VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ODSTRANĚNÍ NEŽÁDOUCÍCH OBJEKTŮ VE VIDEOSEKVENCÍCH

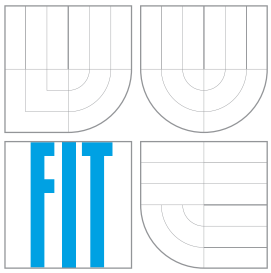
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ONDŘEJ VAGNER

BRNO 2012

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ODSTRANĚNÍ NEŽÁDOUCÍCH OBJEKTŮ VE VIDEOSEKVENCÍCH

REMOVING OF UNWANTED OBJECTS IN THE VIDEOSEQUENCES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ONDŘEJ VAGNER

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. PAVEL ŽÁK

BRNO 2012

Abstrakt

Cílem této práce bylo vytvoření automatické metody pro odstranění nežádoucích objektů z videosekvencí. Navržená metoda je schopná odstranit statický i pohybující se objekt bez zásahu uživatele do procesu zpracování. Uživatel pouze definuje objekt určený k vymazání.

Abstract

The aim of this work was to develop an automated methods for removing unwanted objects from video sequences. The proposed method is able to autonomously tackle the static and the moving object with no user intervention into the process. The user only determines the object to deleted.

Klíčová slova

Odstranění objektu z fotografie, odstranění objektu z videosekvence, optický tok, syntéza textur, image inpainting.

Keywords

Removing an object from a photograph, removing an object from video sequences, optical flow, texture synthesis, image inpainting.

Citace

Ondřej Vagner: Odstranění nežádoucích objektů ve videosekvencích, diplomová práce, Brno, FIT VUT v Brně, 2012

Odstranění nežádoucích objektů ve videosekvencích

Prohlášení

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

.....
Ondřej Vagner
20. května 2012

Poděkování

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

© Ondřej Vagner, 2012.
Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

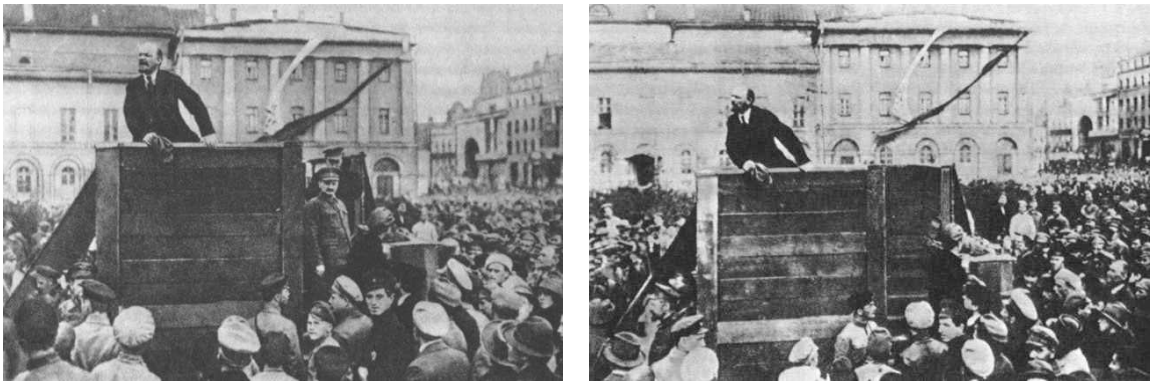
Obsah

Kapitola 1

Úvod

Při pořizování fotografií či videozáznamu může velmi snadno dojít k jeho "znečištění" nechtěnými prvky, které někdy nemusí být na první pohled vůbec znát, a jejich přítomnost odhalíme až později. Většinou se jedná o menší objekty v pozadí snímané scény. Uživatelé však požadují i možnosti odstraňování větších objektů ze svých fotografií nebo videosekvencí. Problém vyplnění prázdné oblasti zanechané tímto objektem není triviální a je mnohdy i velmi výpočetně náročný.

Z historického hlediska jsou úpravy fotografií (neboli takzvané retušování) stejně staré jako je fotografie sama. Ať už se jednalo o mírné úpravy a opravy drobných nedokonalostí či mazání celých, mnohdy vskutku nežádoucích, osob. Typickým příkladem je fotografie pořízená 20. května 1920, kdy Lenin na snímku hovoří na Sverdlovském náměstí a po jeho boku stojí Lev Davidovič Trockij a Lev Borisovič Kameněv. Ze snímku byli později Trockij a Kameněv komunistickým režimem odstraněni (Obrázek 1.1). Úprava fotografií byla v těchto dobách spíše umělecká nežli technická práce. S nástupem filmu se začaly podobné retuše provádět i pro filmové sekvence, kdy musel umělec pracovat na filmovém pásu doslova okénko po okénku. S příchodem digitálního věku se úprava fotografií stala méně komplikovanou činností a stávala se též více dostupnou větší části populace. Postupně začaly vznikat specializované počítačové programy určené pro úpravy fotografiií a videosekvencí.



Obrázek 1.1: Původní a retušovaný snímek [30]

V současné době je mnoho možností pro odstranění nežádoucích objektů z fotografií a to jak manuálních [9], tak plně automatizovaných [6]. Manuální techniky jsou založeny na využívání nástrojů pro kopírování pozadí z jiné části scény, přičemž vše je plně v režii

uživatele, který ruční mazání provádí.

Existují i postupy na odstranění nežádoucích objektů z videosekvencí, které využívají běžné video editory a grafické editory [3]. Tento přístup však není plně automatizován a vyžaduje velkou spoluúčast uživatele.

Cílem této práce je navrhnout a vytvořit aplikaci, která bude schopna nežádoucí objekt z videosekvence, případně fotografie odstranit plně automaticky. Po uživateli bude vyžadováno pouze označení objektu, který bude následně vymazán.

V následujícím textu bude nejdřive uvedena teorie k problematice odstraňování objektů z fotografií (Kapitoly 2 a 3) a sledování objektů ve videosekvencích (Kapitola 4). Návrhu aplikace je věnována kapitola 5, postupem pro odstranění objektu z fotografie a zpracováním videosekvence se zabývají kapitoly 6 a 7. Průběh a výsledky testování jsou popsány v kapitole 9.

Kapitola 2

Odstranění objektu ze statického snímku

Tato části práce se bude zabývat metodami na odstranění objektů ze statických snímků – fotografií. Hlavním a jediným cílem algoritmů řešících tento problém je vyplnění oblasti, která vznikne po vymazání objektu, vhodným pozadím. Existují dva hlavní směry řešení a to syntéza textur [5] [8] a image inpainting [10] [1].

2.1 Syntéza textur

Textura je vzor v obraze, který je rozšířený, snadno rozpoznatelný a těžko definovatelný. Zda efekt nazveme texturou nebo ne, závisí na měřítku, ve kterém je zobrazen. O texturách lze říct, že se jedná o obraz splňující určité podmínky a zároveň zachovávající určitou strukturu. Z tohoto vyplývají specifické vlastnosti dané textury. Mezi ně patří návaznost části textury na jiné části textury, prostorové vztahy těchto částí a také vysoká pravděpodobnost jejich vzájemné podobnosti. Textury lze rozlišovat v různých stupních od regulární po stochastické. Regulární textury jsou ty, v nichž jsou jasně patrné deterministické vztahy mezi objekty – prostorové a podobnostní. Příkladem takových textur je pravidelná mřížka nebo šachovnice. Naopak stochastická textura je taková, kde jsou vzájemné vztahy obtížně definovatelné a rozeznání jednotlivých objektů je velmi problematické. Takovými texturami je například koberec nebo písečná pláž (Obrázek 2.1).

Existují tři základní operace, které se s texturami provádí, a to segmentace textury, získání tvaru z textury a syntéza textur [8].

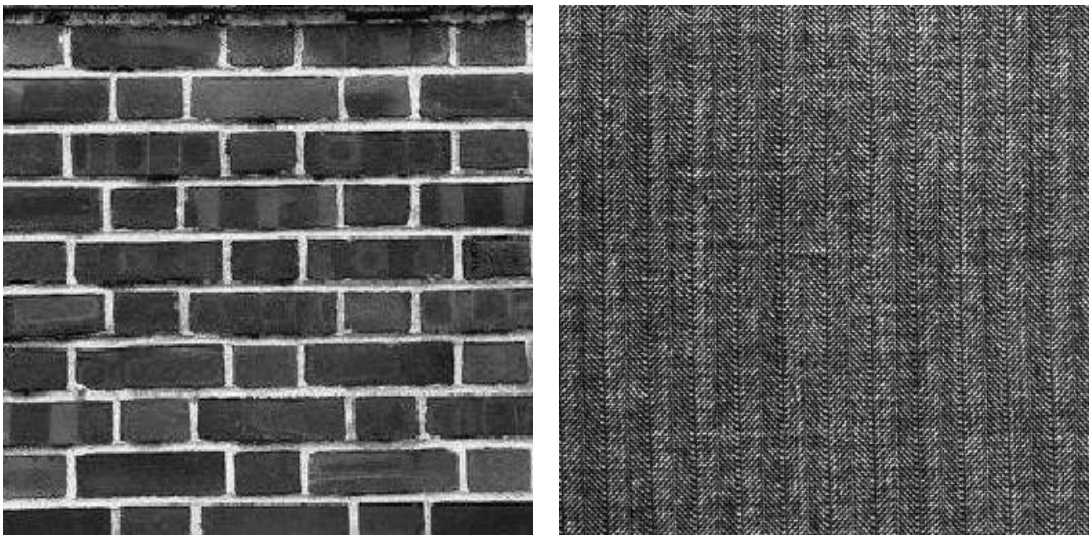
Syntéza textur je proces algoritmické výstavby velkého digitálního obrazu z malého digitálního vzorku obrazu pomocí využití jeho strukturálního obsahu. V oblasti počítačové grafiky je využíván v mnoha oblastech, jako je úprava digitálního obrazu, 3D počítačové grafiky a nebo post-produkce filmů.

Výsledný vzhled a forma textury může být zadána pomocí gramatiky. Tento způsob je používám jenom v případě, kdy se jedná o textury obsahující konkrétní, opakující se objekty. Běžnějším způsobem zadání vzhledu textury je použití zdrojové textury, která má být rozšířena na větší plochu nebo pomocí níž má být zaplněna mezera v obraze.

Techniky syntézy textur pro potřebu vyplnění oblasti po odstraněném objektu mohou být klasifikovány do tří kategorií. Prvním je syntéza textury na základě simulace fyzikální procesu jejího generování. Druhou možností je vytvoření parametrického modelu za pomoci analýzy vstupní textury a následná syntéza textury výstupní. Třetím typem je vytvoření

výsledné textury z malého vzorku [13].

Syntéza textur je využívána při vyplnění oblasti vzniklé při odstranění rozměrnějšího objektu. V případě malých oblastí (například škrábanců) bývá využívána metoda Image inpainting.



Obrázek 2.1: Příklad regulární [19] a stochastické textury [20]

2.2 Image inpainting

Inpainting je uměleckým synonymem pro interpolaci obrazu a je využíván zejména muzejními restaurátory již velmi dlouhou dobu. Technika inpaintingu je velmi subjektivní. Záleží na restaurátorovi a konkrétním obraze, nicméně existují čtyři základní principy

- obraz musí působit jednotně
- linie vstupující do chybějící oblasti musí být vhodně doplněny
- regiony ohraničené liniemi jsou vyplněny příslušnou barvou z okraje neznámé oblasti
- jsou dokresleny detaily

Autoři algoritmů se zaměřili na druhý a třetí bod a pokusili se je převést do diskrétního prostoru. Pojem digitální inpainting se poprvé objevil v [1]. Algoritmy využívající jednoduché metody inpaintingu mají velké problémy s vyplněním větších regionů, jejich výsledky jsou většinou značně rozmazané, a tudíž nevěrohodné. Existují i více sofistikované metody využívající image inpainting [4].

Kapitola 3

Algoritmus pro odstranění objektu z fotografie

V současné době existuje řada metod a postupů pro vyplnění prázdného místa v obraze vzniklého po vymazání nějakého objektu. Tyto metody jsou většinou založeny na jednom z principů popsaných v předchozí kapitole.

3.1 Hybridní algoritmus s mechanismem pro detekci artefaktů

Tento algoritmus je popsán v [13] [12] a jeho hlavní výhodou oproti ostatním postupům [4] je použití dvou různých technik pro vyplnění mezery vzniklé vymazáním objektu. Algoritmus využívá sub–patch syntézu textur (sub–patch texture synthesis) a váhovanou interpolační metodu (weighted interpolation method). V dalších částech textu bude bílé místo vzniklé po odstranění objektu (Obrázek 3.1) nazýváno mezerou.

Obrázek 3.1: Vstupní snímek s označeným nechtěným objektem a mezera určená k vyplnění

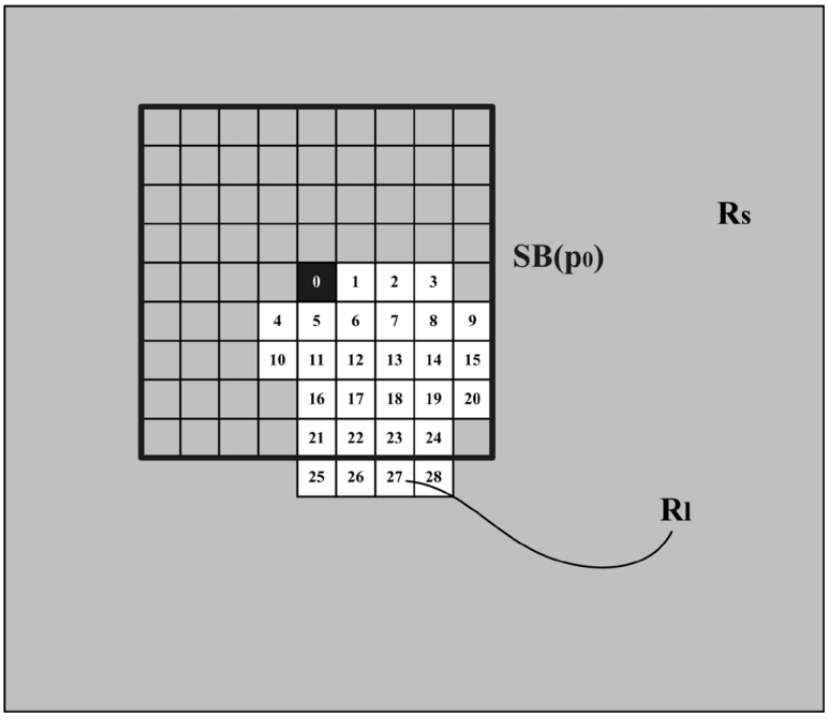
3.2 Analýza distribuce barev v textuře

Prvním krokem metody je analýza textury v okolí mezery, na základě jejichž výsledků je následně rozhodnuto o použití jedné z technik. V předchozích technikách využívajících exemplar-based inpainting [4] zabral výpočet nejlepších vzorků mnoho času a vedl ke značnému zpomalení celého výpočtu a to zejména u stochastických textur. Na základě této analýzy je rozhodnuto, která z technik bude pro danou oblast použita. Samotná analýza se skládá ze tří částí – výpočtu charakteristické hodnoty, vytvoření klasifikační mapy a výběru metody na základě klasifikační mapy.

