Na tomto místě bude oficiální zadání vaší práce

- Toto zadání je podepsané děkanem a vedoucím katedry,
- musíte si ho vyzvednout na studijním oddělení Katedry počítačů na Karlově náměstí,
- v jedné odevzdané práci bude originál tohoto zadání (originál zůstává po obhajobě na katedře),
- ve druhé bude na stejném místě neověřená kopie tohoto dokumentu (tato se vám vrátí po obhajobě).

České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačové grafiky a interakce

Bakalářská práce

Kontrola barevné korony na zeleném pozadí

Peter Šourek

Vedoucí práce: Ing. Roman Berka, Ph.D

Studijní program: Softwarové technologie a management, Bakalářský

Obor: Web a multimédia

20. května 2014

Poděkování

Děkuji panu Ing. Romanu Berkovi, Ph.D za vedení práce a za cenné rady. Dále chci poděkovat panu Ing. Danielovi Sýkorovi, Ph.D za pomoc při hledání materiálů, a v neposlední řadě svým blízkým za podporu při psaní této práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 17. 5. 2014 .....

Abstract

The aim of this bachelor thesis is to create algorithm that is able to detect and suppress color spill on chroma-keyed images. Based on previous analysis and existing solutions to this problem, three solutions are introduced and tested on either static pictures and video sequences. Algorithms were designed not only to produce quality results, but also to be fast and not to be dependent on user input.

Abstrakt

Cílem této práce je na základě analýzy několika testovacích záběrů a zkoumání již existujících řešení, navrhnout algoritmus pro detekci a odstranění barevné korony, vznikající při klíčování na zelené pozadí. V práci jsou představena tři řešení tohoto problému, která jsou otestována jak na fotografiích, tak na video záběrech. Při návrhu těchto řešení, byl kromě kvality výstupu, kladen důraz i na rychlost výpočtu a jednoduchost používání.

Obsah

1 Úvod **1**
1.1 Problém barevné korony . . . 1
1.1.1 Vznik . . . 1
1.1.2 Důsledky . . . 2

2 State of the art **5**
2.1 Klíčování na pozadí . . . 5
2.1.1 Klíčovací rovnice . . . 5
2.1.2 Řešení klíčovací rovnice bez uživatelského vstupu . . . 6
2.1.2.1 Absence modré barvy . . . 6
2.1.2.2 Šedá a odstíny kůže . . . 6
2.1.2.3 Triangulace . . . 7
2.1.3 Algoritmy image mattingu . . . 7
2.1.3.1 Vlahosovy předpolklady . . . 7
2.1.3.2 Klíčování s měkkými hranami . . . 8
2.1.3.3 Bayesian matting . . . 8
2.1.3.4 Laplacian matting . . . 9
2.1.4 Shrnutí . . . 10
2.2 Algoritmy potlačující barevnou koronu . . . 10
2.2.1 Vlahosovo řešení . . . 11
2.2.2 Eliminace korony výpočtem globálního osvětlení . . . 11
2.2.3 Řešení s využitím kontroly pozadí . . . 12
2.2.4 Shrnutí . . . 13
2.3 Cíle práce . . . 13

3 Analýza chování barevné korony a návrh algoritmů **15**
3.1 Analýza . . . 15
3.1.1 Testovací záběry . . . 15
3.1.2 Hodnoty barevných složek . . . 16
3.1.3 Poloha korony . . . 17
3.1.4 Barva korony . . . 18
3.2 Návrh algoritmů . . . 19
3.2.1 Odstranění korony omezením zelené složky . . . 19
3.2.2 Odstranění korony desaturací v oblasti gradientu . . . 20
3.2.3 Desaturace pixelů v oblastech s odstínem klíčovací barvy . . . 20

3.3 Návrh klíčovacího algoritmu . . . 21
3.4 Shrnutí . . . 22

4 Implementace **23**
4.1 Požadavky na aplikaci . . . 23
4.2 Použité technologie . . . 23
4.2.1 FreeGLUT . . . 23
4.2.2 OpenCV . . . 24
4.3 Struktura aplikace . . . 24
4.4 Důležité body návrhu a implementace . . . 25
4.4.1 Výběr vzorků barev . . . 25
4.4.2 Zpracování vzorků pozadí a popředí . . . 26

5 Testování **29**
5.1 Testovací data . . . 29
5.1.1 Fotografie . . . 29
5.1.2 Videosnímky . . . 30
5.2 Návrh testovacích metod . . . 31
5.2.1 Subjektivní posudek . . . 31
5.2.2 Porovnání s existujícím řešením . . . 31
5.2.3 Test rychlosti . . . 31
5.3 Test rychlosti . . . 32
5.4 Testování na statických snímcích . . . 32
5.4.1 Subjektivní test . . . 33
5.4.2 Testování rozdílem výstupů . . . 34
5.5 Testování na videu . . . 35
5.5.1 Subjektivní test . . . 35
5.5.2 Testování rozdílem výsledků . . . 36
5.6 Shrnutí . . . 37

6 Závěr **39**
6.1 Možná vylepšení . . . 39

A Obsah přiloženého CD **43**

Seznam obrázků

1.1 Znázornění vzniku barevné korony na objektech v popředí 2
1.2 Pozorovatelné efekty barevné korony na klíčovaných objektech. Zleva: původní obrázek, průhledná korona, neprůhledná korona 2

2.1 Hexoctahedrony v 3D prostoru souřadnic RGB (převzato z [9] a upraveno) . . 9
2.2 Ukázka trimapy. Černá oblast reprezentuje pozadí obrázku, bílá popředí a šedá barva je neznámý region (převzato z [1]) 10
2.3 Schéma systému využívajícího kontrolu pozadí pro potlačení barevné korony (převzato z [6] a upraveno) 13

3.1 Testovací obrázky použité při analýze barevné korony 16
3.2 Výsledky testu závislosti barevných kanálů 16
3.3 Výsledky výpočtu gradientu odchylky od klíčovací barvy na testovacích záběrech 17
3.4 Výsledky testu odchylky odstínu od klíčovací barvy 18
3.5 Znázornění koulí v RGB prostoru. Menší z koulí obsahuje všechny body pozadí, větší neobsahuje žádné body popředí 22

4.1 Class diagram testovací aplikace 25

5.1 Fotografie použité při testování navržených metod kontroly barevné korony . 29
5.2 Ukázky videosnímků použitých pro testování navržených algoritmů 30
5.3 Ukázky výstupu algoritmů pro potlačení korony 33
5.4 Ukázky výstupu algoritmů pro potlačení korony pro video vstupy 35

Seznam tabulek

5.1 Popis testovacích fotografií 30
5.2 Popis testovacích video záběrů 31
5.3 Přehled hodnot FPS při kontrole barevné korony na testovacích fotografiích (vyšší FPS = rychlejší algoritmus) 32
5.4 Subjektivní posouzení výsledků testování na statických snímcích. Každý parametr byl ohodnocen na stupnici od 1 do 10, kde 10 je nejvyšší hodnota . . . 34
5.5 Odchylky výstupů od referenčního řešení 34
5.6 Subjektivní posouzení výsledků testování na videu. Každý parametr byl ohodnocen na stupnici od 1 do 10, kde 10 je nejvyšší hodnota 36
5.7 Odchylky video výstupů navržených algoritmů od referenčního řešení 37

Kapitola 1

Úvod

Klíčování na pozadí je velice často využívanou technikou ve filmovém průmyslu. Při klíčování však ve výsledném obraze vznikají vlivem nesprávného prostředí (světlo, nevhodné rozmístění objektů ve scéně) některé nežádoucí artefakty, které nepříznivě ovlivňují výsledek i samotný průběh tohoto procesu. Jedním z nich je problém barevné korony (také blue spill, nebo blue flare). Pokud tento problém není při klíčování odstraněn, pak je ve výsledném obrazu na konturách objektů v popředí patrný odraz zeleného resp. modrého klíčovacího plátna, nebo jsou části objektů v popředí zasažené tímto odrazem vyhodnoceny jako pozadí a jsou tak nahrazeny novým pozadím.

Cílem této práce je prozkoumat chování barevné korony na konturách objektů v popředí a na základě tohoto výzkumu pak navrhnout postupy, které tento problém odstraňují. Při návrhu těchto algoritmů bude kladen důraz na rychlost výpočtu, případně možnost paralelizace tak, aby bylo možné tato řešení využívat pro klíčování videa v reálném čase.

V první části práce je nejdříve popsán problém barevné korony, jeho vznik a způsoby jak tomuto problému předcházet. Dále je věnována pozornost vybraným algoritmům image mattingu, které jsou v dnešní době využívány ke klíčování na zelené, případně modré pozadí. Tato část je pak zakončena výčtem některých současných technik pro odstranění barevné korony.

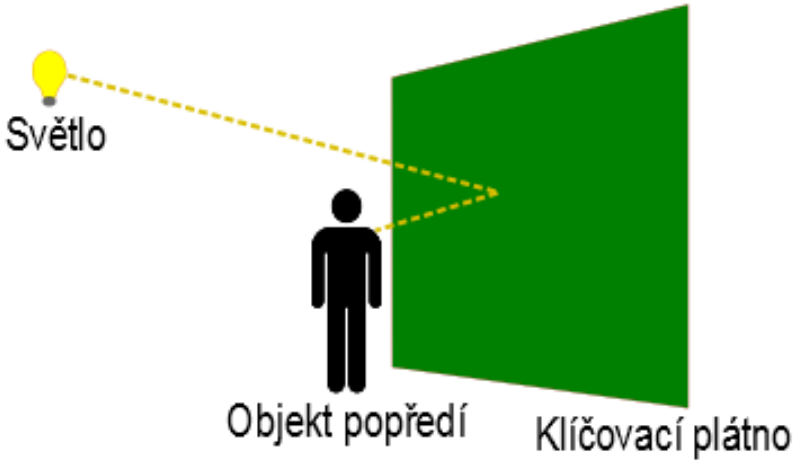
Druhá část práce je věnována zkoumání chování barevné korony a návrhu algoritmů pro její odstranění. Dále následuje popis implementace těchto technik spolu s prezentací výsledků a testováním.

1.1 Problém barevné korony

1.1.1 Vznik

Problém barevné korony je efekt, který vzniká již při snímání klíčované scény. Pokud jsou klíčované objekty příliš blízko klíčovacímu plátnu, popřípadě je scéna nevhodně osvětlena (příliš ostré světlo na pozadí scény), pak je světlo, které dopadá na klíčovací plátno, obarveno jeho barvou a vrací se zpět do scény, v důsledku čehož vzniká viditelný odraz barvy klíčovacího pozadí na objektech v popředí (viz obrázek 1.1).

Na první pohled se může zdát, že prevence vzniku tohoto problému je jednoduchá a že stačí pouze dodržet požadovanou vzdálenost objektů od klíčovacího plátna a správně osvětlit danou scénu. Toto však není v praxi vždy možné, ať už z prostorových důvodů, nebo v důsledku absence kvalitního osvětlení.



Obrázek 1.1: Znázornění vzniku barevné korony na objektech v popředí

Bohužel i při dodržení výše zmíněných zásad se může stát, že se ve výsledném obrazu korona objeví. Pokud se ve scéně vyskytují například zrcadla, nebo poloprůhledné objekty, tak se může stát, že si s takto složitou scénou klíčovací algoritmus neporadí, nebo nebude výsledek vypadat tak, jak by si autor představoval.

1.1.2 Důsledky

Jak již bylo řečeno, existují dva efekty barevné korony, které lze pozorovat ve výsledném obraze po klíčování. V prvním případě se po klíčování na objektech popředí vyskytují oblasti zasažené barvou pozadí a tyto pak působí ve scéně rušivě. K tomuto defektu dochází pokud klíčovací algoritmus nevyhodnotí tyto odrazy jako pozadí a ponechá je tak neprůhledné.

Obrázek 1.2: Pozorovatelné efekty barevné korony na klíčovaných objektech. Zleva: původní obrázek, průhledná korona, neprůhledná korona

Pokud je naopak při procesu klíčování vyhodnocena barva odrazu jako klíčovací barva, jsou plochy zasažené tímto odrazem ve výsledku buď částečně, nebo zcela průhledné. Objekt zasažený koronou i oba zmíněné důsledky jsou znázorněny na obrázku 1.2.

Jak je patrné, barevná korona je velice často přítomný jev při klíčování na monochromatické pozadí a její efekt na výslednou kompozici je velice rušivý. Pokud nechceme, nebo nemůžeme výsledný obraz upravovat v post-produkci (živé vysílání), nebo nelze při snímání scény vytvořit takové podmínky, aby tento efekt nevznikal, je pro úspěšné klíčování naprosto zásadní, aby klíčovací algoritmus tento problém řešil.

Kapitola 2

State of the art

2.1 Klíčování na pozadí

Klíčování na pozadí (v anglické literatuře též image matting) je proces, při kterém je původní pozadí snímané scény nahrazeno jiným pozadím. Tato technologie je v současné televizní a obrazové tvorbě často využívána ať už se jedná o post-produkci při tvorbě filmů, nebo real-time klíčování v televizních studiích. Existuje nepřeberné množství algoritmů a postupů, které se zabývají touto úlohou a do jisté míry se tento problém již dá považovat za dostatečně prozkoumaný. Na druhou stranu velká většina těchto řešení je příliš komplexní a složitá na to, aby mohla být s úspěchem využita při klíčování v reálném čase.

