VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

VLOŽENÍ 2D GRAFIKY DO SCÉNY ZABÍRANÉ STACIONÁRNÍ KAMEROU

INSERTION OF 2D GRAPHICS INTO A SCENE CAPTURED BY A STATIONARY CAMERA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **SON HAI NGUYEN**
AUTHOR

VEDOUCÍ PRÁCE **prof. Ing. ADAM HEROUT, Ph.D.**
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Nguyen Son Hai**
Program: Informační technologie
Název: **Vložení 2D grafiky do scény zabírané stacionární kamerou**
Insertion of 2D Graphics into a Scene Captured by a Stationary Camera
Kategorie: Zpracování obrazu

Zadání:

1. Seznamte se s problematikou rozšířené reality a zpracování obrazu z fixní kamery.
2. Prostudujte algoritmy detekce lidské postavy a algoritmy pro detekci pohybu v obraze.
3. Experimentujte s dostupnými řešeními podle bodu 2. Pořiďte za tím účelem vhodnou datovou sadu krátkých videí z nepohyblivých kamer.
4. Navrhněte řešení, které umožní vkládat 2D grafiku do vybraných míst ve scéně v duchu rozšířené reality.
5. Iterativně implementujte a vylepšujte vyvíjené řešení, sbírejte potřebná data.
6. Zhodnoťte dosažené výsledky a navrhněte možnosti pokračování projektu; vytvořte plakátek a krátké video pro prezentování projektu.

Literatura:

- Gary Bradski, Adrian Kaehler: Learning OpenCV; Computer Vision with the OpenCV Library, O'Reilly Media, 2008
- Richard Szeliski: Computer Vision: Algorithms and Applications, Springer, 2011

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3, značné rozpracování bodů 4 a 5.

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Herout Adam, prof. Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 1. listopadu 2018

Abstrakt

Rozšířená realita zobrazuje další informaci v reálném světě. Hlavním cílem této práce je dosažení přirozeného vzhledu vkládané 2D grafiky do scény pořízené stacionární kamerou. Přestože několik metod řeší problem segmentace popředí, tak většina z nich není dostanečně robustní v různonorodých scénách. Modifikovaná verze algoritmu ViBe produkuje nejpřesnější výsledky, avšak kvůli popisu alfa masky pouze dvěma hodnotami hrany segmentovaných objektů jsou hrubé. Pro vyhlazení daných hran je použit algoritmus Global Sampling Matting, provedené vyhlazení zlepšuje perceptuální kvalitu alfa masky. Jelikož ViBe nerozlišuje stíny, objevovaly se artefakty po zpětném vložení popředí do scény. Daný problem byl vyřešen navrhnutým algoritmem na segmentaci stínů, který porovnává odstín a texturu mezi modelem pozadí a popředím. Pro odstranění umělého vzhledu vložené grafiky, byl navrhnut algoritmus na propagaci textur. Segmentační a matting algoritmy jsou otestovány na různých datasetech. Výsledný system je následně demonstrován na různých scénách.

Abstract

Augmented reality visualizes additional information in real-world environment. Main goal of this work is achieving natural looking of the inserted 2D graphics in a scene captured by a stationary camera with possibility of real time processing. Although several methods tackled foreground segmentation problem, many of them are not robust enough on diverse datasets. Modified background subtraction algorithm ViBe yields best visual results, but because of the nature of binary mask, edges of the segmented objects are coarse. In order to smooth edges, Global Sampling Matting is performed, this refinement greatly increased the perceptual quality of segmentation. Considering that the shadows are not classified by ViBe, artifacts were occurring after insertion of segmented objects on top of the graphics. This was solved by the proposed shadow segmentation, which was achieved by comparing the differences between brightness and gradients of the background model and the current frame. To remove plastic look of the inserted graphics, texture propagation has been proposed, that considers the local and mean brightness of the background. Segmentation algorithms and image matting algorithms are tested on various datasets. Resulted pipeline is demonstrated in various scenes.

Klíčová slova

rozšířená realita, počítačové vidění, zpracování obrazu

Keywords

augmented reality, computer vision, image processing

Citace

NGUYEN, Son Hai. *Vložení 2D grafiky do scény zabírané stacionární kamerou.* Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce prof. Ing. Adam Herout, Ph.D.

Vložení 2D grafiky do scény zabírané stacionární kamerou

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana prof. Adama Herouta. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Son Hai Nguyen
13. května 2019

Poděkování

Tímto bych rád poděkoval svému vedoucímu prof. Adamu Heroutovi za jeho rady a morální podporu.

Obsah

1 Úvod **2**

2 Oddělení popředí od pozadí **3**
2.1 Rozdíl mezi segmentací a *matting* problémem 3
2.2 Matting rovnice 4
2.3 Definice trimapy a její využití v *matting* algoritmech 4

3 Segmentační metody **6**
3.1 Sémantická segmentace 6
3.2 Metoda klíčování 8
3.3 Metody odečítání pozadí 9

4 Metody řešení *matting* problému **15**
4.1 Systém DeepMatting 15
4.2 Algoritmus Global Sampling Matting 15

5 Evaluace algoritmů **20**
5.1 Segmentační algoritmy 20
5.2 *Matting* algoritmy v kombinaci s ViBe 22

6 Návrh systému pro vkládání 2D grafiky do videa **24**
6.1 Inicializace modelu pozadí 24
6.2 Hrubá segmentace 25
6.3 Vyhlazení segmentační masky 28
6.4 Vložení grafiky 30

7 Implementace a zhodnocení systému **34**
7.1 Paralelizace na GPU 34
7.2 Zhodnocení systému 37

8 Závěr **40**

Literatura **41**

Kapitola 1

Úvod

V posledních letech je rozšířená realita široce používána v různých aplikacích, od virtuální reklamy ve sportovních přenosech až po hry kompletně postavené na rozšířené realitě. Ve sportovních přenosech je rozšířená realita využita především pro dosažení neinvazivního přidání obrazového materiálu, kde pozadí je zakryto vloženým obsahem, ovšem objekty popředí jsou stále vidět. Proto se vložený obsah jeví jako by byl namalován na podklad.

Existující systémy [5, 23] se zabývají převážně pozicováním vkládaných objektů kvůli pohybující se kameře. Nicméně přirozený vzhled virtuálních objektů již není řešen, z toho důvodu vložená grafika vypadá uměle.

Nejsložitější část systému na vkládání virtuálního obsahu představuje segmentace popředí, která se skládá z několika algoritmů. Pro docílení co nejpřesnější segmentace popředí byly vytvořeny algoritmy na segmentaci stínů a generování trimapy, dále byly představeny modifikace algoritmů ViBe [3] a Global Sampling Matting [16], které produkují přesnější alfa masku než stávající verze. Také byl navrhnut algoritmus na propagaci textur, který vytváří efekt, že je vložená grafika promítáná na podklad.

V následujích kapitolách bude popsán návrh a implementace systému na vkládání 2D grafiky do scény pořízené stacionární kamerou, výběr dílčích algoritmů je podpořen evaluací zobrazenou v kapitole 5. Pro zlepšení perceptuálního vzhledu výsledného snímku, budou představeny dva nové algoritmy na propagaci textur a segmentaci stínů, dále budou navrženy modifikace algoritmu Global Sampling Matting [16], které zvyšují přesnost produkované alfa masky. Na závěr bude implementace akcelerovaná verze navrženého systému bude otestována na různých scénách zahrnujících fotbal, házenou či zelené plátno.

Kapitola 2

Oddělení popředí od pozadí

Rozlišení popředí od pozadí je klíčovým prvkem systému na vkládání grafiky v duchu rozšířené reality. Některé metody odečítání pozadí [9, 3, 28, 22] klasifikují jako popředí všechny objekty, které se pohybují a nezůstávají stát na jednom místě příliš dlouho. V rámci sémantické segmentace [25, 6] se předpokládají za popředí všechny objekty určité třídy, například člověk, či auto.

2.1 Rozdíl mezi segmentací a *matting* problémem

Metody produkující pouze binární alfa masku, kde hodnota 1 označuje popředí a hodnota 0 pozadí, se označují jako segmentační metody.

Jak uvádí R. Szelisky [30], při pořizování fotografie se v barvě hran objektů popředí neprojeví pouze barva popředí, ale kombinace jak barvy popředí, barvy tak pozadí, proto je při kompozici popředí s jiným pozadím binární alfa maska nedostačující. Alfa maska nabývající hodnot z intervalu $\langle 0, 1 \rangle$ netrpí tímto nedostatkem a umožňuje interpolaci mezi barvou pozadí a popředí, metody produkující takovou alfa masku se nazývají alpha matting metody. Rozdíl mezi binární maskou a alfa maskou produkovanou matting metodami je zobrazen na obrázku 2.1.

(a) Vstupní snímek (b) Binární maska (c) Alfa maska produkovaná matting metodami

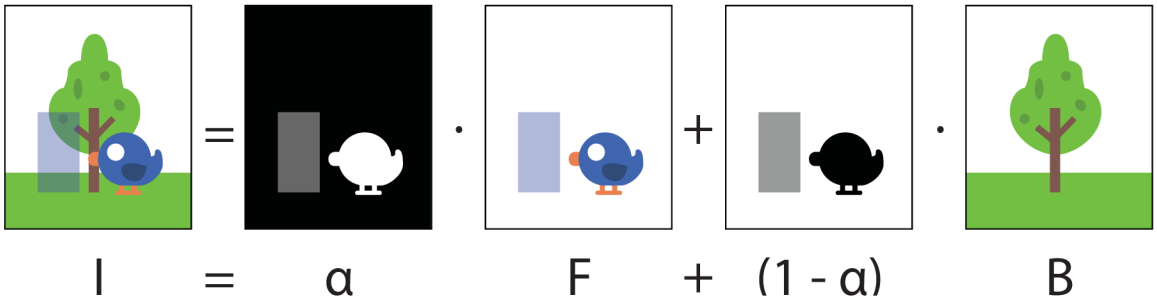
Obrázek 2.1: Porovnání výstup matting algoritmu se segmentačním algoritmem. Segmentační algoritmy produkují pouze binární alfa masku (b) narozdíl od matting algoritmů, které vytvářeji alfa masku (c) v rozsahu hodnot $\langle 0, 1 \rangle$.

2.2 Matting rovnice

Rovnice matting vyjadřuje kompozici popředí **F** a pozadí **B** pomocí alfa masky α, může být zapsána v následujícím tvaru:

$$\mathbf{I} = \alpha\mathbf{F} + (1 - \alpha)\mathbf{B}, \tag{2.1}$$

kde **I** představuje vstupní snímek. V matting rovnici jsou pro každý pixel známy pouze 3 hodnoty (barva **I**) a 7 hodnot je neznámých (barva **F**, barva **B** a koeficient α), což znamená, že neexistuje jednoznačné řešení této rovnice.



Obrázek 2.2: Ilustrace rovnice (2.1). Hodnoty alfa masky α leží v intervalu $\langle 0, 1\rangle$, kde bílá barva označuje hodnotu 1 a černá barva označuje hodnotu 0.

2.3 Definice trimapy a její využití v *matting* algoritmech

Jak bylo uvedeno v přechozí kapitole, matting rovnice nemá jednoznačné řešení, proto je potřeba poskytnout více známých hodnot, například formou trimapy, kde pouze pro zlomek pixelů označených jako neznámé je řešena matting rovnice, zbylé pixely jsou již oklasifikované jako pozadí nebo popředí, tím vzniknou tři množiny pixelů:

1. Pixely, které patří určitě do popředí.
2. Množina pixelů, které patří určitě do pozadí.
3. Neznámé pixely, jejichž α koeficient je odhadnut matting algoritmem pomocí dvou předchozích množin.

Takové rozdělení se nazývá trimapa, která je uložena jako šedotónový obraz, kde intezita o hodnotě 255 označuje popředí, hodnota 0 pozadí a hodnota 128 neznámé pixely, jak je uvedeno na obrázku 2.3. Trimapu lze vygenerovat automaticky nebo ručně, avšak ručně vytvořit trimapu je nutné označit vhodně neznámou oblast, proto se používají takzvané „scribble“ [21], kde uživatel označí popředí a pozadí malým množstvím křivek.

(a) Vstupní snímek se „scribbles“

(b) Älfa maska odhadnutá ze „scribbles“

(c) Trimapa

(d) Alfa maska odhadnutá z trimapy

Obrázek 2.3: Porovnání alfa masky vygenerované ze scribbles a z trimapy. (a) Vstupní obrázek se scribbles s řídkým označením: černé křivky označují pozadí, bílé křivky označují popředí. (b) Alfa maska odhadnutá z řidkého vstupu od uživatele. (c) Ručně vytvořená trimapa (šedá = neznámá oblast, černá = pozadí, bílá = popředí). (d) Alfa maska odhadnutá z trimapy. Obrázky převzaty z Levin et al. [21].

Kapitola 3

Segmentační metody

R. Szelisky [30] definuje segmentaci jako shlukování pixelů s podobnými vlastnostmi, tato definice je obecná a umožňuje různé interpretace, ovšem v této práci bude zmiňována segmentace pouze v kontextu segmentace popředí.

Jak již bylo zmíněno v kapitole 2.1, segmentační metody produkují binární alfa masku, které lze docílit například pomocí metod odečítání pozadí nebo pomocí sémantické segmentace, která je realizována konvolučními neuronovými sítěmi.

3.1 Sémantická segmentace

Sémantická segmentace neshlukuje pixely do segmentů pouze na základě vlastností pixelu, ale bere v potaz také prostorovou informaci – jaké vlastnosti mají okolní pixely.

Sémantická segmentace je realizována konvolučními neuronovými sítěmi (CNN), jež jsou založeny na poskytnutých datech, ze kterých se následně naučí charakteristiku hledaných tříd. Přestože charakteristiky naučené konvoluční neuronovou sítí jsou popsány váhami konvolučních jader, není možné jednoznačně určit, jak daná síť charakterizuje dané třídy – není možné zaručit, že se síť naučila správné charakteristiky.

Tento nedostatek lze řešit velkým množstvím různorodých trénovacích dat, což vede k lepší generalizaci výsledné CNN, většina druhů CNN potřebuje k učícím datům také *„ground truth“* (GT) neboli referenční data. Vytvoření referenčních dat pro sítě, které pouze klasifikují objekt na obrázku číslem třídy, je oproti vytvoření segmentační masky pro každý snímek méně časově náročné. Datasety na klasifikaci obsahují milióny oklasifikovanách dat, například ImageNet [26], část datasetu je zobrazena na obrázku 3.1. Nicméně vytvoření segmentační masky je narozdíl od předem zmíněného druhu dat časově náročný na vytvoření. Z toho důvodu segmentační datasety nejsou velké s porovnáním s ImageNet datasetem.

