SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
Stavebná fakulta

Evidenčné číslo: SvF-5343-67725

Analýza písma matematickými metódami spracovania obrazu

Diplomová práca

Študijný program:	matematicko-počítačové modelovanie
Číslo študijného odboru:	1114
Názov študijného odboru:	9.1.9 aplikovaná matematika
Školiace pracovisko:	Katedra matematiky a deskriptívnej geometrie
Vedúci záverečnej práce:	Ing. Michal Smíšek
Konzultant:	prof. RNDr. Karol Mikula, DrSc.

Bratislava 2014 **Bc. Monika Miklašová**

Čestné prehlásenie

Vyhlasujem, že som diplomovú prácu vypracovala samostatne pod vedením vedúceho diplomovej práce na základe získaných poznatkov počas štúdia a použitím nižšie uvedenej literatúry.

Bratislava 21. mája 2014 .....

Vlastnoručný podpis

Poďakovanie

Chcela by som sa poďakovať všetkým, ktorí ma pri vypracovaní diplomovej práce usmerňovali svojimi pripomienkami a postrehmi.
Moja vďaka patrí hlavne môjmu vedúcemu diplomovej práce, Ing. Michalovi Smíšekovi a konzultantovi prof. RNDr. Karolovi Mikulovi, DrSc., za profesionalitu, vedenie práce, ochotu a odbornú pomoc.

Annotation

This work deals with mathematical methods of image processing in order to analyze a handwritten text. Conventionally used methods use blurring of 2D image of a letter and then compare it to an atlas of other, already blurred letters. Unlike conventional methods, we aim to represent a letter as a curve, to obtain 1D descriptor of the curve, and also to create a concept of an atlas for classification. The reduction of dimensionality of this task results in possibility to formulate simpler and computationally more efficient methods to compute distance between two letters by using a simple metric.

A feasible way to represent a letter by a closed curve is considering outer boundary of this letter. We obtain a descriptor of this curve by using a signed curvature of the curve. In order to have a notion of blurring of the descriptor, to overcome imperfections in the written letters, we use so-called curvature diffusion.

Since we consider 1D descriptor, there are multiple ways to formulate a metric to compute distances between descriptors of various letters. As a first metric we consider absolute value of difference of two descriptors. As a second metric we compute distance function from the graph of one descriptor and evaluate it in discrete points of the second descriptor. As a third metric we compute distance function from the graph of one descriptor and evaluate it in points of graph of the second descriptor. This way we demonstrate, it is easy to formulate an arbitrary metric, once we have reduced the problem to 1D.

Anotácia

Práca sa zaoberá matematickými metódami spracovania obrazu za účelom analýzy ručne písaného textu. Tradične používané metódy sú založené na zhladení písmen v 2D obrázku a ich následnom porovnaní s atlasom iných zhladených písmen. Na rozdiel od tradičných metód je našim cieľom reprezentovať písmeno pomocou krivky na získanie 1D deskriptora týchto kriviek a taktiež vytvoriť návrh atlasu na ich klasifikáciu. Redukcia dimenzie nám umožňuje formulovať jednoduchšie a výpočtovo efektívnejšie metódy na výpočet vzdialenosti medzi dvomi krivkami.

Spôsob ako reprezentovať písmeno pomocou krivky je nájdenie vonkajšej hranice daného písmena. Ako deskriptor tejto krivky uvažujeme znamienkovú krivostnú funkciu. Na zhladenie a odstránenie nedokonalostí používame krivostnú difúziu.

V 1D existuje niekoľko spôsobov, ako sformulovať metriku pre výpočet

vzdialenosti medzi deskriptormi jednotlivých písmen. V práci sa zaoberáme výpočtom absolútnej hodnoty rozdielu dvoch deskriptorov (prvá metrika), výpočtom vzdialenostnej funkcie z grafu jedného deskriptora a následným vyhodnotením týchto hodnôt v bodoch grafu druhého deskriptora (druhá metrika) a výpočtom vzdialenostnej funkcie z grafu jedného deskriptora a vyhodnotením týchto hodnôt v grafe druhého deskriptora (tretia metrika).

Týmto spôsobom demonštrujeme, že akonáhle zredukujeme problém na 1D, vieme sformulovať rôzne metriky.

Obsah

Úvod **1**

1. Postup práce **2**

2. Definícia písmena pomocou deskriptora **3**
2.1 Thresholding 3
2.2 Dilatácia 3
2.3 Získanie krivky z hranice písmena 4
2.4 Krivosť 5

3. Krivostná difúzia **7**
3.1 Numerická implementácia krivostnej difúzie 8
3.2 Thomasov algoritmus 9
3.3 Sherman-Morrisonova formula 10
3.3.1 Postup v programe 10

4. Invariantnosť voči počiatočnému bodu **12**

5. Metriky **14**
5.1 Rozdielová metrika 14
5.2 Bodovo-vzdialenostná metrika 15
5.2.1 Vzdialenostná funkcia 15
5.2.2 Funkcionalita Bodovo-vzdialenostnej metriky 17
5.3 Grafovo-vzdialenostná metrika 18

6. Popis softvéru **19**
6.1 Opis grafického rozhrania 20

7. Experimenty **22**
7.1 Porovnanie vplyvu krivostných parametrov na deskriptor . . . 22
7.1.1 Porovnanie deskriptora pri zmene veľkosti časového kroku τ 22
7.1.2 Porovnanie deskriptora pri zmene počtu časových krokov 23

7.1.3 Porovnanie deskriptora pri zmene veľkosti parametra ϵ 24
7.2 Porovnanie deskriptorov modifikovaného písmena 25
7.2.1 Porovnanie deskriptora pri zmene veľkosti písmena . . 25
7.2.2 Porovnanie deskriptora pri zmene rotácie písmena . . 26
7.2.3 Porovnanie deskriptorov rôznych podôb písmena *a* . . . 27
7.3 Nevýhody krivostného deskriptora 28

8. Záver **29**

Literatúra **30**

Úvod

V dnešnej dobe sú aplikácie rozoznávajúce text často používané. Všeobecný názov takéhoto typu aplikácii je OCR (optical character recognition). Používa sa na zjednodušenie a urýchlenie činností ako napríklad transformácia ručne písaného dokumentu do počítačom rozoznateľného textu. V automobilovom priemysle je použitý v navigáciach a v senzoroch rozoznávajúcich cestné značky, čím prispieva k zvýšeniu bezpečnosti na cestách.
Cieľom tejto práce bolo rozoznanie ručne písaného textu pomocou metód spracovania obrazu, pričom sme sa chceli vyhnúť tradičným metódam využívajúcim napríklad neurónové siete či porovnávanie zhladených písmen z textu s databázou iných zhladených písmen [1]. V práci sme popísali metódy, pomocou ktorých sme sa chceli dopracovať k vhodnému deskriptoru dostatočne opisujúcemu písmeno. Ďalej sme vytvorili metódy, ktoré pracujú s deskriptormi písmen a odhadujú ich podobnosť. Na záver sme popísali grafické rozhranie softvéru a pomocou experimentov sme porovnali deskriptory písmen.

1. Postup práce

Naším prvým cieľom bola reprezentácia vstupného 2D písmena pomocou krivky. Náš počiatočný zámer bol vypočítať skeleton a pomocou neho definovať 2D písmeno zo vstupného obrázka. Skeleton môžeme vypočítať viacerými postupmi - morfologické operácie opísané v [2], pomocou funkcie vzdialenosti [3] alebo krivkovou evolúciou hranice písmena. Tu sme si uvedomili, že bude vhodnejšie za krivkovú reprezentáciu uvažovať hranicu písmena, ktorú vieme definovať ako parametrickú (uzavretú) krivku, čo pri skeletone vzhľadom k jeho topológii nevieme.

Ďalším cieľom bolo zvoliť deskriptor, ktorý nám poskytne vhodnú informáciu o krivke reprezentujúcu písmeno. Najčastejšie používaným deskriptorom v spracovaní obrazu je krivosť, preto sme si ju zvolili za deskriptor aj my [2].

Výstupom výpočtu krivosti pre hranicu písmena je diskrétna funkcia, ktorú je potrebné zhladiť. Na zhladenie sme zvolili tzv. krivostnú difúziu [4], ktorá nám zároveň umožnila redistribúciu bodov krivky a tým robustnejší výpočet krivosti.

Ďalej sme navrhli metriky na výpočet vzdialenosti deskriptorov jednotlivých písmen, pričom sme postupovali od najjednoduchšej, ktorá vykonávala ich odčítanie až po zložitejšie využívajúce funkciu vzdialenosti ku grafu funkcie krivosti.

Celý postup sme implementovali v softvéri.

2. Definícia písmena pomocou deskriptora

Ako najefektívnejšiu metódu na získanie hranice písmena sme zvolili metódu využívajúcu dilatáciu. Vzľadom k tomu, že dilatácia pracuje najlepšie s binarizovaným obrázkom, museli sme najprv na vstupný obrázok aplikovať metódu zvanú thresholding.

2.1 Thresholding

Thresholding patrí medzi najjednoduchšie a základné metódy spracovania obrazu [2]. Pomocou thresholdingu binarizujeme šedotónové obrázky a to tak, že sa zvolí hodnota, na základe ktorej thresholding pracuje. Všetky hodnoty pixelov vačšie alebo rovné ako daná hodnota sa nastavia na 255 a všetky menšie na 0. Keďže nevieme dopredu, či bude vstupný obrázok šedotónový, predtým ako naňho použijeme thresholding ho automaticky upravíme na šedotónový.

2.2 Dilatácia

Dilatácia je ďalšia základná operácia v spracovaní obrazu [2]. Pracuje pomocou ľubovoľne zvoleného štrukturálneho elementu. Dilatácia je definovaná nasledovne:

$$A \oplus B = \{z|(\hat{B})_z \cap A \neq \emptyset\} \ , \tag{2.1}$$

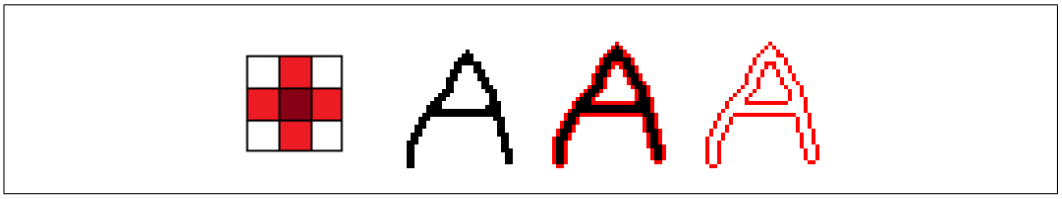
kde A je binárny obrázok, B je štrukturálny element a

$$\hat{B} = \{\omega|\omega = -b, pre\ b \in B\} \tag{2.2}$$

$$(\hat{B})_z = \{c|c = b + z, pre\ b \in B\} \tag{2.3}$$

Dilatácia je teda definovaná ako zjednotenie množiny A a všetkých posunov štrukturálneho elementu vzľadom na A.

