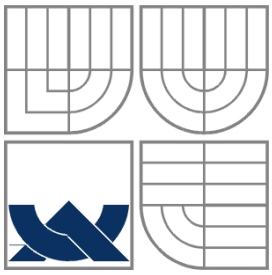
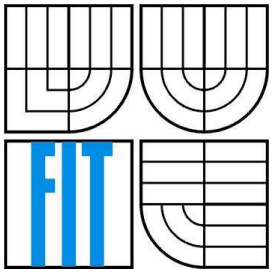


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ČASOSBĚRNÉ VIDEO

TIME-GATHERING VIDEO

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

PAVLÍNA KÁRNÍKOVÁ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. ADAM HEROUT, Ph.D.

BRNO 2007

Zadání

Časosběrné video

1. Seznamte se s problematikou časosběrného fotografování.
2. Vyhledejte, prostudujte a popište existující techniky vytváření časosběrného videa.
3. Navrhněte techniky zpracování obrazu, které umožní pořizovat kvalitní videa z časosběrných fotografií.
4. Vytvořte program, který uplatní navržené techniky při vytváření časosběrného videa.
5. Demonstrujte funkčnost programu na vhodných příkladech.
6. Zhodnoťte dosažené výsledky a navrhněte možnosti pokračování projektu; vytvořte plakátek pro prezentování projektu.

Licenční smlouva

Abstrakt

Tato práce popisuje problematiku tvorby časosběrného videa, zpracování obrazu a teorii fotografování. Cílem je vytvořit program, který by uměl upravit fotografie a následně je zpracovat do podoby videa. Program je implementován v jazycích C a C++ pomocí knihovny OpenCV. Je určen pro operační systém Windows.

Klíčová slova

Video, fotografie, AVI, zpracování obrazu, konvoluce, jas, kontrast, vzájemná informace.

Abstract

This bachelor essay describes problems of creating time-gathering video, image processing and the theory of photography. Purpose is created program, which can modify photos and after that create video. Program is implemented in the C and C++ language and uses the OpenCV library. Application can run in the operation system Windows.

Keywords

Video, photography, AVI, image processing, convolution, brightness, contrast, mutual information.

Citace

Pavlína Kárníková: Časosběné video, bakalářská práce, Brno, FIT VUT v Brně, 2007

Časosběrné video

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně pod vedením pana Ing. Adama Herouta, Ph.D.

Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

……
Pavlína Kárníková
Datum

Poděkování

Ráda bych poděkovala panu Ing. Adamu Heroutovi, Ph.D. za jeho cenné rady, nápady a připomínky v průběhu tvorby mé bakalářské práce.

Obsah

Úvod

První fotografie spatřila světlo světa v roce 1826. Od té doby zaznamenala nebývalý rozmach. Fotografování nejprve bylo záležitostí malé skupinky lidí, většinou profesionálních fotografů, ale od první poloviny 20. století se fotografie stávala čím dál tím více záležitostí i fotografů amatérů.

V roce 1981 společnost Sony uvedla na trh první fotoaparát, který místo filmu na chemickém principu zaznamenával obraz na elektronické prvky CCD. Tento nový způsob fotografování se stával postupně čím dál tím více populární a po roce 2000 aparáty používající digitální záznam začaly vytlačovat běžné kinofilmové. Výhodou digitálního fotoaparátu oproti klasickému je hlavně schopnost ukládat snímky na paměťové médium, možnost jejich následné úpravy pomocí počítače a možnost tisku jen nejzdařilejších snímků. Odpadají tím pádem finanční náklady za fotografický film a za jeho následné vyvolání. Tyto nesporné klady digitální fotografie otevřely cestu k experimentování s fotografií i pro amatérské fotografy.

Experimenty mohou fotografa dovést i k pokusu zachytit na sérii snímků běh času. Příkladem může být série snímků, na kterých je zaznamenáno, jak se rozevírá květ leknínu. Jenže dané snímky mají své omezení. Vyniknou jedině tehdy, když jsou vedle sebe nebo když jsou promítnuty bezprostředně za sebou. Pokud si toto uvědomíme, zbývá nám již jen malý krůček k opravdovému videu.

Video je technologie pro zachycování, zaznamenávání, přehrávání, přenos a obnovu pohyblivých obrázků používající elektronické signály nebo digitální média. Existuje několik způsobů jak získat video. Nejčastější je natočení videa pomocí videokamery. Bohužel, možnost vytvoření videa z digitálních fotografii, není zcela běžná. Existuje několik programů, které dokáží vytvořit krátké video. Jenže obvykle jsou součástí rozsáhlejší aplikace, která není bezplatně dostupná.

Neexistence volně dostupného programu a můj zájem o fotografování byly pro mě dostatečně silnými důvody pro zpracování tohoto tématu jako bakalářskou práci.

Cílem projektu by měla být aplikace, která umí vytvořit nejen video z fotografií, ale bude také počítat s případnými defekty na snímcích a bude umět tyto rušivé prvky odstranit. Aplikace by měla být určena široké veřejnosti, proto je u ní nezbytností vytvoření uživatelského rozhraní, které uživateli umožní ovládat jednotlivé funkce programu.

V 1. – 4. kapitole uvádím teorii fotografování, zpracování obrazu, práce s knihovnou OpenCV a tvorby uživatelských rozhraní. V 5. kapitole popisuji návrh systému a 6. kapitola je věnována tomu, jak se mi návrh povedlo realizovat. V závěru je uvedená předpokládaná práce do budoucna.

1 Časosběrné video

Problematika časosoběrného videa je rozsáhlá. Kvalita výsledného videa v prvé řadě závisí na kvalitě dodaných fotografií. Proto bylo třeba se zamyslet nad procesem pořizování fotografie. Jak bude výsledná fotografie vypadat, ovlivňuje nejen fotograf, ale také i fotoaparát. V následujících oddílech se věnuji principu pořizování fotografií a jejich případným defektům.

1.1 Pořizování fotografie

Princip digitálního fotoaparátu je poměrně jednoduchý a vychází z konstrukce klasického fotoaparátu. Jádrem přístroje je světlocitlivá plocha snímače na bázi technologie CCD nebo CMOS. Na plochu senzoru je promítán obraz přes systém optických čoček v objektivu. Světelná energie, která přichází ze snímaného prostoru (scény), je v jednotlivých pixelech (obrazových bodech) převáděna na elektrický signál a uložena v podobě vázaného náboje (u technologie CCD). Náboj vzniká postupně během expozice čipu, kdy je otevřena uzávěrka fotoaparátu a světlo může dopadat na čip. Princip vzniku elektrického náboje je založen na fotoelektrickém jevu s tím rozdílem, že náboje neodtékají okamžitě do vnějšího obvodu, ale jsou izolovány v nábojových zásobnících v elektricky izolované struktuře čipu.

Po uzavření uzávěrky jsou vygenerované náboje z čipu postupně odváděny a měřeny speciálním zesilovačem pro každý jednotlivý pixel. Takto získaný signál je a dále převeden AD převodníkem na digitální signál v binárním kódu. Vzniklý datový proud je pak pomocí mikroprocesoru různě upravován a převeden do některého grafického formátu používaného pro záznam obrazových dat, např. JPG nebo TIFF.

Dnes se téměř výhradně používají digitální fotoaparáty se snímači umožňujícími pořizovat barevné fotografie. To ve většině případů zajišťuje tzv. Bayerova maska, v níž jsou z každých čtyř buněk snímače dva překryty zeleným filtrem, jeden červeným a jeden modrým. Toto uspořádání je dáno návazností na spektrální citlivost lidského zraku, který je v oblasti zelené barvy nejvíce účinný. Obvyklý čtyřmegapixelový snímač tedy obsahuje dva miliony bodů citlivých na zelenou, a po milionu bodů citlivých na červenou a modrou. Zbylá barevná informace se ve výsledném snímku dopočítává.

Výjimkou jsou senzory Foveon, složené ze tří vrstev citlivých na různé barvy. U nich má každý pixel plnou barevnou informaci, a interpolace tedy není třeba. Dalším alternativním typem senzorů je Super CCD, které mají čtvercovou síť otočenou o 45˚, viz [6].

