

UNIVERZITA KOMENSKÉHO V BRATISLAVE  
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

SEGMENTÁCIA OBJEKTOV VO VIDEU,  
ROZOSTRENÝCH RÝCHLYM POHYBOM  
DIPLOMOVÁ PRÁCA

2019  
Bc. RÓBERT SARVAŠ

UNIVERZITA KOMENSKÉHO V BRATISLAVE  
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

SEGMENTÁCIA OBJEKTOV VO VIDEU,  
ROZOSTRENÝCH RÝCHLYM POHYBOM

DIPLOMOVÁ PRÁCA

Študijný program: Aplikovaná Informatika  
Študijný odbor: 9.2.9 - aplikovaná informatika  
Školiace pracovisko: FMFI.KAI - Katedra aplikovanej informatiky  
Školiteľ: RNDr. Zuzana Černeková, PhD.

Bratislava, 2019  
Bc. Róbert Sarvaš



Univerzita Komenského v Bratislave  
Fakulta matematiky, fyziky a informatiky

## ZADANIE ZÁVEREČNEJ PRÁCE

**Meno a priezvisko študenta:** Bc. Róbert Sarvaš  
**Študijný program:** aplikovaná informatika (Jednoodborové štúdium,  
magisterský II. st., denná forma)  
**Študijný odbor:** aplikovaná informatika  
**Typ záverečnej práce:** diplomová  
**Jazyk záverečnej práce:** slovenský  
**Sekundárny jazyk:** anglický

**Názov:** Segmentácia objektov vo videu, rozostrených rýchlym pohybom  
*Segmenting motion blurred objects in video*

**Anotácia:** Naštudovať problematiku segmentovania objektov. Oboznámiť sa s metódami a aplikáciami postprocessingu videa. Analyzovať existujúce riešenia publikované v dostupnej odbornej literatúre. Navrhnuť metódu, ktorá umožní vysegmentovanie objektu s rozostrenými hranami spôsobené rýchlym pohybom. Vyhodnotiť dosiahnuté výsledky.

**Ciel:** Naštudovať problematiku segmentovania objektov. Oboznámiť sa s metódami a aplikáciami postprocessingu videa. Analyzovať existujúce riešenia publikované v dostupnej odbornej literatúre. Navrhnuť metódu, ktorá umožní vysegmentovanie objektu s rozostrenými hranami spôsobené rýchlym pohybom. Vyhodnotiť dosiahnuté výsledky.

**Vedúci:** RNDr. Zuzana Černeková, PhD.  
**Katedra:** FMFI.KAI - Katedra aplikovanej informatiky  
**Vedúci katedry:** prof. Ing. Igor Farkaš, Dr.

**Dátum zadania:** 13.10.2017

**Dátum schválenia:** 13.10.2017

prof. RNDr. Roman Ďuríkovič, PhD.  
garant študijného programu

.....  
študent

.....  
vedúci práce

**Poděkovanie:** Chcel by som poděkovat mojej školiteľke RNDr. Zuzane Černeko-vej, PhD., za jej odborné vedenie a za ochotu poradiť a pomôcť s prácou. Ďalej děkujem najmä môjmu otcovi, ktorý mi bol vždy pri štúdiu veľkou oporou a spolu s matkou ma podporovali, keď som to potreboval. V neposlednom rade děkujem priateľke za ideálne podmienky vytvorené pre štúdium tejto témy.

## Abstrakt

Naštudovať problematiku segmentovania objektov. Oboznámiť sa s metódami a aplikáciami postprocessingu videa. Analyzovať existujúce riešenia publikované v dostupnej odbornej literatúre. Navrhnúť metódu, ktorá umožní vysegmentovanie objektu s rozostenými hranami spôsobené rýchlym pohybom. Vyhodnotiť dosiahnuté výsledky.

**Kľúčové slová:** video, sledovanie objektu, rozostený objekt

## Abstract

To study tracking and segmenting of objects. Learn about actual methods and applications of post-processing of videos. Analyze actual results and study state of the art of the task. To implement a method for detecting a object with motion blur with aim to precision. Show the results.

**Keywords:** video, motion blur, deblurring, object segmentation

# Obsah

<b>Úvod</b>	<b>1</b>
<b>1 Analýza súčasných riešení</b>	<b>3</b>
1.1 Segmentácia objektu z obrázku . . . . .	4
1.1.1 Technika Magic Wand . . . . .	4
1.1.2 Intelligent Scissors . . . . .	5
1.1.3 Bayes Matte . . . . .	5
1.1.4 Knockout 2 . . . . .	5
1.1.5 Graphcut . . . . .	5
1.1.6 GrabCut . . . . .	6
1.2 Segmentácia objektu z videa . . . . .	6
1.2.1 Video SnapCut: Robustná segmentácia objektu z videa používajúca lokálne klasifikátory. . . . .	7
1.2.2 OSVOS . . . . .	9
1.3 Detekcia rozmazania objektu pohybom . . . . .	10
1.3.1 Metódy používajúce neurónové siete . . . . .	11
1.3.2 Ostatné metódy . . . . .	14
1.3.3 Invertible Motion Blur in Video . . . . .	15
<b>2 Metodika</b>	<b>18</b>
2.1 Segmentácia objektu z obrázku . . . . .	18
2.1.1 Grabcut . . . . .	19
2.2 Segmentácia objektu z videa . . . . .	23
2.2.1 OSVOS: One-Shot Video Object Segmentation . . . . .	23
2.3 Rozmazanie pohybom . . . . .	26
2.3.1 Charakteristika efektu rozmazania pohybom . . . . .	26
2.3.2 Alpha Matting . . . . .	30
2.3.3 KNN Matting . . . . .	31
<b>3 Implementácia</b>	<b>35</b>
3.1 Softvérové požiadavky . . . . .	35

3.2	Dataset . . . . .	36
3.3	Grafické rozhranie a funkcia lita . . . . .	37
3.4	Grabcut metóda . . . . .	38
3.5	Integrácia OSVOS siete . . . . .	39
3.5.1	priečinkový systém OSVOS tehchnológie . . . . .	40
3.6	KNN Matting . . . . .	40
<b>4</b>	<b>Výsledky</b>	<b>41</b>
4.1	Vyhodnotenie . . . . .	41
4.1.1	Zvýšenie presnosti segmentácie objektu . . . . .	41
4.1.2	Limitácie aplikácie . . . . .	43
4.2	Testovanie . . . . .	45
	<b>Záver</b>	<b>50</b>

# Zoznam obrázkov

1.1	Graf podproblemov . . . . .	4
1.2	Segmentačné techniky . . . . .	4
1.5	Štruktúra OSVOS siete . . . . .	10
1.6	Príznaky z článku . . . . .	12
1.7	Príznaky z článku . . . . .	13
1.8	Optical flow na rozmazaných obrázkoch . . . . .	14
1.9	Porovnanie výsledkov zaostrenia obrázkov . . . . .	14
1.10	Jadro rozmazania pohybu . . . . .	15
1.12	Prehľad algoritmu automatického vyplnenia PSF funkcie . . . . .	17
2.1	Degradácie obrázku a tažké objekty na segmentovanie . . . . .	19
2.2	Grabcut . . . . .	20
2.3	Príklad OSVOS . . . . .	23
2.4	Štruktúra OSVOS metódy . . . . .	24
2.5	Dvojprúdová FCNN architektúra . . . . .	25
2.6	Tabuľka presnosťí OSVOS techniky . . . . .	25
2.7	Rozčlenenie obrázku na superpixely . . . . .	27
2.8	Výsledky SVM, KNN a Lineárnych klasifikátorov . . . . .	28
2.9	OD Rozmazania pohybom k alpha Matting . . . . .	30
2.10	KNN vs Nonlocal matting . . . . .	32
2.11	Matica pribuznosťí . . . . .	32
2.12	Funkcia jadra . . . . .	33
3.1	GUI rozhranie aplikácie . . . . .	37
3.2	Použitie Grabcut algoritmu v aplikácii . . . . .	39
4.1	Sekvencia snímok bežiaceho psa . . . . .	42
4.2	Príklady vylepšenia segmentácie objektov KNN Matting Metódou . . . . .	42
4.3	Gymnastka - porovnanie po OSVOS a OSVOS+KNN matting metódach	43
4.4	Segmentácia hráča tenisu . . . . .	44
4.5	Demonštrácia limitácie . . . . .	44

4.6 Segmentácia hráča tenisu . . . . .	46
4.7 Segmentácia hráča tenisu . . . . .	47
4.8 Segmentácia hráča tenisu . . . . .	47
4.9 Segmentácia hráča tenisu . . . . .	48
4.10 Tabuľka výsledkov metód . . . . .	48

# Zoznam tabuliek

# Úvod

Nástup umelej inteligencie a autonómnych robotov je v posledných rokoch enormný a zdá sa, že v blízkej budúcnosti výrazne ovplyvní takmer všetky odvetvia priemyslu a nahradí množstvo elektronických zariadení v našich domácnostiah. Autonómne autá a inteligentné domy sú len špičkou ľadovca zariadení, ktoré sú dnes už používanou širokou verejnosťou a aj, preto je dnes počítačové videnie jedným z najdynamickejšie rozvíjajúcich sa odvetví v informačných technológiách.

