Математички Билтен
19 (XLV)
1995 (107–116)
Скопје, Македонија

ISSN 0351–336X

ПРЕПОЗНАВАЊЕ НА РАКОПИСНО НАПИШАНИ ЗБОРОВИ СО ПРЕТХОДНО ТЕНЧЕЊЕ НА ПОТЕЗИТЕ

Силвана Чепреганова

Апстракт

Во трудов е презентиран нов алгоритам за препознавање на ракописно напишани зборови.

Потезите на зборовите прво се тенчат во потези со дебелина од 1 пиксел, а потоа се сегментираат и се препознаваат. Методот на препознавање е статистичко–синтаксички. За пресметување на карактеристиките на ликовите се користат моментите на Зернике.

1. Вовед

Важен проблем во анализата на шеми е автоматското препознавање објекти на сцена, со обзир на нивната позиција, величина и ориентација. Во овој труд предложен е нов алгоритам за препознавање (читање) на ракописно напишани зборови.

Најтежок проблем кој се јавува во препознавањето на ракописно напишани зборови е проблемот на сегментација на зборовите. Во досегашните трудови за препознавање на ракописни букви предложени се некои методи за тенчење (thinning technique). Во овој труд искористена е оваа идеја за тенчење на буквите и реализиран е нов алгоритам за тенчење на букви, како главен предуслов за нивна сегментација.

Влез во алгоритмот се дигитализирани слики од ракописно напишани зборови, чија дебелина на потезите е поголема од еден пиксел.

Зборовите се напишани на лист хартија, снимени се со камера и дигитализирани.

Кога регионот од интерес на сликата е дефиниран, тој може да биде презентиран и опишан на многу начини.

Една популарна, математички комплетна репрезентација е добиена со помош на дводимензионални моменти [1].

Hu [1] дава општа процедура за развивање функции од моменти кои се инваријантни на општите геометриски трансформации.

Фундаментален елемент на сите овие шеми е дефиниција на карактеристики на ликовната презентација и редукција на податоци.

Кога инваријантните карактеристики се добиени, тие се влез во дизајнирано класифицирачко правило за одлучување на начинот на обележување на ликот.

2. Опис на алгоритмот за тенчење на буквите

Во овој алгоритам е реализирана идејата да повлекувањето на потезите со дебелина од еден пиксел во најголема мера личи на пишување со рака. Линијата која се повлекува ќе ја наречеме контурна линија (контура). Контурната линија која се повлекува, воглавно ја следи надворешната граница на потезот со позадината, а точките кои припаѓаат на другата граница на потезот со позадината се маркираат и тие не припаѓаат на контурата. Тоа се точките N_1, N_2, ... на сл. 1 иможеме да ги наречеме „невидливи" точки.

„Невидливите" точки се добиваат како пресек на нормалата во секоја точка од контурната линија со другата граница на потезот со позадината.

Како тангента во секоја точка од контурата се зема правата која проаѓа низ таа точка и е паралелна на векторот одреден со два нејзини соседи.

Алгоритмот почнува со наоѓање на првата контурна точка, која се маркира како прва прекидна точка (P_1 на сл. 1). Од оваа точка почнува повлекувањето на линијата во еден смер. Контурната линија почнува да се повлекува од секоја прекидна точка па сè до крај на потезот.

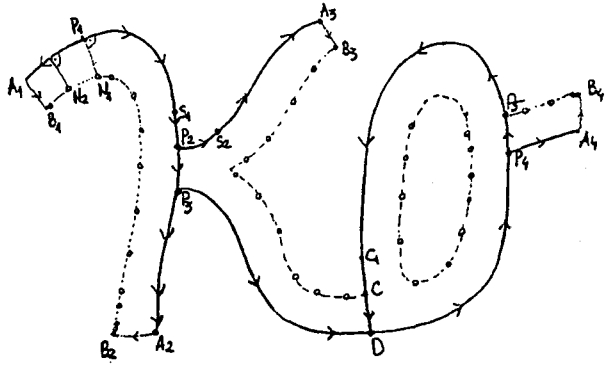
Контурната линија се повлекува вдолж границата на буквите со позадината, а истовремено се маркираат и „невидливите" точки $N_1, N_2, \ldots$ Во секоја точка $P_i(i_2, j_2)$ од контурата се пресметува аголот на нејзиниот тангентен вектор во однос на локалниот координатен систем со центар на $P_i(i_2, j_2)$. Краевите на овој вектор се соседите на точката $P_i(i_2, j_2)$: $P_{i-1}(i_1, j_1)$ и $P_{i+1}(i_3, j_3)$, т.е. $\overrightarrow{T} = \overrightarrow{P_{i-1}P_{i+1}}$. Контурната линија се повлекува сè додека аголот