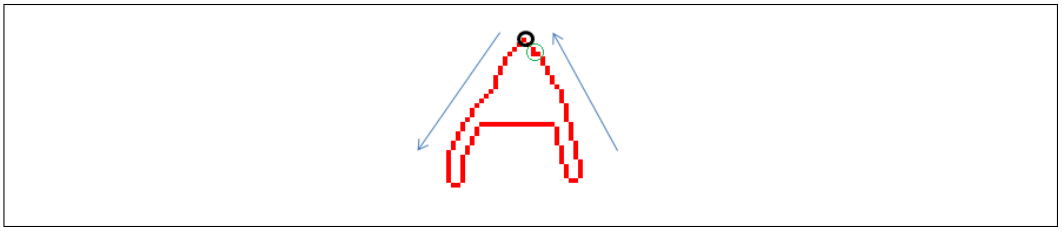
My sme zvolili 4-susedový štrukturálny element veľkosti 3×3. Ako vidieť na obrázku 2.1, hranicu písmena sme dostali odrátaním pôvodného písmena od písmena s dilatáciou.



Obr. 2.1: Štrukturálny element, pôvodné písmeno, dilatácia, hranica písmena

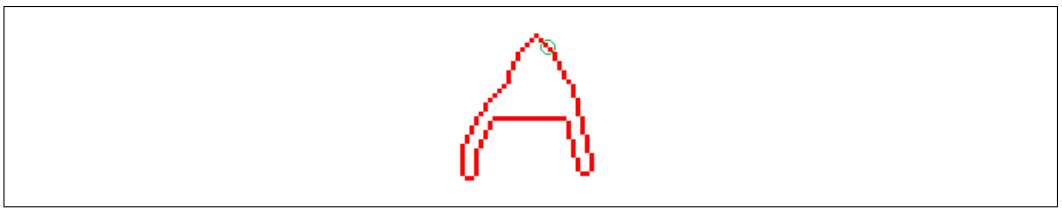
2.3 Získanie krivky z hranice písmena

Počiatočný bod krivky sme zadefinovali ako najvyššie položený pixel patriaci hranici písmena. Od tohto pixela sme postupovali proti smeru hodinových ručičiek s tým, že sme vybrali len najbližší pixel v tomto smere a postupne sme ukladali body v danom poradí až kým sme sa nedostali späť k počiatočnému pixelu.



Obr. 2.2: Smer výberu pixelov, počiatočný bod je označený čiernym krúžkom. Zelený krúžok označuje oblasť, v ktorej niektoré pixely vo finálnej hranici nebudeme uvažovať.

Ako vidieť na obrázku 2.3, pomocou tohto postupu sa taktiež odstránili nepotrebné pixely.



Obr. 2.3: Finálna hranica písmena. Zelený krúžok označuje miesto zmeny v porovnaní s predošlým obrázkom 2.3.

Vzhľadom na to, že sme spočiatku nevedeli aký typ metrík budeme používať a čo všetko budú vyžadovať, dĺžka kriviek (počet pixelov reprezentujúcich

hranicu písmena) bola pre každé písmeno rôzna. Avšak, vzhľadom na neskoršie poznatky sme sa rozhodli upraviť krivky získané z hraníc písmen na rovnaký počet bodov N, ktorý bol definovaný podľa krivky s najväčším počtom pixelov N. Ďalej sme potrebovali správne určiť dĺžku kroku h. Preto sme si vypočítali celkovú dĺžku L upravovanej krivky vzhľadom na euklidovslú vzdialenosť bodov, v tomto prípade stredov pixelov, a vydelili sme ju dĺžkou N. S daným krokom sme sa pohybovali po krivke dĺžky L a postupne sme ukladali nové hodnoty.

2.4 Krivosť

Ďalším krokom našej práce bol výpočet krivosti, ktorú sme si zvolili za deskriptor kriviek opisujúcich písmeno. Krivosť definujeme nasledovne:

$$\kappa = \frac{1}{R} \ , \tag{2.4}$$

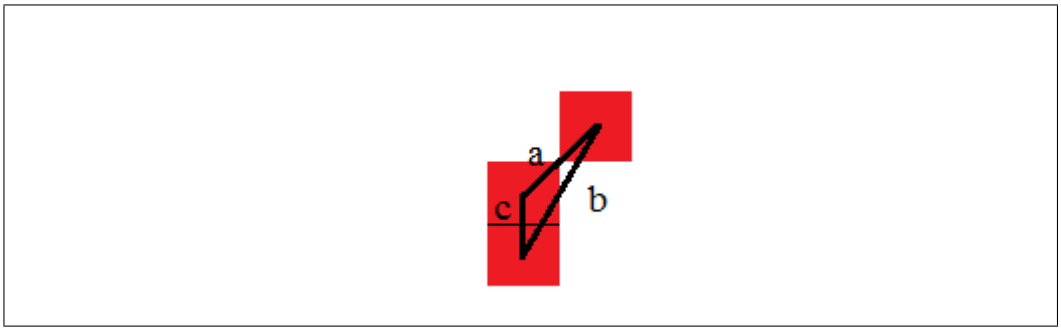
kde R je polomer kružnice opísanej trojuholníku so stranami a, b, c, viď obrázok 2.4. Veľkosť polomeru opísanej kružnice vyjadruje nasledovný vzťah

$$R = \frac{a}{2\sin\alpha} = \frac{b}{2\sin\beta} = \frac{c}{2\sin\gamma} \ , \tag{2.5}$$

kde a, b a c sú dĺžky strán a α, β a γ sú ich protiľahlé uhly v príslušnom trojuholníku. Dĺžky strán trojuholníka sme vypočítali použitím Pytagorovej vety a použitím kosínusovej vety sme získali príslušné uhly, napr.

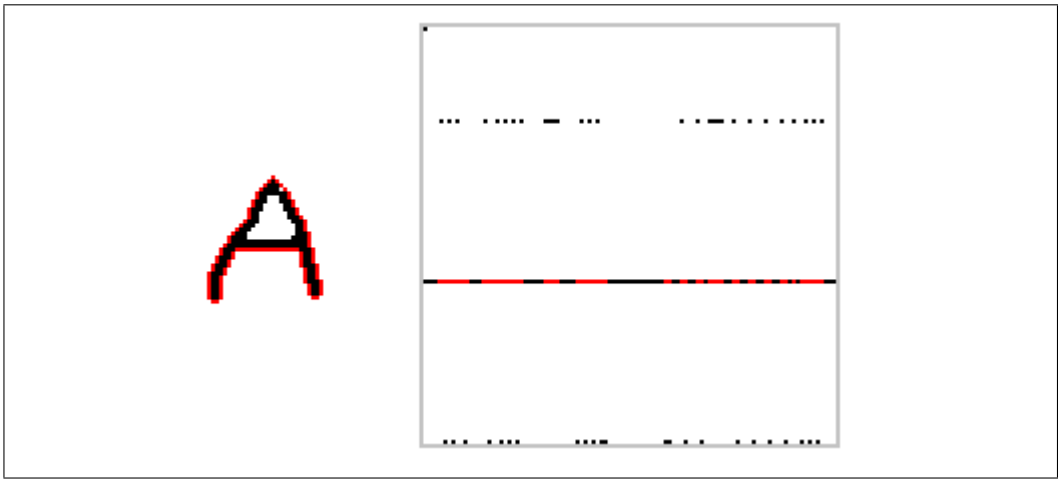
$$\alpha = \arccos(-\frac{a^2 - b^2 - c^2}{2bc}) \ . \tag{2.6}$$

Krivosť sme vzhľadom na ďalšiu prácu ukladali do poľa aj v preškálovanom



Obr. 2.4: Zobrazenie trojuholníka (časť z hranice písmena), ktorému opisujeme kružnicu na nájdenie krivosti pre stredný pixel.

tvare vzhľadom na dĺžku danej krivky, teda najvyššiu hodnotu krivosti sme nastavili na veľkosť dĺžky krivky (počet pixelov krivky), najnižšiu na nulu a ostatné odvíjajúc sa od týchto hodnôt.

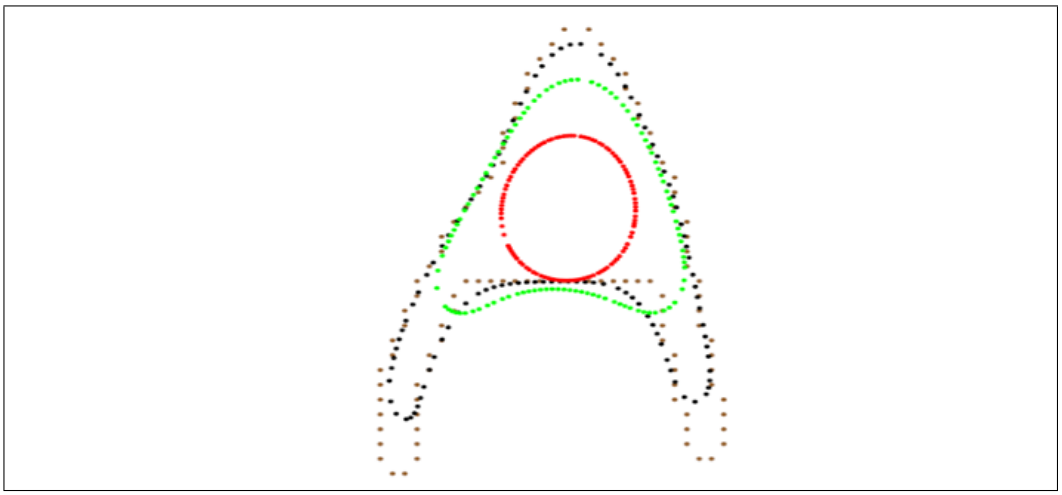


Obr. 2.5: Originálne písmeno s vypočítanou hranicou, graf krivosti (červená čiara znázorňuje x-ovú os v bode y=0, najvyššia hodnota krivosti je rovná 1 a najnižšia je rovná -0.63). Vidíme, že graf funkcie nadobúda konečný počet hodnôt - v tomto prípade 4. Dôvodom je diskrétna povaha postupnosti pixelov reprezentujúca hranicu písmena.

