Comparative Study of Algorithms for Voronoi Diagram Construction on Segmentation of Arabic Hand Writing

Jabril Ramdan and Khairuddin Omar

School of Computer Science, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43200 Bangi, Selangor, Malaysia, Center of Artificial Intelligent.

Abstract: Segmenting Arabic characters is one of the challenging and tedious tasks in the character recognition process. This work proposes an approach to construct Arabic characters segmentation based on Voronoi area. The proposed approach is constructed based on four existing algorithms, such as Voronoi Diagrams (VD), Divided-and-Conquer algorithm (DAC), Half Plane Intersection algorithm (HPI) and Incremental algorithm (I). The VD method performs the segmentation processes by obtaining extracting the line between the connected components, based on the neighbours graph. Whereas the DAC is a fundamental paradigm used for designing efficient algorithms, where the original problem is recursively divided into several simpler sub-problems of approximately equal size. Then the solution of the original problem will be obtained, by merging the solutions of the sub-problems. The HPI algorithm is based on Delauney Triangulation. The VD is constructed by HPI algorithm as follows: i): Connect connect Nearest nearest neighbors and; ii): draw the perpendicular bisector for each Delauney line. Meanwhile, the ‘I’ algorithm calculates the VD by incremental insertion of Voronoi regions, which makes it more efficient and numerically robust to produce the best structures and yield better results in the segmentation process. The proposed algorithm determines the neighbours graph by drawing a line from the centre of the connected components to trace the boundaries of the neighbours in a white background. If there is a gap between the connected components, then they are not considered as neighbours. The Euclidian distance is used as a base to draw line segment between the connected components, which is called as VD. In this research the IFN/ENIT1 dataset will be used. This dataset consists of 569 handwritten Arabic images of the Tunisian towns' names namesof Tunisian towns'. Several experiments has been carried out with the above mentioned base algorithms and compared in terms of time, speed of construction, the number of vertices, and edges. The early results shows that in the static algorithm category, the performance of DAC is promising than the HPI in the static algorithm category, whereas the dynamic I algorithm consumes more time than static algorithms.

Key words: Voronoi Diagram, VD Segmentation, Handwritten word, OCR, neighbours, connected component.

INTRODUCTION

Voronoi Diagram (VD) is one of the finest innovations in the field of mathematics in the 19th century. The 19th century witnessed a technical innovation in mathematics in the form of Voronoi Diagrams (VD). The VD is the fundamental and essential structure arbitrated by the non-statutory schemes. The number of lines isolating the central point, and the points of its neighbours are represented by the VD. In addition, the dividing lines and lines of communication are a pinnacle to one another (Zeki, 2006), (Shatnawi and Omar, 2009). The basic Voronoi concept involves tessellating an m-dimensional space with respect to a finite set of objects by assigning all locations in the space to the closest member of the object set (Barry Boots1 et. al., 2005). The VD is capable of creating a negligible, yet absolute amount of neighbours of an element, which means that only the nearest elements are obtained. The VD of a group of geometric objects is a divider of space from cells. Each of which comprises all the points nearer to a specific object than others. The VD divides the whole space into evenly dislodged subspaces, based on the nearest neighbour rule. For the past few decades, the implementation of VD in a variety of applications is drawing huge attention (Yue Lu et al., 2005).

The line segments that comprise, half lines or infinite lines are called as Voronoi edges and they, constitute the limits of Voronoi regions. These edges are obtained by drawing bisectors at a 90 degree angle to the line joining the two points in the plane. The points that are used in producing Voronoi edge are termed as Voronoi neighbours. The midpoint of ane vacant circle that touching touches three or more sites areis called as Voronoi vertex. Shatnawi and Omar (2009) state that, the convex polygon shaped Voronoi regions are limited or unlimited, Drysdale (1993).
Fig. 1: Voronoi Diagram. pi: site points, q: free point, e: Voronoi edge, v: Voronoi vertex.

