Paralelní implementace fuzzy filtrů ve zpracování obrazu na GPU

Ladislav Horký

FJFI ČVUT v Praze

Vedoucí práce: Ing. Tomáš Oberhuber Ph.D.
Konzultant: doc. Ing. Jaromír Kukal Ph.D.

2. února 2012

Obsah

Cíle práce

- Seznámit se se základy fuzzy filtrů pro zpracování obrazu
- Seznámit se s programovacím prostředí CUDA
- Implementace vybraných algoritmů filtrace na CPU
- Provést paralelizaci algoritmů a implementaci na GPU
- Porovnat urychlení vůči CPU při použití různých datových typů
- Aplikovat algoritmy na reálná 3D obrazová data

Proč fuzzy?

Výhody

- Solidní teoretický základ pro analýzu filtrů, postavený na Łukasiewiczově algebře s odmocninou
- Matematický model je přímo aplikovatelný na data, tak jak jsou reprezentována v počítači
- Specializovaná na robustní odšumovací filtry
- Redukce na jednoduché matematické operace
- Možnost návrhu a analýzy sítí filtrů

Nevýhody

- Několik oblíbených filtrů zde nelze implementovat (Gauss)

Proč fuzzy?

Výhody

- Solidní teoretický základ pro analýzu filtrů, postavený na Łukasiewiczově algebře s odmocninou
- Matematický model je přímo aplikovatelný na data, tak jak jsou reprezentována v počítači
- Specializovaná na robustní odšumovací filtry
- Redukce na jednoduché matematické operace
- Možnost návrhu a analýzy sítí filtrů

Nevýhody

- Několik oblíbených filtrů zde nelze implementovat (Gauss)

Łukasiewiczova algebra s odmocninou

Definice

Nechť $\mathbf{L} = [0, 1]$ a $a, b \in \mathbf{L}$. Nechť $\langle \mathbf{L}, \wedge, \vee, \otimes, \rightarrow, 0, 1 \rangle$, kde

$$a \wedge b = \min(a, b) \qquad a \otimes b = \max(a + b - 1, 0)$$
$$a \vee b = \max(a, b) \qquad a \rightarrow b = \min(1 - a + b, 1)$$

je **reziduovaný svaz** s **Łukasiewiczovou t-normou** $\otimes$, s největším prvkem 1 a nejmenším prvkem 0, pak $\langle \mathbf{L}, \wedge, \vee, \otimes, \rightarrow, sqrt, 0, 1 \rangle$, kde

$$\mathrm{sqrt}(a) = (1 + a)/2$$

je **Łukasiewiczova algebra s odmocninou**. Značíme $ŁA_{sqrt}$.

Łukasiewiczova algebra s odmocninou

Definice

Nechť $\mathbf{L} = [0, 1]$ a $a, b \in \mathbf{L}$. Nechť $\langle \mathbf{L}, \wedge, \vee, \otimes, \rightarrow, 0, 1 \rangle$, kde

$$a \wedge b = \min(a, b) \qquad a \otimes b = \max(a + b - 1, 0)$$
$$a \vee b = \max(a, b) \qquad a \rightarrow b = \min(1 - a + b, 1)$$

je **reziduovaný svaz** s **Łukasiewiczovou t-normou** $\otimes$, s největším prvkem 1 a nejmenším prvkem 0, pak $\langle \mathbf{L}, \wedge, \vee, \otimes, \rightarrow, sqrt, 0, 1 \rangle$, kde

$$\mathrm{sqrt}(a) = (1 + a)/2$$

je **Łukasiewiczova algebra s odmocninou**. Značíme $\mathcal{L}A_{sqrt}$.