2.1.1 Klíčovací rovnice

Mějme barvu v bodě p zapsanou jako $C(p) = [R, G, B]$. Můžeme si všimnout, že v zápisu barvy není přítomná průhlednost α. Pro takovýto zápis tedy předpokládáme, že hodnoty R, G a B jsou již vynásobeny touto průhledností a že popředí i pozadí je neprůhledné, tedy $\alpha_f = \alpha_b = 1$. Pokud známe klíčovací barvu C_k (ve většině případů zelená, nebo modrá), pak můžeme barvu popředí C_f v každém bodě zapsat jako kompozici této klíčovací barvy a barvy objektu v popředí C_0 takto[10]:

$$C_f(p) = C_0(p) + (1 - \alpha_0)C_k(p) \tag{2.1}$$

Pokud tedy má být klíčovací barva nahrazena barvou pozadí, bude výsledná hodnota barvy obrazového bodu po klíčování C určena jako:

$$C(p) = C_0(p) + (1 - \alpha_0)C_b(p) \tag{2.2}$$

Pokud přepíšeme rovnici 2.1 pro každou barevnou složku, získáme

$$\begin{aligned} R_f(p) &= R_0(p) + (1 - \alpha_0)R_k(p) \\ G_f(p) &= G_0(p) + (1 - \alpha_0)G_k(p) \\ B_f(p) &= B_0(p) + (1 - \alpha_0)B_k(p) \end{aligned}$$

Pro získání výsledné barvy popředí C_0 musí být známy hodnoty R_0, G_0, B_0 a α_0. Tato soustava má nekonečně mnoho řešení, protože se v ní vyskytují čtyři neznámé a pouze tři

rovnice. Z toho vyplývá že algoritmus pro klíčování musí mít k dispozici ještě další informaci (uživatelský vstup), s jejíž pomocí je schopen určit výslednou hodnotu C_0 [10].

2.1.2 Řešení klíčovací rovnice bez uživatelského vstupu

Smith[10] ve svém článku uvádí několik případů, pro které je tato soustava řešitelná i bez uživatelského vstupu. Velkým nedostatkem všech těchto řešení je velice malá množina situací, pro které tyto postupy fungují. Všechna následující řešení kladou na objekty popředí a barvu pozadí velice silné omezující podmínky, nebo v případě triangulace vyžadují speciální postup, který lze v praxi jen těžko provést.

2.1.2.1 Absence modré barvy

První z těchto případů je situace, kdy se na objektu v popředí scény nenachází žádná modrá barva. Samozřejmě to samé platí pokud použijeme jako klíčovací barvu zelenou.Jestliže víme že C_0 neobsahuje modrou barvu, tedy $C_0 = [R_0, G_0, 0]$ a C_k obsahuje pouze modrou barvu, $C_k = [0, 0, B_k]$, pak platí že:

$$C_f(p) = C_0(p) + (1 - \alpha_0)C_k(p) = [R_0, G_0, (1 - \alpha_0)B_k]$$

Z tohoto vztahu lze dále odvodit

$$C_0 = \left[R_f, G_f, 0, 1 - \frac{B_f}{B_k}\right]$$

Zjevnou nevýhodou takového řešení je velice silné omezení na objekty v popředí a na barvu pozadí. V praxi je téměř nemožné zajistit podmínky, které by toto splňovaly.

2.1.2.2 Šedá a odstíny kůže

Dále Smith představil řešení pro šedou barvu popředí, tedy $R_0 = G_0 = B_0$ (dále uvidíme že řešení ve skutečnosti platí pro mnohem obecnější množinu barev). Pokud R_0 nebo G_0 je rovno $aB_0 + b\alpha_0$ a pro C_k platí že $aB_k + b \neq 0$ a zároveň $C_k = [0, 0, B_k]$. Po aplikaci těchto podmínek lze zapsat barvu popředí takto(pro zelenou složku barvy):

$$C_f = [R_0, aB_0 + b\alpha_0, B_0 + (1 - \alpha_0)B_k]$$

a po vytknutí B_0 z výrazů pro G_f a B_f lze zapsat řešení včetně složky α_0 následujícím způsobem:

$$C_0 = \left[R_f, G_f, B_\Delta + \alpha_0 B_k, \frac{G_f - aB_\Delta}{aB_k + b}\right], aB_k + b \neq 0$$

kde $B_\Delta = B_f - B_k$. Odstíny šedi tyto podmínky zjevně splňují a pokud uvažujeme barvu lidské kůže jako $[d, 0.5d, 0.5d]$,pak tato vyhovuje podmínkám také(d je takzvaný zesvětlovací faktor). Důležitým předpokladem u tohoto řešení je čistá modrá(případně zelená) klíčovací barva.

2.1.2.3 Triangulace

Poslední řešení, které Smith ve svém článku označuje jako triangulaci (i přes to, že tento pojem pochází původně z trigonometrie a jedná se o způsob zjišťování vzdáleností), je založeno na podmínce, že klíčovaný objekt je nasnímán před dvěma různými odstíny modré(zelené). Tento předpoklad lze matematicky popsat následujícím způsobem:

$$\begin{aligned} C_{f_1} &= [R_0, G_0, B_0 + (1-\alpha_0)B_{k_1}] \\ C_{f_2} &= [R_0, G_0, B_0 + (1-\alpha_0)B_{k_2}] \end{aligned}$$

Pokud zkombinujeme výrazy pro B_{f_1} a B_{f_2}, tak získáme vztah pro α_0:

$$\alpha_0 = 1 - \frac{B_{f_1} - B_{f_2}}{B_{k_1} - B_{k_2}}, B_{k_1} \neq B_{k_2}$$

Podmínka $B_{k_1} \neq B_{k_2}$ je splněna vždy, protože předpokládáme dvě rozdílná pozadí. Následně je určeno finální řešení klíčovacího problému jako:

$$\begin{aligned} R_0 &= R_{f_1} = R_{f_2} \\ G_0 &= G_{f_1} = G_{f_2} \\ B_0 &= \frac{B_{f_2}B_{k_1} - B_{f_1}B_{k_2}}{B_{k_1} - B_{k_2}} \end{aligned}$$

a obdobně pokud bychom použili pro klíčování zelenou barvu.

Toto řešení podává velice dobré výsledky, avšak jeho použití je silně omezeno faktem, že je zapotřebí objekt nasnímat před dvěma různými odstíny modré. Toho je do jisté míry možné dosáhnout při klíčování samostatných fotografií, na druhou stranu při klíčování videa je tento postup prakticky nepoužitelný.

2.1.3 Algoritmy image mattingu

V této části práce budou vyjmenovány a popsány vybrané algoritmy image mattingu, které se v současné filmové a televizní tvorbě používají, nebo používaly. Nejdříve je věnována pozornost algoritmům image mattingu a poté jsou zmíněny i některé algoritmy natural image mattingu, které se zabývají oddělením popředí od předem neznámého pozadí. Tyto algoritmy jsou mnohem silnější a komplexnější, protože nepotřebují pro svůj chod, aby byl objekt snímán před pozadím s konstantní, nebo téměř konstantní barvou.

2.1.3.1 Vlahosovy předpolklady

Petro Vlahos ve svých patentech [11] a [12] položil základy takzvaného bluescreen mattingu. Omezil soustavu klíčovacích rovnic několika podmínkami (Vlahosovy předpoklady) a umožnil tak nalézt řešení těchto rovnic. První z těchto předpokladů říká, že pro objekty v popředí platí vztah

$$B_0 \leq a_2 G_0$$

kde a_2 je parametr, který může uživatel ovlivňovat a jeho hodnota je typicky omezena na interval $(0.5, 1.5)$ [12]. Výraz pro α_0 vypadá podle Vlahose(notace převzata z [10]) takto:

$$\alpha_0 = 1 - a_1(B_f - a_2 G_f)$$

kde a_1 a a_2 jsou opět uživatelské proměnné a hodnota α_0 je oříznuta na interval $< 0, 1 >$. Vlahos doporučuje nahradit v klíčovaném snímku hodnotu B_f za $min(B_f, B_k)$, kde B_k je klíčovací barva. Pokud klíčovací barva není zcela konstantní (téměř vždy), pak je použita minimální hodnota B_k. Obecnější Vlahosův předpoklad, který uvádí v [11] vypadá následovně:

$$\alpha_0 = 1 - a_1(B_f - a_2(a_5 max(r, g) + (1 - a_5) min(r, g)))$$

kde $r = a_3 R_f, g = a_4 G_f$ a a_i jsou uživatelské proměnné. Hodnota je opět oříznuta na interval $< 0, 1 >$ a hodnota modré barvy je opět upravena jako v předchozím případě. Po jednoduché úpravě lze tyto předpoklady použít i pro klíčování na zeleném plátně.

2.1.3.2 Klíčování s měkkými hranami

V roce 1994 Yasushi Mishima publikoval způsob klíčování na monochromatické pozadí s měkkými hranami [9]. Tento algoritmus nejdříve z uživatelem zvoleného vzorku pozadí spočítá průměrnou hodnotu barvy a kolem této pak sestrojí v 3D prostoru souřadnic RGB útvar, který Mishima označuje jako hexoctahedron [1] a tento obsahuje všechny body reprezentující pixely daného vzorku pozadí. Dále systém vytvoří druhý hexoctahedron, který je naopak největší možný takový, aby neobsahoval žádné pixely popředí (viz obrázek 2.1).

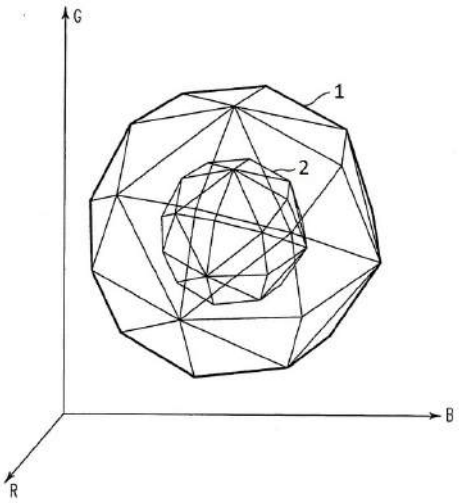
Obraz je dále zpracováván tak, že pixely, které se nachází uvnitř menšího z těchto útvarů (2), mají danou hodnotu $\alpha_0 = 0$, protože jsou považovány za pixely pozadí. Pokud se pixel ocitne vně většího z octahedronů (1), pak jeho hodnota $\alpha_0 = 1$ a je tedy považován za pixel popředí. Ostatní hodnoty RGB (tedy ty co leží vně 2 a zároveň uvnitř 1) jsou považovány za částečně průhledné a hodnota α_0 pro takovéto body je lineárně interpolována.

2.1.3.3 Bayesian matting

Další algoritmy již budou řešit problém takzvaného natural image mattingu, tedy separaci objektu v popředí na předem neznámém pozadí. Tato řešení jsou zde uvedena jen pro úplnost, protože pro účely této práce není třeba se zabývat takto obecným problémem klíčování (při klíčování v televizní tvorbě je v drtivé většině využíváno techniky bluescreen resp. greenscreen mattingu). Na druhou stranu i tyto metody mohou pro budoucí práci přinést některé cenné poznatky. Většina takových řešení využívá uživatelem definovanou trimapu, což je rozdělení obrázku do tří regionů, z nichž jeden reprezentuje část, která je určitě pozadí scény, druhá je popředí a poslední z těchto oblastí je takzvaný neznámý region.

Bayesian matting [3] je technika, která předpokládá že odstíny barev, nacházejících se v pozadí resp. v popředí pochází ze stejného pravděpodobnostního rozdělení. Tato pravděpodobnostní rozdělení jsou získána na základě známých vzorků poskytnutých uživatelem prostřednictvím trimapy. Hodnoty F, B a α_0 jsou pak vypočítány jako nejpravděpodobnější hodnoty pocházející z těchto rozdělení.

[1] Hexoctahedron je termín který původně pochází z mineralogie a označuje izometrický útvar, který se skládá z 48mi stejných trojúhelníkových ploch [2]



Obrázek 2.1: Hexoctahedrony v 3D prostoru souřadnic RGB (převzato z [9] a upraveno)

2.1.3.4 Laplacian matting

V roce 2008 bylo publikováno řešení problému natural image mattingu [8], které pro svůj chod nepotřebuje kompletní trimapu 2.2, ale stačí mu poskytnout informace o popředí a pozadí pomocí několika tahů "štětcem". Tato metoda je založena na předpokladu, že barva popředí, resp. pozadí, neobsahuje vysoké lokální změny. Jinými slovy lze říci, že barva pixelů v okolí zkoumaného bodu se nebude skokově měnit.

