Abstract

Dental X-ray image analysis applications are important in helping dentist procedures and diagnosis, and in post-mortem identification. The primary step in such applications is the estimation of teeth contour in order to permit efficient feature extraction (segmentation). The increase in the need for teeth segmentation for example as a result of increase in the volumes of cases that need to be investigated by the forensic specialists brings the need for automation of this process. To obtain the proper result, it is required to perform the accurate and efficient segmentation approach which proved itself in the aspect of X-ray image segmentation.

In this paper, we investigated different teeth contour extraction techniques. These include canny edge detector, Laplacian Gaussian, images differences, thresholding.

Keywords: X-ray images,

1 Introduction

Ante mortem (AM) identification, which is identification prior to death, is usually possible through comparison of many biometric identifiers, post-mortem (PM) identification, that is identification after death, is impossible using behavioural biometrics (e.g. speech, gait) especially under severe circumstances, such as those encountered in mass disasters (e.g. airplane crashers) or if identification is being attempted more than a couple of weeks post-mortem, under such circumstances. Most physiological biometrics may not be employed for identification, because of the decay of soft tissues of the body to unidentifiable states. Teeth, being the hardest and the most impregnable part of human body, cannot easily decay. Dental features are biometric identifiers that are resistant to early decays, hence their survivability and diversity, makes them the best candidates for post-mortem biometric identification. With the evolution in information technology and the huge volume of cases that need to be investigated by forensic specialists, it has become important to automate forensic identification systems.

Dental caries is an infectious, communicable disease resulting in destruction of tooth structure by acid-forming bacteria found in dental plaque, an intra-oral biofilm, in the presence of sugar. The infection results in the loss of tooth minerals beginning with the outer surface of the tooth and can progress through the dentine to the pulp, ultimately compromising the vitality of the tooth. The early detection and characterization of caries lesions is very important because surgical restorative procedures could be reduced. If detected at an early stage, the dentist and dental professionals can implement measures to reverse and control caries, such as identifying patients in need of preventive care, implementing fluoride treatments, implementing plaque control measures, and identifying patients that are at high risk of developing dental caries. Also classification of dental caries is important for the diagnosis and treatment planning of the dental disease. Dental imaging allows dentists to be proactive seeing problems before they become visible. It is also helpful for conducting detailed study and investigations about the nature of the dental disease. Automating the process of analysis of dental images is an important application; it helps the dentist procedures and diagnosis. Tooth segmentation from the radiographic images and feature extraction are essential steps.

2 Related work

Several good progresses have been made for teeth segmentation in the past few years. Jain and Chen [1] proposed a semi-automatic contour extraction method for tooth segmentation by using integral projection and Bayes rule, in which an initial valley gap point is required for applying the integral projection to tooth

Zhou and Abdel-Mottaleb [2] used active contour models (based on snakes) for extracting the contour of the teeth. Active contour models based on edges are driven by the gradient of the image intensities. In many dental images, the gradient between the teeth and the background is not prominent and, hence, such technique is not able to accurately extract the tooth contour. Also, snake-based schemes rely on a parametric representation of the contour. Parametric representations typically fail to evolve in noisy conditions [6]. Further, parametric contours may not be able to split and merge upon encountering local minima in the region of interest ([7], [8], [9], [10]).

Most of these segmentation algorithms are greatly affected by noise embedded in images, which will lead to poor segmentation performance; hence it is crucial to apply any type of image enhancement before applying the segmentation process. Also due to different ways in which dental curies manifest in different types of teeth, it would be important to classify each tooth.

3 Methodology

In this paper we did thorough reviews of the dental X-ray image segmentation techniques, implemented the segmentation techniques and features extraction methods. The comparison of the segmentation methods is split into a number of processes; the input of an image file, the pre-processing (involves cropping the image), enhancement, teeth segmentation using different segmentation techniques. Figure 1. below shows the pictorial view of the methodology together with the proposed techniques or algorithms.

3.1 Cropping

The first step in segmentation is cropping. We created an interface that shows the image and by simply highlighting the part of the image needed, that part is extracted. Figure 2 and 3 below shows an example of an image and the two cropped images respectively. These images will be used throughout this paper.