на тангентниот вектор е поголем или еднаков на аголот на тангентниот вектор во претходната точка (во случај кога позадината е од десна страна на насоката на повлекувањето на линијата, а кога оваа насока е спротивна, важи обратното: овој агол треба да биде помал од аголот на тангентниот вектор во претходниот сосед); или додека не се дојде до „невидлива" точка.

Со помош на овие тангенти, всушност ја установуваме промената на кривината на кривата.

Секогаш кога се доаѓа до точка во која е пореметен овој однос меѓу аглите на тангентите, се врши процедура која испитува да не е дојдено до почеток на нов нов потез. На пример таков е случајот во точката P_2: се мери аголот меѓу векторите $\overrightarrow{S_1P_2}$ и $\overrightarrow{P_2S_2}$. Прагот на овој агол овде е земено да биде $5\pi/12$. Ако овој агол е поголем од овој праг се установува почеток на нов потез, а во спротивен случај контурата се повлекува понатаму.

Ако се востанови почеток на нов потез, се маркира точката во која е пореметен односот на тангентните вектори (тоа се точките P_2 и P_4 на сл. 1), а контурната линија се продолжува понатаму во ист правец. (На сл. 1 правецот P_2P_3 и P_4P_5.)



Сл. 1

Повлекувањето на линијата во внатрешноста на буквите се врши така да нејзините точки се секогаш максимално оддалечени од правата P_2S_1 или минимално оддалечени од точката S_1, сè додека не се дојде до гранична точка на буквата со позадината, која се маркира како следен прекид. (На сл. 1 тоа се точките P_3 и P_5.)

Во точките P_3 и P_5 се продолжува повлекувањето на линијата во ист смер, сè додека не се дојде до крај на потезот. Тогаш овде се можни два случаја:

Прв случај е кога се доаѓа до точките B_1, B_2, , B_3 и B_4 кои се „невидливи" и тогаш се бришат сите точки од A_1 до B_1, од A_2 до B_2, од A_3 до B_3 и од A_4 до B_4 (т.е. бришење онолку точки, колку што е најголемата дебелина на потезот).

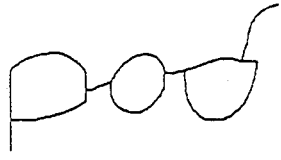
Втор случај е кога се доаѓа до точка C и треба потезот да се продолжи до веќе повлечената линија.

Овие два случаја се востановуваат, така што после точките B_1, B_2, B_3, B_4 или C се испитуваат следните неколку пиксели во правец на веќе повлечениот потез: правците A_1B_1, A_2B_2, A_3B_3, A_4B_4 или C_1C. Ако овие пиксели не припаѓаат на ликот (како кај B_1, B_2, B_3 и B_4), тогаш тоа е првиот случај и повлечената линија од A_i до B_i ($i = 1, \ldots, 4$) се брише; а ако овие точки припаѓаат на ликот (како кај точката C), тогаш тоа е вториот случај и потезот се продолжува до точката D.

Резултатите од овој алгоритам се на сл. 3, 5, 7, 9 и 11.

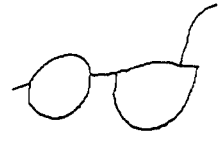
Резултатот е всушност еден регион со дебелина од еден пиксел, кој понатаму полесно се сегментира а со тоа и полесно се препознава.

Сл. 2



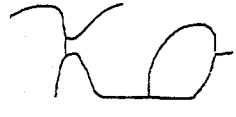
Сл. 3

Сл. 4



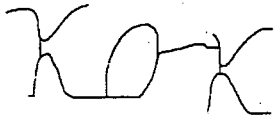
Сл. 5

Сл. 6

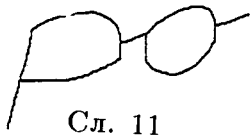


Сл. 7

Сл. 8



Сл. 9



Сл. 11

3. Опис на алгоритмот за препознавање на ракописно напишани зборови со потези со дебелина од еден пиксел