VD is widely used in geometrical structures and many other fields, including Astronomy and Biology. Based on Mady and Omar (2011) the states that, sampling intervals $R$ are crucial points in the construction of VD. It plays a fundamental role in the VD construction performance. Based on the comparative study using different sampling intervals of $R$, the divide-and-conquer algorithm DAC is faster than the half-plane intersection HPI and incremental algorithms I for VD construction.

The character recognition systems fall into two categories such as online and off-line systems. In the online category, the system recognizes the symbols during the course of its insertion, whereas in the offline category the recognition is done after writing or printing processes are completed, resulting in the chronological loss of information of the text. The offline systems contain another drawback, as they have to deal with low resolution documents. Basically, the online systems have an edge of advantage over the offline in terms of simplicity, Amin (1997).

Character recognition systems differ widely in how they acquire their input (online versus offline), in the mode of writing (handwritten versus machine printed), in the restriction on the fonts (single font versus many multiple fonts), and in the connectivity of text (isolated characters versus cursive words). Apart from these typographic features, the character segmentation and classification of Arabic characters, also depends on the contextual information (Hamid 2001). According to Amin (2002), the segmentation problem is not an exclusive issue for computers, but also to humans. The humans approximately have a 4% error rate in reading the Arab script, in the absence of context. The inaccuracy in reading happens due to the differences in shapes, based on behavior, approach, education, social surroundings, physical condition, psychological state and other conditions affecting the writer. The other elements that add on to the above mentioned problem are: writing tools, platforms, scanning softwares and machine recognition algorithms (Amin & Mari 1989).

As mentioned earlier the segmentation is the most tedious, protracted, yet crucial phase that represents the challenges of the Arabic Character Recognition (ACR) system. This challenge is due to the complexity of the image shapes. The existing algorithms to produce VD are sluggish, and hence it is vital to introduce a fast phased VD algorithm. Hence most accepted static and dynamic styles of algorithm development have been selected, in order to find the optimal algorithm that is ideal and produce the best structures, that which can produce best results on the segmentation process. The styles are: It divided-and-conquer algorithm (DAC), half plane intersection algorithm (HPI), and Incremental algorithm (I). In order to perform the segmentation process, In this work for segmentation processing, is needs to determine neighbours of connected components, called as the neighbour graph. Generally, the ACR systems have problems in determining the neighbours’ graph.

A methodology will be utilized, to select the best algorithm, calculating the time spent in the construction of the VD and to calculating the number of polygons. When the construction time and polygon numbers are the lowest, the algorithm will be chosen as the best, and, utilize a The proposed algorithm to will be utilized to determine the neighbours for making line segment between areas of connected components. Then to and use a propose algorithm for segmentation segmenting of Arabic hand writing corresponding to neighbours and minimum distance between connected components by using VD.

2. Current Research of VD:

No matter which algorithm is used in the segmentation, it must be derived based on the nature of cursive connection in Arabic text (Al-Waily 1989). Hence, an in-depth understanding of the characteristics of the Arabic script is necessary for the development of an OCR system. This knowledge helps in selecting the best technique to be used and may also lead to the development of new techniques (Amin & Mari 1989; Cheung et al. 2001; Zeki 2005). A variety of algorithms have been proposed to construct VD and DT. Lafon and Blanc (1997) classified the construction methods into two main categories as static and dynamic algorithms.

Static algorithms: In this category, all generators must be known in advance. The divide-and-conquer algorithm of Shamos and Hoey (1975) is a typical method of this type, in which, the set of generators is recursively divided into smaller subsets. The VDs for those subsets are merged into the final diagram.
Dynamic algorithms: In this category, generators are not known in advance. The incremental method, for instance, starts with a simple VD for a few generators and modifies the diagram by adding other generators one by one (Green and Sibson 1978). The incremental method is probably the most popular one due to its relative ease of implementation. It is most practical in terms of time complexity and the robustness (Okabe et al. 1992).

Plane-sweep algorithm is another dynamic algorithm (Fortune 1987) in which, a vertical line is moved over the plane from left to right and the VD is constructed along the line.

