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 MULTMEDIA

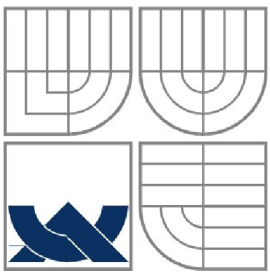
AUTOMATICKÁ TVORBA EFEKTU SVĚTELNÉHO MEČE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

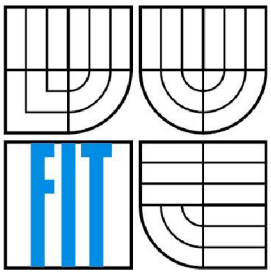
AUTOR PRÁCE
AUTHOR

ONDŘEJ VAGNER

BRNO 2010



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 MULTMEDIA

AUTOMATICKÁ TVORBA EFEKTU SVĚTELNÉHO MEČE

AUTOMATIC CREATION OF LIGHTSABRE EFFECTS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

ONDŘEJ VAGNER

VEDOUCÍ PRÁCE
SUPERVISOR

ING. PAVEL ŽÁK

BRNO 2010

Abstrakt

Efekt světelného meče patří pravděpodobně k nejznámějším filmovým efektů. Jeho vytváření je často velmi časově náročné. Tato práce si klade za cíl automatizaci procesu tvorby efektu světelného meče ve videosekvenci. Klíčovými elementy popsané metody jsou detekce imitace a meče a její následné nahrazení požadovaným efektem.

Abstract

The lightsabre effect is probably one of the most famous movie effects. Its creation is usually done manually which is time consuming. This paper presents an original approach of automatic lightsabre effect creation in the video. Key element of the entire method is detection of sword imitation in the video sequence and its replacement with desired effect.

Klíčová slova

Světelný meč, tvorba efektu světelného meče, detekce barevného objektu, barevná segmentace, kostra objektu.

Keywords

Lightsabre, creation of lightsabre effects, detection of color object, color segmentation, skeletonization.

Citace

Ondřej Vagner: Automatická tvorba efektu světelného meče, bakalářská práce, Brno, FIT VUT v Brně, 2010

Automatická tvorba efektu světelného meče

Prohlášení

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

……
Ondřej Vagner
17.5.2010

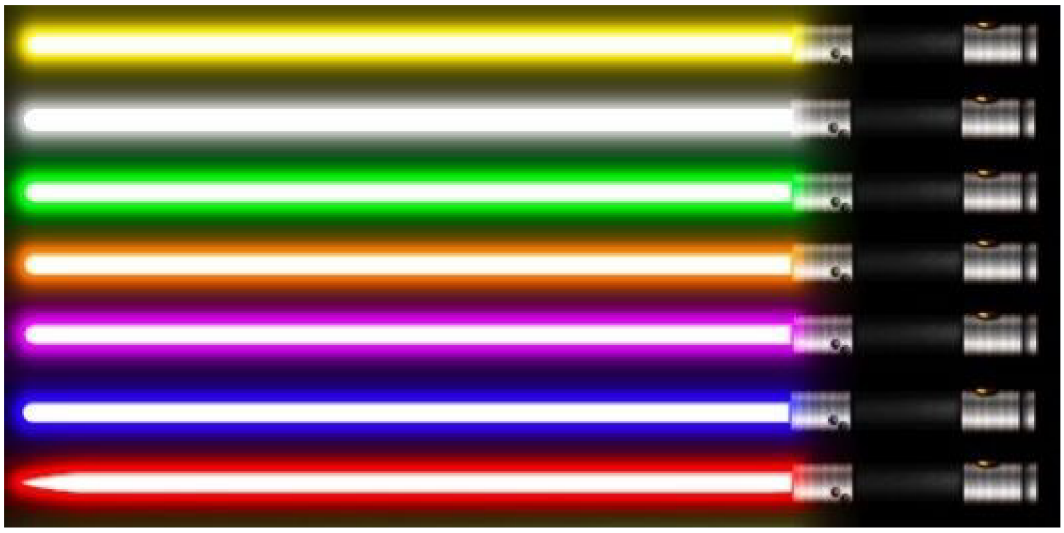
Poděkování

Rád bych poděkoval svému vedoucímu Ing. Pavlu Žákovi za cenné rady při tvorbě a zpracování této bakalářské práce.

Obsah

1 Úvod

Světelný meč je jedna z nejvýjimečnějších zbraní ze světa science-fiction. Pochází z vesmíru Hvězdných válek[1] a je znám i mimo komunitu fanoušků tohoto fenoménu. Jedná se o futuristickou verzi klasického meče, která má místo kovové čepele proud jasně zářící bílé energie obklopené barevnou korónou. Samotný meč tvoří jen asi 30cm dlouhý kovový jílec, který ukrývá zařízení emitující energetický paprsek čepele [8]. Nejčastěji používanými barvami čepelí jsou modrá, zelená a červená, ale vyskytují se i jiné barvy (Obrázek 1).



Obrázek 1: Běžné barvy a tvary čepelí světelných mečů, převzato z [8]

Existuje mnoho různých návodů a postupů [1][2] pro vytvoření efektu čepele světelného meče. Problém s těmito postupy nastává v momentu kdy si chce uživatel vytvořit tento efekt ve videosekvenci. V tomto případě je nutné zpracovat celou videosekvenci snímek po snímku. Zmíněný postup je velmi pracný, zvláště když si uvědomíme že jedna vteřina videosekvence je většinou tvořena 25 až 30 snímky.

Tato práce si klade za cíl vytvoření aplikace, která bude automaticky vkládat efekty čepelí světelných mečů do videosekvencí. Na rozdíl od výše popsaných postupů bude naše aplikace provádět všechny úkony, jako je např. nalezení tyče představující čepel světelného meče a vytvoření výsledného efektu zcela automaticky.

1 První film Hvězdných válek vznikl v roce 1977, jejich autorem a duchovním otcem je George Lucas

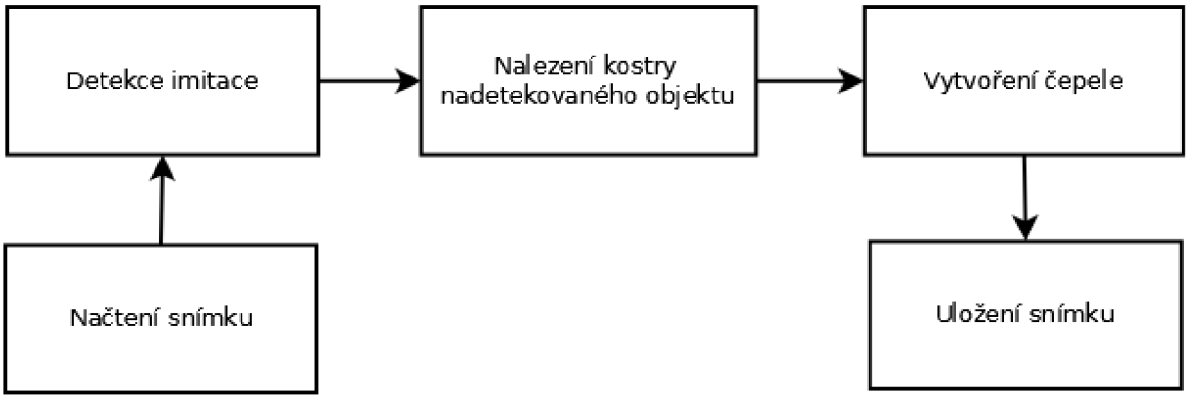
1.1 Imitace světelného meče

Pro natočení videosekvencí je nutné mít připravenou imitaci čepele meče, kterou budeme následně detekovat a nahrazovat efektem čepele světelného meče. Byla použita devadesát centimetrů dlouhá dřevěná tyč, která byla natřena reflexní barvou. Špička byla od zbytku tyče barevně odlišena, aby byla umožněna její detekce. Celé tělo tyče bylo také pomocí tří odlišných barev rozděleno do čtyř zón. Na konec tyče byla umístěna stříbrná imitace jílce světelného meče, která nebude detekována (Obrázek 2).

Obrázek 2: Tyč použitá jako imitace světelného meče

1.2 Stručný popis aplikace

Zpracování jednoho snímku programem probíhá v několika fázích (Obrázek 3). Při zpracování videosekvence jsou ještě nutné přípravné fáze pro rozdělení videosekvence na jednotlivé snímky a konečná fáze, kdy je ze snímků znovu složena videosekvence. Části o detekci imitace je věnována kapitola 6, kapitola 7 se zabývá nalezením kostry nadetekovaného objektu a v kapitolách 8, 9 a 10 jsou popsány postupy pro vytvoření čepele světelného meče.



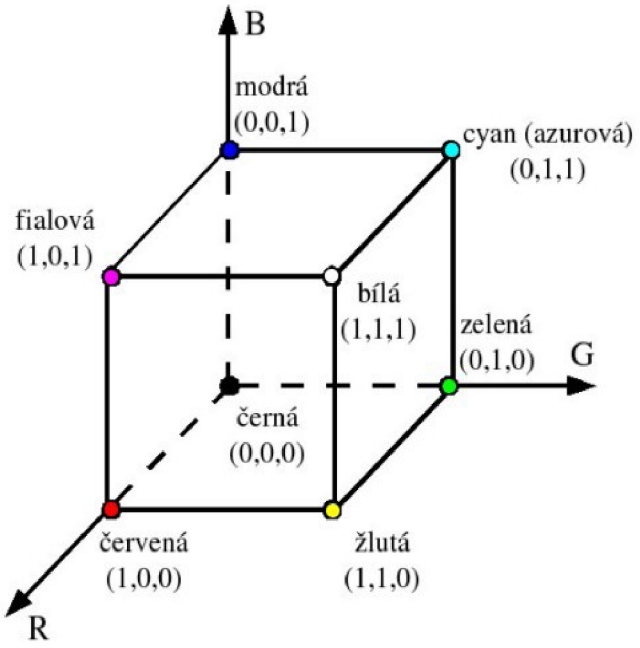
Obrázek 3: Blokové schéma algoritmu pro zpracování jednoho snímku

2 Barevné modely

V této kapitole se seznámíme s barevnými modely, které jsou v naší aplikaci použity. Popis těchto modelů je převzat z publikace [3], obrázky jsou převzaty ze stránek [9].