Výpočet charakteristické hodnoty se provádí následovně. Pro každý pixel mezery je za pomocí čtvercové masky o délce strany 9 pixelů vypočtena hodnota $\alpha(p_i)$ (Obrázek 3.2). Charakteristická hodnota je určena jako suma směrodatných odchylek RGB kanál:

$$\begin{aligned} \alpha(p_i) &= \sqrt{\frac{\sum(R(p_k)-\bar{R}(p_k))^2}{K_i}} + \sqrt{\frac{\sum(G(p_k)-\bar{G}(p_k))^2}{K_i}} + \sqrt{\frac{\sum(B(p_k)-\bar{B}(p_k))^2}{K_i}} \\ \forall p_k &\in SB(p_i) \cap R_S \cap R_l \end{aligned} \tag{3.1}$$

Kde K_i je počet platných pixelů (tedy těch, které nenáleží mezeře), $R(p_k), G(p_k), B(p_k)$ jsou hodnoty pixelů masky v bodě k a $\bar{R}(p_k), \bar{G}(p_k), \bar{B}(p_k)$ je střední hodnota barev platných pixelů masky.



Obrázek 3.2: Znázornění masky pro analýzu distribuce barev [13]. Rl jsou pixely mezery a Rs jsou pixely obrazu

Po dokončení výpočtu charakteristických hodnot je každý pixel mezery označen, čímž je vytvořena klasifikační mapa. Pokud je hodnota pixelu vyšší než stanovený práh (v [13] je

uváděna hodnota 65), bude použita sub–patch syntéza textur (používá se pro nehomogenní textury), v opačném případě bude region vyplněn pomocí váhované interpolační metody (homogenní textury). Pokud se v řádku vyskytuje alespoň jeden pixel indikující nehomogenní texturu, bude pro tento řádek použita syntéza textur.

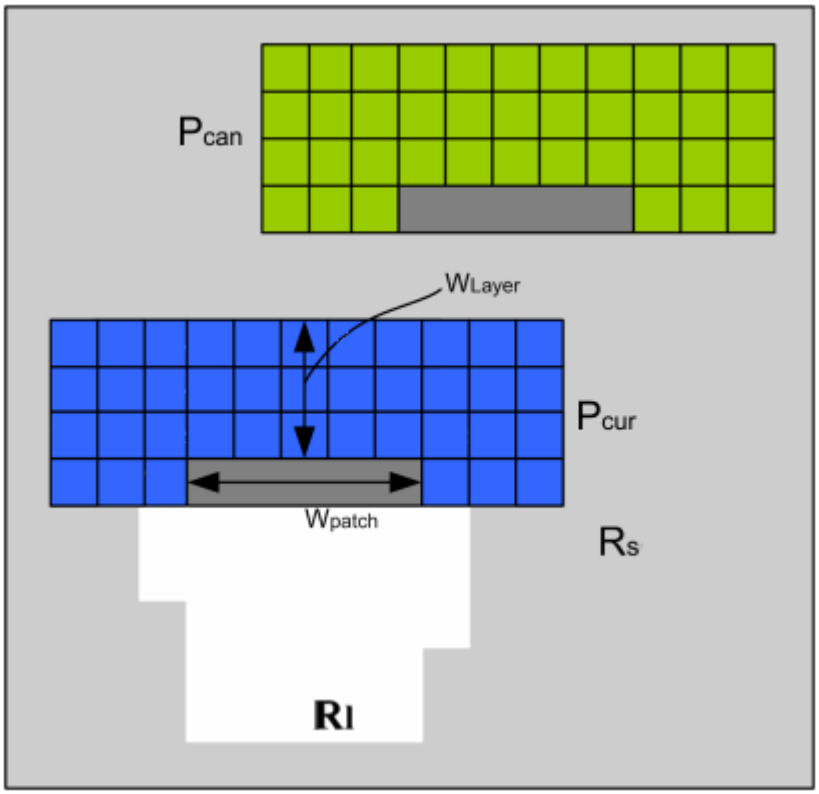
Algoritmus zapracovává mezeru po jednotlivých řádcích. Jakmile je vytvořena klasifikační mapa pro aktuální řádek, provede se ihned jeho vyplnění.

3.3 Sub–patch syntéza textur

U nehomogenních textur není možné provést nahrazení řádku jedním ostínem barvy. Je nutné nalézt ideální část (nejpodobnější) v existující textuře a tou následně zaplnit mezeru. Tuto operaci provádí sub–patch syntéza textur.

Na rozdíl od pixel-based syntéz textur vkládá tato technika vždy celý řádek, což je efektivnější řešení. Metoda vyhledá vzorek textury s největší podobností a tím je nahrazena část mezery (Obrázek 3.3).

Velikost okolí vzorku má vliv na kvalitu výsledného obrazu. Prvotní hloubka je nastavena na velikost 3 a může být zvětšována, dokud není získán dostatek informací o sousedech. Množství sousedních pixelů je určeno následovně:



Obrázek 3.3: Princip sub–patch metody [12]. Rl jsou pixely mezery a Rs jsou pixely obrazu

$$KW_{patch} + \sum_{i=1}^{K} 4i \tag{3.2}$$
$$kde: \ K \ \ je\ hloubka\ vzorku\ W_{layer}$$

V každé kroku je vybraný kandidátní vzorek (P_{can}) porovnán se vzorkem aktuálním (P_{cur}). Sousední pixely těchto dvou vzorků jsou použity pro výpočet jejich podobnosti. Je nutný převod obrazu i do barevného prostoru YCbCr, protože je porovnávána i úroveň jasu mezi kandidátním a aktuálním vzorkem. Podobnost $N(P_{can})^*$ mezi vzorky $N(P_{can})$ a $N(P_{cur})$ je vypočtena pomocí následujících vztahů, kde pk' označuje k–tý pixel náležející $N(P_{can})$ a pk'' pixel z $N(P_{cur})$.

$$N(P_{can})^* = \arg \min_{N(P_{can}) \in R_S} (d_C(N(P_{can}), N(P_{cur})) + \bar{d_C}(N(P_{can}), N(P_{cur}))) \tag{3.3}$$

$$d_C(N(P_{can}), N(P_{cur}) = \sum \omega_{C(i,j)}((R_{(pk')} - R_{(pk'')})^2 + (G_{(pk')} - G_{(pk'')})^2 + (B_{(pk')} - B_{(pk'')})^2)$$

$$kde: \ \omega_{C(i,j)} = \begin{cases} 30, & pokud\ j = K \\ 20, & j = K+1 \\ 5, & j = K+2 \\ 0, & j > K+2 \end{cases} \tag{3.4}$$

$$\bar{d_C}(N(P_{can}), N(P_{cur}) = \sum \gamma((R_{(pk')} - R_{(pk'')})^2 + (G_{(pk')} - G_{(pk'')})^2 + (B_{(pk')} - B_{(pk'')})^2)$$

$$\gamma = (|Y_{(pk')} - Y_{(pk'')}| + 1)^{-1} \tag{3.5}$$

3.4 Váhovaná interpolační metoda

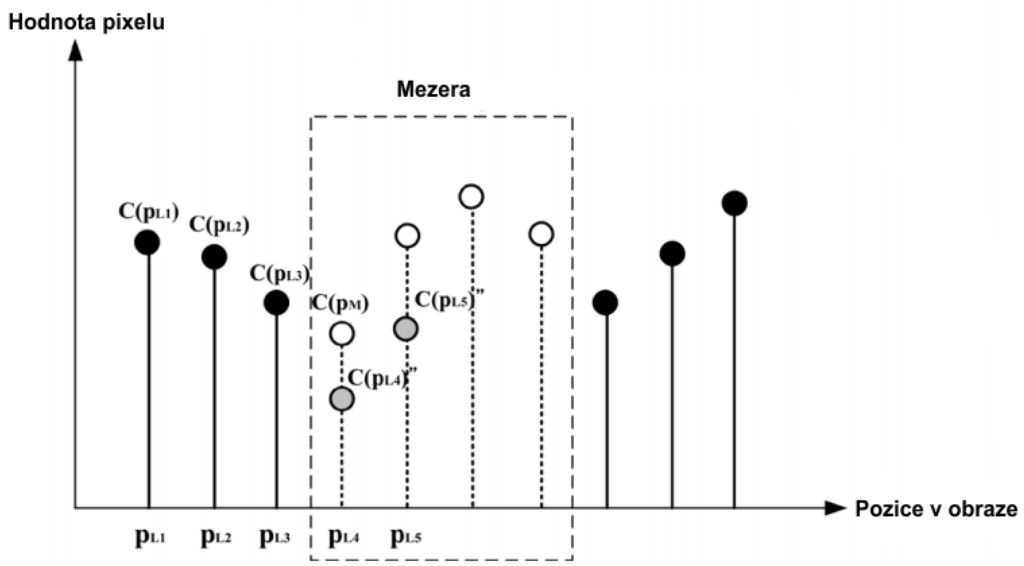
Na rozdíl od nehomogenních textur, je u homogenní textury možné provést nahrazení řádku mezery pomocí jednoho odstínu barvy. Metoda vychází z předpokladu, že pozadí obrazu z reálného světa se skládá převážně s horizontálních textur. Pro výpočet pixelů uvnitř mezery se využívají pixely ze stejného řádku obrazu (Obrázek 3.4).

Výpočet se provádí zleva doprava dle následujících vztahů:

$$C_{(p_M)} = \sum_{i''=-2}^{0} W_{i''} C(p_{L_{i''+3}}) + \sum_{i''=1}^{2} W_{i''} C''(p_{L_{i''+3}}) \tag{3.6}$$
$$W_{-2} = W_2 = 0.05$$
$$W_{-1} = W_1 = 0.25$$
$$W_0 = 0.4$$

$$C''(p_{L_4}) = 2C(p_{L_3}) - C(p_{L_2}) \tag{3.7}$$
$$C''(p_{L_5}) = 2C(p_{L_3}) - C(p_{L_1})$$
$$pokud: \; p_{L_4}, \; p_{L_5} \; \in \; R_l$$

$C_{(p_M)}$ je vypočtená váhovaná hodnota pixelu uvnitř mezery. Již vypočtené pixely se dále používají pro výpočet dalších, dokud není celý řádek mezery vyplněn.



Obrázek 3.4: Váhovaná interpolační metoda [13]

3.5 Detekce artefaktů

Artefaktem je myšlena taková část syntetizované textury, která nezapadá do svého okolí. Typicky se může jednat o náhle změny barvy nebo nežádoucí hrany vznikající většinou na hranicích řádků vyplněných pomocí váhované interpolace a sub–patch syntézy textur.

Mechanismus je založen na podobnosti textur. Po vyplnění celé mezery je na oblast aplikována automatická detekce artefaktů. Vstupní obraz je rozdělen na bloky 8x8 pixelů. Je přidána podmínka offsetu (Vzorec 3.8) pro výpočet gradientu barvy v každém pixelu (Vzorec 3.9). Každý blok v cílovém regionu je porovnán s ostatními bloky pomocí průniku histogramů (Vzorec 3.10).

$$
\begin{aligned}
M_1(R^{\overrightarrow{x_1}}, R^{\overrightarrow{x_2}}, G^{\overrightarrow{x_1}}, G^{\overrightarrow{x_2}}) &= \frac{R^{\overrightarrow{x_1}} G^{\overrightarrow{x_2}} - R^{\overrightarrow{x_2}} G^{\overrightarrow{x_1}}}{R^{\overrightarrow{x_2}} G^{\overrightarrow{x_1}} + R^{\overrightarrow{x_1}} G^{\overrightarrow{x_2}} + 1} \\
M_2(R^{\overrightarrow{x_1}}, R^{\overrightarrow{x_2}}, B^{\overrightarrow{x_1}}, B^{\overrightarrow{x_2}}) &= \frac{R^{\overrightarrow{x_1}} B^{\overrightarrow{x_2}} - R^{\overrightarrow{x_2}} B^{\overrightarrow{x_1}}}{R^{\overrightarrow{x_2}} B^{\overrightarrow{x_1}} + R^{\overrightarrow{x_1}} B^{\overrightarrow{x_2}} + 1} \\
M_3(G^{\overrightarrow{x_1}}, G^{\overrightarrow{x_2}}, B^{\overrightarrow{x_1}}, B^{\overrightarrow{x_2}}) &= \frac{G^{\overrightarrow{x_1}} B^{\overrightarrow{x_2}} - G^{\overrightarrow{x_2}} B^{\overrightarrow{x_1}}}{G^{\overrightarrow{x_2}} B^{\overrightarrow{x_1}} + G^{\overrightarrow{x_1}} B^{\overrightarrow{x_2}} + 1}
\end{aligned} \tag{3.8}
$$

$$
\begin{aligned}
\nabla C_M(C^{\overrightarrow{x_1}}, C^{\overrightarrow{x_2}}, C^{\overrightarrow{x_1}}, C^{\overrightarrow{x_2}}) &= (M(C_1^{x-1,y}, C_1^{x+1,y}, C_2^{x-1,y}, C_2^{x+1,y})^2 \\
&+ M(C_1^{x,y-1}, C_1^{x,y+1}, C_2^{x,y-1}, C_2^{x,y+1})^2)^{\frac{1}{2}}
\end{aligned} \tag{3.9}
$$

$$
D(H_1, H_2) = \frac{\sum_{\overrightarrow{k}=1}^{N} H_1(\overrightarrow{k}) H_2(\overrightarrow{k})}{\sum_{\overrightarrow{k}=1}^{N} (H_1)^2} \tag{3.10}
$$

Blok je označen jako artefakt, pokud je hodnota průniku histogramů menší než práh u více než čtyř sousedních bloků. Tímto je zaručeno, že hranice různých textur nebudou označeny jako artefakty.

Jakmile jsou nalezeny a označeny všechny artefakty, je znovu aplikována metoda pro syntézu textur.

Kapitola 4

Sledování objektů ve videosekvenci

Sledování (video tracking) je proces určení polohy pohybujícího se objektu v čase použitím kamery. Algoritmus vytvořený pro sledování analyzuje video snímky a určuje polohu objektu v jednotlivých snímcích [18].

Videosekvence

Nutnou podmínkou pro funkčnost videosekvence je využití nedokonalosti lidského zraku. Pokud je lidský zrak vystaven rychlému sledu po sobě následujících statických snímků, vzniká dojem pohybující se scény. Bylo zjištěno, že minimální hranice snímkové frekvence dostačující pro tento efekt je 25–30 snímků za sekundu [14]. Na videosekvence je v rámci zpracování obrazu tedy nahlíženo jako na sekvenci po sobě jdoucích snímků s malými změnami, čehož lze velmi dobře využít při sledování pohybujícího se objektu.

Pohyb scény

Pohyb na scéně může být způsoben jedním ze tří faktorů. Prvním z nich je pohyb objektu uvnitř scény při statickém umístění kamery. V tomto případě budou vektory pohybu pro pozadí nulové a změna se bude realizovat pouze na vektorech pohybujícího se objektu.

Druhou možností je pohyb kamery při zachování statické scény. Všechny body scény se nyní pohybují stejným směrem a to proti směru pohybu kamery. Vektory pohybu těchto bodů budou mít všechny stejnou velikost a směr.

Třetím případem je kombinace obou předcházejících možností.