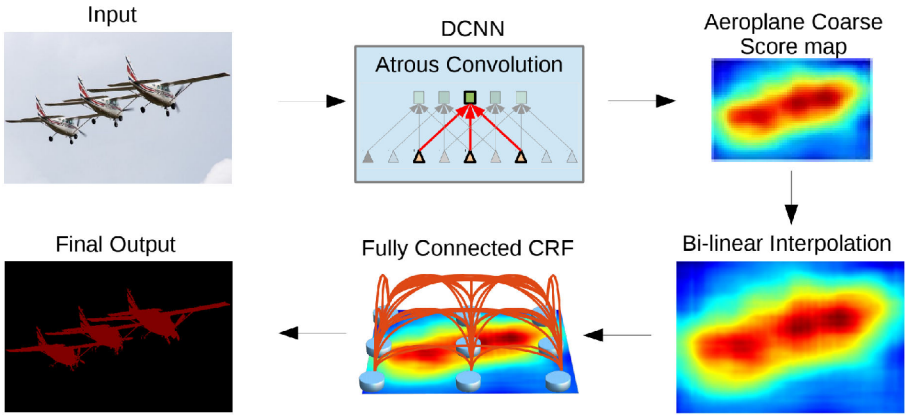
3.1.1 DeepLab

Systém DeepLab [6] (obrázek 3.2) je postaven na hlubokých konvolučních neuronových sítích (Deep Convolutional Neural Networks, DCNN). Autoenkodér představuje jádro DeepLab systému [6]. Hlavní inovace představují takzvaná „atrous“ konvoluční vrstva a plně propojené podmíněná pole (Conditional Random Field, CRF) [20], která zlepšují lokalizaci hranic.

Konvoluční neuronové sítě získávají prostovou informaci z obrazu třemi různými způsoby:

(a) ImageNet (b) DensePose COCO

Obrázek 3.1: (a) Ukázka datasetu ImageNet [26], kde obrázek obsahuje pouze informaci o typu obsahu, který se v něm nachází. (b) Ukázka datasetu DensePose COCO [25], kde každý obrázek obsahuje přesné pozice lidských částí v obrázku. Obrázky převzaty z Guler et al. [25] a z Deng et al. [26]



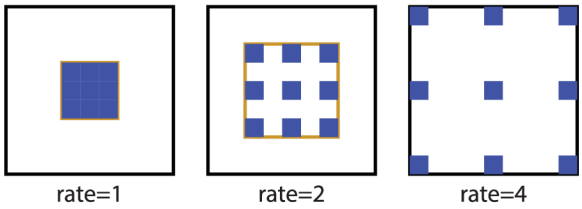
Obrázek 3.2: Jádro systému Deeplab [6] představuje hluboká konvoluční neuronová síť, která může nabývat libovolné architektury, například Resnet101 [18] nebo VGG-16 [27], která je propojená s atrous prostorovou pooling pyramidou sestravenou z atrous konvolučních vrstev.Obrázek je převzat z článku Chen et al. [6].

1. Použitím hlubokých neuronových síťí lze získat prostorovou informaci, avšak s hloubkou sítě postupně také mizí propagované gradienty, což má za následek vrstvy, jejichž parametry se neaktualizují.

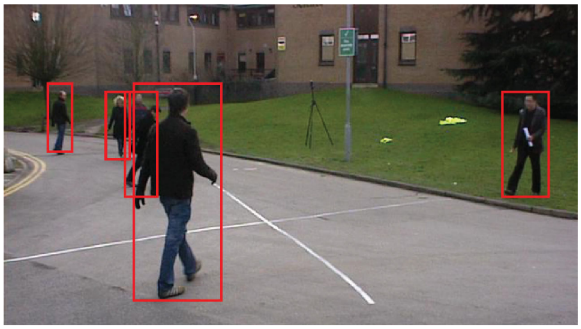
2. Podvzorkováním vstupu se vstup zmenší o násobek dvou, tím pádem jádro o stejné velikosti pokrývá větší prostor obrazu.

3. Využitím velkých konvolučních jader se pokryje větší část obrazu, nicméně velká konvoluční jádra vyžadují velký počet parametrů.

Atrous konvoluce pracuje na způsobu využití velkého konvolučního jádra, avšak snižuje počet parametrů tím, že je dané jádro řídké – velké množství parametrů je nastaveno na nulu bez možnosti je změnit, jak ukazuje obrázek 3.3.



Obrázek 3.3: Atrous konvoluce [6] umožňuje získání prostorového kontextu z obrazu s použitím nízkého počtu parametrů. Na obrázku lze vidět atrous konvoluce s různým rozpětím.



Obrázek 3.4: Červené oblasti představují oblasti, ve kterých se nachází člověk, nalezené detektorem, který je využíván systémem DensePose [25]. Vstupní snímek je z datasetu PETS09.

3.1.2 DensePose

Systém DensePose [25] lze rozdělit na dvě části: detektor oblastí s výskytem lidí a segmentační část. Konvoluční neuronová síť s architekturou Resnet50 [18] představuje dektetor oblastí zájmu – oblasti ve kterých se vyskytuje člověk, jak lze vidět na obrázku 3.4.

Tyto oblasti představují vstup pro druhou část systému, který vytvoří segmentační masku. Kombinací systému DenseReg [14] a architektury Mask-RCNN [15] autoři článku DensePose[25] navrhli nový systém jménem „DensePose-RCNN“, který realizuje segmentaci obrazu.

3.2 Metoda klíčování

Mezi široce používané metody ve filmovém průmyslu patří metoda klíčování (označována také Chroma key). Do význačných vlastností této metody patří znalost pozadí, které tvoří jediná barva, která se vhodně volí, aby měla co nejmenší barevný průnik s objekty popředí, často se volí modré nebo zelené pozadí. Ukázku zeleného pozadí lze vidět na obrázku 3.5.

Metodu Chroma key lze popsat rovnící (3.1), která se podobá základní rovnici (3.2) popisující metody odečítání.

$$\mathbf{S} = \begin{cases} foreground, & (\sum_{i=1}^{3} |K_i - \mathbf{I_i}|) > T \\ background, & \text{otherwise.} \end{cases} \tag{3.1}$$

Na rozdíl od rovnice (3.2) popisující odečítání pozadí rovnice (3.1) *Chroma key* neporovnává popředí se modelem pozadí, ale porovnává všechny pixely snímku $\mathbf{I}$ vůči klíčované barvě $\mathbf{K}$.

Obrázek 3.5: Zelené plátno se často používá ve filmovém průmyslu kvůli snadnější segmentaci popředí.

(a) First frame (b) Input frame (c) (d)

Obrázek 3.6: Při špatné inicializaci pozadí metod založených na odečítání pozadí se v segmentační masce mohou objevit takzvaní duchové. Pokud je například model pozadí inicializován prvním snímkem (a) sekvence, může se duch objevit v segmentační masce (d), duch je způsoben pozadím, které je špatně klasifikováno jako popředí. Při inicializaci modelu pozadí například mediánem určitého počtu snímku, lze docílit absence duchů (c) při zpracování začátku videa.

3.3 Metody odečítání pozadí

Metody odečítání pozadí modelují pozadí $\mathbf{B}$, které se odečítá od testovaného snímku $\mathbf{I}$, je-li rozdíl nulový, testovaný pixel má stejnou barvu jako pixel pozadí. V opačném případě je daný pixel označen za popředí F.

Takový přístup ovšem nepočítá se šumem kamery, z toho důvodu namísto testování nulového rozdílu pixelů byl zaveden práh T, jak uvádí následující rovnice:

$$\mathbf{S} = \begin{cases} foreground & |\mathbf{I} - \mathbf{B}| > T \\ background & otherwise, \end{cases} \tag{3.2}$$

kde $\mathbf{S}$ značí segmentační masku, kde hodnota 0 symbolizuje pozadí a hodnota 1 popředí. I přes tuto úpravu stále přetrvává problém dlouhodobé změny pozadí a pohyby pozadí. Při použití nevhodného snímku pozadí, kde je stále vidět část popředí, vznikne artefakt zvaný duch. Tento arterfakt zapříčíní klasifikaci pozadí za popředí v místě výskytu artefaktu, jak lze vidět na obrázku 3.6.

Algoritmus Mixture of Gaussians (MoG) [29] jako první komplexněji modeloval pozadí – pomocí směsice gaussových funkcí, aby předešel vzniku artefaktů z důvodů dlouhodobých změn osvětlení a pohybu objektu pozadí. Artefakt typu duch se stále může objevit, nicméně daný algoritmus postupně zmenšuje jeho rozsah.

Postupem času byl MoG [29] překonán algoritmy (ViBe [3, 9], SubSENSE [28] či AMBER [32]), které modelují pozadí pomocí vzorků.

3.3.1 Algoritmus ViBe

Jak již bylo zmíněno, algoritmus ViBe [3] modeluje pozadí pomocí množiny vzorků, funkčnost celého algortimu lze rozdělit do dvou částí: detekce popředí a aktualizace modelu pozadí. Jeho kompletní funkce je popsána algoritmem 1.

Detekce popředí není elementarní jak uvádí rovnice (3.2), algoritmus ViBe využívá L2 normu pro výpočet vzdálenosti mezi barvou pozadí a barvou testovaného snímku $\mathbf{I}$. Jelikož ViBe modeluje pozadí n vzorky, porovnává se testovaný pixel se všemi vzorky pozadí $\mathbf{M_i}$, kde i označuje index vzorku. Přesahuje-li barevná vzdálenost testovaného pixelu v určitém počtu vzorků T_{SC} práh T_D, je daný pixel označen za popředí, jak je uvedeno v rovnici (3.3).

$$\mathbf{S} = \begin{cases} foreground, & (\sum_{i=1}^{n}[||\mathbf{M_i} - \mathbf{I}|| > T_D]) > T_{SC} \\ background, & \text{otherwise.} \end{cases} \tag{3.3}$$

Autoři článku ViBe [3] nastavili parametry následovně: $T_D = 20$, $T_{SC} = 2$, $n = 20$, tyto parametry byly nastaveny pro barvy jsou v rozsahu $[0, 255]$.

Aktualizace modelu pozadí Každý pixel, který by klasifikován jako pozadí, má $1/\Phi$ pravděpodobnost, že bude uložen v náhodně vybraném vzorku jako pozadí. Následně je provedena aktualizace sousedů, která má také pravděpodobnost $1/\Phi$, že bude provedena. Aktualizace sousedů je opět provedena v náhodném vzorku, kde se vybere náhodně soused druhého řádu a poté se jeho hodnota přepíše hodnotou testovaného pixelu, který byl klasifikovan jako pozadí.

3.3.2 Rozšíření algoritmu *ViBe*

Neznámější modifikace algoritmu ViBe [3] je nazvána ViBe+ [9], pro zlepšení přesnosti segmentační masky obsahuje následují modifikace oproti originálnímu algoritmu:

Oddělení masky popředí a aktualizační masky činí nejvýznačnější modifikaci – pomocí této modifikace se razantně zlepší kvalita segmentační masky, jak lze vidět na obrázku 3.7. Hlavní myšlenka spočívá v použití morfologických operací, které nejdříve vymažou bloby s obsahem menším než 10 a vyplní díry s obsahem menším než 20. Tato maska je následně použita jako segmentační maska.

Podstata aktualizační masky je opačná oproti segmentační masce – v té se dbá především o opravu špatně klasifikovaných pixelů pozadí označených jako popředí. V aktualizační masce se cílí k opravě pixelů popředí klasifikovaných jako pozadí, čímž se docílí konzervativnější aktualizaci modelu pozadí. Aktualizační maska je realizována vyplněním děr v segmentační masce o obsahu menší než 50.

Algoritmus 1: Pseudokód algoritmu ViBe [3]. Hodnoty byly nastaveny autory článku následovně: $\Phi = 16, N = 20, T_D = 20, T_{SC} = 2$.

```
count ← 0
for i ← 1 to N do
    if ||I(x, y) − M_i(x, y)|| > T_D then
        count ← count + 1
end

if count > T_SC then
    S(x, y) ← foreground
else
    S(x, y) ← background
    if rand(0, Φ − 1) = 0 then
        r ← rand(1, N)
        M_r(x, y) ← I(x, y)
    if rand(0, Φ − 1) = 0 then
        r ← rand(1, N)
        M_r(x + rand(−1, 1), y + rand(−1, 1)) ← I(x, y)
end
```

Inhibice propagace realizuje další omezení pro konzervativnější aktualizaci modelu pozadí. ViBe [3] využívá propagaci pixelu pozadí mezí své sousedy druhého řádu, aby se docílilo postupného mazání artefaktů typu duch. Přestože je tento systém efektivní vůči duchům, způsobuje také postupné klasifikování statických objektů popředí jako pozadí. Pro odstranění tohoto nedostatku autoři ViBe+[9] navrhují omezení propagace mezi sousedy, jestliže hodnota gradientu blobu pozadí je větší než 50.

Utilizace jiné funkce pro barevné vzdálenosti. ViBe+ využívá rozdílnou funkci pro výpočet barevné vzdálenosti převzatou z Kim et al. [19], která převzatá rovnice je popsána následujícími rovnicemi:

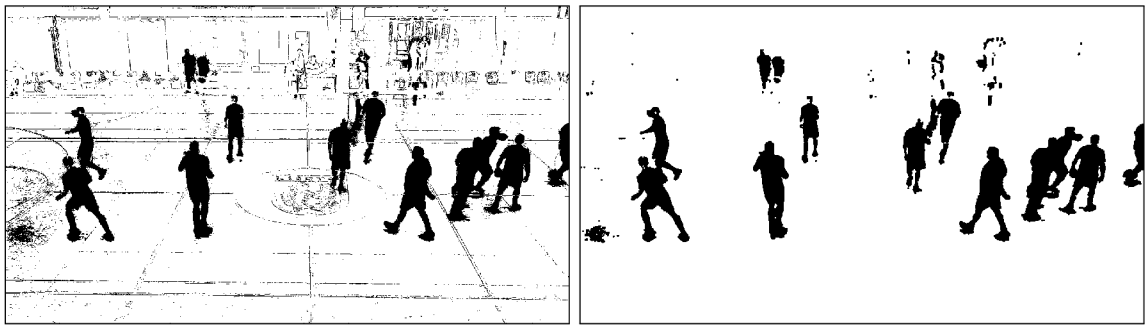
$$\langle I(x,y), M_i(x,y)\rangle^2 = (\bar{R}R + \bar{G}G + \bar{B}B) \tag{3.4}$$

$$p^2 = \frac{\langle I(x,y), M_i(x,y)\rangle^2}{||M_i(x,y)||^2} \tag{3.5}$$

$$dist(x,y) = \sqrt{||I(x,y)||^2 - p^2}, \tag{3.6}$$

kde $I(x,y) = (R,G,B)$ označuje pixel testovaného snímku, $M_i(x,y) = (\bar{R},\bar{G},\bar{B}$ označuje barvu pixelu pozadí i-tého vzorku.