Technológie záznamu obrazu sú dnes už na úrovni, na ktorej už dávno kvalitou predbehli náš ľudský zrak. Jedým z cieľov počítačového videnia je naučiť počítače rozpoznávať obraz a vyhodnocovať ho tak, ako to dokážu ľudia. V tomto smere sú však počítače ešte výrazne vzdialené našim schopnostiam rozpoznávania objektov.

Záznam obrazu pozostáva z dvoch krokov - snímania a digitalizácie obrazu. Pri zázname obrazu, ale vznikajú chyby snímania obrazu - šum, skreslenie objektívom a neostrosť. Neostrosť alebo rozmazanie obrazu sa delí do dvoch kategórii podľa spôsobu, ako vzniká neostrosť zlým zaostrením šošoviek (*ang. Out of focus blur*) a neostrosť spôsobená pohybom kamery alebo objektu v scéne rýchlosťou vyššou, ako je rýchlosť snímania obrazu. Tento typ neostrosti poznáme ako rozmazanie pohybom (*ang. Motion blur*).

Práve problém rozmazania obrazu pohybom sa zvyčajne odstraňuje zmenou parametrov snímania a tu narážame na technologické limity. Okrem hardvérových riešení existujú aj softvérové riešenia ani tieto, ale nevedia úplne odstrániť takéto rozmazanie. Zdá sa, že počítače budeme musieť tak trochu naučiť rozpoznávať takúto formu chýb.

## Ciel'

Cieľom tejto práce je vytvoriť aplikáciu, ktorá dokáže s čo najvyššou presnosťou extrahovať z videa objekt rozmazaný pohybom. Na dosiahnutie tohto výsledku sa musíme zamerať na 3 hlavné časti takejto segmentácie. Prvou časťou je vytvorenie masky daného objektu na ktoromkoľvek snímku vo videu. Druhou časťou je segmentovanie tohto objektu z ostatných snímok videa. Treťou a najdôležitejšou časťou práce je detekcia rozmazaných častí v objekte a jeho okolí. Pri tejto časti budeme skúmať vlastnosti chyby rozmazania pohybom.

# Kapitola 1

## Analýza súčasných riešení

Potreba ľudí komunikovať a zdieľať svoje životy na internete pomocou multimediálneho obsahu výrazne mení internet ako taký. Efektívnosť komunikovať a informovať pomocou multimedálneho obsahu oproti textovému obsahu je zjavná, čo potvrdzujú aj najobľúbenejšie sociálne siete, pre ktoré je zdieľanie vizuálneho multimediálneho obsahu prioritné (Instagram, Facebook, Youtube). Tvorba a upravovanie takéhoto obsahu sa tak stáva čoraz viac používanou aj širokou verejnosťou. Potreba byť “online” a prezentovať svoje produkty alebo služby je dnes už takmer samozrejmostou pre každú firmu. Mohli by sme, teda povedať, že so spracovaním obrazu a videa sa stretol už takmer každý človek, nie len filmové štúdia a spravodajské portály. Proces segmentovania objektu z obrazu, patrí medzi tie najdôležitejšie úkony vrámci editovania a postprocesingu vizuálneho multimediálneho obsahu. Aj napriek výrazným pokrokom za uplynulé dve dekády, zdá sa, že pre niektoré typy objektov tento problém ostáva stále veľkou výzvou.

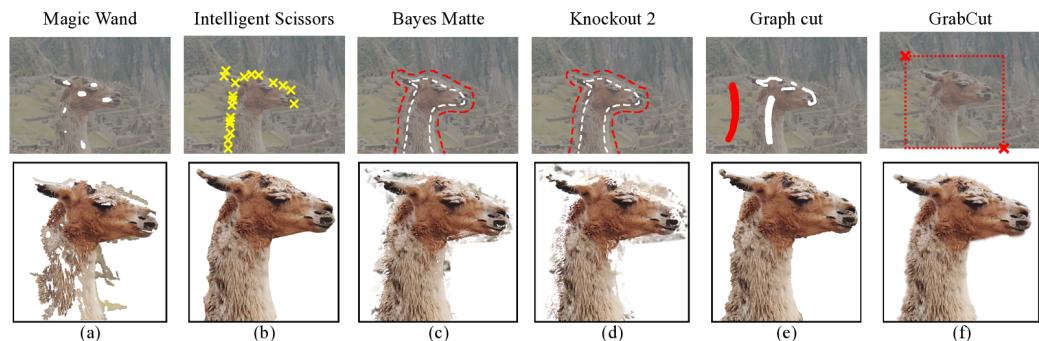
Kým pre človeka je úloha presnej lokalizácie a detekcie objektu vo videu ľahká úloha, a to aj pri rôznych degradáciách obrazu, spôsobených hardvérovou limitáciou alebo zlými svetelnými podmienkami. Skúsmo, preto rozanalizovať heuristiku, ktorú používa človek. V prvom kroku musí človek získať informáciu o danom objekte. Buď ju získa bez videnia akejkoľvek snímky videa, tým že bude informovaný o objekte a jeho vlastnostiach (farbe, morfológiu, atď.), alebo vidí jednu ktorúkoľvek snímku videa na ktorej je aj objekt a vlastnosti objektu sa naučí (spozná farbu, morfológiu, atď.) a pravdepodobne si aj spojí objekt vo videu s objektom ktorý pozná z reality. V druhom kroku je človeku premietaná sekvencia snímok videa a on sleduje objekt a jeho pohyb, na základe informácií ktoré sa naučil. Podproblémy ktoré tento proces ukrýva by sme mohli kategorizovať nasledovne: segmentácia objektu z obrázku, segmentácia objektu z videa a detekcia rozmazania objektu pohybom. Na nasledovnom obrázku 1.1 je možné vidieť vzťahy a súvislosti, ktoré tieto podproblémy majú.



Obr. 1.1: Problém segmenácie objektu z obrázku môžeme vnímať ako podproblém problému segmentácie objektu z videa, nakoľko video je sekvencia snímeiek (obrázkov). Podproblémom segmentácie objektu môže byť problém detekcie rozmazania objektu pohybom. Ktorý je aj podproblémom detekcie rozmazania polybom na obrázku.

## 1.1 Segmentácia objektu z obrázku

Potreba vedieť na obrázku aspoň jedným vstupom od užívateľa, označiť, ktorí objekt chceme segmentovať posúva tento problém do kategórie polo-automatického segmentovania, ktorého úlohou je na čo najnižší počet vstupov od užívateľa, segmentovať objekt najpresnejšie. V nasledujúcej časti si predstavíme a porovnáme techniky ktoré sú znázornené na obrázku 1.2



Obr. 1.2: Ukážka segmentácie objektu z obrázku pomocou hore uvedených techník. Vo vrchnom rade môžeme vidieť interakciu používateľa potrebnú na segmentovanie, bielou farbou je znázornené označenie popredia, červenou pozadia, žltými krížikmi hranica. V druhom rade vidíme presnosť segmentovania daných techník [28].

### 1.1.1 Technika Magic Wand

Táto technika je zamerána na segmentáciu objektov ktoré sú kompaktné vo farbe. Používateľ klikne na objekt ktorý chce segmentovať a farba daného pixla na ktorý klikol

sa uloží a vyhľadávajú sa ďalšie pixely s rovnakou alebo podobnou farbou v okolí pixela ktorý bol zakliknutí. Užívateľom nastavovaný parameter je tolerancia farby, teda jej odchylka od pôvodne zakliknutej farby. Táto technika nefunguje veľmi dobre ak objekt nie je kompaktný vo farbe alebo splýva s pozadím.