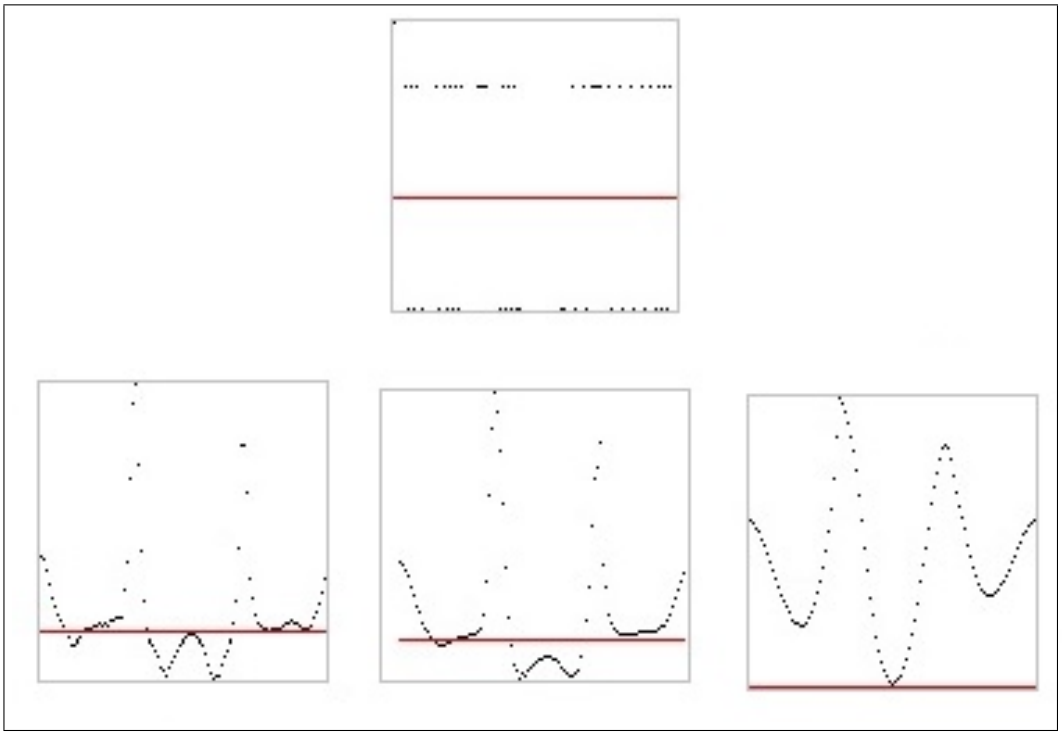
Keďže vidíme, že graf krivosti krivky, zobrazený na obrázku 2.5, nadobúda iba 4 rôzne hodnoty, na prvý pohľad chaoticky usporiadané, nejaví sa v takomto stave ako vhodná voľba deskriptora. Riešením by bolo a) zvoliť presnejší (viacbodový) systém pre odhad krivosti v danom bode, alebo b) realizovať evolúciu krivky, ktorá spôsobí jej postupné zhladzovanie. Zvolili sme druhý postup.

3. Krivostná difúzia

Ako vidieť na obrázku 2.5, diskrétny tvar krivosti nie je hladký a preto je potrebné krivku zhladiť. Zhladenie sme vyriešili pomocou krivostnej difúzie. Krivostná difúzia je veľmi podobná vedeniu tepla a pracuje s krivosťou krivky. S narastajúcou krivosťou narastá aj rýchlosť difúzie, teda časti s najväčšou krivosťou sa vyhladzujú rýchlejšie ako ostatné. Ako vidieť na obrázkoch 3.1 a 3.2, konkávne časti krivky sa postupne posúvajú smerom von a konvexné smerom dovnútra krivky. Krivka sa s narastajúcim počtom krokov krivostnej difúzie zaobluje a zmršťuje.



Obr. 3.1: Počiatočná hranica písmena (hnedá farba), hranica po 5 (modrá), po 10 (zelená), po 25 krokoch krivostnej difúzie (červená).



Obr. 3.2: Krivostná krivka určená z hranice písmena, krivosť po 5, po 10, po 25 krokoch krivostnej difúzie.

3.1 Numerická implementácia krivostnej difúzie

Model, ktorý používame, je reprezentovaný krivostnou difúziou s redistribúciou [4] a je daný nasledujúcou diferenciálnou rovnicou:

$$x_t = \epsilon x_{ss} + \alpha x_s \ . \tag{3.1}$$

Nasledovná rovnica definuje diskrétnu semi-implicitnú schému konečných objemov :

$$\begin{aligned} \frac{h_i^m + h_{i+1}^m}{2}\frac{x_i^{m+1} - x_i^m}{\tau} = \quad & \epsilon\left(\frac{x_{i+1}^{m+1} - x_i^{m+1}}{h_{i+1}^m} - \frac{x_i^{m+1} - x_{i-1}^{m+1}}{h_i^m}\right) + \\ & \alpha\left(\frac{x_{i+1}^{m+1} - x_{i-1}^{m+1}}{2}\right) \ , \end{aligned} \tag{3.2}$$

kde $i = 1, ..., N$; $m = 0, ..., m$, pričom x_i^m reprezentuje pozíciu i-teho bodu krivky v m-tom čase.

Rovnicu (3.2) vieme zapísať v skrátenom tvare:

$$-A_i^m x_{i-1}^{m+1} + B_i^m x_i^{m+1} - C_i^m x_{i+1}^{m+1} = D_i^m \ , \tag{3.3}$$

kde D_i^m je pravá strana,

$$A_i^m = -\frac{\alpha_i^m}{2} + \frac{\epsilon_i^m}{h_i^m},\ C_i^m = \frac{\alpha_i^m}{2} + \frac{\epsilon_i^m}{h_{i+1}^m}\ a\ B_i^m = \left(H_i^m + A_i^m + C_i^m\right)\ , \quad (3.4)$$

pričom

$$H_i^m = \frac{h_{i+1}^m + h_i^m}{2\tau}. \quad (3.5)$$

Systém (3.2) sme riešili pomocou Thomasovho algoritmu a Sherman-Morrisonovej formuly vzhľadom k tomu, že je cyklický a trojdiagonálny.

3.2 Thomasov algoritmus

Thomasov algoritmus je zjednodušená forma Gaussovej eliminácie používaný na riešenie trojdiagonálnych systémov rovníc. Ako prvé počíta nové hodnoty nasledovným spôsobom:

$$c_i' = \begin{cases} \dfrac{C_i}{B_i} & ;\ \ i = 1 \\ \dfrac{C_i}{B_i - A_i c_{i-1}'} & ;\ \ i = 2, 3, \ldots, n-1 \end{cases} \quad (3.6)$$

$$d_i' = \begin{cases} \dfrac{D_i}{B_i} & ;\ \ i = 1 \\ \dfrac{D_i - A_i d_{i-1}'}{B_i - A_i c_{i-1}'} & ;\ \ i = 2, 3, \ldots, n, \end{cases} \quad (3.7)$$

kde A_i, B_i a C_i sme vypočítali z rovníc (3.4),
a pomocou spätnej substitúcie počíta výsledné hodnoty:

$$x_n = d_n'\ x_i = d_i' - c_i' x_{i+1} \qquad ;\ i = n-1, n-2, \ldots, 1. \quad (3.8)$$

3.3 Sherman-Morrisonova formula

Sherman-Morrisonova formula [5] sa v lineárnej algebre využíva na riešenie cyklických trojdiagonálnych systémov.

$$\begin{bmatrix} b_1 & c_1 & & & \alpha \\ a_2 & b_2 & c_2 & & \\ & a_3 & b_3 & \ddots & \\ & & \ddots & \ddots & c_{n-1} \\ \beta & & & a_n & b_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_n \end{bmatrix}, \tag{3.9}$$

kde je $\alpha = a_n$, $\beta = c_1$ a n je počet riadkov systému.

Sherman-Morrison zjednodušuje výpočet takéhoto typu matíc vzhľadom k tomu, že pôvodnú maticu A upraví do tvaru $(A + u \otimes v)$. Teda lineárny systém (3.9) bude vyzerať nasledovne:

$$(A + u \otimes v).x = b \tag{3.10}$$

S použitím rovníc (3.11) odvodených z (3.10) získame vektory y a z,

$$A.y = b \qquad A.z = u \tag{3.11}$$

pomocou ktorých vieme vypočítať hľadané hodnoty :

$$x = y - \left[\frac{v.y}{1 + (v.z)}\right].z \tag{3.12}$$

3.3.1 Postup v programe

V programe sme si u a v zadefinovali nasledovne:

$$u = \begin{bmatrix} \gamma \\ 0 \\ 0 \\ \vdots \\ \beta \end{bmatrix} \qquad v = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ \frac{\alpha}{\gamma} \end{bmatrix}, \tag{3.13}$$

pričom γ je ľubovoľný parameter, ktorý sme si vzhľadom na zachovanie presnosti zadefinovali ako $\gamma = -b_1$ a vektory u a v majú veľkosť n.

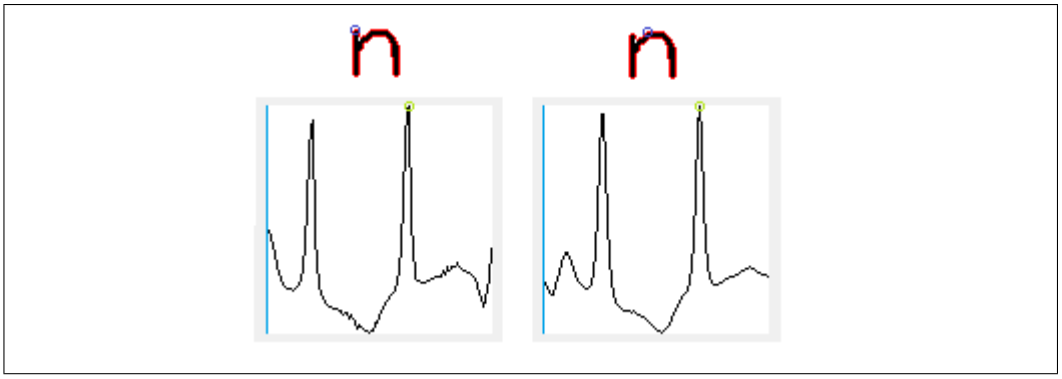
Teda matica A je trojdiagonálna časť systému (3.9) s dvomi modifikovanými členmi:

$$b_1' = b_1 - \gamma, \qquad b_n' = b_n - \frac{\alpha\beta}{\gamma} \tag{3.14}$$

Potom sme vyriešili rovnice (3.11) pomocou už spomínaného Thomasovho algoritmu. A nakoniec sme dosadením hodnôt vypočítaných pomocou Thomasovho algoritmu do rovnice (3.12) získali hľadaný vektor.

4. Invariantnosť voči počiatočnému bodu

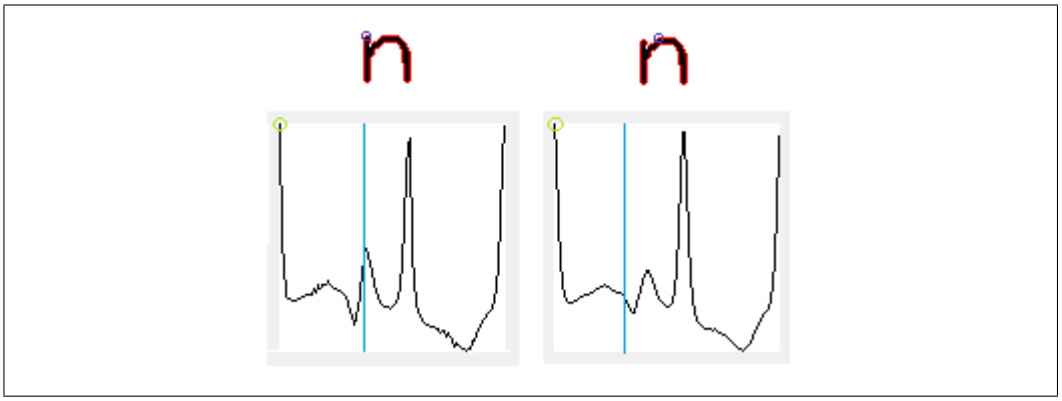
Po preskúmaní správania deskriptorov sme si všimli, že počiatočný bod, t.j. prvý najvrchnejší pixel patriaci hranici písmena, je citlivý voči rotácii a menším zmenám písmena, čo môže, pri niektorých písmenách, znamenať problém. Na obrázku 4.1 je bližšie znázornená situácia, pri ktorej nastáva spomínaný problém. Môžeme si všimnúť, že porovnávame podobné písmená, pričom majú rôzny počiatočný pixel. Ako vidieť, deskriptory sú síce podobné, ale posunuté, čo môže spôsobiť menej presnú informáciu. Na obrázku 4.1 môžeme pozorovať, že diskrétnu funkciu krivosti (deskriptor) sme transformovali na "spojitý"graf. Spravili sme to kvôli prehladnejšej vizualizácii a taktiež tento "spojitý"graf bude potrebný pri práci s metrikami.