Detekce blikajících pixelů funguje na principu porovnání aktuální a předešlé aktualizovací masky. Pokud pixel předešlé aktualizační masky $\mathbf{M_i}$ je jiný než pixel aktuální aktualizační masky $\mathbf{M'}$, zvýší se blikající úroveň (anglicky *blinking level*) o hodnotu 15, jinak se úroveň dekrementuje, jak lze vidět v rovnici (3.7)



(a) (b)

Obrázek 3.7: Při použití morfologických operací navrhnutých v algoritmu ViBe+ [9] se kvalita segmentační masky (a) razantně zlepší (b). Barvy jsou invertované pro lepší viditelnost, tudíž v tomto obrázku představuje černá barva popředí a bílá brava pozadí.

$$blibkinbLevel(x,y)' = \begin{cases} blinkingLevel(x,y) + 15 & M_i'(x,y) \neq M_i(x,y) \\ blinkingLevel(x,y) - 1 & otherwise. \end{cases} \tag{3.7}$$

Pixely s blikající úrovní větší než 30 jsou označeny jako blikající a jsou odstraněny z aktualizační masky.

3.3.3 Algoritmus *SubSENSE*

Přestože je algoritmus SubSENSE [28] na algoritmu ViBe [3], přináší několik modifikací, například jako využití LBSP příznaků [4] nebo automatická úprava prahů. Funkcionalitu algoritmu SubSENSE lze rozdělit na dvě části:

Segmentace se vůči metodě ViBe liší především ve využití LBSP příznaků, které popisují texturu okolí pixelu, LBSP příznaky ilustruje obrázek 3.8. Využití těchto příznaků v detekční části algoritmu umožňuje detekci objektů popředí, které mají podobnou barvu jako pozadí (kamuflované objekty), každý příznak je popsán šestnáctibitovým číslem, která se vypočítá pomocí následující funkce:

$$LBSP(x) = \sum_{p=1}^{P} d(i_p, i_x) \cdot 2^p, \tag{3.8}$$

kde

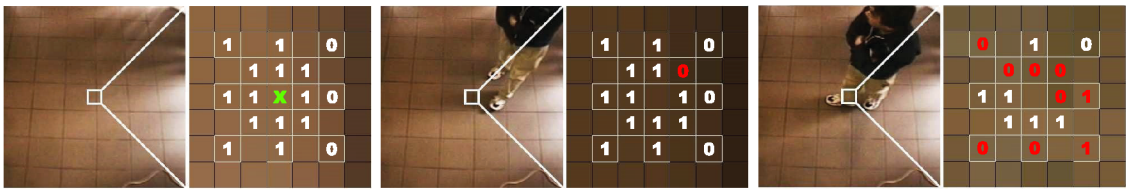
$$d(i_p, i_x) = \begin{cases} 1 & |i_p - i_x| \leq T_d \\ 0 & otherwise. \end{cases} \tag{3.9}$$

P označuje index pozice v LBSP [4] masce (obrázek 3.8), i_x symbolizuje bod, jehož příznak okolní textury se počítá. Autoři SubSENSE [28] experimentálně zjistili, jestliže-li práh T_d má hodnotu 0.3, dosahuje segmentace nejpřesnějších výsledků.

Přestože LBSP příznaky jsou schopny rozlišovat rozdílné textury, pro větší přesnost produkované segmentační masky jsou LBSP příznaky zkombinovány s detekcí barevné rozdílnosti. Fůze daných dvou metod je popsána algoritmem 2, následně algoritmus 3 představuje finální formu segmentační části pracující s modelem pozadí založeném na vzorcích.

Algoritmus 2: Psedokód pro detekci popředí vůči jednomu vzorku modelu pozadí [4]. Představuje fůzi detekce popředí založené klasickém algoritmu ViBe [3] a metodě využívající LBSP příznaky [4], které detekují rozlišnosti v textuře mezi modelem pozadí a testovaným snímkem.

function *classify*$(\mathbf{I}, \mathbf{M})$**:**
- $int\_dist\_sum \leftarrow 0$
- $desc\_dist\_sum \leftarrow 0$
- **for** $c \leftarrow 1$ **to** $channels\_count$ **do**
  - $int\_dist \leftarrow |\mathbf{I}(x,y) - \mathbf{M}(x,y)|$
  - **if** $int\_dist > T_D$ **then**
    - **return** $foreground$
  - $desc\_dist \leftarrow LBSP(\mathbf{I}(x,y)) \oplus LBSP(\mathbf{M}(x,y))$
  - **if** $desc\_dist \geq T_{desc}$ **then**
    - **return** $foreground$
  - $int\_dist\_sum \leftarrow int\_dist\_sum + int\_dist$
  - $desc\_dist\_sum \leftarrow desc\_dist\_sum + desc\_dist$
  - **if** $int\_dist\_sum \leq T_D$ & $desc\_dist\_sum \leq T_{desc}$ **then**
    - **return** $foreground$
- **end**
- **return** $background$



(a) (b) (c)

Obrázek 3.8: LBSP příznaky [4] se používají pro detekci rozdílných textur, jestliže rozdíl intenzit vůči „prostřednímu“ pixelu a danému pixelu je větší než určitý práh, je příznak nastaven na hodnotu 1. Tím vznikne pro každý pixel 16bitový řetězec, který popisuje texturu v okolí testovaného pixelu (a). Při zastínění okolí (b) LBSP příznak [4] nemění značně svou hodnotu, má-li objekt popředí rozdílnou texturu než pozadí (c), LBSP příznak [4] obsahuje jinou hodnotu než příznak pozadí.

Algoritmus 3: Finální algoritmus pro detekci popředí při použití N vzorků.

```
count_samples ← 0
for i ← 1 to N do
    if classify(I(x, y), M_i(x, y)) = foreground then
        count_samples ← count_samples + 1
end

if count_samples ≥ T_SC then
    S(x, y) ← foreground
else
    S(x, y) ← background
end
```

Aktualizace modelu pozadí probíhá identicky vůči algoritmu ViBe [3]. Parametry T_D, T_{desc} a Φ se ovšem nastavují automaticky na základě pohybové entropie, která je vypočítána jako pohyblivý průměr rozdílu LBSP příznaků [4].

Kapitola 4

Metody řešení *matting* problému

Existující metody lze kategorizovat do tří skupin – na metody založené na propagaci, metody založené na vzorcích a metody založené na strojovém učení. Mezi známé metody založené na propagaci se řadí *Closed Form Matting* (CFM) [21], která se umístila druhá ve VideoMatting benchmarku [10] , tato metoda se snaží zjistit interpolaci neznámých pixelů ze známých oblastí, danou interpolaci vypočítá vyřešením afinitní matice, jak uvádí K. He [16].

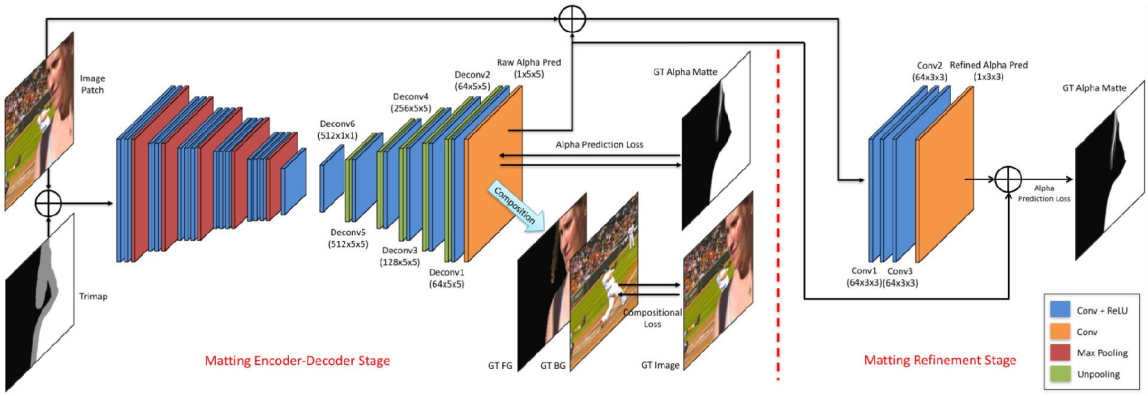
Další třídu matting algoritmů představují metody založené na vzorcích. Tyto metody nepoužívají při odhadu alfa masky každý pixel vstupních obrázků, nýbrž pouze omezenou množinu navzorkovaných pixelů. Do této třídy algoritmů lze zařadit například algoritmus *Global Sampling Matting* (GSM) [16] nebo Shared Matting [12].

4.1 Systém DeepMatting

DeepMatting [33] dosahuje nejlepších výsledků ve VideoMatting benchmarku [10], tento systém je postaven na architektuře kodér-dekodér a používá dvě nové loss funkce představené v Xu et al. [33]. Tvůrci sítě DeepMatting použili VGG-16 [27] síť jako kódér, tradiční VGG-16 obsahuje plně propojené vrstvy, ty byly nahrazeny konvoluční vrstvou. Dekodér je obdobný jako kodér, avšak obsahuje méně konvolučních jader. Následně byla přidána malá síť, která má za úkol vylepšení výstupní masky, jak schéma celého systému lze vidět na obrázku 4.1. Parametry sítě jsou inicializovány pomocí Xavier inicializace [13].

4.2 Algoritmus Global Sampling Matting

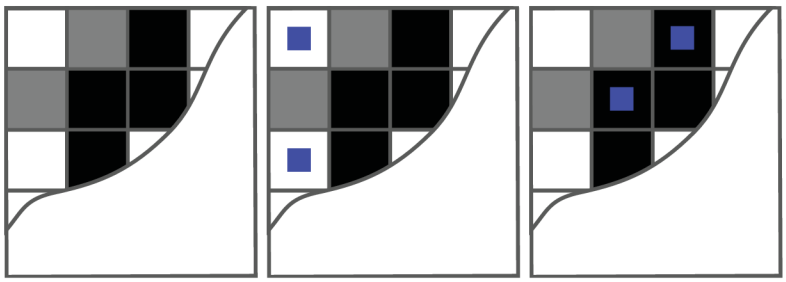
Tato kapitola popisuje algoritmus Global Sampling Matting (GSM) [16], jenž patří do třídy matting algoritmů založených na vzorcích. Podstatou algoritmu je nalezení vhodných vzorků, které budou použity pro odhad neznámých pixelů. Pro každý neznámý pixel je nutné nalézt vhodný vzorek popředí a pozadí, tyto vzorky jsou vybírány na základě ohodnocovací funkcí, která bere v potaz barevnou podobnost i prostorovou vzdálenost vzorku od neznámého pixelu. Z důvodu velkého prostoru, který tvoří vzorky popředí a pozadí, nelze prohledat pro kadý pixel celý prostor vzorků, z toho důvodu tvůrci GSM použili upravenou verzi algoritmus PatchMatch [2] pro prohledání daného prostoru. Algoritmus Global Sampling Matting využívá Guided Filter [17] pro dodatečné vyhlazení výsledké alfa masky.



Obrázek 4.1: Systému DeepMatting [33] je postaven na architektuře kodér-dekodér, kde VGG-16 architektura [27] představuje kodér. Druhou část systému představuje malá konvoluční neuronová síť, která zajišťuje vylepšení výsledné alfa masky. Obrázek převzat z Xu et al. [33].

4.2.1 Vytvoření globální množiny vzorků

Jak vyplývá z názvu algoritmu, množina vzorků se nazývá **globální množina vzorků** Vzorky v globální množině vzorků se dělí do dvou kategorií – vzorky pozadí a vzorky popředí, tím vzniknou dvě podmnožiny globální množiny. Pokud pixel, který je v trimapě označen jako popředí, má v sousedství prvního řádu neznámý pixel, přidá se tento pixel do množiny vzorků popředí, jak ilustruje obrázek 4.2. Tvorba množiny vzorků pozadí probíhá analogicky.



(a) (b) (c)

Obrázek 4.2: Pro vstupní trimapu (a) se vyhledají všechny pixely popředí, které mají v prvním řádu sousedství neznámý pixel, následně dané pixely popředí jsou vloženy do globální množiny vzorků. Tento algoritmus je vyobrazen na obrázku (b), kde šedé pixely označují neznámou oblash, bílé označují popředí a černé označují pozadí, modré značky představují pixely, které jsou přidány do globání množiny. Nalezení vzorků pozadí (c) funguje analogicky k hledání vzorků popředí.

4.2.2 Objektivní funkce

Aby bylo možné vybrat vhodný pár vzorků, je nutné ohodnotit vhodnost daného páru vzorků, k tomu slouží ohodnocovací funkce, která se skládá ze dvou částí:

Barevná vhodnost vzorků představuje první část ohodnocovací funkce. Nejprve je odhadnuta alfa masky $\hat{\alpha}$ pomocí interpolace mezi párem vzorků $(\mathbf{F}^i, \mathbf{B}^j)$:

$$\hat{\alpha} = \frac{(\mathbf{I} - \mathbf{B}^j)(\mathbf{F}^i - \mathbf{B}^j)}{||\mathbf{F}^i - \mathbf{B}^j||^2}, \tag{4.1}$$

kde $\mathbf{F}^i$ představuje vzorek popředí, $\mathbf{B}^j$) značí vzorek pozadí, i a j označují indexy vzorků. Po odhadnití $\hat{\alpha}$ honoty se pomocí ohodnovací funkce (4.2) prozkoumá, jak dobře lze vyjádřit barvu neznámého pixelu jako lineární kombinaci barev páru vzorků pomocí následující rovnice:

$$\xi_c(\mathbf{F}^i, \mathbf{B}^j) = ||\mathbf{I} - (\hat{\alpha}\mathbf{F}^i) + (1 - \hat{\alpha})\mathbf{B}^j)|| \tag{4.2}$$

Jak uvádí He [16] funkce (4.2) intuitivně vyjadřuje barevnou vzdálenost mezi barvou neznámého pixelu a barvou vytvořenou vzorky $\mathbf{F}^i$ a $\mathbf{B}^j$.