### 1.1.2 Intelligent Scissors

Technika hľadá hranicu objektu a pozadia pomocou dynamického programovania a formuluje problém ako problém prehľadávania grafu. Cieľom je nájsť optimálnu cestu medzi začiatočným vrcholom a množinou cieľových vrcholov. Vrámcí hľadania hranice popredia a pozadia na obrázku, technika hľadá optimálnu cestu od štartovacieho pixelu, po cieľovy pixel, kde pixely predstavujú vrcholy grafu a hrany sú tvorené pixlom a jeho 8 susedmi. Optimum je definované ako minimálna cesta od štartovacieho po cieľový pixel. Táto technika segmentuje objekt pomerne dobre, je však potrebných pomerne veľa vstupov od používateľa na dobré definovanie hranice.

### 1.1.3 Bayes Matte

Technika bola vytvorená najmä za účelom segmentácie objektov, ktorých hranica je tvorená gradientom prechádzajúcim od popredia k pozadiu. Takáto hranica vzniká najmä pri vlasoch a srsti. Technika začína s vytvorením si 3 másk. Maska popredia, maska pozadia, a maska hranice. Kvalitná segmentácia touto metódou je podmienená nie príľ veľkou maskou hranice a dobrou farebnou odlišiteľnosťou popredia a pozadia. Vďaka týmto podmienkam je nevyhnutná dostatočná používateľská interakcia.

Následne pre každý pixel z masky hranice počítame rovnicu:

$$C = \alpha * F + (1 - \alpha) * B \quad (1.1)$$

kde C je farba pixelu, F je maska popredia, B je maska pozadia a  $\alpha$  je priehľadnosť.

### 1.1.4 Knockout 2

Technika je podobná technike Bayes Matting, je používana v programe Adobe Photoshop. Jej zakladom je vytvorenie troch masiek, pre popredie, pozadie a masku hranice.

### 1.1.5 Graphcut

Metóda GraphCut je založená na hľadaní optimálneho rezu grafu. V prvom kroku sa obrázok prevedie na graf  $G(V,E)$ , kde V je množina vrcholov a E je množina hrán. Graf je plne prepojený, to znamená že každý vrchol je prepojený zo všetkými ostatnými. Vrcholy zodpovedajú pixelom a hrany reprezentujú susednosť medzi pixelami. V grafe

sa hľadá optimálny rez grafu. Optimálny v tomto prípade znamená, že rez splňa meracie podmienky. Meracími podmienkami môžu byť vzdialenosť pixelov, farebná vzdialenosť pixlov, vzdialenosť intenzity pixlov, textúra a ďalšie.

### 1.1.6 GrabCut

Táto metóda bola vyvynutá z dôvodu zlihávania predchádzajúcich techník v prípadoch, kedy sa popredie a pozadie podobajú farbou, intenzitou alebo textúrou. Technika vychádza z metódy GraphCut. Metóda používa Gaussovské zmiešané modely, jeden pre popredie a jeden pre pozadie, ktoré pomocou algoritmu iteratívnej minimizácie modelujú vrstvu pre popredie a vrstvu pre pozadie. Túto metódu aj metódu graphCut popíšeme bližšie v kapitole 2 Metodika a Metodológia kedže túto metódu v práci používam.

## 1.2 Segmentácia objektu z videa

Úloha polo-automatického segmentovania objektu z videa je dnes veľmi populárna a existuje viacero jej riešení. Metódy segmentácie objektu z videa by sme mohli rozdeliť do dvoch kategórii: metódy používajúce neurónové siete a ostatné. V danej oblasti existuje vedecká súťaž ktorá vyhodnocuje techniky segmentácie objektov z videa. DAVIS súťaž vznikla v roku 2016, obsahuje dataset 50 HD videí a 50 FullHD videí a jej organizátori robia každorčne seminár pod CVPR (Computer vision and pattern recognition) konferenciou. Vrámci súťaže existuje niekoľko vyhodnocujúcich metrík a preto aj najlepšie výsledky vrámci každého roku získa viacero techník. V roku 2016 najlepšie výsledky v danej problematike so zameraním na presnosť a rýchlosť získala technika OSVOS. V roku 2017 jej vylepšenie OSVOS-S [23]. Okrem tejto techniky môžeme spomenúť ešte techniku MaskTrack [26]. Tieto metódy sú založené na konvolučných neurónových sietach a existuje viacero ďalších ich prác v ktorých spájajú tieto techniky s ďalšími neurónovými sietami. Z týchto techník si bližšie predstavíme OSVOS techniku.

Medzi metódy nepoužívajúce neurónové siete patria metódy JumpCut [10], SnapCut[4], Fusionseg [13] a ďalšie. Tieto metódy sú založené na doprednej/spätnej propagácii masky objektu a detekcií pohybu pomocou Optical Flow a iných algoritmov. Metóda Video Snapcut [4] je použitá ako Roto Brush v Adobe After Effects CS5, a túto techniku si bližšie vysvetlíme v nasledujúcej podkapitole.

### 1.2.1 Video SnapCut: Robustná segmentácia objektu z videa používajúca lokálne klasifikátory.

Inšpiráciou môže byť aj tento článok, ktorý navrhol poloautomatickú metódu segmentácie objektov z videa so zameraním na presnosť. Bol implementovaný v Adobe After effects cs5 a technika dokáže dosiahnuť vynikajúce výsledky, presnosť výsledkov je však silne podmienená množstvom vstupov od používateľa.

Bai, Wang a kol [4] navrhli model segmentácie videa prekrývajúcich sa lokalizovaných klasifikátorov. Pozostáva zo skupiny prekrývajúcich sa okien okolo hranice objektu, pričom každý z nich je spojený s lokálnym klasifikátorom, ktorý segmentuje iba lokálnu časť hranice objektu. Pozície týchto klasifikátorov sa posúvajú vzhladom na pohybové vektorov odhadujúce pozíciu objektu na snímkach videa a celková segmentácia objektu z nového snímku sa dosiahne zoskupením výsledkov lokálnych klasifikátorov dohromady. Okrem toho, každý lokálny klasifikátor nesie lokálne príznaky, ako je farba, tvar a pohyb, a adaptívne ich integruje k učelu vytvorenia optimálnej klasifikácie. V porovnaní s inými technikami má tento prístup nasledujúce hlavné charakteristiky a výhody:

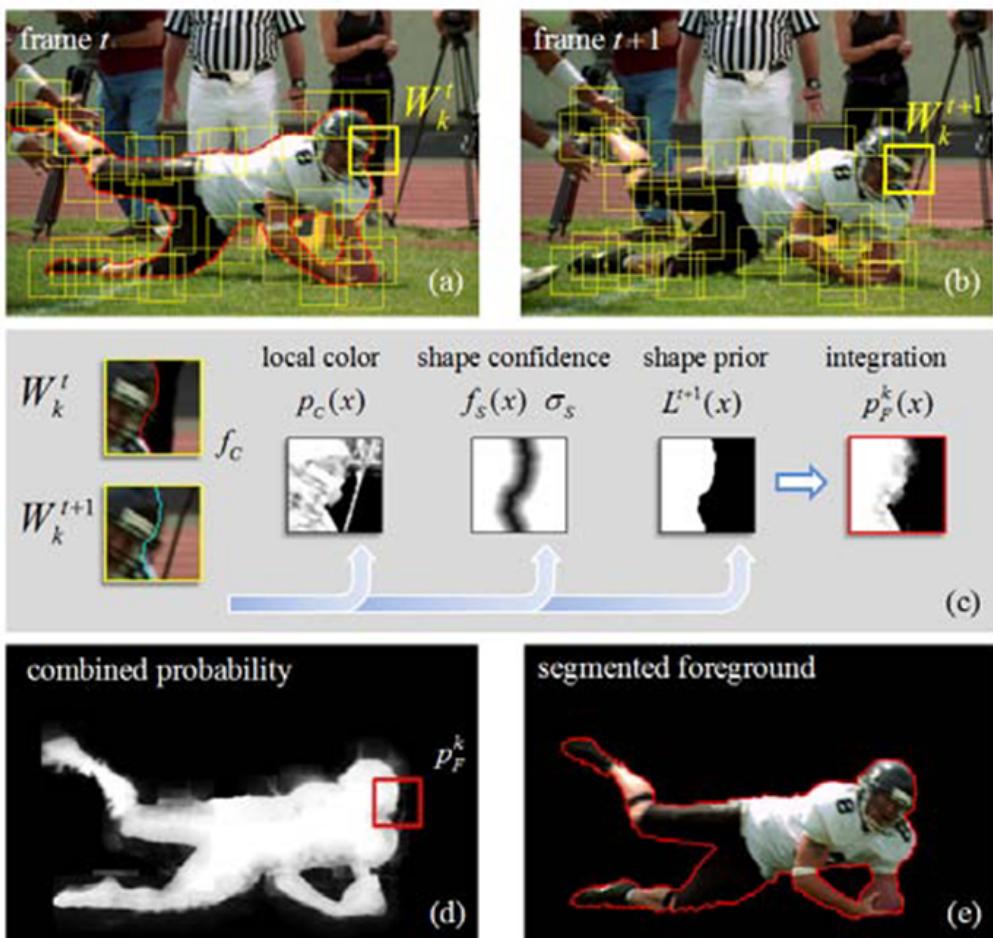
**Schopnosť zvládnuť zložité scény.** Vzhladom na to, že každý klasifikátor vyhodnocuje segmentáciu pre iba lokálnu oblasť, ich spojenie dáva oveľa väčšiu diskriminačnú silu ako metódy založené na globálnej štatistike. Navyše táto metóda sa dokáže adaptovať aj na topologické zmeny.