Obr. 4.1: Porovnanie deskriptorov podobných písmen s rôznymi polohami počiatočného pixela. Prvý riadok: Hranice písmen, pričom fialový krúžok označuje počiatočný pixel hranice. Druhý riadok: "Spojité"grafy deskriptorov písmen. Globálne maximum, teda najvyššia hodnota krivosti je vyznačená zeleným krúžkom. Pomocou modrej čiary zvýrazňujeme pôvodnú polohu počiatočného pixela (označený fialovým krúžkom na hranici písmena) v grafe. Toto značenie sa bude objavovať aj v nasledujúcich obrázkoch.

Na základe daného poznatku sme sa rozhodli zvoliť si jeden pixel, vzhľadom na ktorý posunieme daný deskriptor. Ako najvhodnejší sme si určili pixel, ktorý reprezentuje globálne maximum, teda najvyššiu hodnotu krivosti. Obrázok 4.2 znázorňuje výsledok spomínaného posunu. Je vidieť, že na-

priek rôznym počiatočným pixelom sa "spojité"grafy deskriptorov dvoch podobných písmen na seba omnoho viac podobajú.



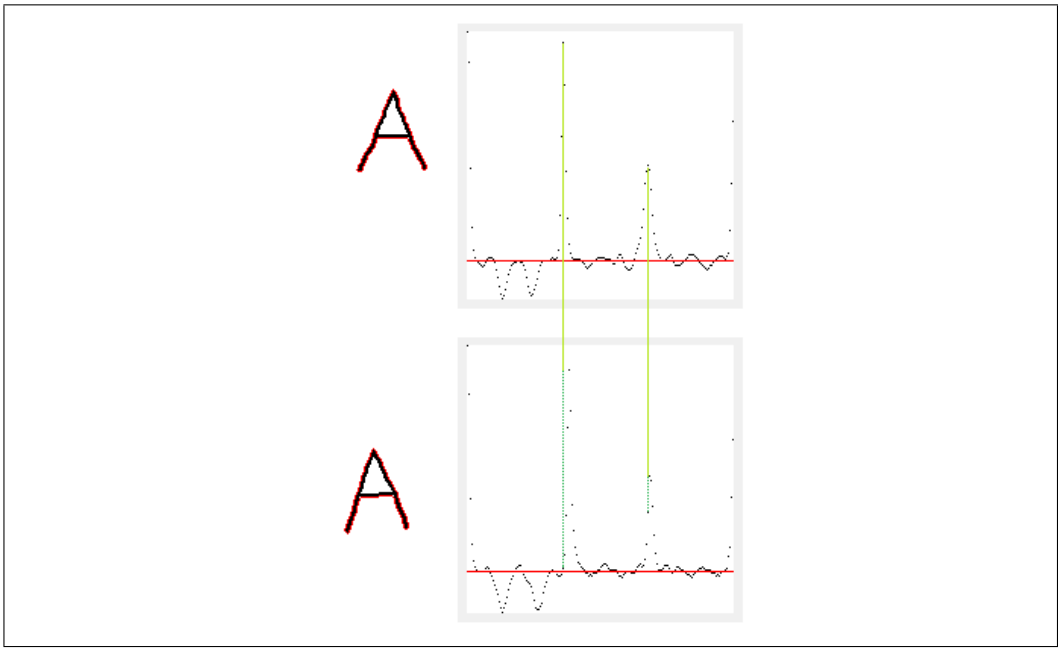
Obr. 4.2: Porovnanie posunutých deskriptorov podobných písmen

5. Metriky

Po získaní vhodnej krivky, respektíve deskriptora definujúceho vstupné písmeno, sme mohli uvažovať nad vhodnými metrikami, ktoré nám budú vedieť definovať podobnosť kriviek reprezentujúcich písmená na základe vzdialenosti. Cieľom je určiť, o aké strojom čitateľné písmeno sa jedná.

5.1 Rozdielová metrika

Za prvú metódu sme si zvolili jednoduché odčítavanie dvoch deskriptorov. Teda porovnávame konkrétny (skúmaný) deskriptor s deskriptorom uloženým v atlase a to tak, že prechádzame naraz po oboch deskriptoroch a odčítavame ich hodnoty, ktoré následne v absolútnej hodnote sčitujeme a ukladáme. Vzhľadom k tomu, že majú všetky deskriptory definovanú jednotnú dĺžku, sa naša práca značne zjednodušila. Za najvhodnejší deskriptor z atlasu vyberieme ten, ktorému sme napočítali najnižší rozdiel od skúmaného deskriptora. Ako vidieť na obrázku 5.1, môže nastať situácia, pri ktorej deskriptory vyzerajú takmer rovnako, ale jeden z nich je posunutý o pár pixelov a Rozdielová metrika tým pádom nemusí správne vyhodnotiť skúmaný deskriptor.



Obr. 5.1: Prvý stĺpec : Skúmané písmená. Druhý stĺpec : 2D grafy deskriptorov príslušných písmen. Pomocou zelených čiar je vyznačené, o koľko je prvý deskriptor posunutý vzhľadom na druhý deskriptor. Slabozelená čiara smeruje od pixela prvého deskriptora k pixelu druhého deskriptora, s ktorým by sa podľa správnosti mal pixel prvého deskriptora porovnávať. Tmavozelená prerušovaná čiara smeruje k pixelu, s ktorým sa bude pixel prvého deskriptora vskutočnosti porovnávať.

Na obrázku 5.1 si môžeme všimnúť, že pri odpočítaní daných pixelov (hodnôt) nám vznikne veľký rozdiel.

5.2 Bodovo-vzdialenostná metrika

Vzhľadom na skúsenosti, ktoré sme nadobudli skúmaním výsledkov z Rozdielovej metriky, sme vytvorili Bodovo-vzdialenostnú metriku, ktorá využíva vzdialenostnú funkciu. Pomocou nej ošetríme nedostatky rozdielovej metriky.

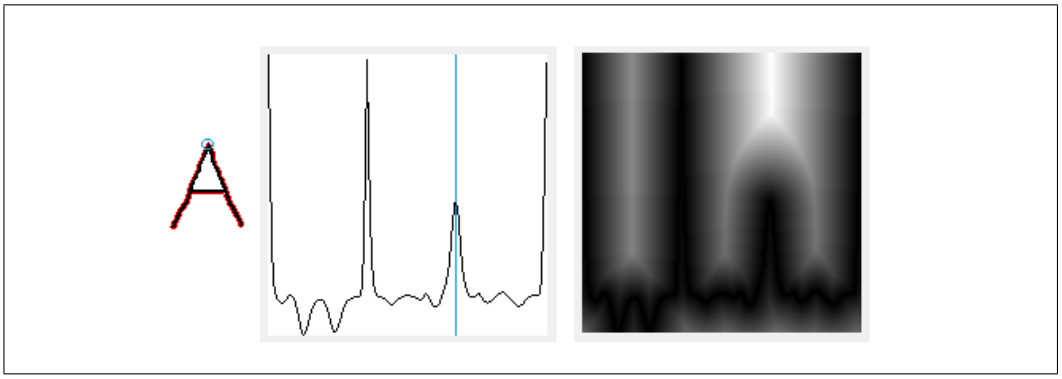
5.2.1 Vzdialenostná funkcia

Vzdialenostnú funkciu sme zostrojili pomocou Rouy-Tourin schémy, bližšie definovanú v [3]. Rovnica, od ktorej sa celý postup v [3] odvíja je Eikonalova rovnica s časovou relaxáciou:

$$d_t + |\nabla d| = 1 \tag{5.1}$$

počítaná na oblasti $\Omega \times [0, T_D]$, kde Ω je doména obrázka s Dirichletovou podmienkou:

$$d(x,t) = 0, \qquad x \in \Omega_0 \subset \Omega \tag{5.2}$$



Obr. 5.2: Pôvodné písmeno, "spojitý"graf deskriptora, preškálovaná vzdialenostná funkcia "spojitého"grafu deskriptora.

a Ω_0 je množina bodov, od ktorej sa vzdialenosť počíta.

Na numerický výpočet rovnice (5.1) s podmienkou (5.2) sme použili explicitnú časovú diskretizáciu s časovým krokom τ_D a pomocou Rouy-Tourin schémy sme diskretizovali (5.1) v priestore s krokom h_D. Rouy-Tourin schéma vyzerá pre (5.1) nasledovne:

$$d_{ij}^{n+1} = d_{ij}^{n} + \tau_D - \frac{\tau_D}{h_D}\sqrt{max(M_{ij}^{-1,0}, M_{ij}^{1,0}) + max(M_{ij}^{0,-1}, M_{ij}^{0,1})}, \tag{5.3}$$

pričom

$$M_{ij}^{pq} = (min(d_{i+p,j+q}^{n} - d_{ij}^{n}, 0))^2 \ . \tag{5.4}$$

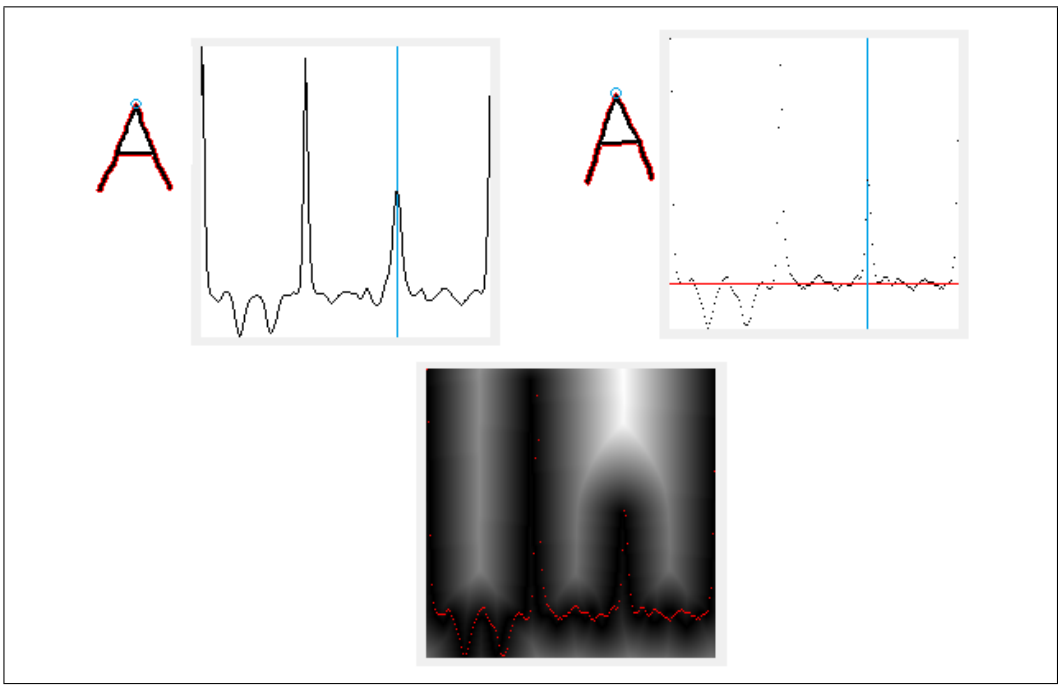
Na obrázku 5.2 je pre lepšiu viditeľnosť zobrazená preškálovaná vzdialenostná funkcia, pričom čierna má hodnotu 0 a biela 255. Teda je vidieť, ako sa postupne "zvyšuje hodnota"pixelov s narastajúcou vzdialenosťou od pôvodnej krivky deskriptora.

