VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

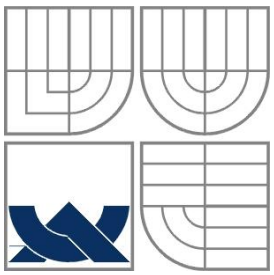
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

DETEKCE OBLOHY V OBRAZE A VIDEU

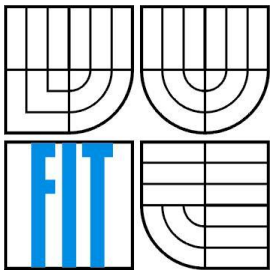
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE CYRIL VOJÁČEK
AUTHOR

BRNO 2012



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

DETEKCE OBLOHY V OBRAZE A VIDEU

SKY DETECTION IN IMAGES AND VIDEO

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

CYRIL VOJÁČEK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. BRONISLAV PŘIBYL

BRNO 2012

Abstrakt

Práce pojednává o problému detekce oblohy v obraze a videu. Je zde popsána vybraná metoda, použitá k vytvoření aplikace. Pozornost je také věnována jejímu možnému vylepšení. Výsledkem je aplikace, která je schopná detekovat více typů oblohy. Práce je zaměřena hlavně na detekci v obraze, ale je zde i možnost práce s videem. Správná funkce detekce je ověřena na testovací sadě a v práci jsou prezentovány výsledky testování.

Abstract

This work is about the sky detection in images and video. This paper describes the selected method of detection, which is used to create application. The focus is also on possible improvement the selected method. The result is an application, which is able to detect more types of sky. The work is mainly focused on the detection in image, but it is also possible to work with video. The correct detection is verified on the test set and the results of testing are presented in this paper.

Klíčová slova

detekce oblohy, Kuwaharův filtr, segmentace obrazu, CSC

Keywords

Sky detection, Kuwahara filter, Image segmentation, CSC

Citace

Vojáček Cyril: Detekce oblohy v obraze a videu, bakalářská práce, Brno, FIT VUT v Brně, 2012

Detekce oblohy v obraze a videu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Bronislava Přibyla. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Cyril Vojáček
16. května 2012

Poděkování

Rád bych na tomto místě poděkoval mému vedoucímu Ing. Bronislavu Přibylovi za vedení, trpělivost a cenné rady. Také bych rád poděkoval své rodině za podporu a zázemí.

Obsah

1 Úvod

V poslední době dochází k rychlému vývoji informačních technologií a vypadá to, že tento trend bude i nadále pokračovat. Tento jev má za následek, že stále více lidí používá prakticky denně nějaký produkt z oblasti informačních technologií. Tím roste také spektrum oblastí, ve kterých hrají informační technologie významnou roli. Jedním z problémů, kterému se informatici několik posledních let věnují, je počítačové vidění. Do této oblasti potom spadají i různé typy detekce. Jednou z nich se zabývá tato bakalářská práce, konkrétně jde o detekci oblohy.

Otázkou však je, zda je taková detekce vůbec k něčemu užitečná a pokud ano, v jakých případech se dá využít. Pokud bychom byli schopni provést kvalitní detekci oblohy, získáme tím mnoho cenných informací o obraze. Asi tou nejzákladnější věcí je správné rozlišení, zda jde o venkovní či vnitřní scénu. Existuje ale i další využití. Detekce oblohy může být součástí programů na analýzu obrazu, kde je snahou rozpoznat co nejvíce objektů. Pokud dokážeme identifikovat část obrazu jako oblohu, nemůže už jít o něco jiného. Narážím teď na případ, kdy nás zajímá pouze to, co se děje v popředí. Obloha jako taková je naproti tomu téměř vždy součástí pozadí. Složitější využití by mohlo být při snaze určit místo, které je na snímku zachyceno. Po úspěšné detekci totiž získáme hranici horizontu, která by mohla být vstupem nějakého rozpoznávacího programu.

Pro výše zmíněné možnosti je potom podstatná kvalita detekčních metod. Největší problémy jsou spojeny s typem oblohy. Většinou předpokládáme, že obloha je modrá. Není divu, už když se jako malí učíme rozpoznávat barvy, vhodným objektem pro modrou barvu je nebe. To však bohužel není pravidlo. Na obloze se velmi často vyskytují mraky, jindy je zase celá šedivá. Proto je potřeba, aby existovala nějaká metoda, která by dokázala zpracovat všechny druhy nebe.

Cílem této bakalářské práce je vytvořit takovou detekční metodu, která by byla schopna co nejlépe řešit problém, zmíněný v předchozím odstavci. Kromě toho by měla vzniklá aplikace umět správně detekovat všechny části oblohy a správně je rozlišovat od podobně barevných objektů. Základem vytvořené aplikace bude zpracování obrázků, její součástí včak bude i ukázka, jak by se implementovaná metoda dala využít při práci s videem.

Práce je rozdělena celkem do sedmi kapitol. Po stručném zasvěcení do řešené problematiky v této kapitole se čtenář dále seznámí se základy zpracování obrazu. Informace, které jsou zde popsány, jsou použity při vytváření aplikace.

Ve třetí kapitole jsou nastíněny různé přístupy k detekci oblohy. Velký prostor je zde věnován jedné konkrétní detekční metodě.

V další části práce jsou stanoveny cíle pro výslednou aplikaci a dochází k výběru detekční metody, která se využije při její tvorbě. Také je zde návrh na její vylepšení.

Pátá kapitola se věnuje samotné tvorbě aplikace. Je zde popsán průběh její činnosti, ukázáno její ovládání a zmíněno její omezení.

Následující kapitola popisuje testování vytvořené aplikace. Dochází k vysvětlení způsobu testování a jsou zde prezentovány dosažené výsledky.

V závěrečné kapitole jsou potom shrnuty získané poznatky a nastíněny možnosti pokračování práce v budoucnu.

2 Zpracování obrazu

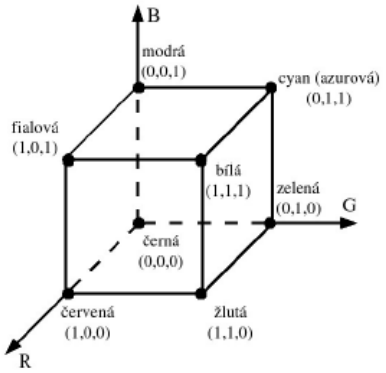
Úvod technické zprávy je věnován teoretickým základům o zpracování obrazu. Účelem této kapitoly není nahradit učebnice zabývající se počítačovou grafikou, ale pouze informovat čtenáře o technikách, které jsou relevantní vzhledem k zaměření celé bakalářské práce. Jde o stručný popis, který usnadní pochopení jednotlivých dílčích kroků při vytváření výsledné aplikace. První podkapitola se zabývá barevnými modely, konkrétně RGB a HSV barevným prostorem. Dále je vysvětlen princip Kuwaharova filtru a závěr kapitoly je věnován CSC segmentaci, která tvoří důležitou součást detekce oblohy.

2.1 Reprezentace obrazu a barevné modely

Abychom mohli s obrazem reálného světa pracovat a dále používat, je nutné jej převést do podoby, která je srozumitelná pro počítačové systémy. Tento proces se nazývá digitalizace. Jde o to, že obraz reálného světa je spojitý a má teoreticky nekonečný počet barev. V takové podobě by ale s obrazem nešlo pracovat, ukládat jej atd. Proces digitalizace spočívá v převodu reálného obrazu popsaného spojitou funkcí *f(x,y)*, kde *x* a *y* jsou souřadnice v prostoru, na obraz popsaný diskrétní funkcí $I_{i,j}$. Nově vzniklý digitální obraz je potom zobrazen v matici *I* x *J*. Platí, že čím jsou rozměry matice větší, tím věrněji je zachován původní spojitý obraz reálného světa. Barvu výsledného digitalizovaného obrazu lze popsat pomocí vhodného barevného modelu. V našem případě nás zajímají barevné modely RGB a HSV. Informace týkající se celé této podkapitoly jsou čerpány z [1] a [2].

2.1.1 RGB model

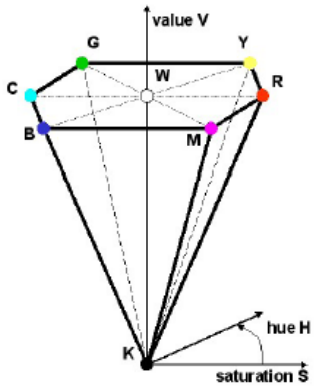
Různé barvy, které se používají při vytváření obrazu, jsou tvořeny kombinací několika základních barev z barevného spektra. V případě RGB modelu jde o složení tří složek – červené (R, red), zelené (G, green) a modré (B, blue). Každá složka pak nabývá hodnot ‹0, 1›. V počítačové grafice se pak často setkáváme s celočíselným rozsahem, kdy je každá složka určena v rozsahu 0 – 255. Výsledná barva je vyjádřena třemi byty, kde každý byte zastupuje právě jednu zmíněnou barevnou složku. Hodnota 0 znamená, že složka není zastoupena, maximální hodnota indikuje, že složka nabývá své největší intenzity. Počet barevných odstínů, které tímto modelem můžeme zobrazit je 256^3, což se rovná téměř 17 milionům barevných odstínů. Takový počet je pro věrné zachycení reálného obrazu dostačující. Na obrázku 2.1 je RGB model prostorově znázorněn jako jednotková krychle.



Obrázek 2.1: Prostorové zobrazení RGB modelu

2.1.2 HSV model

Barevný prostor HSV je definován trojicí složek, které v tomto případě nepředstavují základní barvy, jako tomu bylo u RGB modelu. Třemi hlavními parametry potom jsou barevný tón (H, hue), sytost (S, saturation) a jas (V, value). Barevný tón označuje převládající spektrální barvu, sytost určuje příměs jiných barev a jas je dán množstvím bílého (bezbarvého) světla. Pro prostorové vyjádření HSV modelu se v tomto případě používá šestiboký jehlan – obrázek 2.2. Složky S a V jsou vyjádřeny od 0 do 1, zatímco složka H, která reprezentuje úhel, nabývá hodnot z intervalu ‹0°, 360°›. Vrchol jehlanu představuje černou barvu, střed podstavy barvu bílou. Na plášti jehlanu se nachází dominantní barvy (se sytostí jedna). Čisté barvy jsou na obvodu podstavy.