4.1 Význačné body

Význačný nebo též významný bod v obraze je souhrné označení takové pro místo v obraze, které splňuje několik podmínek [22]:

- má jasnou a matematicky dobře podloženou definici
- má jasně definovanou pozici v obrazovém prostoru
- lokální struktura v obraze kolem tohoto význačného bodu je bohatá na informace vhodné pro další zpracování

- je stabilní z hlediska působení lokálních a globálních deformací v obrazové doméně tak, aby bylo bod možné opět znovu nalézt

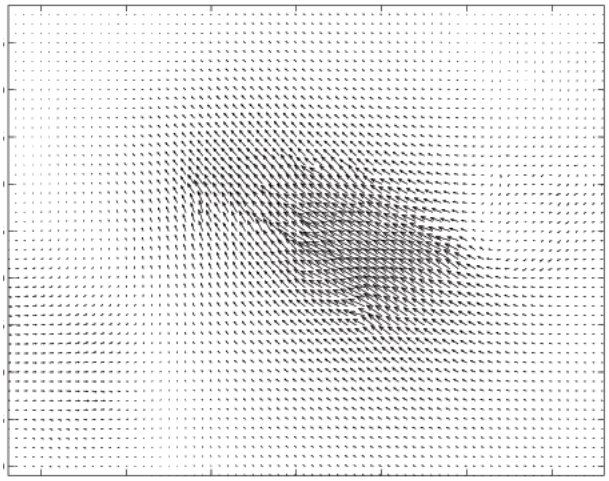
Metody používané pro detekci význačných bodů v obraze se neustále vyvíjí. Jako první se začaly používat hrany a rohy objektů. Jedním z prvních detektorů rohových bodů je Moravcův detektor. Dalším krokem v této kategorii je Harrisův detektor, který je nezávislý na rotaci, posunu a též je odolný vůči šumu. Metody využívající jako význačné body hrany a rohy jsou však nespolehlivé v případě změny měřítka. Pro vyřešení tohoto problému bylo nutné nalézt jiné přístupy.

Metoda SHIFT (Scale Invariant Feature Transform) byla navržena pro extrakci význačných bodůz obrázků a následné spolehlivé vyhledávání společných rysů mezi nimi. Body nalezené touto metodou jsou nezávislé na měřítku, rotaci a metoda nabízí robustní mechanismus porovnání i přes působení afinních deformací, šumu a změny osvětlení.

SURF (Speeded UP Robust Features) je metoda, která se zaměřuje hlavně na rychlost. Autoři se inspirovali metodou SHIFT, ale provedli několik úprav, včetně zmenšení deskriptorů pro popis okolí bodu. Výsledná metoda je vhodná i pro zpracování v reálném čase.

4.2 Optický tok

Změny v obraze způsobené pohybem lze sledovat pomocí optického toku [14] [15]. Optický tok je vektorové pole (Obrázek 4.1), které pro každý bod snímku v časové sekvenci určuje směr a velikost pohybu vzhledem k následujícímu snímku. Je nutné brát na zřetel, že optický tok a jeho změna reprezentuje pohyb v obraze a nikoli pohyb samotného obrazu (Obrázek 4.2).

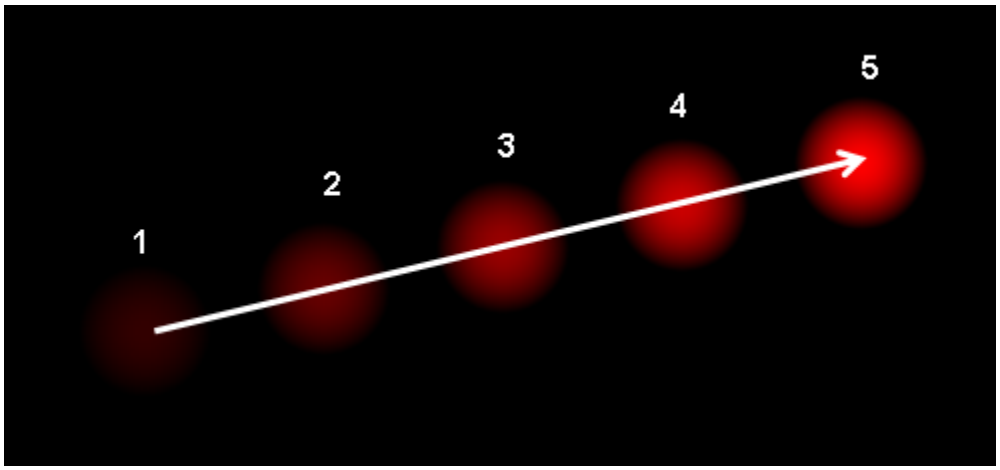


Obrázek 4.1: Vektorové pole [2]

Optický tok se určuje mezi dvěma snímky videosekvence v časech t a $t + dt$, respektive zachycuje všechny změny obrazu za čas dt. Ke zjištění změn v určité lokální oblasti je možné vypočítat a sestrojit parametrický model této oblasti. S použitím parametrizovaných funkcí souřadnic v tomto regionu lze sledovat a přesně popsat změny, ke kterým dochází, a to s malým počtem parametrů. Popisem parametrů v časové ose lze rozeznat a detekovat pohyb regionů (sledování pohybu – tracking).

K výpočtu optického toku lze použít množství přístupů, které se řadí do několika skupin [16] [31]

- diferenční metody
- vyhledávání oblastí
- metody založené na energii
- metody založené na fázi



Obrázek 4.2: Optický tok [16]

4.2.1 Diferenční metody

Jedná se o první techniky pro výpočet optického toku. Jsou založeny na výpočtu parciální diference prvního, či vyššího řádu. Získaná diferenciální rovnice je aproximací předpokladu zachování intenzity. Z této aproximace vychází odhad chyby v optickém toku. Tyto metody neočekávají, že se podaří nalézt optický tok splňující předpoklad zachování intenzity beze zbytku, a proto je zajímá zbytková chyba, kterou se pokouší minimalizovat.

K vyřešení problému apertury (velikost otvoru optické soustavy) se zavádí další chybový člen, který má za úkol propagovat optický tok z části obrazu, kde jej lze spolehlivě odhadnout, do částí trpících nedostatkem gradientu. Některé metody ještě usměrňují propagaci optického toku podle výskytu hran v obraze signalizujících hranice objektů v reálné scéně. S rostoucím počtem omezujících předpokladů dochází k zpřesnění modelu výpočtu a ke snížení vlivu apertury, ale též roste složitost numerického řešení.

Cílem diferenčních metod je minimalizovat celkovou chybu, která v přijatelné míře obsahuje předpoklad zachování intenzity a zároveň vyhovuje reálným omezujícím předpokladům. Metody využívající diference vyšších řádů jsou přesnější, avšak numerický výpočet diferencí vyšších řádů je daleko citlivější na existující chyby v obraze (například šum), což může vést k degradaci metody.

Metoda Horn-Schunck

Metoda byla poprvé publikována v roce 1981 [11]. Jedná se o první metodu využívající předpokladu zachování intenzity. Horn-Schunckova metoda předpokládá hladkost toku celého obrazu – snaží se minimalizovat distorze optického toku a preferovat řešení obsahující více hladkosti.

Tok je formulován jako funkce globální energie, kterou se snažíme minimalizovat. Funkce pro $2D + t$ dimenzionální obraz je definována následovně:

$$f = \int ((\nabla I \cdot V + I_t)^2 + \alpha(|\nabla V_x|^2 + |\nabla V_y|^2))dxdy \tag{4.1}$$

Derivace intenzity obrazu dle složky x a y jsou definovány takto:

$$\nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix} \tag{4.2}$$

Dále ve vzorci 4.1 je I_t derivace dle času $\overrightarrow{V}$ je vektor optického toku s komponentami V_x, V_y. Parametr α je váha kladená na požadavek homogenity optického toku. Čím vyšší je tato hodnota, tím je tok hladší. Tato funkce může být vyřešena výpočtem Euler-Lagrangeovi rovnice, z kterého vychází:

$$\begin{aligned} I_x(I_xV_x + I_yV_y + I_t) - \alpha\Delta V_x = 0 \\ I_y(I_xV_x + I_yV_y + I_t) - \alpha\Delta V_y = 0 \end{aligned} \tag{4.3}$$

Kde δ je Laplaecův operátor:

$$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \tag{4.4}$$

Řešením rovnic pomocí Gaus-Sidelovy metody pro komponenty toku V_x, V_y dostáváme iterační schéma:

$$\begin{bmatrix} I_{x_1} & I_{y_1} \\ I_{x_2} & I_{y_2} \\ . & . \\ . & . \\ . & . \\ I_{x_n} & I_{y_n} \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} -t_1 \\ -t_2 \\ . \\ . \\ . \\ -t_n \end{bmatrix} \tag{4.5}$$

K jejímu řešení lze použít metodu nejmenších čtverců:

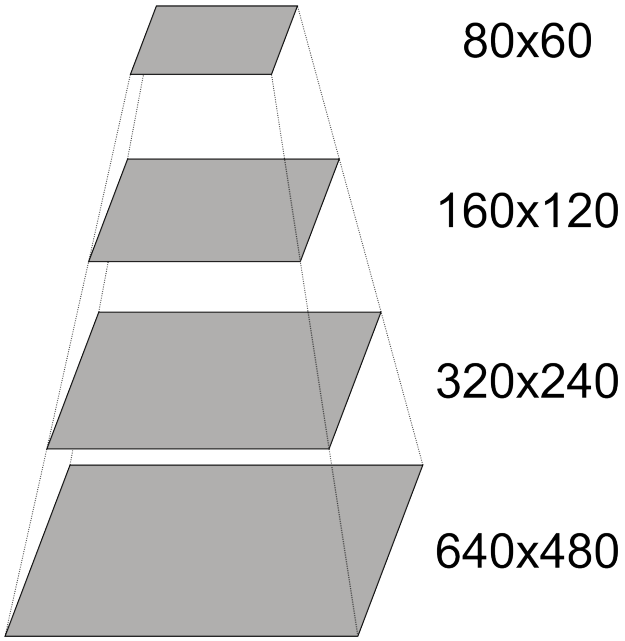
$$\begin{bmatrix} V_x \\ V_x \end{bmatrix} = \begin{bmatrix} \sum I_{xi}^2 & \sum I_{xi}I_{yi} \\ \sum I_{xi}I_{yi} & \sum I_{yi}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum I_{xi}I_{ti} \\ -\sum I_{yi}I_{ti} \end{bmatrix} \tag{4.6}$$

Kde sumy provádíme od 1 do i.

Z výše uvedených vztahů vyplývá, že optický tok lze určit výpočtem derivací dle všech dimenzí.

Pyramidová reprezentace obrazu

Pyramida nebo pyramidová reprezentace je hierarchie po sobě následujících obrazů s nižším rozlišením, která je postavena na originálním obraze. Základnu pyramidy tvoří obraz v nejvyšším, tedy původním rozlišení. Obraz na dalším stupni je vytvořen rozmazáním obrazu pomocí dolnopropustního filtru nebo podvzorkováním obrazu, čímž se zmenší jeho rozlišení na polovinu (Obrázek 4.3). Indexace pyramidy se provádí od základny, tedy od snímku s nejvyšším rozlišením [26].



Obrázek 4.3: Pyramidová reprezentace obrazu

Pyramidová implementace metody Lucas-Kanade

Každé $L \in \langle 0, L_{max} \rangle$ stanoví $u^L = [u_x^L \; u_y^L]$, což jsou souřadnice bodu u v obrázku s pyramidovou hierarchií. Vektor u^L se vypočítá nezávisle na osách následovně:

$$u^L = \frac{u}{2^L} \tag{4.7}$$

Algoritmus nejdříve provede výpočet optického toku pro vrstvu s nejvyšším indexem L_{max} (nejnižší rozlišení). Poté se výsledek přesune o úroveň níže než byl počáteční odhad pro výpočet kultivovaného optického toku (refinde optical flow). Tento postup se opakuje, dokud není dosaženo nejnižší úrovně (L_0), což je obrázek v původní velikosti.

Jeden cyklus výpočtu mezi úrovněmi $L + 1$ a L probíhá následovně. Předpokládejme, že počáteční odhad toku pro úroveň L je k dispozici z předchozího výpočtu z úrovně $L + 1$. Nyní je nutné vypočítat rozdílový pohybový vektor $d^L = [d_x^L \; d_y^L]^T$ při minimální chybové funkci ε^L.

$$\varepsilon^L(d^L) = \varepsilon(d_x^L, d_y^L) = \sum_{x=u_x^L-\omega_x}^{u_x^L+\omega_x} \sum_{y=u_y^L-\omega_y}^{u_y^L+\omega_y} (I^L(x,y) - J^L(x+g_x^L+d_x^L), y+g_y^L+d_y^L))^2 \quad (4.8)$$

Integrační okno tedy zůstává stejně velké pro všechny úrovně L. Počáteční odhad, g^L, je využit k přibližnému vyhledání místa v obraze J. Následkem čehož je velikost optického toku malá, a proto lehce spočitatelná metodou Lucas-Kanade. Výsledek výpočtu se přenese do další úrovně a vytvoří se z něj další počáteční odhad:

$$g^{L-1} = 2(g^L + d^L) \quad (4.9)$$

Výpočet rozdílového toku je opakován přes všechny zbylé úrovně až po dosažení originálního obrázku. Výsledný optický tok je tedy:

$$d = \sum_{L=0}^{L_m} 2^L d^L \quad (4.10)$$

Největší výhodou pyramidové implementace je fakt, že každý rozdílový tok může být velmi malý, i když výsledný rozdílový tok bude velký [7].

4.2.2 Vyhledávání oblastí

Metody pracující na principu vyhledávání oblastí (region-based matching) využívají korelačních funkcí, které jsou schopné určit míru podobnosti dvou různých obrazů. Na optický tok potom lze nahlížet jako na transformační funkci, s níž lze určit obraz v čase t pomocí transformace obrazu v čase $t - dt$ (za předpokladu zachování intenzity). Má-li metoda k dispozici odhad optického toku, je následně schopna měřit jeho přesnost pomocí korelační funkce mezi transformovaným obrazem a známým obrazem. Samotný problém lze tedy v obecné rovině redukovat na prohledávání prostoru možných optických toků, které má za cíl nalezení toku s největší korelační přesností. Díky tomu se metody mohou omezit pouze na prověřování určitých transformací – ve většině případů jen na posun.

Metody pracující na principu prohledání oblastí v základní podobě řeší výpočet optického toku hrubou silou, což vede k jejich univerzálnosti a přesnosti. Na druhou stranu jsou tyto metody výpočetně náročné, neboť je nutné prověřovat všechna možná řešení. Proto se pokročilejší metody soustředí na omezení možných řešení (např. postupným zjemňováním výpočtu).

4.2.3 Metody založené na energii

Metody založené na energii (energy-based methods) používají k určení optického toku frekvenční domény obrazu. Metody jsou tvořeny sadou filtrů. Parametry těchto filtrů jsou směr a rychlost posunu. Z odezvy filtrů jsou následně určeny výsledné rychlosti pro jednotlivé pixely.