**Prirodzený pracovný postup.** Systém obsahuje sledovanie príznakov s ich doprednou propagáciu. Na rozdiel od iných známych prístupov, systém prirodzene podporuje lokálnu priestorovú a časovú úpravu. Po vykonaní ďalšieho zadania používateľa v lokálnom regióne sú ovplyvnené iba blízke klasifikátory. Navyše, vďaka tomu že klasifikátory sú sledované v priebehu času, lokálne úpravy možno ľahko šíriť cez snímky. Používateľ môže ľahko kontrolovať, aký veľký môže byť vplyv takejto úpravy na lokálnu klasifikáciu, a to priestorovo aj časovo. Táto vlastnosť je tiež prínosom pre výpočtovú zložitosť, pretože sa zabráni bežnému problému opäťovného spustenia segmentácie na celom videu pre každú lokálnu korekciu [4].

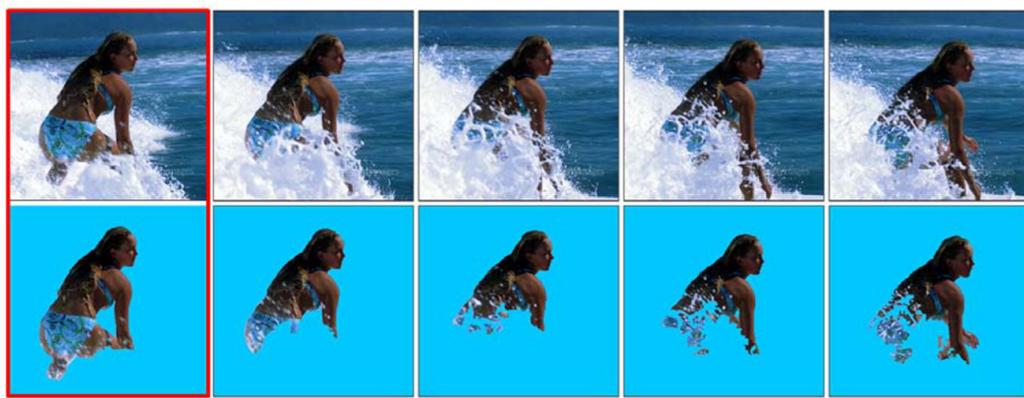
Na nasledujúcich dvoch obrázkoch je demonštrovaný princíp Snapcut metódy a na ďalších dvoch sú demonštrované výsledky Snapcut metódy aj pre náročné scény.



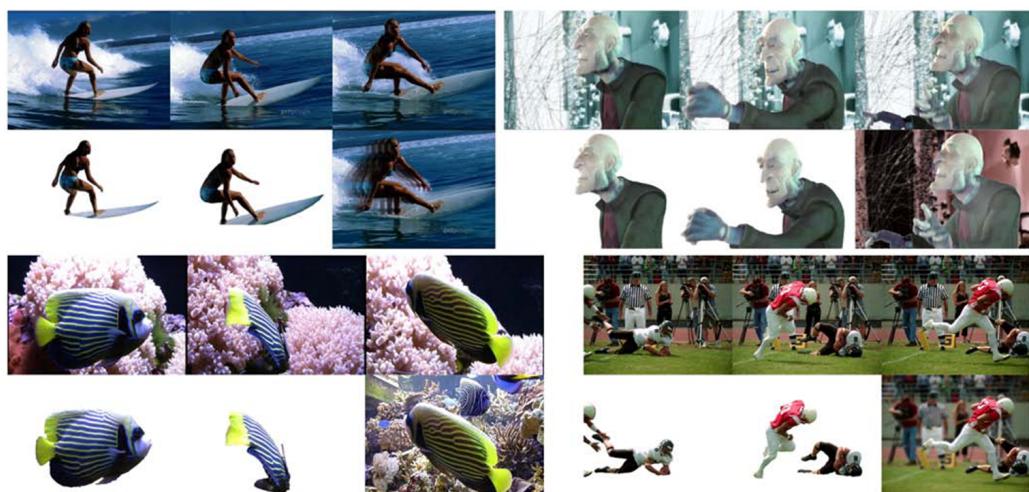
(červená krivka) pred (len globálny pohyb a optický tok (ang. Optical Flow)) (a) po 1. iterácii (b), po 4. iterácii. (c) a 6. iterácii (D). Obrys sa vyvíja tak, aby sa napasoval na hranicu objektu.



(a) Prekrývajúce klasifikátory (žlté štvorce) sa inicializujú pozdĺž hranice objektu (červená krivka) na snímke t. (b) Tieto klasifikátory sa prenášajú do ďalšej snímky pomocou odhadu pohybu. (c) Klasifikátor obsahuje lokálny farebný model a tvarový model, inicializujú sa v snímke t a aktualizuje sa na  $t + 1$ . (d) Lokálne výsledky klasifikácie sa potom kombinujú, aby sa vytvorila globálna mapa pravdepodobnosti popredia. (e) Konečný segmentovaný objekt popredia na snímke  $t + 1$ .



Navrhovaný systém automaticky spracováva komplexné oklúzie uvedené v tomto príklade. Prvá snímka je kľúčová snímka.



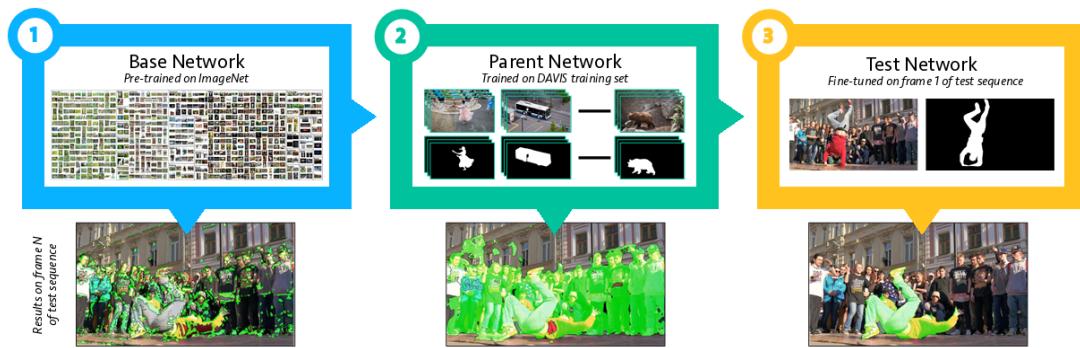
Pre každý zo štyroch príkladov podkategórie v hornom riadku zodpovedajú trom originálnym snímkam, v spodnom riadku nasledujú dve snímky zodpovedajúce segmentovanému poprediu a jedna s použitím špeciálneho efektu.

### 1.2.2 OSVOS

OSVOS (ang. One-Shot Video Object Segmentation) sú iniciály techniky ktoré by sme mohli preložiť ako segmentácia videa na zaklade jedného snímku s vyznačením objektom. Ide o polo-automatickú techniku segmentácie objektu z videa, založenú na FCNN - plne konvolučnej neurónovej sieti. Táto technika dokáže uspešne preniesť generickú sémantickú informáciu, naučenú na ImageNet sieti, na úlohu segmentácie popredia, a nakoniec k naučeniu vzhľadu objektu na základe jedného snímku z videa, v ktorom je vyznačený objekt. Napriek faktu že všetky snímky videa sú spracovávané nezávisle od seba, výsledky sú súvislé a stabilné. Autori posunuli výsledky práce v danej úlohe o významnú hodnotu 79% oproti 68% [7].

