The Nucleus ***53****, No. 2 (2016) 89-98*

www.thenucleuspak.org.pk

The Nucleus
ISSN 0029-5698 (Print)
ISSN 2306-6539 (Online)

Online Urdu Handwriting Recognition System Using Geometric Invariant Features

Z. Jan[1*,] M. Shabir[1], M. A.Khan[2], A. Ali[3] and M. Muzammal[4]

[1]*Department of Computer Science, Islamia College University Peshawar, Pakistan*

[2]*Department of Computer Science, University of Peshawar, Pakistan*

[3]*Department of Mathematics, Islamia College University Peshawar, Pakistan*

[4]*Department of Computer Science, Bahria University, Islamabad, Pakistan*

zahoor.jan@icp.edu.pk; shabir_adam@yahoo.com; bitvox@yahoo.com; arshad_math@hotmail.com; muzammal@bui.edu.pk

ARTICLE INFO

Article history:
Received : 20 January, 2016
Revised : 14 March, 2016
Accepted : 23 March, 2016

Keywords:
Online Urdu Handwriting
Geometric Invariant
Support Vector Machine
Urdu Characters and words classification
Urdu Characters and words recognition

ABSTRACT

Online touch sensitive devices facilitate users by providing an easy way for inputting online handwritten text. Many useful applications are developed for other cursive script language and are practically used in different fields like banking, commerce, academics, administration and education etc. There are also some systems proposed for online Urdu handwriting recognition but either they have low accuracy rates or high constraints on user while writing. Online Urdu handwriting recognition is a difficult task due to its cursive property and writing complexity. The proposed system tries to recognize Urdu characters and words by using geometric features i.e. cosine angles of trajectory, discrete fourier transform of trajectory, inflection points, self-intersections, convex hull, radial feature, grid (orthogonal and perspective), and retina feature. The proposed system is font, rotation, scale and shift invariant due to geometric invariant features. Before feature extraction low pass filtering and resampling is applied on each input stroke trajectory to remove noise caused by input device and hand movement. After feature extraction linear support vector machine is used for training and testing which gives up to 97% classification accuracy on test data. In recognition phase the proposed system gives a very low false rejection rate.

1. Introduction

The word “Urdu” has been derived from Turkish language which means لشکر “Lashkr” Legions or فوج “fauj” army. Urdu language is the fourth largest spoken language in the world [1]. It has been derived from different languages, therefore it has inherited the properties from other languages like Pashto, Persian, Dari, Turkish and Arabic etc. Most of the Urdu alphabet characters are common in these languages, while some characters are specific to Urdu. Almost all of these languages are written from right to left and are cursive in nature. Urdu has 38 alphabetic characters [2], Arabic has 28 [3], Ottoman Turkish has 32 basic alphabetic characters [4]; some other alphabets belong to Ottoman type of Turkish have been added to Urdu and the number of characters become 37 [4], Persian has 32 [5], Dari has 32 [6] and Pashto has 35 basic alphabets. 39 characters of Pashto have been mentioned by Shah et al [7].Urdu is cursive in nature and all the words are constructed from 38 basic alphabets called حروف تہجی “horof-e-thahji”. These alphabets connect vertically at different positions with each other to form words i.e. connected from front, back or from both sides. The basic set of Urdu alphabets حروف تہجی “horof-e-thahji” with Unicode is given in Table 1.

Table 1: Urdu Basic Alphabetic Set and Unicode (حروف تہجی)

ج (62c)	ث (62b)	ٹ (679)	ت (62a)	پ (67e)	ب (628)	ا (627)
ر (631)	ذ (630)	ڈ (688)	د (62f)	خ (62e)	ح (62d)	چ (686)
ض (636)	ص (635)	ش (634)	س (633)	ژ (698)	ز (632)	ڑ (691)
ک (6a9)	ق (642)	ف (641)	غ (63a)	ع (639)	ظ (638)	ط (637)
ھ (6be)	ہ (6c1)	و (648)	ن (646)	م (645)	ل (644)	گ (6af)
				ے (6d2)	ی (6cc)	ء (621)

A stroke is a continuous movement of a finger or stylus pen over the surface of a touch pad or touch screen. A stroke starts when stylus pen touches the surface and ends when it lefts the surface. Each alphabet of Urdu language is constructed from one or more strokes (.i.e. primary and secondary). Primary stroke is the ghost or skeleton and mandatory part of an alphabet while secondary stroke is the additional part of an alphabet which helps to differentiate an alphabet from the others having the same primary stroke. Alphabets having the same primary strokes make the secondary part mandatory as shown in the Table 2.

Table 2: Strokes of alphabets

[*] Corresponding author

Alphabets	Primary Stroke	Secondary Stroke
ٹ	ٮ	ط
ت	ٮ	••
پ	ٮ	∵
ب	ٮ	•
گ	ک	/
ک	ک	None

It is clear from the state of the art of natural language processing, that natural languages have two basic types of processing method that are online and offline. In online handwriting recognition, a user directly interacts with the system and gives data at real time. In offline character recognition system OCR, the real time data is absent; the data is not captured as user writes. In online system, the input data are taken from online stroke trajectories. In OCR, the data is taken from images or from scanned documents; images are in printed form or in handwritten form. The input devices for online handwriting systems are different from those of OCR. In online handwriting recognition mostly signal processing techniques are used while in offline character recognition mostly image processing techniques are used.