Tento algoritmus tedy rozdělí obraz do několika oken w hodnota α pak lze v tomto okně vyjádřit jako

$$\alpha_i = \sum_c a^c I_i^c + b, \quad \forall i \, \epsilon \, w$$

kde c označuje barevné kanály a a^c, b^c jsou konstanty daného okna. Dále lze z obecné nákladové funkce $J(\alpha, a, b)$ [8] určit nákladovou funkci pouze pro hodnotu α jako:

$$J(\alpha) = \alpha^T L \alpha$$

kde L je matice velikosti $N \times N$, jejíž (i,j)-tý prvek je definován jako:

$$\sum_{k|(i,j)\epsilon w_k} \left(\delta_{ij} - \frac{1}{|w_k|} \left(1 + (I_i - \mu_k) \times \left(\Sigma_k + \frac{\epsilon}{|w_k|} I_3 \right)^{-1} (I_j - \mu_k) \right) \right)$$

kde Σ_k je kovarianční matice velikosti 3×3, μ_k je tříprvkový vektor střední hodnoty barev v okně w_k a I_3 je matice identity velikosti 3×3. Optimální hodnoty α lze pak vyjádřit za pomocí výše popsané matice takto:

$$\alpha = arg\ min\ \alpha^T L \alpha$$

Tento postup byl již s úspěchem implementován pro výpočet na grafické kartě za pomocí technologie CUDA [7], avšak i při této implementaci se doby výpočtů pohybovaly v řádů sekund, což znamená že i toto řešení není vhodné pro klíčování v reálném čase.

Obrázek 2.2: Ukázka trimapy. Černá oblast reprezentuje pozadí obrázku, bílá popředí a šedá barva je neznámý region (převzato z [1])

2.1.4 Shrnutí

V dnešní době je spíše kladen důraz na výzkum v oblasti natural image mattingu, tedy separace popředí z obecného pozadí. Tato práce má však za úkol řešit problém barevné korony, který vzniká pouze při klíčování na zelené, resp. modré pozadí, proto v této části nejsou algoritmy natural image mattingu popisovány tak podrobně, jako postupy využívající monochromatické pozadí.

Všechny výše popsané metody fungují do jisté míry spolehlivě a jsou schopny dodat kvalitní kompozice. I tak je ovšem důležité dbát při pořizování snímků pro klíčování na správné osvětlení, protože pokud se například vlivem stínů objeví na klíčovacím pozadí tmavé pixely, algoritmus je může snadno zaměnit za pixely popředí a tak nastavit jejich hodnotu $\alpha = 0$, což je jistě nechtěný efekt.

2.2 Algoritmy potlačující barevnou koronu

V této kapitole jsou popsány některé existující postupy pro potlačení barevné korony (v anglické literatuře také color spill, nebo color flare). Jak již bylo řečeno, tento nedostatek vzniká v důsledku špatného nasvícení klíčované scény a projevuje se viditelným odrazem světla z klíčovacího plátna na konturách objektů v popředí. Současná filmová technika řeší tento problém zejména úpravou nasvícení klíčovacího plátna tak, aby se od něj světlo neodráželo. Toto řešení je samozřejmě korektní, avšak velmi omezuje světelné podmínky scény a tím pádem i možnosti uměleckého vyjádření autora.

Během posledních let se na tomto poli objevilo několik více či méně spolehlivých a použitelných řešení, ovšem tato řešení většinou vyžadují, aby klíčovací algoritmus měl k dispozici informace o scéně, jako je geometrický popis objektů v popředí, barva a směr světel

a podobně, nebo využívají proměnlivé klíčovací pozadí, či jiná omezení samotného procesu snímání klíčované scény.

2.2.1 Vlahosovo řešení

Petro Vlahos ve svém patentu [12] vyřešil problém barevné korony pro zářivé odstíny bílé a barvu kůže. Vlahos předpokládal, že pro objekty v popředí platí, že $max(B_0 - G_0, 0) \leq max(G_0 - R_0, 0)$. Pokud tento předpoklad neplatí, pak je barva považována buď za barvu pozadí, nebo za blue spill.

Smith a Blinn [10] tento předpoklad zahrnuli do klíčovací rovnice 2.2 s použitím další proměnné a to konkrétně barvy C_s, což je příspěvek barevné korony. Tento příspěvek může být modelován jako objekt popředí se svou vlastní průhledností α_s takto:

$$C_f = C_s + (1 - \alpha_s)(C_0 + (1 - \alpha_0)C_k)$$

Při znalosti pouze barvy popředí C_f a klíčovací barvy C_k je problém klíčování mnohem složitější pokud bereme v úvahu také příspěvek barevné korony. Smith a Blinn navrhují zjednodušení při předpokladu že barva C_s je stejná jako klíčovací barva, tedy $C_s = \alpha_s C_k$. Rovnice vyjadřující bravu popředí pak vypadá následovně:

$$C_f = (1 - \alpha_s)C_0 + (1 - \alpha_0 + \alpha_0\alpha_s)$$

Smith a Blinn tento problém doporučují k dalšímu zkoumání, protože i po tomto zjednodušení je hodnota α_s neznámá a tím pádem má tato rovnice stále nekonečně mnoho řešení.

2.2.2 Eliminace korony výpočtem globálního osvětlení

V roce 2006 byl představen nový přístup k problému barevné korony [4]. Tento postup je založen na výpočtu globálního osvětlení scény, které navrhli v roce 1989 Forsyth a Zisserman [5]. Principem této metody je výpočet jasu L_0 jako součet příspěvku světla, které přichází ze zdroje L_0^s a příspěvku L_0^e, který je odrazem světla od pozadí scény. Tento fakt můžeme zapsat následujícím způsobem:

$$L_0(p) = L_0^s(p) + L_0^e(p)$$

pro každý bod scény p. Pokud budeme uvažovat pouze Lambertův odraz[2] a bodový zdroj světla umístěný v nekonečnu, pak lze příspěvek světelného zdroje zapsat jako

$$L_0^s = \rho(p)L_s(\vec{i_s} \cdot \vec{n}(p))$$

kde ρ označuje odrazivost povrchu, tedy poměr světla, které přichází na povrch a světla které odchází po odrazu, $\vec{n}(p)$ je normála k povrchu v bodě p a $\vec{i_s}$ je směrový vektor zdrojového světla. Dále je nutné vyjádřit hodnotu L_0^e, která představuje všechny příspěvky světla z odrazů od ostatních objektů ve scéně. Toho lze dosáhnout následujícím zápisem:

$$L_0^e = \rho(p)L_e \int_\epsilon K(p, p')dp'$$

[2]Lambertův odraz je takový, který světelný paprsek přicházející na daný povrch odráží všemi směry ve stejné intenzitě

kde ϵ je množina bodů p', které přijímají světlo ze zdroje, $K(p, p')$ značí geometrický faktor zisku [5] a $L_e(p')$ je jas v bodě p'.

Po zavedení substituce $k_s(p) = \vec{i_s} \cdot \vec{n}(p)$ a $k_e(p) = \int_\epsilon K(p, p')dp'$ z předchozích vztahů získáme:

$$\rho(p)L_s k_s(p) + \rho(p)L_e k_e(p)$$

Faktory k_s a k_e reprezentují Lambertovy odrazy vznikající na objektu vlivem přímého, nebo nepřímého osvětlení. Pokud toto vyjádříme jako vztah pro výpočet barvy namísto jasu, získáme:

$$C_0 = k_s \rho C_s + k_e \rho C_e$$

Posledním krokem je předpoklad, že barva $C_e = C_b$. nyní lze vyjádřit výslednou barvu jako

$$C = \alpha k_s \rho C_s + ((1 - \alpha) + \alpha k_e \rho)C_e$$

Tento výraz modeluje problém barevné korony při klíčování.

Dupont a Deschenes [4] ve svém článku také zmiňují dva způsoby nalezení řešení této rovnice. Při znalosti odrazivosti ρ lze rovnici vyřešit bez větších problémů. Pokud odrazivost povrchu neznáme, pak Dupont navrhuje využití uživatelem zadané trimapy (pro jednoduchost zde nejsou uvedena přesná znění rovnic pro výpočet se známou i neznámou odrazivostí, pro více informací viz [4]).

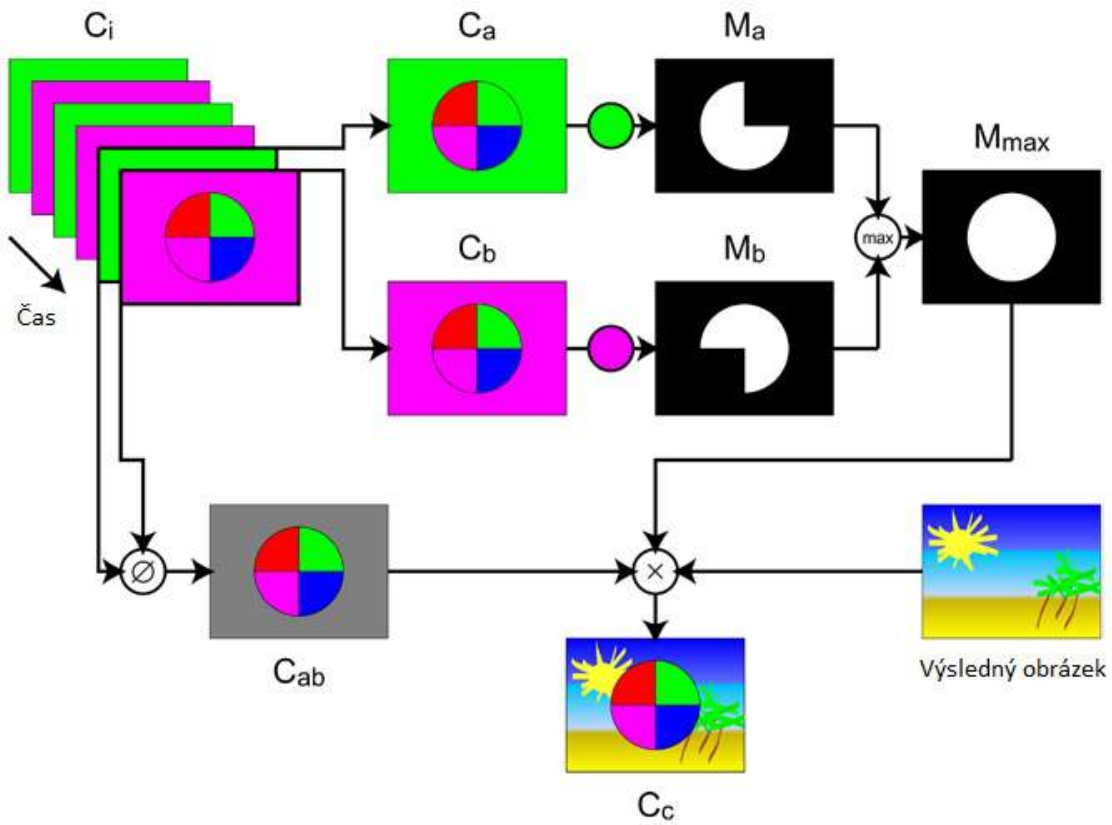
2.2.3 Řešení s využitím kontroly pozadí

Jiný přístup k odstranění barevné korony představený v [6] je založený na principu kontroly klíčovacího pozadí. Autoři zde navrhují zvýšit snímací frekvenci kamery na dvojnásobek požadovaného frameratu, tedy například pro požadovanou frekvenci 30 snímků za sekundu by musela být scéna snímána rychlostí 60 snímků za sekundu, a v každém snímku dynamicky měnit barvu klíčovacího pozadí.

Můžeme si všimnout jisté podobnosti s postupem triangulace který byl navržen v [10] (viz 2.1.2.3). Rozdíl je však v tom, že zde se považují dva po sobě jdoucí snímky za jeden snímek, natočený před různými pozadími a hodnota výsledné barvy snímku je pak vypočtená, jako průměr z těchto dvou obrazů C_a a C_b, tedy $C_{ab} = (C_a + C_b)/2$. Tento postup je znázorněn na obrázku 2.3.

Odstranění barevné korony je pak dosaženo tím, že jako klíčovací pozadí jsou použity doplňkové barvy a v důsledku toho je barevná korona transformována na neutrální osvětlení pozadí. Algoritmus se dále musí vypořádat s faktem, že při pohybu nejsou dva po sobě jdoucí snímky úplně stejné a tak vzniká možnost chyby. Autoři tento problém řeší takzvanou kontrolou švů "seam control".

I přesto, že tento systém funguje korektně a podává dobré výsledky i při klíčování v reálném čase, tak jeho využití může být problematické právě kvůli faktu, že pro jeho chod je nutné mít možnost kontroly barvy pozadí a to pak synchronizovat se snímací frekvencí kamery.



Obrázek 2.3: Schéma systému využívajícího kontrolu pozadí pro potlačení barevné korony (převzato z [6] a upraveno)

2.2.4 Shrnutí

I přesto, že výše popsané algoritmy řeší s úspěchem problém barevné korony, tak jejich využití v praxi je velmi nepravděpodobné (s výjimkou Vlahosových předpokladů), jelikož pro jejich chod je nutná buď znalost struktury scény (odrazivost povrchů, barvy světel apod.), nebo je pro každý snímek třeba dodat uživatelem definovanou trimapu, případně jsou kladeny vysoké technické nároky na pozadí scény (možnost změny barvy a synchronizace s kamerou).

2.3 Cíle práce

Hlavním cílem práce je navrhnout a implementovat algoritmus, který bude schopen detekovat a odstranit barevnou koronu vznikající při klíčování na barevné pozadí. Na rozdíl od dříve zmíněných řešení bude kladen důraz na to, aby navržené řešení bylo možné s úspěchem paralelizovat a aby výsledný algoritmus potřeboval co nejmenší vstup od uživatele.