3.2 Enhancement

Images Are often corrupted by random variants in intensity values such as salt and pepper noise, impulse noise and Gaussian noise. There is a trade-off between edge
strength and noise reduction. More filtering to reduce noise results in a loss of edge strength. In this paper a median filter is used, where each pixel is replaced by the median of its neighbourhood [140].

### 3.3 Detection

Many points in an image have a nonzero value for the gradient, and not all of these points are edges for a particular application. Therefore, some method should be used to determine which points are edge points.

#### 3.3.1 Point Detector

This is achieved by applying a point detecting mask to the image. Each pixel \( g(x,y) \) is given by:

\[
g(x,y) = \sum_{i=-m}^{m} \sum_{j=-n}^{n} w(i,j) f(x+i,y+j)
\]  

where function \( f \) is the initial image, \( w \) is the point detecting mask of size \( n \times m \) and \( g \) is the resultant image.

\[
w = \begin{bmatrix}
-1 & -1 & -1 & -1 & -1 \\
-1 & 1 & 1 & 1 & -1 \\
-1 & -1 & 4 & -1 & -1 \\
-1 & -1 & -1 & -1 & -1 \\
-1 & -1 & -1 & -1 & -1 \\
\end{bmatrix}
\]

It is clear that any pixel at the edge will have a significantly different value as to that of the average of its neighbours. If the value is approximately the same, then the pixel is not at any edge. Figure 4 below shows the result after applying the point detector filter.

A median filter is applied to the image to remove the noise so that the edges are clear. Figure 5 below shows the results after applying the median filter.

![Figure 4: Results of Point detector Filter](image1)

![Figure 5: Point detector after applying median filter](image2)

#### 3.3.2 Laplacian Gaussian

The Laplacian generally is not used in its original form for edge detection due to its unacceptable sensitivity to noise. Consider the function:

\[
h(r) = \frac{r^2}{\sigma^4}
\]

where \( r^2 = x^2 + y^2 \) and \( \sigma \) is the standard deviation. Convolving this function with an image blurs the image, with the degree of blurring being determined by the value of \( \sigma \). The Laplacian of \( h \) (second derivative of \( h \) with respect to \( r \)) is:

\[
\nabla^2 h(r) = \nabla^2 \left( \frac{r^2}{\sigma^4} \right) = \frac{r^4}{\sigma^8} - \frac{4r^2}{\sigma^6}
\]

This is the Laplacian of a Gaussian (LoG). The purpose of the Gaussian function in the LoG formulation is to smooth the image, and the purpose of the Laplacian operator is to provide an image with zero crossing used to establish the location of edges. Smoothing reduces the effect of noise and, in principle, it counters the increased effect of noise caused by the second derivative of the laplacian.

Laplacian of Gaussian first smoothes with a low-pass filter to reduce noise. Laplacian of Gaussian kernel is used for this. Figure 6 below shows Laplacian Gaussian filter.

![Figure 6: Laplacian Gaussian filter](image3)

Figure 7 below shows the results of Laplacian Gaussian filter.

Below are the results after filtering the images.
3.3.3 Double Thresholding

This is one of the most simple segmentation techniques, involves thresholding the image intensities [5]. In this paper this technique is done in two steps, thresholding and calculating gradient to determine edges.

Thresholding

Edge pixels stronger than the high threshold, are marked as strong (white); edge pixels weaker than low threshold are supressed and edges pixels between high and low are marked as weak (grey).

Two threshold values ($T_{\text{high}}$ and $T_{\text{low}}$) are chosen. In this paper, the threshold value ($T_{\text{high}}$ or $T_{\text{low}}$) is the $i^{th}$ pixels value after all the pixels values are sorted in ascending order, where $i$ is given by the equation:

$$i = \alpha \times N$$

(4)

And $\alpha$ in this case is equal 0.4 for $T_{\text{high}}$ and 0.26 for $T_{\text{low}}$. This means that for ($T_{\text{high}}$), the threshold groups 40 percent of the pixels to be below $T_{\text{high}}$ and the other 60 percent will be above $T_{\text{high}}$. The same is done for $T_{\text{low}}$. These threshold values will group the pixels into three subset, those with pixel values above $T_{\text{high}}$, those with pixels values below $T_{\text{low}}$, and those with pixels values below $T_{\text{high}}$ and above $T_{\text{low}}$. These three groups of pixels are given different pixel values, 0 for those below $T_{\text{low}}$, 255 for those with pixel values above $T_{\text{high}}$. The other set is given a value between 0 and 255. Figure 9 below shows the result of double thresholding.