Optimal algorithms for Voronoi tessellation have a worst-case time complexity of $O(n \log n)$ and $O(n)$ on the average, where $n$ is the number of generators (Preparata and Shamos 1988). Parallel algorithms were also proposed (Berman and Lingas 1997; Ju and Gunzburger 2002; Kühn 2001). The above DT definition was built on VD. However, it does not mean that a DT should always be defined with a VD. In fact, DT can be defined without a VD (Sohler 2005; Su and Drysdale 1995; Žalik 2005).

3. Proposed Segmentation Method of VD Using Neighborhood:

A lot of algorithms have been formulated to generate planar VD from the collections of point sets. Surveys of established methods such as divided-and-conquer algorithm, half plane intersection algorithm, Incremental algorithm, Bowyer- Watson algorithm, Fortune's algorithm and Lloyd's algorithm, appear in Franz Aurenhammer (1991). For example divided-and-conquer algorithm, half plane intersection algorithm, Incremental algorithm, Bowyer- Watson algorithm, Fortune's algorithm and Lloyd's algorithm. In this section, the research will present a brief explanation of the common three algorithms of VD construction. In this work we have developed a method to depict the physical constitution of a word; the method employs the neighbour graph, which symbolizes neighbours of linked components. The deploying of the area VD enables the extraction of neighbours with preset labels. Area-VD draws line segments between the connected components. The complete set of those lines will separate the connected components. We have used Euclidean distance to compute the minimum distance between connected components or neighbours, in order to find the edge of segment.

![Fig. 2: The Architecture of the work.](image)

3.1. Divide-and-Conquer Algorithm:

Shamost and Hoey (1975) have presented this algorithm in their approach, in which the generator points ($S$) are divided into two groups, left $S(L)$ and right $S(R)$ of the same size. Then the VDs of both are computed, with left sets $V(L)$, and right sets $V(R)$. Finally, in the merge step it computes the set $B(R,L)$ of all Voronoi edges of $V(S)$. In order to apply this algorithm, several successive stages would be followed:
First Stage:
Points are sorted in ascending order by the \(x\)-coordinates, to be easily divided into two parts, left and right.
\[
S = \{p_1, p_2, \ldots, p_n\} \quad (1)
\]
where, \(S\) is the set of generator points, and \(n\) is the number of points.
\[
T = \frac{n}{2} \quad (2)
\]
So \(S(L) = \{p_1, \ldots, p_T\}\) \quad (3) \\
\(S(R) = \{p_{T+1}, \ldots, p_n\}\) \quad (4)

Second Stage:
In this stage, the generator points are sorted and divided into two groups. Now the VD of each group \(V(L)\), \(V(R)\) is constructed by using any known algorithm.

Third Stage:
It is a stage of integrating the two diagrams to produce the final diagram; this is one of the most important stages. This algorithm is significant from a theoretical standpoint not only because it was the first one, but also because it uses the divide-and-conquer paradigm. The divide-and-conquer is one of the fundamental paradigms for designing efficient algorithms.

According to Shamost and Hoey (1975), to compute integration, the Convex Hull of each side (left, right) is found. Then the lower and upper segment lines that join the left and right Convex Hull are found. The divide-and-conquer algorithm for the Convex Hull was published by Preparata and Hong (1977). The algorithm starts with an unordered set of points defined by Cartesian coordinates - each point has a position on both the \(X\)- and \(Y\)-axes.
Sorting the remaining points into the upper and lower sets requires some function that determines whether a point is above or below a line. Given a set of points on the line, \( P_0, P_1 \) and \( P_2 \) when \( P_0 \) is at 0,0 then take the determinant of \( P_0, P_1 \) and \( P_2 \). The resulting value will be negative if \( P_2 \) angled off in the left direction, positive if it has moved to the right and 0 if it is collinear with the first two points.

\[
A = \frac{(p0_x - p1_x)(p2_x - p1_x)}{(p0_y - p1_y)(p2_y - p1_y)}
\]  

(5)

For this 2x2 matrix, the formula for the determinant will be:

\[
dt = ((p0_x - p1_x) * p2_y - p1_y) - ((p2_x - p1_x) * (p0_y - p1_y)).
\]  

(6)

The resulting partitions will look like the ones shown below in Figure 5.