1.1.1 Clona, expoziční čas, ISO

Správná expozice je bezesporu jedním z klíčových faktorů na cestě ke kvalitní fotografii. Co naplat, že se vám podařil životní záběr, když je beznadějně pod nebo přeexponován. Přitom expozici ovlivňují pouhé 3 faktory - expoziční čas, ISO citlivost a clona.

1.1.1.1 Expoziční čas (rychlost závěrky)

Prvním způsobem, jak ovlivnit expozici, je měnit dobu působení světla na senzor. Tím, že senzor v zásadě počítá dopadající fotony, tak expoziční doba ovlivňuje jejich počet. Stupnice ale není lineární, nýbrž logaritmická. V praxi to znamená, že sousední hodnota na stupnici expozičních časů mění dobu a tím i množství světla vždy 2x.
Typická stupnice expozičních časů je:

.., 8, 4, 2, 1, 1/2, 1/4, 1/8, 1/15, 1/30, 1/60, 1/125, 1/250, 1/500, 1/1000, ... vteřiny

1.1.1.2 ISO citlivost

Druhým způsobem, jak ovlivnit expozici, je změnit citlivost senzoru. Čím vyšší citlivost, tím menší množství světla stačí ke správné expozici. Citlivost se udává v jednotkách ISO a opět sousední hodnota na stupnici ISO mění citlivost vždy 2x. Typická stupnice ISO je:

..., 50, 100, 200, 400, 800, 1600, 3200, ...

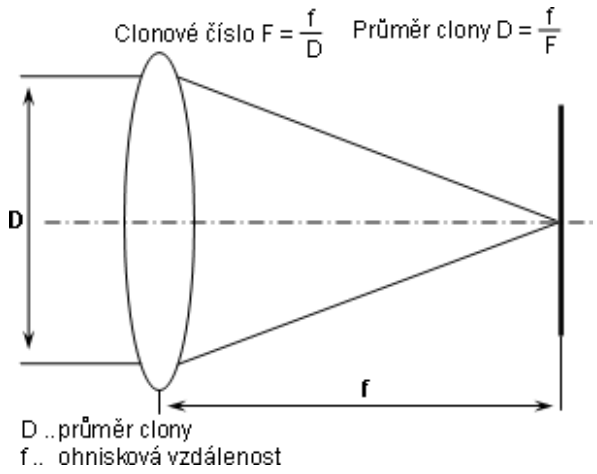
Pokud tedy zvýšíme ISO citlivost 2x (např. z ISO 100 na ISO 200), ke stejné expozici stačí poloviční množství světla (fotonů). Velkou výhodou digitálních fotoaparátů je fakt, že je možné snadno nastavovat ISO, klidně i pro každý snímek jinak.

1.1.1.3 Clona a clonové číslo

Posledním způsobem, jak ovlivnit expozici, je měnit množství světla, které projde objektivem neboli přivírat kruhový otvor ve středu objektivu - clonu.

Množství světla dopadajícího na senzor závisí nejen na otvoru clony, ale též na *ohniskové vzdálenosti objektivu*. Uvažovat ale při expozici ještě s ohniskovou vzdáleností je dost nepraktické. Proto se zavedla *clonová čísla, F*, která z úvah ohniskovou vzdálenost vyřazují. Clonové číslo F např. 2.8 tak zajistí stejné množství světla na senzoru u objektivu s ohniskovou vzdáleností 15mm i 300mm.

průměr clony v mm = aktuální ohnisková vzdálenost v mm / clonové číslo



Obr 1. Clona

Výsledkem je tedy stupnice clonových čísel, která podobně jako expoziční čas a ISO zajišťuje násobky expozice 2x, ale plocha clony roste s druhou mocninou průměru clony. Stupnice clonových čísel proto nejsou násobky 2, ale násobky odmocniny ze 2 tj. ~1.4:

1.0, 1.4, 2.0, 2.8, 4.0, 5.6, 8, 11, 16, 22, 32, 45, ...

1.1.2 Ostření a hloubka ostrosti

Ostrý snímek je vedle expozice další klíčový faktor na cestě ke kvalitní fotografii. Subjektivní ostrost snímku však ovlivňuje řada faktorů - expoziční čas, zaostření, objektiv, hloubka ostrosti i míra a způsob softwarového doostření.

Správně zaostřit fotoaparát znamená zajistit, aby to, co nás na snímku zajímá, bylo ostré. Prakticky to znamená zvolit rovinu zaostření (Focal plane). Rovina zaostření určí zaostřovací vzdálenost, v které se budou předměty ostře zobrazovat na senzor. Předměty byť sebeméně před či za rovinou zaostření budou vždy neostré. Velikost rozostření v blízkosti roviny zaostření není však nijak dramatická a často je okem nepostřehnutelná. Čím dále jsou však předměty od roviny zaostření, tím více jsou rozostřeny.

Aby objektiv dokázal volit rovinu zaostření, musí mechanicky pohybovat jednou nebo více čočkami. Tento pohyb může zajistit ruční otáčení zaostřovacím kroužkem (Focusing ring), u objektivů schopných automatického zaostřování (Auto Focus, AF) je nutné objektiv vybavit motory.

Má-li objektiv zaostřeno na určitou vzdálenost (Focusing distance), tak světelný bod v této vzdálenosti se na senzoru opět zobrazí jako bod a v důsledku toho se všechny hrany v této vzdálenosti zobrazí též jako hrany. Prakticky to potom znamená, že předměty v této vzdálenosti se jeví ostré.

Světelný bod umístěný dále či blíže než je zaostřovací vzdálenost má svůj ostrý obraz před či za senzorem, a tak na senzoru nevytvoří ostrý bod, ale rozmazaný kruh a hrany se zobrazí jako plynulý přechod. Prakticky se předměty umístěné mimo rovinu zaostření budou jevit jako rozmazané, viz [7].

1.2 Očekávané problémy při tvorbě videa

Při tvorbě videa je třeba počítat s „nejhorší“ variantou, tedy že dodané fotografie pořídil člověk, který toho o fotografování mnoho neví, tedy amatérský fotograf.

1.2.1 Snímky nepasují přesně na sebe

Amatérský fotograf málokdy plánuje dlouhodobě dopředu, co vyfotografuje. Proto nejinak tomu bude i u snímků, které bude chtít použít pro časosběrné video. Pravděpodobně se mu zalíbí nějaká událost a jednoduše začne pořizovat fotografie. Jenže nebude mít u sebe stativ. Každá fotografie bude pořízena trochu z jiného místa, protože fotoaparát uživatel držel v ruce. Jaký to bude mít vliv na budoucí video? Obraz bude působit roztřeseným dojmem.

1.2.2 Rozdílná úroveň jasu u jednotlivých snímků

V předchozí části jsem již zmínila základní teorii fotografování a princip funkce fotoaparátů. Málokterý amatérský fotograf rozumí výše zmíněné teorii a dává přednost koupi fotoaparátu, který řeší problémy s expozičním časem a se clonou za něj nebo jednoduše si nemůže z finančních důvodů koupit kvalitní digitální fotoaparát s manuálním ovládáním clony, expozičního času a citlivosti ISO. Co z toho vyplývá pro vzniklé časosběrné fotografie? U každé z nich určoval čas, clonu a ISO automat. Může se tedy stát, že jeden nebo více parametrů vyhodnotí u každé fotografie jinak a tím změní výsledný obraz hlavně po jasové stránce. Tato vlastnost nevadí u samostatných snímků, ale při spojení fotografií do videa se to projeví jako nepříjemné blikání obrazu.

1.2.3 Nechtěný objekt na jedné z fotografií

Na další nechtěnou vlastnost jsem přišla až při tvorbě testovacích snímků. Například projede v zabírané scéně auto, ale protože scéna je snímána vždy jen po určitém časovém intervalu, bude automobil jen na jedné fotografii a na ostatních již ne. Ale tím nám již narušil celý soubor fotografií a bylo by vhodné tento rušivý element odstranit.