Obrázek 2.2: Prostorové zobrazení HSV modelu

2.1.3 Převod RGB do HSV

V některých případech může nastat situace, kdy chceme pracovat v prostoru HSV, ale známe pouze hodnoty RGB prostoru. Proto se následující část věnuje převodu z RGB do HSV.

Jednotlivé složky RGB prostoru jsou v tomto případě reprezentovány reálnými čísly v intervalu od 0 do 1. Stejně tak potom složky S a V z barevného prostoru HSV. Složka H, která představuje úhel, nabývá hodnot mezi 0° a 360°. Jednotlivé složky HSV jsou potom vypočítány následovně:

$$h = \begin{cases} nedefinováno, & jestliže\ max = min \\ 60° \times \dfrac{g-b}{max-min} + 0°, & jestliže\ max = r \text{ a } g \geq b \\ 60° \times \dfrac{g-b}{max-min} + 360°, & jestliže\ max = r \text{ a } g < b \\ 60° \times \dfrac{b-r}{max-min} + 120°, & jestliže\ max = g \\ 60° \times \dfrac{r-g}{max-min} + 240°, & jestliže\ max = b \end{cases} \tag{2.1}$$

$$s = \begin{cases} 0, & jestliže\ max = 0 \\ 1 - \dfrac{min}{max}, & jestliže\ max > 0 \end{cases} \tag{2.2}$$

$$v = max, \tag{2.3}$$

kde písmena *r*,*g* a *b* představují jednotlivé složky RGB prostoru, obdobně písmena *h*,*s* a *v* u prostoru HSV. *Max* a *min* slouží pro označení složek, které mají maximální a minimální hodnotu v RGB prostoru.

2.2 Filtrování obrazu

Obrazové filtry se většinou používají, pokud potřebujeme obraz nějakým způsobem jemně upravit. Jsou filtry, které slouží k potlačení nežádoucího šumu v obraze, což jsou shluky pixelů, které narušují čistotu obrazu. V této podkapitole se zaměříme pouze na jeden typ filtru, který poslouží k předzpracování obrazu před samotnou detekcí oblohy.

2.2.1 Kuwaharův filtr

Kuwaharův filtr se používá v případech, kdy chceme v obraze vyhladit hrany. Jde o filtr, který pracuje s lokálním okolím pixelu. Toto okolí rozdělí na čtyři stejně velké části, kde centrální pixel náleží všem čtyřem oblastem. Výsledná hodnota centrálního pixelu je potom průměrná hodnota té oblasti, která má nejmenší rozptyl hodnot. V článku [3] je popsán základní Kuwaharův filtr, společně s jeho vylepšením.

Mějme šedotónový obraz *I(x,y)* a čtverec o délce 2*a*, který tvoří okolí bodu *(x,y).* Tento čtverec rozdělme na čtyři identické regiony Q_1, Q_2, Q_3 a Q_4:

$$\begin{cases} Q_1(x,y) = [x, x+a] \times [y, y+a] \\ Q_2(x,y) = [x-a, x] \times [y, y+a] \\ Q_3(x,y) = [x-a, x] \times [y-a, y] \\ Q_4(x,y) = [x, x+a] \times [y-a, y] \end{cases} \tag{2.4}$$

,kde × značí skalární součin.

Dále mějme $m_i(x, y)$ a $s_i(x, y)$, kde m_i je průměrná hodnota a s_i směrodatná odchylka počítaná pro každý region $Q_i(x, y)$, $i = 1..4$. Pro každý bod *(x, y)* je dán výstup Kuwaharova filtru $\Phi(x, y)$, který je určen jako lokální průměrná hodnota $m_i(x, y)$, která odpovídá *i*-tému regionu, který má nejmenší lokální směrodatnou odchylku $s_i(x, y)$. To můžeme zapsat jako:

$$\Phi(x,y) = \sum_i m_i(x,y) f_i(x,y) \tag{2.5}$$

,kde

$$f_i(x,y) = \begin{cases} 1, & jestliže\ s_i(x,y) = \min\limits_k \{s_k(x,y)\} \\ 0, & jinak \end{cases} \tag{2.6}$$

Na obrázku 2.3 jsou znázorněny různé výstupy Kuwaharova filtru na základě velikosti zvoleného okolí.

Obrázek 1.3: Vlevo originální obrázek, uprostřed výstup Kuwaharova filtru s okolím 5 x 5, vpravo výstup Kuwaharova filtru s okolím 7 x 7

2.3 Segmentace obrazu

Segmentace obrazu je jednou z velmi důležitých technik při analýze a zpracování obrazu. Cílem segmentace je rozdělit daný obraz na oblasti s podobnými vlastnostmi. Metoda segmentace obrazu má široké využití. Její výsledky se například využívají při zpracování lékařských obrazových dat nebo při klasifikaci objektů v obraze.

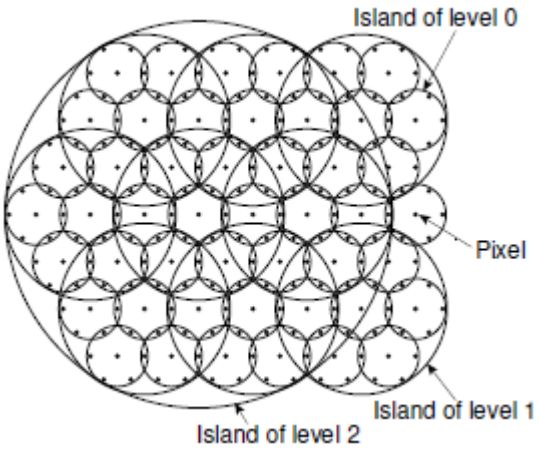
Existuje několik různých metod pro segmentaci barevného obrazu. Mnoho z nich vzniklo díky transformaci jejich jednodušších variant pro šedotónový obraz. V této podkapitole se zaměříme na popis pouze jedné metody, která je důležitá pro popisovanou detekci oblohy.

2.3.1 CSC segmentace

Název CSC segmentace vznikl jako zkratka pro anglické pojmenování *color structure code*. Jde o metodu, která rozděluje obraz na oblasti na základě barevné podobnosti. Využívá pro to speciální datovou strukturu, která je popsána níže. Pomocí těchto struktur potom vznikají jednotlivé segmenty, které se postupně rozrůstají. Vytváření segmentů probíhá paralelně a proto je metoda nezávislá na zvoleném startovním bodě. Díky tomu jde o rychle pracující metodu, která je také dostatečně robustní. Vysvětlení principu této metody se věnuje [4] a [5].

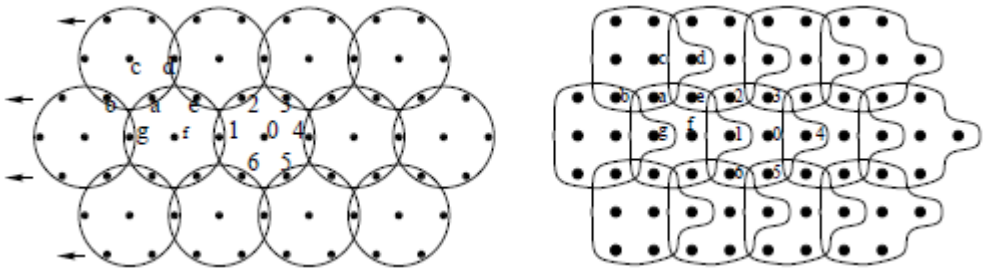
2.3.1.1 Hexagonální struktura

Metoda CSC segmentace je založena na speciální hexagonální struktuře. Tato struktura se nazývá ostrov (*island*). Celý obraz je rozdělen na ostrovy různých úrovní. Každý ostrov obsahuje sedm prvků, jeden centrální a šest jeho sousedů. Na úrovni 0 jsou prvky ostrovu samotné pixely. Sedm ostrovů úrovně 0 vytváří ostrov úrovně 1. Podle stejného principu dochází k vytváření vyšších úrovní a přestává se v momentě, kdy jeden ostrov pokrývá celý obraz. Důležitou vlastností této datové struktury je fakt, že se navzájem překrývají. Každé dva sousední ostrovy na úrovni N mají společný jeden a právě jeden subostrov (ostrov na úrovni $N - 1$). V případě dvou sousedů úrovně 0 jde o samotný pixel. Na obrázku 2.4 lze vidět, jak popisovaná hexagonální struktura vypadá a jak dochází k vytváření ostrovů na vyšších úrovních.



Obrázek 2.4: Hexagonální struktura

Jediným větším problémem při práci se zmíněnou hexagonální strukturou je to, že jednotlivé pixely jsou v obraze umístěny do matice. Zmíněná struktura je potom díky svému tvaru spíše logickou strukturou. Abychom s ní ale mohli pracovat, je nutné si ji nějak převést do skutečné matice pixelů. To se dá jednoduše udělat například tak, že se každý druhý řádek posune o půl pozice doprava či doleva. Touto jednoduchou metodou můžeme naše zobrazení pomoci hexagonálních struktur převést do maticového tvaru. Toto posunutí je znázorněno na obrázku 2.5.

