TECHNICAL TRANSACTIONS | CZASOPISMO TECHNICZNE
AUTOMATIC CONTROL | AUTOMATYKA
2-AC/2013

ALEX TORMÁSI[*], LÁSZLÓ T. KÓCZY[**]

CONCEPT AND DEVELOPMENT OF A FUZZY-BASED MULTI-STROKE CHARACTER RECOGNIZER

KONCEPCJA I ROZWINIĘCIE ROZPOZNAWANIA WIELOLINIOWEGO PISMA ODRĘCZNEGO NA PODSTAWIE LOGIKI ROZMYTEJ

A b s t r a c t

In this paper, the latest member of the FUzzy-BAsed character Recognizer (FUBAR) algorithm family with multi-stroke character support is presented. The paper summarizes the basic concept and development of multi-stroke FUBAR and compares the single-stroke, multi-stroke FUBAR algorithms with the most similar methods found in literature.

Keywords: *fuzzy systems, fuzzy grid, fuzzy-based character recognition*

S t r e s z c z e n i e

W niniejszym artykule opisano najnowsze rozwiązanie z rodziny algorytmów rozpoznawania pisma odręcznego na podstawie logiki rozmytej, wspomagające wykrywanie wieloliniowych liter. W artykule przedstawiono podstawowe pojęcia oraz rozwój autorskiego algorytmu opartego na logice rozmytej, a także porównano go – zarówno w wersji dla jednoliniowych oraz wieloliniowych liter – z podobnymi metodami znalezionymi w literaturze.

Słowa kluczowe: *systemy rozmyte, rozmyte siatki, rozpoznawanie pisma na podstawie logiki rozmytej*

[*] M.Sc. Alex Tormási, e-mail: tormasi@sze.hu, Department of Information Technology, Faculty of Engineering Sciences, Széchenyi István University, Györ.

[**] Prof. D.Sc. Ph.D. László T. Kóczy, Department of Automation, Faculty of Engineering Sciences, Széchenyi István University Györ; Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics.

1. Introduction

The classification problem is a well researched area covering several methods and applications [1, 2]. Character recognition is a multi-dimensional subfield in classification problems, which has also been investigated and researched for a long time by academics and industrial partners as well.

Most of the recognition methods in literature are using various computational intelligence methods and other special solutions to reach high accuracy recognition with a low computational cost [3, 4]. Despite the high accuracy, these methods are not always usable (on devices with limited hardware resources) for on-line (real-time) handwriting recognition as a result of their high computational complexity and processing time. It is very important to find a recognition engine, which is able to process the input strokes within a short time period with an acceptable level of accuracy even on devices with limited resources such as tablets.

LaLomia defined the user acceptance threshold at 97% [5], however, most multi-stroke character recognition methods known from the literature that are applicable for 26 symbols are well below that, on the other hand, with a strict set of symbols (16 gestures) the $N recognizer reached 96.7% [3].

In this paper, we present a new attempt to recognize multi-stroke letters (26 symbols) with a rather good recognition rate (however definitely below LaLomia's 97% threshold). As a starting point, the FUBAR algorithm that was very successful for single-strokes is used, with extensions and modifications towards multi-stroke symbols (up to 3 strokes).

After the introduction, basic concept of the FUBAR algorithms [6, 7] is overviewed. In Section 3, the design and development of the new method with the capability of recognizing multi-stroke symbols is presented. In Section 4, results are presented and the average accuracy of the single-stroke and multi-stroke FUBAR algorithms are compared, for the case of the same methods with a hierarchical rule-bases [8, 9]. The results are summarized and future directions are discussed in the last section.

2. Concept of the FUBAR Algorithm Family

2.1. Features, Goals and Limitations of FUBAR

During the design of the concept of FUBAR methods, four key important features were identified as the necessary but not sufficient condition for modern and acceptable recognition engines, which are the following:

1. Acceptable accuracy: The algorithm must reach the user acceptance threshold.
2. Efficiency: The designed methods must fit to the user's requirements in response time and even in hardware with limited resources such as tablets. This means that complex geometrical transformations and other complex mathematical functions should be avoided.
3. Flexibility of the alphabet: The model of the alphabet must be easily modifiable to support various alphabets and context-sensitive recognition.
4. Learning: The designed system should be able to learn user-specific writing styles.

These features were considered as main goals to achieve; all the used techniques and the model of the designed system were selected based on their properties and abilities to reach the listed goals.

The focus during the development was on the recognition engine itself in order to reduce the influences caused by other sub-problems of the recognition process such as segmentation. The designed recognition engine is on-line (the algorithm uses digital ink information to describe strokes) and personalized (it recognizes the handwriting of a specific person).

