
            if j==0, cols = 1:blk(2); elseif j==nblocks, cols=
(1:nrem(2)); end
            if ~isempty(rows) & ~isempty(cols)
               r = rr(i*mb+rows); c = cc(j*nb+cols);
               e(r,c) = (b(r,c)>cutoff) & ...
               ( ( (bx(r,c) >= (kx*by(r,c)-eps*100)) & ...
               (b(r,c-1) <= b(r,c)) & (b(r,c) > b(r,c+1)) ) |
...
               ( (by(r,c) >= (ky*bx(r,c)-eps*100 )) & ...
               (b(r-1,c) <= b(r,c)) & (b(r,c) > b(r+1,c))));
            end
         end
      end

   elseif strcmp(method,'prewitt')
      op = [-1 -1 -1;0 0 0;1 1 1]/6; % Prewitt approximation to
derivative
      bx = abs(filter2(op',a)); by = abs(filter2(op,a));
      b = kx*bx.*bx + ky*by.*by;
      if isempty(thresh), % Determine cutoff based on RMS
estimate of noise
         cutoff = 4*sum(sum(b(rr,cc)))/prod(size(b(rr,cc)));
thresh = sqrt(cutoff);
      else                   % Use relative tolerance specified
by the user
         cutoff = (thresh).^2;
      end
      rows = 1:blk(1);
      for i=0:mblocks,
         if i==mblocks, rows = (1:nrem(1)); end
         for j=0:nblocks,
            if j==0, cols = 1:blk(2); elseif j==nblocks, cols=
(1:nrem(2)); end
            if ~isempty(rows) & ~isempty(cols)
               r = rr(i*mb+rows); c = cc(j*nb+cols);
               e(r,c) = (b(r,c)>cutoff) & ...
               ( ( (bx(r,c) >= (kx*by(r,c)-eps*100) ) & ...
               (b(r,c-1) <= b(r,c)) & (b(r,c) > b(r,c+1)) ) |
...
               ((by(r,c) >= (ky*bx(r,c)-eps*100) )  & ...
               (b(r-1,c) <= b(r,c)) & (b(r,c) > b(r+1,c)) ) );
            end
         end
      end

   elseif strcmp(method, 'roberts')
      op = [1 0;0 -1]/sqrt(2); % Roberts approximation to
diagonal derivative
      bx = abs(filter2(op,a)); by = abs(filter2(rot90(op),a));
      b = kx*bx.*bx + ky*by.*by;
      if isempty(thresh), % Determine cutoff based on RMS
estimate of noise
         cutoff = 6*sum(sum(b(rr,cc)))/prod(size(b(rr,cc)));
thresh = sqrt(cutoff);
      else                   % Use relative tolerance specified
by the user
         cutoff = (thresh).^2;
      end
      rows = 1:blk(1);
      for i=0:mblocks,
         if i==mblocks, rows = (1:nrem(1)); end
         for j=0:nblocks,
            if j==0, cols = 1:blk(2); elseif j==nblocks, cols=
(1:nrem(2)); end
            if ~isempty(rows) & ~isempty(cols)
               r = rr(i*mb+rows); c = cc(j*nb+cols);
               e(r,c) = (b(r,c)>cutoff) & ...
               ( ( (bx(r,c) >= (kx*by(r,c)-eps*100)) & ...
               (b(r-1,c-1) <= b(r,c)) & (b(r,c) > b(r+1,c+1)) )
| ...
               ( (by(r,c) >= (ky*bx(r,c)-eps*100)) & ...
               (b(r-1,c+1) <= b(r,c)) & (b(r,c) > b(r+1,c-1)) )
);
            end
         end
      end
   else
      error([method,' is not a valid method.']);
   end
end

if nargout==0,
   imshow(e);
else
   eout = e;
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%
%   Local Function : cannyFindLocalMaxima
%
function idxLocalMax = cannyFindLocalMaxima
(direction,ix,iy,mag);
%
% This sub-function helps with the non-maximum supression in the
Canny
% edge detector.  The input parameters are:
%
%   direction - the index of which direction the gradient is
pointing,
%               read from the diagram below. direction is 1, 2,
3, or 4.
%   ix        - input image filtered by derivative of gaussian
along x
%   iy        - input image filtered by derivative of gaussian
along y
%   mag       - the gradient magnitude image
%
%    there are 4 cases:
%
%                         The X marks the pixel in question, and
each
%         3     2         of the quadrants for the gradient
vector
%       O----0----0       fall into two cases, divided by the 45
%     4 |         | 1     degree line.  In one case the gradient
%       |         |       vector is more horizontal, and in the
other
%       O    X    O       it is more vertical.  There are eight
%       |         |       divisions, but for the non-maximum
supression
%    (1)|         |(4)    we are only worried about 4 of them
since we
%       O----O----O       use symmetric points about the center
pixel.
%        (2)   (3)


