DIGITAL FILTER DESIGN ON GPU

Filip Vaverka
Bachelor Degree Programme (3), FIT BUT
E-mail: xvaver07@stud.fit.vutbr.cz

Supervised by: Lukáš Polok
E-mail: ipolok@fit.vutbr.cz

Abstract: This article shows one of the approaches to the design of digital filters using an evolutionary genetic algorithm (GA). Its main contribution to this subject is that the implementation is parallel and it is accelerated by GPU. Genetic operators which are used are modified and GA is using real-value encoded genes. This approach allows automated design of digital filters with given parameters. Resulting digital filters are in form suitable for cascade implementation.

Keywords: Digital filter design, Signal processing, GPU, OpenCL, Genetic algorithm

1 ÚVOD

V tomto článku je ukázán jeden z přístupů k řešení problému návrhu digitálních filtrů s nekonečnou impulzní odezvou (IIR filtry). Popsané řešení je založeno na evolučním genetickém algoritmu a umožňuje navrhnout filtr na základě specifikovaného řádu filtru a jeho charakteristiky (frekvenční nebo také fázové). Díky tomu, že zde prezentované řešení je, na rozdíl od existujících řešení, implementováno na grafickém čipu (GPU), došlo k významnému zrychlení běhu algoritmu. V první části jsou stručně shrnuta existující řešení návrhu digitálních filtrů. Následuje popis řešení zvoleného v této práci. V poslední části jsou pak zhodnoceny vlastnosti výsledné implementace.

2 EXISTUJÍCÍ ŘEŠENÍ

Mezi existující řešení je možné zahrnout klasická řešení výpočtu koeficientů filtru jako: výpočet koeficientů pomocí umístění nul a pólů (Pole-zero placement method) [3], impulse invariant method, matched z-transform (MZT) a bilinear z-transform (BZT) [4]. Tyto metody jsou (s vyjímkou první) založeny na transformaci analogového filtru na digitální, což vyžaduje znalost výchozího filtru.

Jelikož při návrhu digitálního filtru jde především o hledání jeho koeficientů (proměnných), je možné se na problém dívat jako na optimalizační problém, kde koeficienty tvoří parametry. Odchylku frekvenční (případně i fázové) charakteristiky od požadované lze použít jako cenovou funkci. Tento přístup používají například řešení [1] nebo [2].

3 NÁVRH FILTRU POMOCÍ GENETICKÉHO ALGORITMU

Zde prezentované řešení jednak rozšiřuje výsledky práce [2] a také ukazuje jak je možné použitý algoritmus vhodně akcelerovat. Podobně jako ona využívá evolučního genetického algoritmu, kde jsou geny, tvořící koeficienty filtru, uloženy jako reálné hodnoty. Všechny filtry v populaci jsou reprezentovány jako n-tice filtrů druhého řádu (1), kde K je normalizační konstanta, a a b jsou koeficinty filtrů. Výhodou tohoto uspořádání je, že oblast stability koeficientů filtru druhého řádu tvoří trojúhelník.

$$H(z) = K\prod_{i=1}^{N}\left(\frac{1+b_{i1}z^{-1}+b_{i2}z^{-2}}{1+a_{i1}z^{-1}+a_{i2}z^{-2}}\right) \tag{1}$$

3.1 GENETICKÉ OPERÁTORY

Genetické operátory jsou tvořeny funkcemi, jejichž kód je prováděn na straně GPU (OpenCL kernels). V závislosti na povaze operátoru je možné některé provádět paralelně na úrovni jednotlivých filtrů (například operátor křížení), jiné na úrovni jednotlivých stupňů (operátory selekce a mutace). Koeficienty filtrů v populaci jsou, stejně jako jejich charakteristiky, stabilně udržovány v lokální paměti grafické karty. Jediný přenos dat mezi GPU a hostem tak tvoří hodnoty kvality jednotlivých filtrů. Aplikace libovolného operátoru na populaci nesmí vytvořit žádné nestabilní filtry.

Operátor selekce nahrazuje v populaci novými jedinci ty filtry, jejichž kvalita je nižší než daný práh. Práh je určen jako kvalita n-tého nejhoršího filtru v populaci. Tato hodnota je hledána na straně hosta (CPU) pomocí algoritmu quick-select.

Operátor křížení je aplikován paralelně na všechny filtry v populaci z nichž jsou některé náhodně zvoleny ke křížení. Ke každému je následně vybrán, taktéž náhodně, druhý filtr. Tím vzniká dvojice filtrů ($a^{(x)}$ a $a^{(y)}$) a lineární interpolací (2) jejich jednotlivých genů (koeficientů filtru) je vytvořen nový jedinec, kterým je nahrazen méně kvalitní filtr ze vstupní dvojice. Koeficient lineární interpolace α je funkcí (3) rozdílu kvality obou filtrů, což zvyšuje rychlost konvergence (oproti náhodně volenému α). Jelikož oblast koeficientů stabilních filtrů je konvexní, je tímto zajištěna stabilita výsledného filtru.

$$a_i = \alpha a_i^{(x)} + (1-\alpha) a_i^{(y)} \tag{2}$$

$$\alpha(\Delta f) = \frac{1}{2}\Delta f + \frac{1}{2} \tag{3}$$

Kde Δf je rozdíl kvality filtrů, $a_i^{(x)}$, $a_i^{(y)}$ a a_i jsou koeficienty vstupních a výsledného filtru.