Prostorová vzdálenost vzorků od neznámého pixelu je zahrnuta do výsledné ohodnocovací funkce, protože při velkém množství vzorků může nastat situace, kdy velmi vzdálený pár vzorků dobře popisuje barvu neznámého pixelu. Z toho důvodu autoři GSM [16] vytvořili funkci, která ohodnocuje prostorovou vzdálenost vzorku od neznámého pixelu:

$$\xi_s(\mathbf{F}^i) = \frac{||x_{\mathbf{F}^i} - x_{\mathbf{I}}||}{D_F}, \tag{4.3}$$

kde $x_{\mathbf{F}^i}$ a $x_{\mathbf{I}}$ představují prostorové souřadnice vzorku popředí a neznámého pixelu. Proměnná D_F označuje vzdálenost nejbližšího vzorku popředí vůči neznámému pixelu, tedy $D_F = min_i||x_{\mathbf{F}^i} - x_{\mathbf{I}}||$. Tato normalizace zajišťuje nezávislost prostorové ceny na absolutní vzdálenosti vzorku a neznámého pixelu.

Konečná podoba ohodnovací funkce má následující podobu:

$$\xi(\mathbf{F}^i, \mathbf{B}^j) = \omega\xi_c(\mathbf{F}^i, \mathbf{B}^j) + \xi_s(\mathbf{F}^i) + \xi_s(\mathbf{B}^j), \tag{4.4}$$

kde ω představuje koeficient, který určuje poměr mezi barevnou vhodností a prostorovou vzdáleností. Tvůrci algoritmu GSM [16] nastavili $\omega = 1$, když ξ_c je vypočítáno z barev v rozsahu $\langle 0, 255\rangle$. Nízká hodnota ξ reprezentuje pár vzorků, který dobře barevně vyjadřuje neznámý pixel a je zároveň blízko u neznámého pixelu.

4.2.3 Algoritmus *SampleMatch*

Jak uvádí tvůrci algoritmu GSM [16], pro vstupní obrázek o velikosti 800x600 pixelů dosahuje velikost globální množiny pro popředí n_f a pozadí n_b přibližně $10^3 \sim 10^4$ prvků, tudíž při úplném prohledání prostoru vzorků by bylo nutné pro každý otestovat $n_f \times n_b = 10^6 \sim 10^8$ párů vzorků.

Seřadit vzorky lze podle určitého kritéria, autoři vyzkoušeli různá kritéria řazení a došli k závěru, že na daném kritériu příliš nezáleží, a proto zvolili řazení podle intenzity barvy.

Množina vzorků popředí a množina vzorků pozadí vytvoří 2D prostor, který se nazývá **FB** prostor. Pro snížení výpočetní složitosti autoři GSM při prohledávání **FB** prostoru využivají upravenou verzi algoritmu PatchMatch [2], kterou nazývají SamplePatch [16].

Pro každý neznámý pixel se uchovává aktuální nejlepší pár vzorků $\Phi(x, y) = (\mathbf{F}^i, \mathbf{B}^j)$, nejlepší pár vzorků $\Phi(x, y)$ se na začátku inicializuje náhodnou hodnotou z **FB** prostoru. Po inicializaci se následně střídají dva kroky: **propagace** a **náhodná procházka**. Daný proces se volá iterativně, tudíž dané kroky se volají v následujícím pořadí $(P^1, S^1, P^2, S^2, \ldots, P^N, S^N)$, kde P^k představuje krok propagace a S^k značí krok náhodné procházky, autoři článku označili 10 iterací jako dostačující.

Propagace zkoumá nejlepší vzorky v sousedství druhého řádu $\Phi(x', y')$ pixelu $\mathbf{I}(x, y)$, zda nebudou vhodnější pro daný pixel. Jak uvádí He [16], propagace využívá skutečnosti, že sousedící pixely mívají podobnou hodnotu barvu popředí/pozadí. Krok propagace je popsán následující funkcí:

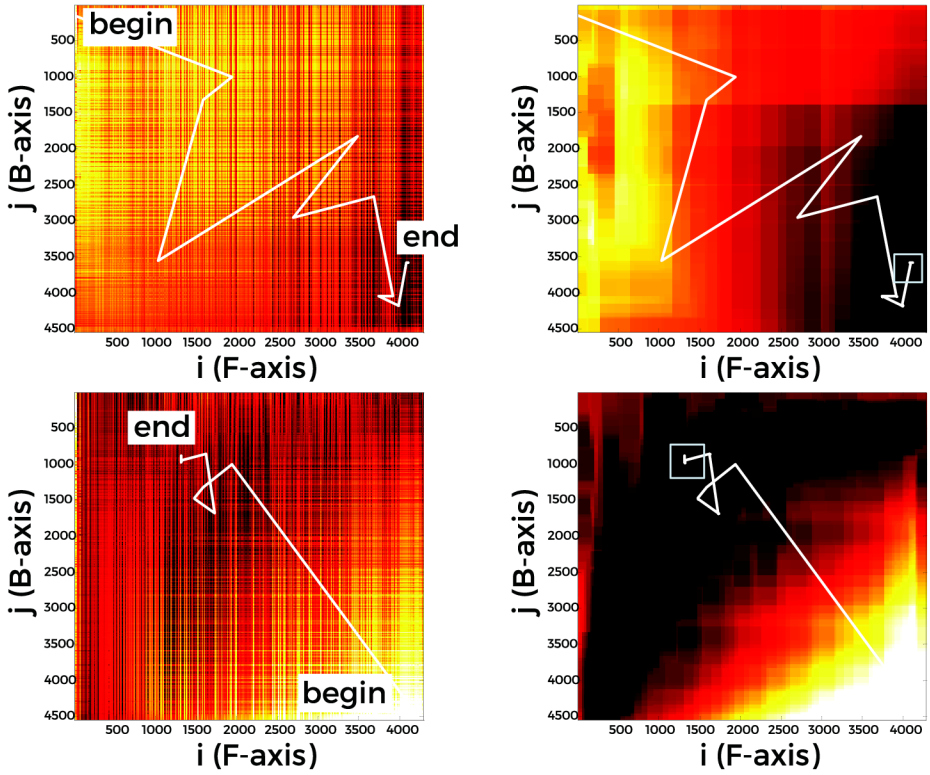
$$\Phi(x, y) = \underset{\Phi(x', y')}{\arg\min}\, \xi(\Phi(x', y')). \tag{4.5}$$

Intuitivně je tedy $\Phi(x, y)$ aktualizováno, pouze pokud má pár vzorků $\Phi(x', y')$ nižší cenu.

Náhodná procházka pro každý neznámý pixel $\mathbf{I}(x, y)$ zkouší sekvenci náhodných vzorků z **FB** prostoru. Prohledávané okolí se v **FB** prostoru exponenciálně zmenšuje, výběr testovaných párů vzorků popisuje následující rovnice:

$$(i_k, j_k) = (i, j) + \omega\beta^k * \mathbf{R}_k, \tag{4.6}$$

kde i_k a j_k jsou indexy testovaných vzorků, i a j představují indexy aktuálních nejlepších vzorků, β^k představuje k-tou mocninu $\beta = 0.5$, ω označuje velikost prohledávaného prostoru, tedy $\omega = [||\mathbf{F}||, ||\mathbf{B}||]$. $\mathbf{R}_k$ je dvojrozměrný řádkový vektor, jehož prvky jsou náhodná čísla z rovnoměrného rozložení v rozsahu $\langle -1, 1\rangle$. Pro k platí $k \in \{x : x \in \mathbb{N} \wedge max(\omega)\beta^x \geq 1\}$. Aktualizace nejlepšího páru vzorků $\Phi(x, y)$ proběhne, pouze pokud nový pár $\mathbf{F}^{i_k}, \mathbf{B}^{j_k}$ má nižší cenu. Ukázka náhodné procházky zobrazena na obrázku 4.3.



Obrázek 4.3: Ilustrace procesu náhodné procházky, která exponenciálně zmenšuje prohledávané okolí – levé obrázky ukazují cestu náhodné procházky v **FB** prostoru, pravé obrázky zobrazují stejnou procházku. Pro lepší vizualizaci, cenová mapa byla vyfiltrována minimalizačním filterem. Světlost barvy označuje velikost ceny, tedy světlá barva označuje vysokou cenu. Obrázek byl převzat z článku K. He et al. [16].

Kapitola 5

Evaluace algoritmů

V této kapitole bude popsána evaluace dílčích algoritmů pro segmentaci popředí, která proběhla před finálním návrhem systému, jelikož bylo nutné zjistit, které dílčí metody jsou nejrobustnější. K porovnání metod je použita také metrika RMSE, kterou lze popsat následující rovnicí:

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}, \tag{5.1}$$

kde n značí počet pixelů, y_j a $\hat{y}_j$ představují porovnávané snímky, tedy ground truth a porovnávaný snímek.

5.1 Segmentační algoritmy

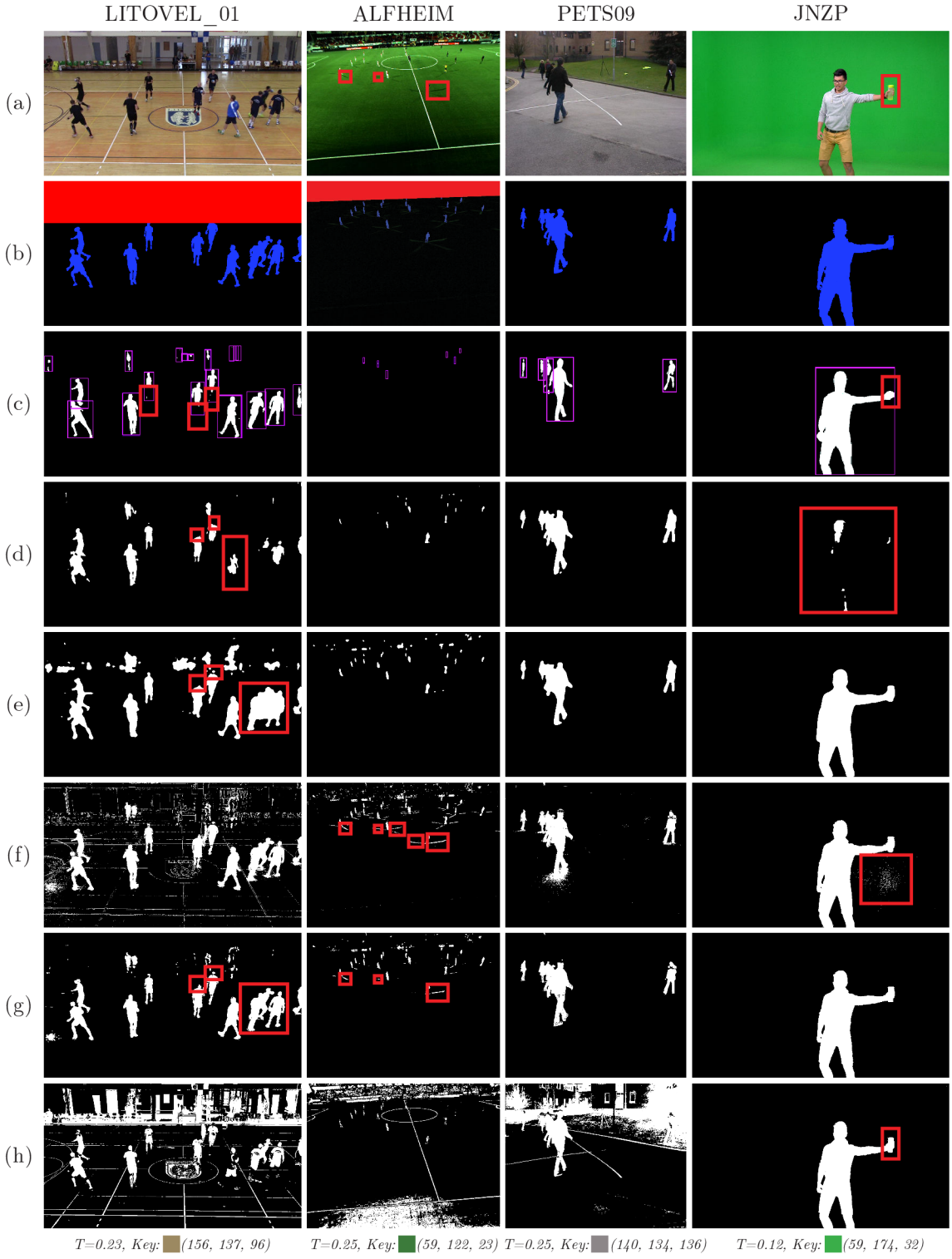
V této kapitole jsou zobrazeny výsledky následujících systémů **DensePose** [25], **DeepLab** [6], **SubSENSE** [28], **ViBe** [3], **ViBe#** a **Chroma-key**. Algoritmus ViBe# reprezentuje modifikovanou verzi algoritmu ViBe [3], která je více popsána v kapitole 6.2.

Jak zobrazuje obrázek 5.1, systémy DeepLab a DensePose selhávají v segmentaci „kamuflovaných“ nebo příliš malých osob. Tato skutečnost je zapříčeněna nedostatkem různorodých dat, na kterých byly systémy natrénovany. Přestože SubSENSE dosahuje v rámci metriky RMSE podle tabulky 5.1 nízké chybovosti, daný algoritmus vynechává detaily a zaobluje ostré hrany, jak dokazuje obrázek 5.1.

Ačkoliv maska vyprodukovaná algoritmem ViBe je velmi zašuměná a zahrnuje stín do popředí, jako jediná zachovává detaily a většinu obsahu objektů popředí, z toho důvodu ViBe využit v navrženém systému.

Tabulka 5.1: Porovnání segmentačních metod pomocí metriky RMSE. *Metrika RMSE je vypočítána pouze z jednoho snímku.*

Algoritmus	PETS09	LITOVEL_01	ALFHEIM	JNZP	**Avg. RMSE**
DensePose [25]	31.9649	39.3016	19.0938	26.8841	29.3111
DeepLab [6]	19.1179	47.2691	**15.6315**	68.8796	37.7245
SubSENSE [28]	**18.9051**	54.1864	20.1774	25.2947	29.6409
ViBe [3]	28.9328	61.8984	29.0520	22.6819	35.6413
ViBe#	20.1093	**35.2635**	20.4717	19.1658	**23.7526**
Chroma-key	113.7187	107.0621	94.9830	**15.0399**	82.7009



Obrázek 5.1: Evaluace segmentačních metod nad různými datasety. (a) Zobrazuje vstupní snímek, (b) zobrazuje ground truth v ssf formátu (modrý kanál = maska popředí, zelený kanál = maska stínů, červená = mimo oblast zájmu). Zobrazené segmentační masky odpovídají následujícím metodám: (c) **DensePose** [25] (růžové oblasti označují detekované oblasti zájmu), (d) **DeepLab** [6], (e) **SubSENSE** [28], (f) **ViBe** [3], (g) **ViBe#**, (h) **Chroma-key**, specifikovaný rovnicí (3.1), spodní parametry označují použitou klíčovanou barvu a práh.