![Figure 6: The upper or lower hull.](image1)

The upper or lower hull is started by simply adding left to the output Convex Hull. Points are then added from the correct input source. While each point is added, if the number of points in the working Convex Hull is equal to three points or more, a test is made to check if the last three points have created a convex angle. Testing for the convex angle is done using the same determinant formula as shown above. If the Convex Hull has points, a test is done to see if \( P_{n-1} \) is above or below the line formed by \( P_{n-2} \) and \( P_n \). If it is realized that the point is above the line when constructing the lower Convex Hull, the convexity is violated and the middle point is removed from the Convex Hull. The opposite test is made when constructing the upper Convex Hull. Shamos and Hoey (1975) state that this test-and-remove process is repeated until the last three points are convex, or if there are fewer than three points in the working Convex Hull.

![Figure 7: Convex hull H(L), H(R).](image2)

Finally, after constructing the convex hull for each side \( H(L), H(R) \), it is necessary to find the connector line of \( H(L) \) and \( H(R) \) from up and down. As shown in the following Figure 8.
Trace the perpendicular bisector of the lower bridge from $-\infty$ and find the lowest intersection point with an edge of the left or right VD. The left and right VD are shown overlaid on top of one another in the following Figure 9.

3.2. Half-Plane Intersection Algorithm:

While the previous algorithm is based on the convex hull; however, the half-plane intersection algorithm is based on Delauney Triangulation (DT). In order to construct VD by this algorithm the following steps must be followed:

**Step One:**

Connect Nearest Neighbors (DT): DT is a technique of connecting the sets with their nearest neighbours without making a line that crosses another. This step is the most difficult part. One way to do it might be, by brute force – which connects every set with other sets, make a true all-to-all and then start deleting the lines which are too long. Another way is to start, with a test triangle of sets and draw their circumscribes that hits all the three sets, rejecting the triangle, if the circle contains another set within it.
Fig. 11: Delauney Triangulation.

**Step Two:**
Draw the perpendicular bisector for each Delauney line. In this step, the mid-point is identified for each Delauney line, by using this equation:

\[
\text{Mid-Point}(x,y) = \left(\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2}\right).
\]  
(7)

Through the midpoint, the perpendicular bisector is drawn, and the intersection point of bisectors is found by this equation:

\[
P(x,y) = \frac{(x_1y_2 - y_1x_2)(x_3 - x_4) - (x_1 - x_2)(x_3y_4 - y_3x_4)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)}.
\]  
(8)

The following figure illustrates the construction steps:

Fig. 12: Half-plane intersection algorithms.
3.3. Incremental Algorithm:

Green and Sibson (1978) calculated the VD by incremental insertion i.e. they obtained $V(S)$ from $V(S_{\{s\}})$ by inserting the set $s$. The region of $s$ can have up to $(n-1)$ edges, for $n=|s|$. Ohya and Murota (1984) have polished up the technique of inserting Voronoi regions, and made the “incremental algorithm” to be efficient and numerically robust. The incremental algorithm is set up with a simple VD for two or three points, and modifies the diagram by adding the sets one by one. For $P=1,2,\ldots,n$, let $V_P$ denote the Voronoi diagram for the first $p$ sets $s_1,s_2,\ldots,s_p$. The major part of this algorithm is to convert $V_{p-1}$ to $V_p$ for each $P$.

The basic idea in the algorithm of incremental VD is as follows: Suppose that the Voronoi diagram has been built. The $V_{p-1}$ is shown by solid lines (figure 13), and a new set $s_p$ is added. First, the set, $s_i$, is found and its Voronoi polygon contains $s_p$, and the perpendicular bisector between $s_j$ and $s_i$ is drawn denoted by $B(s_p,s_i)$. The bisector crosses the boundary of $V(s_i)$ at two points $x_1$ and $x_2$. Set $s_p$ to the left of the directed line segment $x_1x_2$.