Łukasiewiczova algebra s odmocninou

Definice

Nechť $\mathbf{L} = [0, 1]$ a $a, b \in \mathbf{L}$. Nechť $\langle \mathbf{L}, \wedge, \vee, \otimes, \rightarrow, 0, 1 \rangle$, kde

$$a \wedge b = \min(a, b) \qquad a \otimes b = \max(a + b - 1, 0)$$
$$a \vee b = \max(a, b) \qquad a \rightarrow b = \min(1 - a + b, 1)$$

je **reziduovaný svaz** s **Łukasiewiczovou t-normou** $\otimes$, s největším prvkem 1 a nejmenším prvkem 0, pak $\langle \mathbf{L}, \wedge, \vee, \otimes, \rightarrow, sqrt, 0, 1 \rangle$, kde

$$\mathrm{sqrt}(a) = (1 + a)/2$$

je **Łukasiewiczova algebra s odmocninou**. Značíme $Ł A_{sqrt}$.

Łukasiewiczova algebra s odmocninou

Definice

Nechť $\mathbf{L} = [0, 1]$ a $a, b \in \mathbf{L}$. Nechť $\langle \mathbf{L}, \wedge, \vee, \otimes, \rightarrow, 0, 1 \rangle$, kde

$$a \wedge b = \min(a, b) \qquad a \otimes b = \max(a + b - 1, 0)$$
$$a \vee b = \max(a, b) \qquad a \rightarrow b = \min(1 - a + b, 1)$$

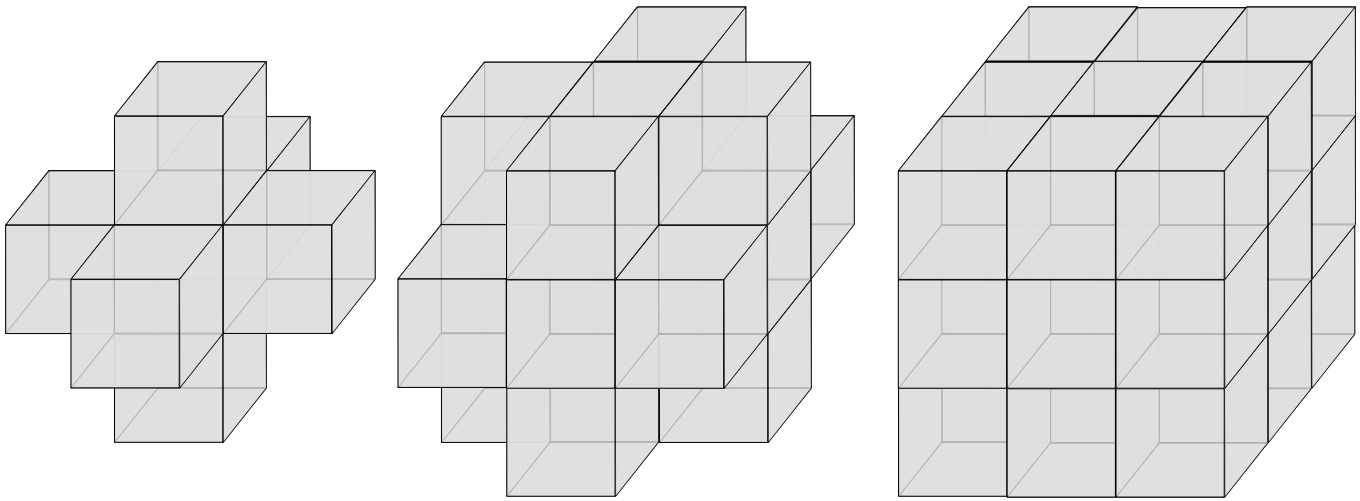
je **reziduovaný svaz** s **Łukasiewiczovou t-normou** $\otimes$, s největším prvkem 1 a nejmenším prvkem 0, pak $\langle \mathbf{L}, \wedge, \vee, \otimes, \rightarrow, sqrt, 0, 1 \rangle$, kde

$$\mathrm{sqrt}(a) = (1 + a)/2$$

je **Łukasiewiczova algebra s odmocninou**. Značíme $ŁA_{sqrt}$.