Dalším požadavkem je, aby odstranění korony bylo nezávislé na použitém klíčovacím algoritmu, tedy aby mohl být klíčovací algoritmus volně nahraditelný, aniž by tento krok

ovlivnil detekci, případně odstranění barevné korony. Tento požadavek je velmi zásadní, jelikož umožní použití algoritmu odstranění barevné korony spolu s již existujícími algoritmy pro klíčování na zeleném, resp. modrém plátně.

Posledním požadavkem na navržené řešení je, aby pro svůj chod nevyžadoval speciální techniku, jako tomu bylo v případě odstranění korony pomocí kontroly klíčovacího pozadí (viz 2.2.3).

Kapitola 3

Analýza chování barevné korony a návrh algoritmů

3.1 Analýza

Tato část práce se zabývá analýzou některých testovacích snímků zasažených barevnou koronou. Hlavním účelem je zjistit jakým způsobem se korona chová a zda existuje nějaká možnost, jak na základě barevné informace určit, zda je daný pixel obrazu ovlivněn klíčovací barvou a zda lze tento defekt opravit.

Samotná analýza je provedena podle tří kritérií. Nejdříve jsou zkoumány závislosti jednotlivých barevných složek RGB na částech obrázku zasažených barevnou koronou a jejich případná podobnost s hodnotami RGB na klíčovacím pozadí. Dále byla zkoumána poloha korony a nejčastější místa kde, má barevná korona tendenci se vyskytovat. Posledním pokusem je pak snaha o nalezení podobnosti odstínu barvy korony v barevném modelu HSV s odstínem klíčovacího pozadí.

3.1.1 Testovací záběry

Při pořizování testovacích záběrů byl kladen důraz na to, aby barevná korona nebyla potlačena, ale naopak se vyskytovala na výsledném snímku co nejvíce. Dalším kritériem bylo pokrýt co největší záběr problematických povrchů, které mají obecně tendenci defektem barevné korony trpět.

Obrázek 3.1a představuje světlý povrch s jemným ochlupením, na kterém se barevná korona objevuje většinou na okrajích objektu a v tomto případě i na částech natočených směrem ke klíčovacímu plátnu. Na obrázku 3.1c se naopak kombinuje lesklý světlý povrch s bílým materiálem vlasů a částečně průhlednými křídly. Barevná korona je zde patrná v podstatě na celém objektu popředí a s velikou pravděpodobností se bude po naklíčování vyskytovat i v oblasti vlasů. Na snímku 3.1b je pak demonstrováno extrémně nekvalitní osvětlení, spolu s objektem popředí, který se barevně neliší tak výrazně od barvy pozadí, jako tomu bylo v předchozích dvou případech. Zde není barevná korona tak výrazná (pouze drobné zabarvení na okrajích objektů).

Na všech testovacích záběrech jsou viditelné značné nerovnoměrnosti v barvě klíčovacího pozadí. Toto je způsobeno právě nevhodným osvětlením a faktem, že klíčované objekty jsou příliš blízko klíčovacímu plátnu, v důsledku čehož se na plátně vytváří stíny.

(a) (b) (c)

Obrázek 3.1: Testovací obrázky použité při analýze barevné korony

3.1.2 Hodnoty barevných složek

První pokus má za úkol potvrdit Vlahosovy předpoklady představené v části 2.1.3.1. K tomuto účelu byl pro každý pixel testovacích snímků spočítán rozdíl zelené a modré, resp. červené složky a tyto pak byly zobrazeny jako barvy pixelu. Pokud tedy zelená složka nebyla v daném obrazovém bodě vyšší než jedna z dalších složek, pak je výsledná barva černá, pokud je zelená složka výraznější, pak je výsledný pixel šedý. Tento postup by měl zvýraznit jak klíčovací pozadí, tak barevnou koronu, která se na objektech popředí vyskytuje.

(a) (b) (c)

Obrázek 3.2: Výsledky testu závislosti barevných kanálů

Na obrázcích 3.2a a 3.2c jsou patrná místa na objektech v popředí, kde světlejší odstíny šedi indikují přítomnost barevné korony, lze tedy předpokládat, že pokud algoritmus bude schopen vyhodnotit vztah mezi jednotlivými složkami, pak bude zároveň schopen tento nedostatek detekovat a případně odstranit.

Výsledek 3.2b však tyto plochy neobsahuje a mezi popředím a pozadím je pouze ostrý

přechod, což může vést na stav, při kterém tento postup vyhodnocování nebude dostačující a s velkou pravděpodobností bude generovat nekorektní výsledky.

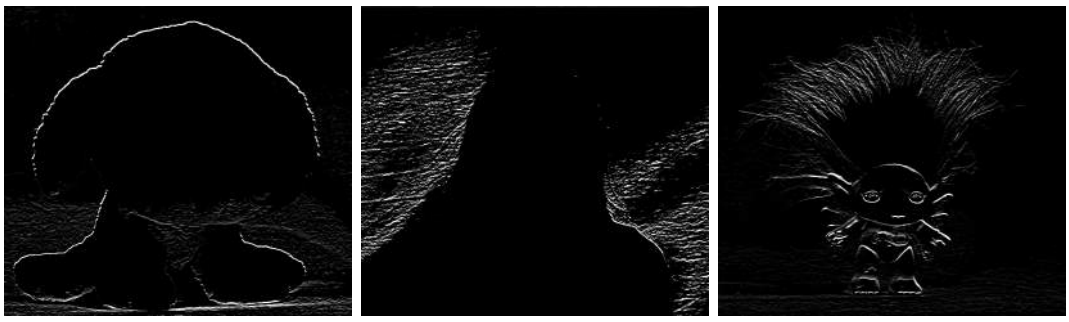
Tento pokus tedy do jisté míry potvrzuje Vlahosovy předpoklady, které tvrdí že pokud $G_0 > R_0 \vee G_0 > B_0$ pak je pixel buď barvou pozadí, nebo barevnou koronou. Velikou výhodou tohoto předpokladu je fakt, že jej lze ověřovat pro každý obrazový bod zvlášť a tak poskytuje velice dobré podmínky pro případnou paralelizaci celého výpočtu.

3.1.3 Poloha korony

Další pokus se nezabývá hodnotou barvy pixelu přímo, ale zkoumá jeho změnu vůči okolním bodům. Jak je vidět na obrázku 3.1a, barevná korona má tendenci se objevovat na hranici mezi popředím a pozadím. Na základě tohoto předpokladu můžeme odvodit že pokud bude algoritmus schopen tyto hranice najít, pak bude schopen najít oblasti zasažené barevnou koronou.

Pro nalezení těchto hranic musíme definovat funkci $computeDifference(C_0, C_k)$, která pro zadanou klíčovací barvu C_k a barvu pixelu C_0 vypočítá index vzdálenosti, na základě kterého pak bude algoritmus hledat hrany ve vstupním obrazu.

Ke hledání hran byly použity Sobelovy operátory a vstupní obraz byl před výpočtem rozmazán pomocí konvoluce Gaussovým operátorem rozostření, aby byl odstraněn případný vysokofrekvenční šum, který by mohl výsledek znehodnotit.



(a) (b) (c)

Obrázek 3.3: Výsledky výpočtu gradientu odchylky od klíčovací barvy na testovacích záběrech

Na výsledcích 3.3a a 3.3c je vidět, že dříve zmíněný postup výpočtu gradientu pomocí Sobelových operátorů a odchylky od klíčovací barvy zvýrazňuje oblasti, které mohou být potencionálně zasaženy barevnou koronou. Zejména se jedná o okraje objektů a u obrázku 3.3c i oblasti vlasů, které mají obecně tendenci tímto defektem trpět. Gradient dokonce do jisté míry zachytil i větší plochy zasažené barevnou koronou, avšak informace o těchto plochách trpí velice výraznými nepravidelnostmi, které se nepodařilo odstranit ani Gaussovským rozmazáním. Také si můžeme všimnout, že tento test zvýraznil i nepravidelnosti klíčovacího plátna, což by ve výsledku nemělo nijak uškodit vzhledem k tomu, že tyto oblasti budou po naklíčování zcela průhledné.

Nerovnoměrné pozadí je nejvíce patrné na obrázku 3.3b, u kterého byl značný problém nastavit hodnotu klíčovací barvy tak, aby algoritmus pracoval alespoň částečně. Na tomto

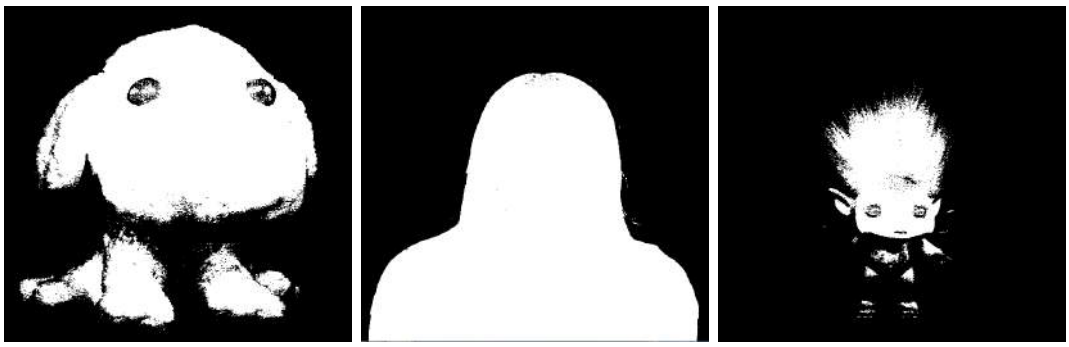
obrázku se tedy hledání korony pomocí gradientu neosvědčilo a můžeme předpokládat že tento algoritmus nebude fungovat pro všechny vstupy. Na druhou stranu je šumem z výpočtu gradientu zasažena jen oblast pozadí a tak by algoritmus alespoň neměl výsledný obraz znehodnotit, pokud nebude schopen pro daný vstupní obraz barevnou koronu odstranit.

3.1.4 Barva korony

Poslední analýza se zabývá detekcí korony na základě odchylky odstínu barvy pixelu od klíčovací barvy. K tomuto účelu byl použit barevný prostor HSV (Hue, Saturation, Value), konkrétně jeho složka odstínu, která reprezentuje úhel na barevném válci představujícím tento barevný prostor.

Základním předpokladem je, že barva korony je stejná, nebo podobná jako barva klíčovacího pozadí, což znamená, že pokud se nachází složka odstínu v HSV reprezentaci zkoumaného pixelu v definovaném okolí odstínu klíčovací barvy, pak lze tento pixel považovat za zasažený barevnou koronou.

Důležitá je pro správný průběh tohoto pokusu volba odchylky o kterou se může korona lišit od klíčovací barvy. Pokud objekty v popředí nejsou dostatečně výrazně barevně odlišné od klíčovací barvy, pak se může stát, že algoritmus vyhodnotí objekty popředí jako koronu, což může později vést na znehodnocení výsledného obrazu.



(a) (b) (c)

Obrázek 3.4: Výsledky testu odchylky odstínu od klíčovací barvy

Obrázek 3.4b zobrazuje výsledky detekce barevné korony na základě barevné odchylky. V prvních dvou případech byla použita odchylka 36° a v druhém 13°. Bílá barva značí popředí bez korony a černou barvou jsou označeny pixely, které jsou buď zasaženy koronou, nebo se jedná o pozadí scény.

Ze snímků 3.4a a 3.4c je patrné, že tento způsob detekce korony pracuje přesně a že bílou barvou jsou skutečně označeny pouze ty části obrázků, které nejsou zasaženy barevnou koronou. Na druhou stranu se ve výsledku objevují díry způsobené prahováním. To, jak výrazně tento nedostatek ovlivní výslednou kvalitu klíčovaného snímku, je dáno tím, jaká technika bude použita pro odstranění korony a jak výrazné toto odstranění bude.

Obrázek 3.4b, stejně jako u předchozích dvou metod, trpí nepřesnostmi a nedostatečnou kvalitou detekce korony, avšak u tohoto postupu lze do jisté míry nepřesnosti korigovat pomocí změny velikosti odchylky.

3.2 Návrh algoritmů

Na základě předchozí analýzy, budou v této části navrženy algoritmy detekce a potlačení korony. Každý z následujících algoritmů lze rozdělit na dvě části: detekce a potlačení korony. V části 3.1 bylo prostřednictvím tří pokusů představeno několik řešení detekce barevné korony. Algoritmus založený na detekci korony zkoumáním závislosti hodnot barevných kanálů využívá k potlačení korony metodu omezení zeleného kanálu navrženou Petro Vlahosem. Další dva algoritmy potlačují koronu omezením saturace v barevném prostoru HSV.