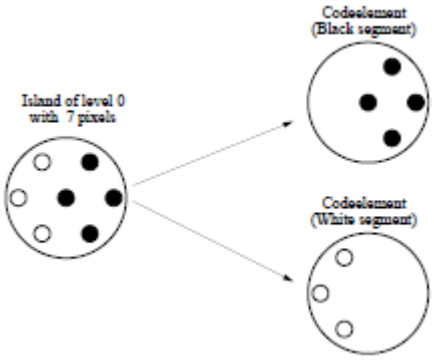


Obrázek 2.5: Vlevo hexagonální struktury, vpravo výsledek pro převedení do maticového zobrazení

2.3.1.2 Průběh segmentace

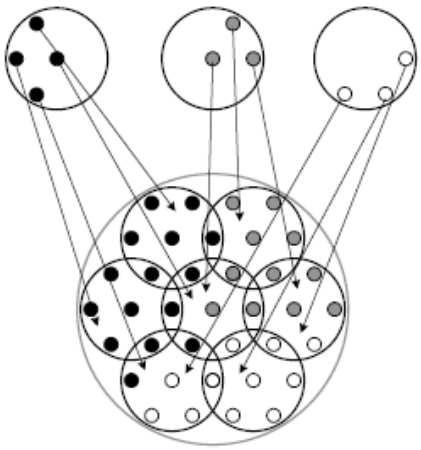
Samotný průběh segmentace se dá popsat třemi fázemi a jejím cílem je rozdělit obraz na segmenty, které mají společnou vlastnost - barvu. Tyto vytvářené segmenty jsou reprezentovány tzv. elementy (*code-elements*). Při jejich vzniku pokrývají pouze několik málo pixelů. V průběhu segmentace se tyto elementy spojují dohromady, až ve výsledku tvoří celý barevný segment.

Jako první u CSC segmentace přichází na řadu **inicializační fáze.** V této fázi se pracuje pouze s ostrovy na úrovni 0 a dochází k inicializaci výše zmíněných elementů. To se děje tak, že se porovnávají pixely v rámci ostrovu. Pokud se zjistí, že jsou dva sousední pixely barevně podobné, sloučí se dohromady a vznikne element. Barevnou podobností je myšleno to, že rozdíl jejich barevných hodnot je menší, než zvolená prahová hodnota. Na obrázku 2.6 lze vidět vznik dvou elementů v rámci jednoho ostrovu. Výhodou je, že vytváření elementů v jednom ostrovu je nezávislé na tvorbě v ostatních ostrovech. Díky tomu není nutné volit určitý startovní bod. Výsledkem této prvotní fáze je potom velké množství vytvořených elementů, kde každý obsahuje pouze několik málo pixelů. V další fázi segmentace se tyto elementy hierarchicky rozrůstají.



Obrázek 2.5: Vznik dvou elementů

Po úvodní inicializaci následuje nejdůležitější část segmentace, a to tzv. **spojovací fáze**. Během ní dochází k postupnému rozrůstání vytvořených elementů. Při vytváření těchto elementů v předchozí fázi bylo využito ostrovů na úrovni 0. Nyní dojde k posunu o úroveň nahoru a propojování elementů v rámci ostrovů na úrovni 1. Pokud jsou dva sousední elementy barevně podobné a mají navíc nějakou společnou část, dojde k jejich sjednocení. Společnou částí je na této konkrétní úrovni 1 myšleno to, že oba elementy sdílejí společný pixel, jak je to znázorněno na obrázku 2.6. Tato operace se provede pro všechny ostrovy v rámci všech úrovní, počínaje úrovní 1. Naposledy se spojování provede na nejvyšší úrovni, kdy jeden ostrov pokrývá celý obraz. Tímto postupným spojováním elementů dochází ke vzniku tzv. stromu elementů. Vždy, když vznikne nový element spojením elementů nižších úrovní, jsou spolu s ním uloženy i ukazatele na tyto subelementy. Pokud nastane na nějaké úrovni situace, kdy element nemá nikoho k dalšímu spojování, vytvoří tento element kořen svého stromu. Každý takto vytvořený strom představuje jeden barevný segment. Výsledná barva segmentu je průměrnou barvou všech jeho pixelů. Vytváření segmentů takovým způsobem je efektivní a to právě díky hexagonální struktuře a vzájemnému překrývání těchto struktur.



Obrázek 2.6: Vytvoření tří nových elementů v rámci ostrovu na úrovni 1

Při spojování elementů může dojít k nežádoucím jevům, kdy dojde ke spojení elementů, které nejsou barevně podobné. To může nastat kvůli lokálnímu zpracování v rámci ostrovu. Teď stačí, aby dva elementy měly společný subelement a dojde k jejich spojení. Předpokládá se, že pokud je daný subelement barevně podobný oběma svým elementům, budou pak barevně podobné i oba elementy. To ovšem nemusí platit a z tohoto důvodu je zde tzv. **rozdělovací fáze.** Během ní dochází ke kontrole spojovaných elementů na každé spojovací úrovni. Pokud je barevný rozdíl mezi elementy větší než

zvolená prahová hodnota, ke spojení elementů nedojde i přesto, že mají společný subelement. Následně je velmi důležité správně rozdělit společný subelement mezi oba elementy.

Obrázek 2.7: Ukázka CSC segmentace, vlevo originální obraz, vpravo výsledek CSC segmentace

3 Metody detekce oblohy

Obloha patří mezi útvary, které jsou velmi často k vidění na fotografiích. A právě kvůli jejímu častému výskytu je vhodné ji umět v obraze nějakým způsobem detekovat. Správná detekce oblohy pak poskytuje určité informace o obraze. Například lze rozlišit, zda se jedná o vnitřní či venkovní scénu. Určit, co je v obraze obloha a co ne, není jednoduchá záležitost. Obloha totiž může být velmi rozmanitá. Nejjednodušší situace je u čistě modré oblohy, ale často se člověk setkává i s deštivou oblohou, kdy je celé nebe pokryté mraky. Kromě této variability jsou zde i další problémy. Obloha může být překryta jinými objekty a tím rozdělena na více částí. Dobře fungující metoda by měla být schopna tyto části rozpoznat a správně je určit jako oblohu.

Existuje několik různých postupů při detekci oblohy, které se navzájem liší tím, jaké konkrétní vlastnosti oblohy považují za nejdůležitější. Roli také hraje, zda má metoda sloužit pouze pro detekci v obraze nebo má být použita při zpracování videa. V druhém případě pak může být snaha o zvládnutí detekce v reálném čase.

V této kapitole nejprve popíšu několik různých přístupů k detekci oblohy a poté detailně vysvětlím, jak pracuje jedna konkrétní metoda.

3.1 Různé přístupy k detekci oblohy

Jako první se zde budu věnovat přístupu k detekci oblohy, který je představen v [6]. Snahou autorů bylo vytvořit rychlou metodu detekce oblohy, která by se dala použít na zpracování videa a tím pádem dokázala pracovat v reálném čase. Z tohoto důvodu se autoři rozhodli pracovat na úrovni jednotlivých pixelů a to právě proto, že rozdělení obrazu na segmenty se společnými vlastnostmi je poměrně časově náročné. Výstupem po provedené detekci je potom tzv. pravděpodobnostní mapa, která pro každý pixel určuje pravděpodobnost, že daný pixel znázorňuje oblohu.

Jak už samotný název článku napovídá, metoda je zaměřena převážně na zpracování čistě modré oblohy. Je důležité si uvědomit, že i v tomto případě má obloha určité specifické vlastnosti. Na první pohled se může zdát, že je obloha jednobarevná, ve skutečnosti však zaujímá široký barevný rozsah. Na horním okraji obrazu je barva oblohy sytější a více tmavá, naopak čím více se přibližujeme k horizontu, tím barva na své sytosti ztrácí a stává se světlejší. Toho si jsou autoři metody vědomi a ve své práci na tuto skutečnost mysleli. Detekci rozdělují na tři samostatné fáze.

V první fázi detekce dochází k vytvoření tzv. inicializační pravděpodobnosti oblohy. Ta se počítá pro každý pixel na základě barvy, vertikální pozice v obraze a textury. Analýza textury v sobě obsahuje vertikální a horizontální gradient. Tato prvotní pravděpodobnost je pevně dána a využívá se v dalších fázích detekce. Během druhé fáze jsou pomocí vypočítané pravděpodobnosti nalezeny oblasti v obraze, kde je největší pravděpodobnost výskytu oblohy. Podle toho, kde se tyto oblasti nachází, je určena vertikální pravděpodobnost výskytu oblohy v obraze. Na základě doposud zjištěných informací je vytvořen 2D prostorový model, který pro každou oblast v obraze předepisuje očekávanou barvu oblohy. V poslední fázi potom na základě vytvořeného modelu a informacím ze vstupního snímku dochází k vypočítání skutečné pravděpodobnosti oblohy pro každý pixel.

Dalším možný přístup k detekci oblohy je popsán v [7]. Zde se nepracuje na úrovni pixelů, jako tomu bylo v předchozím případě, ale obraz je rozdělen na segmenty. Na výstupu detekce je potom každý segment ohodnocen pravděpodobností s jakou náleží obloze. Autoři průběh detekce rozdělují na tři hlavní části.

V první části dochází k barevné klasifikaci pomocí neuronových sítí. Čistě na základě barvy vzniká pravděpodobnostní mapa, podobně jako tomu bylo u předchozí metody. Tato mapa slouží jako vstup do druhé části, kdy dochází k vytváření regionů. Vzniklý region obsahuje ty pixely, které mají podobnou hodnotu pravděpodobnosti. V poslední části detekce se pro každý vzniklý region vypočítá vertikální a horizontální gradient a na základě dosažených výsledků dochází k vytvoření konečné pravděpodobnosti pro každý region.

Nevýhodou u této metody je to, že často může dojít k chybné detekci oblastí, které mají podobnou barvu jako obloha. Mezi takové objekty může patřit oblečení nebo části vodní hladiny. Navíc malé regiony, které představují oblohu, jako například oblasti mezi větvemi stromů, nejsou do výstupu detektoru zahrnuty. Důvodem je právě jejich velikost, pro kterou je obtížné vypočítat gradient.

Na metodu popsanou v předchozím odstavci navazují v [8] a snaží se řešit problém s oblastmi oblohy, které ovšem nebyly jako obloha označeny. První fází této metody je tedy vytvoření jednotlivých regionů a zjištění jejich pravděpodobností, přesně tak, jak je to popsáno v předchozí metodě. Následně se z těchto regionů vybere jeden hlavní region, pomocí kterého dojde k vytvoření polynomického modelu oblohy. Výběr hlavního regionu je velmi jednoduchý. Pro všechny vytvořené regiony, které mají nenulovou pravděpodobnost, se vypočítá hodnota na základě velikosti regionu a jeho pravděpodobnosti. Ten region, který má tuto vypočítanou hodnotu nejvyšší, se stává hlavním regionem. Všechny ostatní regiony, které mají nenulovou pravděpodobnost a nejsou dosud považovány za oblohu, tvoří tzv. kandidáty.

