Weighted-function based Algorithm for Retrieving Handwriting Trajectory from Off-line Data

Nermina Ahmic, Emir Sokic, Melita Ahic-Djokic
Faculty of Electrical Engineering, University of Sarajevo
Bosnia and Herzegovina
{nahmic1, esokic, amelita}@etf.unsa.ba

***Abstract*—Individuality of handwriting is the reason why it is used as a common base element for detecting character traits of the writer. It is believed that dynamic information improve the accuracy of the analysis, but they are not contained in an off-line handwritten text. In order to recover dynamic information, a novel approach for handwriting trajectory recovery is proposed in this paper. The procedure is based on computing the objective function, which depends on parameters such as the angle of movement, path length, air pen tip movements, etc. The analysis is performed in MATLAB program package, using the text samples from IAM-OnDB database. The experimental results indicate that the average effectiveness of the proposed algorithm is above 75%.**

I. Introduction

Handwriting is a unique characteristic of human personality, as it is a result of subconscious or a habit, just like other body movements. The information about psychological condition and emotions of a person who is writing are incorporated in handwritten text by hand movements which enable transmitting electrical impulses from brain to the writing hand. Despite the handwriting uniqueness, there are some features that can be used to determine the gender [1]–[5], handedness [3], or even age and nationality of the writer [4].

Most common attributes for gender determination are writing speed, carefulness, neatness, size of the letters, number of breaks, margins, spacing between the words and regularity of written words or letters [5], [6], which can easily be extracted if the on-line handwriting data is available. On the other hand, off-line analysis explores only the image of a handwritten text and cannot provide dynamic information that are used in algorithms for detecting personal traits from on-line handwriting, such as writing speed and direction, curvature and log curvature radius, speed and acceleration in x- and y-direction, or overall acceleration [3]. In order to enhance the results of off-line text analysis, recovery of the dynamic information is needed. The most valuable information that can be recovered are pen pressure [7] and stroke order [8].

Handwriting strokes are mostly analysed in the algorithms for handwriting recognition based on the properties of the basic strokes that have been used to generate a character [9] and in signature verification algorithms that analyses idiosyncratic features of the strokes to characterize a signer and check his identity [10]. Moreover, stroke analysis is performed in neuroscience to characterize neurodegenerative processes like Alzheimer [11] and Parkinson disease [12].

Mentioned studies usually conduct their analysis over the static images of the handwritten text, captured with a scanner or a camera device. The absence of dynamic information causes lower accuracy of off-line recognition systems in comparison with their on-line counterparts [9]. Psychology researches suggest that the character recognition can be improved by the humans' perception of dynamic information from static images [13]. Authors in [14] claim that the proper trajectories recovery could significantly improve the performance of automatic off-line handwriting recognition systems. The most common approach used in papers related to recovery of the stroke driving order is following continuity criteria, which takes into account direction, length, and width of the strokes [15], [16]. In contrast, stroke order recovery can be made by searching a graph using the global and the local criteria [17] and by applying a Kalman filter on the analysed static image to follow the stroke trajectory [18]. Recovering process is usually done on the single-stroke handwriting image [16], [19], or on single characters (allographs), although it is shown that handwritten words carry more individuality than most allographs [20].

In this paper, a novel approach to recover a handwritten text trajectory from a static image is proposed, based on analysing handwritten words instead of characters or single-stroke written text. This method is based on weighted objective function that includes parameters relevant for the trajectory recovery and is somewhat similar to [15], but includes more computational parameters, such as path jumpiness and angle of pen tip movement.

The paper is organized as follows: The procedure of recovering the pen tip trajectory from a static image is described in Section II. Section III presents the experimental results. Conclusions and guidelines for future work are given in the last section.

II. Proposed algorithm

Quality of the handwriting static image largely determines the performance of recovering writing trajectory. In order to extract meaningful information, the image needs to be processed prior to handwriting trajectory analysis, throughout two steps: preprocessing and segmentation. Afterwards, trajectory extraction can be performed.