1.2.4 Ostatní nežádoucí defekty

Ve výčtu komplikací, které mohou při pořizování fotografií nastat, bych mohla i nadále pokračovat. Dále lze zmínit špatném zaostření a tím pádem o rozmazání fotografie, ale tento a další defekty fotografií již nejsou tak časté a proto spoléhám na uživatele, že výrazně nevydařené fotografie z celého souboru pro budoucí video vyřadí.

2 Zpracování obrazu

Zpracování obrazu je odvětví využití počítačů, které se zabývá digitalizací obrazu, zpracováním digitalizovaného obrazu, ukládáním digitalizovaného obrazu na počítačová média, reprodukcí těchto obrazů, předzpracováním obrazů pro další odvětví výpočetní techniky a využitím výše uvedených postupů v aplikacích, viz [1]

Prvním krokem zpracování obrazu je snímání, digitalizace a uložení obrazu v číselné formě do počítače. Při snímání se převádějí vstupní optické veličiny na elektrický signál spojitý v čase i úrovni. Vstupní informací může být jas (z kamery, z fotoaparátu, ze scanneru), nebo několik spektrálních složek (červená, zelená, modrá) při barevném snímání.

Digitalizací se převádí vstupní spojitý signál do diskrétního tvaru. Vstupní analogový signál je popsán funkcí f(x,y) dvou proměnných – souřadnic v obraze. Funkční hodnota odpovídá např. jasu. Vstupní signál je vzorkován a kvantován. Výsledkem je matice čísel popisujících obraz, kde jednomu prvku se říká obrazový element – pixel, viz [2].

2.1 Barevné modely, jas a kontrast

Vzhledem k tomu, že bakalářská práce se věnuje mimo jiné také úpravě barevných digitálních fotografií, považuji za vhodné se v této kapitole okrajově zmínit o dvou základních barevných modelech, se kterými jsem v při návrhu nebo při implementaci programu přišla do styku.

Barevné modely se používají především pro zjednodušení záznamu barevné informace. Jejich pomocí je definován soubor základních barev, pravidla jejich míchání a měněné barevné charakteristiky.

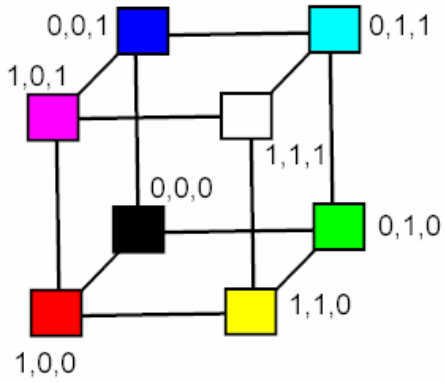
S barevnými modely úzce souvisí další vlastnosti obrazu a to jas a kontrast. Obě tyto veličiny nám pomáhají popsat, jak oko vnímá daný obraz.

2.1.1 RGB

Model RGB vychází z faktu, že lidské oko obsahuje tři základní druhy buněk citlivých na barvu. Tyto buňky jsou citlivé na vlnové délky, které zhruba odpovídají červené (vlnová délka 630 nm), zelené

(530 nm) a modré (450 nm) barvě. Kombinací červené, zelené a modré lze získat téměř všechny barvy barevného spektra.

RGB model je součtový a lze jej vyjádřit pomocí jednotkové krychle, kdy v počátku (0,0,0) leží černá barva a v protilehlém vrcholu (1,1,1) barva bílá - obecně lze říci, že v protilehlých vrcholech krychle leží vzájemně komplementární barvy, jejichž součtem získáme bílou barvu. Barevné odstíny vznikají skládáním základním barev, jejichž intezita se udává v intervalu <0,1>. V počítačové grafice se používá dělení intervalu intezity základní barvy na 256 dílů (0-255) - libovolnou barvu pak můžeme udávat 24 (3x8) bity - barvy udávané pomocí 24 bitů označujeme jako true colors. V rámci true colors můžeme zobrazit 256^3, tzn. 16 777 216 barev. K vyjádření jednotlivých barevných složek se často používá hexadecimální číselná soustava, konkrétně hodnoty 0-ff. Barevnou reprezentaci RGB modelu (ale i jiných barevných modelů) na 15 a 16 bitech nazýváme high color.



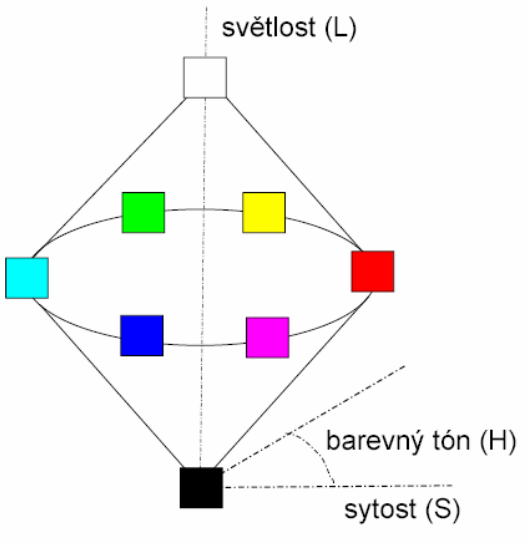
Obr. 2. Barevný model RGB

Variantou modelu RGB je model označovaný jako **RGBA**. Jde o klasický model RGB, kdy barva navíc nese informaci o průhlednosti (alfa kanál). Tato hodnota, která se ukládá do jednoho bytu, určuje poměr smíšení barvy s barvou pozadí. Zcela neprůhledná má hodnotu A rovnou jedné, naopak průhledná (transparentní) barva má hodnotu alfa kanálu rovnou 0. Barevný prostor RGBA používá například rastrový grafický formát PNG.

2.1.2 HLS

Model HLS má tři parametry: sytost, barevný odstín a světlost. Sytost leží na vodorovné ose, světlost na svislé ose a barevný tón představuje úhlová hodnota. Tvar modelu odpovídá skutečnosti. Schopnost rozlišování barevných odstínů skutečně klesá se ztmavováním a zesvětlováním základní čisté barvy, zvyšování a snižování světlosti barvy skutečně spočívá v přidávání světlého nebo

tmavého pigmentu. Model HLS bývá někdy nazýván modelem psychologickým nebo psychofyzikálním. Model HLS, na rozdíl od RGB, umožňuje měnit jeden parametr barvy, zatímco ostatní dva zůstanou zachovány. Tato možnost je důležitá pro počítačové grafiky, tiskaře i kartografy a já o ní uvažovala v souvislosti se změnou jasu fotografie v kapitole 5.2.1 Nástroj na úpravu jasu. Více informací viz [3].



Obr. 3. Barevný model HLS

2.1.3 Jas

Okem vnímaný jas je logaritmickou funkcí intenzity přicházejícího světla, viz [11]Lidské oko vnímá různým způsobem intenzitu barevných složek (nejcitlivější je na zelenožlutou), proto je nutné používat pro výpočet jasu empirický vztah:

$$I = 0,299\ R + 0,587\ G + 0,114\ B.$$

2.1.4 Kontrast

Slovo kontrast znamená nepodobnost nebo také rozdíl mezi dvěmi věcmi. V oblasti počítačové grafiky se kontrast obvykle týká rozdílu mezi jednotlivými barvami. Pokud však pořizujeme fotografii, bereme kontrast obvykle jako vlastnost dané scény, tedy jednoduše řečeno rozdíl mezi nejsvětlejším a nejtmavším místem fotografie.

2.2 Detekce míst v obraze

Jak bylo v první kapitole zmíněno, někdy se stane, že jednotlivé fotografie pro časosběrné video budou pořízeny každá z trochu jiného místa. Abychom je mohli na sebe správně napasovat, je nutné nejprve detekovat místo, které je zachyceno na všech snímcích..

2.2.1 Vzájemná informace

Vzájemná informace je mírou množství informace, kterou obsahuje jedna náhodná proměnná o druhé, viz [4].
Je netradičním způsobem měření podobnosti dat. Odhaduje obecnou (tj. i nelineární) závislost dvou souborů dat.

Registrace pomocí vzájemné informace je tedy robustnější a je vhodná i pro registraci dat získaných z různých zobrazovacích modalit. Algoritmy pro její výpočet jsou velmi obecné a jsou použitelné pro široké spektrum dat.