Pro každou barevnou složku hlavního regionu je vytvořen polynomický model. Pomocí těchto modelů je pro všechny barevné složky určených kandidátů vypočítána tzv. chyba *E*. Pokud je chyba nízká pro všechny barevné složky kandidáta, je tento region prohlášen za součást oblohy.

3.2 Detekce využívající CSC segmentaci

Nevýhodou většiny detekčních metod je to, že pracují pouze s čistě modrou oblohou. Jak ale víme, obloha je málokdy čistě modrá a proto by bylo vhodné, aby si detekce uměla poradit s různými typy oblohy. A právě tímto problémem se zabývá metoda popisovaná v [9].

Aby mohla detekce správně fungovat, je nutné splnit jednu zásadní podmínku. Pokud se v obraze nachází obloha, musí se dotýkat horního okraje snímku. Metoda pracuje na jednoduchém principu. Nejprve dochází k rozdělení obrazu na regiony, které obsahují barevně podobné pixely. Následně se tyto regiony testují, zda náleží obloze a samotné testování začíná právě u regionů, které leží na horním okraji snímku.

Kromě vertikální pozice regionů je také velmi důležitá jejich barva. Jelikož má metoda správně fungovat na více typech oblohy, je potřeba zvolit široký barevný rozsah. Na čistě modré obloze se můžou vyskytovat bílé mraky nebo v případě deště je celá obloha zahalena šedivými, někdy až tmavými mraky. V obraze se ale mimo oblohy nachází plno šedivých či bílých objektů. Proto je pro správné rozlišení oblohy nutné využít další vlastnosti, jako je třeba tvar regionů. Samotná detekce by se dala rozdělit na tři hlaví části.

3.2.1 Předzpracování obrazu

Každý vstupní snímek je potřeba před samotnou detekcí určitým způsobem upravit. Jako první se na obraz aplikuje Kuwaharův filtr s okolím 3x3. Díky tomuto filtru dojde k vyhlazení hran, které usnadní další zpracování. Následně dochází k rozdělení obrazu na jednotlivé barevné regiony.

K tomuto účelu se využije CSC segmentace a vzniklé regiony se nazývají segmenty. Technik pro segmentaci obrazu existuje několik, avšak pro popisovanou metodu detekce oblohy je potřeba použít právě CSC segmentaci. Kdyby se měl obraz rozdělit na segmenty například pomocí techniky *split-and-merge*, hranice mezi jednotlivými regiony by byly převážně tvořeny rovnými čárami. Ovšem z důvodu, že pro rozlišení oblohy je využito i tvaru segmentů, je tato technika nevhodná. U CSC segmentace jsou hranice mezi vzniklými segmenty většinou nepravidelné. Avšak hranice mezi oblohou a například budovami zůstávají i nadále rovné a díky tomu je můžeme od oblohy rozlišit i přesto, že jejich barva může být šedivá čili podobná deštivé obloze.

3.2.2 Zjištění informací o segmentech

Nyní máme vytvořené segmenty, které pokrývají celý obraz. V této fázi o nich zjistíme určité informace, které se později využijí k samotné detekci oblohy.

Mějme segment *S*, který se skládá z barevně podobných pixelů. Horní hranici segmentu $b_u(S)$ tvoří všechny souřadnice (x, y), kde pixel (x, y) leží uvnitř *S* a pixel (x, i) leží mimo *S*, pro všechna *i*, kdy platí $i < y$.

Barva segmentu *S* je dána jako barevný průměr všech pixelů, které tvoří segment *S*.

Průměrný vertikální gradient (*mean vertical gradient*) je vytvářen pro společnou hranici mezi segmentem *S* a jeho horním sousedem. Pro každý pixel z $b_u(S)$ je zjištěn rozdíl v jasu mezi ním a jeho horním sousedem. Pokud pixel nemá shora žádného souseda, bere je jako rozdíl hodnota 0. Výsledný vertikální gradient je pak dán jako průměrná hodnota těchto rozdílů.

Průměrná hraniční druhá derivace (*mean bounded second derivative*) je počítána pro $b_u(S)$ a informuje o tom, zda je horní hranice segmentu *S* tvořena rovnou čárou nebo má spíše nepravidelný tvar. Počítá se následovně. Mějme x_{min} a x_{max} jako x-ové souřadnice nejlevějšího a nejpravějšího pixelu v *S*. Pro každé $i \in \mathrm{N}$ z $[x_{min}, x_{max}]$ existuje právě jeden pixel z $b_u(S)$, jehož x-ová souřadnice je *i*. Poté můžeme $b_u(S)$ považovat za funkci *x*, kde $b_u(S, x) := y$ jestliže $(x, y) \in b_u(S)$. Funkce:

$$\delta^2(S, x) \coloneqq b_u(S, x-1) - 2 * b_u(S, x) + b_u(S, x+1) \tag{3.1}$$

je potom druhou derivací $b_u(S)$ na pozici *x*.

$|\delta^2 = (S, x)|$ dosahuje nízkých hodnot v případě, že je hranice mezi segmenty rovná, naopak vysokých pokud se jedná o nepravidelnou hranici. Nyní vypočítáme aritmetický průměr z $|\delta^2 = (S, x)|$ pro celou šíři segmentu a tím dostaneme hodnotu vyjadřující jeho rovnost. Problém nastává u krajních bodů hranice $b_u(S)$, kde může dojít k velkému výkyvu hodnot a tím ke zkreslení potřebné informace o segmentu. Z tohoto důvodu se pro hraniční hodnoty využívá následující:

$$m(S, x) = \begin{cases} 1, & jestliže\ |\delta^2(S, x)| \geq 1 \\ 0, & jestliže\ |\delta^2(S, x)| < 1 \end{cases} \tag{3.2}$$

Hodnotu průměrné hraniční druhé derivace pro horní okraj segmentu potom vypočítáme:

$$\frac{\sum_{(x,y) \in b_u(S)} m(S, x)}{\sum_{(x,y) \in b_u(S)} 1} \tag{3.3}$$

3.2.3 Analýza barvy a detekce oblohy

Určit segmenty oblohy čistě na základě jejich barvy je prakticky nemožné. V obraze se velmi často vyskytují objekty, které mají podobnou barvu jako obloha, ale ve skutečnosti nejsou její součástí. Přesto je barva jednou z velmi důležitých dílčích informací o obloze. Protože se u této zmiňované metody detekce využívá široký barevný rozsah, který by dokázal pokrýt rozmanitost oblohy, je obtížné vhodně určit podmínky, kdy je možné segment na základě barvy označit jako případnou oblohu. Autoři metody se pro analýzu barvy rozhodli použít barevný prostor HSV, ve kterém se dají lépe definovat intervaly, kdy barva odpovídá obloze. Proto je potřeba převést barvu vyjádřenu v prostoru RGB na jednotlivé odpovídající složky prostoru HSV a pro každý segment určit průměrnou barvu. Díky analýze několika snímku dospěli k několika podmínkám:

- *saturation* (sytost) < 13 a *value* (jas) > 216
- *saturation* < 25 a *value* > 204 a *hue* (barevný tón) > 190 a *hue* < 250
- *saturation* < 128 a *value* > 153 a *hue* > 200 a *hue* < 230
- *value* > 88 a *hue* > 210 a *hue* < 220

Pokud segment splní alespoň jednu z těchto podmínek, můžeme ho označit za barevně odpovídající obloze.

Teď, když jsou známy všechny potřebné informace o segmentech, může dojít k samotné detekci oblohy. Mějme dva seznamy segmentů L_s a L_c, které jsou na počátku prázdné. V seznamu L_s se budou ukládat segmenty, které jsou oblohou, v L_c potom tzv. kandidáti na oblohu. Všechny segmenty, které barevně odpovídají obloze a současně pro ně platí podmínka, že se minimálně polovina jejich horní hranice dotýká horního okraje snímku, jsou považovány za oblohu a vloženy do seznamu L_s. Každý segment, který se shora dotýká alespoň jednoho segmentu v L_s, je potom považován za kandidáta a vložen do L_c. Pro každý segment S ze seznamu L_c jsou kontrolovány tyto tři kritéria:

1) segment S barevně odpovídá obloze

2) minimálně dvě třetiny horní hranice segmentu S se dotýkají buď horního okraje snímku a nebo segmentů v seznamu L_s

3) segment S splňuje alespoň jednu z následujících podmínek:

    - velikost segmentu S je menší než 500 pixelů (segment je příliš malý pro podrobnější analýzu)
    - průměrný vertikální gradient segmentu S je menší než 25
    - průměrná hraniční derivace horní hranice segmentu $b_u(S)$ je větší než 0,3.

Pokud segment S splňuje všechny tři uvedená kritéria, je považován za oblohu a přemístěn do seznamu L_s. Následně dojde k aktualizaci seznamu kandidátu L_c. Pokud segment nesplní všechny uvedená kritéria, je ze seznamu kandidátů odstraněn. Tento postup se stále opakuje až do doby, kdy seznam kandidátu L_c zůstane prázdný.

Tímto způsobem dochází k postupnému „nalepování“ segmentů na sebe a vzniká tak jedna velká plocha, která představuje oblohu. Je vidět, že je zde zcela klíčová podmínka, aby se obloha

dotýkala horního okraje snímku. Pokud by to tak nebylo, nebyly by na začátku žádné segmenty označeny jako obloha a tím pádem by ani žádná v obraze nevznikla. V momentě, kdy se zkontrolují všechny segmenty pod vznikající oblohou a žádný z nich nesplňuje výše uvedené podmínky, obloha se dále nerozrůstá a detekce končí. Vzniklou oblast oblohy tvoří jedna velká propojená plocha. Tady lze vidět největší nedostatek této detekční metody a to sice ten, že v případě, kdy je obloha rozdělena nějakým objektem na více částí, tyto části nejsou klasifikovány jako obloha.