3.2.1 Odstranění korony omezením zelené složky

Pokud budeme vycházet z poznatků získaných v části 3.1.2, pak lze navrhnout řešení problému barevné korony, které by omezilo rozsah zelené složky ve vstupním snímku podle dalších dvou složek v barevné reprezentaci RGB.

function *spillSuppress(input image* C_0*)*
  for *each* $p \in C_0$ **do**
    if $G_p > R_p \vee G_p > B_p$ **then**
      $G_p = average(R_p, B_p)$;
    end
  end

Algoritmus 1: Funkce spillSuppress

Algoritums 1 popisuje funkci *spillSuppress*, která potlačuje ve vstupním obrázku C_0 barevnou koronu. Pokud je hodnota zelené složky právě zkoumaného pixelu G_p větší než jedna z dalších složek RGB, pak je tato nastavena na průměr zbylých dvou složek. Vlahos sice navrhuje omezit zelený kanál pouze podle jedné ze složek B_p a R_p, ale postup s omezením zeleného kanálu na průměr ostatních kanálů se u testovacích záběrů více osvědčil a podával lepší výsledky.

Výhodou tohoto algoritmu je jednoduchost a dobrá možnost paralelizace. Cyklus, který prochází všechny body p z obrázku C_0, lze s úspěchem počítat na grafické kartě, nebo rozložit data obrázku, tak aby je mohla zpracovávat vícevláknová aplikace.

for *each* $p \in C_0$ **do**
  $\alpha_0 = \text{computeDifference}(p, C_k)$;
  $\alpha_p = \alpha_0$;
end
`spillSuppress(`C_0`)` ;

Algoritmus 2: Klíčovací algoritmus zkombinovaný s funkcí odstranění korony

Integrace klíčovacího algoritmu s funkcí spillSuppress popsanou dříve, je ukázána v algoritmu 2. Můžeme si všimnout, že je dodržen požadavek na nezávislost klíčovacího algoritmu a algoritmu potlačení barevné korony. Drobné zpomalení by mohlo nastat v důsledku toho, že je celý vstupní snímek čtený dvěma cykly. Tento nedostatek lze samozřejmě odstranit přesunutím cyklu nacházejícího se ve funkci spillSuppress do klíčovacího algoritmu a tak vše počítat v rámci jednoho průchodu obrázkem.

3.2.2 Odstranění korony desaturací v oblasti gradientu

Další z navrhovaných řešení vychází z analýzy 3.1.3 a používá k detekci korony gradient α_0. S využitím této informace o vstupním obrázku můžeme detekovat přechody mezi objekty popředí a pozadí. Právě v těchto místech má barevná korona nejvíce tendenci vznikat.

Důležitou částí tohoto postupu je samotné potlačení korony. K tomuto účelu bylo použito potlačení saturace v barevné reprezentaci HSV daného pixelu. Tato metoda se osvědčila, jelikož nemění samotný odstín barvy pixelu, ale pouze jeho sytost, takže výsledný efekt působí spojitě.

function *spillSuppress(input image C_0)*
  for *each* $p \in C_0$ **do**
    $g = 1-$ computeGradient(p, C_0);
    $p =$ toHSV(p);
    $S_p = g * S_p$;
    $p =$ toRGB(p);
  end

Algoritmus 3: Funkce spillSuppress s použitím gradientu

Funkce spillSuppress v algoritmu 3 využívá pro svojí práci barevný prostor HSV a funkci computeGradient. Tato funkce zde pro jednoduchost není rozepsána, ale v podstatě se jedná pouze o konvoluci nejdříve Gaussovým operátorem rozmazání a poté Sobelovými operátory s tím že všechny tyto operace nejsou prováděny přímo na barvě okolních pixelů, ale na jejich hodnotě α_0, která je výstupem funkce computeDifference.

Tuto metodu lze opět snadno paralelizovat díky tomu, že její výpočet probíhá pro každý pixel vstupního obrázku zvlášť. Integrace s klíčovacím algoritmem probíhá stejně jako v předchozím případě (viz Algoritmus 2), avšak narozdíl od předchozí metody potřebuje funkce computeGradient volat funkci computeDifference, která se nachází přímo v klíčovacím algoritmu. V důsledku tohoto volání vzniká coupling algoritmu potlačení korony na algoritmus klíčování. Pokud však bude klíčovací algoritmus poskytovat volání této funkce, pak lze klíčovací algoritmy libovolně zaměňovat i přes toto provázání.

Možným nedostatkem tohoto řešení je, že funkce computeGradient počítá hodnoty α_0 pro okolní pixely zkoumaného bodu a tyto pak nejsou dále ukládány. Nastává tedy situace, při které se průhlednost každého pixelu vstupního obrázku počítá několikrát(v závislosti na velikost konvoluční matice při rozmazání a výpočtu gradientu). Pokud by tato nedokonalost v budoucnu působila problémy s výkonem celého procesu, pak lze výpočet hodnot průhlednosti optimalizovat za pomocí globálního pole, do kterého by algoritmus ukládal již vypočítané hodnoty. Zde pak nastává problém s přístupem do tohoto globálního pole a v případě paralelizování celého algoritmu by musela být řešena synchronizace zápisu do tohoto pole tak, aby nedocházelo ke kolizím.

3.2.3 Desaturace pixelů v oblastech s odstínem klíčovací barvy

Poslední algoritmus pro detekci a potlačení barevné korony bude vycházet z poznatků získaných v části 3.1.4 a bude tedy detekovat koronu na základě podobnosti odstínu barvy

korony s odstínem klíčovacího plátna. Potlačení korony bude realizováno stejně jako v předchozím případě, tedy desaturováním daných pixelů.

```
function spillSuppress(input image C_0)
    for each p ∈ C_0 do
        q = toHSV(C_k);
        p = toHSV(p);
        min = min(H_p, H_q);
        max = max(H_p, H_q);
        dist = min(abs(min − max),abs(min + 1 − max));
        if dist < i then
            suppressFactor = dist/i;
            S_p = suppressFactor * S_p;
        end
        p = toRGB(p);
    end
```

Algoritmus 4: Algoritmus odstranění korony na základě odstínu

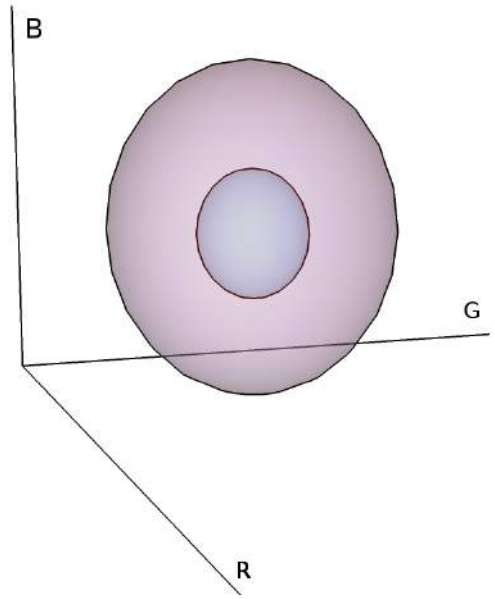
Při návrhu algoritmu 4 nastává problém s faktem, že odstín barvy v barevném modelu HSV reprezentuje úhel, proto nelze rozdíl dvou hodnot odstínu vypočítat prostým rozdílem. Dalším důležitým krokem je výpočet proměnné suppressFacor, která představuje údaj o tom, jak silně se má potlačit saturace daného pixelu. Pro tento algoritmus byl zvolen postup kdy se hodnota faktoru potlačení saturace pixelu lineárně interpoluje mezi hodnotami 0 a 1. Tím je dosaženo toho, že barva, která má stejný odstín jako klíčovací barva, je zcela desaturována a barvy, jejichž odstín je vzdálený o i a více od odstínu klíčovací barvy nejsou tímto algoritmem změněny.

Stejně jako v předchozích dvou případech, i zde je dobrá možnost paralelizace a jedinný vstup, který vyžaduje toto řešení od klíčovacího algoritmu je odstín klíčovací barvy, což znamená, že závislost na klíčovacím algoritmu je i zde velmi malá.

3.3 Návrh klíčovacího algoritmu

Jako algoritmus pro samotné klíčování obrazu byla zvolena úprava řešení, které představil Mishima a které bylo zde diskutováno v části 2.1.3.2. Tento postup byl zvolen z důvodu rychlosti tak, aby nebyly algoritmy pro kontrolu barevné korony omezeny výkonností klíčovacího algoritmu. Samozřejmě jak bylo dříve zmíněno, tak všechny navržené algoritmy pro potlačení korony, mohou pracovat s libovolným klíčovacím algoritmem.

Úprava, která byla zvolena, spočívá v tom, že na rozdíl od původního řešení jsou hexoctahedrony nahrazeny soustřednými koulemi. Algoritmus nejdříve v barevném prostoru RGB sestrojí kouli se středem v průměrné barvě klíčovacího pozadí tak, aby obsahovala všechny pixely daného vzorku pozadí. Dále je vytvořena druhá koule se stejným středem, která je naopak největší možná taková, aby neobsahovala žádný bod ze zvoleného vzorku popředí (viz obrázek 3.5). Volba vzorků popředí a pozadí je pro chod algoritmu velmi důležitá, protože pokud jsou vzorky zvoleny nevhodně, pak se může stát, že koule popředí bude mít menší poloměr než koule pozadí, v důsledku čehož nebude klíčování probíhat správně.



Obrázek 3.5: Znázornění koulí v RGB prostoru. Menší z koulí obsahuje všechny body pozadí, větší neobsahuje žádné body popředí

Dále již algoritmus pracuje stejně jako původní postup, takže pokud se zkoumaný pixel nachází uvnitř menší z koulí, pak je považován za pixel pozadí, podobně pokud se pixel nachází vně větší z koulí, tak je považován za pixel popředí. Hodnoty pixelů, ležících v prostoru mezi těmito dvěma koulemi, jsou lineárně interpolovány.

3.4 Shrnutí

Na základě analýzy, která byla provedena v části 3.1, byly navrženy tři algoritmy detekující a potlačující barevnou koronu, která vzniká při klíčování na zelené, resp. modré plátno. Řešení 3.1.2 je sice navrženo pouze pro klíčování na zeleném pozadí, avšak jednoduchou úpravou jej lze převést na klíčování na modrém plátně.

Při návrhu všech algoritmů se podařilo dodržet dobrou možnost paralelizace a rychlosti výpočtu, stejně jako nezávislost algoritmu potlačení korony na procesu klíčování. Tyto předpoklady budou dále testovány, aby bylo skutečně potvrzeno, že všechny algoritmy pracují dostatečně rychle na to, aby zvládly potlačovat koronu v reálném čase.

Dále byl navržen algoritmus pro samotné klíčování na monochromatickém pozadí, který je založen na řešení popsaném v 2.1.3.2. Tento postup byl zvolen pro svoji jednoduchost implementace a možnost paralelizace. Integrace algoritmů pro kontrolu barevné korony s klíčovacím algoritmem byla popsána v části 3.1.2.

Kapitola 4

Implementace

Tato část práce se zabývá popisem implementace testovací aplikace pro algoritmy detekce a potlačení barevné korony, navržené v části 3.1. Nejdříve budou popsány hlavní požadavky na vlastnosti aplikace a budou vyjmenovány a představeny technologie, které byly pro implementaci použity. Dále zde bude také popsána struktura aplikace a některé důležité body návrhu a implementace.

4.1 Požadavky na aplikaci

Aplikace, která bude sloužit pro testování dříve navržených algoritmů, musí být schopna otestovat všechny předpoklady, které byly ohledně těchto řešení vysloveny. Algoritmy by tedy měly být implementovány jako paralelní. Aplikace by měla být schopna načítat obrázky a video a tyto předávat klíčovacímu algoritmu, resp. algoritmům pro potlačení korony.

Protože tato aplikace je využívána pouze pro účely testování, není třeba vyvíjet složité uživatelské rozhraní, takže komunikace s uživatelem bude probíhat pouze pomocí příkazové řádky. Testování bude následně probíhat na operačním systému Windows, takže je třeba, aby aplikace neměla problém s během pod tímto systémem.

Pro správný chod klíčovacího algoritmu je dále třeba, aby aplikace umožňovala uživateli zvolit vzorek barvy popředí a pozadí. Výsledek procesu klíčování a kontroly korony, stejně jako výběr klíčovací barvy a barvy popředí bude zobrazen ve zvláštním okně.

4.2 Použité technologie

Testovací aplikace je napsána v jazyku C++. Tento jazyk byl zvolen kvůli tomu, že nabízí dobrý poměr rychlosti a pohodlí při implementaci, jelikož v něm lze programovat objektově. Dalším důvodem byla dobrá podpora OpenGL.

4.2.1 FreeGLUT

Pro implementaci byla dále použita knihovna FreeGLUT, která nabízí jednoduché rozhraní pro správu oken a vstupu od uživatele. FreeGLUT se také stará o překreslování oken a

sám volá callback funkce pro obsloužení událostí jako jsou například stisknutí kláves, pohyb myší a podobně.

Existuje mnoho OpenGL toolkitů jako např. SFML, nebo SDL, avšak pro účely této testovací aplikace byla zvolena nejjednodušší knihovna, která ale stále poskytuje všechny potřebné funkce pro vytvoření tohoto typu aplikace. Výhodou FreeGLUT je jeho nezávislost na použité platformě.