Výpočet vychází ze statistické teorie: 2 náhodné veličiny x a y jsou statisticky nezávislé právě tehdy, když jejich simultánní hustota pravděpodobnosti je součinem jejich marginálních hustot. Tedy platí:

$$p(x, y) = p_x(x)p_y(y)$$

a maximálně statisticky závislé jestliže platí:

$$p(x, y) = p_x(x) = p_y(y).$$

Vzájemná informace náhodných veličin I a T(J) měří stupeň jejich závislosti Kullback-Leiblerovou vzdáleností mezi sdruženou hustotou pravděpodobnosti $p_{IJ}(I(i), J(i))$ a součinem marginálních hustot $p_I(I(i))p_J(J(i))$.

Vypočítá se vztahem (jednotkou jsou bity):

$$MI(I, J, T) = X_i \; p_{IJ}(I(i), T(J)(i)) \log_2 \frac{p_{IJ}(I(i), T(J)(i))}{p_I(I(i))\, p_J(T(J)(i))}$$

Optimální registrace je dosaženo pro maximum funkce. Jiný způsob výpočtu vzájemné informace využívá Shannonových entropií:

$$MI(I, J, T) = H(I) + H(J(T)) - H(I, T(J)), \quad (2.7)$$

kde H(I) a H(T(J)) jsou Shannonovy entropie veličin I a T(J) a H(I,T(J)) jejich vzájemná entropie:

$$H(I) = -X_i$$
$$p_I(I(i)) \log_2 p_I(I(i)), \quad (2.8)$$
$$H(T(J)) = -X_i$$
$$p_J(T(J)(i)) \log_2 p_J(T(J)(i)), \quad (2.9)$$
$$H(I, T(J)) = -X_i$$
$$p_{IJ}(I(i), T(J)(i)) \log_2 p_{IJ}(I(i), T(J)(i)). \quad (2.10)$$

2.2.2 Konvoluce

Konvoluce je matematický operátor zpracovávající dvě funkce. Spojitá konvoluce (značí se hvězdičkou) jednorozměrných funkcí f(x) a h(x) je definována vztahem:

$$f(x) * h(x) = \int_{-\infty}^{\infty} f(x-\alpha)h(\alpha)\,\mathrm{d}\alpha$$

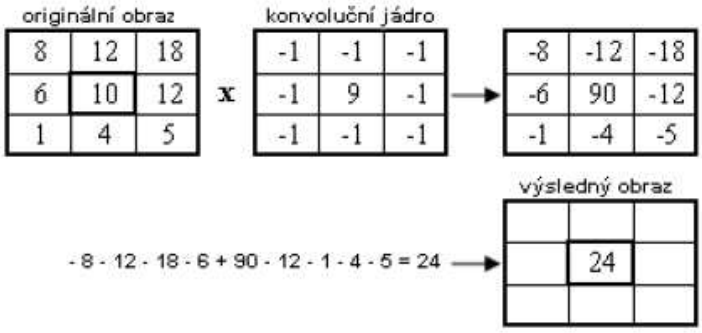
Funkci $h(\alpha)$ se říká konvoluční jádro. Protože jinak celý výraz je vždy stejný, jsou operátory definovány právě jen konvolučním jádrem.

2.2.2.1 Využití v počítačové grafice

Konvoluce se často používá při algoritmech zpracování dvourozměrného diskrétního obrazu v počítačové grafice. Vzorec diskrétní konvoluce má potom tvar:

$$f(x,y) * h(x,y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f(x-i, y-j) \cdot h(i,j)$$

V případě diskrétní konvoluce lze jádro chápat jako tabulku (konvoluční maska), kterou položíme na příslušné místo obrazu. Každý pixel překrytý tabulkou vynásobíme koeficientem v příslušné buňce a provedeme součet všech těchto hodnot. Tím dostaneme jeden nový pixel.



Obr. 4. Výpočet konvoluce

Koeficienty uvnitř konvoluční masky udávají vliv hodnoty pixelu pod nimi. Lze tak nadefinovat velké množství operací např. derivace obrazu (u diskrétního obrazu mluvíme o tzv. odhadu derivace), tedy zvýraznění hran, viz [5].

2.3 Filtrace obrazu

Filtrací obrazu rozumíme v obecné rovině operace s digitálním obrazem, které slouží ke zvýraznění nebo k potlačení určité informace. Rozlišujeme celou řadu metod filtrace obrazu od velmi jednoduchých (např. prosté průměrování) až po poměrně komplexní, sofistikované metody.

Filtrace může být využívána pro vyhlazení obrazu, potlačení šumu, zvýraznění kontrastu, detekci hran, postklasifikační zpracování obrazu a řadu dalších úloh. Vzhledem k rozsahu většiny digitálních dat je z technického hlediska nevhodné řešit podobné úlohy najednou v celém obrazu. Daný filtr je tak definován jako šablona rastrové matice (tzv. "moving window", v české literatuře se často využívá termín "kernel") - tedy pohybujícího se (plovoucího) okna. Jde o matici (většinou čtvercovou) tvořenou lichým počtem řádků a sloupců, která se při výpočtu pohybuje nad maticí originálních dat (ve směru řádků a sloupců). Nová hodnota rastrové buňky je určena na základě aritmetické operace či statistické veličiny definované filtrem a hodnot originálních dat. Do výpočtu tak vstupuje na rozdíl například od *podílu obrazu* množina hodnot a nikoliv pouze jedna jediná hodnota. Jde tedy o *lokální* a nikoliv *bodové* operace. Nejčastěji se využívá velikost kernelu 3x3 a 5x5.

Filtry lze rozdělit na dva základní typy:

- filtry s nízkou (*low pass*) propustností, které ořezávají vysoké frekvence (dochází tak k potlačení výraznějších detailů (např. liniových prvků) a současně zdůraznění jevů s malými změnami)
    - omezují odchylky od lokálního průměru
    - vyhlazuje detaily původního obrazu

    - redukují rozsah stupňů šedi
- filtry s vysokou (*high pass*) propustností, které naopak zdůrazňují vysoké frekvence (to znamená zdůraznění výraznějších detailů a současné potlačení jevů s malými změnami). Jde především o ostřící filtry a hranové operátory.
    - zvětšují detaily v ploše snímku

2.3.1 Průměrování

Klasické průměrování znamená, že nová hodnota (zaokrouhlena na celé číslo) středového pixelu je vypočtena jako aritmetický průměr hodnot pixelů pokrytého kernelem. Ve svém důsledku tento filtr likviduje liniové elementy (jde o nízkofrekvenční filtr), naopak je vhodný pro vyhlazení výsledku klasifikace u rozsáhlejších homogenních ploch či pro částečné odstraňování šumu. Rovnice pro velikost kernelu 3x3:

$$p(i,j) = \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f(i+k, j+l)$$

3 OpenCV

OpenCV je knihovna určená na zpracování obrazu. Vyvinula ji společnosti Intel specificky pro jejich procesory. Knihovna může být používána buď v jazyce C, nebo Python. Součástí knihovny jsou také ukázkové příklady, které uživateli dají představu, co vše je v knihovně dostupné a dají se na nich pochopit základní principy práce s OpenCV.

Nevýhodou OpenCV je velmi skromná dokumentace a malé množství příkladů, které jsou dostupné na internetu. Nicméně na internetu existují diskusní skupiny z nichž asi největší je na serveru Yahoo, která čítá 20 000 členů.

Knihovna OpenCV je open-source. Vývoje knihovních funkcí se mohou účastnit i její uživatelé. Stačí, když svůj vytvořený algoritmus poskytnou společnosti Intel. Její zaměstnanci zlepší jeho efektivitu a kód pomocí jejich instrukčních souborů. Algoritmus je následně vložen do příštího vydání OpenCV a originální tvůrce algoritmu získá algoritmus s mnohem větší efektivitou. Tímto způsobem získala knihovna více než 400 různých funkcí, které jsou natolik flexibilní, že mohou být užity jak v oblasti zpracování obrazu, tak i ve výzkumu a v jiných oborech.