## OSVOS - Trénovacie detaily

Táto technika kombinuje offline a online tréning. Základná konvolučná sieť architektúry OSVOS je predtrénovaná na ImageNet sieti ktorá rozpoznáva objekty. Autori OSVOS pretrénovali túto sieť na Davis Datasets, k naučeniu generickej informácie ako segmentovať objekt od pozadia, jeho zvyčajných tvarov atď. Použili Stochastic Gradientny zostup (SGD) s momentom 0.9 pre 50 000 iterácií. Rozšírili data pomocou zrkadlenia o priblíženia. Uroveň učenia (learning rate) je nastavená na 10<sup>-4</sup>, a je znižovaná. Po tomto offline tréningu, sieť je naučená extrahovať popredie z pozadia. Online tréning ako môžeme vidieť aj na obrázku 1.5 [7].



Obr. 1.5: Štruktúra OSVOS siete: 1.) Základná sieť je CNN predtrénovaná na ImageNete 2.) Rodičovská sieť vychádza zo základnej a trénovaná na DAVIS datasete. 3.) Testovacia sieť je vrchnou vrstvou, ktorá vďaka nášmu vstupu - segmentácie objektu na jednom snímku videa, rapídne zvýši presnosť segmentovania objektu vo videu. [7]

## 1.3 Detekcia rozmazania objektu pohybom

Väčšina digitálnych obrázkov obsahuje dva typy regiónov: ostrý a rozmazaný. Rozmazanie môžeme rozdeľovať hlavne do dvoch kategórii, rozostrenie a rozmazanie pohybom. Efekt rozmazania pohybom je určitá forma deformácie fotky alebo snímky videa. Spôsobuje ho nedostatočná rýchlosť uzávierky snímača fotoaparátu vzhľadom na pohybujúce sa objekty pred kamerou alebo na pohyb samotnej kamery. Výsledkom je deformácia na pixeloch kde je zachytený objekt. Intenzita svetla dopadajúca na časti snímača ktoré reprezentujú tieto pixely je posunutá v smere pohybu objektu a výsledkom je prelínanie hrán a textúry objektu. Ak sa hýbe kamera, a žiadny z objektov sa nehýbe rovnakou rýchlosťou ako kamera, jedná sa o globálny efekt rozmazania pohybom, ak sa hýbe len jeden alebo niekoľko objektov, efekt rozmazania je lokálny.

Existuje množstvo prác zameraných na detekciu a segmentáciu rozmazaných oblastí na obrázkoch. Aj v tejto oblasti spracovania obrazu zaznamenali veľký úspech metódy

založené na metódach rozpoznávania obrazcov. Tieto práce môžeme rozdeliť do dvoch kategórii: práce používajúce neurónové siete, a práce používajúce iné riešenia.

### 1.3.1 Metódy používajúce neurónové siete

Existuje veľa dôvodov prečo sú metódy používajúce neurónové siete populárne v tejto téme. Snaha vytvoriť dostatočne robustné riešenie je na prvom mieste. Rozmazanie pohybom môže byť globálne alebo lokálne, radiálne alebo lineárne, uniformné alebo neuniformné, rozmazanie vzdialením/priblížením alebo môžu byť niektoré kategórie kombinované. V nasledujúcej časti si predstavíme dva články. Jeden popisuje príznaky vhodné na detekciu rozmazania pohybom a druhý opisuje neurónovu sieť ktorá rozpoznáva rozmazanie pohybom.

#### Príznaky popisujúce rozmazanie pohybom

V nasledujúcej sekcií popíšem ako Usman Ali a Muhammad Tariq Mahmood v článku [2] popisujú meranie rozmazania pohybom. Nech  $I_b(x, y)$  je rozmazaný vstupný obrázok z ktorého budeme počítať množstvo rozmazania pre každý pixel. Toto dosiahneme aplikovaním operátora rozmazania  $B$  (okno/maska) ktorého stredom bude pixel  $\omega(x, y)$ . Aplikovaním rovnakého spôsobu pre všetky pixely v obrázku dostaneme mapu rozmazania.

$$M(x, y) = B(I_b(x, y)) \quad (1.2)$$

Normalizovaný tvar v intervale  $[0,1]$

$$M(x, y) = (M(x, y) - \min(M)) / (\max(M) - \min(M)) \quad (1.3)$$

kde  $\min(M)$  a  $\max(M)$  sú minimálne a maximálne hodnoty v inicializovanej mape rozmazania. Operátory rozmazania môžeme na základe princípov akými fungujú rozlíšiť do 4 skupín.

- **Derivačné Operátory (ang. Derivative Based Operators)** - meranie rozmazania sa zakladá na derivácii hodnôt obrázku. Tieto operátory sú založené na predpoklade že ostré časti obrázkov majú ostré hrany v porovnaní s rozmažnými. Takéto rozdelenie na základe ostrosti hrán sa robí pomocou prvej a druhej derivácie.
- **Štatistické operátory (ang. statistical-based operators)** - tieto operátory obsahujú niekoľko štatistických meraní ktoré sa počítajú na oknách teda operátoroch a ich susednosti a vyhodnocovaní ktoré časti sú rozmažané.
- **Transformačné operátory** - Táto skupina operátorov pracuje na frekvenčnej urovni a snaží sa vytvoriť repliku pôvodnej informácie.

- **Zmiešané operátory** - operátory ktoré nepatria ani do jednej z predchádzajúcich kategórii [2].

Na obrázku 1.6 môžeme vidieť konkrétné príznaky a ich zaradenie do skupín.

Sr. No.	Blur Operator	Abbr.	Sr. No.	Blur Operator	Abbr.
1	Gradient Histogram Span	DER01	17	Gray-level local variance	STA06
2	Kurtosis	DER02	18	Normalized Gray-level variance	STA07
3	Gaussian derivative	DER03	19	Histogram entropy	STA08
4	Gradient energy	DER04	20	DCT energy ratio	STA09
5	Squared gradient	DER05	21	DCT reduced energy ratio	STA10
6	Tenengrad	DER06	22	Power spectrum	TRA01
7	Tenengrad variance	DER07	23	High-frequency multiscale Fusion and Sort Transform (HiFST)	TRA02
8	Energy of Laplacian	DER08	24	Sum of wavelet coefficients	TRA03
9	Modified Laplacian	DER09	25	Variance of wavelet coefficients	TRA04
10	Diagonal modified Laplacian	DER10	26	Ratio of wavelet coefficients	TRA05
11	Variance of Laplacian	DER11	27	Brenner's measure	MIS01
12	Singular value decomposition	STA01	28	Image contrast	MIS02
13	Sparsity of dark channel	STA02	29	Image curvature measure	MIS03
14	Total variation	STA03	30	Steerable filters-based	MIS04
15	Local binary pattern	STA04	31	Spatial frequency	MIS05
16	Gray-level variance	STA05	32	Vollath's autocorrelation	MIS06

Obr. 1.6: Príznaky testované v článku a ich zaradenie do kategórii. Kategória derivačných operátorov je značená *DER/[číslo]*, štatistické operátory *STA/[číslo]*, transformačné operátory *TRA/[číslo]*, zmiešané operátory *MIS/[číslo]*

Vo výsledkoch tohto článku je vidno že je možné segmentovať rozmazené a ostré časti obrázku, ak sú zvolené správne príznaky. Autori vytvorili šablónu systému ktorý obrázky vyhodnocuje a na základe určitých vlastností vyberá tie najvhodnejšie príznaky.