4 Analýza a návrh

Před tím, než se člověk pustí do vytváření aplikace, je velmi důležité si co nejpřesněji stanovit, co od dané aplikace očekáváme. Na základě stanovených cílů je potřeba vhodně vybrat metodu, která se pro aplikaci použije. Tato fáze je velmi důležitá. Kdyby došlo ke špatnému výběru, aplikace nemusí splňovat stanovené požadavky. V této kapitole se zaměřím na stanovení cílů práce a na jejich základě potom k výběru vhodné detekční metody. Jsou zde zváženy všechny klady a zápory a navrhnuto řešení, jak minimalizovat nežádoucí chování. V závěru kapitoly je potom nastíněno, jak se dá zvolená detekční metoda aplikovat na detekci oblohy ve videu.

4.1 Stanovení cílů a výběr metody

Cílem mé bakalářské práce je vytvořit kvalitní aplikaci pro detekci oblohy v obraze, kterou by bylo možné použít i na detekci ve videu. Když se řekne obloha, většina lidí si představí modrou barvu. V tom mají pravdu, obloha je skutečně modrá. Bohužel ale zdaleka ne ve všech případech. Například během zimy, kdy převažuje spíše bíla či šedivá barva, je velmi obtížně setkat se s čistě modrou oblohou. V našich podmínkách navíc poměrně často dochází k dešťovým srážkám, kdy je celé nebe pod mrakem. Proto mi přijde nedostačující zaměřit se pouze na modrou oblohu. Moje aplikace by měla být schopna označit všechny výše popsané druhy oblohy jako nebe. Kromě toho bych po mé aplikaci chtěl, aby v obraze dokázala správně rozlišit i drobné oblasti oblohy. Pokud se například podíváme na fotografii z městské oblasti, uvidíme na ní, že je obloha rozdělena na více částí o různé velikosti. To může být způsobeno elektrickým vedením nebo také korunami stromů. Při plnění těchto dvou podmínek může dojít k chybné detekci. Za oblohu jsou považovány třeba části budov, které mají šedivou barvu a problém mohou způsobovat i odrazy oblohy od vodní hladiny a podobně. Proto je nutné zvolit kompromis tak, aby se minimalizovala chybná detekce a přitom došlo k pokrytí co největší části oblohy. Jak je i z názvu práce patrné, měla by být aplikace použitelná i pro detekci ve videu. Z mých předchozích požadavků je však patrné, že kladu důraz spíše na kvalitu detekce, než na její rychlost. Z tohoto důvodu je pro mě prvořadá detekce oblohy v obraze a videu se věnuji spíše okrajově. Jde hlavně o to, ukázat možný způsob práce s videem.

Metod pro detekci oblohy existuje několik, bohužel velká část z nich se věnuje převážně detekci čistě modré oblohy. Narazil jsem však na jednu, která splňuje mou podmínku o rozmanitosti oblohy. Tato metoda je popsána v kapitole 3.2. Rozhodl jsem se tedy vytvořit mou aplikaci na základě této metody. Bohužel je zde jedna hlavní nevýhoda, která je v rozporu s mou druhou podmínkou, týkající se uzavřených částí oblohy. I přesto je tato metoda pro mne zajímavá a návrh, jak tuto její nevýhodu eliminovat, popíšu v následující podkapitole.

4.2 Návrh na vylepšení metody

Princip metody je podrobně popsán v kapitole 3.2. Já teď pouze stručně zmíním, co se vlastně při detekci děje. Obraz je nejprve rozdělen na segmenty, kde každý segment obsahuje pixely, které jsou barevně velmi podobné. Tyto segmenty jsou potom zkoumány, zda vyhovují podmínkám pro to, aby mohly tvořit oblohu. Jako první se testují segmenty, které tvoří horní okraj snímku a postupně se jde ve snímku stále níž. Obloha tak vzniká „přilepováním“ dalších segmentů k již vzniklé obloze. Proces detekce končí v době, kdy již žádný ze segmentů ležících bezprostředně pod oblohou nesplňuje dané podmínky. Zde je právě problém s oddělenými částmi oblohy. Stačí pouze malý objekt mezi touto

částí a vzniklou oblohou, jako třeba elektrické vedení, a oddělená část ani nedostane příležitost k ověření podmínek, zda je oblohou.

Pokud chci vytvořit aplikaci, která za oblohu označí pokud možno všechny její části, je potřeba popsanou detekční metodu nějakým způsobem rozšířit. Po ukončení předchozí detekce proto navrhuji dále zkoumat všechny ty segmenty, které barevně odpovídají obloze. U těchto segmentů však nemůžu využít dříve zjištěných vlastností. Protože jsou od zbytku oblohy oddělené cizím předmětem, nelze se spoléhat na charakteristiku jejich horní hranice. Ta je velmi často rovná a rozdíl v jasu s jejím horním sousedem poměrně vysoký, čímž by došlo k vyloučení tohoto segmentu z oblohy. Pro jeho analýzu jsem se rozhodl využít barevné podobnosti s již vytvořenou oblohou. K tomu použiju barevný prostor RGB.

Nejprve si zjistím průměrné hodnoty všech složek barevného prostoru z celé oblasti reprezentující oblohu. Následně zjistím barevný rozdíl mezi zkoumaným segmentem a těmito průměrnými hodnotami:

$$rozdíl = (P_R - S_R)^2 + (P_G - S_G)^2 + (P_B - S_B)^2 \tag{4.1}$$

, kde P odpovídá barevným složkám průměru oblohy a S barevným složkám zkoumaného segmentu. Pokud je vypočítaný rozdíl menší než zvolená hodnota prahu, lze zkoumaný segment označit za oblohu. Pokud by byl práh nastaven na vysokou hodnotu, mohlo by docházet k chybným rozhodnutím. Proto je lepší zvolit spíše nižší práh. To by ovšem nepokrylo všechny oddělené části oblohy. Z tohoto důvodu jsem se rozhodl pro další kritérium a tím je rozdíl v modré složce modelu RGB mezi vzniklou oblohou a zkoumaným segmentem. Prahová hodnota pro tento rozdíl v modré složce je nastavena podle celkového barevného rozdílu. Čím větší je barevný rozdíl, tím je zvolena menší hodnota prahu. Aby byl segment prohlášen za oblohu, musí potom splnit jednu ze dvou podmínek. Buďto je velmi podobný vzniklé obloze a jeho barevný rozdíl je malý a nebo může být podobnost o něco menší, současně ale musí splňovat blízkost v modré barvě.

Všechny segmenty, které odpovídají zmíněným podmínkám, lze úspěšně označit jako barevně odpovídající vzniklé obloze. Všechny tyto segmenty však nemusí být skutečnou oblohou. V obraze je totiž mnoho různých objektů, které mají podobnou barvu jako obloha na snímku. Většinou se však nalézají pod horizontem. Proto jsem se rozhodl, že výše popsaným způsobem se budou testovat pouze ty segmenty, které alespoň částečně leží nad nejnižším bodem oblohy. Tím se například eliminuje odraz oblohy od vodní hladiny.

Pokud byl ve snímku nalezen nějaký segment, který byl označen za oblohu, jsou jeho dolní sousedi označení za kandidáty a provede se s nimi detekce, která je popisována v metodě, ze které vycházím. To se děje z důvodu, kdy je oddělená část oblohy velká a obsahuje například mraky. Tyto mraky nebudou přímo označeny jako obloha, protože se barevně mohou od vzniklé oblohy lišit, avšak jsou její součástí. Po každé takto provedené detekci je potom potřeba zkontrolovat, zda nedošlo k posunu spodní hranice oblohy směrem dolů. Pokud ano, opětovně dochází k testování segmentů, které je popsáno v této kapitole. Detekce oblohy je ukončena v momentě, kdy nedojde k žádnému rozšíření oblohy směrem dolů.

4.3 Detekce oblohy ve videu

Jak již bylo naznačeno, hlavní důraz v mé práci kladu na co nejlepší výsledek detekce v obraze. Z tohoto důvodu je práce s videem prezentována spíše jako ukázka možného použití vybrané metody při detekci oblohy ve videu. Nevýhodou metody, kterou jsem si vybral pro vytvoření aplikace, je to,

že je poměrně dost časově náročná a to především kvůli segmentaci obrazu. Proto aplikace neslouží pro detekci oblohy v reálném čase, ale pracuje s již vytvořenými videosoubory.

Každé video je sekvencí po sobě jdoucích snímků. Mou představou je postupně tyto snímky brát a pracovat s nimi stejně, jako by šlo o obraz. Na snímek se tedy aplikuje výše popsaná metoda detekce a výsledné video vznikne opětovným složením výsledných snímků do sebe. Jde o velmi jednoduchý způsobem, který má ovšem poměrně slušné výsledky. Aby práce s videem nebyla tak zdlouhavá, rozhodl jsem se detekovat oblohu pouze na každém třetím snímku vstupní videosekvence. Dojde tak k ušetření času a výsledné video vypadá lépe. Nevýhodou u snímků videa je to, že mají většinou menší rozlišení než fotografie a nejsou tedy tak kvalitní. Kvůli tomu se občas chybně detekují určité oblasti. Abych co nejvíce tento jev potlačil, rozhodl jsem se lehce rozšířit podmínky, které označují daný segment za barevně odpovídající obloze.

5 Implementace aplikace

Následující kapitola se věnuje popisu vytvořené aplikace pro detekci oblohy. Byla vytvářena v kombinaci jazyků C/C++ a hlavní myšlenkou bylo to, aby co nejlépe splnila požadavky uvedené na začátku předchozí kapitoly. Její hlavní zaměření se týká detekce oblohy v obraze, ale je schopná rozpoznat oblohu i ve videu, ikdyž v tomto případě jde spíše o ukázku, jak by mohla detekce vypadat. Jde o jednoduchou konzolovou aplikaci, jejímž vstupem je buďto obrázek ve formátu .jpg nebo video ve formátu .avi. Výstupem je obraz či video, které znázorňuje detekovanou oblohu. V této kapitole zmíním použité externí soubory a zaměřím se na popis implementace vybrané detekční metody.