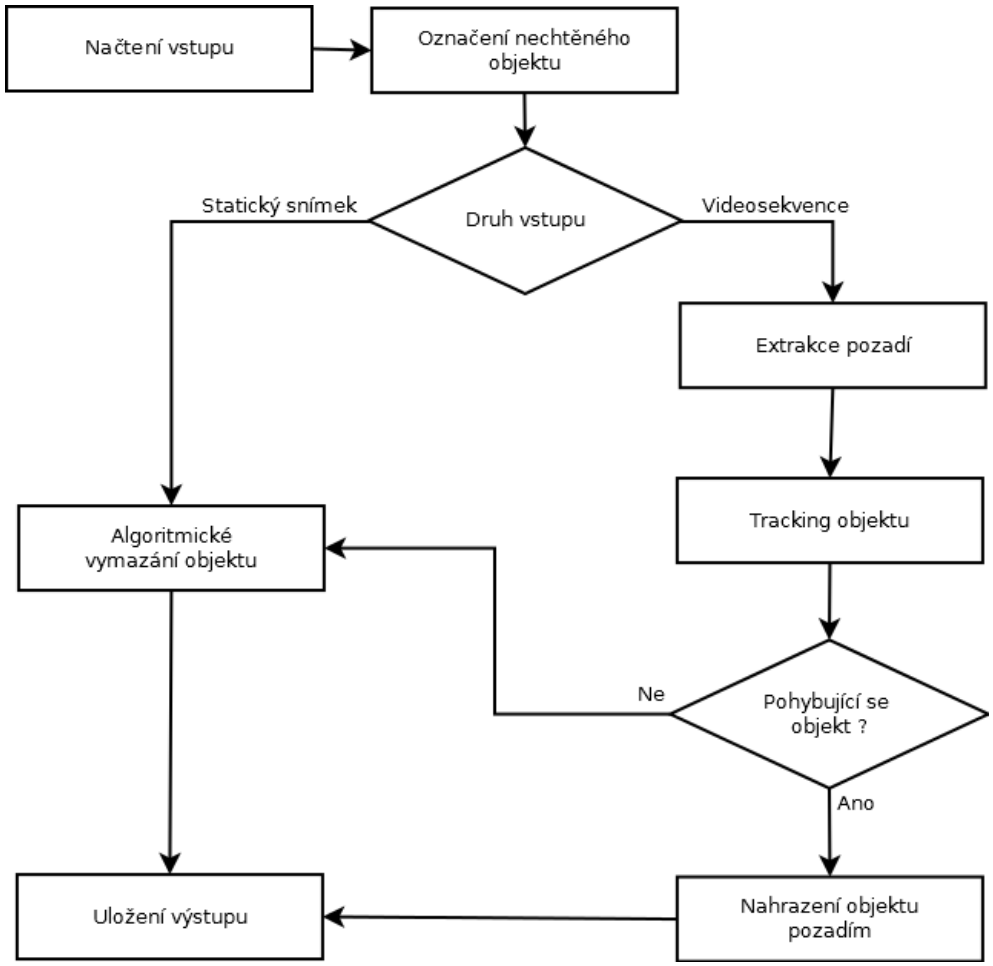
4.2.4 Metody založené na fázi

Metody založené na fázi (phase-based methods) jsou principiálně podobné metodám založených na energii. Tyto využívají vlastností Fourierova obrazu vstupní sekvence a vycházejí z poznatku, že posun v obraze v prostorové doméně se projeví posunem fáze v doméně frekvenční. Základní rovnice pro výpočet jsou analogické s gradientními přístupy. Rozdílem je pak využití gradientu fáze namísto gradientu intenzity v obraze. Gradient fáze je ve většině případů stabilnější, a proto může být výhodnější pro výpočet optického toku.

Kapitola 5

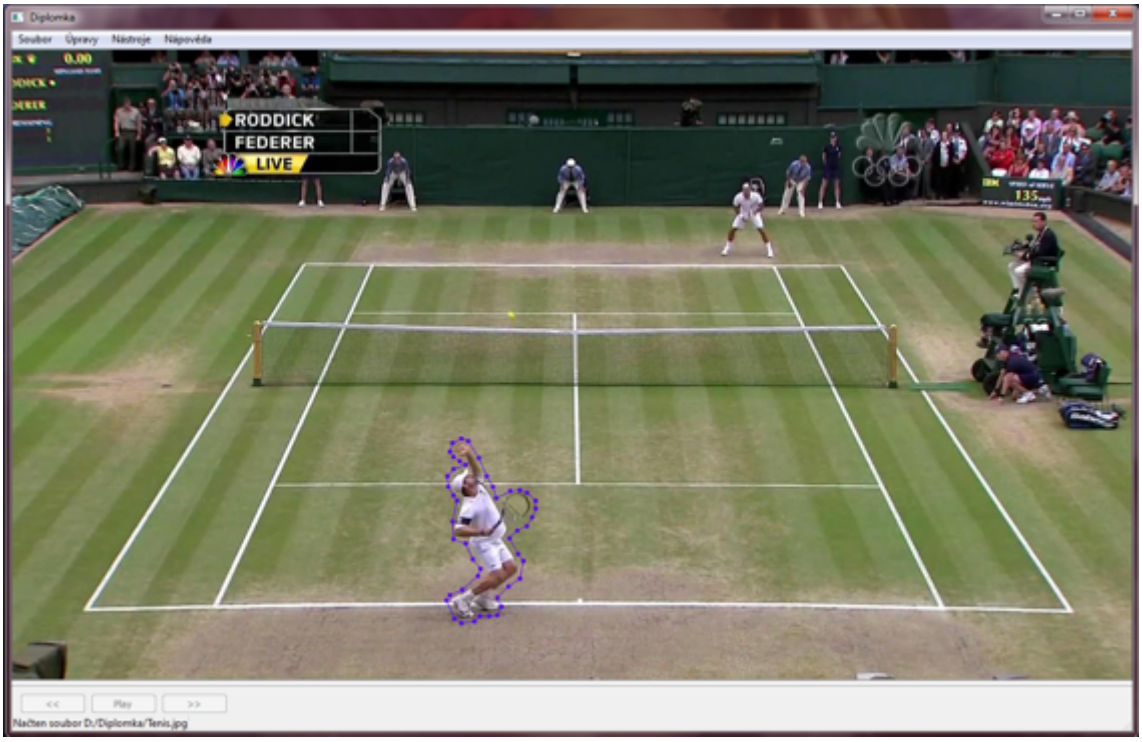
Návrh aplikace

Aplikace bude prioritně určena k odstraňování nežádoucích objektů z videosekvencí, ale bude schopna pracovat i se statickými snímky (fotografiemi). Objekt může být po celou dobu záznamu statický nebo se může po scéně pohybovat. Výsledná aplikace bude při práci s videozáznamem počítat s jeho pořízením pomocí statické kamery bez zoom-ování.



Obrázek 5.1: Blokové schéma aplikace

Aplikace se bude skládat zejména ze čtyř stěžejních částí – modulu pro odstranění objektu ze statického snímku (nebo též pro odstranění statického objektu z videosekvence), modulu pro extrakci pozadí z videosekvence, modulu pro sledování pohybu objektu ve videosekvenci (dále tracker) a modulu pro nahrazení pohybujícího se objektu pozadím.



Obrázek 5.2: Označení nechtěného objektu

Předpokládejme, že na vstupu se nachází videosekvence, ve které již uživatel označil nechtěný objekt (Obrázek 5.2). Prvním důležitým krokem, který vyžaduje kompletní průchod videosekvencí, je extrakce pozadí. V dalším průchodu je poté proveden tracking uživatelem definovaného objektu. Pro každý snímek je určeno, zda–li se objekt pohybuje nebo je naopak statický. V případě statického objektu se předpokládá, že tento objekt se nachází i na extrahovaném pozadí. Je proto provedena změna v extrahovaném pozadí a tento objekt je z něj pomocí algoritmu pro odstranění nežádoucího objektu z fotografie odstraněn. Nahrazení objektu následně proběhne jednoduchým zkopírováním části extrahovaného pozadí, kterou pohybující se objekt zakrýval.

V případě statického objektu, fotografie, je nutné využít algoritmu pro odstranění nežádoucího objektu z fotografie.

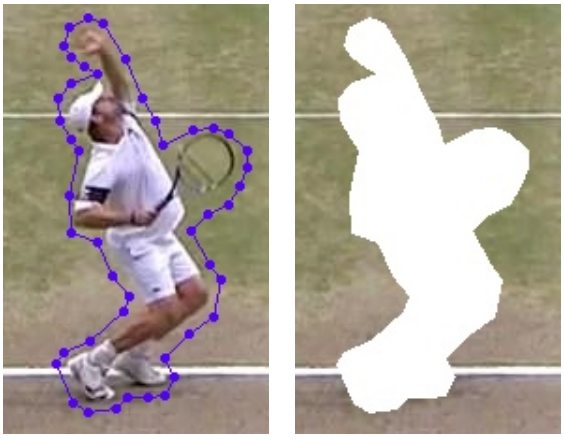
Kapitola 6

Zpracování statického snímku

V následující kapitole je popsán postup jakým aplikace provádí odstranění nežádoucího objektu ze statického snímku - fotografie.

6.1 Vytvoření masky

Uživatel definuje nechtěný objekt (Obrázek 5.2) zadáním bodů určujících jeho okraje. Nyní je nutné z těchto bodů vytvořit vyplněný polygon. Vyplnění je provedeno pomocí metody semínkového vyplňování, která jako vstup vyžaduje pixel nacházející se uvnitř oblasti. Jedná se o problematiku určení, zda–li se bod nachází uvnitř n–úhelníku či nikoliv. Tento bod je určen postupem vycházejícím z [23]. Uživatelem definované body popisující masku budeme v dalším textu nazývat okrajovými body.



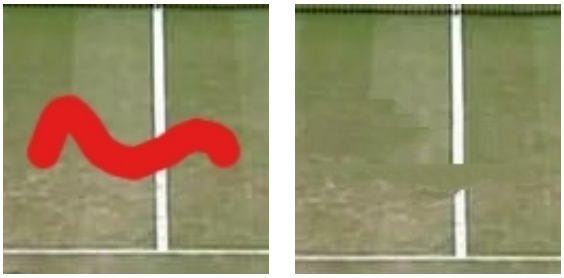
Obrázek 6.1: Vstup a výstup metody pro vytvoření masky

Postupně jsou brány jednotlivé okrajové body a je zkoumáno jejich okolí od vzdálenosti 1 až po vzdálenost 9. Aby mohl být zkoumaný bod I označen jako vnitřní bod, musí být pro každou dvojici okrajových bodů A a B splněna následující dvojice podmínek:

- $(A(y) < I(y) \wedge B(y) \geq I(y)) \vee (A(y) \geq I(y) \wedge B(y) < I(y))$
- $A(x) + \frac{(I(y)-B(y))}{B(y)-A(y)} \cdot (B(x) - A(x)) < I(x)$

6.2 Určení směru textury

Metoda pro vymazání nežádoucího objektu předpokládá v daném snímku převážně horizontální orientaci textury, a proto provádí nahrazování pozadím postupně po řádcích. Pokud se na jejím vstupu ocitne obrázek obsahující vertikálně oritentované textury, může velmi snadno dojít k vyplnění dané oblasti nevhodným pozadím (Obrázek 6.2).



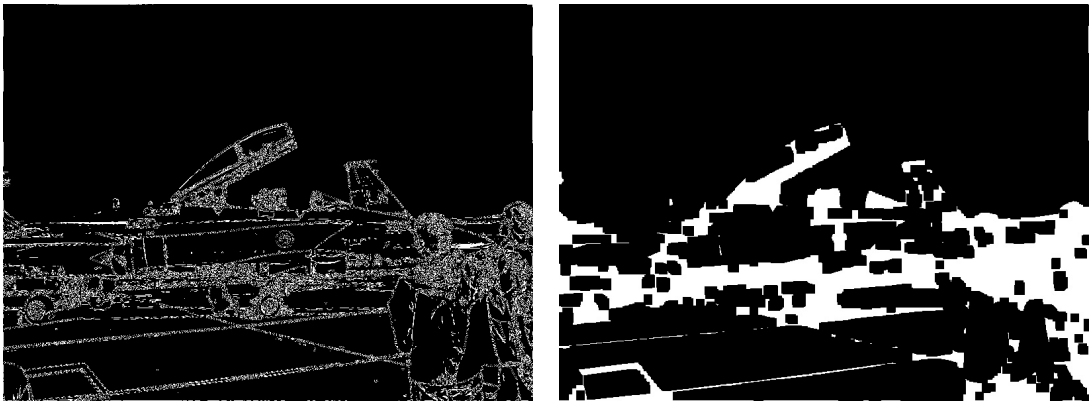
Obrázek 6.2: Vstup a výstup metody bez předzpracování pomocí metody pro určení směru textury

Navržený postup pro určení směru textury byl inspirován [28]. Prvním krokem algoritmu je detekce hran ve snímku. Hrany jsou zde považovány za hlavní prvek určující směr textur. Detekci hran provádím pomocí Kirschova operátoru [29], který se ukázal být mnohem vhodnější namísto původně testovaného Sobelova operátoru. Kirschův operátor nám totiž dodává větší množství potřebných inforamcí než jeho konkurent. Ideální práh je určován pro každý snímek dynamicky pomocí metody Otsu [27] a následně ještě přepočten pro vyšší efektivitu (Vzorec 6.1). Výsledný obraz je po prahování ještě podroben postprocesingu, který odstraní nežádoucí šum a zesílí nalezené hrany. Tento postprocesing je v naší práci prováděn pomocí morfologických operací (Obrázky 6.3 a 6.8).

Obrázek 6.3: Vstupní snímek a výstup po aplikování Kirschova operátoru

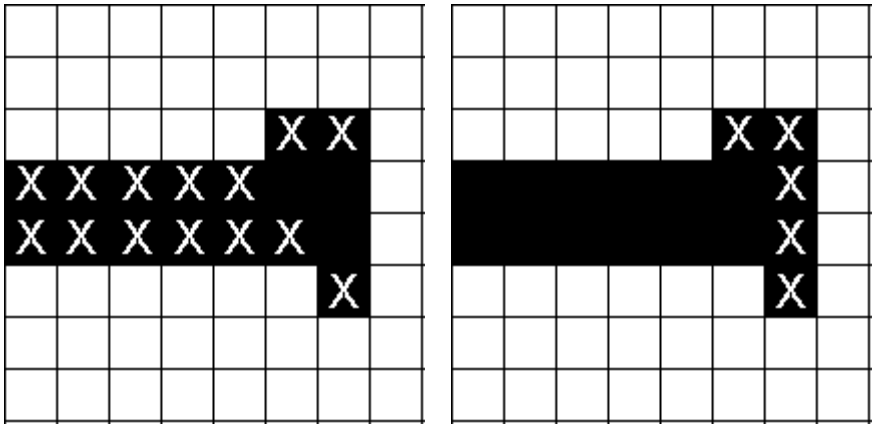
$$práh = Otsu \cdot 1.75 \tag{6.1}$$

Vstupní obraz je rozdělen do oken velikosti 8x8 pixelů. V každém okně je následně provedena analýza, která rozhodne, zda je textura v tomto okně horizontálního či vertikálního



Obrázek 6.4: Výstup po prahování a výsledný snímek

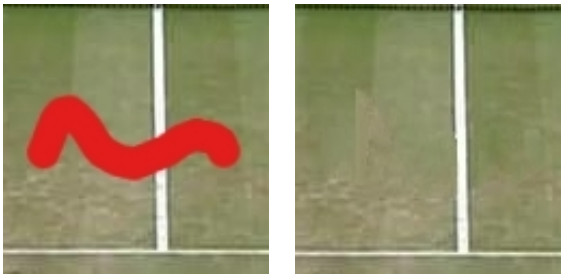
charakteru. Prvním krokem této analýzy je určení počtu pixelů reprezentujících horizontální hrany a počtu pixelů představujících vertikální hrany.