2.2. Input Handling and Processing

The FUBAR method collects the positions of the digital pen used during the writing process (a three-dimensional continuous input signal is shown in Fig. 1), which is represented by a list of two-dimensional coordinates in chronological order.

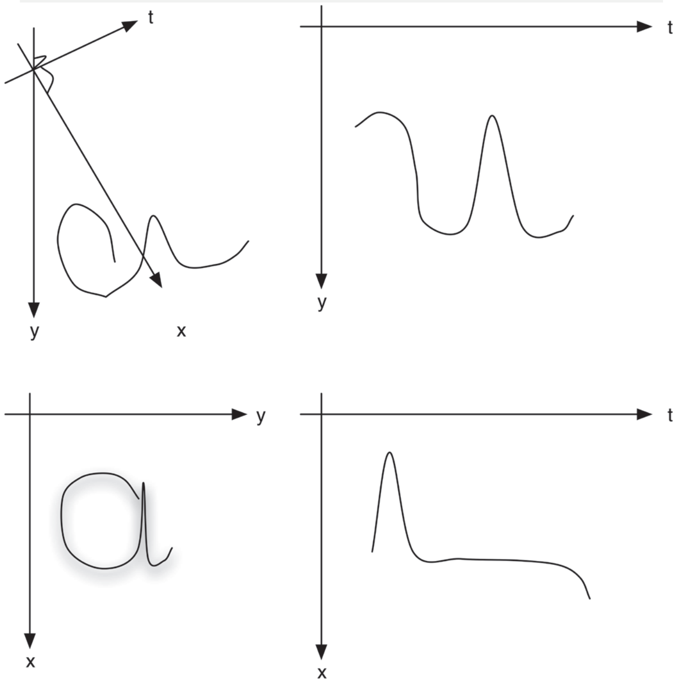

Fig. 1. The multi-dimensional input stroke from various aspects

The received stroke contains empty spaces (it is non continuous) depending on the writing speed and the BUS and CPU usage due to hardware and bandwidth limitations. It is more difficult to process the stored stroke as a result of the stochastic properties of the point distribution.

For further processing, the input stroke should be re-sampled, which provides a low-level anti-aliasing for the stroke and an almost equal distance between the sampled points of the stroke as seen in Fig. 2.

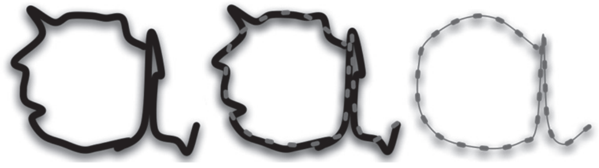
Fig. 2. Comparing the original and re-sampled input stroke

2.3. Character-Feature Extraction

The next step of the FUBAR method is feature extraction. The method use two kinds of features: (1) the width/height ratio of the stroke; and (2) the average number of points in rows and columns of a grid drawn around the stroke.

The first FUBAR algorithm used general grids with sharp borders, but the method reached a low average recognition rate as a result of the italic writing style of the test subjects as seen in Fig. 3.

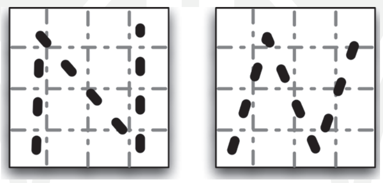
Fig. 3. Strokes with normal and italic writing style in a grid

Other similar recognizers are using mathematical transformations such as rotation to resolve the italic writing style problem, this increases the complexity of the algorithm. In the FUBAR algorithms, a fuzzy grid [6] is used, where the columns and rows of the grid are defined by fuzzy sets [10] to keep low the computational cost of the method. The points of a stroke may belong to more than one row or column with various membership values as seen in Fig. 4.

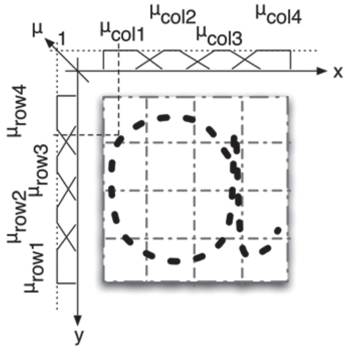

Fig. 4. A fuzzy grid example

2.4. Inference

In FUBAR algorithms, a discrete Takagi-Sugeno method [11] is used for the inference phase. The fuzzy sets in the initial rule-base of the system were determined statistically from previously collected samples.