5.2 *Matting* algoritmy v kombinaci s ViBe

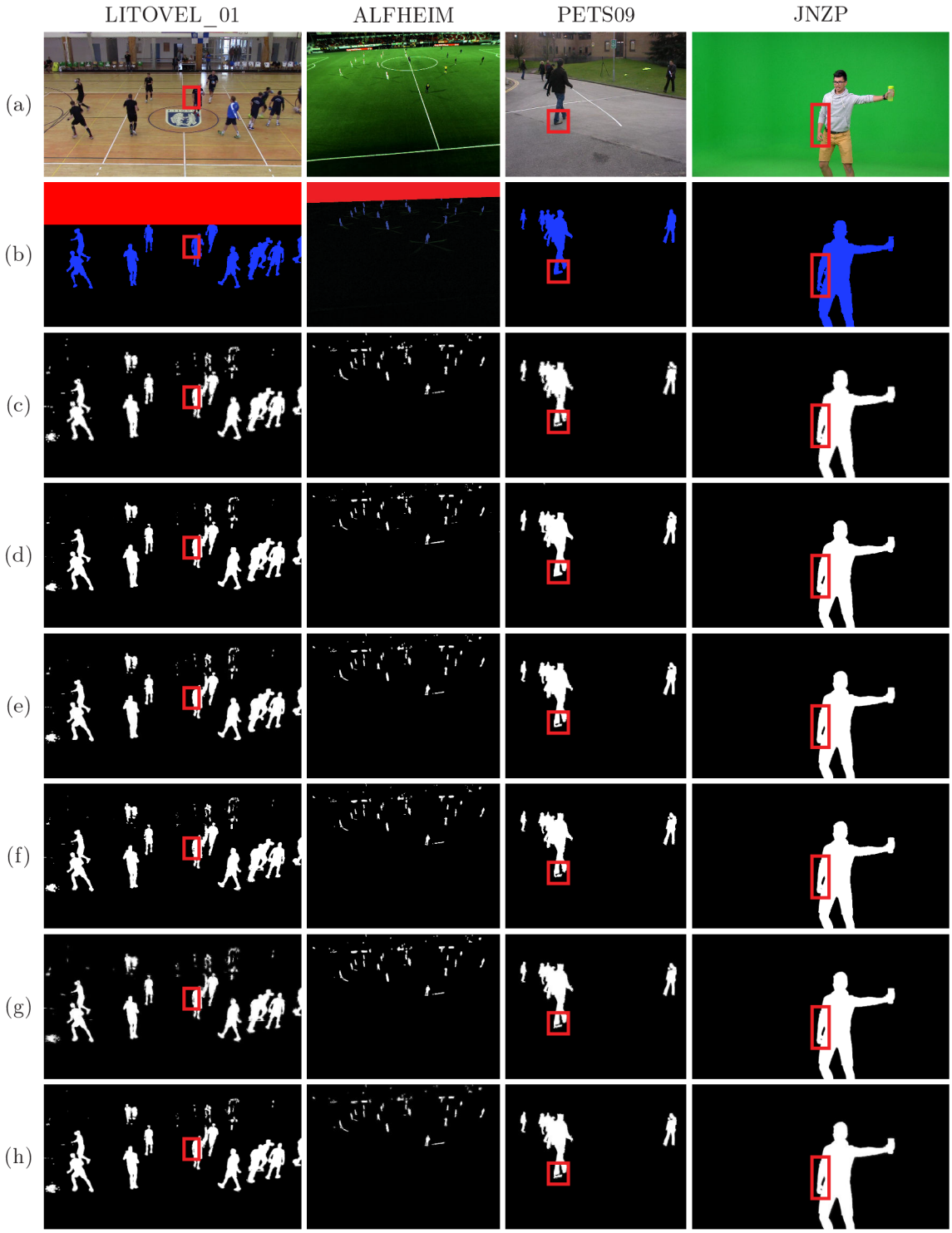
V této kapitole jsou provedeny experimenty s následujícími algoritmy **DeepMatting** [33], **Closed Form Matting** (CFM) [21], **Shared Matting** [12], **Global Sampling Matting** (GSM) [16] a **MGSM** (modifikovaná verze GSM, která je popsána v kapitole 6.3.2).

Rozdíly mezi matting metodami zobrazenými na obrázku 5.2 nejsou příliš radikální jako v evaluaci segmentačních metod, daný fenomén ukazuje i tabulka 5.2, kde se metrika RMSE také příliš mnoho nemění. Nicméně i tyto malé změny jsou viditelné po vložení grafiky do snímku.

Metoda DeepMatting dosahuje nejlepších výsledků ve VideoMatting bechmarku [10], přesto v mých experimentech tento systém mírně zaostává za mnohem staršími algoritmy jako GSM [16] nebo CFM [21], to je způsobeno kvalitou vygenerované trimapy, jelikož ve VideoMatting benchmarku [10] neznámá oblast představuje relativně velkou oblast vůči vstupnímu snímku narozdíl od automaticky vygenerované trimapy, kde neznámá oblast pokrývá především hrany vysegmentovaných objektů a malé díry v daných objektech. Na základě výsledků mých experimentů je ve výsledném systému použit algoritmus GSM, který dosahuje nejlepších výsledků hned po jeho modifikované verzi, která je popsána v kapitole 6.3.2.

Tabulka 5.2: Porovnání matting metod pomocí metriky RMSE. *Metrika RMSE je vypočítána pouze z jednoho snímku.*

Algoritmus	PETS09	LITOVEL_01	ALFHEIM	JNZP	**Avg. RMSE**
ViBe#	20.1093	35.2635	20.4717	19.1658	23.7526
DeepMatting [33]	18.0358	34.9800	18.7648	18.4985	22.5698
CFM [21]	18.1254	33.9091	18.0556	18.8331	22.2308
Shared [12]	17.3682	33.0373	18.0758	18.1938	21.6688
GSM [16]	16.2281	**29.1107**	18.1327	17.8631	20.3337
MGSM	**16.1880**	31.8924	**15.7739**	**16.5525**	**20.1017**

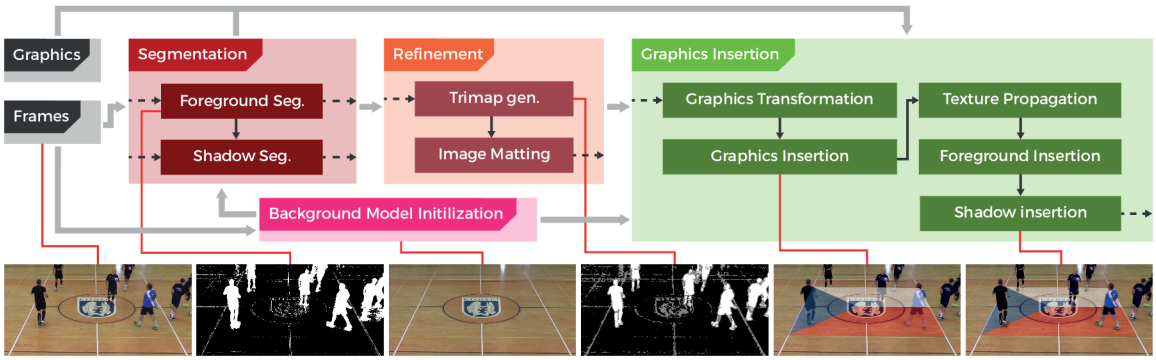


Obrázek 5.2: Porovnání matting metod nad datasety LITOVEL_01, PETS09, ALFHEIM a JNZP. (a) Zobrazuje vstupní snímek, (b) ground truth v ssf formátu. (c) Zobrazuje vstupní trimapu, která byla vygenerována automaticky ze segmentační masky vyprodukovanou algoritmem **ViBe#**. Následně jsou zobrazeny výsledky matting metod v následujícím pořadí: (d) **DeepMatting** [33], (e) **Closed Form Matting** [21], (f) **Shared Matting** [12], (g) **Global Sampling Matting** (GSM) [16], (h) **Modifikovaný GSM**.

Kapitola 6

Návrh systému pro vkládání 2D grafiky do videa

Tato kapitola popisuje návrh systému pro vkládání 2D grafiky do videa pořízeného stacionární kamerou, jenž je schopen zpracovávat snímky v reálném čase. Navržený systém je ilustrován na obrázku 6.1. Dále budou popsány dva nové algoritmy v kapitolách 6.2.1 a 6.4.1 na segmentaci stínů a na propagaci textur.



Obrázek 6.1: Schéma navrženého systému lze rozdělit na tři části: **hrubá segmentace**, **vyhlazení** a **vložení grafiky**. Algoritmus **ViBe** [3] realizuje hrubou segmentaci popředí, po odstranění stínů ze segmentační masky jsou provedeny open a close morfologické operace pro odstranění šumu. Poté je trimapa vygenerována pomocí dilatace hran segmentační masky. Vygenerovaná trimapa je společně se vstupním snímkem použita algoritmem **Global Sampling Matting** [16] k vyhlazení a vylepšení alfa masky. Následně je 2D grafika transformována a vložena podle uživatelem zadaných požadavků. Pro odstranění plastického vzhledu je využita **propagace textur**, zbývající část pak již představuje pouze vložení objektu popředí a stínů nad vloženou grafiku.

6.1 Inicializace modelu pozadí

Modelů pozadí založených na vzorcích je nutné incializovat, nejjednodušší způsob inicializace představuje nastavení všech vzorků pozadí na první snímek videa. Jsou-li v prvním

snímku přítomny objekty popředí, mohou vznikat takvzaní duchové, jak dokazuje obrázek 3.6.

(a) LITOVEL_01 (b) ALFHEIM (c) PETS09 (d) JNZP

Obrázek 6.2: Pozadí odhadnuté vypočítáním mediánu 300 snímků. Ve většině datasetů bylo 300 snímků dostačující, ovšem v méně dynamických scénách se v odhadnutém pozadí mohou stále objevit části objektů popředí, které následně způsobí vznik duchů.

Nicméně samotnou inicializaci lze (zne)užít ke krátkodobému zlepšení přesnosti. Selhala-li aktualizace pozadí a aktuální model pozadí neodpovídá skutečnému pozadí, je možné okamžitě přenastavit všechny vzorky pozadí.

Odhadnutí pozadí jsem dosáhl pomocí výpočtu mediánu několika set snímků – u více dynamických scén bylo dostačující odhadnout pozadí ze 300 snímků, jak lze vidět na obrázku 6.2.

Odhadnutí pozadí založené na modusu [35] dosahuje vyšší rychlosti než řešení založené na mediánu, jak lze vidět v evaluaci provedené v článku Zheng et al. [35]. Modus potřebuje méně snímků než medián pro přesné odhadnutí pozadí – na vybrané metodě ve výsledku tolik nezáleží, jelikož předpověď pozadí je kontrolována uživatelem.

6.2 Hrubá segmentace

Jak je uvedeno v kapitole 5.1, algoritmus ViBe [3] dosahuje vhodných výsledků, které lze mírnými úpravami razantně vylepšit, z toho důvodu je použit v navrženém systému. Vhodnými výsledky se myslí skutečnost, že algoritmus ViBe produkuje převážně chyby typu *false positive* než *false negative*, jak dokazuje obrázku 6.3. *False positive* chyby se lépe zpracovávájí, jelikož se jedná pouze o „odstraňování“ pixelů ze segmentační masky. Další chybu, kterou produkuje algoritmus ViBe, je nerozlišování objektů popředí od jejich stínů, proto jsou stíny označeny jako popředí.

6.2.1 Segmentace stínů

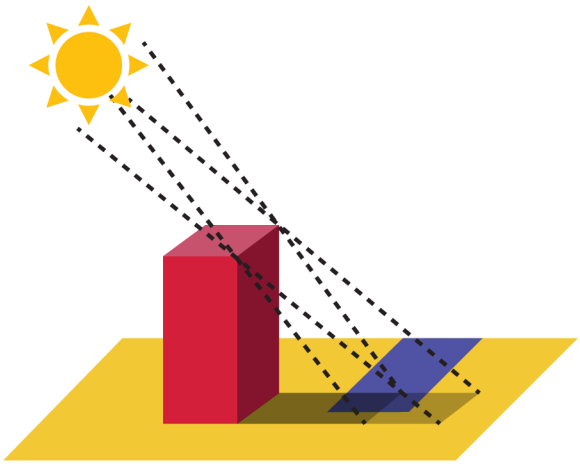
Jak bylo zmíněno v předchozí kapitole, algoritmus ViBe [3] nerozlišuje objekty popředí od jejich stínů, tím mohou vzniknout artefakty stejné jako vyobrazené na obrázku 6.5.

Divya [8] definuje stín jako část prostoru, který není plně osvětlen skrze zaclonění jiným objektem, proto stín vypadá jako tmavší pozadí. Vzniku stínu ilustruje obrázek 6.4.

Metody pro segmentaci stínů navržené v Venkatraman et. al [31] nebo Cucchiara et al. [7] využívají pro segmentaci stínů snímek pozadí. Přestože obě metody mají k dispozici snímek pozadí, ani jedna metoda nevyužívá skutečnosti, že textura zastíněné plochy je stejná jako nezastíněné – neplatí pro plně zastíněnou plochu, kdy se žádný paprsek neodrazí od dané plochy. Proto jsem vytvořil algoritmus na segmentaci stínů, který bere v potaz také texturu.

Po převedení testovaného snímku $\mathbf{I}$ a pozadí $\mathbf{M_1}$ do LAB barevného prostoru, se následně porovnají pouze A a B kanály, které představuji odstín barvy. Porovnání odstínů popisuje následující rovnice:

Obrázek 6.3: Segmentační maska vyprodukovaná algoritmem ViBe [3] je velmi zašuměná, z toho důvodu je nutné další zpracování.



Obrázek 6.4: Okluze určité plochy jiným objektem způsobí zmenší množství paprsků světla dopadajících na danou plochu, což způsobí tmavší vzhled části plochy, která je zacloněna jiným objektem.

$$\Delta\mathbf{D} = \sqrt{\sum_{i\in(A,B)} (\mathbf{I_i} - \mathbf{M_i})^2} \tag{6.1}$$

Tmavší barva je pouze jeden z příznaků stínu, další příznak činí zachování textury zastíněné plochy. Pro porovnání textur se vstupní snímky převedou do šedotónového obrazy, na které se aplikuje Sobelův operátor. Výpočet rozdílů mezi texturami vyjadřuje následující rovnice:

$$\Delta\mathbf{G} = ||\mathbf{G_I} - \mathbf{G_M}||, \tag{6.2}$$

kde $\mathbf{G_I}$ značí gradienty vypočítané testovaného snímku a$\mathbf{G_M}$ označuje gradienty pozadí.