2.1 Barevný prostor RGB a RGBA

Nejpoužívanějším barevným modelem je model RGB. Na obrazovce vidíme barvu jako výsledek tří složek – *červené* (R, red), *zelené* (G, green) a *modré* (B, blue). Barvy lze vyjádřit trojicí (barevným vektorem), jejíž složky nabývají hodnot z intervalu <0, 1>. Bývají vyjádřeny i v celočíselném rozsahu 0 – 255, což odpovídá kódování každé ze složek RGB v jednom bytu. Hodnota 0 znamená, že složka není zastoupena, maximální hodnota indikuje, že složka nabývá své největší intenzity.



Obrázek 4: Geometrická reprezentace barevného modelu RGB

Barevný rozsah můžeme v prostoru RGB zobrazit prostorově jako jednotkovou krychli umístěnou v osách označených postupně *r*, *g* a *b* (Obrázek 4). Počátek souřadnic odpovídá černé barvě, zatímco vrchol o souřadnicích [1, 1, 1] odpovídá bílé. Vrcholy krychle, které leží na osách,

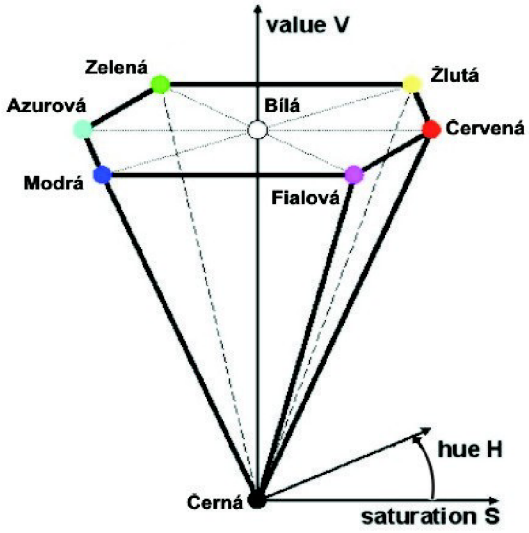
představují základní barvy a zbývající vrcholy reprezentují doplňkové barvy ke každé ze základních barev.

Rozšířením barevného prostoru RGB je prostor RGBA. Zkratka RGBA (či výstižněji RGBα) je používána pro vyjádření skutečnosti, že barevný obraz zapsaný v prostoru RGB je doplněn o informaci o průhlednosti. Každý barevný bod takového obrazu s sebou nese skalární údaj (v rozmezí 0-1), který určuje v jakém rozsahu pokrývá barva plochu obrazového bodu. Hodnota 0.0 znamená neprůhledný barevný bod, maximální hodnota bod zcela průhledný.

Této vlastnosti barevného prostoru RGB je využito při vytváření barevného efektu okolo čepele světelného meče.

2.2 Barevný prostor HSV

Dalším existujícím barevným modelem je model HSV. Tato reprezentace barev se od RGB výrazně liší. U modelu HSV nepředstavuje trojice složek základní barvy. Třemi základními parametry prostoru HSV jsou *barevný tón* (H, hue), *sytost* (S, saturation) a *jasová hodnota* (V, value). Barevný tón označuje převládající spektrální barvu, sytost určuje příměs jiných barev a jas je dán množstvím bílého (bezbarvého) světla.



Obrázek 5: Geometrická reprezentace barevného modelu HSV

Pro zobrazení prostoru se nepoužívá krychle, ale šestiboký jehlan (Obrázek 5), jehož vrchol leží v počátku soustavy souřadnic HSV. Souřadnice *s* a *v* se mění od 0 do 1, souřadnice *h* reprezentuje úhel a nabývá hodnot z intervalu <0°, 360°>. Vrchol jehlanu představuje černou barvu a střed

podstavy reprezentuje bílou barvu. Sytost odpovídá relativní vzdálenosti bodu od osy jehlanu. Dominantní barvy (mají sytost 1) tedy leží na plášti, čisté barvy jsou na obvodu.

2.3 Barevný prostor YUV

Tento barevný prostor patří k rodině barevných prostorů využívaných pro televizi a videotechniku. YUV se používá pro přenos televizních signálů v normě PAL. Hlavním rysem tohoto barevného prostoru je oddělení jasové složky od barevných informací. Díky tomu lze využít jasový signál Y jak pro barevné, tak i černobílé televizory. Signály U a V nesou informace o velikosti barevných složek obrazu.

3 Kresba silné čáry a odstranění šumu

Tato kapitola nás obeznámí s metodou pro kresbu silné čáry, která je v naší aplikaci využita. Aplikace též využívá jednoduchého postupu pro odstranění šumu. Samotný postup pro kresbu silné čáry i postup pro odstranění šumu jsou převzaty z [3].

3.1 Kresba silné čáry

Pro kresbu silné čáry obecně existují dva postupy. Jednoduchý postup je rychlý, ale méně přesný. Druhý postup je přesnější, což ovšem vyžaduje náročnější matematické výpočty. Do jednoduchých postupů patří upravená verze Bresenhamova algoritmu [3], při které se místo jednoho pixelu kreslí v každém kroku několik pixelů nad sebou nebo vedle sebe. Tato metoda má dva důležité negativní vlivy:

1. Síla čáry se mění podle sklonu úsečky.
2. Zakončení čar je nepřirozené. Je ostré a rovnoběžné s jednou se souřadnicových os.

První nedostatek souvisí s metrikou rastru, kdy se šikmé čáry jeví tenčí než svislé a vodorovné. Tento nedostatek lez částečně odstranit přepočítáním požadované tloušťky čáry t na počet pixelů t_p v závislosti na sklonu úsečky (Vzorec 1).

$$t_p = t \frac{\sqrt{(\Delta x)^2 + (\Delta y)^2}}{|\Delta x|}$$

Vzorec 1: Přepočet tloušťky čáry na počet pixelů podle úhlu sklonu

Kvalitnější algoritmy pracují se silnou čárou jako s vyplněnou plochou. Tato metoda je vhodná i z hlediska kresby lomených čar. Nejjednodušším tvarem silné čáry je obdélník. Jeho hranice se snadno vypočítají, problém ale nastává při kresbě lomené čáry. Proto je lepší využít výpočetně náročnější zakončení do oblouku. Toto řešení je i mnohem estetičtější.

3.2 Odstranění šumu

Pro odstranění šumu v obraze se používají různé filtry, které mají za následek určitý stupeň rozmazání obrazu.

V nejjednodušším filtru se nová hodnota bodu počítá jako průměr všech bodů z jeho okolí. Na rozměrech průměrovaného okolí závisí úroveň rozmazání obrazu, platí zde přímá úměra, což znamená čím větší je okolí, tím větší oblast je rozmazána. Této vlastnosti bylo využito v naší aplikaci, kde díky nastavení rozměrů okolí lze určit výslednou velikost barevného efektu (Kapitola 9).

Po aplikaci toho filtru zmizí z obrazu vysoké frekvence a hrany se rozmažou. Tato technika se nazývá *prosté průměrování*.

Existují i složitější filtry jako Gaussovský nebo Fourierův filtr.

4 Morfologické operace

Tato kapitola je věnována popisu morfologických operací, které jsou v naší aplikaci hojně využívány. Za pomoci těchto operací je možné odstranit nežádoucí objekty, vyplnit drobné mezery ve výsledném obrazu nebo zvětšit, či zmenšit původní objekt.

V [4] je morfologie definována jako matematický nástroj pro předzpracování a segmentaci obrazů. Následující text vychází z [4]. Obrázky použité v této kapitole jsou převzaty z [4].

4.1 Typy morfologických operací

V naší aplikaci byly použity tři morfologické operace a to binární dilatace, binární otevření a binární uzavření. Pro úplnost zde uvádím i binární erozi.

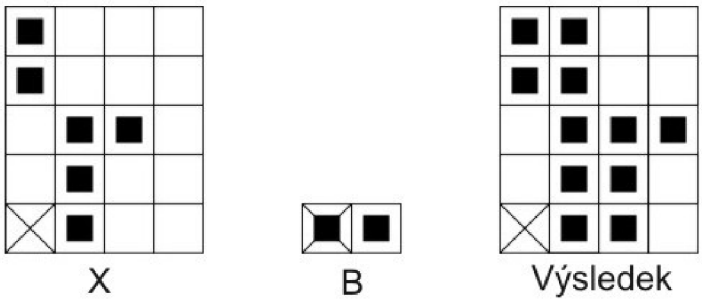
Binární dilatace

Dilataci můžeme vyjádřit jako sjednocení posunutých bodových množin (Vzorec 2).

$$X \oplus B = \bigcup_{b \in B} X_b$$

Vzorec 2: Binární dilatace

Dilatace se používá k zaplnění malých děr a úzkých zálivů v objektech. Při této operaci dochází ke zvětšení původního objektu (Obrázek 6).



Obrázek 6: Binární dilatace

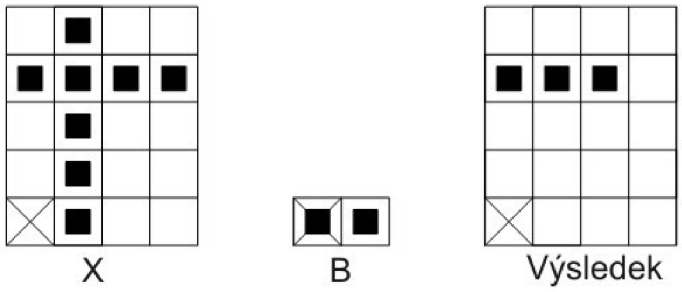
Binární eroze

Jedná se duální operaci k binární dilataci. Erozi můžeme vyjádřit jako průnik všech posunů obrazu X o vektory $-b \in B$ (Vzorec 3).

$$X \ominus B = \bigcap_{b \in B} X_{-b}$$

Vzorec 3: Binární eroze

Eroze se využívá ke zjednodušování struktury objektu. Objekty menší než strukturní element B jsou vymazány (Obrázek 7).



Obrázek 7: Binární eroze

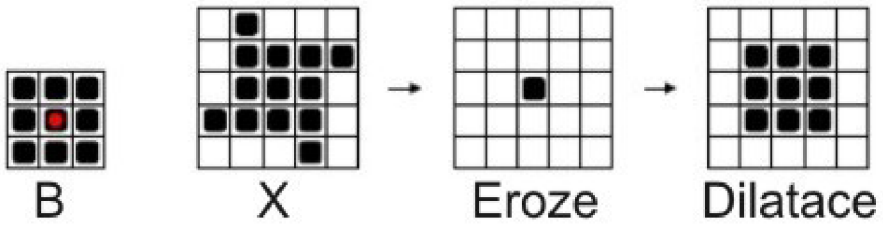
Binární otevření

Binární otevření je operace skládající se z operací binární eroze a binární dilatace. Operace binární je otevření, je tedy binární eroze následovaná binární dilatací za použití totožného strukturního elementu (Vzorec 4).

$$X \circ B = (X \ominus B) \oplus B$$

Vzorec 4: Binární otevření

Binární otevření se používá pro odstranění nežádoucích pixelů při zachování velikosti původního obrazu (Obrázek 8).