The proposed algorithm is implemented using MATLAB software. It is developed for the purposes of analysing any

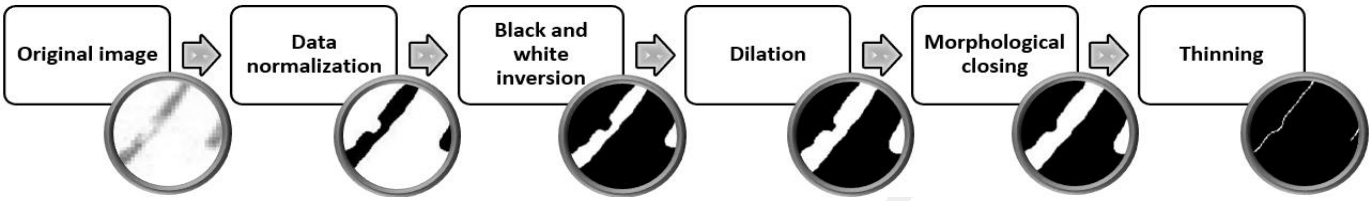

Figure 1: Image preprocessing steps with corresponding examples of the results obtained by analysing an off-line image.

type of scanned images and is verified using the images obtained from the IAM On-line Handwriting Database (IAM-OnDB) [21]. This database consists of XML files. Each word can be plotted as a single figure by linearly connecting all the points recorded in the file. For the sake of testing the algorithm, the figures were saved as images.

A. Image processing

1) Preprocessing: Images that are scanned or captured with the camera device are often degraded by noise and need to be processed in order to enhance their quality and recover the desired information. Preprocessing includes data normalization, noise reduction and morphological image processing. Due to the usually small dimensions of analysed images, the first step of the proposed algorithm is to resize the image to larger resolution for ensuring better performance.

Data normalization is done by adjusting the contrast, which enables easier noise reduction. Images obtained from on-line databases are not occupied by noise, but it is a common problem for off-line data.

For the purpose of noise reduction, techniques of adaptive thresholding and average filtering are used. Afterwards, conversion from grayscale to black and white image and color inversion are done. This procedure is conducted in order to simplify stroke extraction from the image.

Two consequent morphological operations are applied on the image, using disk-shaped structuring element with a radius equal to 1. Dilation is done for the object thickening. This operation results in bolded handwritten text and it is particularly important if the text is written with a graphite pencil or is very degraded by the scanning process. Drawback of using this operation is that it can result in small loops when thickened letters include small hollows, such as the letter e. Afterwards, filling in the object contour was performed using morphological closing operation. This operation helps recovering the continuous trajectory of handwritten text that was degraded by scanning.

The final step in preprocessing is thinning the contours of the object presented in an image. Thinning, or skeletonization process, was done using the improved Zhang-Suen algorithm described in [22]. Resulting image represents the approximation of original pen tip trajectory. Artefacts, such as loops, are rarely present due to the image resizing made at the beginning of the procedure. Preprocessing steps with corresponding examples of results are shown in Fig. 1.

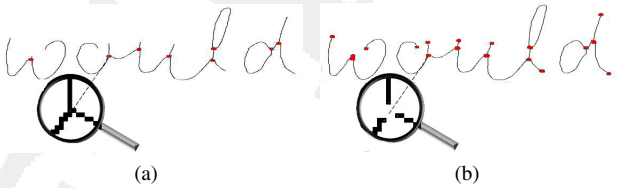

Figure 2: Segmentation results. (a) Branching points detection; (b) Branching points separation and detection of all characteristic points.

2) Segmentation: The most important process in text segmentation is the detection of the characteristic points. There are two types of points that need to be identified: endpoints and branching points. The term of branching point refers to an intersecting point of at least two writing segments. Endpoint is a white pixel (a point that is a part of an illustrated object) whose all neighboring pixels except for one belong to the background of the image.

Branching points detection was done by singling out 3×3 matrices and checking if they match one of the created branching patterns. If there is a pattern matching, the central pixel of analysed matrix is marked as a branching point, as shown in Fig. 2 (a).