Finální klasifikace porovnává, zda rozdíl textury a odstínu barvy je dostatečně malý, aby daný pixel mohl být klasifikován jako stín. Dané porovnání je popsáno následující rovnicí:

(a) (b)

Obrázek 6.5: Přestože na vstupním snímku (a) nejsou stíny příliš viditelné, v segmentaci mohou vytvořit artefakty, které lze vidět na obrázku (b).

$$\mathbf{O} = \begin{cases} \mathbf{L_{bg}} - \mathbf{L_{frame}}, & \Delta\mathbf{D} < T_C \wedge \Delta\mathbf{G} < T_G \\ 0, & \text{otherwise}, \end{cases} \tag{6.3}$$

prahy T_C, T_G byly experimentálně nastaveny na hodnoty $T_C = 8$ a $T_G = 60$.

(a) Vstupní snímek (b) Bez segmentace stínů (c) Se segmentací stínů a zpětnou rekonstrukcí (d) Po vyhlazení stínů

Obrázek 6.6: Ukázka segmentace stínů navrženým algoritmem a zpětné rekonstrukce stínů. Intenzita stínů byla třikrát zvýšená pro lepší viditelnost stínů.

Vyprodukovaná maska **O** označuje intenzitu stínů v barevném prostoru LAB. Při zpětné rekonstrukci stínů je potřeba pouze odečíst masku **O** od L kanálu daného snímku, výsledky navrženého algoritmu jsou zobrazeny na obrázku 6.6.

Po segmetaci stínů proběhne odstranení stínů ze segmentační masky, které je popsáno následující rovnicí:

$$\mathbf{S_s} = \begin{cases} 0 & \mathbf{O} \neq 0 \\ \mathbf{S}, & \text{otherwise}, \end{cases} \tag{6.4}$$

kde $\mathbf{S_s}$ značí segmentační masku po odstranění stínů.

6.2.2 Modifikace algoritmu *ViBe*

Algoritmus ViBe+ není použitý, jelikož nemá veřejně dostupný zdrojový kód a má implementaci nedosáhla stejných výsledků jako uvedených v článku, proto jsem převzal použití

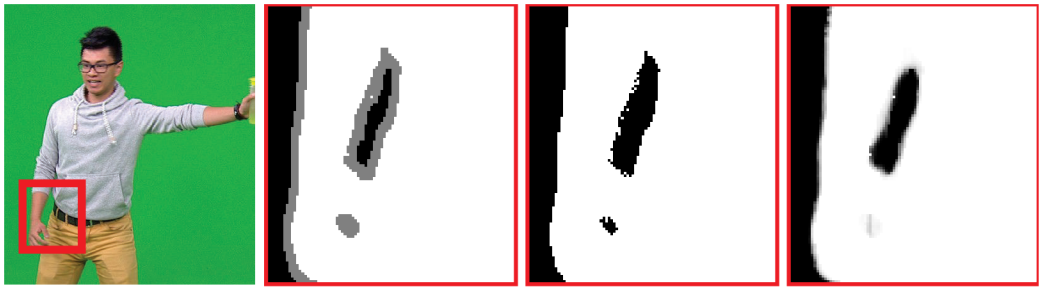
morfologických operací z algoritmu ViBe+ [9], které zmírní zašumění segmentační masky, jak dokazuje obrázek 3.7. Využití morfologických operací je následující:

$$\mathbf{S_m} = (\mathbf{S_s} \circ E_o) \bullet E_c, \tag{6.5}$$

kde $\circ$ symbolizuje morfologickou operaci otevření a $\bullet$ značí uzavření. Strukturní element E_o má tvar čtverce s délkou strany 4, E_c má také tvar čtverce o straně 3, $\mathbf{S_m}$ představuje segmentační masku vylepšenou morfologickými operacemi. Algorimus ViBe se segmentací stínů a s morfologickými operacemi bude nadále nazýván jako **ViBe#**.

6.3 Vyhlazení segmentační masky

Přestože je možné využít segmentační masku pro extrakci objektů popředí, hrany segmentovaných objektů jsou ostré, což způsobí méně přirozený vzhled při zpětném vložení objektů popředí. Rozdíl mezi hrubou segmentační maskou a alfa maskou je zobrazen na obrázku 6.7.



(a) Input frame (b) Trimap (c) ViBe (d) DeepMatting

Obrázek 6.7: Porovnání binární masky (c) a alfa masky (d). Červená oblast označuje část snímku, která byla přiblížena.

6.3.1 Automatické generování trimapy

Hlavním cílem je vyhladit hrany segmentovaných objektů, proto je část trimapy označující neznámou oblast generována z hran pomocí Sobelova operátoru. Následně jsou tyto hrany rozšířeny pomocí dilatace pouze do oblastí, které byly označeny jako pozadí. To umožňuje mírnou opravu segmentační masky, kde pixely byly špatně klasifikovány jako pozadí. Jak dokazuje obrázek 6.8 při umožnění rozšíření neznámé oblasti do oblastí označených jako popředí by neznámá oblast mohla pohltit oblasti popředí, čímž se ztrácí informace o barvě popředí.

6.3.2 Použitý *matting* algoritmus

Na základě evaluace popsané v kapitole 5.1 je využit v navrženém systému algoritmus Global Sampling Matting [16].

Při ohodnocení vhodnosti páru vzorků se počítá také vzdálenost nejbližšího vzorku, nalezení nejbližšího souseda je relativně časově náročný výpočet na GPU a podle experimentů uvedených v Zampirolli et al. [1] může trvat až 10 *ms*, což je omezující pro systém, který cílí na zpracování 25 snímků za sekundu. Z toho důvodu hlavní náplní této kapitoly je popis optimalizace algoritmu GSM pro GPU. Modifikovaný algoritmus GSM je dále nazýván **modifikovaný GSM (MGSM)**. Implementované modifikace lze rozdělit na dvě části:

(a) Dilatace neznámé oblasti bez omezení

(b) Dilatace neznámé oblasti pouze do oblastí označených jako pozadí

Obrázek 6.8: Při neomezené dilataci neznámé oblasti (a) může dojít ke ztrátě informace o popředí.

Využití modelu pozadí Matting algoritmy jsou postaveny tak, aby využívaly pouze trimapu a vstupní snímek. Nicméně při využití algoritmu postaveném na odečítání pozadí je k dispozici také model pozadí, který lze využít v matting algoritmu.

Při využití modelu pozadí $\mathbf{M_1}$ není potřeba odhadovat barvu pozadí pomocí vzorků – redukuje se prohledávaný dvojrozměrný **FB** prostor pouze na jednorozměrný **F** prostor. Při využití popsané modifikace se změní ohodnocovací funkce (4.4) a (4.2) následovně:

$$\xi_c(\mathbf{F}^i) = ||\mathbf{I} - (\hat{\alpha}\mathbf{F}^i) + (1 - \hat{\alpha})\mathbf{M_1})|| \tag{6.6}$$

$$\xi(\mathbf{F}^i) = \omega\xi_c(\mathbf{F}^i) + \xi_s(\mathbf{F}^i), \tag{6.7}$$

rovnice (4.1) pro odhadnutí alfa hodnoty se změní obdobně:

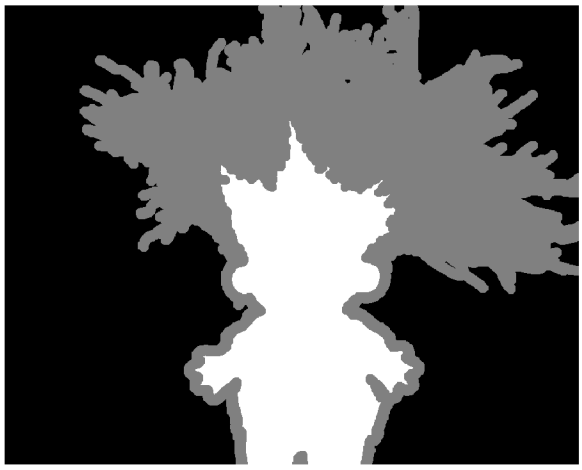
$$\hat{\alpha} = \frac{(\mathbf{I} - \mathbf{M_1})(\mathbf{F}^i - \mathbf{M_1})}{||\mathbf{F}^i - \mathbf{M_1}||^2}. \tag{6.8}$$

Při využití modelu pozadí $\mathbf{M_1}$ se zmenší počet stupňů volnosti ze šesti na tři, což způsobí více podmíněný výběr vzorek popředí. Využití modelu pozadí výrazně zlepší přesnost odhadu alfa masky, jak dokazuje evaluace popsaná v kapitole 5.2.

Eliminace výpočtu nejbližšího vzorku Přestože se po výše zmíněné modifikaci nepočítá vzdálenost nejbližšího vzorku pozadí, stále zůstává výpočet vzdálenosti nejbližšího vzorku popředí D_F. Hodnota D_F funguje pouze jako normalizace, aby cenová funkce nebyla závislá na absolutní vzdálenosti. Taková normalizace má význam u trimap, kde má neznámá oblast velké rozměry, jak ilustruje obrázek 6.9. Ovšem jak dokazuje obrázek 6.8, při použití automatického generování trimapy z hran objektů je neznámá oblast dostatečně tenká, aby se vzdálenost nejbližšího vzorku příliš neměnila, proto ji lze aproximovat konstantní hodnotou. Následující rovnice představuje aproximaci rovnice (4.3):

$$\xi_s(\mathbf{F}^i) = \frac{||x_{\mathbf{F}^i} - x_{\mathbf{I}}||}{10}, \tag{6.9}$$

kde hodnota 10 byla experimentálně zjištěna.



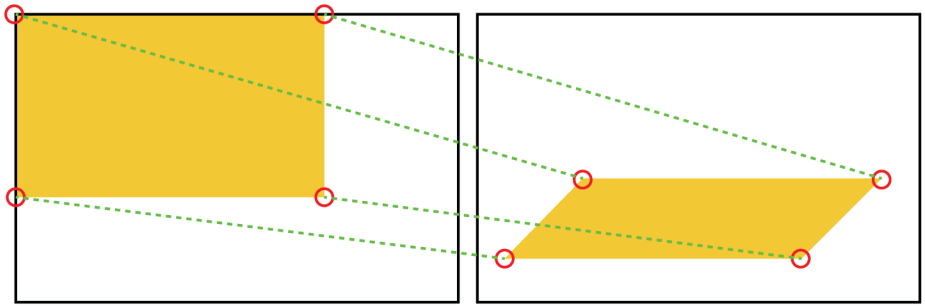
Obrázek 6.9: Ukázka trimapy s velkou neznámou oblastí. Zobrazená trimapa je součástí AlphaMatting datasetu, který je dostupný na http://alphamatting.com/datasets.php.

6.4 Vložení grafiky

Tato kapitola popisuje vložení 2D grafiky do vstupního snímku. Před vložením grafiky se vkládaná grafika $\mathbf{x}$ transformuje na základě uživatelova vstupu, transformaci grafiky lze popsat následující rovnicí:

$$\mathbf{x}' = \mathbf{H}\mathbf{x}, \tag{6.10}$$

kde $\mathbf{x}'$ značí transformovanou grafiku a $\mathbf{H}$ představuje transformační matici. Jak zobrazuje obrázek 6.10, při výpočtu matice jsou zadané čtyři body uživatelem, které představují rohové body obdélníkové obálky transformované grafiky, což je minimální počet bodů potřebný pro výpočet perspektivní transformační matice, jak uvádí Szeliski [30].

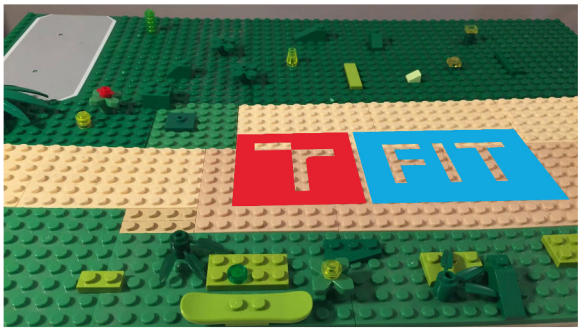


Obrázek 6.10: Pro výpočet transformační matice grafiky se využívají pouze čtyři rohové body grafiky.

6.4.1 Propagace textur

Jak dokazuje obrázek 6.11, při naivním vložení grafiky do scény vypadá vložená grafika uměle a je snadno rozpoznatelná.

Jak uvádí Farbman [11], pro kompozici virtuální grafiky do scény nelze využít algoritmy typu seamless cloning [24, 11, 34], kvůli absenci textury na krajích grafiky, která je podobná textuře cílové oblasti. Z toho důvodu jsem vytvořil algoritmus na propagaci textur



Obrázek 6.11: Pří naivním vložení grafiky do scény, vypadáná vložená grafika příliš uměle a nezapadá do scény.

a lokálních změn osvětlení, který pracuje se snímky v barevném prostoru LAB. Nevýhodou tohoto algoritmu je požadavek snímku pozadí. Daný algoritmus spočítá průměrnou hodnotu L kanálu ve snímku pozadí, tuto hodnotu následně odečte od L kanálu snímku pozadí, čímž vznikne rozdílová mapa, která popisuje texturu. Princip funkčnosti propagace textur je popsán algoritmem 4. Porovnání propagace textur s naivními metodami vložení grafiky je zobrazeno na obrázku 6.13.

Algoritmus 4: Algoritmus na propagaci textur.

$\mathbf{V} \leftarrow$ graphics
$\mathbf{M} \leftarrow$ modeled background

$\mathbf{V_L}, \mathbf{V_A}, \mathbf{V_B} \leftarrow LAB(\mathbf{V})$
$\mathbf{M_L} \leftarrow ComponentL(LAB(\mathbf{M}))$
$\mathbf{M_{Avg_L}} \leftarrow Avg(\mathbf{M_L})$

$\mathbf{Texture} \leftarrow \mathbf{M_L} - \mathbf{M_{Avg_L}}$
$\mathbf{V_L} \leftarrow Clamp(\mathbf{V_L} + \mathbf{Texture}, 0, 255)$
$\mathbf{V} \leftarrow (\mathbf{V_L}, \mathbf{V_A}, \mathbf{V_B})$

6.4.2 Kompozice

Po vložení grafiky do scény se přenesou objekty popředí nad vloženou grafiky pomocí následující rovnice:

$$\mathbf{I'} = \hat{\alpha}\alpha_V\mathbf{I} + (1 - \alpha_V)\mathbf{I}, \tag{6.11}$$

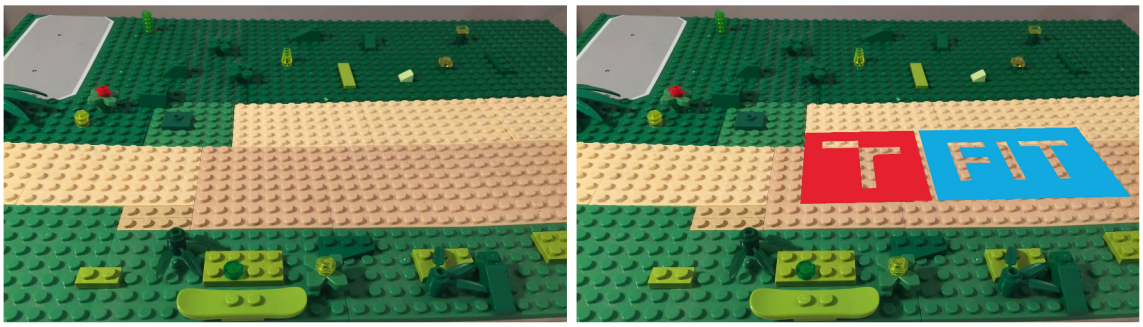
kde α_V představuje alfa masku vkládané grafiky, $\hat{\alpha}$ označuje alfa masku popředí a $\mathbf{I}$ označuje vstupní snímek. Následně jsou přidány stíny v barevném prostoru LAB, přidání stínů popisuje algoritmus 5. Jednotlivé kroky kompozice ilustruje obrázek 6.14.