This paper presents an online geometric invariant Urdu handwriting recognition OGIUHR system. OGIUHR system is a general purpose for the unconstrained Urdu handwriting in the sense that it is font free, rotation, scale and shift invariant. The online handwriting Urdu stroke trajectories are filtered and then performed resampling, after that features are extracted which are stored in data files. Linear support vector machine SVM, take the features as input and classify it by generating classification models. In recognition phase unseen strokes are processed; extract the feature's data and recognize the strokes.

In this paper section 2 presents related work. Section 3 explains the proposed approach to the OGIUHR system. Section 4 consists of result and discussion of our experiments. Conclusion is given in section 5. Section 6 presented future direction for further work.

2. Related Work

Online Urdu handwriting recognition is a difficult task due to its cursive writing style. A limited research work is available for online Urdu handwriting recognition. The available research work has some constraints and limitations as well.

Shahzad et al. [8] proposed an online handwriting Urdu isolated characters recognition system. For classification, a common structure is used which are similar in most of characters. The main factors which differentiate characters from each other are the number of dots, position of dots and Toe (ط) a secondary stroke symbol as shown in Table 2. Features used for primary stroke to train the classifier are bounding box diagonal length, bounding box diagonal angles, first and last point distance, first and last point cosine angle, first and last point sine angle, primary stroke total length, total traversing angles, each point angle sum value and the squared value summation for those angles. The features which are selected for secondary stroke are total number of secondary strokes, secondary strokes length, secondary strokes position and number of dots that are present in secondary stroke. A rejection mechanism is used to reject an invalid character stroke. Their proposed system can recognize only one alphabet at a time.

Malik et al. [9] did some work on online Urdu character recognition. Their proposed system can recognize 39 Urdu alphabets, 10 Urdu numbers and 2 characters Urdu words with a single diacritic hat feature if it exists in the dictionary. For isolated characters, recognition accuracy rate is 93% and for Urdu numeral character accuracy rate is 78%. Malik et al. [9] used six features in their proposed work. They performed necessary filtering steps to eliminate repetition and zigzag noise. Their proposed technique recognized 200 words only, which are composed of two characters. Their proposed system cannot recognize more than two characters at a time.

Husain et al [10] performed some important work to recognize online handwritten Urdu words having 1, 2 and 3 characters by using holistic approach. They use twenty features for the recognition of primary stroke and six features for secondary stroke. The twenty-six features are large in number and become computationally very expensive if the same work is applied on mobile smart phones. They performed smoothing and de-hocking technique on chain code of each character. The extracted data are passed from back propagation neural network (BPNN). This technique cannot recognize more than three characters at a time.

Razzaq et al. [11] presented preprocessing techniques for Urdu character recognition. They reduce noise, distortion and variation from input stroke. They applied online and offline steps in preprocessing which are interpolation, segmentation, de-hooking, baseline, combining stroke and smoothing. Unconnected characters which affect the recognition of a single stroke are combined to form a single ligature in an image. They take online input strokes and then process it in both ways (i.e. online and offline). However there is no classification and recognition mechanism in the paper.

In another paper, Razzak et al. [12] proposed several preprocessing steps to normalize online handwriting Urdu character strokes by using fuzzy logic and performed several fuzzy operations. As a first step they removed hooks and then performed smoothing. Missing points are interpolated via bresenhams line drawing algorithm. They also estimate primary stroke slants in local way with the help of neighbor stroke trajectory. Secondary strokes are by nature slanted. They also estimate baseline and correct the skew in the path trajectory. They extract features for online handwriting characters by using fuzzy operation and logic. The system has claimed 89.2% accuracy. The accuracy will decrease by increasing the number of strokes. Rotated and scaled words are not under consideration in this approach.

Safdar et al. [13] proposed a system which recognizes online initial single character from a single stroke. This work is very limited and need to extend at least upto full single stroke.

3. Proposed Method

This section explains the preprocessing steps, different feature extraction technique with algorithms. After that Urdu stroke trajectories are discussed with respect to classification and recognition. Hence the proposed system consists of four working phases’ which are preprocessing, Feature extraction, classification (.i.e. training) and recognition (.i.e. testing).

3.1 System Architecture

The GUI reads motion and coordinates of the mouse or stylus from the user. Both stylus and mouse reporting rate are affected by the speed with which user moves on the device. The mouse/stylus readings are passed through low-pass filter. Low pass filter performs smoothing and minimize zigzag noise. The output from the low-pass filter then goes to Ramer Douglas Peucker RDP algorithm which tries to minimize the number of line segments representing the line. The amount of reduction in RDP is controlled by a threshold.

The Feature Extractor module extracts geometrical features of the stroke. A complete feature vector for a stroke is the concatenation of eight computed features. Featured vectors are then labeled manually for supervised training. These labeled feature vectors are added to training dataset. The training dataset is then used to train classifier. After training phase, machine-learning models are used for recognition of unseen strokes. Diagrammatical architecture of OGIUHR system is given below in Fig. 1.