Obrázek 8: Binární otevření

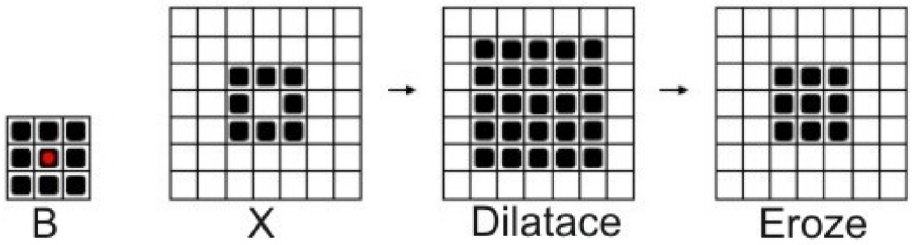
Binární uzavření

Binární uzavření je operace velmi podobná binárnímu otevření. Jediný rozdíl je v pořadí provedených operací, u binárního uzavření se nejdříve provede binární dilatace a poté binární eroze (Vzorec 5).

$$X \bullet B = (X \oplus B) \ominus B$$

Vzorec 5: Binární uzavření

Binární uzavření se používá pro vyplnění drobných mezer v objektu při zachování jeho tloušťky (Obrázek 9).



Obrázek 9: Binární uzavření

5 Vzdáleností mapa

Tato kapitola je věnována kostrám objektů a metodám jejich nalezení. Zvláštní pozornost je věnována vzdálenostním mapám, které jsou v naší aplikaci hojně využívány.

5.1 Kostra objektu a metody pro její nalezení

Formální definice kostry objektu se opírá o pojem maximálního kruhu. Kruh $B(p, r)$ se středem p a poloměrem $r \geq 0$ je množina bodů, pro něž je vzdálenost $d \leq r$. Maximální kruh B vepsaný do množiny X se dotýká hranice X ve dvou a více bodech. Pro nalezení kostry objektu existuje několik metod a postupů [4]:

1. **Vpisování kruhů:** Vychází přímo z definice kostry objektu a prakticky se nepoužívá. Přílišná výpočetní složitost. Porušuje se souvislost. Kostra objektu tloušťky > 1.

2. **Sekvenční ztenčování:** Oblast se eroduje vhodným strukturním elementem, který zaručí, aby nebyla porušena souvislost. Obvykle homotopické ztenčování, s využitím strukturních elementů z Golayovy abecedy.

3. **Vzdálenostní mapa:** Rychlý výpočet za pomocí vzdálenostní transformace. Nejčastěji používané. Tato metoda je použita v naší aplikaci, a proto se jí budeme detailněji zabývat.

4. **Koutková reprezentace:** Oblasti se nejdříve bezeztrátově komprimují, čímž vznikají tzv. koutky. Kostra objektu se získá vpisováním maximálních obdélníků přímo v komprimovaných datech

5.2 Vzdáleností mapa

Vzdáleností mapa pracuje nad binárním obrazem a udává nám nejmenší možnou vzdálenosti jednotlivých pixelů od okraje objektu. Samotný okraj objektu je na mapě reprezentován hodnotou 1 (Obrázek 10). Jedná se o dvouprůchodovou metodu.

Následující způsob výpočtu vzdáleností mapy vychází z metod popsaných v práci J. Krále [5]. Nejprve je nutné zvolit vzdálenostní metriku:

1. **Metrika k_1:** Je použit pouze vertikální a horizontální krok o velikosti 1. Použitá maska vypadá jako kříž. Tato metrika se nazývá „*městská metrika*“.

2. **Metrika $k_{1,1}$:** Kromě vertikální a horizontální kroku o velikosti 1 je povolen i diagonální krok o téže velikosti. Tato metrika se nazývá „*šachovnicová metrika*“.

3. **Metrika $k_{1,2}$:** Tato metrika se od metriky $k_{1,1}$ liší pouze velikostmi kroků. Vertikální a horizontální krok zůstává 1. Diagonální krok má velikost 2. Použitím této metriky získáme lepší aproximaci vzdálenosti.

4. **Metrika $k_{1,2,3}$:** U této metriky má vertikální a horizontální krok opět velikost 1a diagonální krok velikost 2. Navíc je přidán krok připomínající pohyb koně po šachovnici o velikosti 3. Touto metrikou získáme ještě lepší aproximaci vzdálenosti než v předchozím případě.

V naší práci je využita metrika $k_{1,1}$, která pokrývá i diagonály (Obrázek 10).

0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	0	0
0	1	1	2	2	2	2	1	1	0
0	1	2	2	3	3	2	2	1	0
0	1	2	2	3	3	2	2	1	0
0	1	1	2	2	2	2	1	1	0
0	0	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0

Obrázek 10: Binární obraz a jeho vzdáleností mapa po transformaci s metrikou $k_{1,1}$, převzato z [5]

6 Detekce imitace v obraze

Prvním důležitou částí naší aplikace je detekce imitace čepele světelného meče ve snímku. Na kvalitním výsledku tohoto postupu je závislý celý další běh aplikace.

6.1 Detekce barvy založená na modelu RGB

Protože světelné podmínky nejsou ve všech částech snímku stejné, není možné porovnávat jednotlivé pixely pouze s jediným odstínem hledané barvy. Tento problém řeší použití euklidovské vzdálenosti mezi dvěma vektory.

$$T = \sqrt{\left(\left(R_P - R_H\right)^2 + \left(G_P - G_H\right)^2 + \left(B_P - B_H\right)^2\right)}$$

Vzorec 6: Výpočet euklidovské vzdálenosti

Ve (Vzorec 6) jsou indexy *P* označeny hodnoty z testovaného pixelu a indexy *H* jsou označeny referenční hodnoty. Výsledná vzdálenost *T* je poté porovnána s předem nastaveným prahem. Výsledky nižší než je hodnota prahu jsou označeny jako součást hledané čepele. Tento postup se ukázal jako málo přesný, velice často docházelo k mylné detekci pixelů, které nenáležely do hledané čepele. Proto byl předcházející postup nahrazen detekcí barvy založenou na barevném modelu HSV.

6.2 Detekce barvy založená na modelu HSV

Pro detekci barevného objektu založenou na barevném modelu HSV jsou postačující pouze první dva parametry, barevný tón *H* a sytost *V*. Jsou určeny minimální a maximální hodnoty barevného tónu, které se mohou na čepeli vyskytovat. Dále je určen práh, tedy minimální hodnota sytosti, která je pro hledanou barvu ještě přípustná. Pokud barevný tón testovaného pixelu náleží do intervalu <min, max> je následovně porovnána její sytost s nastaveným prahem. Pokud je sytost vyšší než práh, je pixel označen jako součást čepele. Tato metoda se ukázala být přesnější než metoda založená na barevném modelu RGB.

6.3 Průběh detekce barvy

Při detekci je vytvářen černobílý snímek, který bude následně sloužit jako maska při nahrazení imitace čepele barevným efektem čepele světelného meče.

Hodnoty prahů pro základní barvu, špičku a barevné značky umístěné na čepeli jsou přednastaveny a jejich hodnota může být upravena uživatelem při startu programu. Vstupem je snímek v barevném modelu RGB (Obrázek 11), který je pro potřeby detekce nutné nejdříve převést na barevný model HSV.

Obrázek 11: Vstupní snímek

V první fázi je detekována základní barva imitace. Snímek se prochází bod po bodu a každý pixel je testován na přítomnost hledané barvy. Výsledkem jsou v ideálním případě čtyři shluky pixelů připomínající obdélníky. K těmto čtyřem obdélníkům je přistupováno jako k jedinému objektu. Jsou určeny minimální a maximální hodnoty na osách x a y, kterých pixely tvořící obdélník nabývají. Je porovnána vzdálenost mezi body, které mají extrémy na ose x a body s extrémy na ose y. Body s větší vzdáleností jsou označeny jako vrcholy. Tyto body v ideálním případě tvoří úhlopříčku nadetekovaného objektu. Za pomocí Bresenhamova algoritmu pro rasterizaci úsečky [3] jsou procházeny jednotlivé pixely tvořící úhlopříčku. Pokud zkoumaný pixel nenáleží masce detekovaného objektu je provedena funkce, které za pomoci algoritmu založeném na semínkovém vyplňování oblastí [3] hledá přítomnost jedné ze tří možných barevných značek. Pixel zaslaný tomuto

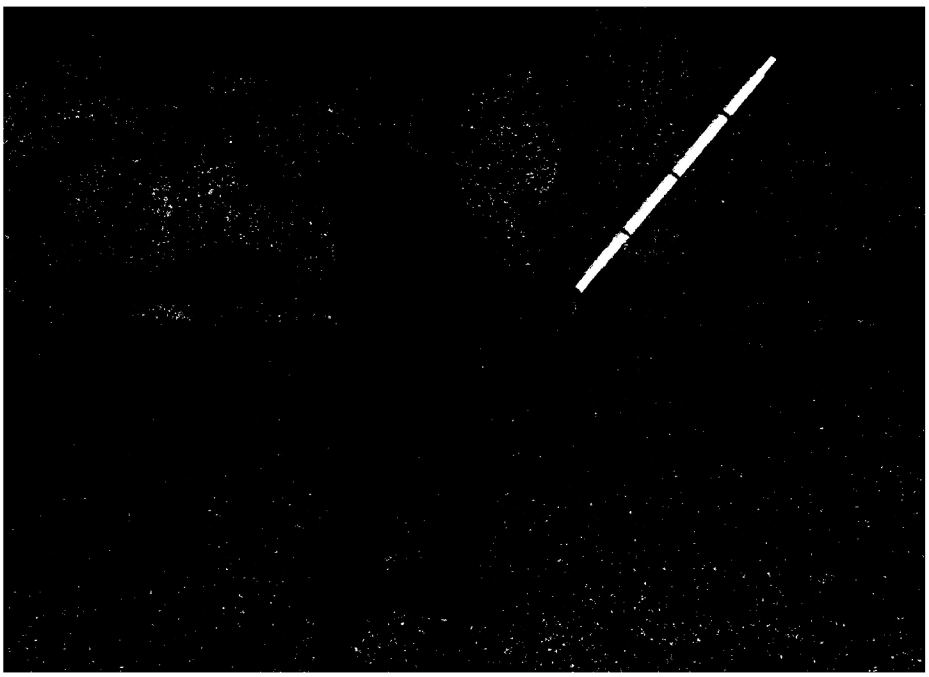
algoritmu je použit jako prvotní semínko a jeho osmiokolí se testuje na přítomnost barvy značky shodné s barvou prvního pixelu. Každý pixel, který obsahuje hledanou barvu, se stává novým semínkem.