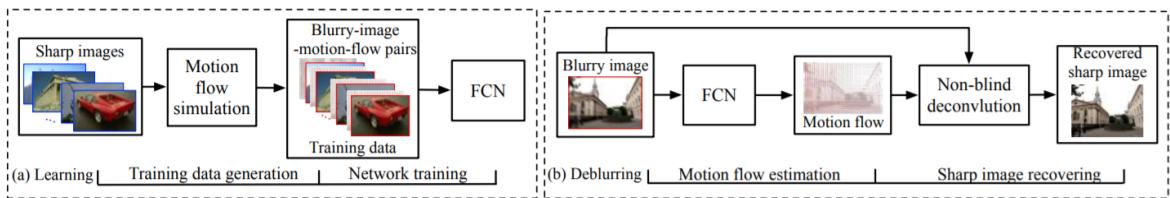
### Rozmazanie pohybom pomocou detektie pohybu: FCNN

V práci [11] autori popisujú metodu ktorou detekujú rozmazanie pohybom ako tok pohybu pomocou metódy Optical Flow a plne konvolučnej hlbokej neurónovej siete k obnoveniu ostrého obrazu.

Odstránenie heterogénneho rozmazania pohybom na úrovni pixelov sa zdá byť tažko vyriešiteľným problémom. Prevládajúce riešenia sú zamerané na získanie jadra rozmazania pomocou predošej informácie, ale rozsiahla literatúra o tejto téme naznačuje ťažkosti z identifikáciou predchádzajúcej informácie, ktorá je dostatočne informatívna a všeobecná. Prístup tejto práce spočíva v zameraní sa na učenie z dát [11].

V rámci detektie rozmazania objektov pohybom pomocou neurónových sietí je jedným z najdôležitejších faktorov vytvorenie dostatočne veľkého a všeobecného datasetu pre danú úlohu. Jediný dostupný dataset, ktorý som aj po prečítaní veľkého množstva odborných článkov našiel je dataset pozostávajúci z 1000 obrázkov (približne

700 pre rozmazanie rozostrením (ang. Out of Focus Blur), približne 300 pre rozmazanie pohybom) a k nim prislúchajúcim presným maskám. Tento dataset je zverejnený na stránkach oddelenia informatiky a počítačového inžinierstva Čínskej univerzity v Hong Kongu. (odkaz: <http://www.cse.cuhk.edu.hk/leojia/projects/dblurdetect/dataset.html>) Takto malý dataset je aj po augmentácií príliš malý na natrénovanie dobrej neurónovej siete. Prínosom tohto článku je architektúra ktorá vytvorí z ostrého obrázku, obrázok s rozmazaním pohybom pre určitý región. Následne sa sieť trénuje na naučenie sa tohto regiónu rozmazaného pohybom. Na obrázku 1.7 môžeme vidieť architektúru systému.

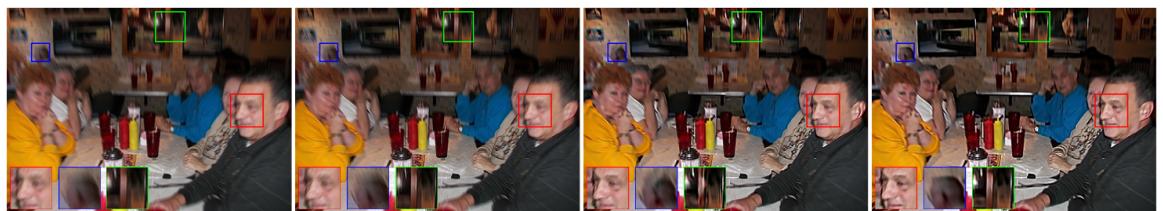


Obr. 1.7: Architektúra metódy Rozmazanie pohybom pomocou detekcie toku pohybu. V časti (a) môžeme vidieť časť architektúry, ktorá vytvára dataset na učenie siete. Z ostrých obrázkov pomocou simulácie rozmazania pohybom vytvoríme dataset na ktorom necháme sieť učiť sa. V časti (b) vidíme proces testovania kde z vstupného obrázku rozmazaného pohybom pomocou neurónovej siete a Optical Flow metódy získame ostrý obrázok.

Pre generovanie trénovacieho datasetu použili autori 200 obrázkov veľkosti 300x 400 pixelov z BSD500 datasetu, ako vstupné ostré obrázky. Následne nezávisle nasimulovali 10 000 máp rozmazania pohybom pre určité regióny na obrázku. Každému obrázku tak pridelili 50 rôzných máp pre rozmazanie pohybom. Dataset na trénovanie tak obsahoval dokopy 10 200 vstupov. Na nasledujúcich dvoch obrázkoch 1.8 a 1.9 môžeme vidieť detekciu rozmazania pohybom v porovnaní s metódou [32] a výsledný zaostrený obrázok pomocou tejto metódy v porovnaní s metódou [37] a [32].



Obr. 1.8: Detekcia pohybu pomocou algoritmu Optical flow. Prvý riadok obsahuje 4 vstupné obrázky čiastočne rozmazané pohybom. Druhý riadok je detekcia pohybu pomocou Optical Flow techniky v metóde [32]. Tretí riadok je metóda autorov článku a jej detekcia pohybu pomocou Optical Flow metódy [11]



Obr. 1.9: Porovnanie výsledkov zaostrenia obrázkov: Zľava pôvodný rozmazaný obrázok, zaostrenie pomocou [37] metódy, zaostrenie pomocou [32] metódy, zaostrenie pomocou metódy Rozmazanie pohybom pomocou detektie pohybu [11]

### 1.3.2 Ostatné metódy

Medzi tieto metódy patria metódy používajúce aj Optical flow, alebo zameriavajúce sa na niektoré konkrétné funkcie.(Blind image de-convolution, Low depth of field, Edge sharpness analysis, Low directional high frequency energy.)

Rozmazanie pohybom na digitálnom obrázku môže byť modelované konvolúciami medzi obrázkom a jadrom rozmazania pohybom, ktorý obsahuje distribuciu faktoru rozpínania sa (ang. point spread factor = PSF) rovnú uhlu v ktorom sa faktor rozpína. Príklad jadra rozmazania pohybu s dĺžkou 7 a uhlom 45 je na obrázku 1.10.

0	0	0	0	0	0.0145	0
0	0	0	0	0.0376	0.1283	0.0145
0	0	0	0.0376	0.1283	0.0376	0
0	0	0.0376	0.1283	0.0376	0	0
0	0.0376	0.1283	0.0376	0	0	0
0.0145	0.1283	0.0376	0	0	0	0
0	0.0145	0	0	0	0	0

Obr. 1.10: Jadro rozmazania pohybu s dĺžkou 7 a uhlom 45 stupňov

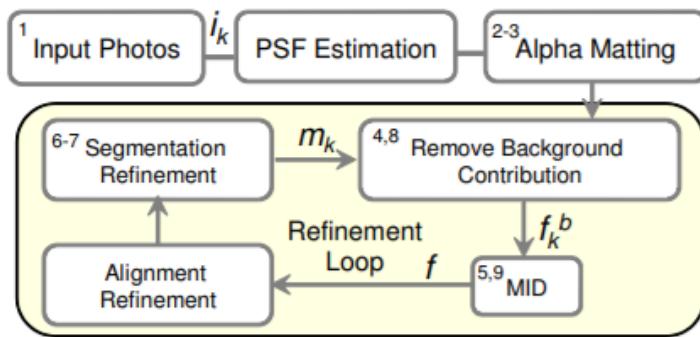
### 1.3.3 Invertible Motion Blur in Video

Amit Agrawal a Ramesh Raskar [1] v roku 2009 navrhli metódu ktorá dokáže na základe informácií z videa vytvoriť spätnú PSF funkciu bez núl a pomocou nej odstrániť rozmazanie pohybom z videa.