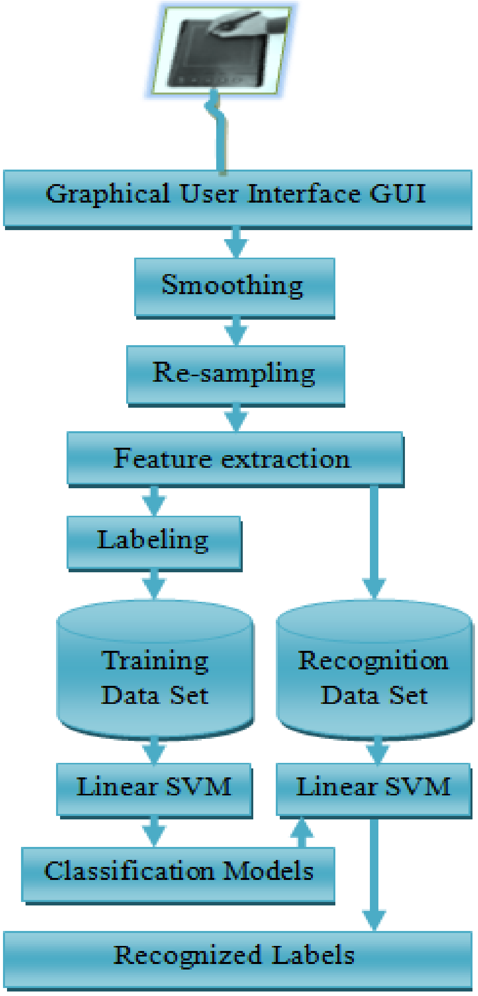


Fig. 1: System architecture

3.2 Preprocessing

Before training, machine learning models require normalized, errorless, efficient and sufficient data. Un-normalized data results in marginal learning and lead towards failure and give less accurate result that is why preprocessing step is very important. Clever selection of feature is also a key to success.

3.2.1 Smoothing/Filtering

Filters are used to remove unwanted signals/data and to extract signals/data of interest. The uses of filters depend upon the applications, some applications need low pass filter while some need high pass filter. Low pass filters allow low frequency components and stop high frequency components from a compound signal. High pass filters allow high frequency and deny low frequency components. The proposed system uses resister capacitor RC, low pass filter in discrete-time form. This filter works as weighted moving average given in equation 1. This is also known as exponentially-weighted moving average.

$$p_n = p_{n-1} \times (1-\alpha) + p_{mouse} \times \alpha \; 0 \leq \alpha \leq 1 \quad (1)$$

In Eq. (1) p_{n-1} is previous filtered point, *(1-α)* is the contribution of previous point to the next point, p_{mouse} is the current position of mouse and α is the smoothing factor or value. This algorithm is similar to a low pass filter which is used in audio signaling to separate unwanted high frequency signals that make change in the paths. The exponential smoothing technique can be used for time series data, in order to produce smooth data to make forecasts or for presentation. The time series data is an observation of sequence data points at a particular time. The observation criteria for sequential data points may be a particular order or may be a random process, but would not be noise free process. On the other hand in the simple moving weighted average, similar weight is assigned to previous observed data equally. In exponential smoothing with the passage of time exponentially decreasing weights are assigned. The low pass filter formula which is used in this research is the contribution of Brown, and work well in such types of problem Brown et al. [14].

3.2.2 Resampling

Redundant, irrelevant and too many data always create problem in every system. Redundant data create the problem of inconsistency, irrelevant data decreases the accuracy rate and too many data about a particular object increase the computation/processing time. The proposed technique used U. Ramer [15] algorithm to quantize the stroke trajectory and reduce the data points.

3.3 Feature Extraction

The proposed technique uses eight features which are cosine angle of trajectory, discrete fourier transform, inflection points, self-intersection, convex hull, radial feature, grid (orthogonal and perspective), and retina feature. SVM not converged efficiently upto 6th feature, but when 7th and 8th feature is added accuracy increase drastically. The data obtained from these features are used for classification.

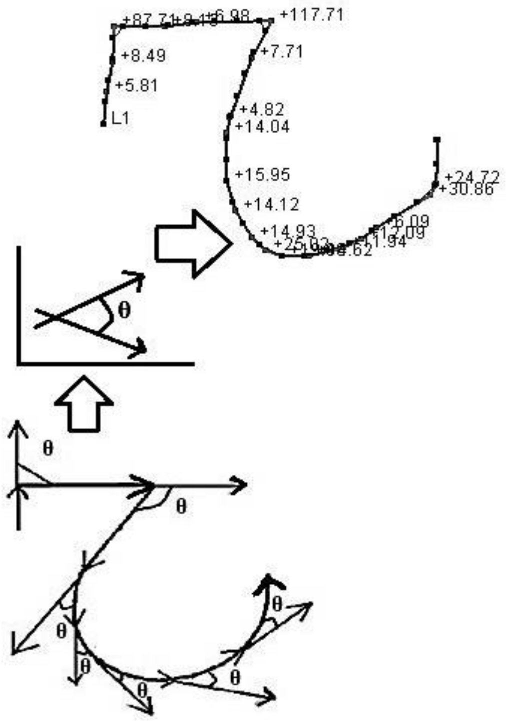
3.3.1 Cosine Angles of Trajectory

The whole trajectory is divided into equal length of fixed number of vectors, which make angles of cosine (relative angle) as data feature from dot product as shown in the Fig. 2.