[m,n,o] = size(mag);

% Find the indices of all points whose gradient (specified by
the
% vector (ix,iy)) is going in the direction we're looking at.

switch direction
```

```
case 1
   idx = find((iy<=0 & ix>-iy)  | (iy>=0 & ix<-iy));
case 2
   idx = find( (ix<0 & -iy<=ix));%idx = find((ix>0 & -iy>=ix)   |
(ix<0 & -iy<=ix));
case 3
   idx = find((ix>=0 & ix<iy) );%idx = find((ix<=0 & ix>iy) |
(ix>=0 & ix<iy));
case 4
   idx = find((iy<0 & ix<=iy) | (iy>0 & ix>=iy));
end

% Exclude the exterior pixels
if ~isempty(idx)
   v = mod(idx,m);
   extIdx = find(v==1 | v==0 | idx<=m | (idx>(n-1)*m));
   idx(extIdx) = [];
end

ixv = ix(idx);
iyv = iy(idx);
gradmag = mag(idx);

% Do the linear interpolations for the interior pixels
switch direction
case 1
   d = abs(iyv./ixv);
   gradmag1 = mag(idx+m).*(1-d) + mag(idx+m-1).*d;
   gradmag2 = mag(idx-m).*(1-d) + mag(idx-m+1).*d;
case 2
   d = abs(ixv./iyv);
   gradmag1 = mag(idx-1).*(1-d) + mag(idx+m-1).*d;
   gradmag2 = mag(idx+1).*(1-d) + mag(idx-m+1).*d;
case 3
   d = abs(ixv./iyv);
   gradmag1 = mag(idx-1).*(1-d) + mag(idx-m-1).*d;
   gradmag2 = mag(idx+1).*(1-d) + mag(idx+m+1).*d;
case 4
   d = abs(iyv./ixv);
   gradmag1 = mag(idx-m).*(1-d) + mag(idx-m-1).*d;
   gradmag2 = mag(idx+m).*(1-d) + mag(idx+m+1).*d;
end
idxLocalMax = idx(gradmag>=gradmag1 & gradmag>=gradmag2);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%
%   Local Function : parse_inputs
%
function [I,Method,Thresh,Sigma,H,kx,ky] = parse_inputs
(varargin)
% OUTPUTS:
%   I      Image Data
%   Method Edge detection method
%   Thresh Threshold value
%   Sigma  standard deviation of Gaussian
%   H      Filter for Zero-crossing detection
%   kx,ky  From Directionality vector

error(nargchk(1,5,nargin));

I = varargin{1};

checkinput(I,{'double','logical','uint8','uint16'},...
             {'nonsparse'},mfilename,'I',1);

% Defaults
Method='sobel';
Thresh=[];
Direction='both';
Sigma=2;
H=[];
K=[1 1];

methods = {'canny','prewitt','sobel','marr-
hildreth','log','roberts','zerocross'};
directions = {'both','horizontal','vertical'};

% Now parse the nargin-1 remaining input arguments

% First get the strings - we do this because the intepretation
of the
% rest of the arguments will depend on the method.
nonstr = [];   % ordered indices of non-string arguments
for i = 2:nargin
   if ischar(varargin{i})
      str = lower(varargin{i});
      j = strmatch(str,methods);
      k = strmatch(str,directions);
      if ~isempty(j)
         Method = methods{j(1)};
         if strcmp(Method,'marr-hildreth')
            warning('''Marr-Hildreth'' is an obsolete syntax,
use ''LoG'' instead.');
         end
      elseif ~isempty(k)
         Direction = directions{k(1)};
      else
         error(['Invalid input string: ''' varargin{i} '''.']);
      end
   else
      nonstr = [nonstr i];
   end
end