The rule-base represents the alphabet; each character is defined by a single rule. The antecedent of the rules consists of the previously collected stroke-features, while the consequent part of the rule defines the degree of matching between the features of the input stroke and the character represented by the given rule.

The algorithm returns the character assigned to the rule with the highest matching value.

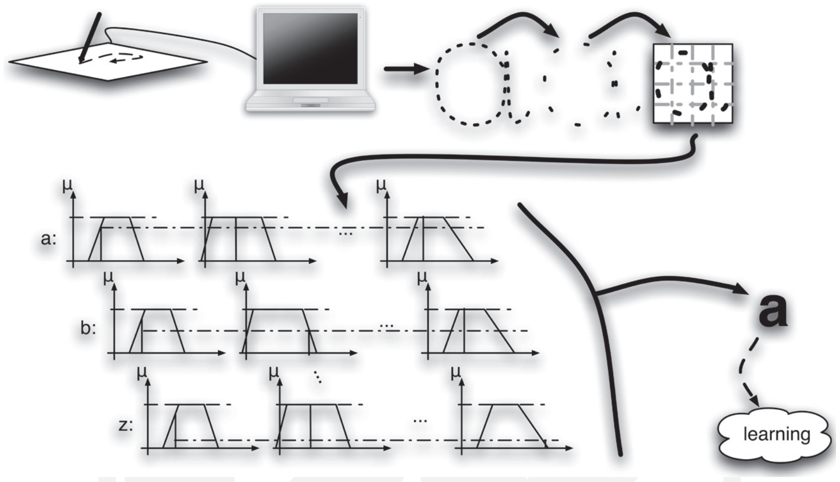

Fig. 5. Concept of FUBAR algorithms

After the inference step, the system is able to change the initial rule-base according to the features of the input stroke, this gives the ability to learn user specific writing styles. The learning phase was disabled during the tests to avoid its influence on the results caused by the heuristic properties of the used algorithm.

3. Multi-Stroke Character Support of FUBAR

3.1. Motivation and Concept

The first members of the FUBAR algorithm family supported only single-stroke symbols, but the handling of multi-stroke characters was the next step in the development of the recognition engine.

The designed method is independent from the order or the number of sub-strokes representing a multi-stroke character, but the basic shape must be identical to the model stored in the rules. This provides freedom to the users and the possibility to use any permutations of the sub-strokes representing the character with the same look.

The algorithm merges together each input sub-stroke to represent a multi-stroke character, which means that it will be represented by a single list of *x* and *y* coordinates in chronological order like a single-stroke character; this merged stroke is used during the same steps of recognition as described in the previous section.

3.2. Determining the Initial Rule Base

In the first members of FUBAR, the parameters extracted from a collection of 60 single-stroke character samples were used to determine the rule-base for the fuzzy system. The quartiles of the character-features were used to calculate the break points of the trapezoidal shaped fuzzy sets. The same method was used during the determination of the initial rule-base for the multi-strokes to have a better basis for the comparison of the algorithms.

3.3. Reducing Computational Complexity with Hierarchical Rule Base

There are many papers dealing with the use of hierarchical rule-bases in fuzzy systems in different areas [8, 9]. The use of hierarchical fuzzy rule-bases could reduce the computational cost of the single-stroke FUBAR method by decreasing the number of the evaluated rules. The details of building the hierarchical rule structure by rule input parameters for the single--stroke alphabet were presented in [12].

In the multi-stroke FUBAR algorithm, the number of sub-strokes representing a character is also used during the inference. Only those rules are evaluated, which have exactly the same number of sub-strokes as the input character has. This means that each character must be written with a predetermined number of sub-strokes, which is a limitation compared to the original multi-stroke FUBAR with a flat rule-base, but the number of the evaluated rules were significantly reduced.

4. Results

Both single-stroke and multi-stroke FUBAR algorithms were tested using the same context and conditions. A training set with 60 samples per character from various test subjects was used to determine the initial rule-base by calculating (using the quartiles of the dataset as breakpoints of trapezoid membership functions) the fuzzy sets in antecedents of the rules describing the ‘template symbols’ (the knowledge is stored as a single fuzzy rule per characters, also known as the template symbol). Another 120 samples per character (validation set) were used to determine the average accuracy of the methods.

The best result for the multi-stroke FUBAR was achieved by the algorithm using a 3×4 fuzzy grid; the letter-wise average recognition rates are listed and compared with the results of the similar single-stroke FUBAR method (with both flat and hierarchical rule-bases) in Table 1.