3.1 Práce s fotografiemi pomocí OpenCV

Pro moji bakalářskou práci je rozhodující práce s fotografiemi. Pro jejich zpracování a manipulaci s nimi poskytuje OpenCV velké množství funkcí a struktur. V této kapitole bych se proto ráda, alespoň okrajově, zmínila o nejdůležitějších z nich.

K reprezentaci obrazu v knihovně OpenCV slouží struktura *IplImage*. Její důležitou součástí je *depth* (hloubka), kde je uloženo jaké hodnoty bude nabývat jednotlivý pixel obrazu. K dispozici je několik typů od *unsigned char* (8 bitů) až po *float* (32 bitů). Další rozhodující vlastností obrazu ve struktuře *IplImage* je počet kanálů. Obraz ve stupních šedi bude mít jeden kanál, zato barevný obraz má tři.

Za zmínku stojí, že obrazy nejsou uloženy tradičně v RGB prostoru, ale v BGR prostoru. Další zvláštností je, že barevné obrazy nejsou uloženy po pixelech. Místo toho jsou uloženy jako pole barevných úrovní, které znamená menší režii pro procesor ohledně přístupu k jednotlivým hodnotám. Datové pole *IplImage* vypadá následovně:

imageData[0]	imageData[1]	imageData[2]	imageData[3]	imageData[4]	...
B	G	R	B	G	...

Obvykle ale bývají obrazová data uložena následovně:

imageData[0]			imageData[1]		
→red	→ green	→blue	→red	→ green	→blue

Obvyklý způsob uložení dat má jednu nespornou výhodu. Snadný a intuitivní přístup k pixelům pro programátora. Proto jednou z prvních věcí, které jsem v OpenCV udělala, bylo nadefinování struktur pro jednoduchý přístup k jednotlivým barevným složkám pixelu. Inspiraci pro toto jsem nalezla na webové stránce: **Introduction to programming with OpenCV** viz [8].

Do struktury *IplImage* je někdy také potřeba načíst daný obrázek. To se děje prostřednictvím funkce *cvLoadImage.* Tato funkce si umí poradit nejen s obrázky a fotografiemiv různých obrazových formátech (například JPG, PNG, TIFF, BMP a jiné), ale také umí do struktury načíst obraz přímo ve zvoleném formátu a nezáleží na tom, jaký byl původní. To znamená, že barevná fotografie může být ve struktuře následně uložena ve stupních šedi, nebo naopak původně obraz ve stupních šedi bude mít tři barevné kanály.

Když už bylo zmíněno načítání obrazu, je také vhodné si daný obraz později uložit. K tomu slouží funkce *cvSaveImage*, která zvolený obrázek uloží v daném obrazovém formátu. Bohužel při vývoji svého programu jsem zjistila, že tato funkce neumí uložit obrázek ve 32bitovém formátu a je ho nutné konvertovat na 8 bitů.

V OpenCV lze také vytvořit zcela nový obraz. Pomocí funkce *cvCreateImage*, která má za parametry rozměry obrazu, depth a počet barevných kanálů.

Na závěr kapitoly je třeba se zmínit o funkci *cvRelaseImage*, která nám strukturu *IplImage* uvolní, viz [9].

3.2 Funkce pro práci videem v OpenCV

Funkcí pro tvorbu a práci s videem má OpenCV poněkud více než by se dalo očekávat. Já jsem se konkrétně zaměřila na funkce spojené s videem typu AVI a v této kapitole bych je ráda představila a uvedla jejich nejdůležitější vlastnosti.

K pomyslnému uchopení videa slouží struktura *CvCapture*. Tato struktura se alokuje navázáním na daný video soubor s příponou avi pomocí *cvCaptureFromAVI*. Pokud si ovšem přejeme získat z videa jeden snímek, je třeba použít funkci *cvQueryFrame*, která vrací ukazatel na strukturu *IplImage.*

Nový video soubor se v OpenCV dá vytvořit pomocí dvou různých sad funkcí, které si jsou do jisté míry velmi podobné a dají se zaměňovat. Nejprve je třeba vytvořit strukturu typu *CvVideoWriter* a naplnit ji pomocí funkce *cvCreateVideoWriter*. Jejími parametry je jméno souboru, kód codeku, fps, rozměry snímku a typ barevnosti. Fps znamená počet snímků za sekundu. Typ může nabývat hodnot 0 (pro černobílé video) a 1 (pro barevný snímek).

Kód codeku může nabývat následujících hodnot:

```
CV_FOURCC('P','I','M','1')    = MPEG-1 codec
CV_FOURCC('M','J','P','G')    = motion-jpeg codec (does not work well)
CV_FOURCC('M', 'P', '4', '2') = MPEG-4.2 codec
CV_FOURCC('D', 'I', 'V', '3') = MPEG-4.3 codec
CV_FOURCC('D', 'I', 'V', 'X') = MPEG-4 codec
CV_FOURCC('U', '2', '6', '3') = H263 codec
CV_FOURCC('I', '2', '6', '3') = H263I codec
CV_FOURCC('F', 'L', 'V', '1') = FLV1 codec
```

Pod Win32 je možné uvést kód -1, který byl měl otevřít v OS Windows okno pro výběr kodeku.
Volba kódu 0 znamená, že video bude bez komprese.

Samotné zapisování fotografií do videa probíhá pomocí funkce *cvWriteFrame*, které předáme ukazatele na video writer a na obrázek ve struktuře *IplImage*, který chceme do videa zapsat.

Možná se někdo podiví, že jsem zde ve výčtu funkcí neuvedla žádnou typu „saveVideo“. To je dáno tím, že funkce *cvWriteFrame* zapisuje snímky přímo do souboru uvedeného ve funkci

cvCreateVideoWriter a ne do žádného pomocného formátu. Po skončení zapisovaní můžeme strukturu uvolnit pomocí *cvReleaseVideoWriter*.

Druhou možností, jak vytvořit video soubor, je použít soubor funkcí se zkratkou AVI v názvu. Struktura pro tvorbu videa by se měla jmenovat: *CvAVIWriter*. Funkce pro naplnění struktury má název *cvCreateAVIWriter* a liší se od své obdoby tím, že jí schází parametr typ (neurčuje se barevnost snímku). Funkce pro zapisování do souboru se jmenuje *cvWriteToAVI*. Nadefinovaná struktura by se měla uvolnit funkcí *cvReleaseAVIWriter*.

Pozorný čtenář si ale jistě všiml, že jsem v předchozím odstavci dvakrát použila podmiňovací způsob. Je to tím, že i když jsem v několika referenčních manuálech nalezla uvedenou strukturu *CvAVIWriter*, ve skutečnosti tato struktura neexistuje. V OpenCV zkrátka není definována a tím pádem funkce *cvRelaseAVIWriter*, i když existuje, nemá reálné využití. V kapitole implementace proto naleznete, jak jsem se s tímto nečekaným problémem vypořádala.

4 Uživatelské rozhraní

Důležitou součástí mojí bakalářské práce je také uživatelské rozhraní. Výsledný program by měl být určen široké veřejnosti. Uživatel potřebuje mít přehled o dostupných fotografiích, o jejich časové posloupnosti a musí mít možnost je snadno a efektivně upravovat. Všechny tyto vlastnosti by šly jen velice těžko realizovat pomocí konzolové aplikace a tak tvorba uživatelského rozhraní byla samozřejmostí.

4.1 Co je to uživatelské rozhraní?

Za uživatelsky orientovaná rozhraní lze považovat taková rozhraní, která jsou orientována na uživatele a u nichž prvky rozhraní neodpovídají přímo funkčním celkům zařízení. Rozhraní tohoto typu se typicky objevují v aplikacích a na výrobcích, u nichž uživatel nemusí rozumět principu funkce.

Příkladem mohou být programy pro kancelářské práce, které se snaží uživatele „odstínit“ od principu práce počítače a soustředí se na vlastní práci uživatele.

Různá rozhraní orientovaná na uživatele je vhodné sjednotit, zejména ve smyslu shodné realizace stejných funkcí. Sjednocení nabízí uživateli možnost práce s různými aplikacemi, aniž by bylo nutno věnovat příliš velkou pozornost základním ovládacím prvkům, jejichž ovládání se uživatel učí jen jednou.