% Now get the rest of the arguments

switch Method

case {'prewitt','sobel','roberts'}
   threshSpecified = 0;  % Threshold is not yet specified
   for i = nonstr
      if prod(size(varargin{i}))<=1 & ~threshSpecified % Scalar
or empty
         Thresh = varargin{i};
         threshSpecified = 1;
      elseif prod(size(varargin{i}))==2  % The dreaded K vector
         warning(['BW = EDGE(... , K) is an obsolete syntax.
'...
           'Use BW = EDGE(... , DIRECTION), where DIRECTION is a
string.']);
         K=varargin{i};
      else
         error('Invalid input arguments');
      end
   end

case 'canny'
   Sigma = 1.0;          % Default Std dev of gaussian for canny
   threshSpecified = 0;  % Threshold is not yet specified
   for i = nonstr
      if prod(size(varargin{i}))==2 & ~threshSpecified
         Thresh = varargin{i};
         threshSpecified = 1;
      elseif prod(size(varargin{i}))==1
         if ~threshSpecified
            Thresh = varargin{i};
            threshSpecified = 1;
         else
            Sigma = varargin{i};
         end
      elseif isempty(varargin{i}) & ~threshSpecified
         % Thresh = [];
         threshSpecified = 1;
      else
         error('Invalid input arguments');
      end
   end

case 'log'
   threshSpecified = 0;  % Threshold is not yet specified
   for i = nonstr
      if prod(size(varargin{i}))<=1  % Scalar or empty
         if ~threshSpecified
            Thresh = varargin{i};
            threshSpecified = 1;
         else
            Sigma = varargin{i};
         end
      else
         error('Invalid input arguments');
      end
   end

case 'zerocross'
   threshSpecified = 0;  % Threshold is not yet specified
   for i = nonstr
      if prod(size(varargin{i}))<=1 & ~threshSpecified % Scalar
or empty
         Thresh = varargin{i};
         threshSpecified = 1;
      elseif prod(size(varargin{i})) > 1 % The filter for
zerocross
         H = varargin{i};
      else
         error('Invalid input arguments');
      end
   end

case 'marr-hildreth'
   for i = nonstr
      if prod(size(varargin{i}))<=1  % Scalar or empty
         Thresh = varargin{i};
      elseif prod(size(varargin{i}))==2  % The dreaded K vector
         warning('The [kx ky] direction factor has no effect for
''Marr-Hildreth''.');
      elseif prod(size(varargin{i})) > 2 % The filter for
```

```
zerocross
         H = varargin{i};
      else
         error('Invalid input arguments');
      end
   end

otherwise
   error('Invalid input arguments');
end

if Sigma<=0
   error('Sigma must be positive');
end

switch Direction
case 'both',
   kx = K(1); ky = K(2);
case 'horizontal',
   kx = 0; ky = 1; % Directionality factor
case 'vertical',
   kx = 1; ky = 0; % Directionality factor
otherwise
   error('Unrecognized direction string');
end

%
% if isrgb(I)
%    error('RGB images are not supported. Call RGB2GRAY
first.');
% end
```

```
function [eout,thresh] = edgeH2(varargin)
%  Jindan Zhou @ University of Miami
%  Revision: 1.0 $   $Date: 05/04/2003 $
%  Edge detection using canny method to find type 2 horizontal
edges.


   %smooth the image out
   aSmooth=imfilter(a,gau,'conv','replicate');         % run the
filter accross rows
   aSmooth=imfilter(aSmooth,gau','conv','replicate');  % and
then accross columns

   %apply directional derivatives
   ax = imfilter(aSmooth, dgau2D, 'conv','replicate');
   ay = imfilter(aSmooth, dgau2D', 'conv','replicate');

   mag = sqrt((ax.*ax) + (ay.*ay));
   magmax = max(mag(:));
   if magmax>0
      mag = mag / magmax;   % normalize
   end

   % Select the thresholds
   if isempty(thresh)
      [counts,x]=imhist(mag, 64);
      highThresh = min(find(cumsum(counts) >
PercentOfPixelsNotEdges*m*n)) / 64;
      lowThresh = ThresholdRatio*highThresh;
      thresh = [lowThresh highThresh];
   elseif length(thresh)==1
      highThresh = thresh;
      if thresh>=1
         error('The threshold must be less than 1.');
      end
      lowThresh = ThresholdRatio*thresh;
      thresh = [lowThresh highThresh];
   elseif length(thresh)==2
      lowThresh = thresh(1);
      highThresh = thresh(2);
      if (lowThresh >= highThresh) | (highThresh >= 1)
         error('Thresh must be [low high], where low < high <
1.');
      end
   end