Handwritten text contour has to be separated into multiple segments in order to predict the trajectory of the writing. Separation of intersected segments is easily implemented by inverting the color of each branching point and erasing its closest neighboring points of the same color, as shown in Fig. 2 (b). This process is followed by detection of newly formed segment endpoints. Endpoints detection is made by singling out 5×5 matrices around the pixels that represent the branching points. Possible endpoints are positioned in the first or the last row and in the first or the last column of these matrices. If a row or a column includes more than one pixel that belongs to an object, the one closer to the center of matrix is chosen as an endpoint. The original contour endpoints are detected by summing all the elements of each 3×3 singled out matrix with verification of the sum result. The summation result equal to two implicates the existence of only one neighboring point that is a part of an object and a central point of the matrix is detected as an endpoint.

Line detection is done by fetching all the white pixels between each pair of endpoints of contour segments. Each of the N segments $\{s_j\}, j = 1, ..., N$ of the contour has a defined set of four attributes $\{a_j^{(1)}, a_j^{(2)}, a_j^{(3)}, a_j^{(4)}\}$ and a set of

points $\{p_{j,k}\}, k = 1, ..., M_j$, where M_j is the total number of points of j-th segment and $p_{j,k} = \{y_{pj,k}, x_{pj,k}\}$, where $y_{pj,k}$ is a value of the ordinate and $x_{pj,k}$ is a value of the abscissa for k-th point of j-th segment. The attribute $a_j^{(1)}$ indicates a total length of the segment, computed according to relation (1), while $a_j^{(2)}$ stores the information about center of the mass in an ordered pair $\{y_{cj}, x_{cj}\}$, where y_{cj} and x_{cj} are values on ordinate and abscissa axis of the center of the mass for the j-th segment, respectively.

$$a_j^{(1)} = \sum_{k=2}^{M_j} \sqrt{(x_{pj,k} - x_{pj,k-1})^2 + (y_{pj,k} - y_{pj,k-1})^2} \quad (1)$$

The information about coordinates and branching of the characteristic points are stored in an attribute $a_j^{(3)}$, where $a_{j,1}^{(3)}$ and $a_{j,2}^{(3)}$ contain pairs of point positions $\{y_{j,1}^{(3)}, x_{j,1}^{(3)}\}$ and $\{y_{j,2}^{(3)}, x_{j,2}^{(3)}\}$, respectively, in an analogous way as $a_j^{(2)}$, while the relation (2) is applied to $a_{j,l}^{(2)}$, where $l = 3, 4$.

$$a_{j,l}^{(3)} = \begin{cases} 1, & \text{if } a_{j,l-2}^{(2)} \text{ is branching} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

Let the segment endpoints be numbered as 1 and 2. Possible movement direction schemes are: 1-2, 2-1 (non-repeating trajectories), 1-2-1 and 2-1-2 (repeating trajectories). The attribute a_4 contains a set of points for each of the listed paths.

Lines (segments) of contour are sorted in the ascending order according to the attributes $a_j^{(2)}$ and $a_j^{(3)}$. All of the listed attributes are then used in process of trajectory extraction.

B. Trajectory extraction

Computation of the objective function is done for each possible order of R successive segments and for each possible movement direction by the same. After analysing the first set of R segments $\{s_1, s_2, ..., s_P, s_{P+1}, ..., s_R\}$, the other R are singled out starting from the q-th one, where $q = P + m(R - P)$ and m is a number of iteration ($m = 0, 1, ..., M - 1$). For the second iteration ($m = 1$), the objective function is computed on a set $\{s_P, s_{P+1}, ..., s_R, s_{R+1}, ..., s_{P+R-1}\}$ and a procedure of singling out the segments is carried out until the total number of N contour segments is reached. If the written word consists of smaller number of segments, or if there are less than R segments left in the last iteration, the greatest number of available segments is analysed. Value of the objective function in the m-th iteration is given in (3).

$$f_m = f(s_{m(R-P)+1}, ..., s_{m(R-P)+R}) \quad (3)$$

Possible orders of segments are determined by computing all permutations on R isolated segments. The reduction of number of possibilities is made by excluding those permutations that contain the orders with difference between the indices of two neighboring segments greater than 4. This number is chosen because the empirical results show that one letter usually does not consist more than 5 segments. Moreover, they indicate that the optimal values are $R = 5$ and $P = 3$.

Movement directions are determined by computing all combinations with repetition of the 4 schemes with repeating and non-repeating trajectories. In order to reduce the running time of the algorithm, combinations with at least two repeating trajectories in a row are not considered, because of the small possibility of repeating one point for 4 times when writing.