3.3.2 Discrete Fourier Transform (DFT)

Another feature used is DFT. 1-D Discrete Fourier Transform algorithm proposed by Cooley et al [16] is used. The sequence of data point's (angles) in a stroke trajectory as shown in Fig. 3 is generated at a discrete time sequence so the number of data points depends upon number of time unit. Fig. 4 represents frequency spectrum of Fig. 3 after applying Eq. (2). In this research, angles trajectory represents a signal in spatial domain with respect to time. The forward discrete Fourier transform (DFT) of a one-dimensional complex array X of n size; calculated in an array Y as shown in Eq. (2), where:

$$Y_k = \sum_{j=0}^{n-1} X_j e^{-\pi jk\sqrt{-1}/n} \quad (2)$$



$$\Delta_{x1} = b_x - a_x$$
$$\Delta_{y1} = b_y - a_y$$
$$\Delta_{x2} = d_x - c_x$$
$$\Delta_{y2} = d_y - c_y$$
$$l_1 = \sqrt{\Delta_{x1}^2 + \Delta_{y1}^2}$$
$$l_2 = \sqrt{\Delta_{x2}^2 + \Delta_{y2}^2}$$
$$\theta = \frac{\Delta_{x1} \times \Delta_{x2} + \Delta_{y1} \times \Delta_{y2}}{l_1 \times l_2}$$

Fig. 2: Cosine angles of trajectory generated by the proposed technique

3.3.3 Inflection points

If the direction of stroke trajectory is suddenly changes at a point with a certain angle or from negative to positive and vice versa, that point is called inflection point. In stroke trajectory if the value of curve angle is greater than 90° then that angle is an inflection point. Count the total number of inflections in a stroke trajectory and then add to the feature's vector. Inflection points are shown in the Fig. 5.

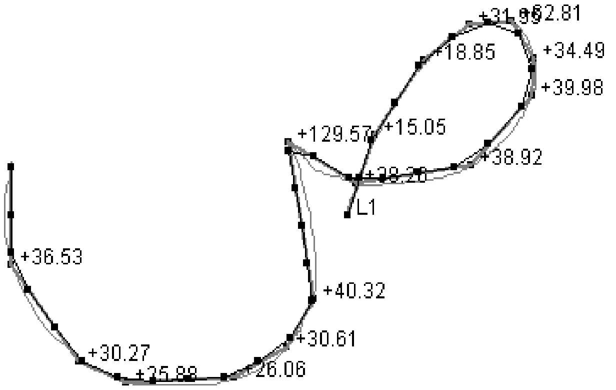


Fig. 3: ص character in time domain

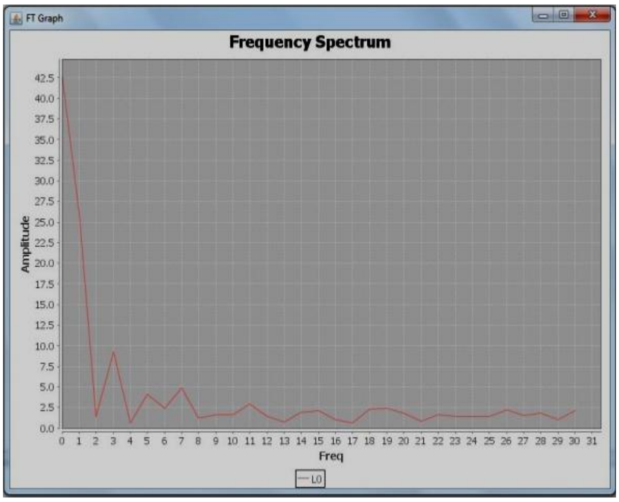


Fig. 4: ص character in frequancy domain (DFT)

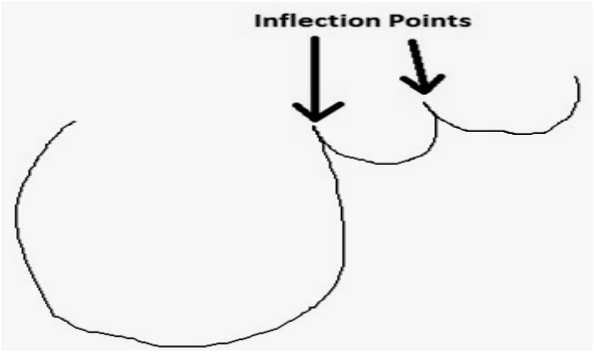


Fig. 5: Inflection points in character س

Inflection Point Algorithm

```
SET Inflection_count = 0
FOR each T_angle in T_angles_list
IF T_angle>I_threshold_angle then
Inflection_count + = 1
```

3.3.4 Self-Intersections

Intersection is the overlapping of two or more than two lines at a point and that point is common in their data set as shown in the Fig. 6. Intersections in stroke trajectory are data features for classification that how many times a stroke trajectory intersects itself.