   % The next step is to do the non-maximum supression.
   % We will accrue indices which specify ON pixels in strong
edgemap
   % The array e will become the weak edge map.
   idxStrong = [];
   for dir = 2:3
      idxLocalMax = cannyFindLocalMaxima(dir,ax,ay,mag);
      idxWeak = idxLocalMax(mag(idxLocalMax) > lowThresh);
      e(idxWeak)=1;
      idxStrong = [idxStrong; idxWeak(mag(idxWeak) >
highThresh)];
   end
%    for dir = 1:4
%       idxLocalMax = cannyFindLocalMaxima(dir,ax,ay,mag);
%       idxWeak = idxLocalMax(mag(idxLocalMax) > lowThresh);
%       e(idxWeak)=1;
%       idxStrong = [idxStrong; idxWeak(mag(idxWeak) >
highThresh)];
%    end
   rstrong = rem(idxStrong-1, m)+1;
   cstrong = floor((idxStrong-1)/m)+1;
   e = bwselect(e, cstrong, rstrong, 8);
   e = bwmorph(e, 'thin', 1);  % Thin double (or triple) pixel
wide contours

elseif any(strcmp(method, {'log','marr-hildreth','zerocross'}))
   % We don't use image blocks here
   if isempty(H),
      fsize = ceil(sigma*3) * 2 + 1;  % choose an odd fsize > 6
*sigma;
      op = fspecial('log',fsize,sigma);
   else
      op = H;
   end

   op = op - sum(op(:))/prod(size(op)); % make the op to sum to
zero
   b = filter2(op,a);

   if isempty(thresh)
      thresh = .75*mean2(abs(b(rr,cc)));
   end

   % Look for the zero crossings:  +-, -+ and their transposes
   % We arbitrarily choose the edge to be the negative point
   [rx,cx] = find( b(rr,cc) < 0 & b(rr,cc+1) > 0 ...
      & abs( b(rr,cc)-b(rr,cc+1) ) > thresh );   % [- +]
   e((rx+1) + cx*m) = 1;
   [rx,cx] = find( b(rr,cc-1) > 0 & b(rr,cc) < 0 ...
      & abs( b(rr,cc-1)-b(rr,cc) ) > thresh );    % [+ -]
   e((rx+1) + cx*m) = 1;
   [rx,cx] = find( b(rr,cc) < 0 & b(rr+1,cc) > 0 ...
      & abs( b(rr,cc)-b(rr+1,cc) ) > thresh);    % [- +]'
   e((rx+1) + cx*m) = 1;
   [rx,cx] = find( b(rr-1,cc) > 0 & b(rr,cc) < 0 ...
      & abs( b(rr-1,cc)-b(rr,cc) ) > thresh);    % [+ -]'
   e((rx+1) + cx*m) = 1;

   % Most likely this covers all of the cases.   Just check to
see if there
   % are any points where the LoG was precisely zero:
   [rz,cz] = find( b(rr,cc)==0 );
   if ~isempty(rz)
      % Look for the zero crossings: +0-, -0+ and their
transposes
      % The edge lies on the Zero point
      zero = (rz+1) + cz*m;   % Linear index for zero points
      zz = find(b(zero-1) < 0 & b(zero+1) > 0 ...
         & abs( b(zero-1)-b(zero+1) ) > 2*thresh);     % [- 0
+]'
      e(zero(zz)) = 1;
      zz = find(b(zero-1) > 0 & b(zero+1) < 0 ...
         & abs( b(zero-1)-b(zero+1) ) > 2*thresh);     % [+ 0
-]'
      e(zero(zz)) = 1;
      zz = find(b(zero-m) < 0 & b(zero+m) > 0 ...
         & abs( b(zero-m)-b(zero+m) ) > 2*thresh);     % [- 0 +]
      e(zero(zz)) = 1;
      zz = find(b(zero-m) > 0 & b(zero+m) < 0 ...
         & abs( b(zero-m)-b(zero+m) ) > 2*thresh);     % [+ 0 -]
      e(zero(zz)) = 1;
   end

else  % one of the easy methods (roberts,sobel,prewitt)