Tomuto trendu napomáhá v současnosti zdaleka nejužívanější způsob komunikace člověka s počítačem, což je komunikace prostřednictvím obrazu, tedy pomocí grafického výstupu počítače a vstupu dat do počítače pomocí kombinace klávesnice a grafických polohovacích zařízení (obvykle myš nebo pero na grafickém tabletu).

4.2 Současné trendy ve tvorbě uživatelských rozhraní

Grafický výstup se obecně považuje za vhodný prostředek pro komunikace člověka s počítačem prostřednictvím statického obrazu. V současnosti však grafický výstup dostává novou kvalitu, která se často nazývá „multimédia“.

Současná uživatelská rozhraní si kladou za cíl efektivní využití všech funkcí, které daná aplikace nabízí a zároveň snadnou ovladatelnost aplikace. Dříve typický systém nabídek, panelů nástrojů, podoken úloh a dialogových nástrojů ustupuje do pozadí. Tento systém fungoval dobře, ale jen pokud měla aplikace omezený počet příkazů. V současné době mají programy mnohem více funkcí a systém nabídek již tak dobře nefunguje. Pro mnoho uživatelů je příliš těžké nalézt velké množství funkcí programu. Z tohoto důvodu je cílem nového uživatelského rozhraní usnadnit uživatelům nalezení a použití všech funkcí, které tyto aplikaci nabízejí. Dalším požadavkem je zachování přehledného pracovního prostoru, který méně rozptyluje uživatele a ti se tak mohou lépe soustředit na svou práci.

Výsledkem jsou aplikace, které sice mají typický systém nabídek, ale zároveň jsou uživateli přímo nabízeny jen ty panely nástrojů, které jsou právě potřeba při jeho současné činnosti. Jiný výrazný rys uživatelských rozhraní jsou galerie, které uživateli nabízejí představu, jak by mohl výsledný produkt po užití dané funkce vypadat. Celkově se snaží uživatelské rozhraní o co nejintuitivnější práci s aplikací za užití audiovizuálních prostředků.

5 Návrh systému

Jedním z prvních kroků na každém projektu je návrh systému. Nejinak tomu bylo i u této bakalářské práce. Bylo třeba si určit, co od budoucí aplikace očekávám a zároveň vyhovět zadání. Naštěstí zadání práce mi poskytlo dostatek volnosti, abych se mohla plně řídit svou představou o budoucí aplikaci.

Aplikace by měla být určena pro širokou veřejnost. Bude mít tedy uživatelské rozhraní zaměřené na snadné a intuitivní ovládání jednotlivých funkcí. Měla by také obsahovat nástroje pro eliminaci chyb a nechtěných efektů při pořizování fotografií.

5.1 Nástroje budoucí aplikace a jejich předpokládané algoritmy

Z výše uvedených vlastností fotografií vyplývá, že před tvorbou vlastního videa bude třeba jednotlivé snímky upravit. Proto jsem si již na začátku utvořila představu o nástrojích na úpravu fotografií, které bych chtěla v aplikaci implementovat.

5.1.1 Nástroj na úpravu jasu

Aplikace bude obsahovat nástroj na úpravu jasu jednotlivých fotografií. Na tuto klasickou operaci existuje několik algoritmů.

- Klasický postup při úpravě jasu je operace pomocí aplikace nějaké funkce na hodnoty v obraze.
- Další možností je převedení fotografie z barevného modelu RGB na model HLS, kde je hodnota světlosti/jasu přímo definovaná a tím pádem její změna bude snadná.

Z uvedených algoritmů tedy vyplývá, že zcela jistě mezi vlastnostmi tohoto nástroje bude vstup programu pro zadání požadované hodnoty jasu a případně i kontrastu.

5.1.2 Nástroj pro úpravu pozice snímků

U toho nástroje jsem si v době práce na návrhu aplikace ještě nebyla jistá, jakým způsobem ho budu implementovat. Napadaly mě myšlenky spíše na nějaký detektor hran nebo ploch v obraze, podle kterých by se nějakým způsobem dala určit míra posunutí jednotlivých snímků a podle toho je napasovat na sebe. Jak jistě čtenář vidí, byly to značně nekonkrétní představy.

Měla bych si zvolit referenční fotografií. Z její malé časti si vytvořit konvoluční jádro a uplatnit konvoluci s tímto jádrem na ostatní fotografie. Konvoluce by mi měla detekovat totéž místo i na ostatních fotografiích. Dále již pak by mělo být snadné určit ořezání fotografií tak, aby na sebe přesně pasovaly.

Také jsem o další, „záchranné“, variantě. Jednalo by se o to, že by uživatel u každé fotografie označil ručně jedno a totéž místo a program by podle toho fotografie na sebe napasoval.

5.1.3 Nástroj na eliminaci nechtěného objektu na jedné z fotografií

Tato funkce by měla následovat po úspěšném napasování snímků na sebe. Pokud by se na jedné z fotografií vyskytl nechtěný objekt. Uživatel bude mít možnost ho ručně označit a spustit funkci.

Celý nástroj by měl fungovat na principu průměrování. Vzaly by se hodnoty pixelu v daném místě (nebo v daném místě v jeho okolí) z předcházející fotografie a ze snímku, který následuje po fotografii s nežádoucím objektem. Z těchto hodnot by se spočítal průměr a výsledná hodnota by se uložila na stejné místo ve fotografii s defektem.

5.1.4 Nástroj na tvorbu videa

Od začátku práce na projektu jsem bylo rozhodnuto, na jaké úrovni se budu tvorbou videa zabývat. Mělo to být na úrovni knihoven. To znamená, že jsem předpokládala existenci funkce nebo funkcí, kterým předám základní parametry a ona/ony mi vytvoří video soubor. Parametry měly být počet snímků za sekundu a případně kódování.

Původní představa aplikace do této oblasti zahrnovala i možnost přehrání vytvořeného videa, ale vzhledem k tomu, že v současnosti existuje velké množství volně dostupných přehrávačů, jsem od tohoto návrhu upustila.

5.2 Návrh uživatelského rozhraní

Když jsem již měla konkrétnější představu o jednotlivých funkcích svého budoucího programu, bylo na čase si vytvořit přibližný návrh uživatelského rozraní. Hlavně co se týče pozice jednotlivých prvků.

V předchozí kapitole jsem se věnovala uživatelskému rozhraní a nejnovějším trendům v této oblasti. V mém návrhu jsem ale raději zůstala u klasického vzhledu uživatelského rozraní.

Rozvržení prvků tedy bylo následující. Na pravé straně aplikace se bude nacházet *listbox*, do kterého se budou načítat názvy jednotlivých fotografií pomocí menu umístěného v panelu nabídek nahoře. Označená fotografie se pak vždy zobrazí na „obrazovce“ uprostřed.

Panel nástrojů jsem plánovala umístit napravo. Funkce úprava jasu a pozicování fotografií budou umístěny každá na jedné záložce, aby manipulace s nimi byla pro uživatele přehlednější. Tlačítko pro export videa se bude nacházet pod panelem nástrojů a nad ním bude možnost si zvolit parametry videa.

V dolní části okna bych ráda umístila *trackbar,* který bude propojení s prvky *listboxu* a bude znázorňovat, v jaké části časové osy videa se vybraná fotografie nachází.

6 Implementace a současný stav

V předchozích kapitolách jsem se věnovala návrhu aplikace a teorii. V této části práce bych ráda popsala to, jak jsem znalosti nabyté při studiu teorie uplatnila, jak jsem postupovala při řešení problémů a v jakém stavu je současná aplikace.

6.1 Tvorba uživatelského rozhraní ve Visual Studiu

Volba programu, ve kterém budu implementovat uživatelské rozhraní, byla další důležitou součástí mé bakalářské práce. Měla jsem na výběr buď Borland Builder nebo Visual Studio. Po kratším váhání jsem zvolila druhou možnost. Ne ani proto, že by to byl lepší program než Borland Builder, ale z toho důvodu, že s Visual Studiem jsem již měla v minulosti možnost jednou pracovat na menším projektu a mohla jsem tedy již nějaké zkušenosti zúročit. Dále jsem také vzala v potaz články na internetu, ve kterých bylo zmiňováno, že Visual Studiu je kompatibilní s knihovnou OpenCV.