Table 1

Letter-wise average recognition rates of various FUBAR algorithms

Symbol	FUBAR algorithms with various properties			
	Single-Stroke FUBAR with 6 × 4 fuzzy grid	Multi-Stroke FUBAR with 3 × 4 fuzzy grid	6 × 4 Single-Stroke FUBAR with hierarchical rule-base	3 × 4 Multi-Stroke FUBAR with hierarchical rule-base
A	100	96.1111	100	96.1111
B	92.7778	89.4444	92.7374	89.4444
C	97.7778	76.6667	97.7654	76.6667
D	98.8889	96.6667	98.8827	96.6667
E	98.8889	92.2222	97.2067	92.2222
F	100	96.1111	100	96.1111
G	100	97.2222	100	97.2222
H	100	95.5556	100	95.5556
I	100	98.3333	100	98.3333
J	100	97.7778	100	97.7778
K	99.4444	96.6667	99.4413	96.6667
L	100	98.3333	100	98.3333
M	100	96.1111	100	96.1111
N	100	96.6667	96.0894	96.6667
O	93.8889	92.7778	93.8548	92.7778
P	100	92.7778	100	92.7778
Q	100	97.7778	100	97.7778
R	100	87.7778	100	87.7778
S	100	97.7778	100	97.7778
T	100	96.1111	100	96.1111
U	100	93.3333	97.2067	93.3333
V	100	88.3333	98.3240	88.3333
W	100	92.2222	100	92.2222
X	98.3333	89.4444	98.3240	89.4444
Y	100	91.1111	99.4413	91.1111
Z	100	85	100	85
Average	99.23	93. 4	98.82	93.4

All the mistakes made by single-stroke FUBAR were related to false-positive results. It was caused by the overlap of fuzzy sets describing rule input parameters; the membership values of some input variables could not been distinguished between different (true and false-positive) symbols (similarly to over fitting).

The results have been analyzed in depth including the search for the reason of the false results in the multi-stroke FUBAR. The mistakes of the multi-stroke FUBAR were caused by inconclusive results. All rules had 0 as the degree of matching for the input, which means that it could not find a rule with parameters (describing a letter) similar to the input. Each recognition process returning with an error could be traced back to the fuzzy sets describing the rule antecedents. The sources of all the false results in the multi-stroke system were pointing at uncovered areas in the antecedents of the fuzzy rules. This means that there was at least one antecedent in each rule with a 0 membership degree for at least one input parameter. This could be solved by tuning the fuzzy sets, covering a wider range over the universal set of the given dimensions or by using a sample based model identification technique like a meta-heuristic optimization algorithm.

The single-stroke FUBAR could reach 99.23% accuracy with the initial rule-base on the validation set, which is well over the user acceptance threshold of 97% [5]. The average recognition rate of the single-stroke FUBAR with a hierarchical rule-base was below the accuracy of the same method using flat a rule-base. The 0.41% drop in the average recognition rate was caused by the used meta-level rules in the hierarchy. The meta-rules are determining which (predefined) subset of the rules should be evaluated in the given case. The used parameter for the meta-rules was the average fuzzified number of points in the third row of the fuzzy grid drawn around the symbol. This influence of the meta-rules could be ruled out or minimized by defining more complex meta-rules, which are able to select the subset more accurately.

Both multi-stroke FUBAR methods with flat and hierarchical rule-bases reached the same 93.4% average recognition rate. This result is below the 97% user acceptance threshold and the results of the $N recognizer [3], but still better, than the accuracy of the Graffiti 2 [4]. In the multi-stroke FUBAR the input parameters of meta-level rules were the number of the sub-strokes representing the characters; the number of sub-strokes is strictly specific for the letters compared to the parameter used in the single-stroke FUBAR. This is why the results with the multi-stroke system did not change like it did in the single-stroke system. It is important to highlight that the results for multi-stroke FUBAR (both flat and hierarchical) could be higher with a better initial rule-base (or an algorithm which identifies it).

5. Conclusions and Future Work

It was shown that after the new FUBAR algorithm was able to recognize multi-stroke alphabets also with a 93.4% average recognition rate. The results indicate that the accuracy should be further increased by the redefinition of the initial rule-base.

Finally in this work, a similar method with the multi-stroke alphabet support using a hierarchical rule-base was presented. The topology of the hierarchy was built based on the number of used strokes. The modified system reached the same accuracy as the original one with the flat rule-base, but in this case, the computational cost of the recognition process was considerably reduced by the limited number of rules to evaluate.

The accuracy of the FUBAR method is moderate compared to the average recognition rate of the modified Palm Graffiti with limited multi-stroke support (known as Graffiti 2) was studied by Költringer and Grechenig in [4] and the $N multi-stroke recognizer introduced by Anthony and Wobbrock in [3].

