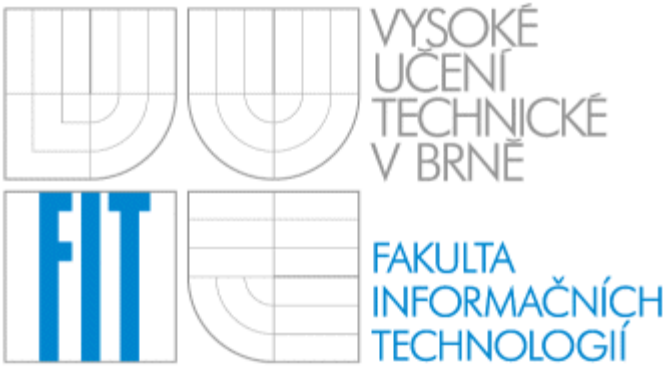


Filtrace videa

Projekt do předmětu Multimédia – MUL

Autoři:

Pavel Csóka, xcsoka00@stud.fit.vutbr.cz

Matúš Fedorko, xfedor01@stud.fit.vutbr.cz

12.5.2014

1 Zadání

Implementuje vybrané obrazové filtry (nejméně 5) libovolnou formou. Příkladem mohou být filtry z následujícího seznamu:

- zostření
- rozmazání
- inverze barev nebo převod na stupně šedi
- deformace
- reliéf
- roztažení kontrastu
- detekce hran
- odstranění šumu
- odstranění prokládání (u prokládaného videa)
- rotace nebo změna velikosti videa
- pixelizace nebo jiné efekty

Funkčnost filtrů demonstrujte na vhodných videosekvencích.

2 Řešení

K řešení jsme se rozhodli využít Qt framework ve verzi 5, který poskytl prostředky pro vytvoření grafického uživatelského rozhraní (GUI) a podpodru vícevláknového zpracování. Pro načítání a ukládání videa jsme zvolili knihovnu OpenCV verze 2. Algoritmy pro filtraci videa jsme implementovali vlasnoručně v jazyce OpenCL, díky čemuž jsou zpacovávany za pomocí grafického adaptéru (GPU). Jako programovací jazyk jsme zvolili C++11 a testování probíhalo na operačních systémech Windows a OS X.

Aplikace je implementována s využitím návrhových vzorů Model-View-Controller (MVC) a Singleton. MVC zajišťuje oddělení vlastní aplikační logiky od GUI aplikace a Singleton, se stará, že za behu aplikace bude k dispozici jen a pouze jedna instance třídy Controller.

2.1 Zpracování videa

Video soubory jsou načítány pomocí OpenCV knihovny do struktury cv::Mat. Pomocí ní se získávají jednotlivé snímky, které jsou následně převedeny do Qimage formátu nad kterým se provádí jejich zpracování, zobrazení a ukládání.

Tyto jednotlivé snímky jsou získávány v cyklu, kde se aplikují filtry. Jestliže není vybrán žádný, tak se pouze zobrazují. Z video souboru je přečteno snímkování videa (FPS) a na jeho základě je určeno zpoždění, se kterým se jednotlivé snímky v cyklu zobrazují. Je zohledněna doba zpracování snímku (aplikace filtrů), aby výsledné snímkování bylo stejné jako původní. Pokud aplikování filtru trvá delší dobu, než za jakou by měl být následující snímek zobrazen, není uvedena žádná prodleva.

Přehrávání videa (filtrování) je prováděno v odděleném vlákně, aby GUI zůstalo responsivní i při vysokém vytížení systému a nachází se ve třídě *Player*. Stejně tak je provedena konverze do souboru (třída *Converter*).

Zobrazení videa v GUI je uskutečněno pomocí třídy *GLRenderer,* která obdržený snímek upraví na velikost zobrazovací plochy a vykreslí pomocí OpenGL. Při změně velikosti je zachován poměr stran videa.

Samotná aplikace filtrů na snímek je provedena nejprve vybráním a nastavením filtrů ve třídě *FilterPipeline,* které se provádí z GUI a následě předáním každého načteného vstuního snímku této třídě. Výstup je zobrazen v GUI nebo uložen do souboru.