Obrázek 6.5: Pixely označené jako horizontální a pixely označené jako vertikální

Jako horizontální je označen pixel, který má ve vertikálním směru, posun na ose y o $+1$ a -1, alespoň jednoho souseda jiné barvy (v binárním obrazu vzniklém po prahování se vyskytují pouze dvě barvy, černá a bílá). Vertikální pixely jsou určeny analogicky, posun na ose x o $+1$ a -1. Výsledkem tohoto kroku je počet horizontálních a vertikálních pixelů v daném okně (Obrázek 6.5). Okno je ohodnoceno jako horizontální, pokud obsahuje více horizontálních než vertikálních pixelů a naopak. Pokud je počet těchto pixelů stejný, okno se nebude do celkového výsledku počítat. Směr textury ve vstupním obraze je následně určen podle ohodnocení všech oken.

Díky zařazení detekce směru textury před samotné zpracování je možné adekvátně reagovat na zjištěný směr a provést nahrazení nežádoucího pozadím s poznatelně vyšší kvalitou (Obrázek 6.6).



Obrázek 6.6: Vstup a výstup metody s předzpracování pomocí metody pro určení směru textury

6.3 Vyplnění oblasti vzniklé vymazáním objektu

Analýza distribuce barev, váhovaná interpolační metoda a metoda pro detekci artefaktů jsou implementovány podle vztahů a postupů popsaných v kapitole 3 a k jejich korektnímu chodu není potřeba žádná větší preprocesingová či postprocesingová část, a proto se jimi již nebudeme dále zabývat. U sub-patch syntézy textur je situace poněkud komplikovanější a před započetním samotného procesu porovnávání aktuálního a kandidátních vzorků je nutné vyřešit několik preprocesingových kroků.

6.4 Sub-patch syntéza textur

Řádek zpracovávaný touto metodou není vyplněn naráz. Bylo totiž zjištěno, že při dlouhých řádcích není plně zaručena kvalita výsledného obrazu a celý proces hledání a výběru kandidátních vzorků může trvat (a v drtivé většině případů opravdu trvá) neúměrně dlouhou dobu. Mezera je proto rozdělena na sadu menších částí o maximální délce 8 pixelů a její vyplnění je provedeno postupně zleva doprava pomocí kandidátních vzorků hledaných pro tyto menší části (Obrázek 6.8).

Určení okolí pro porovnání aktuálního a kandidátních vzorků

Prvním nutným krokem je určení velikosti okolí, jež se bude používat při porovnávání kandidátních vzorků s vyplňovanými pixely mezery (Obrázek 3.3). V teoretické části byl uveden i vztah (Vzorec 3.2) pro výpočet počtu sousedních pixelů. Bylo též uvedeno, že hloubka okolí použitého k porovnávání je původně nastavena na 3 a může být zvětšována, dokud okolí neobsahuje dostatek informací nutných k porovnávání. Toto zvětšování probíhá tak dlouho, dokud není akumulovaná hodnota normy gradientu Laplacianu vyšší než stanovený práh [13].

$$\frac{grad_{akum}}{100} \leq (max_{Lapla} - min_{Lapla}) \tag{6.2}$$

Postup pro určení ideální hloubky okolí je tedy následující. Nejprve pro celý snímek (vyjma mezery) vypočítáme maximální a minimální hodnotu Laplaciánu pomocí konvolučního jádra (Obrázek 6.7). Poté nastavíme prvotní hloubku okolí na 3 a vypočteme akumulovanou hodnotu gradientu Laplaciánu všech pixelů v tomto okolí. Pokud je splněn vztah

0	-1	0
-1	4	-1
0	-1	0

Obrázek 6.7: Konvoluční jádro použité při výpočtu Laplaciánu

(6.2) je výpočet ukončen, v opačném případě je zvětšena hloubka. Maximální hodnota hloubky je 11. Experimentálně bylo zjištěno, že vyšší hloubka již není vhodná, neboť dochází k významnému zpomalení celého procesu porovnávání bez signifikantního nárustu kvality syntetizované oblasti.

Určení oblasti pro vyhledávání kandidátních vzorků

Autoři této metody původně uvádějí, že je vhodné kandidátní vzorky hledat v oblasti odpovídající jedné polovině rozlišení celého snímku, což u větších snímků představuje značně rozsáhlou oblast a znatelně zpomaluje celý proces vyhledávání. Použitá implementace vyhledává kandidátní vzorky v obdélníku, jehož velikost je dána vzorci 6.3 a 6.5, se středem obdélníku (vertikálním i horizontálním) ve středu vyplňované mezery. Pokud je vypočtená šířka větší než jedna třetina celkové šířky snímku, využije se pro výpočet rozměrů obdélníku vztahů 6.4 a 6.5.

Obrázek 6.8: Ukázka rozdělení mezery a umístění oblasti pro prohledávání

$$\textit{šířka\_obdélníku} = \frac{mezera}{8} \cdot K$$

$$kde: \ K = \begin{cases} 40, & \frac{mezera}{8} \leq 5 \\ 32, & \frac{mezera}{8} > 5 \end{cases} \tag{6.3}$$

$$\textit{šířka\_obdélníku} = \begin{cases} mezera + 128, & \frac{\textit{šířka\_obrazu}}{3} - mezera < 128 \\ \frac{\textit{šířka\_obrazu}}{3}, & \frac{\textit{šířka\_obrazu}}{3} - mezera \geq 128 \end{cases} \tag{6.4}$$

$$\textit{výška\_obdélníku} = \frac{\textit{výška\_obrazu}}{\textit{šířka\_obrazu}} \cdot \textit{šířka\_obdélníku} \tag{6.5}$$

kde : mezera je celkový počet pixelů aktuálně vyplňovaného řádku masky

Při procesu vyhledávání ideálního kandidátního vzorku jsou vynechávány pixely náležející masce definující vyplňovanou oblast. Bylo by sice možné používat k vyplnění i již dokončení předchozí řádky, ale výsledný obraz je poté daleko méně kvalitní než v opačném případě.

Barevný model YCbCr

Tento barevný model je v aplikaci nutný kvůli porovnávání kandidátních vzorků se vzorkem aktuálním (Vzorec 3.5). YCbCr je barevný prostor, kde již nejsou zastoupeny barvy jako u RGB, ale Y představuje světelnost a Cb a Cr představují chromizační signály [24].¨Přepočet barev se provádí podle následujících doporučených vztahů.

$$\begin{aligned} Y &= 0,299 \cdot R + 0,587 \cdot G + 0,114 \cdot B \\ Cb &= -0,1687 \cdot R - 0,3313 \cdot G + 0,5 \cdot B + 128 \\ Cr &= 0,5 \cdot R - 0,4187 \cdot G - 0,0813 \cdot B + 128 \end{aligned} \tag{6.6}$$

Kapitola 7

Zpracování videosekvence

Tato kapitola je věnována odstranění nežádoucího objektu z videosekvence. V následujícím textu budou popsány metody a postupy nutné pro sledování pohybu objektu, posunu uživatelem definované masky a samotnému procesu vymazání, potažmo nahrazení nežádoucího objektu extrahovaným pozadím.

7.1 Extrakce pozadí

Extrakce statického pozadí zaujímá ve zpracování videosekvence klíčovou roli, protože toto pozadí je využíváno téměř ve všech dalších krocích. Samotná extrakce se provádí pomocí metody založené na výpočtu rozdílu mezi jednotlivými po sobě následujícími snímky [17].

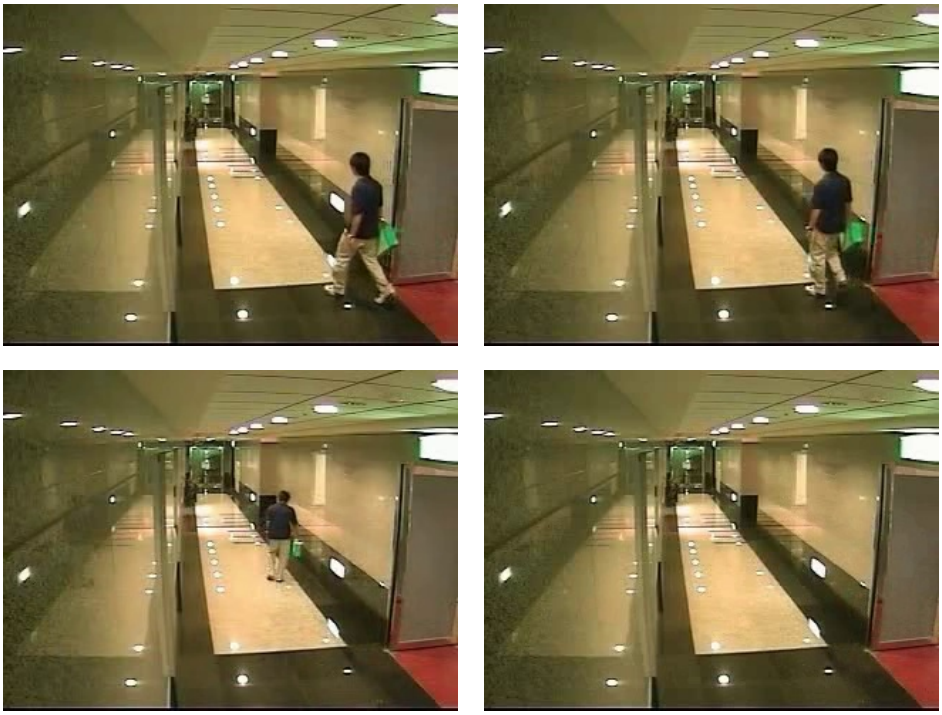
Původní metoda používala k získání pozadí všechny snímky v dané videosekveci. Tento postup je však velmi časově náročný a též málo efektivní. Naše metoda pro extrakci pozadí využívá pouze omezený počet snímků, experimentálně byl zjištěn vztah mezi počtem snímků za sekundu (FPS) a maximálním počtem snímků, které lze při extrakci přeskočit, aniž by byla výrazně ovlivněna kvalita výsledného pozadí (Vzorec 7.1).

$$krok = \frac{FPS}{5} \tag{7.1}$$

Prvním krokem při porovnání dvou snímků je jejich převod do šedotónového obrazu. Následně jsou oba snímky od sebe odečteny a výsledek je prahován (experimentálně byl určen ideální práh 5). Pixely, jejichž rozdílová hodnota je nižší než zvolený práh, jsou označeny jako potenciální pozadí a jsou uloženy do předem připravené struktury pro další zpracování. Pokud se ve struktuře na dané pozici již vyskytuje pixel se stejnou hodnotou, je pouze navýšena hodnota čítače u této hodnoty. Jako pozadí jsou poté pro každý pixel určeny hodnoty s nejvyšším stupněm výskytu (Vzorec 7.2).

$$\begin{aligned} &Bcg(x,y) = mode(S_1(x,y), S_2(x,y), ..., S_N(x,y)) \\ &kde: \ Bcg(x,y) \textit{ je pixel pozadí na souřadnicích } (x,y) \\ &\quad S_N \textit{ jsou jednotlivá potenciální pozadí} \end{aligned} \tag{7.2}$$

V rámci úspory paměti a celkového zrychlení celého procesu extrakce jsou podobné pixely, absolutní rozdíl hodnot těchto pixelů může být maximálně 5 (zjištěno experimentálně), považovány za totožné. Pomocí této metody lze získat velmi přesné pozadí (Obrázek 7.1) za přijatelnou dobu trvání celé extrakce.

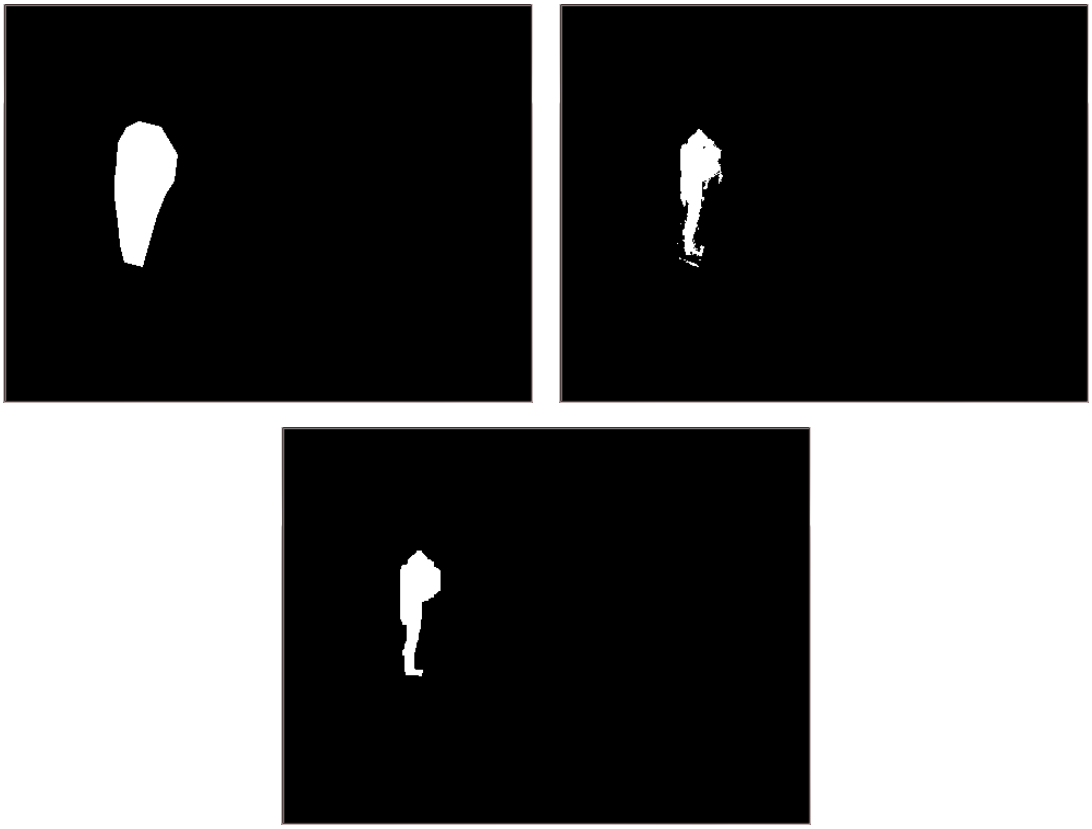


Obrázek 7.1: Příklad několika snímků videosekvence a získaného pozadí

7.2 Výběr význačných bodů

Ke sledování pohybu objektu ve videosekvenci je použita pyramidová implementace metody Lucas-Kanade, jejíž princip je popsán v kapitole 4. Pro účely aplikace je využita její implementace obsažená v knihovně OpenCV [21], z níž je použita metoda pro nalezení ideálních bodů pro trackování (implementaci využívá jako význačné body rohy nalezené pomocí Harrisovy metody) a metoda pro výpočet nových souřadnic těchto bodů v dalším snímku. Pokud ovšem metodu pro získání význačných bodů aplikujeme přímo na celou uživatelem definovanou masku, dochází často k výběru velmi nevhodných bodů, které vůbec nepatří sledovanému objektu, a ve většině případů se jedná o statické pozadí za objektem. Sledováním těchto bodů, které se vlastně ani nepohybují, by docházelo k nežádoucímu znehodnocení celého výsledku. Je proto nutné tuto masku více zpřesnit, k čemuž lze využít extrahované pozadí.