   % Determine edges in blocks for easy methods
   nr = length(rr); nc = length(cc);

   blk = bestblk([nr nc]);
   nblks = floor([nr nc]./blk); nrem = [nr nc] - nblks.*blk;
   mblocks = nblks(1); nblocks = nblks(2);
   mb = blk(1); nb = blk(2);

   if strcmp(method,'sobel')
      op = [-1 -2 -1;0 0 0;1 2 1]/8; % Sobel approximation to
```

```
derivative
      bx = abs(filter2(op',a)); by = abs(filter2(op,a));
      b = kx*bx.*bx + ky*by.*by;
      if isempty(thresh), % Determine cutoff based on RMS
estimate of noise
         cutoff = 4*sum(sum(b(rr,cc)))/prod(size(b(rr,cc)));
thresh = sqrt(cutoff);
      else                     % Use relative tolerance specified
by the user
         cutoff = (thresh).^2;
      end
      rows = 1:blk(1);
      for i=0:mblocks,
         if i==mblocks, rows = (1:nrem(1)); end
         for j=0:nblocks,
            if j==0, cols = 1:blk(2); elseif j==nblocks, cols=
(1:nrem(2)); end
            if ~isempty(rows) & ~isempty(cols)
               r = rr(i*mb+rows); c = cc(j*nb+cols);
               e(r,c) = (b(r,c)>cutoff) & ...
               ( ( (bx(r,c) >= (kx*by(r,c)-eps*100)) & ...
               (b(r,c-1) <= b(r,c)) & (b(r,c) > b(r,c+1)) ) |
...
               ( (by(r,c) >= (ky*bx(r,c)-eps*100 )) & ...
               (b(r-1,c) <= b(r,c)) & (b(r,c) > b(r+1,c))));
            end
         end
      end

   elseif strcmp(method,'prewitt')
      op = [-1 -1 -1;0 0 0;1 1 1]/6; % Prewitt approximation to
derivative
      bx = abs(filter2(op',a)); by = abs(filter2(op,a));
      b = kx*bx.*bx + ky*by.*by;
      if isempty(thresh), % Determine cutoff based on RMS
estimate of noise
         cutoff = 4*sum(sum(b(rr,cc)))/prod(size(b(rr,cc)));
thresh = sqrt(cutoff);
      else                     % Use relative tolerance specified
by the user
         cutoff = (thresh).^2;
      end
      rows = 1:blk(1);
      for i=0:mblocks,
         if i==mblocks, rows = (1:nrem(1)); end
         for j=0:nblocks,
            if j==0, cols = 1:blk(2); elseif j==nblocks, cols=
(1:nrem(2)); end
            if ~isempty(rows) & ~isempty(cols)
               r = rr(i*mb+rows); c = cc(j*nb+cols);
               e(r,c) = (b(r,c)>cutoff) & ...
               ( ( (bx(r,c) >= (kx*by(r,c)-eps*100) ) & ...
               (b(r,c-1) <= b(r,c)) & (b(r,c) > b(r,c+1)) ) |
...
               ((by(r,c) >= (ky*bx(r,c)-eps*100) )  & ...
               (b(r-1,c) <= b(r,c)) & (b(r,c) > b(r+1,c)) ) );
            end
         end
      end

   elseif strcmp(method, 'roberts')
      op = [1 0;0 -1]/sqrt(2); % Roberts approximation to
diagonal derivative
      bx = abs(filter2(op,a)); by = abs(filter2(rot90(op),a));
      b = kx*bx.*bx + ky*by.*by;
      if isempty(thresh), % Determine cutoff based on RMS
estimate of noise
         cutoff = 6*sum(sum(b(rr,cc)))/prod(size(b(rr,cc)));
thresh = sqrt(cutoff);
      else                   % Use relative tolerance specified
by the user
         cutoff = (thresh).^2;
      end
      rows = 1:blk(1);
      for i=0:mblocks,
         if i==mblocks, rows = (1:nrem(1)); end
         for j=0:nblocks,
            if j==0, cols = 1:blk(2); elseif j==nblocks, cols=
(1:nrem(2)); end
            if ~isempty(rows) & ~isempty(cols)
               r = rr(i*mb+rows); c = cc(j*nb+cols);
               e(r,c) = (b(r,c)>cutoff) & ...
               ( ( (bx(r,c) >= (kx*by(r,c)-eps*100)) & ...
               (b(r-1,c-1) <= b(r,c)) & (b(r,c) > b(r+1,c+1)) )
| ...
               ( (by(r,c) >= (ky*bx(r,c)-eps*100)) & ...
               (b(r-1,c+1) <= b(r,c)) & (b(r,c) > b(r+1,c-1)) )
);
            end
         end
      end
   else
      error([method,' is not a valid method.']);
   end
```