Algoritmus 5: Algoritmus na přidání stínů. $\mathbf{O}$ představuje masku s intenzitou stínů.

$\mathbf{I'_L}, \mathbf{I'_A}, \mathbf{I'_B} \leftarrow LAB(\mathbf{I'})$
$\mathbf{I'_L} \leftarrow \mathbf{I'_L} - \alpha_V\mathbf{O}$
$\mathbf{I'} \leftarrow (\mathbf{I'_L}, \mathbf{I'_A}, \mathbf{I'_B})$

(a) (b)

Obrázek 6.12: *Seamless cloning* algoritmy fungují optimálně v situacích, kdy vkládaný objekct má okolo okrajů podobnou texturu jako okraje cílové oblasti (a). Virtuální grafika nemá skoro žádnou texturu, tudíž způsobí artefakt, který vypadá jako rozmazání cílové oblasti (b).



(a) Scéna (b) Hrubé vložení grafiky

(c) Vložení grafiky s průhledností (d) Vložení grafiky s propagací textur

Obrázek 6.13: Při hrubém vložení grafiky (b) do scény (a) nevypadá grafika příliš přirozeně. Při snížení průhlednosti grafiky (c) vypadá daná grafika více jako promítaná na podklad, nicméně na úkor sytosti barev. Použitím algoritmu na propagaci textur (d) vypadá vložená grafika nejrealističtěji.

(a) Hrubé vložení grafiky

(b) Propagace textur

(c) Opětovné vložení popředí

(d) Přidání stínů

Obrázek 6.14: Vyobrazení jednotlivých kroků kompozice. Po přidání grafiky (a) se aplikuje propagace textur (b). Následně se přenese popředí (c) nad vloženou grafiku. Na závěr se zrekonstruují stíny (d).

Kapitola 7

Implementace a zhodnocení systému

Prototyp systému byl implementován v jazyce Python 3, kvůli snadnější implementaci experimentálních heuristik v rané fázi návrhu systému. Implementoval jsem téměř celý systém včetně algoritmu ViBe [3] pomocí knihovny NumPy, která je akcelerována pomocí SIMD instrukcí. Algoritmy Global Sampling Matting [16] a GuidedFilter [17] byly převzaty z veřejně dostupného repozitáře[1]. Implementoval jsem také téměř celou akcelerovanou verzi systému na GPU, využil jsem knihovnu OpenCV[2] a implementaci morfologických operací akcelerovaných na GPU[3].

7.1 Paralelizace na GPU

Pro akceleraci navrženého systému je použito CUDA SDK. Hlavní priority akcelerace na GPU jsou v tomto pořadí:

1. Omezení alokace paměti při zpracovávání snímku.
2. Omezení počtu přenosů paměti mezi CPU a GPU.
3. Dosáhnutí koalescence při čtení/zápisu do globální paměti GPU.
4. Omezení přístupů do globální paměti.

Omezení alokace při zpracovávání snímku je docíleno alokací sdílené mezipaměti při spuštění systému. Všechna potřebná paměť pro ukládání vstupního snímku a výsledného snímku je také alokována na začátku při spuštění systému.

Druhou prioritu nebylo nutné řešit, vzhledem k tomu že se pouze přenese vstupní snímek na GPU a výsledný snímek na CPU, pak již není nutné přenášet jiná data mezi GPU a CPU.

7.1.1 *ViBe*

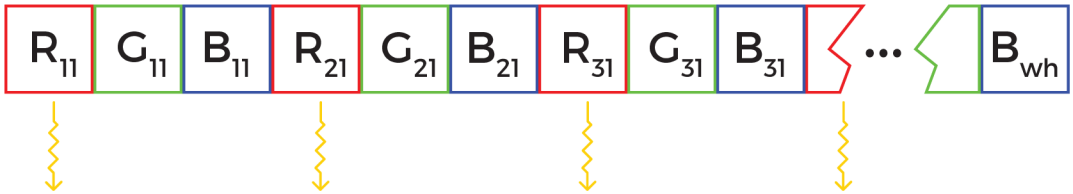
Tato kapitola se zabývá akcelerací algoritmu ViBe [3] na GPU. Hlavní výhodou algoritmu ViBe je nezávislost zpracování pixelu na ostatních pixelech – není potřeba znát hodnoty

[1] Dostupné na https://github.com/atilimcetin/global-matting
[2] Více na https://opencv.org/
[3] Dostupné na https://github.com/mompes/CUDA-dilation-and-erosion-filters

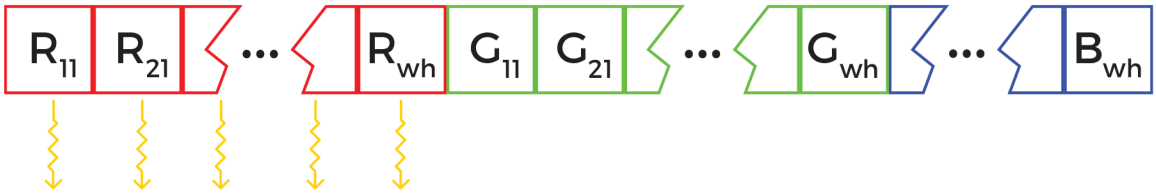
okolních pixelů, tudíž každé vlákno zpracovává jeden přidělený pixel. Segmentační část včetně první části aktualizace modelu pozadí jsou implementované v jednom kernelu. Jelikož při propagaci sousedů je potřeba globální synchronizace, je propagace sousedů implementována v druhém odděleném kernelu.

Pro další akceleraci daného algoritmu jsem optimalizoval přístup do paměti, aby bylo dosaženo koalescence. Je-li snímek uložen v RGBA formátu, lze snadno dosáhnout koalescence při využití datové struktury uchar4, která načte všechny čtyři bajty naráz, kde dané bajty představují R, B, G, α kanály. Přestože v navrženém systému není alfa hodnota potřebná u vstupního snímku, je přenášena na GPU, jelikož přeskládání RGBA formátu na upravený tříbajtový RGB formát trvá déle než načtení nadbytečného bajtu.

Ovšem u modelu pozadí se vyplatí přeskládat RGB formát, jelikož daný model se za běhu systému nijak nepřepisuje snímkem uloženým na CPU, což umožňuje použití libovolného způsobu uložení modelu v paměti. Obrázek 7.1 ilustruje způsob uložení RGB formátu v paměti pro dosáhnutí koalescence.



(a) Při zachování RGB formátu, kde jsou kanály pixelu uloženy za sebou, není možné dosáhnout koalescence.



(b) Při dekompozici kanálů je dosáhnuto koalescence.

Obrázek 7.1: Porovnání rozdílných způsobů uložení RGB obrázku v paměti. Při dekompozici kanálů (b) oproti (a) je dosáhnuto koalescence – kontinuální přístup do paměti.

7.1.2 *Global Sampling Matting*

Stejně jako u algoritmu ViBe [3] hlavní výhodou algoritmu GSM [16] je nízká nezávislost pixelů na okolních pixelech. Implementace algoritmu GSM je rozdělena do čtyř následujících kernelů:

Inicializace globální množiny Tento kernel vytváří pole pro uchování vzorků popředí, při přidávání prvků je použit atomický čítač. Pro zapouzdření informací o vzorku popředí, je využita struktura MattingSample, která obsahuje barvu a pozici vzorku popředí, jak je znázorněno v ukázce 7.1, struktura je zarovnaná na 8 bajtů.

Inicializace neznámých pixelů funguje obdobně jako inicializace globální množiny. Neznámé pixely jsou uloženy v poli pomocí atomického čítače. Data pro neznámý pixel jsou uložena v datové struktuře UnknownPixel, která obsahuje barvu pozadí, barvu aktuálního snímku na dané pozici, nejnižší dosažené ohodnocení, hodnotu alfa $\hat{\alpha}$ a pozici daného neznámého pixelu, struktura je ukázána v úryvku 7.2. Ovšem index

```
struct __align__(8) MattingSample
{
    uint8_t R, G, B;
    uint16_t x, y;
    uint8_t _;
};
```

Výpis 7.1: Struktura MattingSample obsahuje informace o vzorku popředí $\mathbf{F}_i$ – barvu a pozici vzorku.

```
struct __align__(16) UnknownPixel
{
    uint8_t bgR, bgG, bgB;
    uint8_t frameR, frameG, frameB;

    float bestCost;
    uint8_t currentAlpha;

    uint16_t x, y;
    uint8_t _;
};
```

Výpis 7.2: Struktury UnknownPixel zapozdřuje informace o neznámé pixelu. Obsahuje barvu pozadí, barvu snímku, dosavadní nejnižší cenu, aktuální hodnotu $\hat{\alpha}$ a pozici pixelu. Pozice pixelu je uložena v datové struktuře, jelikož instance dané struktury jsou uloženy v poli za sebou, tudíž index nevyjadřuje pozici v obraze.

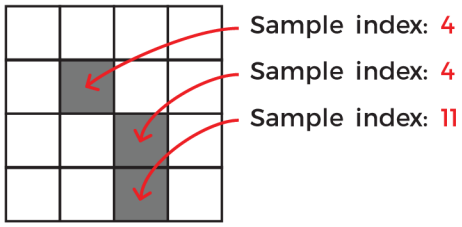
vzorku s nejnižší cenou pro daný neznámý pixel je uložen v poli, které reflektuje rozměr vstupního snímku.

Je možné sloučit tento kernel s kernelem na inicializaci globální množiny, nicméně by došlo k větvění kernelu – což vede k nechtěnému snížení obsazenosti.

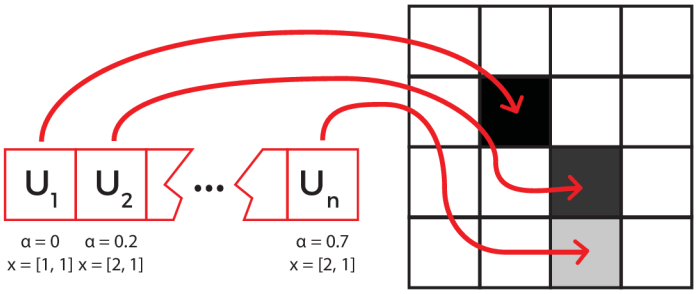
Iterace algoritmu SamplePatch Jádro algoritmu GSM [16] představuje prohledávání **FB** prostoru v tomto případě **F** prostoru. Jelikož propagace závisí na aktuálně nejlepších vzorcích sousedních pixelů, je potřeba globální synchronizace po každé iteraci dvojice operací – propagace a náhodné procházky. Implementace, kde jedno zavolání kernelu představuje jednu iteraci, způsobí globální synchronizaci vláken po každém zavolání kernelu.

Při nalezení optimálnějšího vzorku, je index daného vzorku uložen do pole, které má stejné rozměry jako vstupní snímek, jak zobrazuje obrázek 7.2.

Renderování alfa masky je nutné, protože $\hat{\alpha}$ hodnoty jsou uložené v poli, kde index nereprezentuje pozici v obraze. Jedno vlákno zpracovává jeden neznámý pixel, kde načte strukturu typu UnknownPixel a zapíše hodnotu $\hat{\alpha}$ na reálnou pozici v alfa masce. Funkci vyrenderování ilustruje obrázek 7.3.



Obrázek 7.2: Indexy nejoptimálnějších vzorků jsou uloženy v poli, který má stejné rozměry jako vstupní snímek. Šedá políčka představují neznámé pixely – ne všechna políčka jsou tedy použita – tento způsob uložení je ovšem vhodný pro vyrenderování alfa masky.



Obrázek 7.3: Data neznámých pixelů jsou uloženy v poli, kde index nereprezentuje pozici v obraze. Z toho důvodu byl vytvořen kernel na vykreslení $\hat{\alpha}$ hodnot neznámých pixelů do alfa masky.

7.1.3 Konvoluce

Konvoluce je použita v podobě Sobelových filtrů a průměrovacích filtrů v navrženém systému při počítání gradientů a v algoritmu Guided Filter [17]. V naivní implementaci filtru o velikosti PxQ je nutné provést $WHPQ$ načtení hodnot, kde W, H značí rozměr snímku a P, Q rozměr filtru. Dvojrozměrné filtry lze rozdělit na dva jednorozměrné filtry o velikost $Px1$ a $1xQ$. Při dekompozici filtru je potřeba provést pouze $WH(P + Q)$ načtení hodnot narozdíl od naivní verze, kde je potřeba $WHPQ$ načtení hodnot. Dvojrozměrný filtr h lze dekomponovat na dva jednorozměrné filtry, pouze pokud platí $rank(h) = 1$, tato podmínka platí jak pro Sobelův operátor tak i průměrovací filtr.

7.2 Zhodnocení systému

V této kapitole jsou zhodnoceny výsledky a je zobrazeno porovnání rychlosti navrženého systému. Měření rychlosti systému bylo provedeno na procesoru Intel i7-7500U a na grafické kartě Nvidia GTX 1070 se snímky o rozlišení 1280×720, jak uvádí tabulka 7.1, v akcelerované verzi systému na GPU došlo k enormnímu zrychlení vůči CPU verzi systému. Výsledný systém tedy zpracovává snímky **v reálném čase**.

Na základě experimentů s cílovým publikem, jsem zjistil, že popředí špatně klasifikované jako pozadí je více rozeznatelné než pozadí špatně klasifikované jako popředí. Na základě těchto zjištění byly upraveny prahy jednotlivých algoritmů, nicméně realističnost připadla testovaným subjektům lepší oproti naivním metodám.