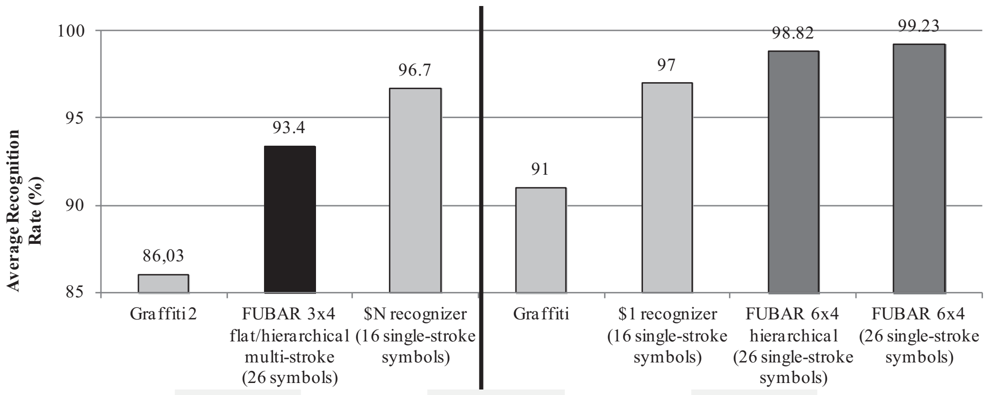

Fig. 6. Average accuracy of various recognition engines

The Graffiti 2 reached only 86.03% accuracy, while 96.7% average recognition rate was achieved for only 16 different single-stroke symbols with the $N recognizer. Both the single--stroke and multi-stroke versions of FUBAR performed well over the results of Graffiti 2. The $N algorithm reached a better average recognition rate compared to the FUBAR method, but the number of symbols was much less and the symbols used during the evaluation of the system were single-stroke. The results are shown in Fig. 6.

This paper was supported by the Hungarian Scientific Research Fund (Hungarian abbreviation: OTKA) K105529, K108405 and TÁMOP-4.2.2.A-11/1/KONV-2012-0012.

References

[1] Kowalski P.A., Kulczycki P., *Data Sample Reduction for Classification of Interval Information using Neural Network Sensitivity Analysis*, Lecture Notes in Artificial Intelligence, Vol. 6304, Springer-Verlag, 2010, 271-272.

[2] Lilik F., Botzheim J., *Fuzzy based Prequalification Methods for EoSHDSL*, Technology, Acta Technica Jaurinensis, Vol. 4, No. 1, Györ 2011, 135-144.

[3] Anthony L., Wobbrock J.O., *A Lightweight Multistroke Recognizer for User Interface Prototypes*, Proc. GI 2010, Ottawa 2010, 245-252.

[4] Költringer T., Grechenig T., *Comparing the Immediate Usability of Graffiti 2 and Virtual Keyboard*, Proc. CHI EA ’04, New York, 2004, 1175-1178.

[5] LaLomia M.J., *User acceptance of handwritten recognition accuracy*, Companion Proc. CHI ’94, New York 1994, 107.

[6] Tormási A., Botzheim J., *Single-stroke character recognition with fuzzy method*, New Concepts and Applications in Soft Computing SCI, Vol. 417, V.E. Balas et al. (eds.), 2012, 27-46.

[7] Tormási A., Kóczy T.L., *Comparing the efficiency of a fuzzy single-stroke character recognizer with various parameter values*, Proc. IPMU 2012, Part I. CCIS, Vol. 297, S. Greco et al. (eds.), 2012, 260-269.

[8] Sugeno M., Griffin F.M., Bastian A., *Fuzzy hierarchical control of an unmanned helicopter*, Proc. IFSA ’93, Seoul 1993, 1262-1265.

[9] Kóczy T.L., Hirota K., *Approximate inference in hierarchical structured rule-bases*, Proc. IFSA ’93, Seoul 1993, 1262-1265

[10] Zadeh L.A., *Fuzzy sets*, Inf. Control, Vol. 8, 1965, 338-353.

[11] Takagi T., Sugeno M., *Fuzzy identification of systems and its applications to modeling and control*, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-15, 1985, 116-132.

[12] Tormási A., Kóczy T.L., *Improving the Efficiency of a Fuzzy-Based Single-Stroke Character Recognizer with Hierarchical Rule-Base*, Proc. 13th IEEE International Symposium on Computational Intelligence and Informatics, Óbuda 2012, 421-426.