Gradient

From the result shown above, it is clear that the boundaries (edges) can be easily extracted. To determine the edges, gradient magnitude for each pixels is calculated. All the pixels will clearly have the same gradient value of zero, except for those pixels at the edges. Hence any pixels with gradient greater than 0 is categorised as a boundary (is given a value of 1). This draws edges with two pixel thickness.

Gradient is computed for every pixels. The gradient of the image $f(x, y)$ at location $(x, y)$ is defined as the vector

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

(5)

It is well known from vector analysis that the gradient vector points in the direction of maximum rate of change of at coordinates $(x, y)$. The gradient magnitude is the magnitude of this vector,

$$|\nabla f| = \left[ \frac{\partial f}{\partial x}^2 + \frac{\partial f}{\partial y}^2 \right]^{1/2}$$

(6)

In this paper we implemented the Sobel operator to find a first order partial derivative. The Sobel operator computes the approximation of gradients along the horizontal ($x$) and the vertical ($y$) directions for each pixels[6]. Any pixel with a gradient magnitude greater than 0 is classified as an edge.
3.3.4 Image Difference

Gaussian Filter

The image is first smoothed by applying a Gaussian filter with standard deviation of $\sigma = 1.4$. The filter $(B)$ is shown in figure 11 below is a Gaussian filter with $\sigma = 1.4$.

\[
B = \frac{1}{150} \begin{bmatrix}
2 & 4 & 5 & 4 & 2 \\
4 & 9 & 12 & 9 & 4 \\
5 & 12 & 15 & 12 & 5 \\
4 & 9 & 12 & 9 & 4 \\
2 & 4 & 5 & 4 & 2
\end{bmatrix}
\]

Figure 11: Gaussian filter with standard deviation of 1.4

Original image minus filtered image

The smoothed image is subtracted from the original image.

\[
g(x,y) = f(x,y) - h(x,y)
\]

where $g(x,y)$ is the resultant image, $f(x,y)$ is the original image, and $h(x,y)$ is the filtered image. The results are shown in figure 12 below.

Figure 12: Results of subtracting the filtered image from the original image

Applying averaging filter

The resultant image above has too much noise, hence requires filtering the noise. A lowpass (averaging) filter is applied to the image. Each pixel value is replaced by the average of the gray levels in the neighbourhood defined by the filter mask. Because random noise typically consists of sharp transitions in gray values these are reduced, together with false contour. On the other hand the filter has an undesirable effect of blurring the edges, but in this case, the edges will not be much affected because they are already clearly defined. The lowpass filter used in this paper is a weighted averaging filter. Pixels are multiplied by different coefficients, thus giving more importance to some pixels at the expense of others. The weights are given to the filter depending on the distance from the centre. A median filter is then applied to further clean the image. Figure 13 below shows the results of applying the average filter.

Figure 13: Results of applying the average weighted filter

Erosion/Edge thinning

Because the resultant image has thick contour, there is need to reduce the thickness of the edges. This is achieved by applying erosion on the image. For sets $A$ and $B$ in $\mathbb{Z}^2$ the erosion of $A$ by $B$, is defined as
This indicates that the erosion of A by B is a set of all points z such that B, translated by z, is contained in A. The minimum filter was used to to erode the edges. Figure 14 below shows the final results after erosion. It can also be seen that erosion has also removed the noise on the image.

\[ A \ominus B = \{ z \mid (B)_z \subseteq A \} \]  
(9)

### 3.3.5 Canny Detector

This is a multi-step process, implemented as a sequence of filters. The algorithm involves the following steps:

**Noise Filtering**

Blurring the image to remove noise. The image is first smoothed by applying a Gaussian filter. In this paper we used a Gaussian filter with a standard deviation of 1.4, shown in figure 11.