Po prohledání všech bodů úsečky jsou, za použití totožného postupu jako v předchozím případu, znovu určeny vrcholy obdélníku a pro oba nalezené vrcholy je použita funkce pro určení, zda se jedná o špičku meče. Této funkci je předán k prozkoumání vrchol a jeho okoli je prohledávano na přítomnost barvy špičky, až do vzdálenosti, která je dána následující podmínkou: Pokud je šířka obrázku větší než *800,* je použit vztah *sirka_obrazku /100/2*. V opačném případě je použit vztah *sirka_obrazku /30/2.* Když je barva nalezena, postupuje se stejným způsobem jako ve funkci pro hledání barevných značek. Jediným rozdílem je fakt, že je uložen počet nalezených pixelů dané barvy. Tento počet je následně testován a to takto: Pokud je šířka obrázku větší než *800* musí být minimální počet *50 pixelů*, pro šířku nižší než *800* je tato hodnota *10 pixelů.* Pro další zpracování je uložena pozice vrcholu, u kterého se nachází nalezená špička.

Nakonec je vstupní obrázek převeden zpět do barevného modelu HSV.

6.4 Odfiltrování nežádoucích pixelů z obrázku

Během celého procesu detekce barvy dochází k detekování nežádoucích pixelů (Obrázek 12). Tyto pixely většinou tvoří jen malé shluky, přesto je nutné jejich odstranění.

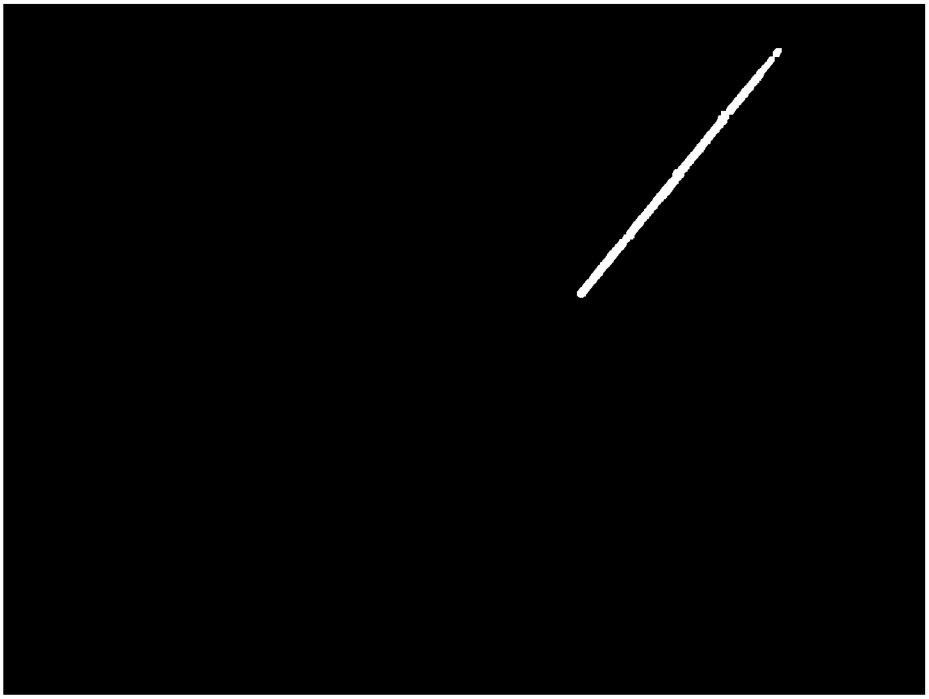


Obrázek 12: Nežádoucí detekované pixely

Tento problém nastává hlavně při detekci základní barvy a musí být odstraněn dříve, než jsou hledány vrcholy imitace. Odstranění se provede za pomoci morfologické operace binární otevření (Kapitola 4.1). Stejná operace je opětovně provedena i po nalezení špičky.

Po nalezení barevných značek se provede operace binární uzavření (Kapitola 4.1), které vyplní drobné mezery ve vzniklém obrázku. Tato operace je opětovně provedena i po nalezení špičky.

Výsledek detekce po odstranění nežádoucích pixelů je patrný z obrázku (Obrázek 13).



Obrázek 13: Ukázka výsledku po detekci imitace čepele

7 Nalezení osy objektu

Výsledný objekt vzniklý po detekci imitace čepele není příliš vhodný pro další zpracování. Proto je tento objekt nahrazen svou středovou osou, která se získá pomocí postupu popsaném níže.

7.1 Zpracování vzdálenostní mapy

Postupy pro práci s nalezenou kostrou vycházejí s metod popsaných v práci J. Krále [5], byly ovšem modifikovány pro použití v naší vzdálenostní mapě.

V dalším textu bude právě testovaný bod označen jako *X* a body okolí jako *S*.

Nalezení bodů tvořících kostru

Z bodů tvořících objekt je nutné vybrat pouze ty body, které jsou součástí kostry objektu. Z definice kostry objektu (Kapitola 5.1) je zřejmé, že body tvořící kostru jsou lokální maxima v určitém prostoru. V naší aplikaci je vždy prohledáván ten nejmenší možný prostor okolo zkoumaného bodu (Obrázek 14).

Postup pro nalezení bodů tvořících kostru objektu je následující: Postupně jsou procházeny všechny body tvořící kostru a každý z nich je testován, zda není lokálním maximem, vzhledem ke svému okolí. Bod je lokálním maximem, pokud je splněna následující podmínka: Bod musí být větší nebo roven nejvyšší hodnotě, kterou nabývají body tvořící jeho okolí.

Při hledání bodů náležejících kostře objektu je také zjištěna maximální šířka zkoumaného objektu. Zjištění údaj poté poslouží jako šířka vytvořené čepele.

8	1	2
7	X	3
6	5	4

Obrázek 14: Okolí bodu *X* s označením jednotlivých bodů

Propojení nalezených bodů

Body nalezené v předcházejícím bodu netvoří souvislou kostru, a proto je nutné je navzájem propojit. Postupně jsou procházeny všechny body tvořící kostru. U každého bodu X je určen jeho soused s nejmenším rozdílem. Tento rozdíl se vypočítá jako $rozd = (hodnota_X - hodnota_S)$. Pokud by mělo více okolních bodů stejnou hodnotu rozdílu, bude jako bod kostry přednostně označen bod ležící na lichém okolí, reprezentovaném indexy *1, 3, 5, 7*. V těchto bodech má vždy přednost bod s menším indexem. Stejný postup platí i pro sudé body ležící na diagonále.

Ztenčení kostry na jednotkovou šířku

Nalezená kostra není na všech místech široká pouze jeden pixel, a proto je nutné ji ztenčit. Tato procedura se stává se dvou kroků. V prvním kroku jsou jako součást kostry ponechány body, které sousedí s jinými body kostry všech lichých indexech. Ostatní body pokračují na další analýzu do druhého kroku. Zde jsou z kostry zrušeny všechny body, pro které platí následující podmínka: S_i a S_{i+2} jsou body kostry a S_{i+5} není bodem kostry pro $i \in \{1,3,5,7\}$. Dále musí existovat alespoň jeden horizontální nebo vertikální (body s lichým indexem) soused bodu X, který není bodem kostry.

Nalezení koncových bodů kostry

Nalezení koncových bodů je nutný předpoklad pro odstranění krátkých větví a také pro konečné nalezení osy objektu. Jako koncový označíme každý bod kostry, který nemá žádného nebo právě jednoho souseda náležející kostře objektu. Jako koncový je též označen bod mající právě dva sousedy a to sousedy S_i a S_{i+1} pro $i \in \{1,2,3,4,5,6,7,8\}$.

Odstranění krátkých větví

Při předchozích operacích mohli vzniknout krátké větve, které nejsou součástí kostry. Tyto větve je nutné odstranit. Nejprve jsou nalezeny koncové body kostry. Poté budeme od každého koncové bodu postupně procházet kostrou a body, přes které jsme již prošli ukládat do zásobníku. Procházení je ukončeno pokud se narazí na konec kostry nebo na křižovatku. Pokud se narazí na křižovatku jsou body uložené na zásobníku krátkou větví a můžeme je odstranit. Křižovatka je bod, který má dva sousední body, a musí platit že body S_i a S_{i+1} nesmí být body kostry pro $i \in \{1,2,3,4,5,6,7,8\}$. Dále je jako křižovatka označen bod, který má tři a více sousedů.

7.2 Určení středové osy objektu

Po zpracování kostry objektu postupem popsaným v předcházející kapitole jsou znovu určeny koncové body kostry. V ideálním případu by měly nyní zbývat pouze dva body. Toho však nemusí být v důsledku nedostatečné detekce imitace dosaženo. Nadetekovaný objekt může být v určitých místech příliš tenký, čímž mohou vznikat mezery v kostře. Pokud je nalezeno více než dva koncové body, je nutné rozhodnout které z nich budou označeny jako počáteční a koncový bod osy objektu. K určení těchto bodů je využit stejný postup jako v případě hledání vrcholů v kapitole 6.3.

Jako poslední je určena orientace osy. Tento údaj je následně využit při vytváření čepele. Orientace se určí pomocí následující podmínky: Pokud je vzdálenost mezi koncovými body na ose *x* nebo ose *y* menší než třetina celkového počtu bodů tvořících kostru je orientace nastavena na číslo *2* v opačném případě je orientace rovna *1*. Tento údaj bude blíže vysvětlen v následující kapitole.

8 Vytvoření čepele světelného meče

Po nalezení osy objektu je již možné nahradit původní nadetekovaný objekt novým objektem, který bude představovat čepel světelného meče. Čepel bude vytvořena ve dvou krocích, kdy je nejdříve vytvořena úsečku o požadované tloušťce. Tato úsečka bude mít zaoblené konce, což je vhodné pro špičku meče, ne pro druhý konec, který je napojen na jílec meče. Proto je nutné v druhém kroku upravit toto napojení na hranaté.

8.1 Vytvoření hranatého zakončení jílce

Pro vytvoření spodní části čepele, která je napojena na jílec meče by bylo možné využít algoritmus pro kresbu silné čáry s hranatým zakončením. Nicméně se ukázalo, že rychlejší způsob je využití původní osy objektu, která je následně rozšířena pomocí binární dilatace (Kapitola 4.1).