There are six factors relevant for choosing the right trajectory. They are given as follows:

- w_1: position of a starting point of the contour on the x-axis,
- w_2: boolean value of statement that the starting point of the text (word) is a branching point,
- w_3: angle of movement through the branching point,
- w_4: path length,
- w_5: path jumpiness (transitions between remote segments),
- w_6: air pen tip movements.

Factor w_1 considers position of both endpoints of the segment, relative to the x-axis, as in (4).

$$w_1 = \sum_{\text{j=m(R-P)+1}}^{\text{m(R-P)+R}} v_{1,1} x_{cj} + \frac{v_{1,2} \times v_{1,3}}{g(p_{j,1}, p_{j,M_j})} \quad (4)$$

where $g(p_{j,1}, p_{j,M_j}) = \sqrt{(x_{pj,1} - x_{pj,M_j})^2 + (y_{pj,1} - y_{pj,M_j})^2}$. Multiplicators $v_{1,1}$ and $v_{1,2}$ are chosen from a set of two positive constants, as noted in Table I. A lower value of $v_{1,1}$ is chosen when $a_j^{(2)} < a_{j+1}^{(2)}$, while a higher value from the set is chosen otherwise. The factor $v_{1,2}$ is chosen from the set in a similar fashion, according to the inequality $x_{pj,1} > x_{pj,M_j}$. Multiplicator $v_{1,3}$ is equal to one for the starting segment of the contour and is zero otherwise. Factor w_1 has lower value for the point closer to the coordinate origin. This is equivalent to positioning the starting point more to the left, because writing direction of Latin alphabet is from left to right.

The information about starting point branching is contained in factor w_2. Non-branching point is preferred. This factor is relevant only for determination of the contour starting point, so the multiplicator v_2 in (5) is equal to 1 if the contour starts from j-th segment and is 0 otherwise.

$$w_2 = v_2 \times \begin{cases} a_{j,3}^{(3)}, & \text{if } a_{j,1}^{(3)} \text{ is the starting point} \\ a_{j,4}^{(3)}, & \text{if } a_{j,2}^{(3)} \text{ is the starting point} \end{cases} \quad (5)$$

Factor w_3 gives an information about the angle of movement through the branching points. For each pair of singled out neighboring contour segments, algorithm first checks the value of $a_{j,l}^{(3)}$, (l=3 or l=4) to get the information about branching of the endpoint of previous and the starting point of the next segment. If they both branch, virtual points whose coordinates are equal to the arithmetic mean of the next L points, relative to each branching point are created. Let the distance between the branching point and the corresponding virtual point of the first segment be noted as s, the same distance for the next segment as t and the distance between two virtual points as u. Angle between the movement direction of segment lines is then computed according to relation (6),

where $\phi = \left|arccos\left(\frac{s^2+t^2-u^2}{2st}\right) \times \frac{180}{\pi}\right|$. Probability of correct prediction is greater as the angle changes are smaller, which is included by the constant v_3.

$$w_3 = v_3 \times \begin{cases} 180-\phi, & \text{if } \phi \leq 180 \\ \phi - 180, & \text{otherwise} \end{cases} \quad (6)$$

Path length is computed based on the total number of segment pixels, as in (7), where $l_1, l_2 \in \{1,2\}$, depending on which of the characteristic points is an endpoint and which one is the starting point and $g\left(a^{(3)}_{j,l_1}, a^{(3)}_{j+1,l_2}\right) = \sqrt{\left(x^{(3)}_{j,l_1} - x^{(3)}_{j+1,l_2}\right)^2 + \left(y^{(3)}_{j,l_1} - y^{(3)}_{j+1,l_2}\right)^2}$. Evaluation is done corresponding to the gradient of line determined by two points - observed pixel and the transit one. Result is stored in w_4. Multiplicator v_4 has greater value for repeating trajectories than for non-repeating ones.

$$w_4 = \sum_{\text{j=m(R-P)+1}}^{\text{m(R-P)+R}} v_4 \times a_{j,1} + g\left(a^{(3)}_{j,l_1}, a^{(3)}_{j+1,l_2}\right) \quad (7)$$