Vzhľadom na to, že deskriptor je graf krivosti jednoduchej uzavretej krivky, v neskorších etapách práce sme usúdili, že výsledky budú presnejšie, ak budeme počítať vzdialenostnú funkciu ku grafu funkcie tak, že na pravom a ľavom okraji budeme uvažovať cyklickú okrajovú podmienku.

5.2.2 Funkcionalita Bodovo-vzdialenostnej metriky

Ako prvý krok sme potrebovali diskrétny tvar deskriptora písmena v atlase transformovať na "spojitý"2D graf vzľadom na požiadavky vzdialenostnej funkcie. Následne sme vypočítali vzdialenostnú funkciu, ktorú sme si uložili. Tento proces sa vykonáva automaticky pre daný deskriptor po jeho pridaní do atlasu.

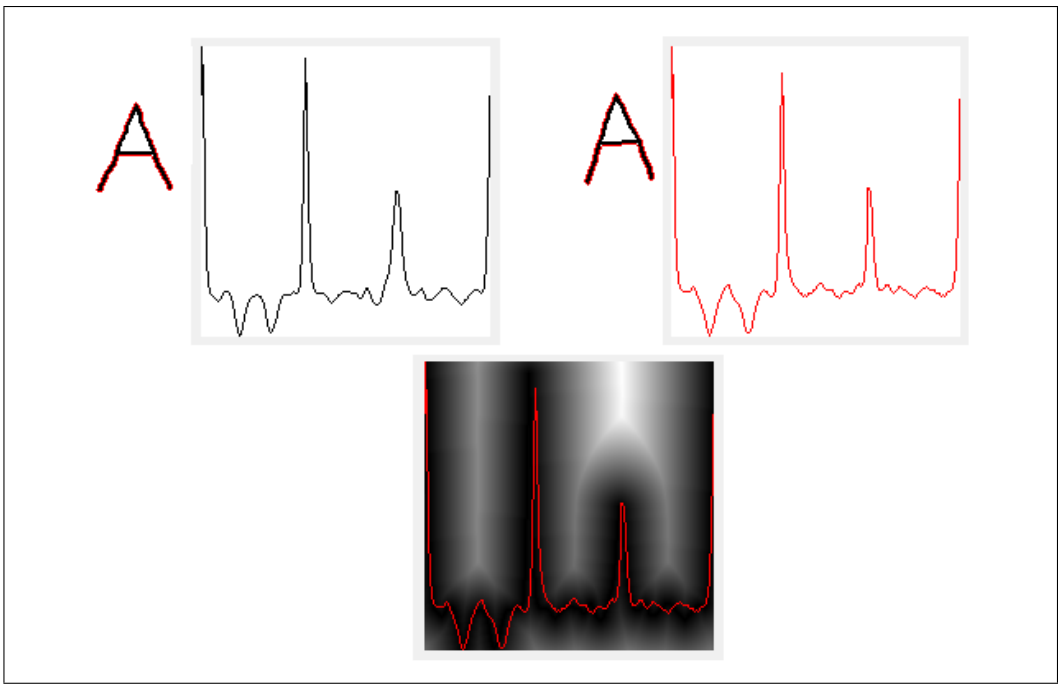
Ďalej sme mohli porovnávať skúmaný deskriptor s deskriptormi v atlase a to tak, že sme sa pohybovali po skúmanom deskriptore a zároveň sme sledovali, akú hodnotu má aktuálny bod vo vzdialenostnej funkcii deskriptora v atlase, následne sme hodnoty sčítali. Bodovo-vzdialenostná metrika vyberá ako najvhodnejší deskriptor z atlasu pre skúmaný deskriptor ten, pri ktorom je suma sčítaných hodnôt najnižšia.



Obr. 5.3: Prvý riadok: Pôvodné písmená s deskriptormi. Druhý riadok: Preškálovaná vzdialenostná funkcia deskriptora prvého písmena s deskriptorom druhého písmena (červená farba).

5.3 Grafovo-vzdialenostná metrika

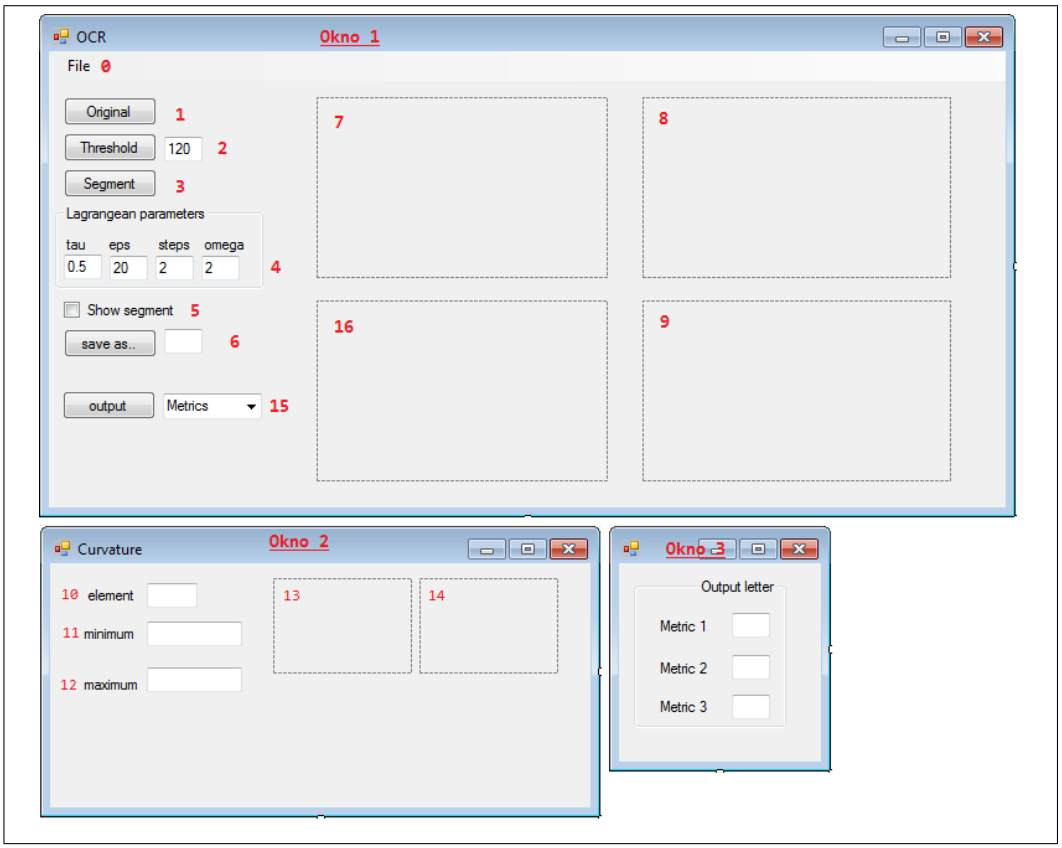
Pracuje rovnako ako Bodovo-vzdialenostná metrika s rozdielom, že graf vzdialenostnej funkcie deskriptora v atlase porovnávame so "spojitým"2D grafom skúmaného deskriptora. Teda sledujeme aké hodnoty majú pixely "spojitého"2D grafu deskriptora skúmaného písmena v grafe krivosti deskriptora v atlase. Za najvhodnejší deskriptor vyberáme z atlasu ten, pri ktorom nám metrika vracala najmenšiu hodnotu (vzdialenosť). Týmto spôsobom vieme presnejšie určiť podobnosť ako pomocou Bodovo-vzdialenostnej metriky.



Obr. 5.4: Prvý riadok: Pôvodné písmená so "spojitýmii"2D grafmi deskriptorov. Druhý riadok: Preškálovaná vzdialenostná funkcia deskriptora prvého písmena so "spojitým"2D grafom deskriptora druhého písmena (červená farba).

6. Popis softvéru

Všetky metódy sme implementovali v softvéri, ktorý sme nazvali všeobecnou skratkou OCR. Softvér bol vytvorený v Microsoft Visual Studio 2013 v jazyku C# 5.0 .
Jeho základná myšlienka je, že užívateľ nastaví parametre na získanie najvhodnejšieho deskriptora. Akonáhle program vytvorí a uloží, okrem iného, deskriptory pre každé písmeno zo vstupného obrázka, užívateľ vyznačí písmeno (kliknutím naňho), ktoré chce uložiť, napíše do texboxu jeho názov (počítačom rozoznateľný znak - písmeno/číslo z klávesnice) a potvrdí ho, čím sa dané písmeno uloží do atlasu. Do atlasu sa ukladá deskriptor, vzdialenostná mapa a užívateľom zvolený názov. Akonáhle klikneme na ďalšie písmeno, zobrazí sa nám výstup z jednotlivých metrík.