4.2.2 OpenCV

Jelikož FreeGLUT neposkytuje možnost načítání video souborů a fotografií, byla pro tento účel použita knihovna OpenCV. Tento framework poskytuje široké rozhraní funkcí a objektů pro práci s kamerami, video soubory a fotografiemi. Další výhodou OpenCV je jednoduchost používání a to do takové míry, že otevření, případně uložení videa, resp. fotografie je provedeno pouze jedním voláním metody.

Stejně jako FreeGLUT je i OpenCV multiplatformní, takže přenesení aplikace z Windows na jiný operační systém by neměl být problém.

4.3 Struktura aplikace

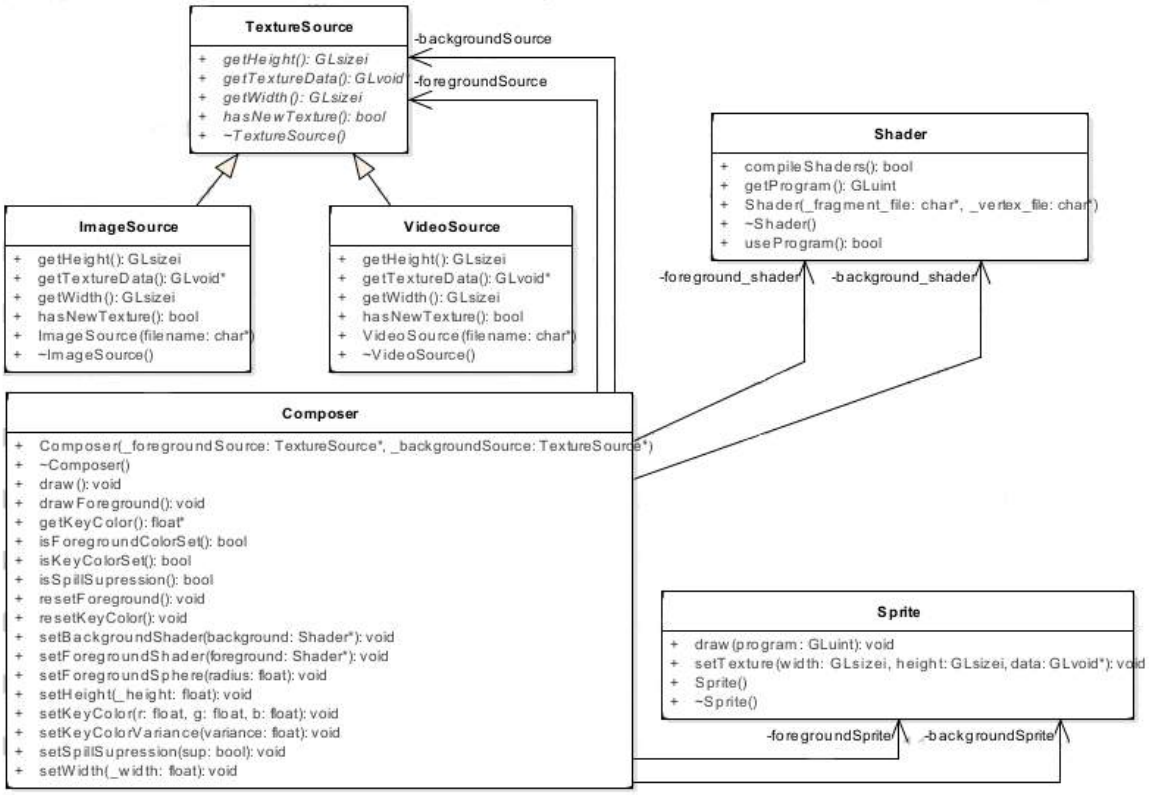
Strukturu aplikace popisuje obrázek 4.1. Hlavní komponentou celé aplikace je Composer, který se stará o správné vykreslování pozadí a popředí, spravuje GLSL shadery a řídí načítání textur ze zdrojů. Composer je z vnějšku ovládán logikou aplikace, která reaguje na události volané knihovnou FreeGLUT a jsou mu předávány všechny potřebné parametry pro běh klíčování, jako je klíčovací barva, současný stav aplikace a podobně. Samotný klíčovací algoritmus a algoritmy pro detekci a potlačení korony jsou pak implementovány pomocí GLSL shaderu, což znamená že aplikace se stará pouze o nahrání potřebných dat na grafickou kartu a zbytek výpočtů je již proveden ve fragment shaderu.

Načítání textur je zde řešeno komponentou TextureSource, která má dvě možné varianty a to ImageSource, který načítá fotografie ze souboru, a VideoSource, který se stará o zpracování vstupního videa.

Třída Shader představuje abstrakci nad GLSL shader programem, což znamená, že se stará o načítání vertex a fragment shaderů ze souborů, jejich kompilaci a linkování do shader programu.

Samotný proces klíčování probíhá tak, že se najdřív vykreslí Sprite pozadí, který je beze změny, poté se provede vykreslení popředí. Fragment shader spočítá výsledný obraz popředí a do alpha kanálu uloží vypočítanou hodnotu α_0. OpenGL se poté pomocí blendingu postará o výslednou kompozici obrazu.

Pokud je aktivován mód výběru klíčovací barvy, nebo mód výběru vzorku popředí, pak není vykreslováno pozadí a popředí je vykresleno s hodnotou $\alpha = 0$. Toto zajistí možnost opětovného výběru barev, aniž by obraz, z kterého je vzorek vybírán, byl ovlivněn procesem klíčování, nebo potlačení barevné korony.



Obrázek 4.1: Class diagram testovací aplikace

4.4 Důležité body návrhu a implementace

4.4.1 Výběr vzorků barev

Důležitou částí aplikace je výběr vzorků popředí a pozadí. Pokud by se výběr omezil jen na jeden pixel (například výběr kliknutím myši), pak by mohlo docházet k tomu, že by se vybraný pixel mohl příliš lišit od skutečného odstínu klíčovací barvy. Další problém by nastal při vytváření koulí v prostoru RGB, jak bylo popsáno v 3.3, protože algoritmus potřebuje dostatečně velikou množinu vzorků pozadí, aby mohl pracovat korektně.

V testovací aplikaci je výběr vzorků realizován tažením myši po aktivaci módu výběru klíčovací barvy, nebo módu výběru vzorku popředí. Výsledný vzorek je pak přečten za pomocí funkce glReadPixels a dále zpracováván.

Při čtení pixelů pomocí glReadPixels je čtená oblast definována pomocí levého dolního rohu, výšky a šířky obdélníku, který má být přečten. Tento fakt je nutné vyřešit, jelikož tažení myší může začít a skončit v libovolném rohu vybíraného obdélníku.

Procedura čtení pixelů z uživatelem vybrané oblasti je zobrazena v 4.1. Jedná se o část obslužné funkce stisknutí tlačítek myši, kterou definuje a volá knihovna FreeGLUT. Protože je funkce volána při každém stisku tlačítka myši, je třeba uložit pozici startovního kliknutí

do proměnné aplikace a při uvolnění tlačítka pomocí této hodnoty vypočítat parametry zvoleného obdélníku.

```
Point start;
int xsize,ysize;
if(mstate == GLUT_DOWN){
  state.startDrag.x= x;
  state.startDrag.y = y;
  }
else if(mstate == GLUT_UP){
  start.x = std::min(x,state.startDrag.x);
  start.y = std::min(y,state.startDrag.y);
  xsize = std::abs(x-state.startDrag.x);
  ysize = std::abs(y-state.startDrag.y);
  const int size = xsize*ysize*3;
  GLubyte *pixels =new GLubyte[size];
  glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
  state.composer->drawForeground();
  glPixelStorei(GL_PACK_ALIGNMENT, 1);
  glReadPixels(
        start.x,
        start.y,
        xsize,
        ysize,
        GL_RGB,GL_UNSIGNED_BYTE, pixels);
  /* Process pixels */
  free(pixels);
  glutPostRedisplay();
}
```

Ukázka zdrojového kódu 4.1: Ukázka procedury pro čtení pixelů z obrazovky

4.4.2 Zpracování vzorků pozadí a popředí

Po vybrání části vstupního obrazu (viz 4.4.1), následuje proces zpracování těchto vzorků, zobrazený v ukázce kódu 4.2. V této ukázce je zobrazena část funkce pro zpracování vybraného vzorku pozadí (zpracování výběru popředí funguje analogicky).

Nejdříve je ze vzorku vypočtena průměrná hodnota klíčovací barvy, poté je pomocí Eukleidovské vzdálenosti nalezen pixel z vybraného vzorku, který se v barevném prostoru RGB vyskytuje nejvíce daleko od průměrné klíčovací barvy vypočtené dříve. Vzdálenost těchto dvou bodů, tedy průměrné klíčovací barvy a barvy s největší odchylkou od barvy klíčovacího pozadí, je poté označena jako průměr koule se středem v bodě C_k, tedy koule, která by měla obsahovat všechny body pozadí.

Funkce pro výpočet koule popředí je napsána v podstatě stejně, avšak v tomto případě již není třeba počítat průměrnou barvu pozadí (pokud není průměrná barva pozadí zvolena,

pak výpočet nemůže proběhnout) a namísto hledání bodu s největší vzdáleností je nalezen bod s nejmenší vzdáleností.

```
std::vector<float> average = std::vector<float>(3);
float length = 0.0f;
for(int i =0; i<size;i=i+3){
  average[0]=(average[0]+(float)pixels[i])/2.0f;
  average[1]=(average[1]+(float)pixels[i+1])/2.0f;
  average[2]=(average[2]+(float)pixels[i+2])/2.0f;
}
for(int i =0; i<size;i=i+3){
  float dx = std::pow(average[0]-pixels[i],2.0f);
  float dy = std::pow(average[1]-pixels[i+1],2.0f);
  float dz = std::pow(average[2]-pixels[i+2],2.0f);
  float PixLen = std::sqrt(dx+dy+dz);
  if(PixLen>length){
    length = PixLen;
  }
}
```

Ukázka zdrojového kódu 4.2: Ukázka zpracování vzorků pozadí

Kapitola 5

Testování

V této kapitole bude provedeno testování dříve navržených a implementovaných metod pro detekci a odstranění barevné korony. Testování bude provedeno jak na statických fotografiích, tak na video snímcích, které se liší v podmínkách osvětlení, různorodostí objektů v popředí scény a rychlostí pohybu těchto objektů.

5.1 Testovací data

5.1.1 Fotografie

Pro testování na statických snímcích budou použity stejné fotografie, které byly použity pro analýzu chování korony v částí 3.1. Tyto snímky byly vybrány tak, aby reprezentovaly různé typy povrchů a osvětlení a aby zahrnovaly co nejširší paletu situací, které mohou při klíčování na zelené, resp. modré plátno nastat.

(a) Sheep (b) Person (c) Troll

Obrázek 5.1: Fotografie použité při testování navržených metod kontroly barevné korony

Tabulka 5.1 popisuje fotografie z obrázku 5.1. Jak již bylo řečeno, tak tyto fotografie byly pořízeny s důrazem na vznik barevné korony a vytvoření co nejméně příznivých podmínek pro klíčování. Za normálních okolností by osvětlení a celková kompozice byla upravena tak, aby mohl klíčovací algoritmus pracovat co nejlépe.

Název	Popis scény	Testované parametry
Sheep 5.1a	Plyšová ovce před zeleným pozadím. Objekt blízko klíčovacího plátna.	Silná barevná korona, Neostrý přechod mezi popředím a pozadím, Jemné kontury objektu
Person 5.1b	Osoba stojící před zeleným pozadím.	Špatné nasvícení, nejednotná barva klíčovacího plátna, malý rozdíl odstínu barvy popředí a pozadí, vlasy
Troll 5.1c	Figurka stojící na zeleném pozadí	Hladké povrchy na objektu popředí, silná barevná korona, jemné vlasy, poloprůhledné části objektu v popředí

Tabulka 5.1: Popis testovacích fotografií

5.1.2 Videosnímky

Dále budou algoritmy testovány na čtyřech video záběrech, které se liší v typech objektů popředí, rychlosti jejich pohybu, osvětlení a hloubkou ostrosti. Podle zadání měly být algoritmy otestovány na šesti záběrech, ale díky komplexnosti jednotlivých testovacích snímků bude dostačující, použít pouze čtyři videa. Během natáčení byla dynamicky měněna clona, hloubka ostrosti, osvětlení i přiblížení, čímž bylo docíleno toho, že takové záběry otestují co největší množinu situací, které mohou při klíčování nastat.

(a) Fan

(b) Ball

(c) Person

(d) Troll

Obrázek 5.2: Ukázky videosnímků použitých pro testování navržených algoritmů

V tabulce 5.2 jsou vypsány jednotlivé parametry testovacích videí. Můžeme si všimnout, že videa skutečně pokrývají širokou paletu vlastností a testovacích parametrů.