Při zvětšování objektů pomocí binární dilatace je velmi důležitým aspektem volba správného strukturního elementu. Při určitých úhlech je nutné použít strukturní element ve tvaru kříže, jindy je vhodnější použití klasického čtvercového tvaru. Pokud je použit nevhodný strukturní element dochází k deformaci objektu (Obrázek 15).

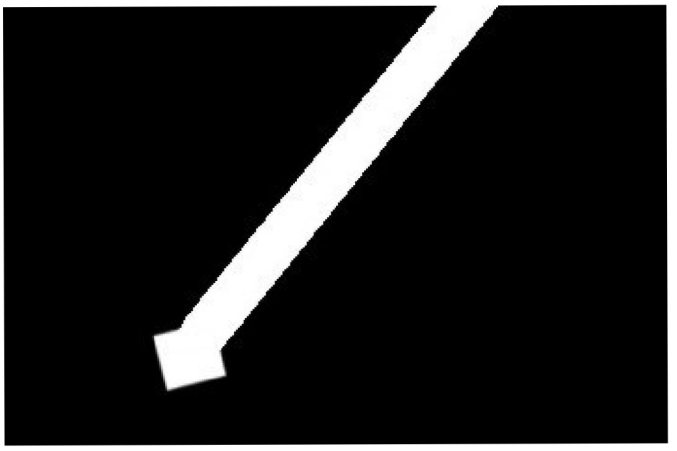
Při vytváření napojení čepele na jílec se využije následující postup: Nejdříve je vykreslena přímka o šířce jeden pixel, jejímž počátečním a koncovým bodem jsou body získané z analýzy kostry objektu. Tato přímka je následně rozšířena strukturním elementem. Pokud je orientace čepele (Kapitola 7.2) *1*, použijeme jako strukturní element *kříž*. Počet binárních dilatací v tomto případě je stanoven vztahem $počet = 1.2 \cdot šířka\ čepele$. Pro orientaci čepele *2* se jako strukturní element naopak využije *čtverec*. Počet binárních dilatací je nyní dán vztahem $počet = 0.8 \cdot šířka\ čepele$.

8.2 Vytvoření čepele

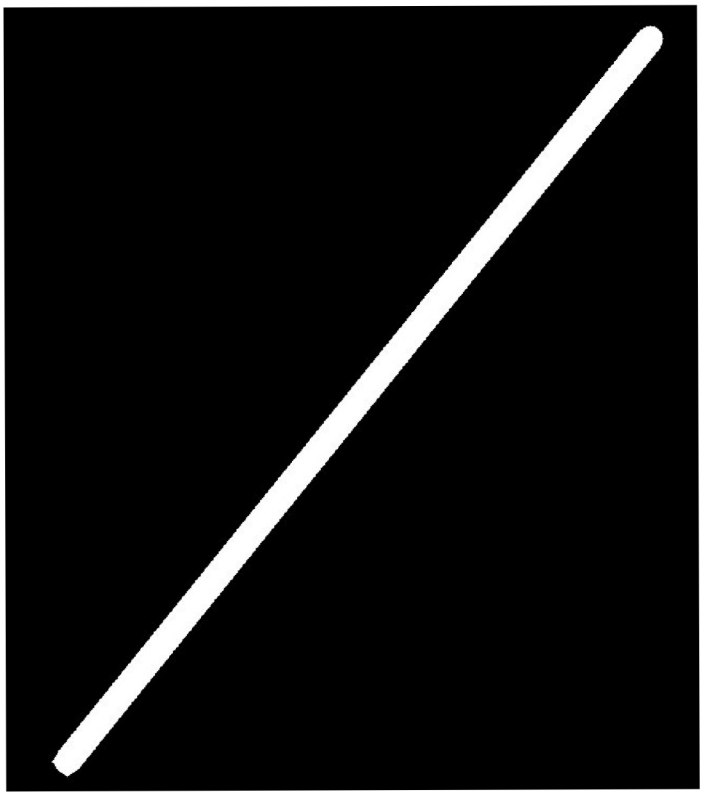
Základ čepele je vytvořen pomocí algoritmu pro vykreslení silné čáry se zaoblenými konci. Tato čára má šířku podle vztahu $šířka = 2 \cdot maximální\ šířka$. Počátečním a koncovým bodem jsou opět body získané z analýzy kostry objektu. Pomocí údaje o pozici špičky (Kapitola 6.3) je určen konec, na kterém bude ponecháno zakulacené zakončení. Z přímky je následně vytvořen korekční maska, která se skládá z původní přímky a jejího posunu ve směru od špičky k jílci. Pro přehlednost budeme této masce říkat *maska_1*. Posun se provádí o celou délku původní přímky. Do další pomocné masky je vykreslen kruh o velikosti: $velikost = 2.5 \cdot šířka\ čepele$. Tuto masku označme jako *maska_2*. Tento kruh se využije při přenosu hranatého zakončení na čepel. Samotný přenos hranatého zakončení

probíhá následovně: Pixel se z přímky vytvořené v předcházejícím odstavci přenese na čepel, pokud zároveň leží na čepeli v *masce_1* a kruhu v *masce_2*.

Výsledek tohoto postupu je patrný z (Obrázek 16).



Obrázek 15: Výsledek s nevhodným strukturním elementem a bez použití *masky_1*



Obrázek 16: Čepel světelného meče se zakončením pro napojení na jílec

9 Vytvoření barevného efektu okolo čepele světelného meče

Na vytvořenou čepel je nyní nutné nanést barevný okraj a také vyrobit barevný efekt (barevnou korónu) okolo samotné čepele.

9.1 Barevný okraj čepele

Chceme, aby měla čepel okolo celého svého obvodu, vyjma strany napojené na jílec, barevný okraj. Znovu je využita vzdálenostní mapa. Před jejím vytvořením je nutné nejdříve prodloužit čepel směrem od špičky k jílci. Prodloužení je nutné provést, jinak by se nám vytvořil barevný okraj i na straně jílce. Toto prodloužení musí mít ovšem menší šířku než původní čepel, jinak by barevný okraj byl u jílce příliš úzký. Proto je vytvořena nová úsečka o *šířka = 1,8 · maximální šířka.* Tato přímka je poté posunuta o celou délku čepele směrem od špičky k jílci. Postup je naprosto stejný jako v předchozí kapitole.

Postupně se procházejí všechny body čepele a ze vzdálenostní mapy se zjišťuje zda budou obarveny nebo ne. Bod je obarven pokud je jeho vzdálenost větší nebo rovna *jedné pětině šířky čepele*.

9.2 Barevný efekt okolo čepele

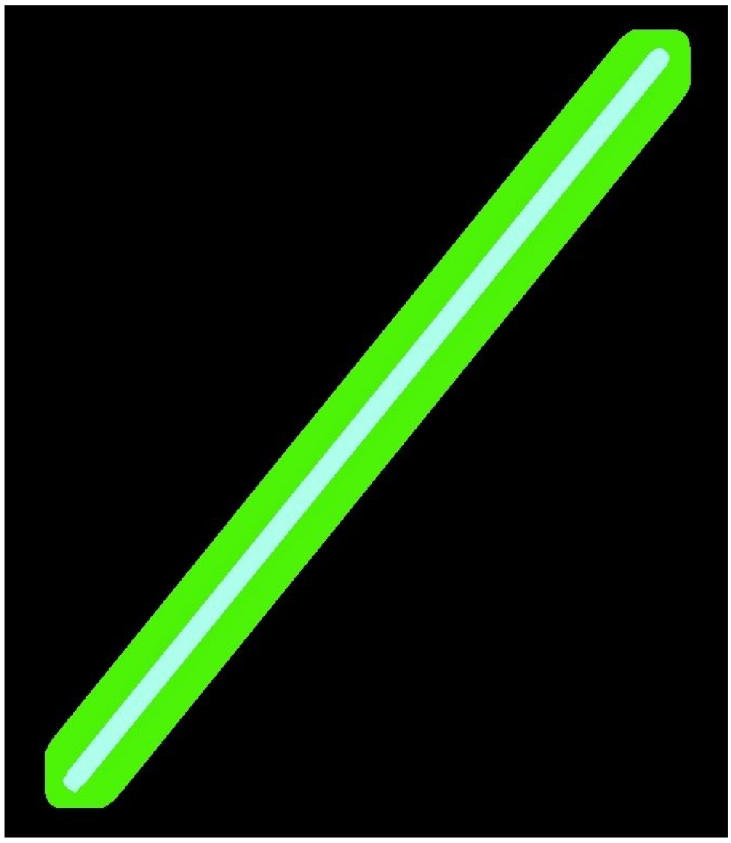
Okolo čepele nyní chceme vytvořit záři o požadované barvě. Tento problém by byl řešitelný pomocí binární dilatace. V tomto případě bychom rozšířili původní čepel o určitý počet pixelů a následně rozšířenou oblast obarvili požadovanou barvou. Tento postup má však úskalí, které bylo popsáno v kapitole 7.2. Zde ovšem není možné opravit přesahy vzniklé po binární dilataci, proto není tato metoda využita. Místo ní je využit postup pro odstranění šumu (Kapitola 3.2).

Při vytvoření žádaného barevného efektu se postupuje následovně: Celá čepel se vybarví požadovaným odstínem barvy a její okolím podobným odstínem, který je ovšem mírně pozměněn podle vztahu (Vzorec 7).

$$R_{okoli} = R_{čepel} + 5$$
$$G_{okoli} = G_{čepel} + 5$$
$$B_{okoli} = B_{čepel} + 5$$

Vzorec 7: Přepočet barvy čepele na barvu okolí pro rozmazání

Poté je provedena operace rozmazání. Velikost okolí je dána vztahem *okolí = 6 · šířka čepele.* Po dokončení rozmazání je barva okolí nahrazena barvou černou. Nakonec je vzniklému barevnému efektu přidána čepel s obarvenými hranami (Obrázek 17).



Obrázek 17: Výsledná maska tvořená čepelí a barevným efektem

10 Vložení výsledného efektu do původního obrázku

Čepel světelného meče je nyní již hotová a zbývá ji vložit zpět do původního snímku. Při vkládání čepele zpět do původního snímku je využita průhlednost, tedy vlastnost barevného prostoru RGBA (Kapitola 2.1).

10.1 Nastavení různého stupně průhlednosti barevného efektu

Aby barevný efekt okolo čepele světelného meče nevypadal příliš uměle a lépe splynul s okolím, je nutné nastavit různé stupně průhlednosti tohoto efektu v závislosti na vzdálenosti od samotné čepele. Pro splnění tohoto úkolu opět využijeme služeb vzdálenostní transformace.