S prací jsem začala tvorbou formuláře. Postupně jsem si do něj naskládala jednotlivé prvky, které jsem chtěla použít a snažila jsem se naimplementovat jednotlivé funkce. Povedlo se mi naprogramovat téměř všechny funkce, které jsem od uživatelského rozhraní vyžadovala. Jako například otevírání souborů (je možné otevřít jeden nebo i více současně). Jejich názvy se zobrazí v listboxu. Výběr souboru (fotografie) způsobí, že se fotografie zobrazí pomocí bitmapy v pictureboxu. S výběrem lze posouvat jak pomocí šipek, tak i myší. Od plánovaného trackbaru jsem ale upustila úplně. Během implementace jsem usoudila, že byl v návrhu téměř zbytečný. Místo toho jsem do aplikace zařadila progresbar , aby ukazoval v jakém stádiu zrovna je ta a ta operace. Na závěr jsem vytvořila ještě volbu „Uložit jako...“ pro případné uložení jednotlivých fotografií. Jediným nezdařeným cílem byla snaha o vytvoření volby „Vytvořit projekt“. To se mi zatím bohužel nezdařilo.

Obr.5. Uživatelské rozhraní programu na tvorbu časosběrného videa

6.2 Funkce pomocí OpenCV

Tvořit funkce v OpenCV jsem začínala postupně. Zvolila jsem si postup nejprve napsat algoritmus nanečisto a pak teprve následně se ho pokusit spojit s uživatelským rozhraním. Toto řešení se ale neukázalo nejšťastnějším. Při linkování knihovny OpenCV s objektově orientovaným prostředím ve Visual Studiu se vyskytly menší komplikace a já byla nucena pozměnit kód jedné z knihovních funkcí a to funkce cvRound. Toto byl ale jediný problém, jinak knihovní funkce ve Visual Studiu fungují nad očekávání dobře.

6.2.1 Funkce na úpravu jasu a kontrastu

Pro tuto funkci jsem použila klasický postup, protože knihovna OpenCV k tomu poskytuje již hotové funkce. Stačí jen si vytvořit vyhledávací tabulku o tolika místech, kolik je jasových úrovní. Nové hodnoty jasu jsou dány výsledkem transformace. V programu byl pro ni použit níže uvedený algoritmus a výsledek se již mohl uplatnit na vstupní obraz.

```
if( contrast > 0 ){
        double delta = 127.*contrast/100;
        double a = 255./(255. - delta*2);
        double b = a*(brightness - delta);
        for( i = 0; i < 256; i++ )
        {
            int v = cvRound(a*i + b);
```

```
            if( v < 0 )
                v = 0;
            if( v > 255 )
                v = 255;
            lut[i] = (uchar)v;
        }
    }
    else{
        double delta = -128.*contrast/100;
        double a = (256.-delta*2)/255.;
        double b = a*brightness + delta;
        for( i = 0; i < 256; i++ )
        {
            int v = cvRound(a*i + b);
            if( v < 0 )
                v = 0;
            if( v > 255 )
                v = 255;
            lut[i] = (uchar)v;
        }
    }
```

Algoritmus pro úpravu jasu a kontrastu v obraze

6.2.2 Funkce pro vyhledání shodných prvků v obraze

U této funkce jsem vyzkoušela několik postupů. Věděla jsem, že pro vyhledání shodných prvků použiji konvoluci s vhodným konvolučním jádrem, ale dlouho se mi nedařilo nalézt způsob, jak toho dosáhnout.

Nejprve jsem si vytvořila funkci, která mi prováděla konvoluci se zadaným jádrem. Otestovala jsem ji na obrázku, který se skládal z černého pozadí a bílého textu. Za konvoluční jádro jsem si zvolila písmeno „e“. Konvoluce proběhla bez problémů a v obraze se po prahování detekovali očekávaná písmena „e“ a dvakrát písmeno „a“. Když ale vezmeme v potaz podobnost obou písmenek, daly se chybné detekce očekávat a tím pádem jsem považovala konvoluci za funkční.

Zkusila jsem tu samou funkci uplatnit na fotografii a pro začátek jsem jako konvoluční jádro zvolila výřez z téhož obrázku. Kýžený výsledek se však nedostavil.

Chybu jsem hledala v uplatnění konvoluce na barevný obraz. Zkusila jsem tedy provést funkci jen nad jednou barevnou vrstvou, ale bohužel opět žádný uspokojivý výsledek.

Důvody, proč konvoluce nefungovala, bylo třeba hledat jinde. Všimla jsem si, že se vždy detekovalo nejsvětlejší místo v obraze. Prozkoumala jsem tedy opět správnou funkci konvoluce a uvědomila jsem si, jak moc je barevný obraz od černobílého rozdílný. Černobílý, na kterém jsem to testovala, mohl nabývat hodnoty jen 0 a 255. Tedy nejvyšších hodnot mohly dosáhnout jen ty místa, které měly největší shodu s jádrem. Ale u barevného obrazu pixel jednoho kanálu může nabývat hodnot 0 – 255 a tedy konvoluce s jádrem dá největší hodnotu ne tehdy, kdy panuje mezi výřezem obrazu a jádrem největší shoda, ale jednoduše tehdy, když součet všech hodnot pixelů v obraze vynásobených jádrem bude nejvyšší.

Výsledkem tohoto zjištění bylo vytvoření funkce, která v zadané části obrazu vyhledá oblast, která má nejvyšší hodnoty pixelů a vytvoří z ní konvoluční jádro. Toto programem vytvořené jádro se již může uplatnit nad obrazem a s vysokou pravděpodobností správně detekuje požadovanou oblast.

Daný předpoklad jsem si ověřila na sérii testovacích fotografií. Vybrala jsem na jedné fotografii malou oblast, o které jsem věděla, že je během toku času na snímcích neměnná. Uplatnila jsem nad ní funkci, pro výběr konvolučního jádra. Poté jsem si vzala jinou fotografii ze série časosběrných snímků, zvětšila jsem na ní vybranou oblast na každé straně přibližně o 50 pixelů a provedla jsem konvoluci.

Podobně jsem postupovala i pro více fotografií. Jen s tím rozdílem, že bylo vždy třeba si uložit souřadnice nalezených bodů. Získané body jsem následně pomyslně přiložila na sebe a ořezala jsem všechny fotografie podle nejmenší části. Tím jsem dosáhla toho, že fotografie pasují na sebe a nevytváří se již takový „efekt roztřesené ruky“. Naneštěstí jsou snímky nejen posunuté nahoru, dolu, doleva a doprava. Někdy jsou také mírně pootočené a tento nežádoucí efekt se mi během dosavadní práce na projektu nepodařilo eliminovat.

Zde bych ráda ukázala několik testovaných fotografií. První je vždy uživatelem vybraná oblast a na ní je červeně vyznačen bod, který funkce zvolila jako střed budoucího kovolučního jádra. Dále pak následují fotografie s rozšířenou oblastí prohledávání a vyznačeným bodem, který detekovala kovoluce.

Obr. 6. a 7. Uživatelem zvolená oblast s vyznačeným hledaným bodem a nalezené místo v dalším snímku

Na horních dvou snímcích můžete vidět ukázku správné detekce stejného místa na dvou různých fotografiích. Záměrně jsem vybrala dva komplikované výřezy fotografií, které nenásledují bezprostředně za sebou a liší se úrovní jasu. Druhou úspěšnou detekci je možné vidět na následující sérii třech fotografiích.

Obr. 8. Uživatelem zvolená oblast a hledaný bod

Obr. 9. a 10. Dva výřezy z různých fotografií se správně nalezeným místem v obraze

Ne všechny testy byly úspěšné. Níže uvádím příklady chybné detekce.

Obr. 11. a 12. Uživatelem zvolená oblast s vyznačeným hledaným místem a výběr z následujícího snímku s chybnou detekcí

Obr. 13. Uživatelem zvolená oblast

Obr. 14 a 15. První výřez je ze snímku, který byl pořízen bezprostředně po referenční fotografii. Čtenář může vidět, že bylo nalezeno přibližně shodné místo. Druhý snímek pochází z fotografie, která byla pořízena později a již bezprostředně na snímky nenavazuje. Můžete si všimnout, že místo bylo označeno naprosto chybně.