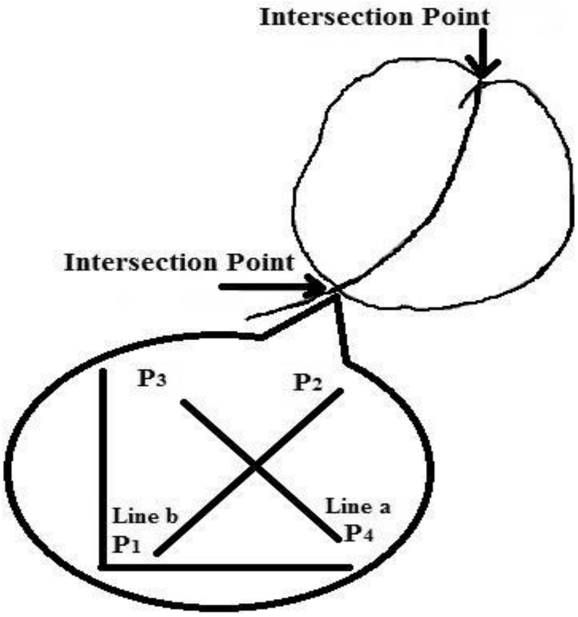


Fig. 6: Self-intersection in character ھ

Self intersection algorithm

```
SET n = path.segments.length
SET intersection_count = 0
FOR i = 0 to n-1
FOR j = 0 to n-1
IF (i ≠ j)
SET x = path.segments[i]
SET y = path.segments[j]
IF intersect (x, y) then
intersection_count + = 1
RETURN intersection_count
```

3.3.5 Convex hull

A convex hull in a two dimensional planor in three dimensional space is a small set of data points which just touch the outer boundary and are connected through line segments in such a way that all other data point are come inside the connected points as shown in the Fig. 7. Graham's Scan convex hull algorithm is used to find the convex hull of finite number of points in the plane [17]. In this feature center of gravity of the convex hull has computed, then draw line segments from the center in omni direction, intersecting the boundary of the convex hull. Angle of intersection between line segments and convex hull boundary is computed and are added into the feature's vector.

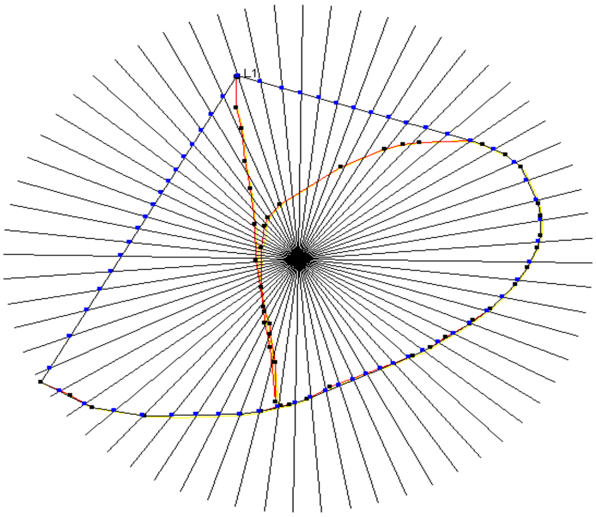


Fig. 7: Convex hull of character ط

3.3.6 *Radial*

In Radial feature, vectors of equal length are projected from the center of gravity of stroke trajectory in omni direction starting from the first point of the stroke trajectory and spread around the stroke to make a circular wheel. The distances between two consecutive vectors are equal and the radius length of the radial is the longest possible vector. All other vectors will follow the same length intersecting the stroke trajectory or some time not intersecting at all as shown in Fig 8. Each single intersecting angle between stroke trajectory and line of radial is calculated and then add it to the feature's vector.

Radial Algorithm

1. Compute convex hull for the stroke
2. Compute the Geometrical center C of the resulting convex hull polygon
3. Find the point on stroke which is at maximum distance from the center of convex hull polygon, this point would become radius R of the stroke enclosing circle
4. Split the circle in K sectors located at center C with radius R
5. For each sector compute the average of segment INTERSECTION ANGLE and add it to feature vector

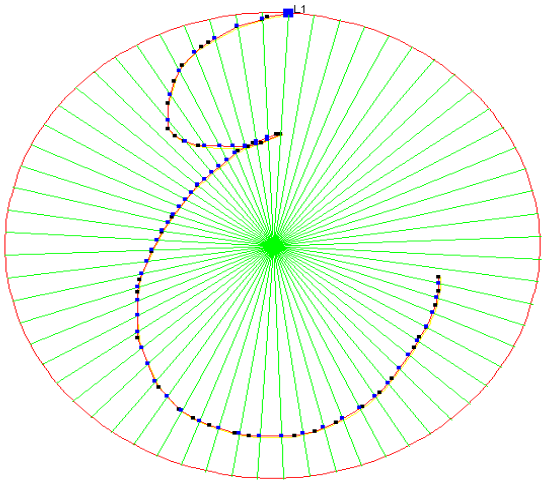


Fig. 8: Radial feature for character ع

3.3.7 *Grid*

For the identification of stroke trajectory another feature that is used is grid. Grid is divided into two types' orthographic parallel and perspective projection. Before the construction of orthographic parallel grid and perspective projection grid, center of gravity is found through convex hull.