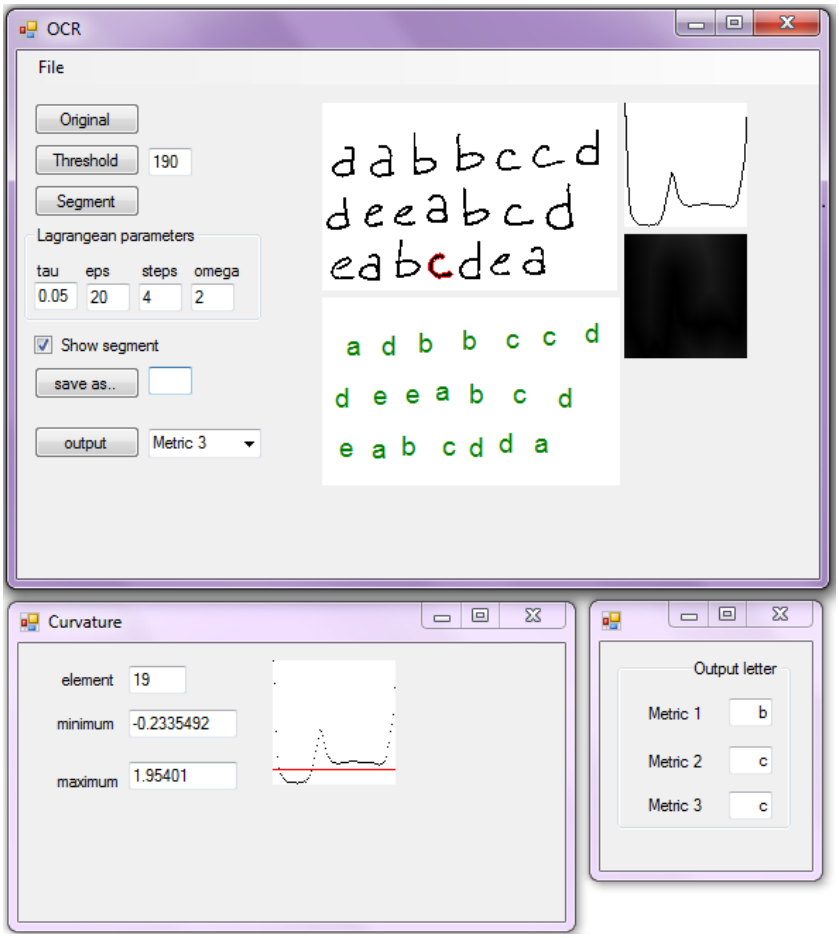
Obr. 6.1: Grafické rozhranie

6.1 Opis grafického rozhrania

1. Pomocou File → Open (0) sa nám objaví dialógové okno, pomocou ktorého si môžme zvoliť ľubovoľný vstup (obrázok) z počítača alebo pripojeného externého disku.
2. Tlačítko Threshold (2) nám umožňuje aplikovať thresholding na základe hodnoty, ktorú zadáva užívateľ pomocou príslušného textBoxu.
3. Pomocou tlačítka Segment (3) sa vykonajú všetky potrebné operácie na získanie a uloženie deskriptorov pre každé písmeno z aktuálneho vstupného obrázka. Pred použitím tlačítka Segment treba nastaviť parametre vstupujúce do krivostnej difúzie (4).
4. Do pictureBoxu (7) sa načíta užívateľom zvolený vstup (obrázok). pB (7) reaguje na klik s myšou.
5. CheckBox zaktívnime, ak chceme, aby sa v pictureBoxe (7) zobrazila aj hranica daného písmena (aktívne písmeno), na ktorý sme klikli v (7).
6. V pictureBoxe (8) sa po kliknutí na písmeno v (7) objaví 2D graf deskriptora aktívneho písmena a zároveň sa nám objaví Okno 2.
7. TextBox (10) vypíše polohu aktívneho písmena v poli, textBox (11) vypíše minimálnu hodnotu krivosti (deskriptora) aktívneho písmena a textBox (12) vypíše maximálnu hodnotu krivosti (deskriptora) aktívneho písmena.
8. V pictureBoxe (13) sa nám objaví graf diskrétnej krivky deskriptora aktívneho písmena. pB (13) reaguje na dvoj-klik myšou.
9. Po dvoj-kliku myšou na pB (13) sa nám graf z pB (13) objaví v pB (14) a bude tam dovtedy, kým opäť neklikneme do pB(13). Tento pictureBox slúžil hlavne pri sledovaní zmeny deskriptorov písmen zo vstupu.
10. Ak chceme deskriptor aktívneho písmena uložiť do atlasu, napíšeme do textBoxu (6) jeho názov, teda počítačom rozoznateľný znak definujúci písmeno a stlačíme tlačítko save as..
11. V pB (9) sa nám zobrazí graf vzdialenostnej funkcie práve uloženého písmena. Akonáhle máme v atlase aspoň jeden uložený deskriptor spolu s jeho názvom, po kliknutí do pictureBoxu (7) sa nám otvorí okno 3, ktoré nám prezentuje meno deskriptora (počítačom čitateľný znak) ktorí určili jednotlivé metriky.
12. Ak chceme vidieť konečnú transformáciu textu do počítačom čitateľných znakov, pomocou scrollBaru (15) si zvolíme metriku, ktorej predikciu chceme vidieť a stlačením príslešného tlačítka output sa v pictureBoxe (16) objaví daný text.

Tlačítkom Original (1) vrátime upravovaný vstupný obrázok do pôvodného stavu.

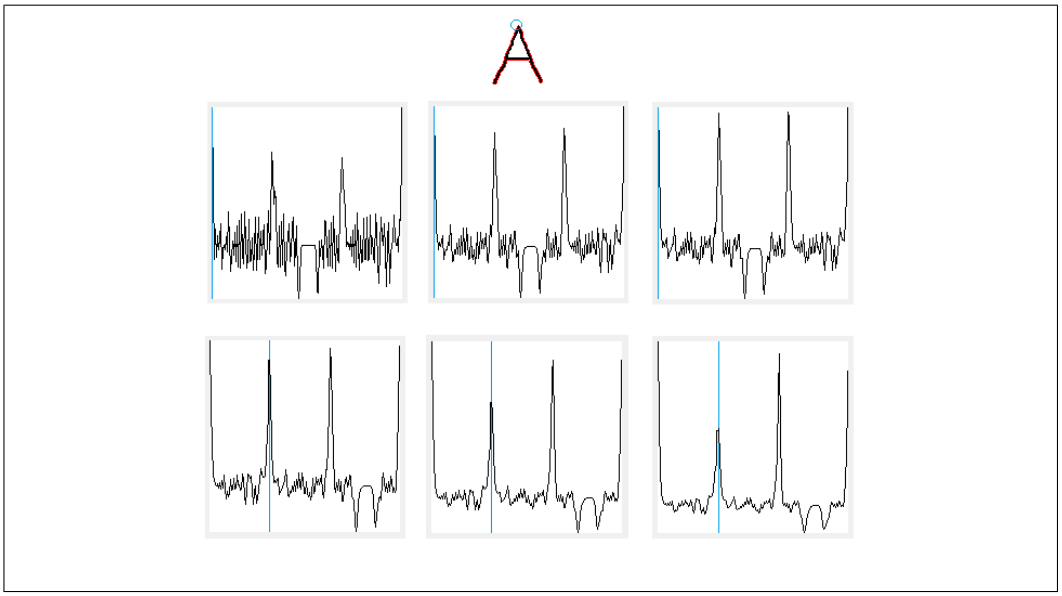
Nasledovný obrázok ilustruje použitie popísaného softvéru v praxi.


OCR
File
Original
Threshold
190
Segment
Lagrangean parameters
tau
eps
steps
omega
0.05
20
4
2
Show segment
save as..
output
Metric 3
Curvature
element
19
minimum
-0.2335492
maximum
1.95401
Output letter
Metric 1
b
Metric 2
c
Metric 3
c

7. Experimenty

7.1 Porovnanie vplyvu krivostných parametrov na deskriptor

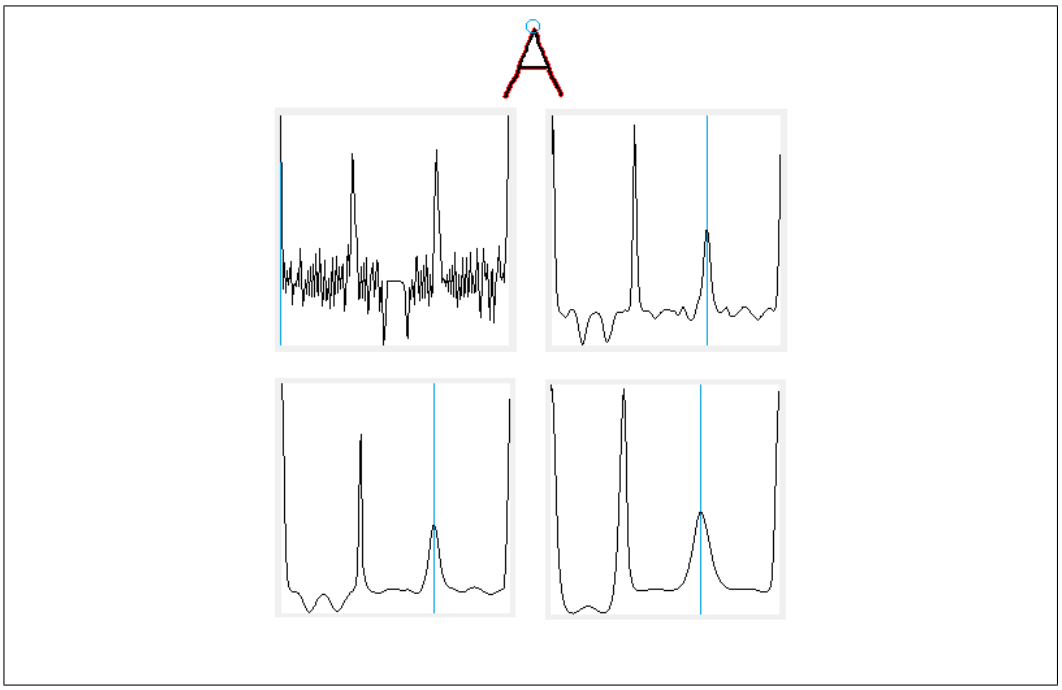
7.1.1 Porovnanie deskriptora pri zmene veľkosti časového kroku τ



Obr. 7.1: Prvý riadok: Písmeno A s threshold = 190. Druhý a tretí riadok: Deskriptory písmena A pri zmene τ : (τ = 0.01, ϵ = 20, počet časových krokov = 2, ω = 2), (τ = 0.03, ϵ = 20, počet časových krokov = 2, ω = 2), (τ = 0.04, ϵ = 20), (τ = 0.05, ϵ = 20, počet časových krokov = 2, ω = 2), (τ = 0.07, ϵ = 20, počet časových krokov = 2,ω = 2), (τ = 0.09, ϵ = 20, počet časových krokov = 2, ω = 2).

Na obrázku 7.1 je vidieť, ako sa deskriptor s narastajúcim τ vyhladzuje.