Nejdříve je vytvořena vzdálenostní mapa celé masky. Z této mapy se zjistí maximální šířka, od které bude odečtena šířka samotné čepele. Výsledné číslo udává maximální vzdálenost barevného efektu od hrany čepele světelné meče.

V závislosti na maximální vzdálenosti barevného efektu od hrany čepele bude tento efekt rozdělen do čtyř nebo šesti zón s různou průhledností. Toto rozhodnutí je řízeno jednoduchou podmínkou: Pokud je maximální vzdálenost menší než šest bude použito rozdělení do čtyř zón, jinak bude využito rozdělení do zón šesti. Celková velikost barevného efektu okolo čepele se poté rovnoměrně rozdělí do těchto zón.

Při použití šesti zón jsou stupně průhlednosti nastaveny podle (Vzorec 8), kde i je číslo zóny (číslováno ve směru od čepele).

$$průhlednost_i = 30 + 10 \cdot i$$

Vzorec 8: Výpočet průhlednosti pro šest zón

Pokud je barevný efekt rozdělen pouze do čtyř zón, jsou stupně průhlednosti pro první tři zóny nastaveny dle (Vzorec 9). Pro poslední zónu platí jiný vztah (Vzorec 10). Číslování zón je stejné jako v předchozím případě.

$$průhlednost_i = 50 + 10 \cdot i$$

Vzorec 9: Výpočet průhlednosti pro první tři zóny při rozdělení do čtyř zón

$$průhlednost_i = 45 + 10 \cdot i$$

Vzorec 10: Výpočet průhlednosti čtvrté zóny při rozdělení do čtyř zón

10.2 Vložení čepele do původního snímku

Posledním krokem při zpracování snímku je vložení nově vytvořené čepele světelného meče zpátky do původního snímku, kde tato čepel překryje imitaci. Tomuto procesu se říká maskování. Při maskování slouží jeden snímek jako maska, ze které je přenesen požadovaný kus obrazu do druhého snímku. Maska má většinou černý podklad, na kterém je umístěn maskovaný objekt. Naše maska je vytvořena s černým podkladem. Při přenosu objektu z masky jsou potom kopírovány pouze ty pixely, které nemají černou barvu. Při kopírování barevného efektu okolo čepel je uplatněna předem nastavená průhlednost. Výsledek zpracování jednoho snímku lze vidět na (Obrázek 18).

Obrázek 18: Výsledný snímek po zpracování programem

11 Zpracování videa

V předchozích kapitolách byl popsán postup pro zpracování samostatného obrázku či jednoho snímku videa. Při zpracování celého videa je nutné zavést určité kontroly a korelační přepočty pro zajištění plynulosti pohybu čepele.

11.1 Zjištění chybné detekce

Jelikož jsou hodnoty prahu nastaveny za pomoci prvního snímku videa je velká pravděpodobnost, že u některých snímků dojde k chybné detekci čepele. Tato chybná detekce může mít více podob. Nejdůležitějšími a nejnepříjemnějšími chybami, které chceme eliminovat, jsou detekce jiných pixelů, než jsou ty náležející čepeli, a detekce pouze části čepele.

Detekce chyb je zařazena ihned za získáním osy detekovaného objektu (Kapitola 7). Díky tomu, že výstupem z tohoto zpracování jsou pouze dva body, je výsledná detekce poměrně jednoduchá. Pro účely detekce chyb je nutné si uchovávat body počátku a konce osy objektu z předchozí správné detekce, tyto body jsou rozděleny na bod reprezentující špičku a bod reprezentující jílec. Body jsou poté porovnávány s nově získanými body následujícím způsobem: Pokud je nový bod špičky (jílce) na ose x (y) vzdálen od předchozího uloženého bodu o více než *počet snímků za vteřinu* (*fps*), je celá detekce označena jako chybná, původní snímek je uložen a přechází se k dalšímu snímku. Zde se celé porovnávání opět opakuje, ovšem počet snímků je vynásoben dvěma, neboť v předchozím snímku došlo k chybné detekci. Násobitel počtu snímků roste s počte chybných detekcí. Jakmile je opět nalezen snímek se správnou detekcí, je násobitel znovu nastaven na *1* a pozdržené chybné snímky se opraví.

Pro zajištění přesnosti oprav je stanoven maximální počet po sobě následujících chyb. Pokud nastane situace, kdy by po sobě následovalo více než *10 chyb*, program se ukončí.

11.2 Oprava snímků s chybnou detekcí

Po identifikaci snímků s chybnou detekcí a nalezením prvního snímku se správnou detekcí, který následuje okamžitě po poslední chybné detekci, jsou chybné snímky opraveny. Pro tuto opravu je nutné znát pozice špičky a jílce pro poslední správnou detekci před vznikem chyb a také první správnou pozici špičky a jílce první správné detekce po chybných snímcích. Z těchto hodnot jsou vypočteny nové hodnoty (Vzorec 11) pro obě osy špičky a jílce.

$$hodnota = hodnota_{\,po\,chybách} - hodnota_{\,před\,chybami}$$

Vzorec 11: Výpočet nových hodnot pro chybné detekce

Postupně se projdou všechny snímky uložené pro opravu, které jsou očíslovány od nejstaršího po nejnovější, a vypočítají se jim nové pozice špičky a jílce pro obě osy (Vzorec 12).

$$pozice = hodnota_{\,před\,chybami} + číslo\,snímku \cdot hodnota$$

Vzorec 12: Výpočet nových hodnot pro opravované snímky

S nově vypočtenými pozicemi špičky a jílce se nyní vykreslí čepel, vloží se do uloženého snímku a ten se poté přidá do videa.

11.3 Chybná detekce špičky

Je nutné též ošetřit možnost chybné či žádné detekce špičky meče. Bez řádně detekované špičky se může lehce stát, že bude špička umístěna na jílec nebo že bude čepel meče o něco kratší než má být. Proto je pro každý snímek porovnávána pozice špičky s její pozicí v předcházejícím snímku. Mohou nastat tři možné chyby:

1. Špička nebyla detekována vůbec
2. Špička je detekována na straně jílce
3. Jsou detekovány dvě špičky

První chyba je způsobena příliš vysokou hodnotou prahu pro detekci špičky, druhá a třetí chyba je způsobena příliš nízkou hodnotou prahu a také barevnou blízkostí špičky a jílce při určitých světelných podmínkách.

Oprava první chyby je velmi prostá, stačí pouze určit, který z vrcholů v aktuální detekci je blíže ke špičce z předchozí detekce. Nyní, když známe pozici špičky, provedeme její posunutí o *dvě procenta délky* směrového vektoru ve směru od jílce ke špičce.

Druhá a třetí chyba se opravuje stejným způsobem jako chyba první, pokud totiž došlo k detekci dvou špiček nebo určení jílce jako špičky je délka nadetekovaného objektu nedostatečná a je nutné jej prodloužit jako v prvním případě.

11.4 Korelační přepočet mezi snímky

Různé odchylky mezi detekcemi v jednotlivých snímcích způsobují místy nepříjemné třepotání výsledného obrazu ve videu. Tento problém řešíme váženým průměrováním jednotlivých hodnot špiček a jílců ve snímcích s hodnotami v předcházejícím a následujícím snímku. Výpočet váženého průměru se opět provede pro hodnoty špičky a jílce na obou osách (Vzorec 13).

$$hodnota = \frac{hodnota_{\text{předchozí snímek}} + 2 \cdot hodnota_{\text{aktuální snímek}} + hodnota_{\text{následující snímek}}}{4}$$

Vzorec 13: Výpočet váženého průměru

Aby bylo možné tento korelační výpočet, provést je nutné snímek vždy pozdržet a počkat na načtení a zpracování dalšího snímku.

12 Knihovna OpenCV

Při tvorbě výsledné aplikace byla použita knihovna OpenCV [6][7], která obsahuje spoustu užitečných funkcí pro zpracování obrazu. Knihovnu je možné použít pro načítání jednotlivých snímků i videí. S její pomocí lze snadno rozdělit video na jednotlivé snímky i jednotlivé snímky spojit do videa. V knihovně OpenCV jsou též kromě implementace kresby základních geometrických tvarů a převodů mezi barevný modely implementovány též morfologické operace a funkce pro vytvoření vzdálenostní mapy. Toto není konečný výčet všech funkcí a možností této knihovny, jedná se pouze o hrubý nástin. OpenCV ovšem obsahuje některé odlišnosti, na které je si třeba dát pozor.

12.1 Odlišnosti při práci s obrázky

První odlišností je způsob při práci s obrázky ve formátu RGB. V OpenCV jsou totiž informace nesoucí barvy obrázku uloženy v invertované podobně, tedy jako BGR. Tato odlišnost ovšem neplatí pro barevný prostor HSV. U barevného prostoru HSV jsou však jiné intervaly, kterých mohou nabývat jednotlivé prvky barevného prostoru. H má v OpenCV interval <0°, 180°>, S a V nabývají hodnot od 0 do 255. Na tyto rozdíly je třeba brát při používání OpenCV zřetel.

12.2 Funkce OpenCV využité v aplikaci

Následující funkce nebyly v naší aplikaci implementovány, neboť jsou místo nich využity metody z knihovny OpenCV:

- načítání, ukládání obrázků a operace s nimi jako je např. kopírování celých snímků
- načítání videa, jeho rozdělení na jednotlivé snímky a jeho následné uložení
- převody mezi barevnými modely RGB a HSV
- kresba silné čáry
- morfologické operace
- vytvoření vzdálenostní mapy

13 Implementace a testovaní

V této kapitole je popsán návrh a implementační stránka aplikace. Většina kapitoly je věnována testování.

13.1 Návrh a implementace

Program byl od začátku koncipován jako konzolová aplikace, později bylo ovšem nutné přidat okno pro nastavení prahů pro aktuální video nebo obrázek. Program je navržen jako tři samostatné třídy *frame*, *rozhrani* a *stream*. Třída *frame* je nejdůležitější ze všech tří tříd, neboť zpracovává vstupní snímek a vrací již snímek s přidaným efektem světelného meče. Třída *rozhrani* slouží pro práci s oknem, které slouží uživateli pro nastavení prahů jednotlivých barev a třída *stream* pomáhá při zpracování videa. Při návrhu všech funkcí a tříd byl kladen velký důraz na jednoduchost a přehlednost celého řešení.

Celá aplikace je implementována v programovacím jazyce C++ z využitím knihovny OpenCV, která je popsána v předchozí kapitole. Jazyk C++ byl zvolen pro jeho objektovou orientovanost.

Aplikace byla vyvíjena a testována v operačním systému Ubuntu 9.04. Kompatibilita s operačním systémem Windows nebyla testována.