Práce s 3D obrazem

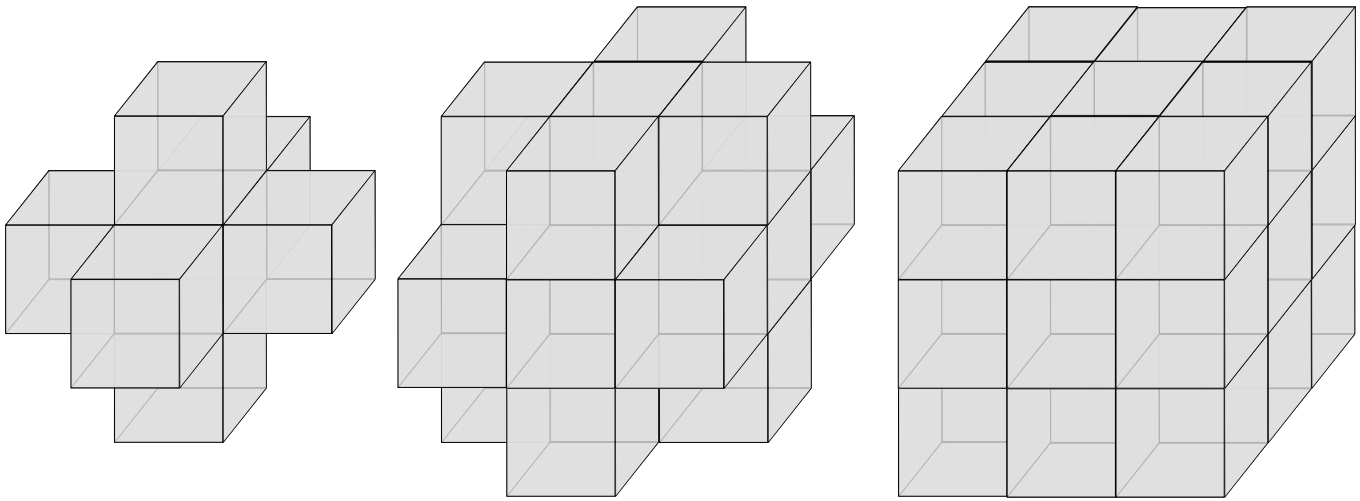
- Pouze jednokanálové obrazy (odstíny šedi)
- Hodnota výtupního voxelu je funkcí pouze voxelů ze sousedství (definovaného tzv. strukturním elementem) zdrojového voxelu
- Tvar strukturního elementu má velký vliv na výsledek
- Z použití sousedství voxelu plynou problémy s okraji obrázku



Obrázek: Tradiční strukturní elementy

Práce s 3D obrazem

- Pouze jednokanálové obrazy (odstíny šedi)
- Hodnota výtupního voxelu je funkcí pouze voxelů ze sousedství (definovaného tzv. strukturním elementem) zdrojového voxelu
- Tvar strukturního elementu má velký vliv na výsledek
- Z použití sousedství voxelu plynou problémy s okraji obrázku