Extrahované pozadí i aktuální snímek jsou převedeny do stupňů šedi. Snímky od sebe nyní odečteme, přičemž nás zajímá oblast ohraničená uživatelem definovanou maskou (Obrázek 7.2), zbytek snímku ignorujeme. Následuje proces prahování výsledku, který nám pomůže určit přesnější rozsah masky. Výběr ideálního prahu se provádí pro každý snímek zvlášť pomocí metody Otsu [27]. Výstup z této metody je ještě upraven (Vzorec 7.3), neboť bylo zjištěno, že při této hodnotě je dosahováno nejlepších výsledků. Výstup po provedení prahování je pro zlepšení kvality ještě podroben postprocesingu pomocí morfologických operací – konkrétně binárnímu otevření následovaného vícerým binárním uzavřením (Obrázek 7.2).



Obrázek 7.2: Zpřesnění masky určené pro výběr význačných bodů

$$práh = \frac{Otsu}{4} \tag{7.3}$$

7.3 Filtrace význačných bodů

Bylo zjištěno, že metoda pro výpočet posunu význačných bodů mezi dvěma snímky může generovat i posuny, které jsou několikanásobně větší než ostatní. Tyto chybné posuny mají značně negativní vliv na výsledný výpočet posunu okrajových bodů, a proto je nutná jejich filtrace. Dle vzorce 7.4 určíme maximální možný posun význačného bodu mezi dvěma po sobě jdoucími snímky.

$$posun_{max} = \frac{300}{FPS} \tag{7.4}$$

Pokud je absolutní hodnota posunu nějakého z význačných bodů (v x-ové nebo y-ové ose) větší než maximální možný posun, provede se dodatečná analýza tohoto bodu.

Prvním krokem této analýzy je výpočet rozsahu – maximální a minimální hodnoty v obou osách – původních hodnot (tedy hodnot z předchozího snímku) význačných bodů. Následně je vypočítána maximální vzdálenost (Vzorec 7.5) od testovaného bodu, ve které budeme hledat další význačné body.

$$vzdálenost_{max} = \left| \frac{max - min}{5} \right| \tag{7.5}$$

Ze všech význačných bodů, které leží v maximální vzdálenosti vypočtené v předchozím kroku, je určen průměrný posun pro každou osu (Vzorec 7.6).

$$posun = \frac{\sum_{i=1}^{D}(n - s)}{|D|} \tag{7.6}$$

$$D = \{(vzdálenost_i, n_i, s_i), i = 1, 2, 3, ... | vzdálenost_i < vzdálenost_{max}\}$$

Pokud v se v okruhu daném maximální vzdáleností $vzdálenost_{max}$ neleží ani jeden význačný bod, kromě bodu testovaného. Tedy pokud $|D| = 0$, je bod označen za neplatný. V opačném případě je označen jako neplatný, pokud je splněn následující vztah (Vzorec 7.7):

$$|n - s| \leq 2 \cdot posun \tag{7.7}$$
$$kde: \ n \text{ je pozice zkoumaného bodu v předchozím snímku}$$
$$s \text{ je pozice zkoumaného bodu v aktuálním snímku}$$

Jako další stupeň pro zlepšení kvality masky a dodatečnou úpravu pozic okrajových bodů bylo uvažováno použití algoritmu RANSAC. Jelikož se ovšem výše popsaná metoda pro filtraci význačných bodů ukázalo jako postačující, nebylo k tohoto stupně nakonec využito.

7.4 Výpočet posunu okrajových bodů

Při sledování objetku ve videosekvenci je nutné řešit problém posunu okrajových bodů definujících masku. V předchozím textu byl popsán postup pro získání význačných bodů pro trackování a pro výpočet jejich pozic v následujícím snímku. Nyní je nutné z posunů těchto bodů extrapolovat nové pozice okrajových bodů masky. Prvním krokem výpočtu je seřazení význačných bodů podle vzdálenosti od aktuálně zpracovávaného okrajového bodu. Následně je pomocí vzorce 7.8 vypočítána nová pozice okrajového bodu. Ve výpočtu se počítá nejen s posunem nejbližšího řídícího bodu, ale i s dalšími body ležícími v předem definované vzdálenosti od okrajového bodu. Vzdálenost význačného bodu dává tomuto bodu též váhu, čímž je zaručeno, že nejbližší význačný bod bude mít největší vliv na velikost a směr posunu okrajového bodu. Také je ovšem zaručena určitá korekce nesprávného posunu nejbližších význačných bodů, jejichž nevhodný posun částečně kompenzují právě posuny ostatních bodů.

$$v = v + \frac{\sum_{i=1}^{D}((max - vzdálenost)^2 \cdot (n - s)}{\sum_{i=1}^{D}(max - vzdálenost)^2} \tag{7.8}$$

$$D = \{(vzdálenost_i, n_i, s_i), i = 1, 2, 3, ... | vzdálenost_i < max\}$$

Hodnota max využitá v předchozím vztahu je počítána pro každý snímek, respektive pro každý nový tvar a pozici masky. Specifikuje nám rádius se středem v každém z okrajových bodů, ve kterém je možné hledat význačné body pro výpočet posunu tohoto bodu. Pro výpočet této hodnoty (vzorec 7.9) se využívá rozměrů obalového obdélníku (Obrázek 7.3).

$$max = \sqrt{\left(\frac{šířka_{obdélníku}}{10}\right)^2 + \left(\frac{výška_{obdélníku}}{10}\right)^2} \tag{7.9}$$

$$kde: \ šířka_{obdélníku} a \ výška_{obdélníku} \ udávají \ rozměry \ obalového \ obdélníku$$

Obrázek 7.3: Obalový obdélník

V případě, že v maximální rádiu okolo okrajového bodu neleží ani jeden význačný bod, je při splnění podmínky $vzdálenost \leq 20 \cdot max$ použit nejbližší možný řídící bod. V opačném případě musí být posun tohoto bodu určen pomocí vzájemné polohy tohoto bodu a jeho souseda z předchozího snímku (vzorec 7.10) a nové pozice tohoto souseda ve snímku aktuálním (vzorec 7.11).

$$posun = B_i - B_{i-1} \tag{7.10}$$

$$pozice_n = B_{i-1} + posun \tag{7.11}$$

$$kde: \ B_i \ je \ pozice \ okrajového \ bodu \ a \ B_{i-1} \ je \ pozice \ jeho \ souseda$$

7.5 Rozlišení statického objektu

Aby bylo možné provést korektní nahrazení, je nutné rozlišit statický a pohybující se objekt. Za statický v tomto případě považujeme takový objekt, který se z větší části nachází v extrahovaném pozadí, a tedy není možné jeho nahrazení tímto pozadím. Extrahované pozadí musí být v případě nálezu statického objektu upraveno pomocí metody pro odstranění objektu z fotografie.

Rozlišení statického a pohybujícího se objektu se provádí společně s procesem provádějícím zpřesnění uživatelem definované masky. Objekt je za statický označen v případě, že splňuje podmínky definované ve vztahu 7.12 a zároveň je díky informacím o jeho posunu mezi předchozím a aktuálním snímkem určen jako nepohybující se. Určení pohyblivosti se provádí pomocí informací z okrajových bodů. Pokud je absolutní hodnota posunu alespoň jednoho z těchto bodů v ose x nebo v ose y větší než 0.5, je celý objekt označen jako pohyblivý.

$$(maska - maska_Z) > 3.8 \cdot maska_Z \tag{7.12}$$

kde : *maska je počet pixelů náležejících uživatelem definované masce*
$maska_Z$ *je počet pixelů náležejících zpřesněné masce*

7.6 Nahrazení nechtěného objektu pozadím

Smazání uživatelem definovaného nežádoucího objektu se provádí ve stejném průchodu videosekvecí jako jeho tracking. Jakmile je vypočítána nová poloha okrajových bodů, je znovu vytvořena celá maska, okolo které je určen obalový obdélník. Tento obdélník nám vymezuje oblast, která bude nahrazena extrahovaným pozadím. Vstup a výstup lze vidět na obrázku 7.4.

Obrázek 7.4: Nahrazení objektu pozadím u jednoho snímku videosekvence

Kapitola 8

Použité nástroje

Celá aplikace byla vyvíjena v jazyce C++ pomocí knihovny OpenCV a frameworku QT Creator [25]. Jazyk C++ byl zvolen pro jeho objektovou orientovanost. Pro návrh GUI byl framework QT Creator použit z několika důvodů. Tím hlavním však byla jeho multiplatformnost a fakt, že je určen pro použití s programovacím jazykem C++ a poskytuje kvalitní ladící nástroje.

8.1 OpenCV

Při tvorbě výsledné aplikace byla použita knihovna OpenCV [21], 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. Toto není konečný výčet všech funkcí a možností této knihovny, jedná se pouze o hrubý nástin možností, které nám knihovna OpenCV poskytuje.

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 nebo převody mezi barevnými modely
- načítání videa, jeho rozdělení na jednotlivé snímky a jeho následné uložení
- morfologické operace
- pyramidová implementace metody Lucas-Kanade

Kapitola 9

Testování

Průběh testování implementované metody byl rozdělen do dvou částí – testování metody pro odstranění objektu z fotografie a testování metody zpracovávající videosekvence. Z výsledků těchto procesů vzešla kromě odhalených chyb i řada zjištění o limitacích a úskalích navržené a implementované metody.

Vývoj aplikace a první fáze testování byla prováděna na 64 bitové verzi operačního systému Windows 7. Druhá fáze testů, která byla zaměřená na výkonnost implementované aplikace byla provedena na slabším počítači s 32 bitovou verzí téhož operačního systému. Výsledkům těchto testů je věnována poslední část této kapitoly.

Vliv uživatelem definované masky

Způsob, jakým uživatel označí nežádoucí objekt, hraje významnou roli v celém procesu jeho vymazávání. Je nutno si uvědomit, že při odstraňování statického objektu z videosekvence (nebo při zpracování fotografie), je ideální, aby maska co nejvíce přiléhala k okrajům nežádoucího objektu, neboť bude použit algoritmus pro odstranění nežádoucího objektu z fotografie. Tento při své činnosti využívá okolí mezery vzniklé vymazáním nechtěného objektu, a proto je velmi vhodné mu pro jeho práci dát co nejvíce informací o pozadí a nejbližším okolí mazaného objektu.

U odstraňování pohybujícího se objektu je situace poněkud složitější a ošemetnější. Je nutné vzít v potaz charakter odstraňovaného objektu a podle toho přizpůsobit počet okrajových bodů tvořících masku a též jejich umístění. Při označování lidských postav je ideální vkládat větší počet okrajových bodů a to zejména v blízkosti končetin, jejichž prudší pohyb může působit trackeru problémy. V tomto případě je vhodné též nevkládat okrajové body na okraje objektu, ale je lépe ponechat menší mezeru.

9.1 Statický objekt - fotografie

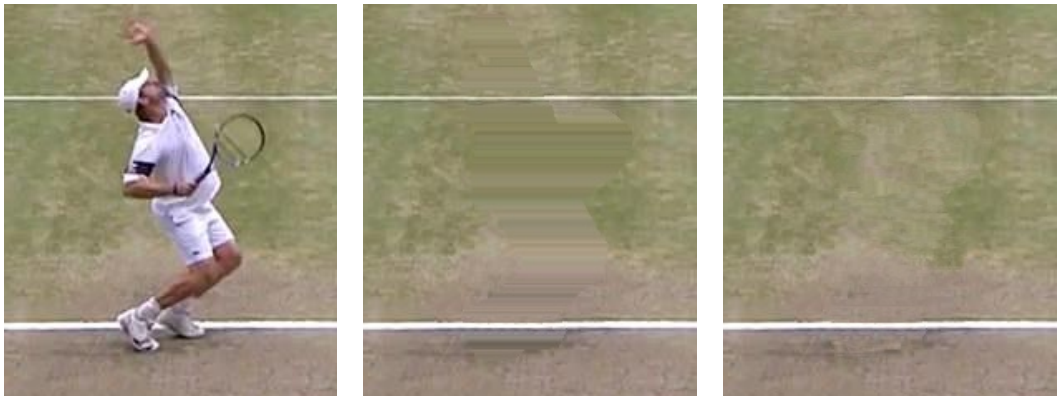
Obsahem této sekce jsou převážně zjištěná omezení implementované metody a též její srovnání s metodou implementovanou v nástroji Adobe Photoshop CS6.

Vliv prahu

Prvním vážným zjištěným při testování metody pro odstranění statického objektu byl nezanedbatelný vliv prahu určujícího výběr metody použité pro vyplnění aktuálního řádku mezery. Tento práh byl původně nastaven na hodnotu uváděnou v příslušné literatuře, ale

ukázal se jako nevhodný. Největší problém nastával u velmi jemně členitých textur jako je například omítka (Obrázek 9.1), betonový povrch nebo trávník (Obrázek 9.2). Bylo rozhodnuto nepoužít pevný práv, ale umožnit jeho změnu podle vůle uživatele. Je možné nastavit šest hodnot tohoto prahu – 85, 70, 55, 40, 25 a 10. Při nastavení nejmenšího prahu aplikaci donutíme použít převážně sub–patch syntézu textur a dosáhneme tím nejkvalitnějšího možného výsledku, ovšem za cenu markantního zpomalení celého procesu vyplňovaní mezery.

Obrázek 9.1: Příklad nahrazení textury omítky při prahu 55 a prahu 10



Obrázek 9.2: Příklad nahrazení textury trávníku při prahu 55 a prahu 10

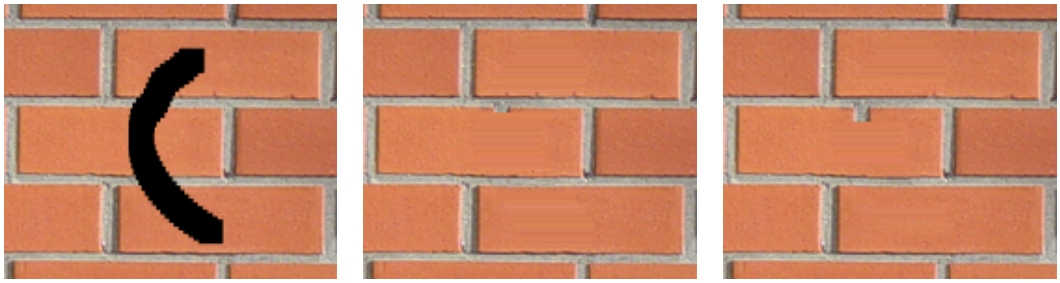
Nastavení ideálního prahu je také významným činitelem ve finálním vzhledu syntetizované textury. V určitých situacích je výhodnější použití vyššího prahu a tím zařazení váhované interpolace pro větší část zpracování mezery (Obrázek 9.3).

Problém T-spoje

T-spojem je v této části myšleno místo spojení vertikální a horizontální linie. Klasickým příkladem bohatým na T-spoje je cihlová zeď.

Pokud máme v místě T-spoje dostatečně malý objekt, je výsledná syntéza uspokojivá a celý spoj nahradí velice přesně (Obrázek 9.4).