13.2 Testování

Výsledný program bylo nutné podrobit sérii testů, aby byla ověřena jeho funkčnost. Jelikož byl program vyvíjen a průběžně testován pouze na jednom snímku, bylo žádané provést testy na dalších snímcích. Tato série testů byla provedena na dalších snímcích a jejich kopiích v různém rozlišení. Po dokončení testů bylo nutné opravit některé chyby vzniklé při vývoje pro jeden snímek. Příkladem takových chyb byla nevhodná reakce programu při umístění imitace rovnoběžně s některou s os.

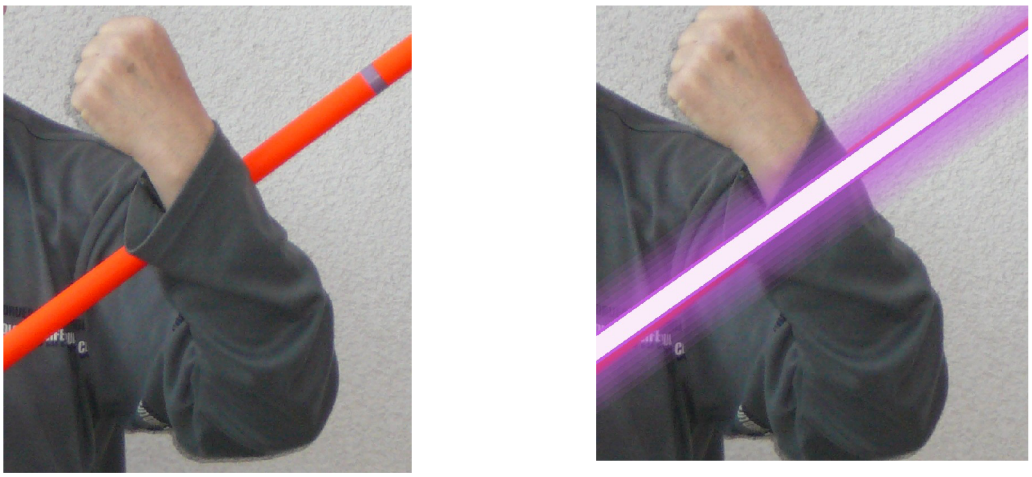
Během těchto prvních testů byla zjištěna nepříjemná vlastnost spojená se zpracováním vzdálenostní mapy. Zvolený postup pro zpracování vzdálenostní mapy na kostru objektu se ukázal být velmi přesný, leč také časově náročnější, než bylo předpokládáno. Tento neblahý rys padá na vrub způsobu, kterým je celá vzdálenostní mapa analyzována. Proto byl celý postup upraven a bylo odstraněno neustále procházení všech pixelů obrázku. Namísto toho jsou body tvořící kostru ukládány do vhodného zásobníku. S touto obměnou se běh aplikace výrazně zrychlil.

Jelikož má aplikace prioritně zpracovávat videosekvence, soustředila se druhá sada testů právě na zpracování videosekvencí. Z těchto testů vyplynula řada poznatků. Prvním byl problém s detekcí špičky, občas se stávalo, že aplikace nenašla špičku, a proto byla ve výsledném videu kromě čepele

vidět i špička imitace. Dalším problémem byl detekce bodů, které nepatřily na imitaci a vznikal nepříjemný efekt, kdy čepel přeskakovala po celé scéně. Posledním z výrazných problémů bylo třepání celé čepele a trhané prodlužování a zkracování špičky a zakončení u jílce. Tyto problémy byly vyřešeny pomocí postupů popsaných v kapitole 11. Problému s prodlužováním a zkracování se nepodařilo plně odstranit, dosažený výsledek je však nesrovnatelně lepší, než byl výsledek před zavedením průměrování.

Dalším problém se týká způsobu opravy chyb. Pokud je například poslední správný snímek uložen během pohybu čepele směrem dolů a následující sada snímků, při kterých se čepel zastaví a začne se pohybovat směrem nahoru, je vyhodnocena jako chybná. Potom není možní rekonstruovat tento pohyb a na malý okamžik se vložený efekt čepele zastaví, zatímco imitace se pohybuje nejdříve dolů a poté nahoru. Tento problém se nepodařilo spolehlivě opravit. Jedinou možností je změna nastavení prahů a opětovné spuštění programu.

Aplikace také není schopná korektně zpracovat situaci, kdy je část čepele překryta cizím předmětem. V tomto případě aplikace vykreslí výsledný efekt čepele světelného meče přes tento cizí předmět (Obrázek 19).

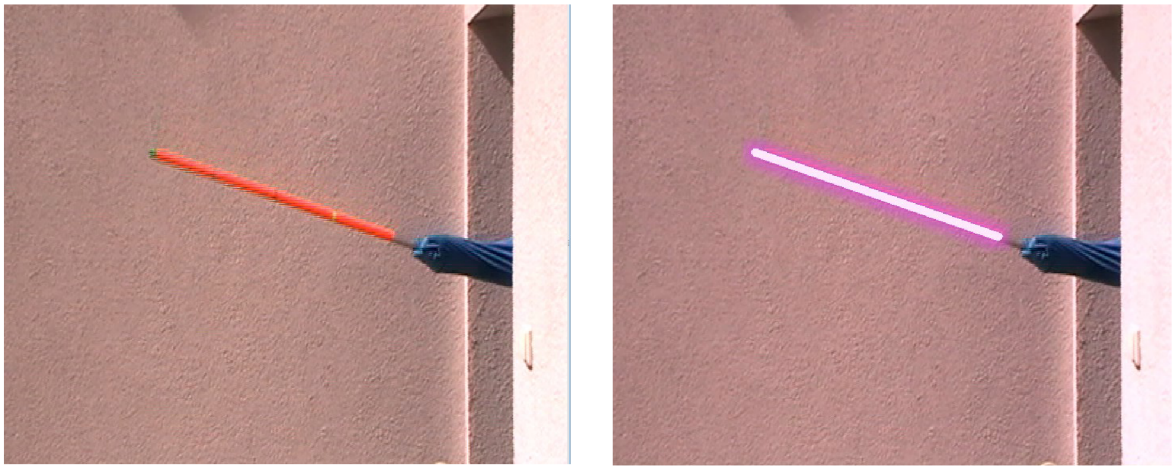


Obrázek 19: Ukázka čepele při částečném překrytí

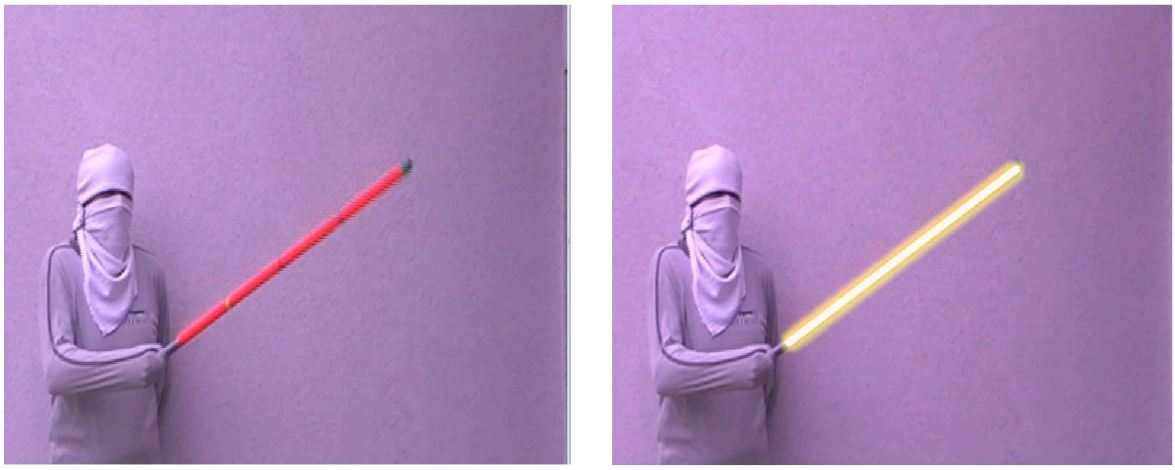
Naše aplikace má oproti postupům popsaných v úvodní kapitole nepochybně ohromnou výhodou v automatické zpracování celé videosekvence. I když rychlost provádění aplikace není ohromující, videosekvenci o rozlišení 704 pixelů na 576 pixelů o délce šesti vteřin a 25 snímcích za vteřinu zpracovává přibližně jednu minutu, stále se jedná o mnohem rychlejší zpracování než při ruční úpravě každého snímku zvlášť.

Jelikož aplikace detekuje imitaci čepele pomocí barevné segmentace, není možné, aby se ve zpracovávaném snímku nebo videosekvenci objevila barva podobná barvě imitace čepele. Aplikace je

schopná filtrovat nežádoucí shluky pixelů, tato schopnost je však limitována velikostí těchto shluků, což blízce souvisí s rozlišením zpracovávaného snímku. Obecně zde platí pravidlo: Čím vyšší je rozlišení snímku či videosekvece, tím lepšího výsledku je aplikace schopna dosáhnout. U menších rozlišení je vyšší pravděpodobnost, že nežádoucí pixely nebudou rozpoznány a celá detekce, potažmo výsledek aplikace je značně znehodnocen. Konečný vzhled výsledného snímku nebo videosekvence je také značně ovlivněn samotným uživatelem. Jestliže uživatel nesprávně nastaví prahy pro detekce barev není možné zaručit dobrý výsledek. S vhodným nastavením jsou ovšem výsledky velmi uspokojivé (Obrázek 20 a 21).



Obrázek 20: Ukázka jednoho snímku videosekvence před a po zpracování



Obrázek 21: Ukázka dalšího snímku videosekvence před a po zpracování

Díky využití knihovny OpenCV (kapitola 12) byla velmi usnadněna práce s videosekvencemi a bylo možné věnovat více času samotnému zpracování jednotlivých snímků. Úskalím při použití této knihovny pro načítání videosekvencí je velmi malý rozsah videosouborů, který je aplikace schopna úspěšně načíst. Lze pracovat pouze se soubory s příponou AVI, které musí být uloženy jako nekomprimované RGB nebo nekomprimované YUV [6].

14 Závěr

Cílem této bakalářské práce bylo prostudovat existující techniky pro tvorbu efektů světelných mečů ve videu a navrhnout aplikaci, která by celý tento proces automatizovala. Navrženou aplikace bylo nutné též implementovat a na základě testování shrnout výhody a nevýhody navrženého řešení.