2.2 Implementace filtrů

Obrazové filtry jsou napsány v jazyce OpenCL. Základem je třída *FilterPipeline,* do které se přidávají samotné třídy filtrů implementující rozhraní abstraktni třídy *Filter.* Pipeline pak provádí na vstupním snímku videa jeho zpracování pomocí nastavených filtrů v pořadí, v jakém byly do této třídy přidány. Vstupní snímky jsou předávány do pipeline z tříd *Player* (při zpracování a zobrazení výsledku v reálném čase) a *Converter* (při konverzi do souboru).

Algoritmy filtrace jsou uloženy v souborech .cl a jedná se o tzv. kernely, které zapouzdřují třídy *Filter*. Jedná se o OpenCL programy, které jsou kompilovány až za běhu aplikace.

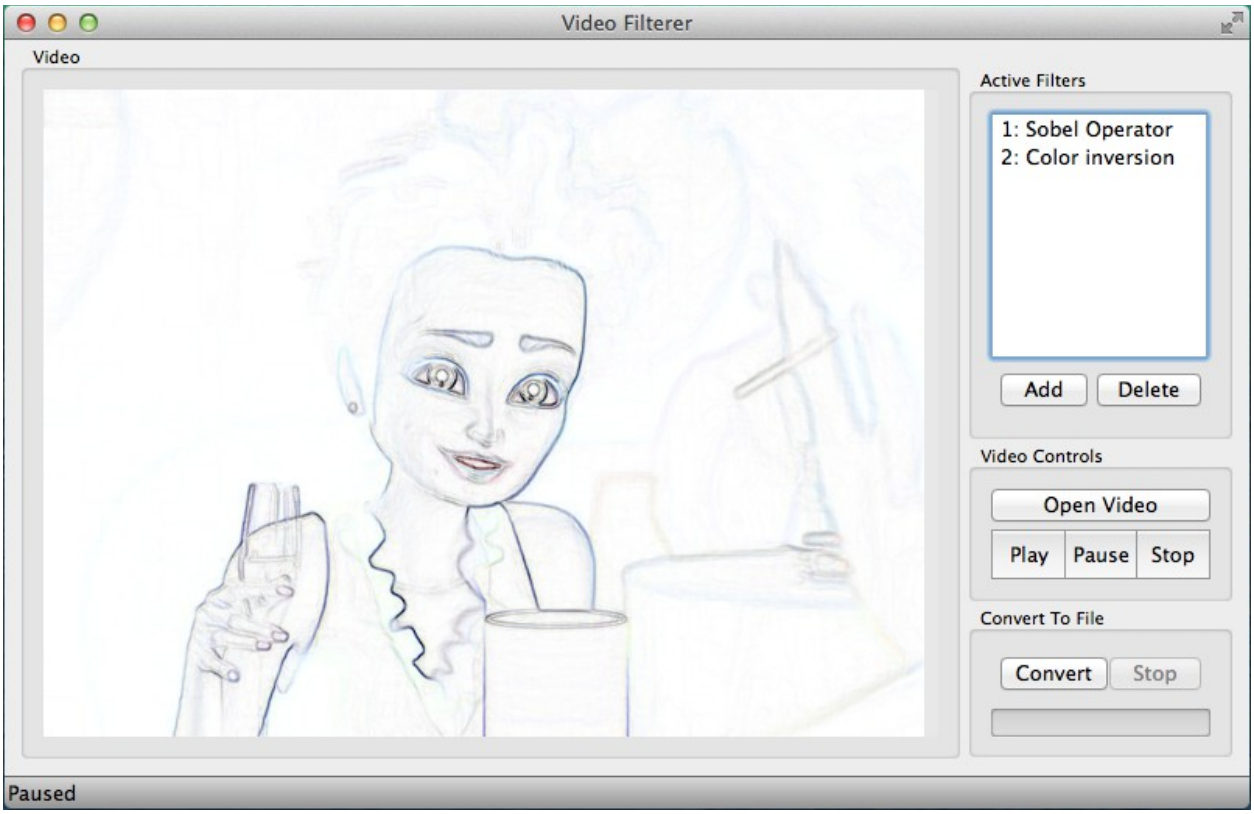
Typy filtrů

Implementovány jsou následující obrazové filtry:

- Grayscale filter – převod do odstínů šedi
- Sepia filter – sépiový filtr
- Color Inversion – inverze barev
- Transformation – změna affiní transformace podle zadané transformační matice
- Convolution 2D – 2D konvoluce obrazu se zadanou maskou (kernelem)
- Separable Convolution 2D – separabilní konvoluce se zadanou maskou (kernelem)
- Sobel Operator – aplikace Sobelova operátoru na obraz
- Embossing – „vytlačení" vzorů obrazu z pozadí
- Gaussian Blur – rozmazání obrazu podle Gaussovské funkce podle zadaných parametrů

2.3 Uživateslké rozhraní a ovládání

Uživatelské rozhraní se skládá z hlavního okna, kde v levé části je zobrazen načtený video soubor a v pravé se nachází ovládací prvky. Ve spodní části je přítomen stavový řádek, který uživatele informuje o poslední provedené akci. Zobrazení reaguje na změnu velikosti okna aplikace a adekvátně se jí přizpůsobuje.



Ve *Video Controls* je umístěno ovládání videa a tlačítko pro jeho načtení ze souboru. Soubor se vybírá pomocí standardního otevíracího dialogu systému.

Rozhraní dále obsahuje část pro výběr a zobrazení seznamu právě aktivních filtru v pořadí jejich výběru – *Active Filters.* Po stisknutí tlačítka *Add* je otevřeno dialogové okno s možnostmi výběru a případného nastavení parametů filtrů. Pro vymazání filtru je potřeba zvolit filtr ze seznamu aktivních filtrů a stisknout tlačítko *Delete.* Při překročení viditelné plochy seznamu je možnost v něm navigovat pomocí kolečka myši.

Zvolené obrazové filtry se aplikují přímo na přehrávané video a jejich efekt je vidět okamžitě po jejich přidání do seznamu aktivních filtrů. Pro konverzi celého videa a její uložení do souboru je připravena sekce *Convert To File,* kde po stisknutí tlačítka *Convert* se otevře dialog pro určení umístění a názvu soboru. Poté je provedena konverze videa podle aktuálně vybraných filtrů v seznamu aktivních filtrů. Průběh je signalozován indikátorem průběhu (progress bar).

3 Dosažené výsledky

V projektu jsme splnili zadání a zároveň jej vylepšili. Program obsahuje uživatelsky přívětivé grafické rozhraní pro snadné ovládání. Filtrování obrazu probíha v reálném čase za využití výkonu grafického adaptéru (GPU) při zpracování obrazu. Díky tomu je také konverze do souboru podstatně rychlejší, než při klasiké implementaci pouze na procesoru (CPU).

Aplikace umožňuje přehrávání načteného video souboru a zobrazení výstupů právě zvolených filtrů. Jejich efekty jsou vidět okamžitě po jejich přidání a je možné je za sebou řetězit, čímž se dá dosáhnout zajímavých výsledků.

U jednotlivých filtrů je možnost nastavení jejich parametrů tam, kde to dává smysl. Filtrace probíha v odděleném vlákně, takže není omezena odezva uživatelského rozhraní.

4 Sestavení projektu a testovací data

Aplikaci lze zkompilovat v Qt Creatoru 3.0. V tomto prostředí byl projekt vytvořen.

Reference

[1] WWW stránky projektu Qt. *Qt framework* [online]. Dostupné na: http://qt-project.org

[2] WWW stránky projektu OpenCV. *Knihovna OpenCV* [online]. Dostupné na: http://opencv.org

[2] WWW stránky standardu OpenCL. *The open standard for parallel programming of heterogeneous systems* [online]. Dostupné na: https://www.khronos.org/opencl/