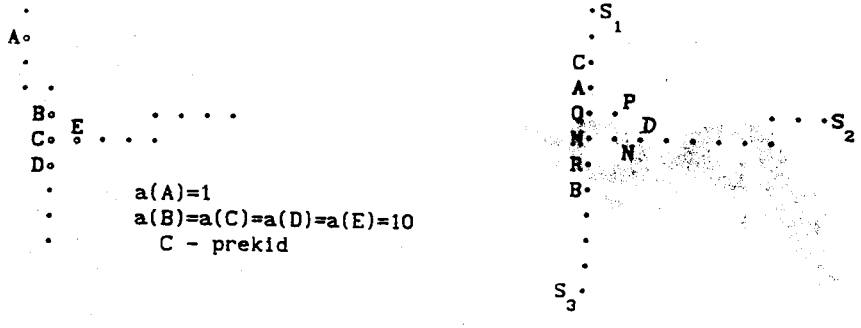
Во оваа точка ќе биде опишан новиот алгоритам што се предлага за препознавање на ракописно напишани зборови.

Кога овие зборови се напишани со потези со дебелина од еден пиксел, тие се погодни за сегментација.

Линијата со која потезите се повлечени, е формирана од пиксели кои имаат по два соседа. Овде е предложено сегментацијата на потезите да биде на местата на спојување на две или повеќе линии. Точката во која се спојуваат две или повеќе линии има три или четири соседи, па овие точки може да бидат места за сегментација на потезите. Начнот на сегментација на потезите е следниот: по линиите од зборовите се „трага" сè додека пикселите што ја формираат линијата имаат два соседа. Кога се наидува на пиксел кој има три соседа или само еден, тогаш тоа е крај на потезот. Точката која има три соседа се маркира како *прекидна точка.* Таа има соседи од кои започнуваат нови потези. Секој потез се наоѓа меѓу две прекидни точки, или меѓу прекидна точка и точка која има само еден сосед. Секој пиксел (x, y) со само два соседа, се маркира во матрица $a(x, y) = 1$, а ако се дојде до прекидна точка, таа се маркира со $a(x, y) = 10$. Соседот на прекидот кој е веќе „преjден" се маркира исто со $a(x, y) = 10$. Соседите на прекидот од кои поаѓаат следните потези не се маркирани и тие се маркираат во почетокот на трагањето за нов потез. Значи, една прекидна точка е почеток на нови потези и од неа се поаѓа сè додека сите нејзини соседи не постанат маркирани со $a(x, y) = 10$

(сл. 12а).

Програмот работи сè додека не се изцрпат сите прекидни точки.



а) б)

Сл. 12

Кај почетокот на трагање за секој потез настануваат следните ситуации:

Прв случај е кај потезот QS_1, кој почнува со прекидната точка Q (сл. 12б) кога точката A, која е втор пиксел од овој потез, има само два соседа (Q и C) и точката C, која е трет пиксел од овој потез има само два соседа. Во овој случај трагањето продолжува сè додека не се дојде до крај на потезот (тоа е друга прекидна точка или точка која има само еден сосед).

Втор случај е кога потезот QS_2, кога вториот пиксел од овој потез (точката P) има два соседа (Q и N), а третиот пиксел од овој потез (точката N) има три соседи (M, P и D). Во овој случај, од овие три соседи, за продолжување на потезот се бира онаа точка која не е сосед на прекидот Q (а тоа е D). D има само два соседа и трагањето продолжува понатаму.

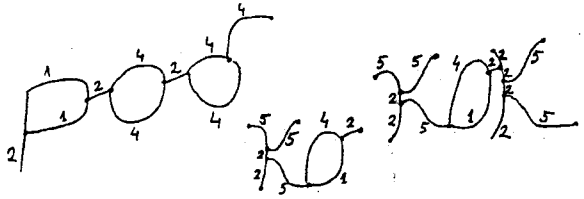
Трет случај настанува кај потезот QS_3: кога вториот пиксел од овој потез (точката M) има три соседа: (Q, N и R). Во овој случај, од овие три соседа, за следен (трети) пиксел од овој потез се бира оној кој има само еден заеднички сосед со прекидната точка Q. Тоа во овој случај е точката R. (N има два соседа заеднички со Q, а Q има три). R има само два соседа, па трагањето се продолжува понатаму сè до крајот на потезот.

Секој сегментиран потез добива име, т.е. секој потез претставува посебна класа облици. Секоја буква е „формирана" од неколку класи и тие можат да бидат застапени во буквата на разни начини: како на пример буквата P се состои од од две класи: тоа

се потезите со име „1" и „2" како на сл. 13.