Během testování jsem přišla na to, že některé části fotografie pro tento způsob detekce shodných míst nejsou vhodné. Je to způsobeno několika důvody. Ve zvětšené prohledávané oblasti se mohou naskytnout místa, která mají celkově vyšší hodnoty pixelů, než měla oblast, ze které se vybíralo konvoluční jádro. Dalším důvodem mohou být měnící se celkové podmínky během pořizování časosběrných snímků. Oba tyto negativní jevy je možné pozorovat právě na výše uvedených výřezech z fotografií pořízených při západu slunce.

6.2.3 Tvorba videa

V jedné z předchozí kapitol jsem uváděla základní funkce knihovny OpenCV pro tvorbu videa. Při jejich vyhledávání jsem narazila na internetu na značné množství příkladů, které demonstrovali jejich činnost, a já předpokládala, že vytvoření AVI souboru nebude nic náročného. Při pokusu o spuštění jednoho takového přikladu se ale ukázalo, že je nefunkční.

Během prvních pokusů o vytvoření videa jsem používala soubor funkcí okolo funkce *cvCreateVideoWriter*. Funkce mi správně vytvořila soubor out.avi, ale tento soubor měl nulovou velikost. Důvody jsem hledala nejprve v rozměrech dodávaných snímků, ale tam problém nebyl. Další logickou volbou byla změna kodeku. Vyzkoušela jsem všechny codeky, které jsem v kapitole o funkcích OpenCV uvedla, zkoušela jsem měnit všechny parametry, které funkce má, ale opět se nedostavil kýžený výsledek. Soubor měl stále nulovou velikost.

Dalším krokem bylo vyzkoušení obdobných funkcí okolo *cvCreateAVIwriter*. Zde jsem narazila na problém neexistence struktury *CvAVIWriter*, ale tento problém jsem vyřešila jejím nahrazením strukturou *CvVideoWriter*. I když funkce fungovala, stále se mi tvořil jen prázdný video soubor. Poté jsem na internetovém fóru objevila pro mě do té doby neznámou volbu kódu codeku. Byla to 0 – jako volba pro video bez komprese. Vyzkoušela jsem ji a ihned se mi vytvořilo moje první video.

Jediný funkční a mnou ověřený postup pro tvorbu videa je tedy následující:

```
CvVideoWriter* writer;
writer = cvCreateAVIWriter( "out.avi", 0, fps, cvSize(frameW,frameH));
…
cvWriteToAVI(writer,img);
…
cvReleaseVideoWriter( &writer );
```

Vytvořené video bylo testováno v několika přehrávačích. Dokázali ho přehrát programy Media Player Clasik, BSplayer, Windows Media Player a Real Player. Jediný program, který měl s videem problémy byl VLC media player, kde během přehrávání byla jen černá obrazovka.

Přesto bych uživateli, který bude aplikaci používat, doporučila, aby k tvorbě videa přistupoval s rozvahou. Je třeba zvážit velkost dodaných fotografií ku parametru fps. Platí, že čím větší rozměry mají dodané fotografie, tím je obtížnější dané video správně přehrát. Přehrávač jednoduše nestíhá všechny snímky v určeném časovém intervalu správně zobrazit. Proto v takovém případě doporučuji buď snížit počet snímků za sekundu nebo zmenšit rozměry fotografií.

6.3 Shrnutí současného stavu

Aplikace se nachází ve stavu, kdy načtení fotografií a tvorba videa jsou již plně funkční, viz podkapitola 5.1.4. Nástroj pro úpravu jasu a kontrastu fotografií se dá také považovat za dokončený, viz podkapitola 5.1.1. Proces vyhledávání shodných míst v obraze byl úspěšně zvládnut, ale protože

není vždy spolehlivý, bylo by vhodné implementovat i alternativní variantu. Například s ručním vyznačením stejných míst na fotografiích, více viz podkapitola 5.1.2.

Vzhledem k tomu, že nebylo spolehlivě dokončeno napasování fotografií na sebe, nebylo možné se zatím věnovat i problematice eliminace nechtěných objektů., podkapitola 5.1.3.

Co se týče uživatelského rozhraní, jsou požadované funkce ovládání programu hotové, ale stále je co zlepšovat, více viz podkapitola 5.2.

Závěr

Cílem této bakalářské práce bylo vytvoření programu, který umožní pořizování kvalitního videa z časosběrných fotografií. Daný úkol byl splněn, i když by mohly být nějaké výhrady ke kvalitě videa, vzhledem k problémům s detekcí shodných míst na fotografiích pomocí kovoluce.

Práce na programu pro tvorbu časosběrného videa bude zcela jistě pokračovat i v budoucnosti. Bylo by vhodné se pokusit dokončit nástroj pro úpravu snímků. Automatická detekce by se stále mohla používat, ale třeba v kombinaci s ručním vyznačením stejných míst na fotografiích. Dále by se mohla implementovat i rotace snímků, aby napasování bylo dokonalé.

Je stále potřeba také implementovat funkci pro eliminaci cizích objektů na fotografii pomocí průměrování. U uživatelského rozhraní je také ještě mnoho možností, jak ho vylepšit. Například doplněním některých voleb jako třeba „Vytvoř projekt“ nebo „Smaž soubor z výběru“. Nástroj na úpravu jasu a kontrastu by se mohl rozšířit o automatickou korekci jasu u všech fotografií.

Toto jsou jen nejzákladnější možnosti, jak na programu pokračovat do budoucna. Pokud bude mít program úspěch, bylo by možné se ho pokusit rozšířit i pro jiné platformy než pro Windows.

Literatura

[1] ZEMČÍK, Pavel, ŠPANĚL, Michal. *Zpracování obrazu : Studijní opora*, 2006.

[2] ŠPANĚL, Michal. *Rozpoznávání gest ve video sekvencích*., 2003. Diplomová práce.

[3] *Barvy v počítači a v kartografii : Barevné modely* [online]. 2005 [cit. 2007-05-10]. Dostupný z WWW: <http://gis.zcu.cz/studium/pok/Materialy/Book/ar03s01.html>.

[4] DANĚK, Ondřej. *Registrace obrazu bunek podle intenzit v obraze*. [s.l.], 2007. 63 s. Diplomová práce.

[5] *Wikipedie, otevřená encyklopedie : Konvoluce* [online]. 6. 5. 2007 [cit. 2007-05-10]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Konvoluce>.

[6] *Wikipedie, otevřená encyklopedie : Digitální fotoaparát* [online]. 20. 4. 2007 [cit. 2007-05-10]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Digit%C3%A1ln%C3%AD_fotoapar%C3%A1t>.

[7] *Fotografovani.cz : Clona tajemství zbavená* [online]. 19.11.2004 [cit. 2007-05-10]. Dostupný z WWW: <http://www.fotografovani.cz/art/fotech_df/rom_aperture.html>.

[8] *Introduction to programming with OpenCV* [online]. January 27, 2006 [cit. 2007-05-11]. Dostupný z WWW: <http://www.cs.iit.edu/~agam/cs512/lect-notes/opencv-intro/opencv-intro.html>.

[9] *Leeds Guide to OpenCV : The IplImage Structure* [online]. uesday 12 December 2006 [cit. 2006-05-11]. Dostupný z WWW: <http://www.comp.leeds.ac.uk/vision/opencv/iplimage.html>.

[10] *Zpracování obrazových dat : Filtrace obrazu, plovoucí okno, vyhlazování obrazu* [online]. 13. 4. 2006 [cit. 2007-05-09]. Dostupný z WWW: <http://gama.fsv.cvut.cz/wiki/index.php/Zpracov%C3%A1n%C3%AD_obrazov%C3%BDch_dat_-_cvi%C4%8Den%C3%AD_%C4%8D.5>.

[11] ŽÁRA, Jiří, et al. *Moderní počítačová grafika*. [s.l.] : [s.n.], 2004. 609 s.

Seznam příloh

Příloha 1. CD