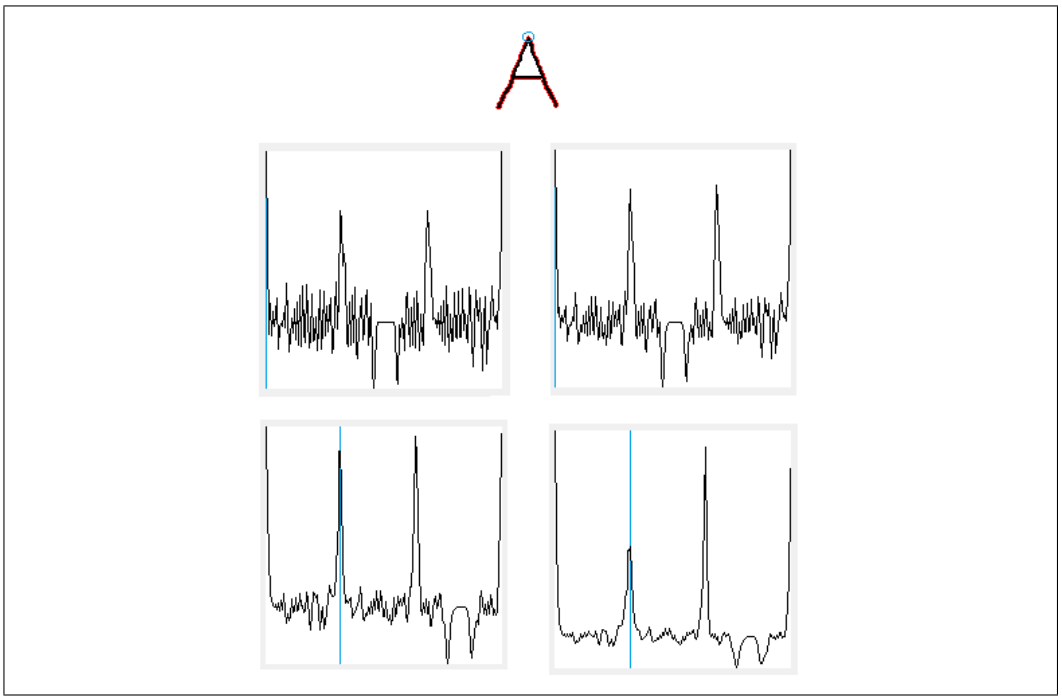
7.1.2 Porovnanie deskriptora pri zmene počtu časových krokov



Obr. 7.2: Prvý riadok: Písmeno A s threshold=190. Druhý a tretí riadok: Deskriptory písmena A pri zmene počtu časových krokov: ($\tau = 0.05$, $\epsilon = 20$, počet časových krokov $= 1$, $\omega = 2$), ($\tau = 0.05$, $\epsilon = 20$, počet časových krokov $= 5$, $\omega = 2$), ($\tau = 0.05$, $\epsilon = 20$, počet časových krokov $= 10$, $\omega = 2$), ($\tau = 0.05$, $\epsilon = 20$, počet časových krokov $= 20$, $\omega = 2$).

Na obrázku 7.2 si môžeme všimnúť, že s narastajúcim počtom časových krokov sa deskriptor vyhladzuje.

7.1.3 Porovnanie deskriptora pri zmene veľkosti parametra ϵ



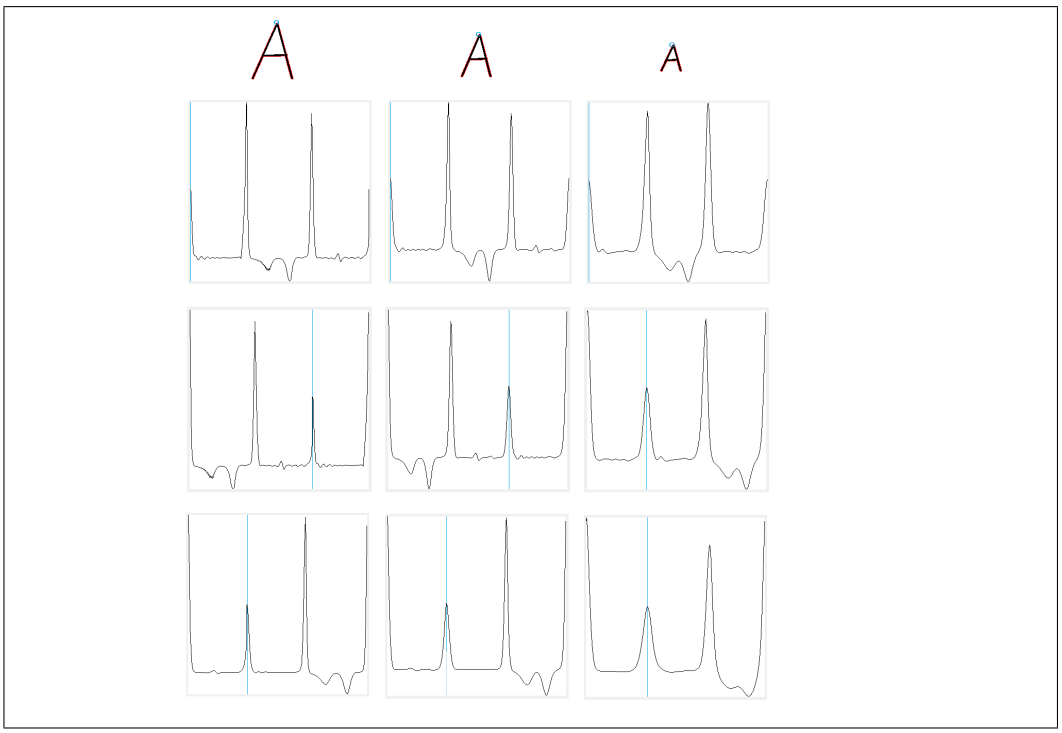
Obr. 7.3: Prvý riadok: Písmeno A s threshold=190. Druhý a tretí riadok: Deskriptor písmena A pri zmene ϵ: ($\tau = 0.05$, $\epsilon = 5$, počet časových krokov $= 2$, $\omega = 2$), ($\tau = 0.05$, $\epsilon = 10$, počet časových krokov=2, ω=2), ($\tau = 0.05$, $\epsilon = 20$, počet časových krokov $= 2$, $\omega = 2$), ($\tau = 0.05$, $\epsilon = 40$, počet časových krokov $= 2$, $\omega = 2$).

Na obrázku 7.3 si môžeme všimnúť, že s narastajúcou hodnotou ϵ sa deskriptor vyhladzuje.

7.2 Porovnanie deskriptorov modifikovaného písmena

V tomto experimente skúmame vplyv zmeny rotácie a zmeny veľkosti písmen na deskriptory. Na obrázkoch 7.4 a 7.5 môžeme vidieť, že pri spomínaných vplyvoch sa deskriptory relatívne dosť podobajú.

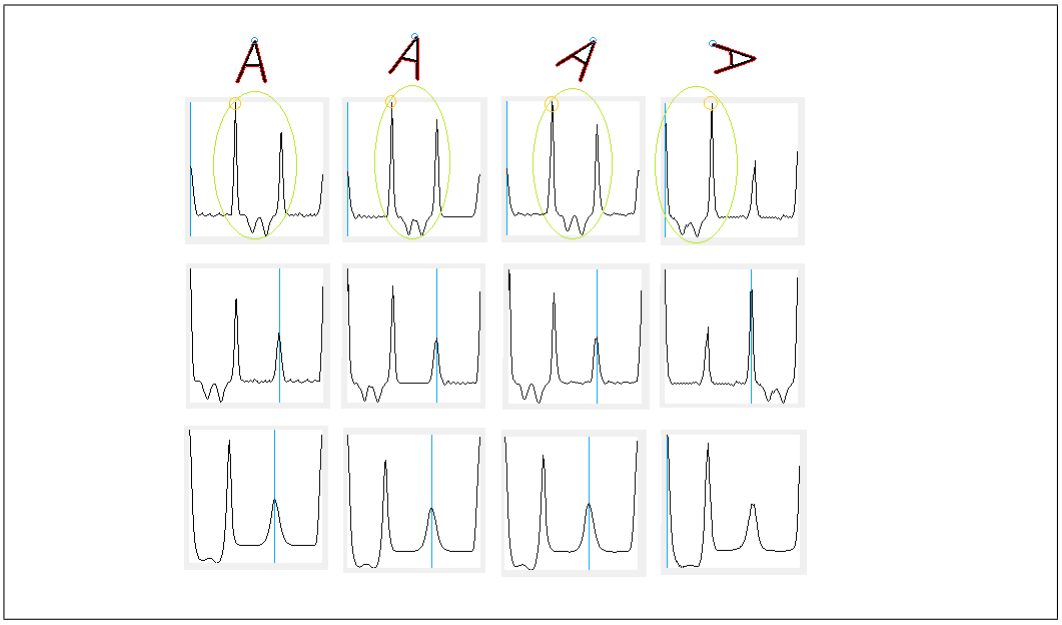
7.2.1 Porovnanie deskriptora pri zmene veľkosti písmena



Obr. 7.4: Deskriptor písmena A pri zmene jeho veľkosti. Prvý riadok: Písmená s Threshold=220, Modrým krúžkom je vyznačený počiatočný bod. Druhý riadok: Deskriptory písmen s neošetrenou invarianciou počiatočného bodu. Vstupné parametre: $\tau = 0.05$, $\epsilon = 20$, počet krokov $= 6$, $\omega = 2$. Tretí riadok: Deskriptory písmen s ošetrenou invarianciou počiatočného bodu. Vstupné parametre: $\tau = 0.05$, $\epsilon = 20$, počet krokov $= 6$, $\omega = 2$. Štvrtý riadok: Deskriptory písmen s ošetrenou invarianciou počiatočného bodu. Vstupné parametre: $\tau = 0.05$, $\epsilon = 20$, počet krokov $= 12$, $\omega = 2$.

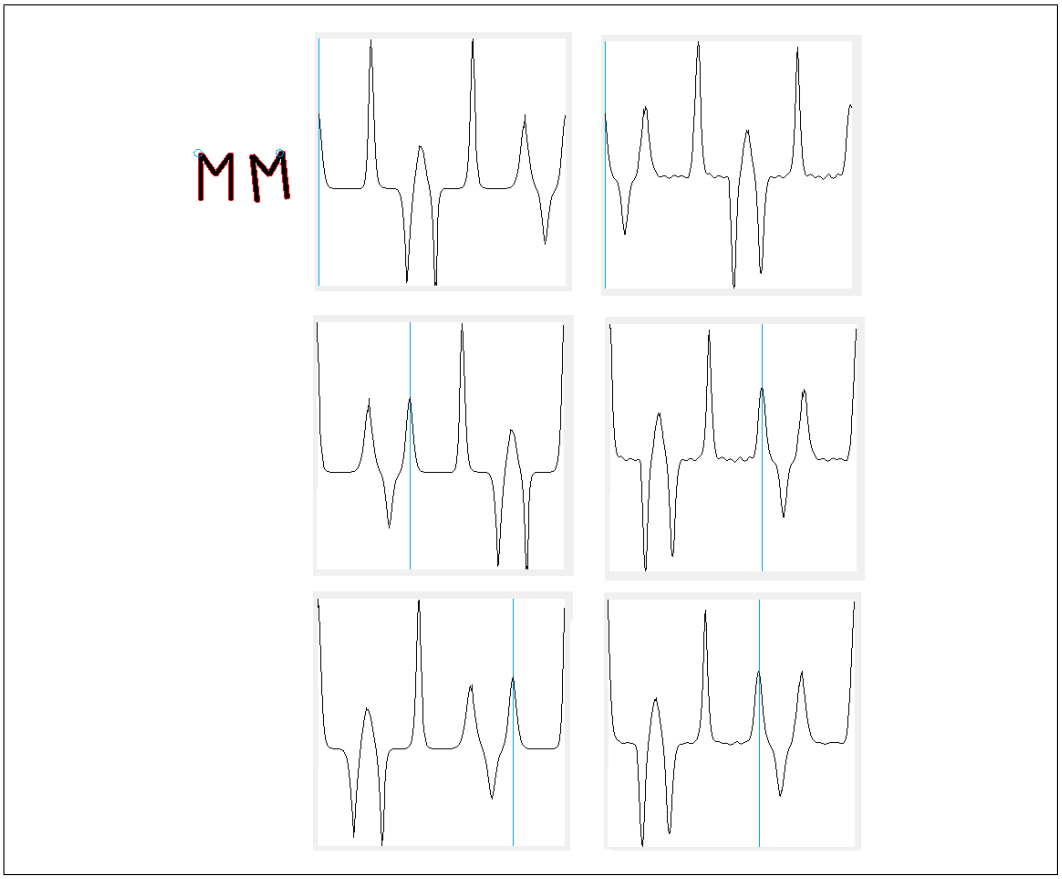
Na obrázku 7.4 si môžeme všimnúť, že "spojitý"2D graf deskriptora najmenšieho písmena A (posledný), je na rozdiel od predošlých dvoch značne posunutý. Je to spôsobené tým, že pri zhladzovaní deskriptorov vzniklo globálne maximum v inom bode. Nastáva to pri písmenách, ktorých deskriptory obsahujú dve a viac viditeľných maxím. Takúto situáciu vyriešime tak, že zadefinujeme v atlase písmeno A pod dvomi deskriptormi alebo vhodne navrhneme vstupné parametre krivostnej difúzie.