Klíčovým problémem celé aplikace je detekce imitace čepele světelného meče popsané v kapitole 6. Bez správně nalezené imitace nebo při nalezení nežádoucích pixelů nedosáhneme uspokojivého výsledku. Detekce za použití barevného modelu HSV byla použita, neboť dosahovala lepších výsledků než metoda založená na barevném modelu RGB. Zvolené řešení, včetně odstranění nežádoucích pixelů pomocí morfologických operací (Kapitola 3), se ukázalo být použitelné a jeho výstup dostatečně kvalitní pro další zpracování.

Objekt vzniklý po dokončení detekce imitace čepele nebyl vůbec vhodný pro další zpracování, a proto bylo rozhodnuto jej nahradit přímkou, která bude tvořit jeho středovou osu. Ideálním prostředkem se stala vzdálenostní mapa, díky níž lze získat požadovanou přímku (Kapitola 7).

Po získání přímky tvořící osu objektu je nejdříve vytvořena samotná čepel se zakulaceným zakončením na straně špičky a hranatým koncem na straně jílce (Kapitola 8). Následně je vytvořen barevný efekt okolo čepele a jako poslední jsou určeny stupně průhlednosti toho efektu (Kapitola 9). Výsledný obrázek díky nastaveným průhlednostem působí mnohem přirozenějším dojmem, na rozdíl od použití pouze jednoho stupně průhlednosti, či žádné průhlednosti (Kapitola 10). Hodnoty průhlednosti byly zjištěny metodou pokusů a omylů. Konečné hodnoty průhledností se nejvíce blíží kýženému efektu, kdy se barevný efekt směrem od čepel pomalu vytrácí do ztracena. Náhradním řešením výše popsaného postupu pro tvorbu čepele a barevného efektu by bylo využití grafické knihovny OpenGL. Jelikož však původní postup vytváří požadovaný efekt, nebylo k řešení pomocí OpenGL vůbec přistoupeno.

Pro zpracování videa bylo nutné přidat do aplikace další pomocné funkce, které zajistí kvalitu výsledného efektu (Kapitola 11).

Veškerým chybám a chování aplikace v různých situacích je věnována kapitola o testování (Kapitola 13.2).

Aplikace je v nynější podobě schopna zpracovávat videosekvence i jednotlivé obrázky a vkládat do nich efekt čepele světelného meče s šesti různými barvami – červenou, modrou, zelenou, žlutou, fialovou a stříbrnou [2].

Do budoucna by bylo vhodné upravit chování aplikace při překrytí části imitace čepele jiným objektem a také rozšířit aplikaci na zpracování více světelných mečů v jedné scéně.

---

2 Všechny vyjmenované barvy čepelí se ve filmech či knihách z prostředí Hvězdných válek vyskytují [9].

Literatura

[1] *Gimpology* [online]. 2007-21-07 [cit. 2010-05-05]. How to create Lightsaber effects . Dostupné z WWW: <http://gimpology.com/submission/view/how_to_create_lightsaber_effects>.

[2] Klekner, Martin. *Grafika On-line* [online]. 2009-09-04 [cit. 2010-05-05]. Adobe After Effects tutoriál: Tvorba světelného meče. Dostupné z WWW: <http://www.grafika.cz/art/vse/aae-svetelnymec-1.html>.

[3] Beneš, B., Felker, P., Sochor, J., Žára., J.: *Moderní počítačová grafika*, Brno, Computer Press 2004, ISBN 80-251-0454-0

[4] Hlaváč, V.:*Matematická morfologie* [online]. 2008-03-20 [cit. 2010-05-05]. Dostupné z WWW: <http://cmp.felk.cvut.cz/~hlavac/TeachPresCz/11DigZprObr/71-03BinMatMorfolCesky.pdf>.

[5] Jiří Král: *Převod křivky z rastru na vektorovou representaci*, bakalářská práce, Brno, FIT VUT v Brně, 2007.

[6] *OpenCV Wiki* [online]. 2010-04-06 [cit. 2010-05-05]. Dostupné z WWW: <http://opencv.willowgarage.com/wiki/Welcome>.

[7] *OpenCV Reference Manual* [online]. 2010-03-18 [cit. 2010-05-05]. Dostupné z WWW: <https://code.ros.org/svn/opencv/trunk/opencv/doc/opencv.pdf>.

[8] *Wookieepedia* [online]. 2010-05-05 [cit. 2010-05-05]. Dostupné z WWW: <http://starwars.wikia.com/wiki/Main_Page>.

[9] *Wikipedia* [online]. 2010-05-05 [cit. 2010-05-05]. Dostupné z WWW: <http://www.wikipedia.org/>.

Seznam příloh

Uživatelská příručka

Následující stránky seznámí uživatele s instalací a ovládáním aplikace pro tvorbu efektů světelných mečů ve videu.

Instalace programu

Program je určen pro operační systém Linux. Pro jeho instalace je nutné mít nainstalovánu a správně nakonfigurovánu knihovnu OpenCV. Dalším nutným požadavkem je překladač g++ a knihovny programovacího jazyka C++. Program se instaluje překladem za použitím přiloženého souboru Makefile. V případě problémů s připojením knihovny OpenCV je nutné změnit cesty k souborům knihovny přímo v Makefilu.

Spuštění programu

Program pracuje jako konzolová aplikace. Při spuštění programu je nutné zadat všechny nutné parametry: *lightsabre -i [název] -s [obrázek/video] -o [název] -c [barva] -h*

Význam jednotlivých parametrů:

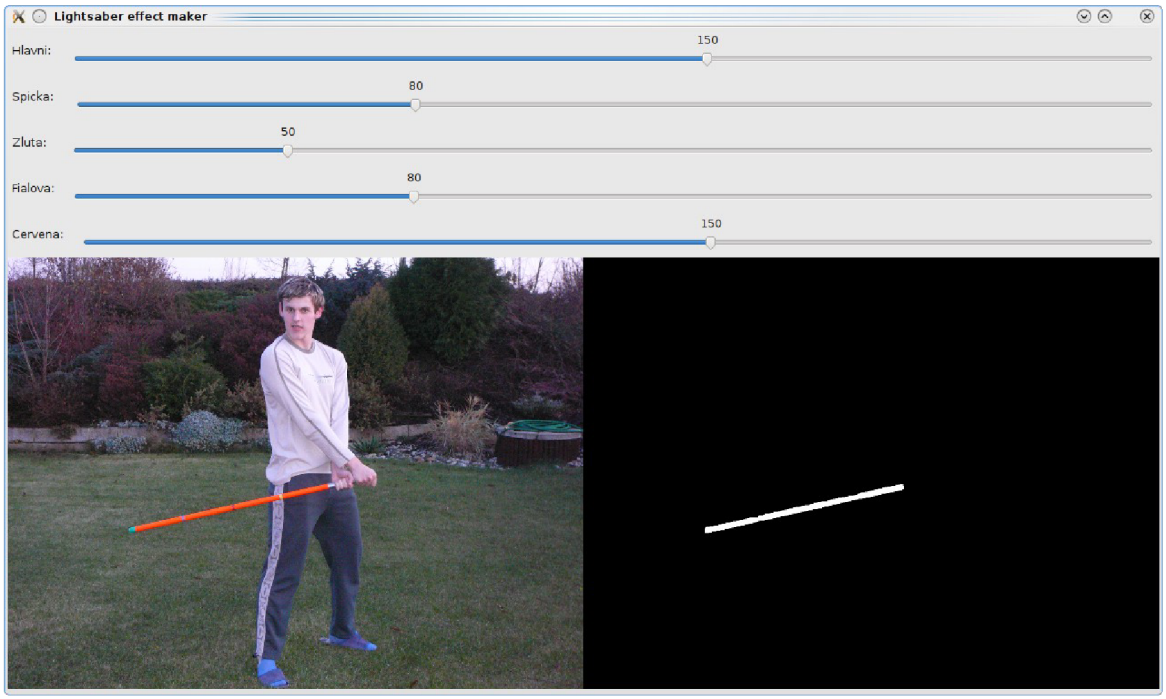
-h	*vypíše nápovědu*
-i [název]	*jméno vstupního souboru*
-s [i, v]	***i** pokud je vstup obrázek, **v** pokud je vstup video*
-o [název]	*jméno výstupního souboru*
-c [r, g, b, v, y, s]	*nastaveni barvy čepele – **r**: červená*
	***g**: zelená*
	***b**: modrá*
	***v**: fialová*
	***y**: žlutá*
	***s**: stříbrná*

Úvodní okno pro nastavení prahů barev

Úvodní okno programu slouží k upravení nastavení prahů pro detekci barev. Je nutné tyto prahy nastavit tak, aby uživatel viděl pouze detekovanou čepel. Detekce jiných části scény jsou nežádoucí a mají za vliv nesprávný výsledek.

Úvodní okno (Obrázek 1) obsahuje pět posuvníků pro nastavení prahů. První slouží pro nastavení hlavní barvy imitace, druhý pro špičku a zbylé tři pro barevné značky umístěné na čepeli.

Po úspěšném nastavení se klávesou S nebo ENTER spustí hlavní program, který nyní již nevyžaduji žádnou interakci s uživatelem. Z úvodního okna se dá program ukončit klávesou Q nebo ESC.



Obrázek 1: Okno pro nastavení prahů detekce

Náhled plakátu

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Automatická tvorba efektu světelného meče

Autor: Ondřej Vagner

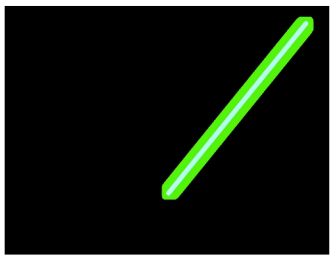
Aplikace určená pro automatickou tvorbu efektů světelných mečů ve videu.

Automatické zpracování celého videa bez nutnosti zásahu uživatele, který pouze nastaví program.

Videa natočena se speciální imitací čepele světelného meče, která se bude detekovat a nahrazovat.

Detekce imitace čepele pomocí barevné segmentace.

Zpracování výsledného objektu vzdálenostní transformací.



Vykreslení čepele pomocí silné čáry se zakulacenou špičkou.

Vytvoření barevné korony díky jednoduchému rozmazání.

Nastavení různé průhlednosti barevné korony.

Možnost vytvoření šesti různých barev:
červené, modré, zelené, žluté, fialové a stříbrné.

Obrázky z Hvězdných válek jsou převzaty ze stránek http://starwars.wikia.com/wiki/Main_Page

Obsah přibaleného CD

- Zdrojové soubory
- Spustitelný program
- Programová dokumentace
- Testovací obrázky a videa
- Ukázky výsledků po zpracování programem
- Text Bakalářské práce ve formátech odt a pdf