Název	Popis scény	Testované parametry
Fan 5.2a	Točící se větrník před zeleným plátnem	Korona na extrémně lesklém povrchu, rychlý pohyb, změna rychlosti pohybu, změna přiblížení
Ball 5.2b	Míč skákající před zeleným pozadím	Rychlý pohyb objektu popředí, matný povrch zasažený koronou
Person 5.2c	Osoba pohybující se před zeleným pozadím	Odstíny kůže, vlasy, středně rychlý a pomalý pohyb objektu popředí, špatné osvětlení
Troll 5.2d	Figurka s vlnícími se vlasy	Jemné vlasy, lesklý povrch velmi zasažený koronou, změna hloubky ostrosti, změna osvětlení a clony, změny přiblížení

Tabulka 5.2: Popis testovacích video záběrů

5.2 Návrh testovacích metod

Testování algoritmů pro odstranění barevné korony je vcelku problematický úkol. Existuje mnoho způsobů jak testovat klíčovací algoritmy na základě jejich přesnosti, nebo rychlosti, zejména v oblasti natural image mattingu existují automatické nástroje pro testování a porovnávání jednotlivých řešení (http://www.alphamatting.com/). Bohužel tyto způsoby testování neřeší problém barevné korony.

5.2.1 Subjektivní posudek

Protože pro tento typ úlohy v podstatě nelze nalézt referenční řešení, které by mohlo být prohlášeno za ideální výsledek, tak jedním ze způsobů zkoumání kvality výsledků může být subjektivní posudek výsledného obrazu. Abychom byli schopni určit naprosto korektní řešení, museli bychom mít k dispozici kompletní informace o světelných podmínkách scény, stejně jako o poloze světel a objektů v ní a i poté by byl výpočet výsledného obrazu velmi časově a výpočetně náročný.

5.2.2 Porovnání s existujícím řešením

Další možností testování je porovnání s výsledky jiného algoritmu pomocí rozdílu pixelů. Pro tento účel je vhodné použít software, který je v dnešní době využíván k profesionálnímu zpracování obrazu, určeného ke klíčování, jako je například Primatte. Bohužel tento způsob testování nemusí podávat korektní výsledky právě z důvodu toho, že každý z algoritmů pro potlačení korony může tento problém řešit jiným způsobem. Může se tedy stát, že všechny výsledky budou pro člověka přirozené a nebudou působit rušivě, a i přesto budou testem vyhodnoceny jako chybné.

5.2.3 Test rychlosti

Posledním aspektem, který bude v této kapitole testován, je rychlost samotného výpočtu výsledku. Zde je opět porovnání s jinými algoritmy problematické, protože k implementaci

řešení navržených v této práci byl použit jazyk GLSL a tak můžeme očekávat, že rychlost algoritmů bude v důsledku paralelizace a výpočtu na grafické kartě o několik řádů větší než u ostatních softwarů.

Všechny výše zmíněné metody testování budou provedeny jak na statických snímcích, tak i na videu. Jelikož předmětem této práce je pouze kontrola barevné korony, nebudou fotografie ani videa pro účely testování klíčovány, ale budou nad nimi pouze spuštěny navržená řešení a výsledky po klíčování budou pak zobrazeny jen pro účely ukázky.

5.3 Test rychlosti

K testování rychlosti byl použit debugovací software pro OpenGL gDebugger. Za pomocí tohoto programu lze zjistit průměrnou zobrazovací frekvenci dané OpenGL aplikace. Protože není framerate v testovací aplikaci nijak omezen, tak tato hodnota ve skutečnosti udává rychlost výpočtu.

Tento test není prováděn zvlášť na fotografiích a zvlášť na videu, ale pouze na fotografiích, protože načítání videa je obecně časově náročnější operace, která jistě ovlivní negativně celkový framerate. Při testování rychlosti na fotografii tak bude načten pouze jeden snímek a framerate bude poté odpovídat skutečné rychlosti výpočtu detekce korony bez časově náročných operací, jako je načítání z disku, nebo nahrávání textur na grafickou kartu.

	Průměrné FPS		
Fotografie	**Omezení barevného kanálu**	**Detekce gradientem**	**Detekce odstínem**
Sheep	2697	869	2221
Person	1361	469	1215
Troll	2704	871	2249

Tabulka 5.3: Přehled hodnot FPS při kontrole barevné korony na testovacích fotografiích (vyšší FPS = rychlejší algoritmus)

V tabulce 5.3 jsou zobrazeny výsledky testu rychlosti navržených algoritmů pro detekci a odstranění korony v programu gDEBuger. Nejrychlejším algoritmem je bezpochyby odstranění pomocí omezení zeleného kanálu barvy, protože tento výpočet se provádí pro každý pixel pouze jednou a na rozdíl od detekce odstínu klíčovací barvy nepotřebuje převod barvy do jiného barevného prostoru.

Nejhorších výsledků v tomto testu dosahuje algoritmus založený na výpočtu gradientu odchylky od klíčovací barvy. Tento výsledek je očekávaný, protože pro každý pixel je bráno v úvahu jeho okolí (v tomto případě 3x3 pixely). Jedná se ovšem pouze o konstantní zpomalení závislé pouze na velikosti používané konvoluční matice, takže i tento algoritmus pracuje na grafické kartě mnohem rychleji, než ostatní běžně dostupná řešení.

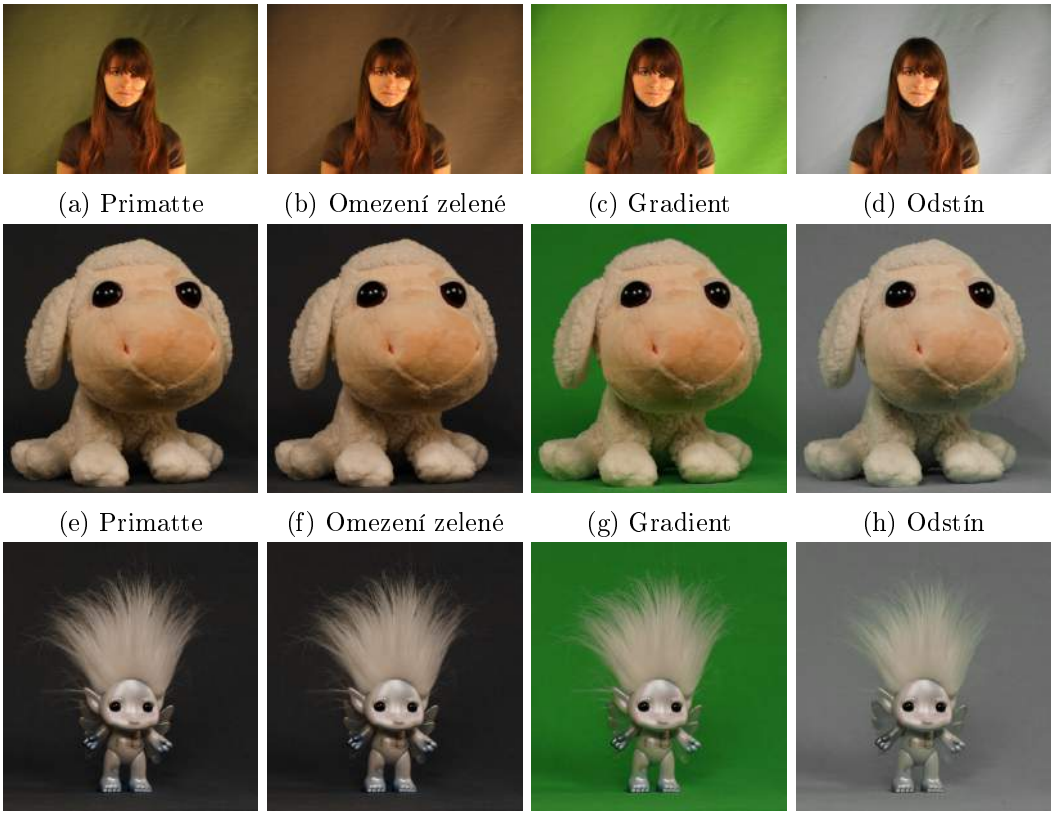
5.4 Testování na statických snímcích

Testování na statických snímcích se liší od testu na video záběrech v tom, že na testovacích fotografiích není patrné rozmazání pohybem. Jinak jsou testy podobné, protože algoritmus

vyhodnocuje výslednou barvu pro každý snímek zvlášť, takže odstranění korony ve videu je v podstatě odstranění korony na sérii fotografií.

5.4.1 Subjektivní test

V tomto testu budou jednotlivá řešení porovnávána na základě několika aspektů, jako je míra odstranění korony, celkový dojem a podobně. Výsledky budou porovnány s komerčním nástrojem pro klíčování na zelené plátno Primatte 5.0, konkrétně s jeho součástí Spill killer.



(a) Primatte (b) Omezení zelené (c) Gradient (d) Odstín

(e) Primatte (f) Omezení zelené (g) Gradient (h) Odstín

(i) Primatte (j) Omezení zelené (k) Gradient (l) Odstín

Obrázek 5.3: Ukázky výstupu algoritmů pro potlačení korony

Na obrázku 5.3 jsou zobrazeny výstupy navržených řešení pro kontrolu barevné korony spolu s výsledky, které byly dosaženy při použití komerční software Primatte 5.0. Můžeme si všimnout, že algoritmus, který potlačuje zelenou složku barvy, má velice podobné výsledky jako Primatte Spill killer, dá se tedy předpokládat, že tento software používá stejné, nebo podobné řešení tohoto problému.

Na druhou stranu, algoritmus využívající výpočet gradientu podává jen málo znatelné výsledky (pouze okraje objektů, které dokonce nejsou při takto výrazném zmenšení patrné). Výsledky jsou vyhodnoceny v následující tabulce.

Tabulka 5.4 zobrazuje vyhodnocení výsledků dosažených algoritmy pro odstranění korony spolu se softwarem Primatte Spill killer. Nejlepší výsledky podává právě Primatte, který má

Název	Algoritmus	Odstranění korony	Poškození barev	Celkový dojem
Sheep	Primatte Spill killer	10/10	9/10	9/10
	Omezení zelené složky	9/10	8/10	9/10
	Gradient	2/10	8/10	2/10
	Odstín klíčovací barvy	8/10	7/10	8/10
Person	Primatte Spill killer	4/10	8/10	5/10
	Omezení barevné složky	6/10	9/10	6/10
	Gradient	3/10	2/10	2/10
	Odstín klíčovací barvy	7/10	8/10	7/10
Troll	Primatte Spill killer	10/10	10/10	10/10
	Omezení barevné složky	10/10	9/10	10/10
	Gradient	1/10	9/10	2/10
	Odstín klíčovací barvy	7/10	10/10	6/10

Tabulka 5.4: Subjektivní posouzení výsledků testování na statických snímcích. Každý parametr byl ohodnocen na stupnici od 1 do 10, kde 10 je nejvyšší hodnota

nejlepší výsledky na všechny testované vstupy. Jediný algoritmus, který dosahuje podobné kvality, je omezení zeleného kanálu, který dokonce v případě 5.3b Primatte předčil. Řešení využívající odstín klíčovací barvy sice na rozdíl od dvou dříve zmíněných postupů odstraňuje koronu jiným způsobem, ale jeho výstup je také kvalitní a záleží na osobních preferencích uživatele, které z těchto řešení odstraňuje koronu nejlépe.

Jak již bylo zmíněno, detekce korony gradientem v tomto případě zcela selhala, ať už se jedná o samotnou detekci korony, tak i v postupu jakým je tato potlačena. Na výstupu se na okrajích objektů objevují silně desaturované plochy, které ale neovlivňují původní barvy objektů popředí, takže po provedení klíčování budou nejspíš odstraněny.

5.4.2 Testování rozdílem výstupů

Tento test spočívá ve vyhodnocení odchylek výstupů zde navržených algoritmů pro detekci a odstranění korony od referenčního řešení. Jako referenční řešení byl zvolen výstup z programu Primatte 5.0 Spill killer. Tento software je v dnešní době velice oblíbený a je často využíván pro klíčování na monochromatické pozadí.