5.1 Použité knihovny a externí soubory

Při vývoji aplikace byla velmi užitečná jedna knihovna, která mi pomáhala při práci s obrazem. Touto knihovnou je **OpenCV**. Jedná se o svobodnou a otevřenou multiplatformní knihovnu pro manipulaci s obrazem, která je především zaměřena na počítačové vidění a zpracování obrazu. Poskytla mi řadu funkcí, které našly při implementaci detekční metody vhodné uplatnění. Pro použití vyhotovené aplikace je nutné, mít tuto knihovnu nainstalovanou v počítači.

Další věcí, o které je třeba se zmínit, jsou externí soubory použité pro CSC segmentaci. Konkrétně se jedná o zdrojové kódy, které jsou dostupné v [10]. Jde o program, který rozdělí vstupní snímek na jednotlivé segmenty. Na výstupu je potom obraz reprezentující barevné regiony. V mojí aplikaci jsem použil pouze část zdrojových kódů, které pro mě rozdělí obraz. Tyto kódy jsem lehce upravil tak, aby mi naplnily mou datovou strukturu pro zaznamenání vzniklých segmentů. Od mých vlastních zdrojových kódů se liší svou předponou (ad, ip).

5.2 Implementace

Jak bylo zmíněno dříve, jedná se o konzolovou aplikaci, kterou jde spustit s několika parametry. Jednotlivé možnosti jsou uvedeny v následující podkapitole. Jako první po spuštění aplikace dojde k načtení všech souborů v zadaném adresáři. Tyto soubory jsou postupně procházeny a v případě, že narazím na obraz v požadovaném formátu, spustím detekční metodu. Pro lepší přehled je na standardní výstup vypsáno jméno zpracovávaného souboru.

Pro uchování potřebných údajů pro průběh detekce jsem si vytvořil dvě datové struktury. Ty jsou definovány v *model.h*. První z nich slouží k uchování informací o pixelech zpracovávaného snímku. U každého pixelu v obraze mě zajímá jeho barva, jak v RGB prostoru, tak i v HSV prostoru. Druhá struktura zaznamenává informace týkající se segmentů, na které bude obraz rozdělen.

Po předání snímku ke zpracování se nejprve aplikuje kuwaharův filtr. Před samotným filtrováním je vytvořeno pole pro uchování informací o pixelech a uložena barva každého pixelu. Následně se vytváří nový snímek, který je výstupem Kuwaharova filtru.

Dalším potřebným krokem je CSC segmentace. Ta je zprostředkována pomocí externích zdrojových kódů a jako vstup slouží nově vytvořený snímek po filtraci. Po segmentaci opět vznikne nový snímek, ovšem ten je vytvořen pouze jako ukázka segmentace a není podstatný pro samotnou detekci. Všechny potřebné informace jsou ukládány do datové struktury reprezentující vytvořené segmenty. Pro uchování všech segmentů je použit datový typ vector.

Nyní jsou známy všechny segmenty a následuje několik výpočtů pro zjištění dalších informací. Prvním krokem je převod RGB složek na HSV v rámci každého pixelu. Následuje výpočet průměrných hodnot obou barevných prostorů pro každý segment. To poslouží k analýze barvy v rámci detekce. Pro ni je ale potřeba zjistit ještě další informace jako střední vertikální gradient a střední hraniční derivaci. To se provede nyní podle instrukcí uvedených v popisu metody.

Dalším krokem je analýza barvy. Každý segment je testován na řadu podmínek a pokud splní alespoň jednu z nich, je prohlášen za barevně odpovídající obloze. Pro tuto informaci je u segmentu použitý datový typ boolean, kde kladná hodnota značí, že by segment mohl být oblohou. Před zahájením samotné detekce je udělána ještě jedna věc. Pro každý segment jsou poznačeni jeho sousedi, shora a zdola.

V tuto chvíli přichází na řadu samotná detekce. Ze všeho nejdřív je u segmentů, které svou barvou odpovídají obloze, zkoumáno, zda-li se dotýkají horního okraje snímku. Pokud je tento dotyk dostatečně veliký, jsou tyto segmenty přesunuty do vectoru pro oblohu. Všichni jejich sousedi jsou považováni za kandidáty a i pro ně je vymezen vector pro jejich uložení.

Následuje smyčka, která testuje segmenty z kandidátů a to tak dlouho, dokud nějaký kandidát existuje. U segmentu se zjišťuje, jestli splňuje podmínky pro oblohu. Pokud ano, dojde k jeho přesunu a vector s kandidáty je aktualizován o jeho sousedy.

V další části je nutné ošetřit oddělené části oblohy. Zjistím průměrné barevné hodnoty RGB pro již vzniklou oblohu a pro každý segment, který není oblohou, spočítám jejich barevný rozdíl a nastavím hodnotu prahu pro modrou složku. Pokud je pak segment dostatečně podobný vzniklé obloze a alespoň částečně leží nad horizontem, je sám prohlášen za oblohu. Jeho sousedi jsou opět vloženi mezi kandidáty. V momentě, kdy se prošly kontrolou všechny segmenty, je nutné provést detekci nad novými kandidáty a to způsobem popsaným v předchozím odstavci. Pokud vlivem nových segmentů oblohy došlo k posunutí její vertikální hranice směrem dolů, celý proces se opakuje a to až do té doby, kdy k žádnému posunu nedojde.

Nyní již vím, který segment je obloha a který ne, ovšem tuto informaci je ještě potřeba převést do grafické podoby. Pro tento účel se vytvoří šedotónový snímek, ve kterém je bílou barvou znázorněna obloha. Šedou barvou jsou vykresleny ty segmenty, které sice barevně odpovídají obloze, ovšem nevyhovují ostatním podmínkám detekce.

Výstupem po provedené detekci jsou tři obrazy. Na jednom jsou vidět změny po aplikaci Kuwaharova filtru, na dalším pak výstup CSC segmentace. Třetí obraz potom zobrazuje detekovanou oblohu.

Výše popsaný postup se používá pro detekci v obraze. Pokud chci detekovat oblohu ve videu, musím nejprve toto video načíst a detekci spustit na každý jeho třetí snímek. Pro ten už je postup stejný jako u obrazu. Odlišnost je pouze v podmínkách, kdy je segment označen jako barevně odpovídající obloze. Také nedochází k ukládání výsledku filtrování a segmentace. Jednotlivé výstupy detekce jsou pak skládány v novou videosekvenci. Před začátkem detekce jsou na standardní výstup vypsány informace o vstupním videu, při samotném průběhu detekce se vypisuje její stav.

5.3 Ovládání aplikace

Aplikace byla vytvářena pod operačním systémem Windows 7 Professional. Díky použité knihovně OpenCV by ovšem měla být platformově nezávislá. Byla vytvořena jako konzolová aplikace, se kterou uživatel komunikuje prostřednictvím příkazového řádku. Pro přeložení zdrojových kódů je vytvořen dávkový soubor *preklad.bat*. Vznikne tak spustitelný soubor *BP.exe*. V následující části práce se pokusím uvést možné způsoby ovládání aplikace.

Základní spuštění aplikace je velmi jednoduché a provádí se příkazem:

```
> BP.exe [-t],
```

kde `BP.exe` je přeložená spustitelná aplikace. V tomto případě se očekává existence složky *images*, ve které jsou soubory pro provedení detekce. Tato složka se musí nacházet ve stejném adresáři jako spustitelná aplikace. Uvnitř složky *images* jsou potom další složky, kam se ukládají jednotlivé výstupy aplikace. Při tomto způsobu spuštění budou zpracovány pouze obrazy ve formátu .jpg. Nepovinný parametr `-t` se použije v případě, kdy je vyžadováno provedení testování při běhu aplikace.

Pokud chce uživatel místo obrazu provádět detekci u videosouborů, je potřeba spustit aplikaci tímto způsobem:

```
> BP.exe -v,
```

kde uvedený parametr signalizuje práci s videem. Jako v předchozím případě se očekává existence zmíněných složek. Při práci s videem nelze použít parametr `-t`, protože testování se provádí pouze pro detekci oblohy v obraze.

Pro ověření kvality detekce lze využít:

```
> BP.exe -test.
```

V tomto případě se spustí testování pro již vytvořené výstupy detekce. Jakým způsobem se aplikace testuje, je popsáno v následující kapitole. Tento parametr lze použít opět pouze pro práci o obrázky a není možné jej kombinovat s jinými parametry.

Dosud se předpokládalo, že soubory ke zpracování jsou obsaženy ve složce, která se nachází ve stejném adresáři jako spustitelná aplikace. Pokud by uživatel chtěl pro vstup nebo výstup použít jiné složky, může tak udělat pomocí příkazu:

```
> BP.exe [-load adresář] [-save adresář].
```

Za parametrem `-load` musí následovat cesta k adresáři, ze kterého se mají čerpat soubory k detekci. Druhý volitelný parametr `-save` zase odkazuje na adresář, kam se budou ukládat výsledky detekce.

Poslední možnost uživatel využije v případě, kdy chce provést detekci pouze jednoho souboru:

```
> BP.exe -file soubor.
```

V tomto případě může jít jak o obrázek, tak o videosoubor a není nutné k rozlišení používat parametr `-v`. Lze kombinovat s ostatními parametry.

5.4 Omezení aplikace

Nevýhodou aplikace je to, že je časově náročná, což se projeví hlavně u obrázků s vysokým rozlišením. Nejvíce času zabere fáze předzpracování snímku, ve které se aplikuje Kuwaharův filtr a probíhá CSC segmentace.

Kromě tohoto problému je aplikace také paměťově náročná, což se negativně projeví při zpracování velkého množství dat. Z těchto důvodů je umožněno hromadné zpracování snímků pouze pro obrázky, které obsahují maximálně milion a půl pixelů. Pokud by chtěl uživatel detekovat oblohu ve větším obraze, musí k tomu použít parametr `-file` a provést detekci samostatného souboru. Hromadné zpracování je navíc omezeno i do počtu. Při zpracování obrázku je tak povoleno pouze maximálně 100 souborů.

Další omezení se týká práce s videem. Jak bylo výše několikrát zmíněno, mým hlavním cílem bakalářské práce bylo vytvoření kvalitního detektoru oblohy v obrázcích. Detekce z videosekvence je zde spíše jako ukázka možného využití zvolené metody pro tento účel. Navíc zpracování i krátkého videa zabere velké množství času. Proto jsem se rozhodl omezit maximální délku videa a to konkrétně na jednu minutu.