The line segment $x_1x_2$ divides the Voronoi polygon $V(s_i)$ into two parts, the one on the left belonging to the Voronoi polygon of $s_i$. Thus, a Voronoi edge on the boundary of the Voronoi polygon of $s_i$ is obtained.

Fig. 13: Construction of point-VD using Half-plane intersection algorithm.

Fig. 14: Adding a new point to Voronoi diagrams.

Starting with the edge $x_1x_2$, the boundary of the Voronoi polygon of $s_i$ is expanded by the following procedure.: The $B(s_i,s_j)$ crosses the boundary of $V(s_j)$ at $x_3$, entering the adjacent Voronoi polygon, say $V(s_j)$. Therefore, the $B(s_i,s_j)$ is drawn and the point, $x_3$ is found at which the bisector crosses the boundary of $V(s_j)$. Similarly, the sequence of segments of perpendicular bisectors of $s$ and the neighboring sets are found until the starting point $x_i$ is reached. Let this sequence be $(x_1x_2, x_2x_3, \ldots, x_{m-1}x_m, x_{m+1})$. This sequence forms a counter clockwise boundary of the Voronoi polygon of the new set $s$. Finally, the substructure inside the new Voronoi polygon is deleted from $V_{p-1}$, resulting in $V_p$. 

1660
3.4 Determine of Neighbours:

According Kise et al., (1998b), identified the centers of connected components then draw a line from center to center for to determine the neighborhoods, this is using used for printed of Latin characters. But based on figure 16 CentersCentres of connected components are first identified from the green circle then a line is drawn to trace boundaries of each connected component, if there is the a blue line,. It it means that the background has white space on the boundary of the components, then it is considered as a neighbour,. Like like the components one and two or two, three and four.

If the line has two colors namely blue and red then they are not neighbours, because some or all other boundary points traverse between them. As in the case of components one and three, a part of the component two is in between them. And in components four and five a part from the component four is in between them. Hence the components one and three are not neighbors, as well as four and five. Shown figure 17. Figure 17 illustrates the neighbours by colors.

The area VD can be interpreted as the representation of neighbours of connected components. As mentioned previously, the following algorithm is applied to determine the neighbours: (the algorithm is missing).
3.5. Construction of Area-VD:

Area-VD draws line segments between the connected components. The complete set of those lines will separate the connected components. We have used Euclidian distance to compute the minimum distance between connected components, in order to find the line segment separating it. And we have used the new algorithm to determine neighbours, because the Euclidian distance measurement is not capable of providing the better result, especially in the case of unknown Arabic handwriting characters neighbours. The constructed area-VD is shown in the result of Figure 22, 23 and 24. Although the edges of the area-VD seem to be smooth curves in some areas, in fact they are made of small line segments between connected components.

RESULTS AND DISCUSSION

Each stage in a multi-stage system has an impact on the final result. Hence, measuring the performance of each stage becomes necessary. A number of experiments are were done in order to measure the performance of the developed methods on different types of input data. Will describe the results of each of those methods. They will be tested on various types of data, e.g. Handwritten documents and, different fonts and sizes. In this work we have used IFN/ENIT data set which includes 569 handwritten Arabic images. These images represent the names of Tunisian towns. Based on the studies by Pechwitz and Maergner (2002) this dataset is a demo version of the IFN/ENIT database. The names of the towns are of maximum three words, and might be having some related sub-words.

4.1. Pre-Processing:

The pre-processing stages to be discussed below include edge detection block, and sampling points.

4.1.1 Edge Detection:

The edges of the objects in the images are detected by edge detection block. This block identifies the locations of pixel, where the degree of the gradient intensity is high. As explained earlier these locations will exist in the boundaries of objects.