Název	Algoritmus	Nejmenší odchylka	Největší odchylka	Průměrná odchylka
Person	Omezení zeleného kanálu	0	22	5.074
	Gradient	0	87	19.000
	Detekce odstínu	0	100	62.171
Sheep	Omezení zeleného kanálu	0	8	0.877
	Gradient	0	82	13.201
	Detekce odstínu	0	90	37.890
Troll	Omezení zeleného kanálu	0	8	0.848
	Gradient	0	78	21.483
	Detekce odstínu	0	96	60.173

Tabulka 5.5: Odchylky výstupů od referenčního řešení

Tabulka 5.5 zobrazuje výsledky výpočtů odchylek na výstupech testovaných algoritmů. Odchylky jsou počítány na průměr barevných kanálů, tedy

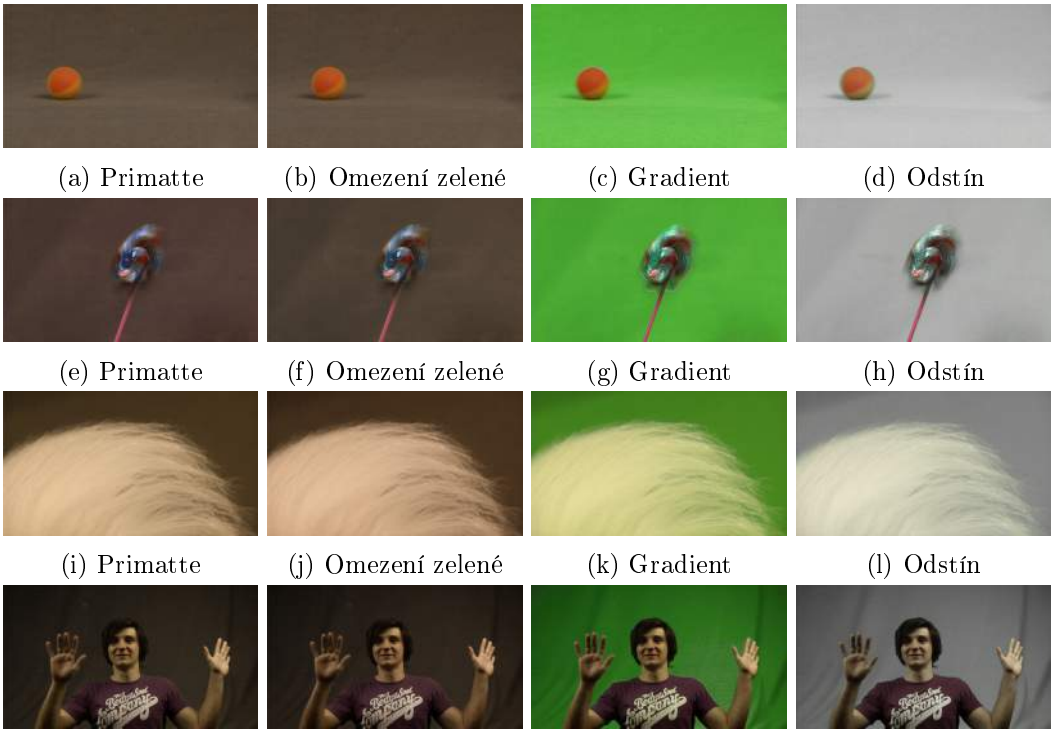
$$mean(x, y) = \frac{R(x, y) + G(x, y) + B(x, y)}{3}$$

Nejmenší rozdíly jsou podle očekávání zaznamenány u řešení, které potlačuje zelenou složku barvy, naopak největší odchylky generuje algoritmus, založený na kontrole odstínu klíčovací barvy. Je důležité si uvědomit, že tento výsledek nemusí nutně znamenat, že by toto řešení bylo nevyhovující. Například test na základě subjektivního posudku tento algoritmus označil jako korektní.

5.5 Testování na videu

5.5.1 Subjektivní test

Subjektivní test bude probíhat stejně jako tomu bylo u testování fotografií. Každý aspekt výstupu algoritmů bude posouzen na stupnici od 1 do 10 a tyto pak budou zobrazeny v tabulce.



(a) Primatte (b) Omezení zelené (c) Gradient (d) Odstín

(e) Primatte (f) Omezení zelené (g) Gradient (h) Odstín

(i) Primatte (j) Omezení zelené (k) Gradient (l) Odstín

(m) Primatte (n) Omezení zelené (o) Gradient (p) Odstín

Obrázek 5.4: Ukázky výstupu algoritmů pro potlačení korony pro video vstupy

Výstupy zde navržených řešení jsou zobrazeny na obrázku 5.4. Můžeme si všimnout, že výsledky jsou podobné, jako tomu bylo u fotografií. Opět nejlepší výsledky podává Primatte spolu s potlačením zelené složky a desaturací v místech s odstínem klíčovací barvy. Detekce gradientu opět naprosto selhala a jediný rozumný výsledek dosahuje na obrázku 5.4o, kde je korona správně detekována v oblasti vlasů.

Název	Algoritmus	Odstranění korony	Poškození barev	Celkový dojem
Ball	Primatte Spill killer	10/10	10/10	10/10
	Omezení zelené složky	10/10	10/10	10/10
	Gradient	1/10	3/10	1/10
	Odstín klíčovací barvy	3/10	10/10	5/10
Fan	Primatte Spill killer	10/10	8/10	9/10
	Omezení barevné složky	10/10	9/10	9/10
	Gradient	0/10	2/10	1/10
	Odstín klíčovací barvy	7/10	9/10	8/10
Person	Primatte Spill killer	7/10	8/10	8/10
	Omezení barevné složky	8/10	8/10	8/10
	Gradient	4/10	8/10	4/10
	Odstín klíčovací barvy	9/10	7/10	8/10
Troll	Primatte Spill killer	10/10	6/10	8/10
	Omezení barevné složky	10/10	7/10	8/10
	Gradient	0/10	9/10	4/10
	Odstín klíčovací barvy	10/10	10/10	10/10

Tabulka 5.6: Subjektivní posouzení výsledků testování na videu. Každý parametr byl ohodnocen na stupnici od 1 do 10, kde 10 je nejvyšší hodnota

Tabulka 5.6 sumarizuje hodnocení jednotlivých algoritmů. Zde je jednoznačně nejlepší výsledek dosažen na obrázku 5.4l, kde je nejen barevná korona přesně odstraněna, ale jsou i zachovány původní barvy objektu popředí. Opět velice kvalitní výsledky podává Primatte a omezení zelené složky, i když si na výstupech můžeme všimnout jistého posunu barev a tím pádem i zkreslení celého výsledku.

Gradient se opět neosvědčil a u všech testovaných videí, s výjimkou osoby před zeleným pozadím, kde byl schopen do jisté míry detekovat a odstranit koronu v oblasti vlasů, naprosto propadl.

5.5.2 Testování rozdílem výsledků

Tento test opět probíhá podobným způsobem jako tomu bylo u fotografií. Odchylky jsou počítány stejným způsobem, avšak tentokrát jsou vyhodnoceny pro každý snímek video záběru a poté je vypočítán průměr těchto odchylek.

Tabulka 5.7 zobrazuje výsledky tohoto testu. Stejně jako tomu bylo u testování fotografií, tak i zde nejlepší výsledky podává algoritmus pro omezení zelené složky barvy. I v tomto testu nalezneme nejvyšší hodnoty odchylek u algoritmu detekujícího odstín klíčovací barvy a u detekce korony gradientem.

Je třeba opět brát v úvahu, že tento typ testu porovnává výstupy na základě podobnosti s referenčním řešením, což nemusí vždy vést ke korektnímu výsledku, protože pokud algoritmus barevnou koronu potlačuje jiným způsobem, může dojít k jinému zabarvení objektů popředí a pozadí, čímž vznikají velké odchylky od referenčního řešení.

Název	Algoritmus	Nejmenší odchylka	Největší odchylka	Průměrná odchylka
Ball	Omezení zeleného kanálu	0	12	0.933
	Gradient	0	97	36.709
	Detekce odstínu	7	121	108.827
Fan	Omezení zeleného kanálu	0	18	1.951
	Gradient	0	101	35.336
	Detekce odstínu	0	123	100.219
Person	Omezení zeleného kanálu	0	15	1.197
	Gradient	0	89	17.193
	Detekce odstínu	0	92	48.777
Troll	Omezení barevného kanálu	0	10	1.600
	Gradient	0	57	17.142
	Detekce odstínu	0	92	47.789

Tabulka 5.7: Odchylky video výstupů navržených algoritmů od referenčního řešení

5.6 Shrnutí

V této kapitole byla otestována řešení problému barevné korony navržená v části 3.1. Jednoznačně nejlepších výsledků dosáhl ve všech testech algoritmus potlačující zelenou složku barvy, který podával nejen nejlepší výkon co se týče rychlosti výpočtu, ale i kvality výstupu.

Řešení detekující koronu výpočtem gradientu klíčovací barvy naopak neobstál ani v jednom z testů. Co se týče rychlosti tohoto řešení, lze jej považovat stále za rychlejší než většina dostupných řešení, avšak jeho výsledky co se týče kvality výstupu a schopnosti detekovat a odstranit koronu jsou velmi neuspokojivé.

Poslední řešení, tedy algoritmus detekující koronu na základě odstínu klíčovací barvy v barevném prostoru, HSV podává ve všech testech velice solidní výsledky a některé výstupy tohoto řešení jsou dokonce lepší než výstupy komerčního software Primatte Spill killer. Pro tento postup je charakteristická velmi dobrá přesnost detekce korony, v důsledku čehož výsledný obraz netrpí degradací barev a celkový dojem je tak velice pozitivní

Kapitola 6

Závěr

V této práci byla po provedení rešerše na téma klíčování na zelené, resp. modré pozadí a potlačení barevné korony provedena analýza několika klíčovacích záběrů, na základě které byly následně navrženy tři postupy řešící tento problém. Dvě z těchto řešení (Omezení barevného kanálu a detekce na základě barevného odstínu) skutečně řeší tento problém a na základě testování lze prohlásit, že jsou porovnatelná s dostupnými řešeními. Algoritmus, který detekuje koronu pomocí gradientu se však během testování příliš neosvědčil.

Všechna představená řešení splňují cíle práce zmíněné v 2.3 ať se jedná o rychlost výpočtu a možnost paralelizace, nebo jejich nezávislost na zvoleném klíčovacím algoritmu. Při návrhu algoritmu detekujícího koronu gradientem se však nepodařil splnit základní cíl, a to samotné odstranění korony. Algoritmus sice na malé množině vstupů je schopen koronu detekovat, například v oblasti vlasů, avšak v kombinaci se zvoleným způsobem potlačení korony, jsou jeho výsledky silně nedostačující.

6.1 Možná vylepšení

Jednou z možných úprav, které by mohly zlepšit výsledky představených řešení, je použití jiné interpolace u potlačování korony v algoritmu detekujícím koronu na základě odstínu barvy. V současné době je tento problém vyřešen lineární interpolací mezi hodnotami 0 a 1.

Další úpravou může být použití jiného barevného prostoru pro detekci korony, jakým je například prostor CIE Lab, který je navržen tak, aby odpovídal lidskému vnímání barev, což by mohlo mít pozitivní vliv na chod navržených algoritmů.

Všechna řešení byla navržena jako automatická retuš, která příliš nevyžaduje vstup od uživatele. Tento požadavek je velice omezující a při jeho dodržení má algoritmus velmi málo informací o vstupním obrazu. Do budoucna by tedy bylo vhodné od tohoto požadavku upustit a více využít informace poskytnuté uživatelem.

Literatura

[1] Alpha Matting Evaluation Website, 2014. Dostupné z: <http://www.alphamatting.com/>.

[2] Http://www.merriam-webster.com/, 2014. Dostupné z: <http://www.merriam-webster.com/dictionary/hexoctahedron>.

[3] CHUANG, Y.-Y. et al. A Bayesian approach to digital matting. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, 2, s. II–264–II–271 vol.2, 2001. doi: 10.1109/CVPR.2001.990970.

[4] DUPONT, J. – DESCHENES, F. Toward a Realistic Interpretation of Blue-spill for Blue-screen Matting. *The 3rd Canadian Conference on Computer and Robot Vision (CRV'06)*. 2006, s. 33–33. Dostupné z: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1640388>.

[5] FORSYTH, D. – ZISSERMAN, A. Mutual illumination. *Proceedings CVPR '89: IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 1989, s. 466–473. Dostupné z: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=37889>.

[6] GRUNDHöFER, A. et al. Color invariant chroma keying and color spill neutralization for dynamic scenes and cameras. *The Visual Computer*. 2010, 26, 9, s. 1167–1176. ISSN 0178-2789. doi: 10.1007/s00371-010-0464-8. Dostupné z: <http://dx.doi.org/10.1007/s00371-010-0464-8>.

[7] HUANG, M. – LIU, F. – WU, E. A GPU-based Matting Laplacian Solver for High Resolution Image Matting. *Vis. Comput.* June 2010, 26, 6-8, s. 943–950. ISSN 0178-2789. doi: 10.1007/s00371-010-0435-0. Dostupné z: <http://dx.doi.org/10.1007/s00371-010-0435-0>.

[8] LEVIN, A. – LISCHINSKI, D. – WEISS, Y. A Closed-Form Solution to Natural Image Matting. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*. Feb 2008, 30, 2, s. 228–242. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1177.

[9] MISHIMA, Y. Soft edge chroma-key generation based upon hexoctahedral color space, October 11 1994. Dostupné z: <https://www.google.com/patents/US5355174>. US Patent 5,355,174.

[10] SMITH, A. R. – BLINN, J. F. Blue screen matting. *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques - SIGGRAPH '96*. 1996, s. 259–268. Dostupné z: <`http://portal.acm.org/citation.cfm?doid=237170.237263`>.

[11] VLAHOS, P. Electronic composite photography with color control, February 8 1977. Dostupné z: <`https://www.google.com/patents/US4007487`>. US Patent 4,007,487.

[12] VLAHOS, P. Comprehensive electronic compositing system, August 10 1982. Dostupné z: <`https://www.google.com/patents/US4344085`>. US Patent 4,344,085.

Příloha A

Obsah přiloženého CD

- **dokumentace**
  Doxygen programátorská dokumentace k testovací aplikaci v html formátu.
- **exe**
  Spustitelné soubory. Aplikace je přeložena v debug verzi pro 32bit OS Windows, pro spuštění na jiném systému je třeba znovu přeložit (viz src)
- **pdf**
  Text bakalářské práce ve formátu pdf
- **src**
  Zdrojové kódy
  - **LaTeX**
    Zdrojové kódy textu ve formátu LaTeX.
  - **src**
    Solution z programu Visual Studio 2012 + zdrojové kódy aplikace.
- **media**
  - **images**
    Fotografie použité k testování.
  - **video**
    Videa použité pro testování ve formátu MPEG.