6 Testování aplikace

V této kapitole se zaměřím na ověření kvality vytvořené aplikace pro detekci oblohy. Testování je zaměřeno na detekci z obrazu, protože vyhodnocení úspěšnosti z videa je problematické. Na úvod kapitoly nejprve popíšu vytvořenou testovací sadu a poté se budu věnovat samotnému způsobu testování. Vyhodnocení provedu mimo samotné aplikace také pro detekci podle původní metody, kterou jsem se inspiroval. V závěru potom výsledky testování vyhodnotím a zjistím, zda mé vylepšení skutečně pomohlo kvalitě detekce.

6.1 Testovací sada

Jak již bylo zmíněno při analýze vybrané detekční metody, kromě barvy segmentu jsou důležitá i jiná kritéria. Jedním z nich je kontrola společné hranice mezi segmenty. Čím je hranice rovnější, tím klesá pravděpodobnost, že spodní segment bude oblohou. Z toho lze usoudit, že zvolená metoda bude dosahovat lepších výsledků v obrazech, kde přechod mezi oblohou a zbytkem snímku tvoří převážně objekty s rovnými hranami. Tato skutečnost měla vliv při vybírání vhodné testovací sady.

Zcela vyhovující pro mě byla databáze obrázků dostupná na [11], která obsahuje více než tři tisíce snímků o rozměrech 1280 x 960, na kterých jsou zachyceny ulice města Boston. Z této sady je vybrán určitý vzorek tak, aby na něm byly pokud možno všechny druhy oblohy.

Protože jsem po mé aplikaci vyžadoval, aby byla schopná detekovat i drobné části oblohy, je nutné provést co nejpřesnější testování. Z tohoto důvodu jsem se rozhodl testovat na úrovni pixelů. K tomuto účelu je pro každý snímek v testovací sadě vytvořena binární maska, kde bílá barva představuje skutečnou oblohu. Binární masky jsem si vytvářel sám, kdy jsem pomocí grafického editoru ručně označil, co oblohou je a co není. Tato činnost byla velmi časově náročná, což se podepsalo na velikosti testovací sady. Ta obsahuje sto snímků, ve kterých je splněna podmínka, že se obloha dotýká horního okraje. Jsou zde k dispozici obrázky s čistě modrou oblohou, šedivou či bílou oblohou a taky s modrou, ve které jsou bílé mraky.

6.2 Způsob a výsledky testování

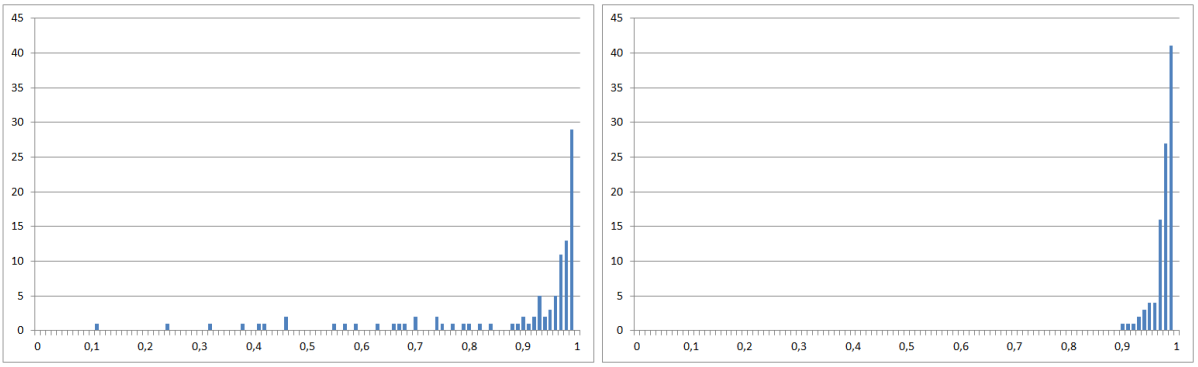
Pro ověření kvality vytvořené aplikace jsem se rozhodl porovnávat výstupy detekce s vytvořenými maskami obrázků z testovací sady. Porovnávání probíhá pixel po pixelu, kdy se zkoumají vždy dva pixely na stejné pozici v masce a ve výstupu aplikace. Dosažené výsledky jsou potom prezentovány dvěma způsoby. Samotné testování lze provést v rámci aplikace, ale je potřeba mít vytvořeny testovací masky. Výstup testování se ukládá do složky se spustitelnou aplikací do textového souboru `hodnoceni.txt`. Zde jsou uvedeny dosažené výsledky pro každý zpracovávaný obrázek. V následující části porovnám výsledky testování pro původní verzi detekční metody spolu s mou vytvořenou aplikací, která vylepšuje detekci oddělených částí oblohy.

Jako první jsem se rozhodl prezentovat výsledky aplikace stejně, jako je tomu u článku, ze kterého jsem vycházel při vytváření mé aplikace [9]. Na úvod je potřeba si definovat dvě proměnné. První z nich je pojmenována *ground truth (GT)* a informuje o tom, kolik pixelů tvoří skutečnou oblohu. Tento počet jde lehké spočítat. Jde o počet bílých pixelů v binární masce pro daný snímek. Druhou proměnnou je *S*, kde je uchován počet pixelů detekované oblohy. Pro vyjádření kvality detekční metody jsou použita dvě měřítka.

Prvním je měřítko úspěšnosti (*coverability rate, CR*), které udává, kolik skutečné oblohy bylo detekováno zvolenou metodou. Vypočítá se následujícím způsobem:

$$CR(S, GT) \coloneqq \frac{|S \cap GT|}{|GT|} \tag{6.1}$$

Na obrázku 6.1 lze vidět dosažené výsledky zobrazené jako dva grafy, pro původní verzi metody a pro mnou vytvořenou aplikaci. Na y-ové ose je vynesen počet snímků, které odpovídají danému měřítku na x-ové ose.

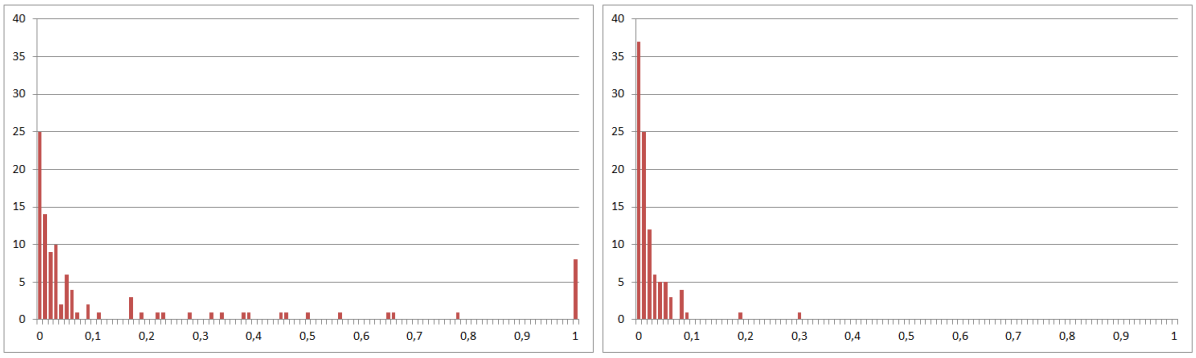


Obrázek 6.1: Grafy zobrazující měřítko úspěšnosti, vlevo původní metoda, vpravo výsledná aplikace

Druhým měřítkem se vyjadřuje, jak velká část detekované oblohy ve skutečnosti oblohou není. Jde o měřítko chyby (*error rate, ER*) a zjišťuje se:

$$ER(S, GT) \coloneqq \frac{|S - GT|}{|S|} \tag{6.2}$$

Na obrázku 6.2 jsou opět k dispozici grafy, ve kterých je zobrazeno měřítko chyby pro obě verze detekční metody.



Obrázek 6.2: Grafy zobrazující měřítko chyby, vlevo původní metoda, vpravo výsledná aplikace

Ze všech uvedených grafů jde vidět, že vybraná detekční metoda dosahuje vysoké měřítko úspěšnosti a nízké měřítko chyb. Ještě lépe je na tom výsledná aplikace, kde dochází navíc k detekci uzavřených částí oblohy. Všech sto snímků dosahuje měřítka úspěšnosti nad hodnotu 0,9. Naproti tomu u původní metody jde pouze o 73%. Vytvořená aplikace je na tom lépe, i co se týče měřítka chyb. Zde je celých

98% pod hodnotou 0,1. To, že je tomu u původní metody jinak, není žádným překvapením. Pokud dojde k úspěšné detekci pouze části oblohy, musí automaticky dojít k vyšším hodnotám chybovosti. Konkrétní srovnání na několika obrázcích je možné sledovat v příloze.

Druhé vyjádření výsledků testování provedu pomocí tzv. matice záměn (*confusion matrix*). Jde o jednoduchou matici 2 x 2, která slouží k lepší představě o výsledcích testování. V jednotlivých polích jsou zaznamenávány:

- **True positives** – vyjadřuje, kolik pixelů detekované oblohy skutečně odpovídá obloze.
- **False positives** – zde jsou započítány ty pixely, které byly detekovány jako obloha, ale ve skutečnosti oblohu nepředstavují.
- **True negatives** – ty pixely, u kterých bylo správně rozpoznáno, že nenáleží obloze.
- **False negatives** – sem patří pixely, které ve skutečnosti tvoří oblohu, ale jako obloha detekovány nebyly.

Opět jsem se rozhodl pro vyobrazení výsledků pro obě verze detekční metody, stejně jako v předchozím případě. Vytvořené matice reprezentují výsledky celé testovací sady. Konkrétní počty pixelů, jsem se rozhodl nahradit procentuelním vyjádřením, protože celkový jejich počet z celé testovací sady by byl příliš velký a znesnadnilo by to porozumění výsledků. Vytvořené matice pro testovací sadu jsou na obrázku 6.3.

true positives 85,11% skutečné oblohy	**false negatives** 14,89% skutečné oblohy
false positives 0,13% skutečné neoblohy	**true negatives** 99.87% skutečné neoblohy

true positives 98,17% skutečné oblohy	**false negatives** 1,83% skutečné oblohy
false positives 0,20% skutečné neoblohy	**true negatives** 99.80% skutečné neoblohy

Obrázek 6.3: Matice záměn, vlevo původní metoda, vpravo výsledná aplikace

Pokud mezi sebou porovnáme obě matice, zjistíme, že největší rozdíl je v prvním řádku. Ten je zaměřen na porovnání detekované oblohy se skutečnou. Oproti výše použitému způsobu testování, je až zde patrné, v čem je skutečná výhoda mého rozšíření.