Posledním genetickým operátorem je mutace. Tento operátor umožňuje vstup nové náhodné informace do populace. V popisovaném řešení je tvořen jednak operací selekce, kde se generují nové náhodné filtry a pak také mírnou modifikací náhodně zvolených filtrů v populaci. Modifikace existujících filtrů probíhá tak, že jsou z populace (tvořené n-ticemi filtrů 2. řádu) náhodně vybírány filtry 2. řádu a jejich koeficienty jsou náhodně upravovány. Stabilita výsledného filtru je zajištěna omezením maximálních náhodných odchylek jednotlivých koeficientů tak, aby filtr zůstal stabilní. Jelikož má mutace negativní vliv na konvergenci algoritmu, je její pravděpodobnost postupně snižována (4).

$$p_m(k) = \frac{0.05 e^{(k_0-k)/k}}{1+e^{(k_0-k)/k}} \tag{4}$$

Kde k je číslo generace a k_0 určuje strmost poklesu pravděpodobnosti mutace.

3.2 HODNOCENÍ KVALITY FILTRU

Charakteristiky hledaného filtru jsou zadány jako minimum a maximum v každém bodě frekvenční a případně i fázové charakteristiky filtru. Kvalita filtru je pak dána rovnicí (5).

$$f = a\frac{1}{1+\sum err_{freq}^2(n)} + (1-a)\frac{1}{1+\sum err_{phase}^2(n)} \tag{5}$$

Kde parametr a určuje váhu jednotlivých složek kvality filtru, n je číslo vzorku charakteristiky filtru, $err_{freq}(n)$ a $err_{phase}(n)$ jsou odchylky od zadaných mezí charakteristik ve vzorku n.

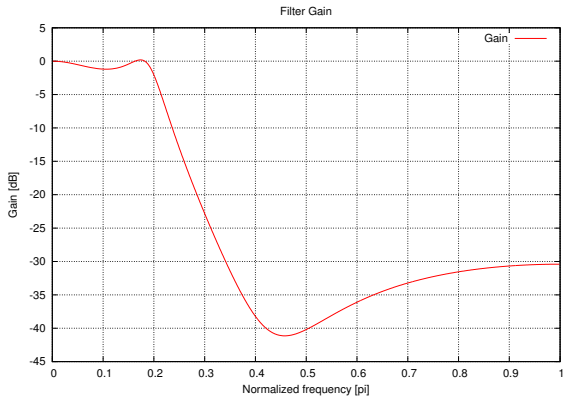
4 VLASTNOSTI VÝSLEDNÉ IMPLEMENTACE

V této kapitole jsou zhodnoceny vlastnosti výsledné implementace a to především z výkonnostního (časového) hlediska. V druhé části jsou uvedeny grafy zobrazující průběh a výsledek návrhu IIR filtru 4. řádu. V tabulce 1 jsou uvedeny doby běhu algoritmu v závislosti na řádu navrhovaného filtru. Pro porovnání byla zvolena jednovláknová implementace shodného algoritmu v C++ přeložena pomocí GCC 4.7 (za použití přepínačů -O3 a -ffast-math). N označuje velikost populace, K počet generací.

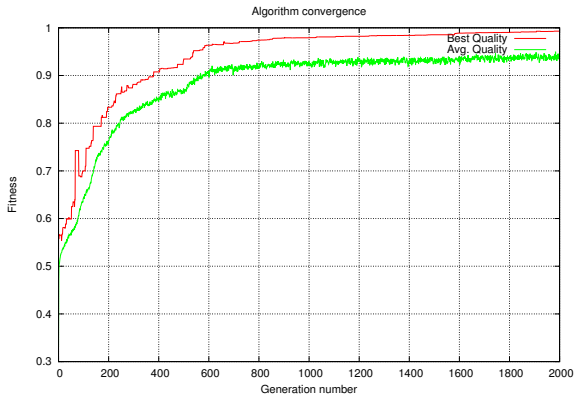
Platforma, Řád filtru:	2	4	6	8
ATI RV770 (GPU - OpenCL)	8.5s	8.8s	8.7s	9.7s
AMD Athlon X2 @ 2.5 GHz (CPU)	165s	193s	224s	256s

Tabulka 1: Doba běhu algoritmu při návrhu filtrů různých řádů, $N = 1024$, $K = 1000$

Následující grafy zobrazují průběh návrhu filtru typu dolní propust (filtr typu IIR 4. řádu) s velikostí generace $N = 1024$ a počtem generací $K = 2000$. Propustné pásmo filtru je na intervalu $< 0, 0.2\pi >$ s maximálním poklesem -1.5dB a zádržné pásmo $< 0.3\pi, \pi >$ s maximálním ziskem -30dB.



(a) Zisk filtru



(b) Konvergence algoritmu

5 ZÁVĚR

V tomto článku byl stručně popsán návrh digitálního filtru typu IIR pomocí paralelního evolučního genetického algoritmu na GPU. Při zvolení tohoto přístupu došlo k průměrně více než dvacetinásobnému zrychlení v porovnání s tradiční implementací. Jedním z možných rozšíření této práce by mohlo být využití hybridního genetického algoritmu, což by mělo vést k dalšímu zlepšení konvergence.

REFERENCE

[1] AHMAD, Sabbir U. Design of Digital Filters Using Genetic Algorithms [online]. University of Victoria, 2008. Dostupné z: http://hdl.handle.net/1828/1294. University of Victoria.

[2] J. D. Jang and S. G. Kong. Design of Optimal Digital IIR Filters using the Genetic Algorithm, Journal of Fuzzy Logic and Intelligent Systems, Vol. 2, No. 2, s.115-121, červen 2002.

[3] JAN, Jiří. Číslicová filtrace, analýza a restaurace signálů. 2. uprav. vydání. Brno: VUTIUM, 2002, s. 124-126. ISBN 80-214-1558-4.

[4] IFEACHOR, Emmanuel C. a Barrie W. JERVIS. Digital Signal Processing: A Practical Approach. Edinburg Gate: Pearson Education Limited, 2002, s. 463-500. Second Edition. ISBN 0-201-59619-9.