Path jumpiness refers to a significant number of discontinuities in handwritten text, which is an indicator of frequent pen-from-paper detach. Shorter and continuous transition paths should be the most probable. Therefore, the factor w_5 described with (8) is included in the objective function. Jumpiness is allowed for the letters with dashes, such as t and f, which is accomplished by detecting the value of ϕ close to 90, as in relation (8).

$$w_5 = \begin{cases} 0, & \text{if } |90-\phi| \leq 10 \\ v_5 \times y^{(3)}_{j+1,l_2} + g\left(a^{(3)}_{j,l_1}, a^{(3)}_{j+1,l_2}\right), & \text{otherwise} \end{cases} \quad (8)$$

Factor w_6 contains the information about air pen tip movements and is computed according to relation (9). The first step in computing the value of this factor is a detection of pen lifting, which is achieved by comparing the shortest distance of two neighboring segments with the threshold value of the maximum distance between two segments generated by separating the contours' branching point in two points. Afterwards, the distance between previous and possible next segment, relative to the x-axis is computed. Closer and shorter segments have higher probability, given by constants $v_{6,2}$ and v_4. Position of the next segment relative to the y-axis is also relevant. After lifting a pen from a paper, the common direction of writing is from top to bottom and w_6 has a lower value for that case, which is taken into account through positive multiplicator $v_{6,1}$.

$$w_6 = \sum_{\text{j=m(R-P)+1}}^{\text{m(R-P)+R}} \frac{v_{6,1} \times y^{(3)}_{j+1,l_2} + \left|x^{(3)}_{j,l_1} - x^{(3)}_{j+1,l_2}\right|}{v_{6,2}} + v_4 \times a^{(1)}_j \quad (9)$$

Weighting multipliers are determined by testing the impact of each of the coefficients on a large set of handwritten text samples. Values that showed optimal performances on the largest number of samples are given in Table I.

Let the factors w_r $(r = 1, ..., r_m)$ computed in the m-th iteration be noted as $w_{m,r}$. Considering that the value of f_m is computed as $f_m = \sum_{r=1}^{r_m} k_r \times w_{m,r}$, the final value of the objective function is given by (10).

Table I: Empirically derived optimal values of weighting factors k_r and multipliers v_r $(r = 1, ..., 6)$.

Factor	Value	Factor	Value	Factor	Value
k_1	1	k_6	1/3	v_3	$\{0, 2\}$
k_2	100	$v_{1,1}$	$\{1, 3\}$	v_4	$\{1, 20\}$
k_3	10	$v_{1,2}$	$\{10, 500\}$	v_5	2
k_4	1/10	$v_{1,3}$	$\{0, 1\}$	$v_{6,1}$	8
k_5	1/3	v_2	$\{0, 1\}$	$v_{6,2}$	2

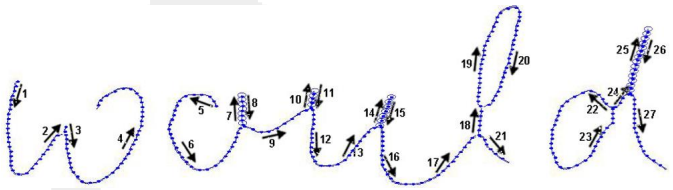

Figure 3: Predicted trajectory; dots indicate non-repeating trajectory and circles indicate the repeating one.

$$f = \sum_{m=0}^{M-1} \min_{\{w_{m,r}\}} \{f_m\} \quad (10)$$

Trajectory that corresponds to the minimal value, computed according to (10), is plotted in real-time with dots that indicate non-repeating trajectory and circles that indicate the repeating one. An example of a final result of analysing the written word "would" is shown in Fig. 3.

III. Experimental results

In order to perform verification of the handwriting trajectory prediction, random samples of handwritten words from IAM-OnDB database have been analysed by the algorithm and results are then compared to the original writing trajectory. Comparison is made for each segment of analysed written text, based on the segments order and movement direction.

All analyses were conducted on computer with Intel Core 2.50 GHz i3-3120M processor, 4 GB RAM and 64-bit Windows 7 Professional operating system.