4.1.2 Sampling Points:

The building of VD is time-consuming. On the average by choosing the sampling interval \( R \) to sampling process, the process time is reduced and the time consumed by all boundary points are used to generate VD. For example, Table 1 displays, the time consumed to process the image shown in Figures 10, 13, and 16. The processing is done using different sampling interval values \( R \). The table 1 also shows, the incredible decrease in time consumed when the interval sample is \( R = 10 \). After all the boundary points are used (when \( R = 1 \)), the processing time is 0.0406 sec in the divide-and-conquer algorithm, 0.0423 sec in half-plane intersection algorithm and 0.0450 sec in the incremental algorithm. Whereas when \( R = 10 \), the processing time becomes 0.0021 sec in the divide-and-conquer algorithm, 0.0023 sec in the half-plane intersection algorithm and 0.0030 sec in the incremental algorithm. Relatively the processing time is reduced by 94.82% in the divide-and-conquer algorithm, 94.56% in the half-plane intersection algorithm, and 93.33% in the incremental algorithm.

<table>
<thead>
<tr>
<th>Sampling interval ( R )</th>
<th>Divide-and-conquer algorithm</th>
<th>Half-plane intersection algorithm</th>
<th>Incremental algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0406</td>
<td>0.0423</td>
<td>0.0450</td>
</tr>
<tr>
<td>2</td>
<td>0.0054</td>
<td>0.0054</td>
<td>0.0063</td>
</tr>
<tr>
<td>3</td>
<td>0.0034</td>
<td>0.0035</td>
<td>0.0044</td>
</tr>
<tr>
<td>4</td>
<td>0.0030</td>
<td>0.0028</td>
<td>0.0034</td>
</tr>
<tr>
<td>5</td>
<td>0.0030</td>
<td>0.0032</td>
<td>0.0035</td>
</tr>
<tr>
<td>6</td>
<td>0.0026</td>
<td>0.0029</td>
<td>0.0029</td>
</tr>
<tr>
<td>7</td>
<td>0.0029</td>
<td>0.0026</td>
<td>0.0031</td>
</tr>
<tr>
<td>8</td>
<td>0.0026</td>
<td>0.0021</td>
<td>0.0030</td>
</tr>
<tr>
<td>9</td>
<td>0.0023</td>
<td>0.0024</td>
<td>0.0030</td>
</tr>
<tr>
<td>10</td>
<td>0.0021</td>
<td>0.0023</td>
<td>0.0030</td>
</tr>
</tbody>
</table>

The table 2 shows, the total average processing time consumed by each stage, for the segmentation of 100 images from the IFN/ENIT dataset, using various sampling intervals \( R \), whereas, the Table 2 also shows the percentage of the time comparatively consumed using sampling intervals \( R \). It is apparent that, more than 30% of the total computation time, was consumed by the point-VD construction. However, segment built based on area-VD does not use much computational time.
Table 2: Average processing time taken by each stage of the VD construction and segmentation of the 569 images of the IFN/ENIT dataset using different sampling intervals $R$.

<table>
<thead>
<tr>
<th>Sampling interval ($R$)</th>
<th>Image acquisition, edge detection time (sec)</th>
<th>Sampling process time (sec)</th>
<th>Time needed to construct point-VD using divide-and-conquer algorithm (sec)</th>
<th>Time needed to construct point-VD using half-plane intersection algorithm (sec)</th>
<th>Time needed to construct point-VD using incremental algorithm (sec)</th>
<th>Time needed to Segment the word (sec)</th>
<th>Total time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.2908</td>
<td>0.0035</td>
<td>0.0406</td>
<td>0.0423</td>
<td>0.0450</td>
<td>0.4855</td>
<td>0.9077</td>
</tr>
<tr>
<td>2</td>
<td>0.0325</td>
<td>0.0029</td>
<td>0.0054</td>
<td>0.0054</td>
<td>0.0063</td>
<td>0.8685</td>
<td>0.9210</td>
</tr>
<tr>
<td>3</td>
<td>0.0078</td>
<td>0.0033</td>
<td>0.0034</td>
<td>0.0035</td>
<td>0.0044</td>
<td>0.9821</td>
<td>1.0045</td>
</tr>
<tr>
<td>4</td>
<td>0.0098</td>
<td>0.0052</td>
<td>0.0030</td>
<td>0.0026</td>
<td>0.0034</td>
<td>0.6346</td>
<td>0.6588</td>
</tr>
<tr>
<td>5</td>
<td>0.0085</td>
<td>0.0039</td>
<td>0.0030</td>
<td>0.0032</td>
<td>0.0035</td>
<td>0.5256</td>
<td>0.5477</td>
</tr>
<tr>
<td>6</td>
<td>0.0087</td>
<td>0.0042</td>
<td>0.0026</td>
<td>0.0027</td>
<td>0.0027</td>
<td>0.4996</td>
<td>0.5209</td>
</tr>
<tr>
<td>7</td>
<td>0.0089</td>
<td>0.0038</td>
<td>0.0029</td>
<td>0.0026</td>
<td>0.0031</td>
<td>0.6731</td>
<td>0.6944</td>
</tr>
<tr>
<td>8</td>
<td>0.0139</td>
<td>0.0037</td>
<td>0.0026</td>
<td>0.0021</td>
<td>0.0030</td>
<td>0.4560</td>
<td>0.4813</td>
</tr>
<tr>
<td>9</td>
<td>0.0135</td>
<td>0.0042</td>
<td>0.0023</td>
<td>0.0024</td>
<td>0.0030</td>
<td>0.3739</td>
<td>0.3993</td>
</tr>
<tr>
<td>10</td>
<td>0.0137</td>
<td>0.0048</td>
<td>0.0021</td>
<td>0.0023</td>
<td>0.0030</td>
<td>0.5528</td>
<td>0.5787</td>
</tr>
</tbody>
</table>