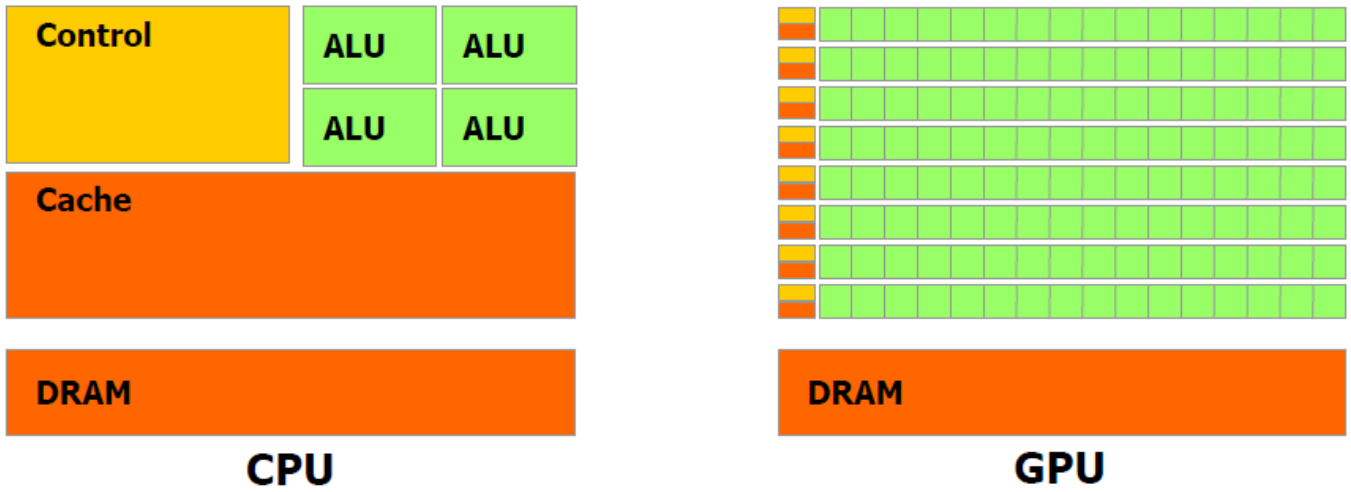
Obrázek: Tradiční strukturní elementy

Filtry

Roztřídění filtrů z implementačního hlediska:

- Pomocné, algebraické (sčítání, prahování)
- Morfologické (eroze, dilatace, hrany)
- Odšumovací využívající seřazené pole (median, BES)
  - Založené na průměrování využívající Walshův seznam (H.L. median, WBES)

Rozdíly CPU a GPU



Obrázek: Rozdíly v architektuře

Hlavní rozdíly v programování

CPU

- Velké odstupňované cache
- Chytrý hardware

- V případě více jader plnohodnotný MIMD
- Cacheování instrukcí, branch-prediction

GPU

- Hrubé odstupňování pamětí
- Manuální, ale větší kontrola nad HW
- Přísná pravidla pro přístup do paměti

- Pouze chytřejší, po částech vektorový procesor
- Výpočet je paralelní, pouze pokud 32 současně běžících vláken vykonává stejný kód

Implementační postřehy pro GPU

Implementace na starší kartě 8800 GTX:

- Překrývání latencí paměti - masivní paralelismus
- Šetření rychlou sdílenou pamětí
- Lepší je hloupější, ale nevětvící se program
- Využití manuální texturové cache
- U odšumovacích filtrů se problém redukoval na nalezení vhodného třídícího algoritmu

Selekční algoritmy

Pro malá pole bylo GPU použito **Zapomnětlivé třídění**:

- Spotřeba jen polovičního množství rychlé paměti, než je velikost tříděného pole (3/4 u BES)
- Větvení programu je minimální pro jakýkoliv vstup
- Lepší rozvrstvení přístupů do pomalé globální paměti pro vstupní data
- Náročnost $\mathcal{O}(n^2)$

Pro velká pole byl použit „hloupý“ $\mathcal{O}(n^2)$ algoritmus založený na zjištění počtu prvků větších a menších, než zkoumaný prvek.

Na CPU byl použit optimalizovaný **Introselect**.

Testovací sestava

	CPU	GPU
Název	Intel Core 2 DUO	NVIDIA GeForce 8800 GTX
Výpočetní jednotky	2 (využita 1)	128 (4 SM $\times$ 32 CUDA-jader)
Frekvence	1860 MHz	1400 MHz
Výpočetní schopnost	—	1.0
Cache	2 MB L2	—
RAM/DRAM	2 GB DDR2	768 MB GDDR3
FSB	800 MHz	1800 MHz