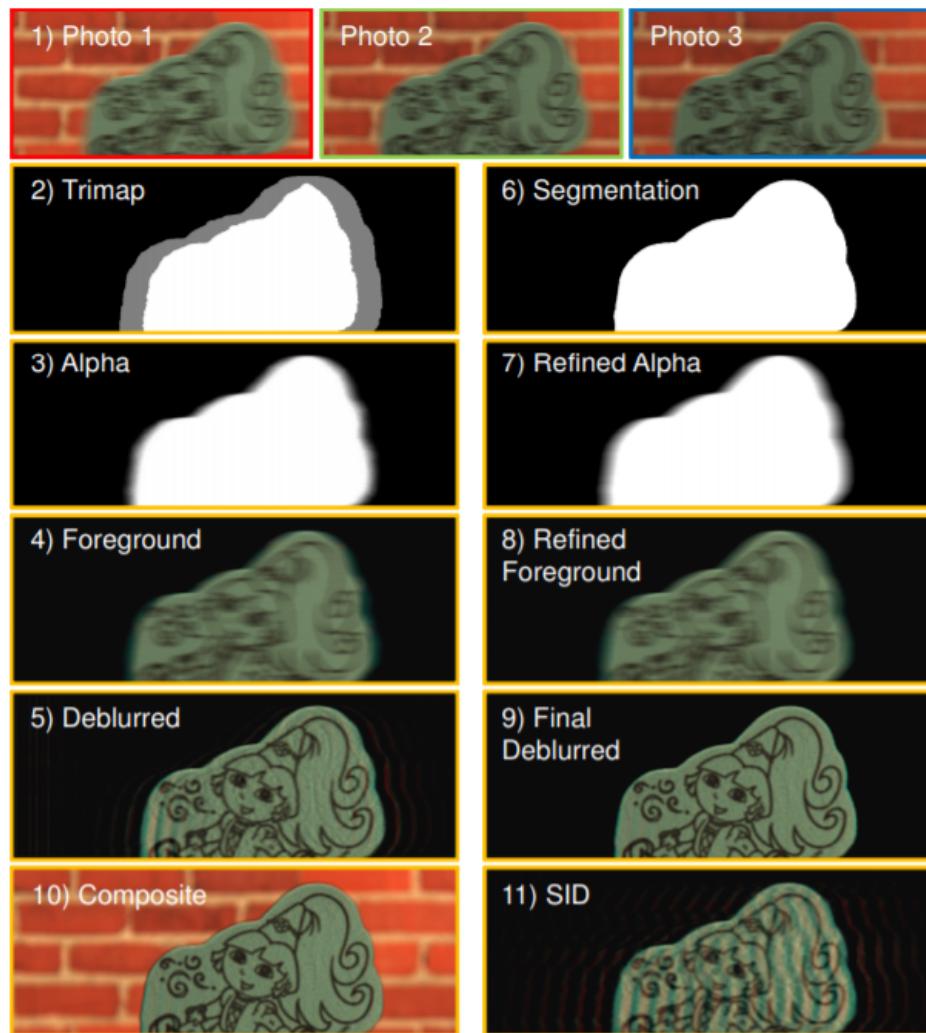
Automatické rozostrenie(ang. Automatic deblurring) obsahuje 3 kritické komponenty. Ziskanie spätej PSF funkcie, zaznamenanie pohybu pohybujúcich sa časti a segmentovanie pohyblivých častí zo statického pozadia. Navyše, presnosť statických častí scény môže byť zachovaná. Predchádzajúce prístupy k problematike viedli k vyriešeniu jednej alebo viacerých častí tohto problému ale žiadna ich neriešila všetky. Síce riešenie týchto problémov pre fotografiu je veľmi zložité, Ramesh Raskar a Amit Agrawal dokázali že riešenie pre video sa zdá byť dosiahnutelnejším [1].

Automatické zaostrenie je možné uskutočniť z viacerých snímok. Klúčovou myšlienkou je zaznamenať rovnaký objekt s rôznymi PSF, takže nulové hodnoty v frekvenčnom komponente jedného snímku môžu byť vyplnené z iných snímkov. Kombinovaná frekvenčná transformácia sa stáva nenulová, čo robí zaostrenie dosiahnutelním. Toto autori dosiahli jednoduchou zmenou doby expozicie po sebe idúcich snímok videa. Ich technika nevyžaduje žiadnen pohyb fotoaparátu ani žiadne ďalšie špeciálne úkony počas času expozície. Môže byť implementovaná na štandardnú video kameru s vlastnosťou automaticky nastaviť expozíciu [1].

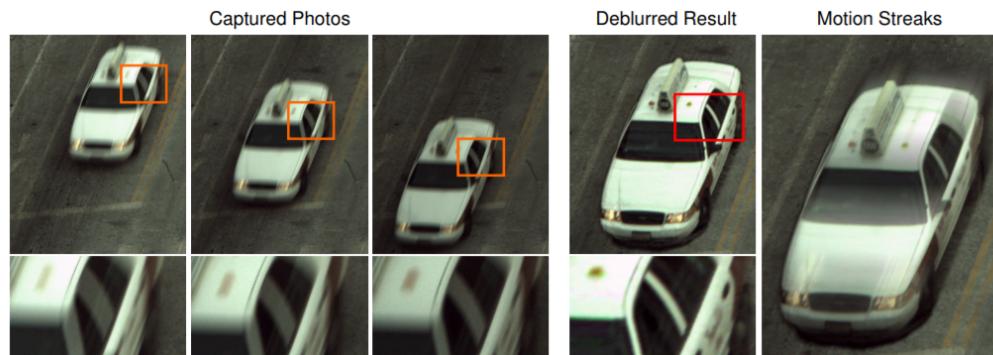
Táto metóda ale môže fungovať len pre lineárne rozmazanie pohybom, ktoré je homogénne, tzn. objekt sa počas snímania pohybuje rovnakou rýchlosťou a zachováva smer pohybu. Na nasleduj[cich obrázkoch môžeme vidieť architektúru funkcie, prislusajúce výsledky po jednotlivých krokoch funkcie, a výsledky použitia tejto metódy.]



Prehľad algoritmu automatického vypočtu PSF funkcie, segmentácie a zaostrenia. Pri každom bode môžeme vpravo hore vidieť indexy ktoré označujú časti obrázka ktorý je nižšie a prezentuje obrázok a jeho zmeny po jednotlivých krokoch v tejto architektúre.



Vstupné obrázky (1). Medzivýsledky (2-10) ukazujú ako sa alfa mapa, zaostrenie a segmentácia zlepšujú počas prvej iterácie. (11) zobrazuje výsledný obrázok.



Obr. 1.12: Jednoduchým zmenením doby expozičie pre video snímky, môžeme odstrániť viacobrazové rozmazanie. (Vľavo) Fotky auta snímané rôznou expozíciou. Všimnite si zmenu v osvetlení a veľkosť rozmazania na fotkách. (Vpravo) Objekt záujmu je automaticky segmentovaný, zaostrený a vložený do pozadia pužitím videa z rôznorodou rýchlosťou uzávierky. Neznáme oblasti okolo objektu môžu byť generované lineárной kombináciou výsledného obrázku a pôvodných obrázkov.

# Kapitola 2

## Metodika a Metodológia

Cieľom tejto kapitoly je podrobne opísať navrh riešenia zadanej úlohy. Podrobne popíšať techniky a princípy z ktorých budem vychádzať pri implementácii a návrhu riešenia.

### 2.1 Segmentácia objektu z obrázku

Jednou z najviac používaných a potrebných techník pre spracovanie obrazu je segmentácia obrazu do jednotlivých častí/vrstiev. Oddeľuje sa popredie, teda objekt záujmu, od pozadia, teda zvyšku obrázka, pre ďalšie spracovanie alebo rozpoznávanie obrazu. Zložitosť daného problému spočíva v segmentácii objektov z obrázkov, na ktorých vzniká určitá degradácia informácie. Ak vnímajme obrázok ako spojitého priestorovú informáciu, existujú anomálie spôsobené hardvérovou limitáciou zariadení ktoré obraz snímajú. Medzi takéto anomálie patrí napríklad rozmazanie objektu pohybom, oklúzie spôsobené inými objektami alebo nedostatočné svetelné podmienky. Ďalšou výzvou sú objekty ktoré sa segmentujú veľmi tažko, ako napríklad stromy alebo kríky, vlasy alebo transparentné objekty, napríklad igelitový sáčok. Spomenuté anomálie a tažko segmentovateľné objekty možeme vidieť aj na obrázku Obr. 2.1.

V nasledujúcej časti opisujem metódu Grabcut, ktorú som v odbornej literatúre našiel ako najvhodnejšiu pre segmentovanie popredia na základe nízkeho počtu vstupov od používateľa.



Obr. 2.1: Prvý riadok obrázok vľavo: objekt prekrývajú kvapky vody, prvý riadok stredný obrázok, degradácia autobusu spôsobená rýchlym pohybom, prvý riadok vpravo, oklúzia objektu spôsobená slnečnými lúčmi. druhý riadok vľavo: tažko segmentovateľný objekt - strom, druhý riadok v strede: tažko segmentovateľný objekt - vlasy, spodný riadok vpravo: tažko segmentovateľný objekt - transparentný sáčok.