3.3.7.1 *Orthographic Parallel Grid*

In Orthographic parallel grid the first projection is started from the first point of the stroke trajectory passing from the center of gravity. Other projections are parallel to the first one in the same direction with equal distance from each other. The grid also possesses same number of projection perpendicular to the first projection. These projections cross all the projections on the way forming a grid of squares of equal sizes as shown in Fig. 9. The stroke and grid lines make angles at the point of intersection in either direction; these angles are taken as data and stored in feature's vector. In case a stroke makes two or more angles with a single grid line then mean angle value is calculated and added to the feature vector.

3.3.7.2 *Perspective Projection Grid*

The second type is perspective projection having four lines of sight; from the corners of Orthographic grid. Opposite corners of orthographic parallel grid are connected through lines. These two lines are called middle lines. Perspective lines originate from corners and project on opposite middle line. Same procedure is followed from each corner to construct a full perspective projection grid as shown in Fig. 9. Stroke trajectories intersect various lines of perspective projection at different points and make angles with each other. Each line from a corner is followed and takes the angle of intersection with trajectory as data.

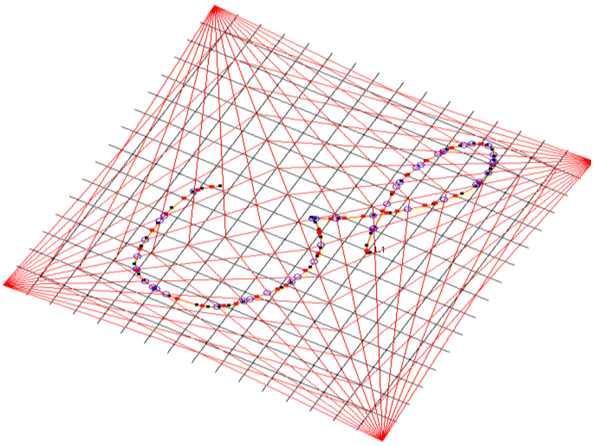


Fig. 9: Orthogonal and perspective grid for character ص

Grid Algorithm

1 Compute convex hull for the stroke.

2 Compute the Geometrical center C of the resulting convex hull polygon.

3 Compute the coordinates of a square S that encloses the circle.

4 Rotate square S clockwise or counter clockwise to align it with the starting coordinate of strokes.

5 Split the square S into equal number of rows and columns.

6 For each column and for each row compute the average angle of intersecting line segments from the stroke and add to feature vector.

7 Divide the square S using two diagonal line segments LS1 and LS2.

8 Divide LS1 in M equi-distant points and LS2 in N equi-distant points lying over LS1 and LS2 respectively.

9 Compute line segments from each corner to from LS1 to M resulting in lines LX and LS2 to N points resulting in lines LY.

10 Compute average value for stroke segment intersection angle for each line in LX and add to feature vector.

11 Compute average value for stroke segment intersections for each line in LY and to feature vector.

3.3.8 Retina

Retina is a closed square geometrical shape. The center of gravity for each online stroke/trajectory is found using convex hull. Retina is further divided into small square closed geometrical shape called cell. The number of cell depends on user selection. Increase the number of cells it will extract more data for training and testing and will increase the accuracy rate. Each cell consists of crystals, online stroke/trajectory travel through the retina and intersect some of the crystals in each cell. Trajectory line and crystal lines make angels where they intersect each other as shown in Fig. 10. These angles are summed and then divided by total number of intersecting points to calculate a single mean value in each cell.

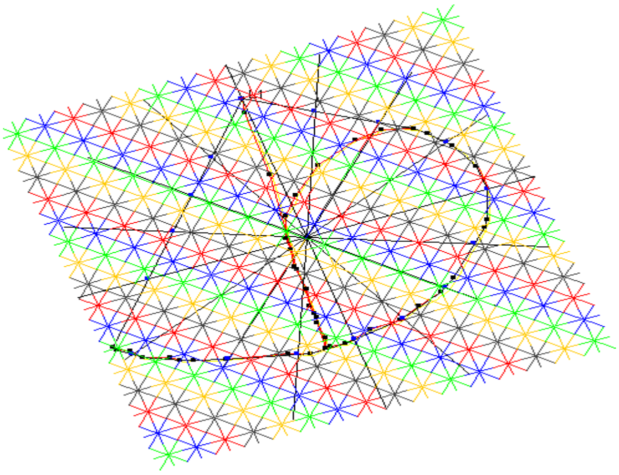


Fig. 10: Retina feature for character ط

Retina Algorithm

1. Compute the stroke enclosing square S.
2. Divide S into equal number of rows and columns to divide into N cells.
3. FOR each cell.
4. Compute the Cell center coordinates C.
5. Divide the Cell C into Sectors from cell center C.
6. FOR each sector compute the sum of stroke intersection angle and then compute the average.
7. Add the average angle value feature vector.