Може да запишеме $P = (1, 2, 1)$. Буквата *о* може да биде прочитана, ако се препознаени класите „2", „4" или „1" и тоа на следниот начин: или како низата (2, 4, 4, 2) или (4, 1, 2). Буквата *б* се препознава ако е препознаена класата „4" и тоа три пати едно подруго. Буквата *к* се препознава во два случаја: или како низа (5, 5, 5, 2, 2) или (2, 2, 2, 2, 5, 5).

Начинот на препознавање на потезите е според добро проверен модел: нека m е број на моменти кои се користат како карактеристики на ликот (потезот) и нека $x^1, x^2, \ldots, x^m$ се дадените моменти. Точката $X = (x^1, x^2, \ldots, x^m)$ во m–димензионалниот простор претставува репрезент на ликот (потезот). Во трудов се користени Зерникевите моменти кои претставуваат посебна класа ортогонални моменти, базирани на теоријата на ортогаонални полиноми.



Сл. 13

Моментите на Зернике [9] се инваријантни само на ротација. За да се добие инваријантност на транслација и скалирање, ликот се нормализира со помош на неговите регуларни моменти. Потоа Зерникевите моменти се извлекуваат од скалираниот и транслираниот нормализиран лик.

Зернике воведува ново множество комплексни полиноми кои формираат комплетно ортогонално множество над кругот $x^2 + y^2 = 1$. Нека множеството на овие полиноми е означено со $\{V_{nm}(x, y)\}$. Формата на овие полиноми е:

$$V_{nm}(x, y) = V_{nm}(\rho, \theta) = R_{nm}(\rho) \exp(im\theta) \tag{1}$$

$n \geq 0$, $m \in \mathbf{Z}$, $n - |m|$ е парно, $|m| \leq n$, ρ е должина на радиус векторот на точката (x, y), θ е агол меѓу векторот ρ и x оската.

R_{nm} е радијален полином дефиниран на следниот начин:

$$R_{nm}(\rho) = \sum_{s=0}^{\frac{n-|m|}{2}} \frac{(-1)^s * (n-s)!}{s! \left(\frac{n-|m|}{2} - s\right)! \left(\frac{n+|m|}{2} - s\right)!} \rho^{n-2s}. \tag{2}$$

Јасно $R_{n,-m} = R_{nm}$.

Овие полиноми се ортогонални, па важи:

$$\iint\limits_{x^2+y^2 \leq 1} [V_{nm}(x, y)]^* V_{pq}(x, y)\, dx\, dy = \frac{\pi}{n+1} \delta_{np}\, \delta_{mq} \tag{3}$$

каде $\delta_{ab} = \begin{cases} 1 \text{ за } a = b \\ 0 \text{ во друг случај,} \end{cases}$ $*$ означува конјугирано комплексно.

Зерникевите моменти претставуваат проекција на функцијата на ликот на овие ортогонални функции.

Зерникевите моменти од ред n со повторување m за непрекинатата функција $f(x, y)$ се дефинирани:

$$A_{nm} = \frac{n+1}{\pi} \iint\limits_{x^2+y^2 \le 1} f(x, y) V^*_{nm}(\rho, \theta)\, dx\, dy\,. \tag{4}$$

За дигитални ликови интегралите се заменуваат со суми:

$$A_{nm} = \frac{n+1}{\pi} \sum_x \sum_y f(x, y)\, V^*_{nm}(\rho, \theta) \qquad x^2 + y^2 \le 1\,. \tag{5}$$

За пресметување на Зерникевите моменти на одреден лик, координатниот почеток се зема во центарот на ликот (негово тежиште) и моментите се пресметуваат за ликот сместен во единична кружница $x^2 + y^2 = 1$.

За секој потез (класа i) се меморира следното знаење:

$$\text{Име}, \qquad X_i, N_i \tag{6}$$

каде *име* е назив на потезот (класата 1). Во слючајов на сл. 13 називи се: „1", „2", „3", ... X_i е векторот од карактеристики (моменти) за потезот i $X_i = (x_i^1, x_i^2, \ldots, x_1^m)$, N_i е бројот на примероци кои припаѓаат во класата i, а $x^k (k = 1, \ldots, m)$ се пресметуваат по формулата (5).

Се формира датотека од вакви слогови (6) која ги чува податоците за сите научени класи облици i.