```
end

if nargout==0,
   imshow(e);
else
   eout = e;
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   Local Function : cannyFindLocalMaxima
%
function idxLocalMax = cannyFindLocalMaxima
(direction,ix,iy,mag);
%
% This sub-function helps with the non-maximum supression in the
Canny
% edge detector.  The input parameters are:
%
%   direction - the index of which direction the gradient is
pointing,
%               read from the diagram below. direction is 1, 2,
3, or 4.
%   ix        - input image filtered by derivative of gaussian
along x
%   iy        - input image filtered by derivative of gaussian
along y
%   mag       - the gradient magnitude image
%
%    there are 4 cases:
%
%                         The X marks the pixel in question, and
each
%         3     2         of the quadrants for the gradient
vector
%       O----O----O       fall into two cases, divided by the 45
%     4 |         | 1     degree line.  In one case the gradient
%       |         |       vector is more horizontal, and in the
other
%       O    X    O       it is more vertical.  There are eight
%       |         |       divisions, but for the non-maximum
supression
%    (1)|         |(4)    we are only worried about 4 of them
since we
%       O----O----O       use symmetric points about the center
pixel.
```

```
%          (2)    (3)


[m,n,o] = size(mag);

% Find the indices of all points whose gradient (specified by
the
% vector (ix,iy)) is going in the direction we're looking at.

switch direction
case 1
   idx = find((iy<=0 & ix>-iy)  | (iy>=0 & ix<-iy));
case 2
   idx = find((ix>0 & -iy>=ix) );%idx = find((ix>0 & -iy>=ix)   |
(ix<0 & -iy<=ix));
case 3
   idx = find((ix<=0 & ix>iy));% idx = find((ix<=0 & ix>iy) |
(ix>=0 & ix<iy));
case 4
   idx = find((iy<0 & ix<=iy) | (iy>0 & ix>=iy));
end

% Exclude the exterior pixels
if ~isempty(idx)
   v = mod(idx,m);
   extIdx = find(v==1 | v==0 | idx<=m | (idx>(n-1)*m));
   idx(extIdx) = [];
end

ixv = ix(idx);
iyv = iy(idx);
gradmag = mag(idx);

% Do the linear interpolations for the interior pixels
switch direction
case 1
   d = abs(iyv./ixv);
   gradmag1 = mag(idx+m).*(1-d)  + mag(idx+m-1).*d;
   gradmag2 = mag(idx-m).*(1-d)  + mag(idx-m+1).*d;
case 2
   d = abs(ixv./iyv);
   gradmag1 = mag(idx-1).*(1-d)  + mag(idx+m-1).*d;
   gradmag2 = mag(idx+1).*(1-d)  + mag(idx-m+1).*d;
case 3
   d = abs(ixv./iyv);
   gradmag1 = mag(idx-1).*(1-d)  + mag(idx-m-1).*d;
   gradmag2 = mag(idx+1).*(1-d)  + mag(idx+m+1).*d;
```

```
case 4
   d = abs(iyv./ixv);
   gradmag1 = mag(idx-m).*(1-d) + mag(idx-m-1).*d;
   gradmag2 = mag(idx+m).*(1-d) + mag(idx+m+1).*d;
end
idxLocalMax = idx(gradmag>=gradmag1 & gradmag>=gradmag2);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   Local Function : parse_inputs
%
function [I,Method,Thresh,Sigma,H,kx,ky] = parse_inputs
(varargin)
% OUTPUTS:
%   I      Image Data
%   Method Edge detection method
%   Thresh Threshold value
%   Sigma  standard deviation of Gaussian
%   H      Filter for Zero-crossing detection
%   kx,ky  From Directionality vector

error(nargchk(1,5,nargin));