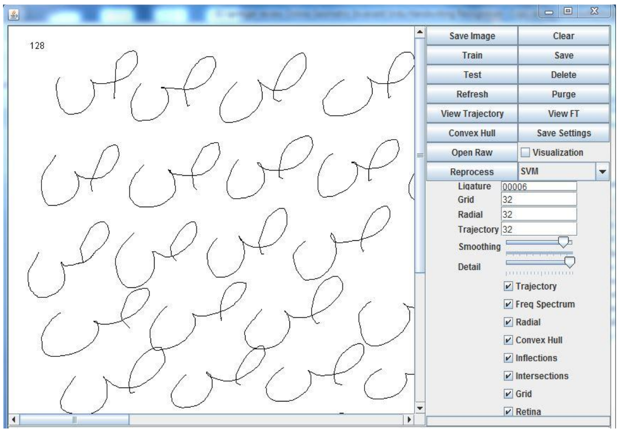
4. Result and Discussion

A number of experiments have been executed for finding optimal size values of the considered features for enabling OGIUHR system to give maximum classification and recognition accuracy. The classification experiments are based on the Urdu alphabetic set. Emphatically Urdu alphabetic set is classified into 18 classes/groups using similarities in primary stroke structure. That is, some of the primary isolated alphabetic stroke structures are similar if the secondary strokes and diacritics are removed. The descriptions of the classes along with labels are shown in Table 3.

For classification a numerical dataset for each of the alphabetic class is generated by using eight features. Primary ligature of each alphabetic class has been defined and written 128, 64, 32 and 16 times (.i.e. samples) in the text area with different sizes and rotations as shown in Fig. 11. During the entire course of datasets construction the number of detail (ε) is set to 17 and the degree of smoothing (α) is set to 61. To construct a dataset all of the features are turned on and the angles (.i.e. Grid, Radial, and Trajectory) are set to 64, 32, 16, 8, 4 and 2.

Table 3: 18 Classes of alphabet

Class No	Frequency	Characters/group						
1	2	ا	آ					
2	7	ب	پ	ت	ٹ	ث	ن	ں
3	4	ح	خ	ج	چ			
4	7	د	ڈ	ذ	ر	ز	ژ	ڑ
5	2	س	ش					
6	2	ص	ض					
7	2	ط	ظ					
8	2	ع	غ					
9	2	ف	ق					
10	2	ک	گ					
11	1	ل						
12	1	م						
13	1	و						
14	1	ھ						
15	2	ہ	ة					
16	1	ء						
17	2	ی	ئ					
18	2	ے	ۓ					



□ Visualization

SVM

Ligature 00006

Grid 32

Radial 32

Trajectory 32

☑ Trajectory

☑ Freq Spectrum

☑ Radial

☑ Convex Hull

☑ Inflections

☑ Intersections

☑ Grid

☑ Retina

Fig. 11: Interface of the proposed system

These values determines the number of angles a feature will use to populate the dataset such as by setting the value to 64 will enable a feature to extract 64 angles for each of the primary stroke entered resulting into a dataset consisting of 128 x 64 entries. Features such as self-intersection, inflection points, and DFT are also computed their values for each primary stroke. At the end, total 18 dataset files (i.e. one file for one class) are produced and online available at http://www.szic.pk/resource.php. The Classification accuracy achieved using the datasets is depicted below in Table 4.

Table 4 shows high accuracy on features size 32 and 16. These two are optimal feature size values on which SVM converges efficiently. Features Size 64 is computationally more expensive and shows less accuracy as compared to features size 32 and 16 because of too many data. Features size such as 8, 4 and 2 are computationally less expensive but shows less accuracy as compared to size 32 and 16 because of insufficient data for classification.

The experiments show that the accuracy rate depends upon features sizes. Increasing and decreasing of features sizes causes the accuracy increase and decrease.

Table 4: Feature size affects the classification accuracy for 128, 64, 32 and 16 samples of alphabet (“α = 61” and **“ε = 17”**)

S. No	Urdu alphabet	Grid, radial and trajectory size	Linear SVM classification accuracy in %			
			No of sample (128)	No of sample (64)	No of sample (32)	No of sample (16)
1	ا to ے	64	95%	95%	95%	96%
2	ا to ے	32	97 %	97%	97%	97%
3	ا to ے	16	97%	97%	97 %	97%

4	ا to ے	8	94.6%	94.3%	94.1%	96%
5	ا to ے	4	93.1%	92%	91.5%	94.2%
6	ا to ے	2	80.6%	79.3%	78.5%	91.3%

Table 5: Recognition for multiple characters

No of characters	Multiple characters	Features used	Grid, radial and trajectory size	Input samples	Recognized sample	Rotated, scale and shift invariant	Falsely rejected multiple characters by linear SVM
2	با	All	32	20	20	Yes	0
3	پیر	All	32	20	20	Yes	0
4	بتیا	All	32	20	20	Yes	0
5	اسلام	All	32	20	19	Yes	1
6	پھیلنا	All	32	20	19	Yes	1
7	پاکستان	All	32	20	19	Yes	1
8	چھنچھنیا	All	32	20	18	Yes	2

To find the recognition accuracy for multiple characters at same time, an experiment has been conducted in which entire features are used. Configuration of the proposed system for multiple characters at same time is; Grid, Radial and Trajectory sizes to 32, “α” smoothing value to 61 and “ε“ detail value to 17. Features size 32 means that 32 angles are taken as data from each feature. The proposed OGIUHR system recognizes the handwritten multiple characters with high accuracy rate as shown in Table 5. In this experiment we entered ا and ب both as a group of 2 characters, پیر as a group of 3 characters and بتیا as a group of 4 characters 20 times each and the proposed system recognized it 20 times with 0 rejections. We also entered 5, 6 and 7 characters group 20 times each and the proposed system recognized it 19 times with 1 rejection. Similarly we also entered a group of 8 characters 20 times; but this time proposed system rejects 2 entries. In this experiment each group of characters is a valid Urdu word.