Testovací data: 3D SPECT obraz 79x95x69 voxelů

Urychlení morfologických filtrů

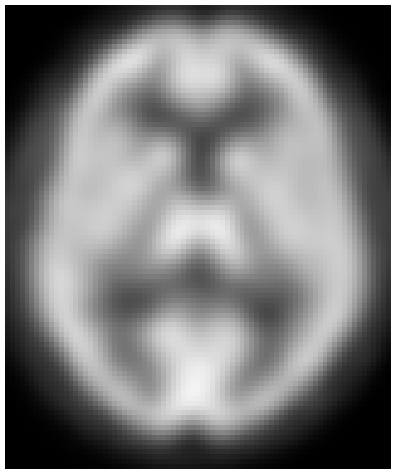
Maska → / Filtr ↓	kapacita masky: 26 GPU (ms)	kapacita masky: 26 Urychlení	kapacita masky: 18 GPU (ms)	kapacita masky: 18 Urychlení
`unsigned char` (0-255)				
Eroze	1,47	**30,2**	1,34	**24,0**
Dilatace	1,47	**30,3**	1,37	**23,8**
Hrany	1,56	**51,8**	1,35	**43,2**
`unsigned int` (0-65535)				
Eroze	1,61	**31,0**	1,37	**26,9**
Dilatace	1,62	**30,9**	1,37	**27,1**
Hrany	1,53	**51,4**	1,32	**43,5**

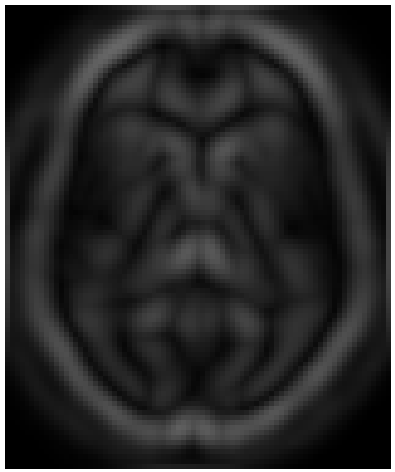
Urychlení morfologických filtrů

Maska → / Filtr ↓	kapacita masky: 26 GPU (ms)	kapacita masky: 26 Urychlení	kapacita masky: 18 GPU (ms)	kapacita masky: 18 Urychlení
`unsigned char` (0-255)				
Medián	9,45	**33,9**	3,62	**58,3**
BES	10,17	**37,1**	4,98	**53,1**
H-L Medián	735,76	**4,3**	342,34	**4,9**
WBES	1124,46	**4,2**	463,81	**5,4**
`unsigned int` (0-65535)				
Medián	25,24	**12,4**	5,96	**34,6**
BES	10,95	**35,6**	6,45	**42,5**
H-L Medián	1327,99	**2,3**	434,21	**3,8**
WBES	1328,10	**3,6**	520,38	**4,9**