Fig. 18: The impact of the sampling interval on the speed.

Fig. 19: The average processing time taken by each stage of the VD construction and Segment in 100 images of the IFN/ENIT dataset using different sampling intervals $R$.

Fig. 20: Edge detection.
Fig. 21: Sampling.

Fig. 22: Construction of point-VD.

Fig. 23: Area-VD for one word.

Fig. 24: Area-VD two words.
Fig. 25: Area-VD three words.

Include obstacles for segmentation on some picture like figure 26 and 27, the line segment not draw among connected components or draw out middle distance between them, this due to the existence of noise in images.

Fig. 26: Area-VD two words.

Fig. 27: Area-VD two words.

**Conclusions:**

The comparative study was tested and validated using different sampling intervals $R$ on the IFN/ENIT handwritten dataset. Results obtained were shown in graphs and output images. With the implementation of segmentation algorithms using VD as discussed, the objectives of the study have been fulfilled. Sampling intervals $R$ are crucial points in the construction of VD. It plays a fundamental role in the VD construction performance. Applying the comparative study using different sampling intervals of $R$ produces different shapes. The experimental result proved that, the comparative study while using the sampling interval $R=5$ has produced the vertex number of 620 vertices in the divide and conquer algorithm and took. It need 0.0030 seconds, compared with the other algorithms we have obtained 653 vertices in the half-plane intersection and. It took 0.0032 seconds and 783 vertices points whereas, in the incremental algorithm. It took 0.0035 seconds. Through the results obtained, and as is evident in the tables Based on the tabulated results, it is evident that the divide-and-conquer algorithm is faster than the half-plane intersection and incremental algorithms for VD construction. These percentages are will be different depending on your based on the computers and the programming languages used. Hence terms of speed and time gave the result of implementation illustrated in this study are not of fixed.
The experimental results also proved that the processing time for the segmentation algorithm depends on the size of images and the sampling interval $R$. WE had used Area-VD was used to separate all connected components (i.e. subwords and secondaries). The method is perfect in separating those components. The time consumed is was optimized by setting the sampling interval $R$ value. It is the method of its type that uses computational geometry to separate Arabic connected components. All advantages of geometrical methods are gained here. For example, such as the capability of Area-VD to represent the neighbourhood of connected components as polygons. The method based on area-VD proved to be perfect in separating the Arabic subwords and secondaries.

REFERENCES


Amin, A., 2002. Structural description to recognising Arabic characters using decision tree learning techniques. Joint IAPR International Workshops on Structural, Syntactic, and Statistical Pattern Recognition (SSPR’02 and SPR’02), Windsor, Ontario, Canada. 6-9 August, 152-158.