5. Conclusion

Online handwriting Urdu character recognition is a difficult and hard problem to solve due to its complex writing style. For this purpose OGIUHR system is proposed, which recognizes geometric invariant online handwriting Urdu characters and words. In order to classify each character, eight features are used i.e. Cosine angle of trajectory, DFT, Inflection points, Self-Intersections, convex hull, Radial feature, Grid (orthogonal and perspective) and Retina feature. For classification, proposed OGIUHR system uses linear SVM which generates models of each character. SVM converge with high accuracy in less amount of time. In recognition for multiple characters the same eight features, preprocessing technique and Linear SVM is used. The above experiments show maximum classification accuracy of 97%. The experiment for multiple characters recognition also shows a good result.

This research has immense potential for future work by adding and testing more geometrical features i.e. hexagonal grid, triangular grid, spiral grid, H-fractal and star fractal etc. The proposed system can be used with other types of SVM having different kernel functions. Other classifier will also consider in near future. The proposed technique can improve by adding dots because dots are unique and important feature in the recognition of Urdu handwriting. The same system can also be used for other languages.

References

[1] A short encyclopedia, “Most widely spoken language of the world” December 1, 2015. [Online]. Available: http://www.einfopedia. com/most-widely- spoken-languages-of-the-world.php. [Accessed: Dec. 12, 2015].

[2] M.A. Khan, A. Habib and M.N. Ali, “Corpus Based Mapping of Urdu Characters for Cell Phones”, Proceedings of the Conference on Language & Technology, Pakistan, vol. CLT09, pp. 121-25, 2009.

[3] M.I. Razzak, A. Belaid and S.A. Hussain, “Effect of ghost character theory on arabic script based languages character recognition”, WASE Global Conference on Image Processing and Analysis, Taiwan, China, GCIA09, inria-00579666, version 1, February, 2009.

[4] Y. Saydam, “Language use in the Ottoman Empire and its problems (1299-1923)”. University of Johannesburg, 2006, [Online]. available:https://ujdigispace.uj.ac.za/bitstream/handle/ 10210/741/YSaydam_thesis_last.pdf, [Accessed March. 14, 2015].

[5] S. Mozaffari, K. Faez, F. Faradji, M. Ziaratban and S.M. Gulzan", A comprehensive isolated Farsi/Arabic character database for handwritten OCR research", Tenth International Workshop on Frontiers in Handwriting Recognition, (IWFHR), pp. 385–389, 2006.

[6] Wahab, Mehreen, H. Amin, and F. Ahmad, "Shape analysis of pashto script and creation of image database for OCR", International Conference on Emerging Technologies, IEEE, Islamabad, Pakistan. pp. 287-90,2009.

[7] A.S. Zahr, "Ligature based optical character recognition of Urdu-Nastaleeq font", Multi Topic Conference, Karachi, Pakistan, pp. 25-25,2002.

[8] N. Shahzad, B. Paulson, & T. Hammond, "Urdu Qaeda: recognition system for isolated urdu characters", Proc. of the IUI Workshop on Sketch Recognition, Sanibel Island, Florida, pp. 1-5, 2009.

[9] S. Malik and S.A. Khan, "Urdu online handwriting recognition" Emerging Technologies, Proc. of the IEEE Symposium, pp. 27-31, 2005.

[10] S.A. Husain, A. Sajjad and F. Anwar, "Online Urdu Character Recognition System", MVA, pp. 98-01, 2007.

[11] M.I. Razzak, S.A. Hussain, M. Sher, and Z.S. Khan. "Combining offline and online preprocessing for online urdu character recognition", Proc. of the International Multi Conference of Engineers and Computer Scientists, Hong Kong, vol. 1, pp.18-20, 2009.

[12] M.I. Razzak, F. Anwar, S.A. Husain, A. Belaid and M. Sher. "HMM and fuzzy logic: A hybrid approach for online Urdu script-based languages' character recognition", Knowledge-Based Systems, vol. 23, pp. 914-23, 2010.

[13] Q. Safdar and K.U. Khan, "Online Urdu handwritten character recognition: Initial half form single stroke characters", IEEE 12th International Conference on Frontiers of Information Technology, pp. 292-29, 2014.

[14] R.G. Brown, "Smoothing, forecasting and prediction of discrete time series", Courier Dover Publications, New York, USA, 2004.

[15] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves", Computer Graphics and Image Process in, vol. 1, pp. 244-56, 1972.

[16] J.W. Cooley and J.W. Tukey, "An algorithm for the machine calculation of complex fourier series", Mathematics of computation, vol.19. pp. 297-01, 1965.

[17] R.L. Graham, "An efficient algorithm for determining the convex hull of a finite planar set", Information Processing Letters, vol. 1, pp. 132-33, 1972.