Dalším typickým problémem u T-spojů (a nejen u nich) je rozhodování o směru textury. Zde se setkáváme s výrazným omezením námi použité metody pro detekci směru v textuře, protože směr je určován vždy před samotným vyplněním pro celou oblast a nikoliv pro jeden



Obrázek 9.3: Vliv nastavení prahu - práh 55 a 10

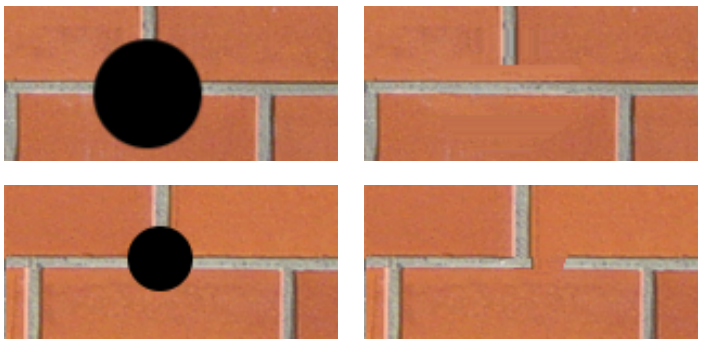
Obrázek 9.4: T–spoj – malý objekt

řádek (tato metoda by byla velmi neefektivní a špatně realizovatelná). Ani zařazení metody pro detekci artefaktů neřeší tento problém. Pokud je jako směr převládající směr v textuře určen směr horizontální, dochází mnohdy k částečné až kompletní ztrátě vertikální linie. Pokud je pro syntézu zvolen směr vertikální, jedná se o totožný problém (Obrázek 9.5).

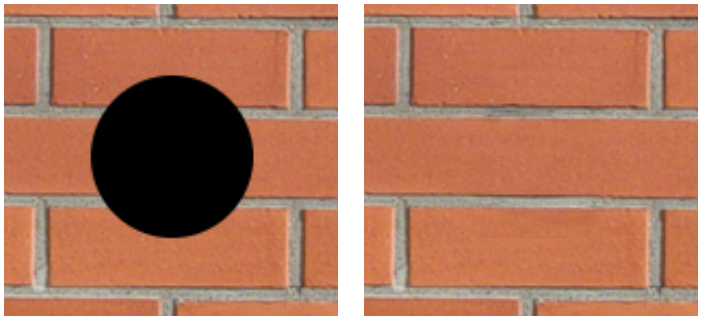
Jestliže objekt zakrývá spoj celý a metoda nemá možnost z žádné strany zjistit možnost horizontální (či vertikální) linie, je celá tato linie v syntetizovaném pozadí úplně ignorována. Lidské oko okamžitě odhalí tuto chybu (pokud ovšem v našem případě nepředpokládáme možnost, že se ve zdi může nacházet větší cihla), ovšem z hlediska algoritmu se jedná o chování odpovídající jeho implementaci (Obrázek 9.6).

Blízkost jiného objektu

Dalším problémem, se kterým se metoda obtížněji vyrovnává, je blízká sousednost vymazávané oblasti s nějakým jiným objektem. Z principu fungování samotné metody pro odstranění nežádoucího objektu z fotografie vyplývá, že pixely ležící nejblíže mezery vzniklé po



Obrázek 9.5: T–spoj – horizontální a vertikální vyplnění



Obrázek 9.6: T-spoj - kompletní překryv

vymazaném objektu mají největší váhu při porovnávání aktuálního a kandidátních vzorků. Díky tomuto charakteristickému chování může dojít ke zkopírování části blízkého objektu, velikost této nechtěné části je ovlivněna několika faktory. Nejvýznamnějším je vzdálenost objektu od hranice mezery, dále již vyplněná část mezery, šířka dolpňovaného řádku a samozřejmě také uživatelem nastavený práh (Obrázek 9.7).

Obrázek 9.7: Problém blízkého objektu

Vznik horizontálních (vertikálních) čar

Tento velmi nepříjemný problém vzniká pouze při nahrazování větších objektů, jejichž okolí tvoří téměř homogenní textura. Termín téměř homogenní textura je myšlena například textura betonu, u které skoro neexistují výrazné hrany a barevný rozsah je minimální. V těchto případech může sub–patch syntéza textur na výstupu produkovat stejný výsledek jako váhovaná interpolace. Dochází defakto k situaci, kdy je pro každý osmi–pixelový aktuální vzorek vybrán totožný kandidátní vzorek (Obrázek 9.8).

Tomuto chování by se dalo zabránit, pokud by se okolí pro prohledávání nevolilo pro celý řádek, ale vždy pouze pro aktuální vzorek. Toto řešení bylo testováno, ale ukázalo

se ne příliš vhodným, neboť jím syntetizované pozadí nedosahovalo takových kvalit jako v případě, kdy je okolí voleno pro kompletní aktuálně vyplňovaný řádek.

Obrázek 9.8: Nežádoucí horizontální čára

Porovnání s Adobe Photoshop CS6

Již v úvodu byl zmíněn nástroj Adobe Photoshop CS6 [6], který umožňuje automatické odstranění nechtěného objektu. Jelikož existuje řada návodů, které obsahují jak vstupní, tak výstupní snímky, rozhodli jsme se provést porovnání námi implementované metody s metodou poskytovanou programem Photoshop. Na obrázku 9.9 je k vidění vstupní snímek a výstup generovaný naší aplikací pomocí metody pro odstranění statického objektu. Photoshop generuje vstup v několika krocích, kdy ve druhém kroku musí uživatel znovu vymezit oblast, kterou chce nahrazovat (Obrázek 9.10).

Obrázek 9.9: Vstupní snímek [6] a výstup z naší aplikace

Obrázek 9.10: Mezikrok a výstup generovaný programem Photoshop CS6 [6]

9.2 Videosekvence

Aplikace zpracová videosekvence s uspokojivými výsledky. Je schopná provést odstranění nežádoucího pohybujícího se, či statického objektu. Také je schopná objekt odstranit až od určitého snímku a nelimituje uživatele nutností pracovat od samotného začátku videosekvence.Následující část se tedy bude zabývat převážně chybami a nedokonalostmi implementované metody při zpracování videosekvence.

Problém rozdělujícího se objektu

Tato charakteristika námi navržené metody vyplývá z celkového postupu zpracování videosekvence. Mějme příklad, kdy vymazávaná osoba má na zádech batoh, který ovšem odloží, a tento batoh leží na stejném místě více než polovinu doby trvání celé videosekvence. Batoh je následně logicky vyhodnocen jako část statického pozadí – což vychází z principu samotné metody použité pro extrakci tohoto pozadí. Následně je během procesu nahrazování toto pozadí využito a objekt se může celý objevit hned na začátku zpracované videosekvence nebo se postupně objevuje dle pohybu sledovaného objektu. Tento problém je zobrazen na (Obrázek 9.11).

Tracking lidských postav

Navržená metoda trackingu pohybujícího se objektu má několik omezení. Nejvýraznější problémy vznikají při sledování lidských postav. Pokud je videosekvence o nízkém rozlišení, nedostáváme dostatečný počet význačných bodů a může selhávat sledování pohybu menších částí těla jako jsou ruce a nohy. Tento problém se též může vyskytovat u videosekvencí s vyšším rozlišením, kdy se na objektu nenachází dostatečný počet význačných bodů. Typicky unikají špičky bot nebo prsty na rukou (Obrázek 9.12).

Zajímavým problémem se ukázala být též tenisová raketa, která na testovaných videosekvencích velmi často na malý okamžik unikala trackeru a objevila se ve výsledku (Obrázek 9.13). Podobný problém nastává i u stínů postav, které se někdy daří vymazávat pouze z části.

U pevných objektů (automobily, letadla, lodě apodobně) se tato problematika nevyskytuje.

Obrázek 9.11: Rozdělený objekt

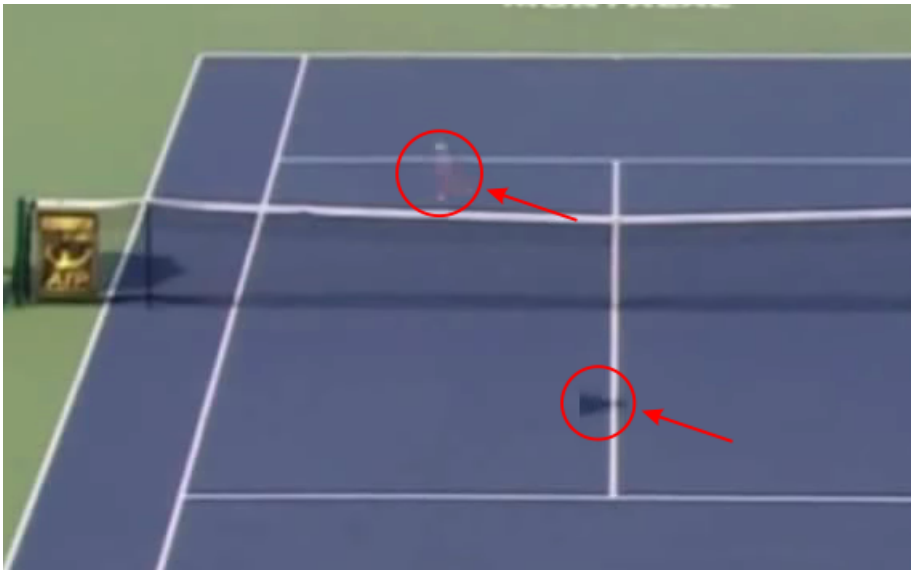
Obrázek 9.12: Nevymazaná část boty

Kolize dvou objektů

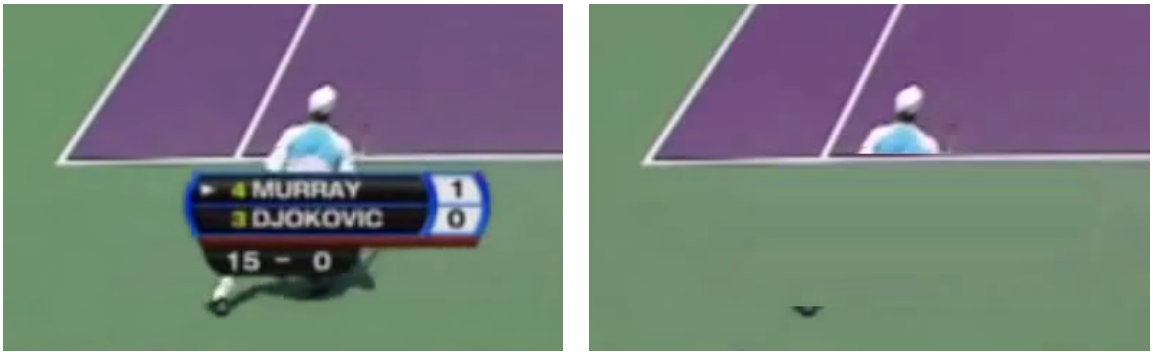
Pokud se nežádoucí objekt setká s jiným objektem, ať už díky pohybu svému, pohybu po scéně nebo pohybu druhého objektu, dochází ve výsledném videu i ke smazání tohoto objektu. Tento problém by bylo možné částečně eliminovat sledováním všech pohybujících se objektů ve scéně, ovšem v případě kolize se statickým objektem (Obrázek 9.14) je tato situace ve stávající implementaci velmi nesnadno řešitelná.

Chybné určení statického/pohybujícího se objektu

Princip rozlišení statického a pohybujícího se objektu byl popsán v kapitole 7.5. Tato metoda se ukázala být velmi efektivní, neboť dokázala rozpoznat statický objekt ve všech testovaných případech. Na druhou stranu však má tato metoda i několik omezení. Prvním problémem je barevná blízkost nechtěného objektu a pozadí. V šedotónovém obrazu je ná-



Obrázek 9.13: Nevymazaná tenisová raketa a část stínu



Obrázek 9.14: Problém překrytí

sledný rozdíl barev pozadí a objektu tak malý, že je objekt označen jako pozadí a metoda aplikuje algoritmus pro odstranění statického objektu. Druhý problém nastává v okamžiku, kdy dojde k neúměrnému zvětšení masky obalující nežádoucí objekt – tato situace nastává v případech, kdy uživatel označí objekt i se stínem, přičemž plocha masky obalující tento stín mnohdy tvoří téměř polovinu plochy kompletní masky. Chybné označení pohybujícího se objektu jako statického může mít vliv na kvalitu celkového výsledku. Toto ovšem platí pouze v případě, že uživatel s touto alternativou nepočítal a má nastavený příliš vysoký práh definující, která z metod na syntézu pozadí má být využita. V opačném případě (tedy pokud je práh nastaven na minimální hodnotu) dochází k prodloužení doby zpracování celé videosekvence.

9.3 Časová náročnost

Doba potřebná ke zpracování jednoho snímku (fotografie) metodou pro odstranění statického objektu je závislá na několika faktorech. Prvním a nejdůležitějším je samozřejmě velikost mezery vzniklé odstraněním nechtěného objektu. Dalším je poté poměr použití sub–patch syntézy textur a váhované interpolace.

U videosekvence je situace poněkud komplikovanější. Doba nutná k získání výsledného videa je ovlivněna prvotně délku samotné videosekvence. Jak bylo již řečeno v kapitolách věnovaným návrhu aplikace (Kapitola 5) a zpracování videosekvence (Kapitola 7) videosekvence je procházena dvakrát. Poprvé je průchod použit pro extrakci statického pozadí a tento výpočet zabírá cca 35 % doby potřebné pro zpracování celé videosekvence (pokud není během videosekvence využita metoda pro odstranění statického objektu).

V následující tabulce (Tabulka 9.1) je uvedena výkonnost implementované metody při několika různých druzích vstupu. Práh metody pro odstranění statického objektu byl nastaven na minimální možnou hodnotu, tedy uvedené časy indikují maximální dobu nutnou pro zpracování jednoho snímku. Nutno ještě uvést, že u obou počítačů použitých pro testování byla použitá knihovna OpenCV 2.3.1 ručně zkompilována ve stadardním nastavení s přidanými následujícími parametry – Threading Building Blocks (Intel TBB), Integrated Performance Primitives (Intel IPP) a není používána podpora Cuda. Počítač 1 byl v průběhu testování osazen procesorem Intel Core i7 s taktovací frekvencí 2,67GHz a 64 bitovým operačním systémem Windows 7. Počítač 2 používal 32 bitovou verzi téhož systému a byl osazen procesorem Intel Core 2 Duo s taktovací frekvencí 1,83GHz.

Vstup a použitá metoda	**Doba zpracování [s]**	
	Počítač 1	**Počítač 2**
Obrázek o rozlišení 800x600 s mezerou o velikosti 3609 pixelů	46	256
Obrázek o rozlišení 450x600 s mezerou o velikosti 7481 pixelů	115	600
Obrázek o rozlišení 1280x720 s mezerou o velikosti 11680 pixelů	177	960
Videosekvence o rozlišení 352x288 o délce 36 framů s FPS 8 (bez použití metody pro odstranění statického objektu)	2	4
Videosekvence o rozlišení 480x360 o délce 331 framů s FPS 29,970 (bez použití metody pro odstranění statického objektu)	7	26
Videosekvence o rozlišení 960x720 o délce 185 framů s FPS 29,970 (bez použití metody pro odstranění statického objektu)	17	55
Videosekvence o rozlišení 960x720 o délce 320 framů s FPS 29,917 (s použitím metody pro odstranění statického objektu)*	155	837

Tabulka 9.1: Výkonnost aplikace

*V tomto případě bylo nahrazení s použitím metody pro odstranění statického objektu provedeno na dvou snímcích celé videosekvence.

Kapitola 10

Závěr

Cílem této diplomové práce bylo seznámit se s problematikou automatického odstraňování objektů v obraze a na základě dosažených poznatků navrhnout a implementovat aplikaci, která bude schopná se vypořádat s odstraněním nežádoucích objektů z videosekvencí (potažmo i statických snímků, tedy fotografií). Implementovanou aplikaci bylo též nutné otestovat a shrnout dosažené výsledky, výhody, nevýhody a navrhnout možná další rozšíření a úpravy.