6.3 Vyhodnocení testování

Vytvořená aplikace byla testována na sadě sta obrázků, které byly vybrány tak, aby zachycovaly různé typy oblohy. Byly provedeny dvě série testů. Nejprve se otestovala funkčnost metody, ze které se vycházelo při tvorbě této bakalářské práce. Druhé testování se pak týkalo mnou vytvořené aplikace, která zvolenou metodu rozšiřuje o detekci oddělených částí oblohy.

Výsledky testování byly prezentovány dvěma způsoby. Prvním bylo použití dvou měřítek, stejně jako tomu bylo v odborném článku, v němž byla představena použitá detekční metoda. U obou těchto měřítek na to byla lépe moje úprava, ovšem není z nich zcela zřejmé, v čem je její hlavní výhoda. K tomu napomohla až matice záměn, kde jsou výsledky lépe prezentovány. Jde vidět, že v případě správného rozlišení oblastí neobsahující oblohu jsou oba způsoby stejně kvalitní. Rozdíl

nastává až při detekci skutečné oblohy. To je dáno tím, že původní metoda neřeší oddělené části oblohy. To má někdy nežádoucí vedlejší účinky. V případě, kdy má segment pouze malou společnou hranici s oblohou, je vyloučeno, aby se stal oblohou. Přesto může oblohou být. Tento problém umí vyřešit moje vytvořená metoda a to přesto, že se nejedná o oddělenou část oblohy jako takovou. V příloze lze potom sledovat srovnání obou metod v různých situacích.

7 Závěr

Cílem této bakalářské práce bylo prostudovat možnosti pro detekci oblohy a podle vybrané metody vytvořit funkční aplikaci. V textu jsou zmíněny možné přístupy k detekci. Podle požadavků na aplikaci, které byly stanoveny na začátku celé práce, pak došlo k výběru vhodné detekční metody. Dále byl vytvořen a implementován návrh na odstranění jejích hlavních nedostatků

Výsledkem práce je aplikace, která je schopná detekovat více druhů oblohy, než jen čistě modrou. Její hlavní zaměření je na detekci oblohy v obraze, ovšem je tu i možnost práce s videem, ikdyž se jedná spíše o ukázku použití vybrané detekční metody pro tento účel. Funkčnost aplikace byla ověřena na vytvořené testovací sadě a výsledky byly prezentovány dvěma způsoby. Z nich lze vidět, že návrh na vylepšení vybrané metody měl své opodstatnění a detekce oblohy je díky tomu kvalitnější.

Na vytvořené aplikaci je možné dále pracovat. Největší pozornost by si zřejmě zasloužila detekce oblohy z videosouboru. Stávající stav není příliš vyhovující a proto je tu značný prostor pro jeho vylepšení. Ideální by bylo pro zpracování videa použít úplně jiný přístup, než který je popsán zde, protože implementovaná metoda je časově náročná a proto ne příliš vhodná pro práci s videem. Další práce by se také mohla týkat detekce dalších zajímavých oblastí v obraze. Následně by mohla vzniknout komplexní aplikace, pro analýzu obrazu.

Literatura

[1] Žára, J.; Beneš, B.; Sochor, J.; aj.: *Moderní počítačová grafika*. Computer Press, druhé vydání, 2005, iSBN 80-251-0454-0.

[2] Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2012-05-13]. Dostupné z WWW: http://cs.wikipedia.org/wiki/HSV

[3] G. Papari, N. Petkov, P. Campisi. Artistic edge and corner enhancing smoothing. *IEEE Transactions on Images Processing*, 16(10): 2449-2462, 2007.

[4] Paulus, D. Dorkó, G. Ahlrichs, U.: Color segmentation for scene exploration [online]. Dostupný z WWW: http://lear.inrialpes.fr/pubs/2000/DPA00/dorko_vfb2000.pdf

[5] Priese, L. Sturm, P.: Introduction to the Color Structure Code and its Implementation [online]. Dostupný z WWW: http://www.uni-koblenz-landau.de/koblenz/fb4/institute/icv/agpriese/research/ColorImgSeg/download/csc.pdf

[6] ZAFARIFAR, B.; DE WITH, P. H. N.: Blue Sky Detection for Picture Quality Enhancement. *In proceedings of Advanced Concepts on Intelligent Vision Systems* 2006, s. 522-532.

[7] J. Luo, S.P. Etz: A physical model-based approach to detecting sky in photographic images, *IEEE Transactions on Image Processing*, 2002, pp. 201-212.

[8] A. C. Gallagher, J. Luo, W. Hao: Improved blue sky detection using polynomial model fit, *IEEE International conference on image processing*, říjen 2004, pp. 2367-2370.

[9] Schmitt, F. Priese, L.: Sky detection in CSC-segmented color images [online]. Dostupný z WWW: http://www.uni-koblenz.de/~lb/publications/Schmitt2009SDI.pdf

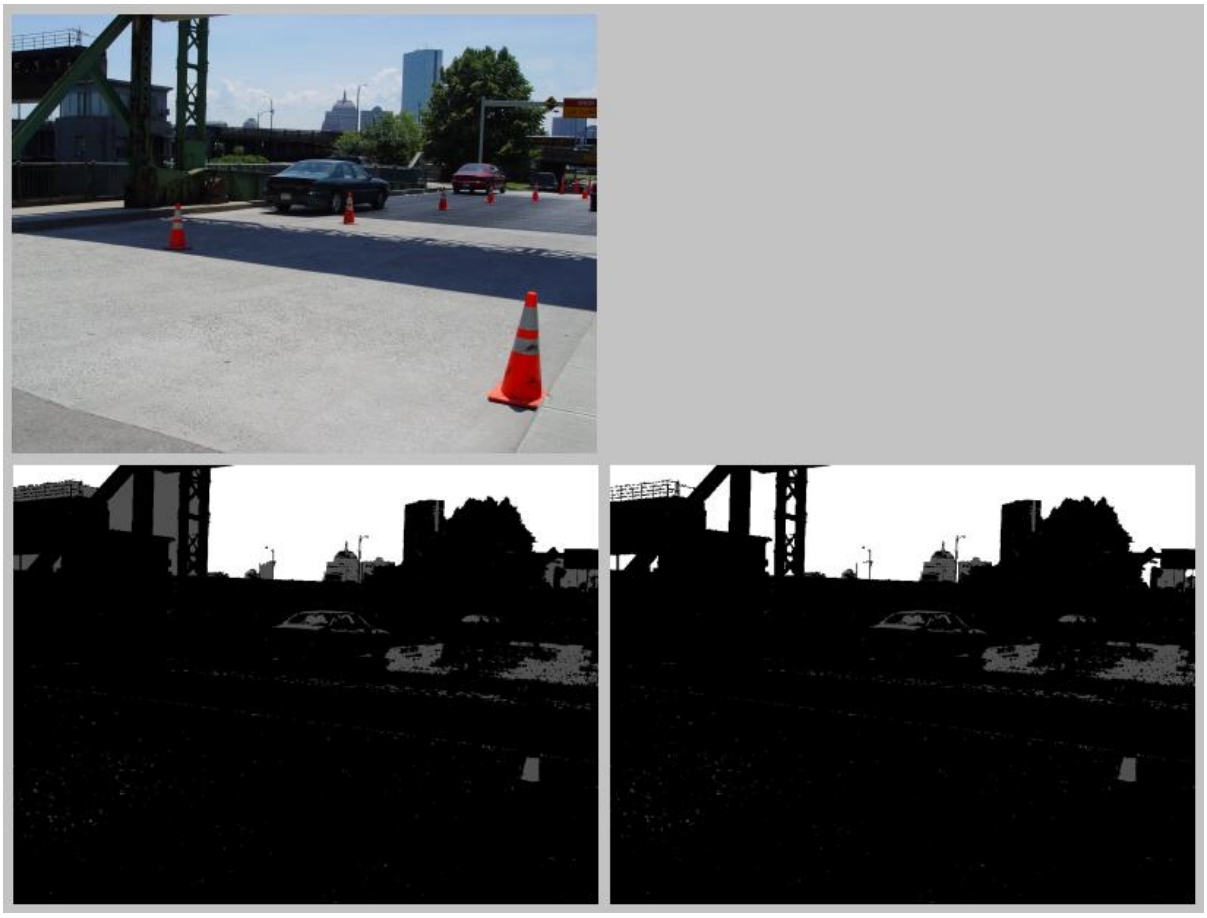
[10] Color Structure Code (CSC) [online]. [cit. 2012-05-13]. Dostupné z WWW: http://www.uni-koblenz.de/~lb/publications/Schmitt2009SDI.pdf

[11] CBCL StreetScenes Challenge Framework [online]. [cit. 2012-05-13]. Dostupné z WWW: http://cbcl.mit.edu/software-datasets/streetscenes/

Seznam příloh

Příloha 1. Výstupy aplikace

Obrázek 1: Vlevo originální snímek, uprostřed původní metoda, vpravo výsledná aplikace



Obrázek 2: Nahoře originální snímek, vlevo dole původní metoda, vpravo dole výsledná aplikace

Obrázek 3: Nahoře originální snímky, uprostřed původní metoda, dole výsledná aplikace. Vlevo je zobrazena modrá obloha s bílými mraky, vpravo je obloha celá šedivá

Příloha 2. DVD

Obsah přiloženého DVD:

- Zdrojové soubory
- Knihovny potřebné pro překlad a běh aplikace
- Spustitelná aplikace pro Windows
- Testovací sada
- Ukázky činnosti aplikace
- Manuál k ovládání aplikace
- Text technické zprávy ve formátu .pdf a .docx