I = varargin{1};

checkinput(I,{'double','logical','uint8','uint16'},...
             {'nonsparse'},mfilename,'I',1);

% Defaults
Method='sobel';
Thresh=[];
Direction='both';
Sigma=2;
H=[];
K=[1 1];

methods = {'canny','prewitt','sobel','marr-
hildreth','log','roberts','zerocross'};
directions = {'both','horizontal','vertical'};

% Now parse the nargin-1 remaining input arguments

% First get the strings - we do this because the intepretation
of the
% rest of the arguments will depend on the method.
```

```
nonstr = [];   % ordered indices of non-string arguments
for i = 2:nargin
   if ischar(varargin{i})
      str = lower(varargin{i});
      j = strmatch(str,methods);
      k = strmatch(str,directions);
      if ~isempty(j)
         Method = methods{j(1)};
         if strcmp(Method,'marr-hildreth')
            warning('''Marr-Hildreth'' is an obsolete syntax,
use ''LoG'' instead.');
         end
      elseif ~isempty(k)
         Direction = directions{k(1)};
      else
         error(['Invalid input string: ''' varargin{i} '''.']);
      end
   else
      nonstr = [nonstr i];
   end
end

% Now get the rest of the arguments

switch Method

case {'prewitt','sobel','roberts'}
   threshSpecified = 0;  % Threshold is not yet specified
   for i = nonstr
      if prod(size(varargin{i}))<=1 & ~threshSpecified % Scalar
or empty
         Thresh = varargin{i};
         threshSpecified = 1;
      elseif prod(size(varargin{i}))==2  % The dreaded K vector
         warning(['BW = EDGE(... , K) is an obsolete syntax.
'...
           'Use BW = EDGE(... , DIRECTION), where DIRECTION is a
string.']);
         K=varargin{i};
      else
         error('Invalid input arguments');
      end
   end

case 'canny'
   Sigma = 1.0;          % Default Std dev of gaussian for canny
   threshSpecified = 0;  % Threshold is not yet specified
   for i = nonstr
      if prod(size(varargin{i}))==2 & ~threshSpecified
         Thresh = varargin{i};
         threshSpecified = 1;
      elseif prod(size(varargin{i}))==1
         if ~threshSpecified
            Thresh = varargin{i};
            threshSpecified = 1;
         else
            Sigma = varargin{i};
         end
      elseif isempty(varargin{i}) & ~threshSpecified
         % Thresh = [];
         threshSpecified = 1;
      else
         error('Invalid input arguments');
      end
   end

case 'log'
   threshSpecified = 0;  % Threshold is not yet specified
   for i = nonstr
      if prod(size(varargin{i}))<=1  % Scalar or empty
         if ~threshSpecified
            Thresh = varargin{i};
            threshSpecified = 1;
         else
            Sigma = varargin{i};
         end
      else
         error('Invalid input arguments');
      end
   end

case 'zerocross'
   threshSpecified = 0;  % Threshold is not yet specified
   for i = nonstr
      if prod(size(varargin{i}))<=1 & ~threshSpecified % Scalar
or empty
         Thresh = varargin{i};
         threshSpecified = 1;
      elseif prod(size(varargin{i})) > 1 % The filter for
zerocross
         H = varargin{i};
      else
         error('Invalid input arguments');
      end
      end

   case 'marr-hildreth'
      for i = nonstr
         if prod(size(varargin{i}))<=1  % Scalar or empty
            Thresh = varargin{i};
         elseif prod(size(varargin{i}))==2  % The dreaded K vector
            warning('The [kx ky] direction factor has no effect for
''Marr-Hildreth''.');
         elseif prod(size(varargin{i})) > 2 % The filter for
zerocross
            H = varargin{i};
         else
            error('Invalid input arguments');
         end
      end

   otherwise
      error('Invalid input arguments');
   end

   if Sigma<=0
      error('Sigma must be positive');
   end

   switch Direction
   case 'both',
      kx = K(1); ky = K(2);
   case 'horizontal',
      kx = 0; ky = 1; % Directionality factor
   case 'vertical',
      kx = 1; ky = 0; % Directionality factor
   otherwise
      error('Unrecognized direction string');
   end


   % if isrgb(I)
   %    error('RGB images are not supported. Call RGB2GRAY
   first.');
   % end
```