**Gradient**

The edges should be marked where the gradients of the images has large magnitudes (greyscale intensity values changes most). Firstly determined by applying sobel-operator to get the gradient in the x and y direction, then use the law of Pythagoras to determine the resultant gradient. However, the edges are typically broad and thus do not indicate exactly where the edges are. To make it possible to determine this, the direction of the edges must be determined and stored. The direction is determined by the equation below.

\[ \theta = \arctan \left( \frac{S_y}{S_x} \right) \]  
(10)

**Non-maximum suppression**

Only local maxima should be marked as edges. The purpose of this is to convert the blurred edges in the image of the gradient magnitudes to sharp edges, by preserving all local maxima in the gradient image and deleting everything else. Firstly, the gradient direction of each pixel is round off to the nearest 45°. The edge strength of each pixel is compared with the edges strength of the pixels in the positive and the negative directions. If the edge strength of that pixel is the largest, its edge strength is preserved, else its is removed.

**Double thresholding**

Potential edges are determined by thresholding. Many of the resultant edges will probably be true edges in the image, but some maybe as a result of noise. Edges pixels stronger than a chosen high threshold \( T_{high} \), are marked as strong edges(white); edge pixels weaker than low threshold \( T_{low} \) are suppressed and edge pixels between high and low are marked as weak edges (grey). The result is shown below.

**Edge tracking by hysteresis**

Final edges are determined by suppressing all edges that are not connected to a very certain (strong) edge. Strong edges are interpreted as certain edges, weak edges are included if and only if they are connected to strong edges. The logic is that noise and other small variations are unlikely to result in a strong edge. The weak edges can either be due to true edges or noise. Edges due to noise will probably be distributed independently of the edges on the entire image, and thus only a small amount will be distributed adjacent to strong edges. Thresholding above would have labelled valid edges as weak (grey) leaving disconnected edges, usually in regions where the gradient fluctuates between above and below the high threshold value \( T_{high} \). Hysteresis solves the problem by discarding \( (=0) \) all pixels with magnitude less than \( T_{low} \). Pixels with magnitude greater than of equal to \( T_{high} \) are retained \( (=1) \). If a pixel have magnitude between \( T_{low} \) and \( T_{high} \), it is retained if any of its neighbours in the 3 x 3 region around it have gradient magnitude greater than \( T_{high} \). If none of pixels\((x, y)\)s neighbours have a
magnitude greater than \((T_{high})\) but at least one falls between \((T_{low})\) and \((T_{high})\), a search is made in the 5 x 5 region and if any of these pixels have a magnitude greater than \((T_{high})\), it is retained else discarded. Figure below shows the results of hysteresis.

![Figure 16: Results of edge tracking by hysteresis](image)

4 Results and comparison

Canny provides better contour in terms of accuracy. It is however affected by illumination variance as shown in figure 17 below. The contour drawn by this technique is 1(one) pixel thick, but sometimes draws lines which are not part of the tooth.

![Figure 17: Effects of illumination variance, the first image is the original image where the contour has to be detected. The second and third images show the results of contour detection using canny detector and double thresholding respectively](image)

Most teeth have uneven surfaces and since double thresholding have only 2 threshold values, it can only detect at most two ridges on the surface of a tooth, leaving others. Its is not much affected by illumination variance as shown in figure 17 above. For teeth that are too close to each other, this technique in most cases is unable to detect contour separating the teeth. It is also not so accurate, though able to detect the shape of the contour. Laplacian of Gaussian and point detector have very high failure rates.

The table below shows the percentage of failure for each technique we implemented. We define percentage failure as the number of images from where the technique didn’t capture any true segmentation region (tooth) to the total number of images conducted in the experiments.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Canny Success rate</th>
<th>Difference Success rate</th>
<th>Laplacian Success rate</th>
<th>Point Success rate</th>
<th>Thresholding Success rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>41.18%</td>
<td>64.71%</td>
<td>82.35%</td>
<td>82.35%</td>
<td>23.3%</td>
</tr>
</tbody>
</table>

![Figure 18: Percentage failure](image)

5 Future work

Implement a new algorithm to accurately detect dental caries.

References