Во почетокот не постои никакво знаење за објектите кои се препознаваат, тие постепено се учат.

Нека $\{X_i\}$ $i = 1, 2, \ldots, N)$ се точки кои презентираат објекти кои се веќе научени (X_i е векторот од Зерникеви моменти за објектот i кои се пресметуваат по формулата (5), а N_i нека е бројот на примероци од i–тиот објект кој е веќе научен. После процесот на учење за i–тиот објект, N_i се заменува со $N_i + 1$ и X_i со

$$\frac{N_i X_i + X}{N_i + 1} = \overline{X}_i \tag{7}$$

каде X е презентација на новиот објект. Ова ново $\overline{X}$ е оочигледно средина од сите $N_i + 1$ примероци кои се веќе научени.

Ако се научени N класи, тогаш за секој нов потез кој треба да се препознае се пресметуваат растојанијата

$$d_i = \sqrt{\sum_{p=1}^{m} \left(\overline{x}^p - x^p\right)^2}\,, \qquad i = 1, \ldots N\,.$$

Најмалото од овие растојанија $d_k = \min\limits_i d_i$ ја дава класата во која може да припадне ликот. Ако ликот припаѓа на класата k,

таа се модифицира со (7), ако не, програмот испишува „не можам да препознаам". Се впишува името на класата и се пребарува во датотеката, да не е тоа веќе некоја научена класа. Ако е така, од датотеката се зема слогот кој има знаење за таа класа и тој се модифицира со (7), а ако не е е пронајдена класа со тоа име, се формира нова класа, т.е. нов слог во датотеката.

Од сликите 3, 5, 7 и 11 погрешно се препознаени само два потеза.

Резултатите од сегментацијата на потезите кај зборот *ро* се на сл. 14.

Сл. 14

Литература

[1] Aggarwal, J. K., Duda, R., Rosenfeld, A: *Computer Methods in image Analysis.*

[2] Rosenfield, A: *Digital Picture Analysis*, New Work, 1976

[3] Paul Cohen & Edwards A. Feiganbaum: III *Handbook of Artifikal Intelligence: Vision.*

[4] Фу, К.: *Структурные методы с распознавании образов.* IEEE Transitions on PAMI 84.

[5] Furuta, M., Kase, N., Emori, S.: *Segmentation and Recognition of Symols For Handwritten Piping & Instrument Diagram*, IEEE Transitions on PAMI 84.

[6] Amin, A., Masini, G. & Halton, P., J.: *Recognition of Haudwritten Arabic Words and Sentences*, IEEE Transitions on PAMI 84.

[7] Чепреганова, С.: *Сегментација на региони и развој на софтвер за издвојување облици од сложена сцена*, Магистерски труд, Белград, 1994.

[8] Shunji Mori, Tsugako Sakakura: *Line filtering and its application to stroce segmentation of handprinted Chinese Characters*, Shinjuku, Tokyo 160, Japan.

[9] Alireza Khotanzad & Yaw Hua Hong: *Invariant Image Recognition by Zernike Moments*, IEEE Transitions On Pattern Analyses And Machnine Intelligence Vol. 12 No. 5 May, 90.

[10] Cho–Huakten & Roland. T. Chin: *On image Analysis by the Methods of Moments*, IEEE Transitions on PAMI Vol. 10. No. 4, July, 1988.

RECOGNITION OF HANDWRITTEN WORDS WITH PREVIOUS THINNING OF THE STOKES

Silvana Chepreganova

Summary

This work involves development of software for recognition (reading) of hanwritten words.

A new algorithm for thinning strokes thicker than one pixel, is suggested. Those strokes later can be easily segmented.

This idea is to draw a line along the border of the stroke with the backround and then measure the curve of that line. If the curve passes a certain limit, it indicates the begining of a new stroke. These spots are marked as interruptions and each stoke starts with one and finishes with another interruption spot – a beginning of a new stroke.

This results into separation of a single region, one pixel thick, wich can be further segmented easily. The spots where two or three different strokes join are chosen for segmentation. The strokes are first normalised and named, i.e. each stroke presents a separate class that should be learned in course of recognition.

Features used for presentation of these forms (strokes) are the Zernike's moments. In order to recognize the letters, it is necessary to learn in advance the possible classes they might be composed of.

It is possible to develop this programme futher, so that it can be used for reading other types of letters.

Gradezhen fakultet
p. fah 560

91000 Skopje,

Makedonija