7.2.2 Porovnanie deskriptora pri zmene rotácie písmena



Obr. 7.5: Prvý riadok: Rotácia písmena A s threshold = 240. Druhý riadok: Deskriptory príslušných písmen s neošetrenou invarianciou počiatočného bodu ($\epsilon = 0.05$, $\tau = 20$, počet časových krokov = 5, $\omega = 2$) . Tretí riadok : Deskriptory príslušných písmen invariantných voči počiatočnému bodu ($\epsilon = 0.05$, $\tau = 20$, počet časových krokov = 5, $\omega = 2$). Štvrtý riadok: Deskriptory príslušných písmen invariantných voči počiatočnému bodu ($\epsilon = 0.05$, tau = 20, počet časových krokov = 7, $\omega = 2$).

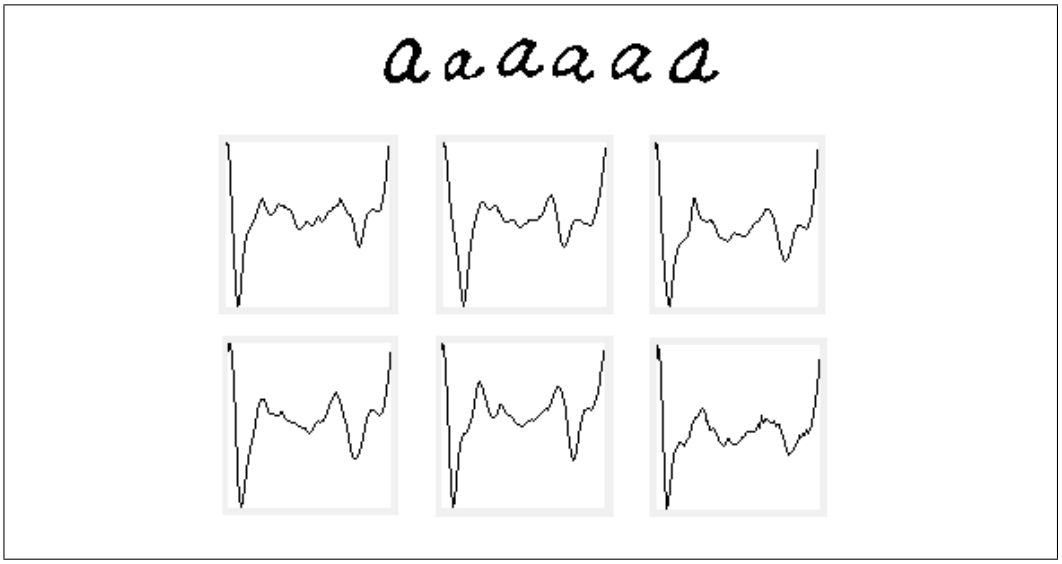
Na obrázku 7.5 je zobrazená rotácia písmena *A*. Pri rotácii sa mierne modifikujú písmená, preto si môžeme všimnúť drobný šum na deskriptoroch. Keď si všimneme časti vyznačené zelenou elipsou, môžeme vidieť, ako sa dekriptor pod vplyvom rotácie posúva. Prvé 3 deskriptory v druhom riadku sú značne podobné, lebo majú jednoznačne rovnaký počiatočný bod. Posledný deskriptor je viditeľne posunutý vzhľadom na odlišný počiatočný bod. Tento problém mal byť vyriešený pomocou posunu grafu deskriptora voči globálnemu maximu (invariantnosť voči počiatočnému bodu). Avšak, keď sa zameriame na globálne maximá deskriptorov (vyznačených v druhom riadku oranžovým krúžkom) vidíme, že deskriptor posledného písmena má iné globálne maximum ako predošlé. Je to spôsobené šumom, ktorý vznikol pri rotácii, a následným zhladením. Preto môžeme v predposlednom riadku vidieť prípad, ktorý sme spomínali v predošlej podsekcii. Písmeno s najväčšou rotáciou (posledné) má vzľadom na ostatné značne posunutý deskriptor. V štvrtom riadku vidíme, že pri zvýšení počtu časových krokov sa deskriptor posledného písmena zmodifikoval natoľko, že globálne maximum vzniklo na mieste rovnakom ako majú ostatné deskriptory.
Podobný prípad môžeme pozorovať na nasledovnom obrázku 7.6.



Obr. 7.6: Prvý riadok : Písmená s threshold = 240, deskriptory príslušných písmen neinvariantných voči počiatočnému bodu ($\epsilon = 0.05$, tau = 20, počet časových krokov = 5, $\omega = 2$) . Druhý riadok : Deskriptory príslušných písmen invariantných voči počiatočnému bodu ($\epsilon = 0.05$, $\tau = 20$, počet časových krokov = 5, $\omega = 2$). Tretí riadok : Deskriptory príslušných písmen invariantných voči počiatočnému bodu ($\epsilon = 0.05$, $\tau = 20$, počet časových krokov = 6, $\omega = 2$).

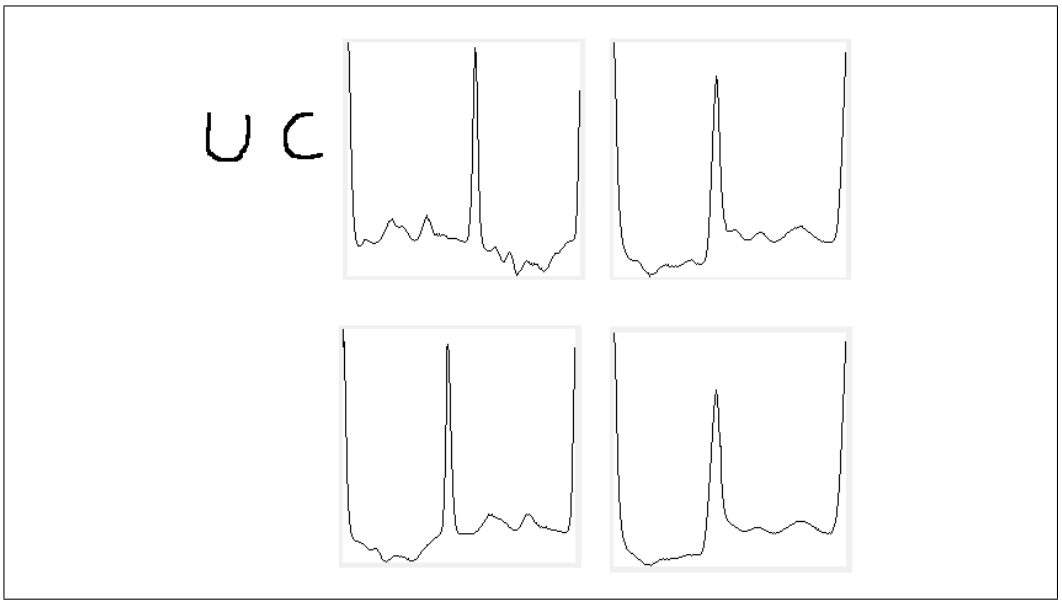
7.2.3 Porovnanie deskriptorov rôznych podôb písmena *a*

V porovnaní s predošlými experimentmi sme v tomto skúmali deskriptory písmen, ktoré neboli len preškálované alebo pootočené kópie jedného písmena. Na obrázku 7.7 môžeme vidieť reálnu situáciu, pri ktorej skúmame deskriptory písmen *a*.



Obr. 7.7: Prvý riadok: Pôvodné písmená s threshold = 240. Druhý riadok: Deskriptory rôznych podôb písmena a pri krivostných parametroch $\tau = 0.05$, $\epsilon = 20$, počet krokov krivostnej difúzie = 4, $\omega = 2$.

7.3 Nevýhody krivostného deskriptora



Obr. 7.8: Prvý riadok : Deskriptory príslušných písmen ($\epsilon = 0.05$, tau = 20, počet časových krokov = 5). Druhý riadok : Deskriptory príslušných písmen ($\epsilon = 0.05$, tau = 20, počet časových krokov = 7).

Na obrázku 7.8 vidíme, že dve rôzne písmená môžu mať podobné deskriptory. Ďalším príkladom sú písmená t a x alebo M a W.

8. Záver

Podarilo sa nám pomocou matematických metód a postupov zostrojiť aplikáciu, ktorá je schopná na základe vytvoreného atlasu rozpoznať ručne písaný text pomocou nami vytvorených metrík. V práci sme stručne popísali matematické metódy implementované v aplikácii na získanie vhodného tvaru deskriptorov a priblížili sme fungovanie metrík. Na záver sme v experimentoch popísali správanie sa deskriptorov v rôznych situáciach, sledovali sme vplyvy vstupných parametrov krivostnej difúzie a taktiež sme poukázali na možné problémy metód, ako aj na ich možné riešenia.

Literatúra

[1] Shunji Mori, Ching Y. Suen, Kazuhiko Yamamoto: *Historical Review of OCR Research and Development*, published in Proceedings of the IEEE, ISSN 0018-9219, IEEE, 1992

[2] Rafael C. Gonzalez, Richard E. Woods: *Digital Image Processing, Third Edition*, Addison-Wesley Pub (Sd), March 1992

[3] Paul Bourgine, Peter Frolkovič, Karol Mikula, Nadine Peyriéras, Mariana Remešíková: *Extraction of the intercellular skeleton from 2D images of embryogenesis using eikonal equation and advective subjective surface method*, in Lecture Notes in Computer Science 5567 (Proceeding of the 2nd International Conference on Scale Space and Variational Methods in Computer Vision, Voss, Norway, June 1-5,2009), Springer (2009) pp. 38-49

[4] Martin Balažovjech, Karol Mikula, Mária Petrášová, Jozef Urbán: *Lagrangean method with topological changes for numerical modelling of forest fire propagation* , ALGORITMY 2012, 19th Conference on Scientific Computing, Podbanske, Slovakia, September 9-14, 2012, Proceedings of contributed papers and posters (Eds. A.Handlovicova, Z.Minarechova, D.Sevcovic), ISBN 978-80-227-3742-5, Publishing House of STU, 2012, pp. 42-52

[5] William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery: *Numerical Recipes, 3rd edition*, Cambridge University Press, 2007