The analysis is conducted on 5 groups of words, each containing 30 samples. Groups are formed according to the number of segments. Words of the first group contain 3-5 segments, 6-7 segments are contained in the second one, 8-9 in the third one, 10-12 in the fourth one and 13-17 in the last one. Grouping and analysing the words that contain more than 17 segments has not been conducted because of the very uneven number of segments distribution in this case. Number of analysed samples for each number of segments contained is given in Table II.

Effectiveness of the algorithm is computed according to the relation (11), where N_j represents the total number of samples for M_j number of segments, o_i is the number of correctly predicted segment orders and d_i is number of correctly predicted

Table II: Number of examined samples in the experimental analysis, with a corresponding number of included segments per each sample.

Number of segments	3	4	5	6	7	8	9	10
Number of samples	8	11	11	15	15	16	14	10
Number of segments	11	12	13	14	15	16	17	
Number of samples	10	10	5	11	4	5	5	

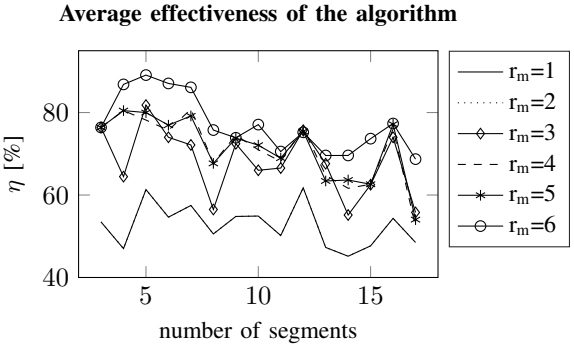

Figure 4: Comparison of an average effectiveness of the proposed algorithm and the results obtained when each of the relevant factors is excluded.

movement directions for i-th sample in j-th subgroup of total 15 analysed segment numbers, as represented in Table II.

$$\eta = \frac{1}{15}\sum_{j=1}^{15}\sum_{i=1}^{N_j}\frac{o_i \times d_i}{M_j \times N_j^2} \times 100\% \tag{11}$$

The experimentally obtained total average effectiveness of the algorithm is 77.12%.

A. Effectiveness of the algorithm

For the task of examining the effectiveness of the algorithm and the impact of individual factors on the accuracy of the results, testing is conducted for the case when all six factors are taken into account and when some of them are excluded. Results of average effectiveness computation for each of these cases are given in Fig. 4. It can be noticed that the effectiveness is the highest when all six factors are included, while the worst results are obtained when excluding all factors except for w_1 (or w_1 and w_2, in this particular analysis, because the curves overlap). The accuracy of prediction does not depend on the number of samples, so the conclusion of effectiveness for the 5 created groups cannot be derived. Accuracy mostly depends of the written letters and the performance of preprocessing procedure.

The algorithm rarely produces errors when predicting the order of analysed segments, while prediction of movement direction can be wrong in case of analysing cursive letters that have little loops, such as b and g. If these letters have low roundness, the preprocessing steps cause connecting the close parts of the written text, resulting in a line in one part of the original loop. Aforementioned problem occurred when the analysis of the word "would" was conducted. As can be seen in Fig. 3, the line on the loop of the letter l appeared and

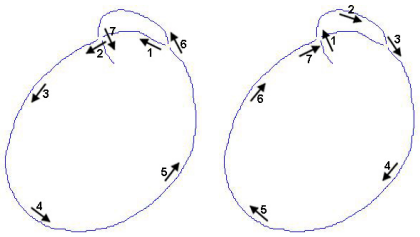

Figure 5: Common issue in predicting the order of segments and direction of movement: (a) Real pen tip trajectory; (b) Predicted pen tip trajectory.

the wrong direction of movement by the loop segment was predicted.

Predicted trajectory of pen tip movement is intuitively possible for most of the analysed handwritten text samples, but in some cases that results in multiple reordered segments and incorrect prediction of direction of movement, which reduces the effectiveness of the algorithm. An example of this problem is shown in Fig. 5. Finding the starting point of cursive letters with the circular structure, such as o and a, is complex process, especially in cases where that point is branching. This is a problem not only for computer analysis, but also for graphologists, because the starting point of the letter highly depends on handedness of the writer [3].