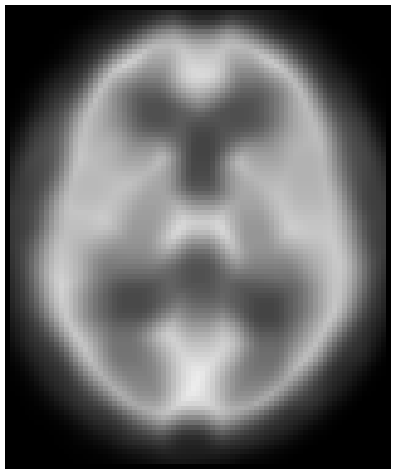
Ukázky morfologických filtrů

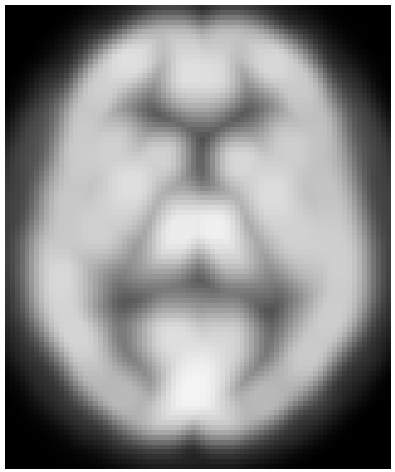
Originál Hrany





Eroze Dilatace

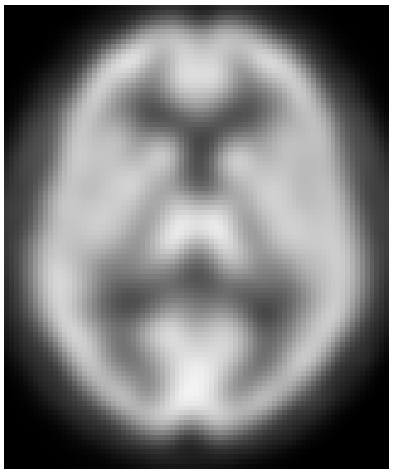


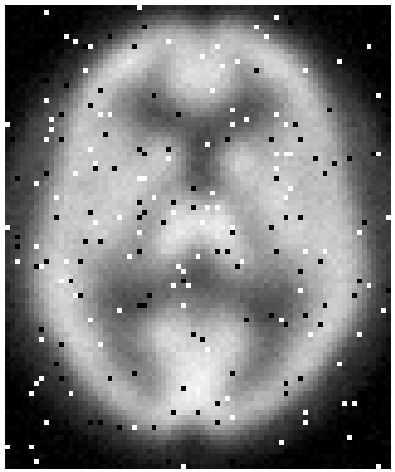


Ukázka BES

Originál

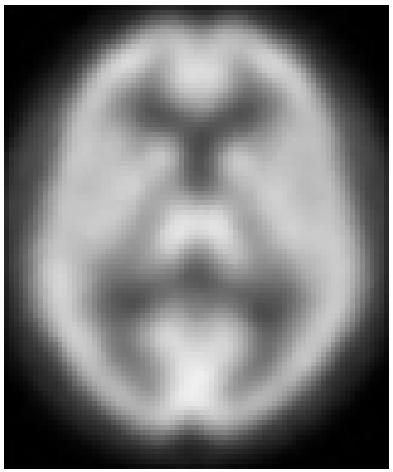
kontaminace 5%
šum $\sigma = 5/256$

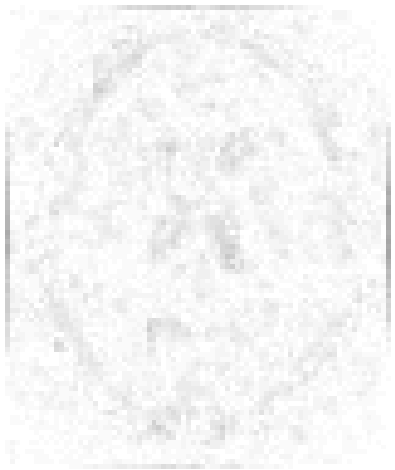




Výsledek BES

Rozdíl od originálu





Závěr

- Fuzzy logika se pro popis filtrů výborně osvědčila
- Lze odhadovat, kterým směrem se bude ubírat vývoj GPU
- Třídění (selekce) malých polí jako zajímavý problém pro GPU
- Některé filtry se podařilo zrychlit na jednotky ms, což je předpoklad pro použití v reálném čase

Děkuji za pozornost.

Dotaz oponenta I

Jakého zrychlení by bylo možné dosáhnout paralelizací na vícejádrovém CPU oproti sériovému výpočtu na CPU?

Odpověď:
Maximální teoretické hranice – n-násobek při použití n jader.

Dotaz oponenta II

Zaznamenal jste při srovnání statistických filtru na CPU a GPU pro různé datové typy kvalitativní změny ve výsledných datech?

Odpověď:
Tyto kvalitativní rozdíly jsem nezkoumal. Z aritmetiky chyb popsané v kapitole 2.1 plyne, že tento rozdíl může být nejvýš o 1 (na škále 256), což je okem nerozlišitelné.