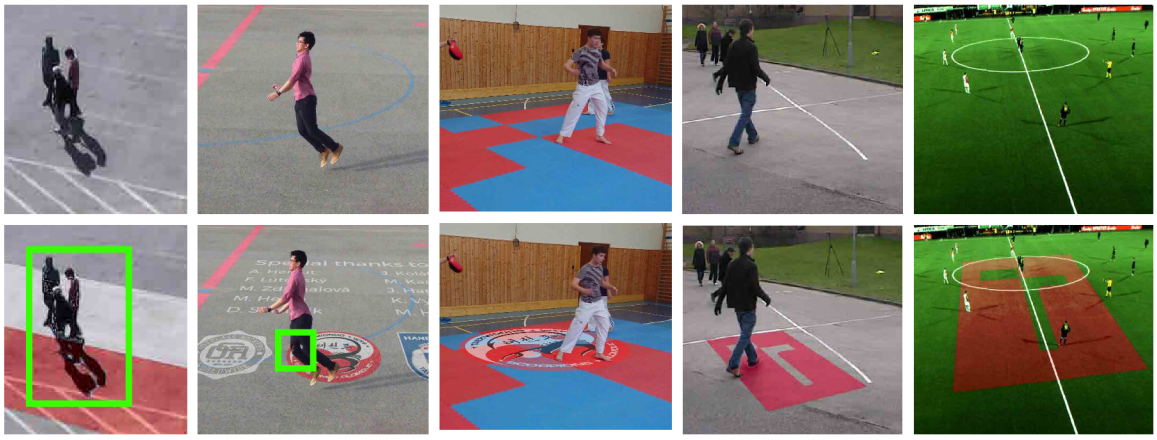
Tabulka 7.1: Porovnání CPU implementace systému s GPU implementací. ViBe# je na CPU implementován v jazyce Python 3 s použitím NumPy knihovny. CPU verze algoritmů MGSM a GuidedFilter jsou implementovány v programovacím jazyce C++. Celý systém je akcelerován na GPU pomocí CUDA SDK. Systém byl měřen při zpracovávání datasetu LITOVEL_01. *Testováno na grafické kartě Nvidia GTX 1070 a na procesoru Intel i7-7500U processor se snímkem o rozlišení* 1280×720.

Algoritmus	CPU [ms]	GPU [ms]	Poměr zrychlení
ViBe#	1520	0.6	$\tilde{2}500$
MGSM	2700	1.4	$\tilde{1}900$
GuidedFilter [17]	650	2.7	$\tilde{2}40$
Celý systém	6250	13	$\tilde{4}80$

Propagace textur částečně selhává, když je svítivost vkládané grafiky blízko hraničním hodnotám oboru hodnot, jelikož dochází k ořezání hodnot na interval $\langle 0, 255 \rangle$, to se projeví slabou viditelností textury pozadí.

Nejslabší článek systému představuje segmentace stínů, která v některých případech označí i část objektu popředí jako stín, tento nedostatek je částečně kompenzován pomocí matting algoritmu, přesto se objevují artefakty, které vypadají jako kus pozadí vložený nad objekt popředí. Přestože navržený algoritmus na segmentaci stínů selhává také u tmavých stínů, není tento příliš nedostatek rozponatelný, jelikož daný stín je natolik tmavý, že se ztrácí informace textury pozadí, jak lze vidět na obrázku 7.4.

Implementovaný systém snadno dosahuje zpracování snímků v reálném čase, jak uvádí tabulka 7.1, je dosaženo zpracování zhruba 70 snímků za sekundu.



(a) VIRAT (b) (c) (d) PETS09 (e) ALFHEIM

(f) (g) (h) JNZP (i) LITOVEL_01

Obrázek 7.4: Otestování systému na různých datasetech. Tmavé stíny (a) jsou chybně klasifikovány jako popředí, nicméně tato chyba není příliš viditelná. Na obrázku (b) špatně klasifikované popředí jako stín způsobý artefakt, který vypadá část pozadí.

Kapitola 8

Závěr

Cílem mé práce bylo navrhnout a implementovat systém na vkládání 2D grafiky do videa v duchu rozšířené grafiky. Tento cíl byl zdárně splněn, jak dokazuje obrázek 7.4, pro zvýšení využitelnosti daného systému jsem urychlil použité algoritmy na GPU, což vedlo k celkovému urychlení systému, který je schopen zpracovávat zhruba 70 snímků o rozlišení 1280×720 pixelů za sekundu.

Během realizace mé práce jsem prostudoval různé metody pro segmentací popředí, systémy řešící *matting* problém a metody klonování. Na základě evaluací popsaných v kapitole 7.4 jsem vybral nejlepší kombinaci segmentačního a matting algoritmu.

Pro docílení přesnější masky popředí jsem navrhnul modifikace použitých algoritmů ViBe [3] a Global Sampling Matting [16], které převážně zvýšily přesnost výsledné alfa masky popředí a nepřímo urychlily akcelerovanou verzi algoritmu Global Sampling Matting na GPU. Jelikož algoritmus ViBe nerozlišuje stíny a objekty popředí, vytvořil jsem algoritmus na segmentaci stínů odstraňující tento nedostatek.

Dále pro odstranění umělého vzhledu vložené grafiky jsem implementoval algoritmus na propagaci textur, který je popsán v kapitole 6.4.1.

Po vytvoření článku o mé práci jsem byl přijat do studenstkých konferencí CESCG a Excel@FIT, kde jsem v rámci podpůrných prezentačních materilů mé práce vytvořil také plakát, prezentaci a demonstrační video[1]. Následně byla má práce přijata i k ústní prezentaci do obou konferencí. Na studenstké konferenci ExcelFIT byla má práce získala *ocenění odborným panelem*, *ocenění partnery* a *cenu Jiřího Kunovského*, na mezinárodní konferenci CESCG získala následující ocenění *Best Video Award* a 2^{nd} *Best Presentation Award*.

V rámci rozšíření mé práce by bylo možné experimentovat s příznaky LBSP [4] pro popis textury v rámci segmentace stínů. Pro odstranění omezení nepohybující se kamery by bylo možné modelovat pozadí jako panorama, kde nejsložitější část představuje lokalizace daného snímku v daném rozšířeném modelu pozadí.

[1] Dostupné na https://www.youtube.com/watch?v=zNo-B2FJkUY

Literatura

[1] d. A. Zampirolli, F.; Filipe, L.: A Fast CUDA-Based Implementation for the Euclidean Distance Transform. In *2017 International Conference on High Performance Computing Simulation (HPCS)*, July 2017, s. 815–818, doi:10.1109/HPCS.2017.123.

[2] Barnes, C.; Shechtman, E.; Finkelstein, A.; aj.: PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, ročník 28, č. 3, Srpen 2009.

[3] Barnich, O.; Droogenbroeck, M. V.: ViBe: A Universal Background Subtraction Algorithm for Video Sequences. *IEEE Transactions on Image Processing*, ročník 20, č. 6, June 2011: s. 1709–1724, ISSN 1057-7149, doi:10.1109/TIP.2010.2101613.

[4] Bilodeau, G.; Jodoin, J.; Saunier, N.: Change Detection in Feature Space Using Local Binary Similarity Patterns. In *2013 International Conference on Computer and Robot Vision*, May 2013, s. 106–112, doi:10.1109/CRV.2013.29.

[5] Cavallaro, R.; Hybinette, M.; White, M.; aj.: Augmenting Live Broadcast Sports with 3D Tracking Information. *IEEE MultiMedia*, ročník 18, č. 4, April 2011: s. 38–47, ISSN 1070-986X, doi:10.1109/MMUL.2011.61.

[6] Chen, L.; Zhu, Y.; Papandreou, G.; aj.: Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. *CoRR*, ročník abs/1802.02611, 2018, 1802.02611.
URL http://arxiv.org/abs/1802.02611

[7] Cucchiara, R.; Grana, C.; Piccardi, M.; aj.: Detecting moving objects, ghosts, and shadows in video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník 25, č. 10, Oct 2003: s. 1337–1342, ISSN 0162-8828, doi:10.1109/TPAMI.2003.1233909.

[8] Divya, K. A.; Roshna, K. I.; Mathai, S.: Shadow detection and removal by object-wise segmentation. In *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, Dec 2015, s. 1–4, doi:10.1109/ICCIC.2015.7435784.

[9] Droogenbroeck, M. V.; Paquot, O.: Background subtraction: Experiments and improvements for ViBe. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, June 2012, ISSN 2160-7508, s. 32–37, doi:10.1109/CVPRW.2012.6238924.

[10] Erofeev, M.; Gitman, Y.; Vatolin, D.; aj.: Perceptually Motivated Benchmark for Video Matting. In *Proceedings of the British Machine Vision Conference (BMVC)*, BMVA Press, September 2015, ISBN 1-901725-53-7, s. 99.1–99.12, doi:10.5244/C.29.99.
URL https://dx.doi.org/10.5244/C.29.99

[11] Farbman, Z.; Hoffer, G.; Lipman, Y.; aj.: Coordinates for Instant Image Cloning. *ACM Trans. Graph.*, ročník 28, č. 3, Červenec 2009: s. 67:1–67:9, ISSN 0730-0301, doi:10.1145/1531326.1531373.
URL http://doi.acm.org/10.1145/1531326.1531373

[12] Gastal, E. S. L.; Oliveira, M. M.: Shared Sampling for Real-Time Alpha Matting. *Computer Graphics Forum*, ročník 29, č. 2, May 2010: s. 575–584, proceedings of Eurographics.

[13] Glorot, X.; Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, *Proceedings of Machine Learning Research*, ročník 9, editace Y. W. Teh; M. Titterington, Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, s. 249–256.
URL http://proceedings.mlr.press/v9/glorot10a.html

[14] Güler, R. A.; Trigeorgis, G.; Antonakos, E.; aj.: DenseReg: Fully Convolutional Dense Shape Regression In-the-Wild. *CoRR*, ročník abs/1612.01202, 2016, 1612.01202.
URL http://arxiv.org/abs/1612.01202

[15] He, K.; Gkioxari, G.; Dollár, P.; aj.: Mask R-CNN. In *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, ISSN 2380-7504, s. 2980–2988, doi:10.1109/ICCV.2017.322.

[16] He, K.; Rhemann, C.; Rother, C.; aj.: A global sampling method for alpha matting. In *CVPR 2011*, June 2011, ISSN 1063-6919, s. 2049–2056, doi:10.1109/CVPR.2011.5995495.

[17] He, K.; Sun, J.; Tang, X.: Guided Image Filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník 35, č. 6, June 2013: s. 1397–1409, ISSN 0162-8828, doi:10.1109/TPAMI.2012.213.

[18] He, K.; Zhang, X.; Ren, S.; aj.: Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, ISSN 1063-6919, s. 770–778, doi:10.1109/CVPR.2016.90.

[19] Kim, K.; Chalidabhongse, T. H.; Harwood, D.; aj.: Real-time foreground-background segmentation using codebook model. *Real-Time Imaging*, ročník 11, 2005: s. 172–185.

[20] Krähenbühl, P.; Koltun, V.: Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. *CoRR*, ročník abs/1210.5644, 2012, 1210.5644.
URL http://arxiv.org/abs/1210.5644

[21] Levin, A.; Lischinski, D.; Weiss, Y.: A Closed Form Solution to Natural Image Matting. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, ročník 1, June 2006, ISSN 1063-6919, s. 61–68, doi:10.1109/CVPR.2006.18.

[22] Lim, L. A.; Keles, H. Y.: Foreground segmentation using convolutional neural networks for multiscale feature encoding. *Pattern Recognition Letters*, ročník 112, 2018: s. 256 – 262, ISSN 0167-8655, doi:https://doi.org/10.1016/j.patrec.2018.08.002.
URL http://www.sciencedirect.com/science/article/pii/S0167865518303702

[23] Monji-Azad, S.; Kasaei, S.; Eftekhari-Moghadam, A.: An efficient augmented reality method for sports scene visualization from single moving camera. In *2014 22nd Iranian Conference on Electrical Engineering (ICEE)*, May 2014, ISSN 2164-7054, s. 1064–1069, doi:10.1109/IranianCEE.2014.6999693.

[24] Pérez, P.; Gangnet, M.; Blake, A.: Poisson Image Editing. *ACM Trans. Graph.*, ročník 22, č. 3, Červenec 2003: s. 313–318, ISSN 0730-0301, doi:10.1145/882262.882269.
URL http://doi.acm.org/10.1145/882262.882269

[25] Riza Alp Güler, I. K., Natalia Neverova: DensePose: Dense Human Pose Estimation In The Wild. *arXiv*, 2018.

[26] Russakovsky, O.; Deng, J.; Su, H.; aj.: ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, ročník 115, č. 3, 2015: s. 211–252, doi:10.1007/s11263-015-0816-y.

[27] Simonyan, K.; Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, ročník abs/1409.1556, 2014, 1409.1556.
URL http://arxiv.org/abs/1409.1556

[28] St-Charles, P.; Bilodeau, G.; Bergevin, R.: SuBSENSE: A Universal Change Detection Method With Local Adaptive Sensitivity. *IEEE Transactions on Image Processing*, ročník 24, č. 1, Jan 2015: s. 359–373, ISSN 1057-7149, doi:10.1109/TIP.2014.2378053.

[29] Stauffer, C.; Grimson, W. E. L.: Adaptive background mixture models for real-time tracking. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, ročník 2, June 1999, ISSN 1063-6919, s. 246–252 Vol. 2, doi:10.1109/CVPR.1999.784637.

[30] Szeliski, R.: *Computer Vision: Algorithms and Applications*. Berlin, Heidelberg: Springer-Verlag, první vydání, 2010, ISBN 1848829345, 9781848829343.

[31] Venkatraman, D.; Makur, A.: Shadow-less segmentation of moving humans from surveillance video sequences. In *2008 10th International Conference on Control, Automation, Robotics and Vision*, Dec 2008, s. 1317–1322, doi:10.1109/ICARCV.2008.4795712.

[32] Wang, B.; Dudek, P.: AMBER: Adapting multi-resolution background extractor. In *2013 IEEE International Conference on Image Processing*, Sep. 2013, ISSN 1522-4880, s. 3417–3421, doi:10.1109/ICIP.2013.6738705.

[33] Xu, N.; Price, B.; Cohen, S.; aj.: Deep Image Matting. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, ISSN 1063-6919, s. 311–320, doi:10.1109/CVPR.2017.41.

[34] Yang, W.; Zheng, J.; Cai, J.; aj.: Natural and Seamless Image Composition With Color Control. *IEEE Transactions on Image Processing*, ročník 18, č. 11, Nov 2009: s. 2584–2592, ISSN 1057-7149, doi:10.1109/TIP.2009.2027365.

[35] Zheng, J.; Wang, Y.; L. Nihan, N.; aj.: Extracting Roadway Background Image: Mode-Based Approach. *Transportation Research Record: Journal of the Transportation Research Board*, ročník 1944, 01 2006: s. 82–88, doi:10.1177/0361198106194400111.