Při analýze dané problematiky se dospělo k závěru, že pro zajištění vyšší kvality výstupů bude nutné, aby aplikace dokázala vymazat i objektem, který je po celou dobu trvání videosekvence statický. Z tohoto důvodu byla do aplikace zařazena metoda pro odstranění nežádoucího objektu z fotografie, jejíž princip je uveden v kapitole 3 a detaily implementace jsou popsány v kapitole 6.

Během návrhu algoritmu pro odstranění nežádoucího objektu z fotografie jsme vycházeli z předpokladu, že pohybující se objekt postupně odkryje pozadí, které původně zakrýval, a toto bude následně využito pro doplnění vzniklé mezery. Zpracování videosekvence je rozděleno do několika kroků (viz Kapitola 5). Důležiým krokem je sledování pohybu objektu ve videosekvenci (tracking), jehož teorii je věnována kapitola 4. Dalšími nezbytnými kroky jsou extrakce pozadí a nahrazení nežádoucího objektu pozadím. Obojí je rozebráno a popsáno v kapitole 7.

Implentovaná metoda byla podrobena sadě testů jak při zpracování fotografií, tak při odstraňování nežádoucích objektů z videosekvencí. Procesu testování a zjištěním o chování a omezeních implementovaného řešení pojednává kapitola 9. Během procesu testování byla aplikace nasazena na dvou různě výkonných strojích, což nám dalo zajímavé srovnání výkonnosti implementovaného řešení.

Aplikace, jež je popsána v této práci, je schopná odstranit nežádoucí objekt ze statického snímku, tedy fotografie, a též vymazat nechtěný objekt z videosekvence. Tato videosekvence musí být pořízena pomocí statické kamery bez zoom-ování. Výstupy aplikace dosahují uspokojivé kvality a to i se srovnáním s aplikací Adobe Photoshop.

Problematika odstraňování objektů z videosekvencí potažmo fotografií nás staví před několik nepříjemných problémů. Největším z nich je problém kvalita/výkon, který se nejvíce projevuje při odstraňování statických objektů. Jako důsledek tohoto dilematu byla vybrána hybridní metoda pro odstranění nežádoucího objektu z fotografie a v rámci implementované aplikace vložena možnost nastavení právě onoho poměru kvalita/výkon.

V dalším vývoji aplikace by bylo ideální provádět co možná největší část operací v rámci GPU, čímž by došlo ke značnému urychlení celé aplikace. Do budoucna by též bylo vhodné upravit zejména část věnovanou sledování objektu ve videosekvenci (tracker), aby byla

schopná se vypořádat i s videosekvencí snímanou pomocí klasickým způsobem s pohyblivou kamerou. Také by bylo vhodné implementovat další tracker, který by byl vhodnější pro sledování lidských postav než je tomu v námi navrženém řešení.

Literatura

[1] Bertalmío M., Sapiro G., Caselles V., Ballester C.: Image Inpainting. New Orleans, USA: SIGGRAPH 2000, 2000.

[2] Connexions [online]: Introduction to Optical Flow. http://cnx.org/content/m14208/latest/, 2006-12-22 [cit. 2012-05-12].

[3] Creativecow [online]: Simple Object Removal. http://library.creativecow.net/articles/drozda_jerzy/SimpleObjectRemoval.php, 2011-05-15 [cit. 2011-11-25].

[4] Criminisi, A., Perez, P., Toyama, K.: Region filling and object removal by exemplar-based image inpainting. IEEE Transactions on image processing, Září 2004.

[5] Ctirad Sousedík: *Syntéza textur*. bakalářská práce, FIT VUT v Brně, Brno, 2010.

[6] designfestival.com [online]: The Best New Features of Adobe Photoshop CS6. http://designfestival.com/the-best-new-features-of-adobe-photoshop-cs6/, 2011-07-05 [cit. 2011-11-25].

[7] DUŠEK, S: *Určení parametrů pohybu ze snímků kamery*. diplomová práce, Fakulta elektrotechniky a komunikačních technologií Vysoké učení technické v Brně, Brno, 2009.

[8] Forsyth D. A. and Ponce J.: *Computer Vision: A Modern Approach*. Prentice Hall, Pearson Education, Inc., January 2003.

[9] Škola Gimpu [online]: Odstranění objektů z obrázku. http://gimp.comuv.com/practice_removing.php, 2009-01-20 [cit. 2011-11-25].

[10] Homepage for Digital Image Inpainting [online]: Image Inpainting. http://www.math.ucla.edu/$\sim$imagers/htmls/inp.html, 2003-02-11 [cit. 2011-11-27].

[11] Horn B.K.P., Schunck B.G.: Determining optical flow. AI 17, 1981, s. 185–204.

[12] Hsu H.J., Wang J.F., Liao S.C.: A novel framework for object removal from digital photograph. IEEE Transactions on image processing, Září 2005.

[13] Hsu H.J., Wang J.F., Liao S.C.: A hybrid algorithm with artifact detection mechanism for region filling after object removal from a digital photograph. IEEE Transactions on image processing, Červen 2007.

[14] LIBIŠ, Z: *Sledování objektu ve video sekvencích*. diplomová práce, Fakulta elektrotechniky a komunikačních technologií Vysoké učení technické v Brně, Brno, 2011.

[15] M. Španěl: *Rozpoznávání gest ve video sekvencích.* diplomová práce, FIT VUT v Brně, Brno, 2003.

[16] Martin Burkot: *Automatická detekce vozidel: Smart car.* bakalářská práce, FIT VUT v Brně, Brno, 2009.

[17] Meijin L., Ying Z., Jiandeng H.: Video Background Extraction Based on Improved Mode Algorithm. International Conference on Genetic and Evolutionary Computing, Říjen 2009.

[18] Metody počítačového vidění [online]: Sledování objektů. http://cw.felk.cvut.cz/doku.php/courses/a4m33mpv/cviceni/4_sledovani_objektu/start, 2011-06-13 [cit. 2011-11-29].

[19] Near-regular Texture Synthesis [online]: Near-regular Texture Synthesis. http://www.cs.cmu.edu/$\sim$wclin/nrt.htm, 2008-08-15 [cit. 2011-11-29].

[20] New PAR/NL Scheme for Stochastic texture Interpolation [online]: Stochastic texture. http://www-scf.usc.edu/$\sim$byungoh/TI/TI.htm, 2009-05-12 [cit. 2011-11-29].

[21] OpenCV Wiki [online]: http://opencv.willowgarage.com/wiki/Welcome, 2012-03-17 [cit. 2012-04-09].

[22] Petr Bílek: *Významné body v obraze: detekce, korespondence a lokalizace ve 3D.* bakalářská práce, České vysoké učení technické v Praze Fakulta elektrotechnická, Praha, 2007.

[23] Point–In–Polygon Algorithm – Determining Whether A Point Is Inside A Complex Polygon [online]: http://alienryderflex.com/polygon/, 2007-08-15 [cit. 2012-04-09].

[24] Přepočet barev z RGB do YCbCr [online]: http://goro.czweb.org/ditr/ycbcr/prepocet.html, 2003-11-21 [cit. 2012-04-27].

[25] QT – Cross–platform application and UI framework [online]: http://qt.nokia.com/, 2011-03-01 [cit. 2012-04-09].

[26] Staněk Stanislav: *Hierarchické metody segmentace obrazu.* bakalářská práce, FIT VUT v Brně, Brno, 2008.

[27] The Lab Book Pages [online]: Otsu Thresholding. http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html, 2010-06-17 [cit. 2012-04-15].

[28] Vaidyanathan G., Lynch P.M.: Texture direction analysis using edge counts. Energy and Information Technologies in the Southeast, Duben 1989, s. 733–738.

[29] wikipedia.org [online]: Kirsch operator. http://en.wikipedia.org/wiki/Kirsch-Operator, 2012-04-04 [cit. 2012-04-15].

[30] wikipedia.org [online]: Fotomanipulace. http://cs.wikipedia.org/wiki/Fotomanipulace, 2012-04-27 [cit. 2012-04-30].

[31] Zdeněk Hrnčíř: *Optický tok v obrazových datech živých buněkh.* diplomová práce, Masarykova univerzita, Brno, 2006.

Příloha A

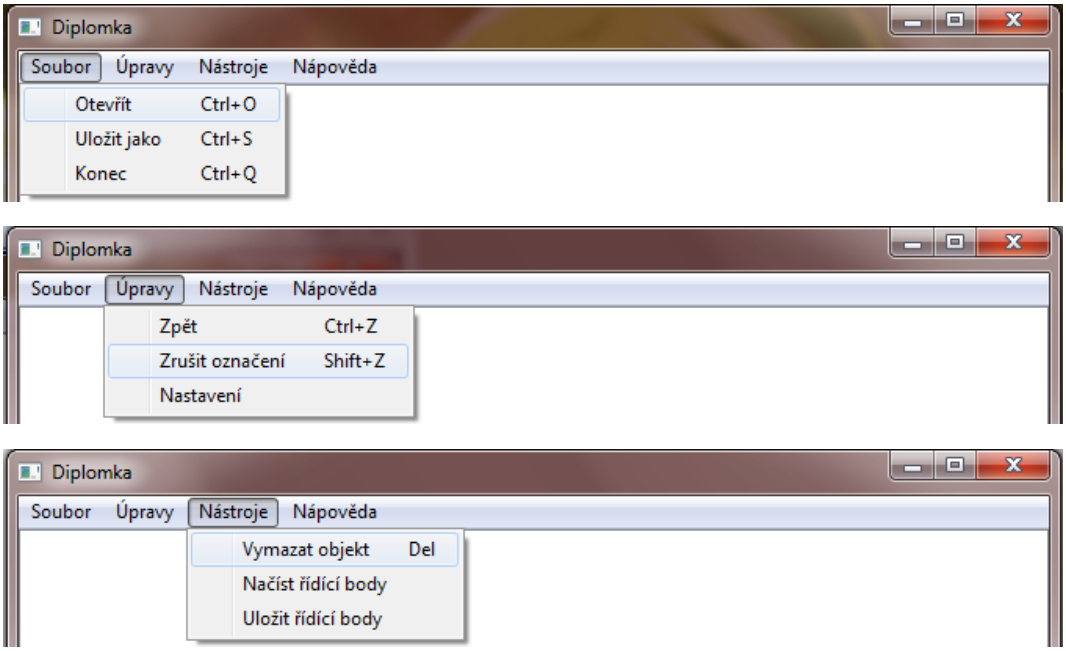
Obsah CD

- Zdrojové soubory
- Spustitelná verze práce
- Programová dokumentace
- Testovací obrázky a videa
- Ukázky výsledků po zpracování programem
- Videoprezentace práce
- Text práce ve fromátu PDF a zdrojové soubory pro LaTeX

Příloha B

Uživatelská příručka

Základní ovládání



Obrázek B.1: Základní ovládání

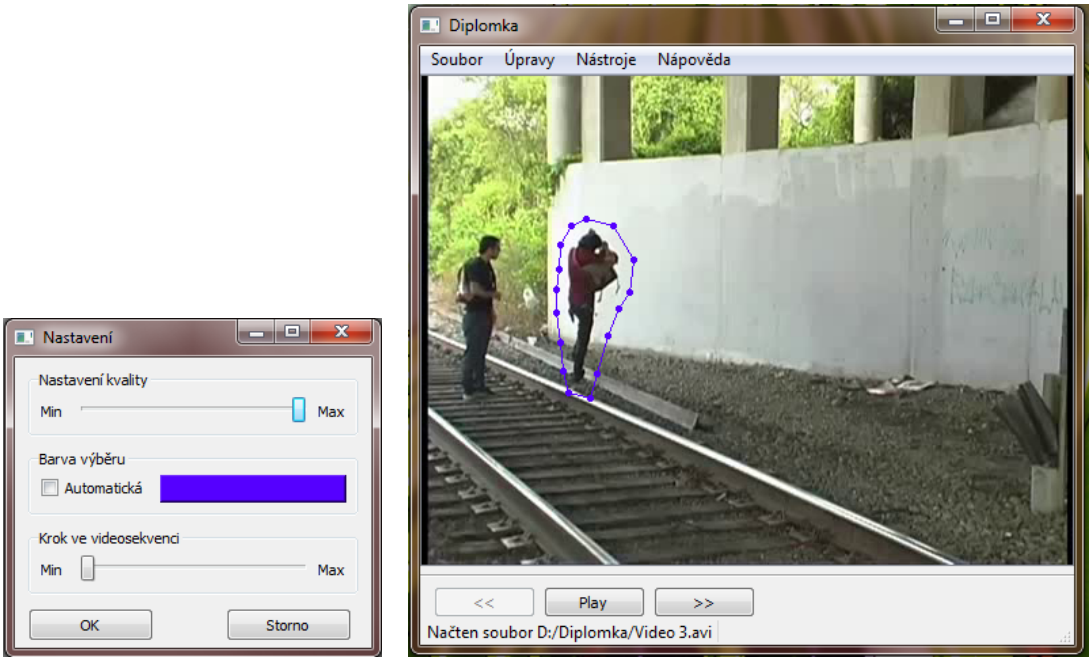
Aplikace disponuje standardním ovládáním pro načítání a ukládání snímků. Při práci s videosekvencí je uživatel vyzván k zadání jména cílového souboru ihned po aktivování metody pro odstranění nežádoucího objektu. **Pozor** – aplikace může mít problém při otevírání souborů, které mají ve svém názvu nebo v názvu cesty diakritiku.

Okno nastavení

Aplikace umožňuje nastavení několika parametrů. Nejdůležitějším je nastavení kvality, které určuje práh pro výběr z metod pro odstranění nežádoucího objektu z fotografie. Čím vyšší je nastavena kvalita, tím kvalitnější by měla být výsledná vyplněná oblast. Ovšem s vyšší kvalitou roste též doba zpracování.

Nastavením barvy výběru určíme jakou barvou chceme zobrazovat řídící body a čáry definující nežádoucí objekt. Volba automatická přepne na černobílý režim těchto bodů a čar, kdy se aplikace automaticky volí černé nebo bílé řídící body.

Posledním je krok ve videosekvenci, který slouží k nastavení rychlosti procházení v samotné videosekvenci pomocí tlačítek « a ».



Obrázek B.2: Okno nastavení a označený objekt

Označení nežádoucího objektu

Označení objektu probíhá umístěním řídících bodů pomocí myši. K umístění značky slouží levé tlačítko myši a výběr uzavřeme pomocí pravého tlačítka myši (dojde ke spojení prvního a posledního bodu). Bez uzavření masky (Obrázek B.2) není možné objekt vymazat. Pomocí klávesové zkratky Ctrl+Z lze vrátit krok zpět a kombinace Shift+Z odstraní všechny umístěné body.

Řídící body lze též načítat a ukládat. **Pozor** – aplikace se při ukládání řídících bodů neptá na název souboru a uloží jej automaticky pod stejným názvem jako je otevřený soubor,

Práce s videosekvencí

Výběr a označení nežádoucího objektu je shodný jako při práci s fotografiemi. Rozdíl je však v možnosti pohybu v rámci videosekvence (tlačítka « a »)

Vestavěný videopřehrávač

Aplikace disponuje možností přehrát si aktuálně načtenou videosekvenci v jednoduchém vestavěném videopřehrávači. Ten se spouští tlačítkem Play a na jeho deaktivaci slouží klávesa Enter.

Příloha C

Ukázky funkčnosti pro fotografie