B. Execution time of the algorithm

Besides the effectiveness measurement, execution time of the algorithm is measured for each of the analysed samples and a set of included factors. Averaged results are shown in Fig. 6. It can be concluded that the average execution time mostly depends on the number of segments, but there are some other factors that are relevant, such as complexity of computing the objective function. Even when all the relevant factors are included, some of them are not calculated in specific cases. For example, if the pen from a paper detach has not been detected, factors w_5 and w_6 are not considered in the objective function and less time is required for the execution.

It can be seen from the Fig. 6 that the lowest required time is needed for analysing samples containing 3 and 4 segments, while increasing the number of segments to 5 causes the running time increase by more than double. Further increase generally causes a slight climbing in the execution time curves, but the changes are not sharp because the algorithm considers 5 segments when an analysed text has a length of 5 or more segments, thus preventing an exponential growth of the running time.

Execution time of the algorithm can be reduced by analysing less than 5 segments in each iteration or reducing the number of permutations for each possible segments order, lowering the allowable difference between the indices of the segments from 4 to 3. Drawback of this solution is that effectiveness of the algorithm will be degraded. If the analysis was conducted on 4 instead of 5 segments, the effectiveness would have been

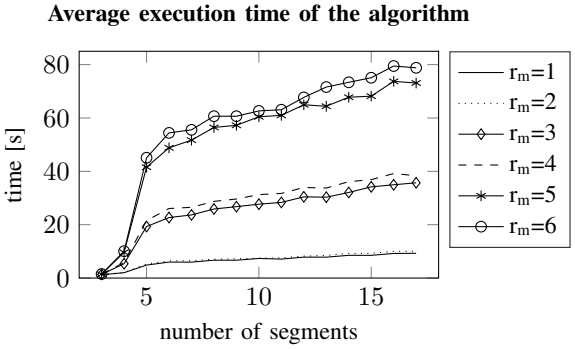

Figure 6: Comparison of an average computation time of the proposed algorithm and the results obtained when each of the relevant factors is excluded.

reduced, but it would not cause significant time reduction. It is common that one letter is composed of 5 segments after the thinning process. Those letters are mostly the ones that include dashes, such as t and f, or the letters written in a way that there are more cross-sections in the contour. For the purposes of accurate prediction in these cases, allowed difference between the indices of segments cannot be lowered.

IV. Conclusion

The experimental results show that the accuracy of prediction of the handwritten text trajectory by the proposed algorithm is averagely 77,12% and that it depends on the content of the analysed text and not on the number of segments contained in the contour that is being analysed. An important trait of the algorithm is that the proposed trajectories are intuitively possible, even if they are not correctly predicted.

Effectiveness of the algorithm can be improved by correcting a text character skeleton, as proposed in [23], which would result in smaller number of line segments, created by merging segments located at the close distance. Given the fact that the poor performance of skeletonization process is mostly the cause of diminishing effectiveness of the proposed algorithm, the biggest progress is expected by improvement of this process. The other way to increase the effectiveness is to implement a full data normalization procedure in preprocessing, which includes baseline, skew and slant normalization. This solution requires the initial state return of the analysed text when the trajectory is proposed, which is a complex procedure if the number of segments are not the same in the initial and the normalized state.

The most significant improvement of the proposed algorithm can be made by employing training methods, which can result in higher prediction accuracy and lower required execution time. Moreover, reducing computational burden will be conducted as part of future work.

References

[1] I. Siddiqi, C. Djeddi, A. Raza, and L. Souici-Meslati, "Automatic analysis of handwriting for gender classification," *Pattern Analysis and Applications*, vol. 18, no. 4, pp. 887–899, 2015.

[2] E. Sokic, A. Salihbegovic, and M. Ahic-Djokic, "Analysis of off-line handwritten text samples of different gender using shape descriptors," in *Telecommunications (BIHTEL), 2012 IX International Symposium on*. IEEE, 2012, pp. 1–6.

[3] M. Liwicki, A. Schlapbach, P. Loretan, and H. Bunke, "Automatic detection of gender and handedness from on-line handwriting," in *Proc. 13th Conf. of the Graphonomics Society*, 2007, pp. 179–183.

[4] S. Al Maadeed and A. Hassaine, "Automatic prediction of age, gender, and nationality in offline handwriting," *EURASIP Journal on Image and Video Processing*, vol. 2014, no. 1, pp. 1–10, 2014.

[5] V. Burr, "Judging gender from samples of adult handwriting: Accuracy and use of cues," *The Journal of social psychology*, vol. 142, no. 6, pp. 691–700, 2002.

[6] V. Kamath, N. Ramaswamy, P. N. Karanth, V. Desai, and S. Kulkarni, "Development of an automated handwriting analysis system," *ARPN Journal of Engineering and Applied Sciences Volume 6*, no. 9, pp. 1819–660, 2011.

[7] J. F. Vargas, M. A. Ferrer, C. M. Travieso, and J. B. Alonso, "Off-line signature verification based on high pressure polar distribution," in *Procedeeins of the 11th International Conference on Frontiers in Handwriting Recognition, ICFHR 2008*, 2008, pp. 373–378.

[8] V. Nguyen and M. Blumenstein, "Techniques for static handwriting trajectory recovery: a survey," in *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*. ACM, 2010, pp. 463–470.

[9] R. Plamondon and S. N. Srihari, "Online and off-line handwriting recognition: a comprehensive survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 1, pp. 63–84, 2000.

[10] R. Plamondon, *Progress in automatic signature verification*. World Scientific, 1994, vol. 13.

[11] A. Schröter, R. Mergl, K. Bürger, H. Hampel, H.-J. Möller, and U. Hegerl, "Kinematic analysis of handwriting movements in patients with alzheimer's disease, mild cognitive impairment, depression and healthy subjects," *Dementia and Geriatric Cognitive Disorders*, vol. 15, no. 3, pp. 132–142, 2003.

[12] J. Walton, "Handwriting changes due to aging and parkinson's syndrome," *Forensic science international*, vol. 88, no. 3, pp. 197–214, 1997.

[13] M. K. Babcock and J. J. Freyd, "Perception of dynamic information in static handwritten forms," *The American journal of psychology*, pp. 111–130, 1988.

[14] C. Viard-Gaudin, P.-M. Lallican, and S. Knerr, "Recognition-directed recovering of temporal information from handwriting images," *Pattern Recognition Letters*, vol. 26, no. 16, pp. 2537–2548, 2005.

[15] G. Boccignone, A. Chianese, L. P. Cordella, and A. Marcelli, "Recovering dynamic information from static handwriting," *Pattern Recognition*, vol. 26, no. 3, pp. 409–418, 1993.

[16] T. Huang and M. Yasuhara, "Recovery of information on the drawing order of single-stroke cursive handwritten characters from their 2d images," *IPSJ Trans*, vol. 36, no. 9, pp. 2–132, 1995.

[17] H. Bunke, R. Ammann, G. Kaufmann, T. M. Ha, M. Schenkel, R. Seiler, and F. Eggimann, "Recovery of temporal information of cursively handwritten words for on-line recognition," in *Document Analysis and Recognition, 1997., Proceedings of the Fourth International Conference on*, vol. 2. IEEE, 1997, pp. 931–935.

[18] P. M. Lallican and C. Viard-Gaudin, "A kalman approach for stroke order recovering from off-line handwriting," in *Document Analysis and Recognition, 1997., Proceedings of the Fourth International Conference on*, vol. 2. IEEE, 1997, pp. 519–522.

[19] Y. Kato and M. Yasuhara, "Recovery of drawing order from single-stroke handwriting images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 9, pp. 938–949, 2000.

[20] B. Zhang and S. N. Srihari, "Analysis of handwriting individuality using word features," in *null*. IEEE, 2003, p. 1142.

[21] M. Liwicki and H. Bunke, "Iam-ondb-an on-line english sentence database acquired from handwritten text on a whiteboard," in *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*. IEEE, 2005, pp. 956–961.

[22] W. Chen, L. Sui, Z. Xu, and Y. Lang, "Improved zhang-suen thinning algorithm in binary line drawing applications," in *Systems and Informatics (ICSAI), 2012 International Conference on*. IEEE, 2012, pp. 1947–1950.

[23] H. N. Vu, I. S. Na, and S. H. Kim, "Correction of text character skeleton for effective trajectory recovery," *International Journal of Contents*, vol. 11, no. 3, pp. 7–13, 2015.