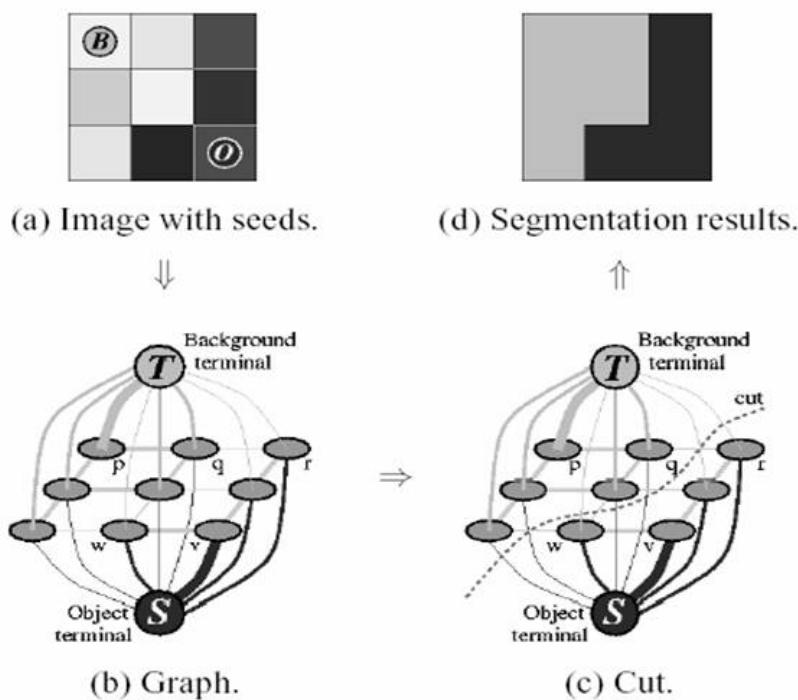
### 2.1.1 Grabcut

Táto metóda bola vyvynutá z dôvodu zlihávania predchádzajúcich techník v prípadoch, kedy sa popredie a pozadie podobajú farbou, intenzitou alebo textúrou. V článku [6] z roku 2001 opisujú autori ako je možné oddelenie popredia od pozadia na šedotónovom obrázku previesť do teórie grafov a pomocou vstupov od používateľa a správnej metriky hľadať minimálny rez grafu. ktorý bude na našom obrázku segmentáciou popredia od pozadia. V roku 2004 autori článku [28] vylepšili túto metódu tak, aby pracovala s farebnými obrázkami a aby bola iteratívna, na záver aplikovali na okraje segmentácie takzvaný border matting, ktorý posunul výslednú segmentáciu do jemnej segmentácie.

#### Graph-cut

Metóda GraphCut je založená na hľadaní optimálneho rezu grafu. V prvom kroku sa obrázok prevedie na graf  $G(V, E)$ , kde  $V$  je množina vrcholov a  $E$  je množina hrán. Vrcholy zodpovedajú pixelom a hrany reprezentuju susednosť medzi pixelmi. Do grafu pridáme ešte dva vrcholy, jeden pre objekt  $S$  a druhý pre pozadie  $T$ . Množina hrán bude pozostávať z dvoch typov: susedných hrán a terminálnych hrán. Každý vrchol (pixel)  $p$  má dve terminálne hrany  $(p, S)$  a  $(p, T)$  spájajúce ho s popredím a pozadím a zároveň 3,5 alebo 8 susedných hrán v závislosti od toho či je daný vrchol v rohu, na okraji alebo v strede obrázka. Ďalším krokom je vytvorenie váh pre hrany v grafe. Nato aby sme mohli nastavovať váhy potrebujeme uvažovať nejakú metriku. Meracími

podmienkami môžu byť vzdialenosť pixelov, farebná vzdialenosť pixlov, vzdialenosť intenzity pixlov, textúra a ďalšie. Kedže metóda Graphcut má inú metriku ako metóda Grabcut, čo je spôsobené aj obrázkami ktoré sú týmito metódami spracovávané (šedotónové a farebné), nie je dôležité tu opisovať metriku metódy Graphcut. Posledným krokom je vypočítanie globálneho optimálneho minimálneho rezu a oddelenia terminálnych vrcholov  $S$  a  $T$ . Graphcut algoritmus používa Max-flow Min-cut algoritmus. V algoritme sa predpokladá že maximálny objem ktorý môže prejsť z bodu  $S$  do bodu  $T$  vzhľadom na váhy medzi týmito bodmi, je zároveň minimálnym rezom grafu. Minimálny rez grafu, je taký rez ktorý rozdelí graf na dva samostatné komponenty tým, že odstráni hrany ktoré majú najnižšiu hodnotu. Na obrázku 2.2 môžeme vidieť ilustráciu prevodu obrázku na grafu, vypočítanie blízkostí pixelov, nájdenie optimálneho rezu grafu a vizualizovanie takéhoto rozdelenia na obrázku.



Obr. 2.2: Ukážka techniky Graphcut s ST minimálnym rezom. 2D segmentácia obrázka veľkosti 3x3. Vrchol  $V$  patrí objektu  $S$  teda poprediu a vrchol  $P$  patrí  $T$  teda pozadiu. Váhy hrán sú vizualizované ich hrúbkou. Hrany s nízkou hodnotou sú odstrané minimálnym rezom grafu [6].

**Grabcut algoritmus:** Algoritmus pracuje na farebnom RGB obrázku. Kým pri GraphCut algoritme, kde sme používali šedotónový obraz, bolo postačujúce pracovať s histogramami intenzity pixlov, pri farebnom obrázku by nebolo efektívne vytvárať histogramy jednotlivých farebných kanálov, ale lepšie je použiť priamo Gaussove zmie-

šané modely (ang. Gaussian Mixture Models, ďálej len GMM). Použijeme dva GMM, jeden pre popredie a jeden pre pozadie, každý tvorí plne kovariančný Gaussov model s K komponentami (typicky K=5). Z pohľadu používania GMM optimálne, vytvoríme vektor  $k = \{k_1, \dots, k_n, \dots, k_N\}$  s  $K_n \in \{1, \dots, K\}$ , priraďujúci každému pixelu každý GMM komponent z modelu popredia alebo modelu pozadia, s určitou pravdepodobnosťou nakoľko do daného komponentu pixel patrí. Gibbsová funkcia energie pre segmentáciu je:

$$E(\alpha, k, \theta, z) = U(\alpha, k, \theta, z) + V(\alpha, z) \quad (2.1)$$

kde  $\alpha$  je priehľadnosť,  $z$  sú dáta,  $U$  vypočítava začlenenie pixelu do modelu popredia alebo pozadia.  $V$  je hladkosť ktorá vyhodnocuje blízkosť medzi susednými pixelmi.

$$U(\alpha, k, \theta, z) = \sum_n D(a_n, k_n, \theta, z_n) \quad (2.2)$$

kde  $D(a_n, k_n, \theta, z_n) = -\log p(z_n | a_n, k_n, \theta) - \log \pi(a_n, k_n)$ , kde  $p$  je gaussové rozdelenie pravdepodobnosti a  $\pi$  sú váhy GMM, takže:

$$D(a_n, k_n, \theta, z_n) = -\log \pi(a_n, k_n) + \frac{1}{2} \log \det \sum(a_n, k_n) \quad (2.3)$$

$$+ \frac{1}{2} [z_n - \mu(a_n, k_n)]^T \sum(a_n, k_n)^{-1} [z_n - \mu(a_n, k_n)] \quad (2.4)$$

z tohto dôvodu sú parametre modelu:

$$\theta = \{\pi(\alpha, k), \mu(\alpha, k), \sum(\alpha, k), \alpha = 0, 1, k = 1 \dots K\} \quad (2.5)$$

kde váhy sú  $\pi$ , priemer je  $\mu$ , a kovariancia  $\sum$  2\*K Gaussových komponentov pre popredie alebo pozadie. Hodnota  $V$  hladkosti je nezmenená od tej používanej v Graphcut metóde pre šedotónový obraz.

$$V(\alpha, z) = \gamma \sum_{(m,n) \in C} [\alpha_n \neq \alpha_m] \exp -\beta ||z_m - z_n||^2 \quad (2.6)$$

kde  $[\phi]$  nadobúda hodnotu 0 aleb 1,  $C$  je množina párov susedných pixelov.  $\gamma$  je konštanta nastavená na 50 a  $\beta$  je:

$$\beta = (2\langle (z_m - z_n)^2 \rangle)^{-1} \quad (2.7)$$

### Segmentácia pomocou iteratívnej minimizácie energie

Schéma iteratívnej minimizácie pracuje iteratívne. Táto vlastnosť dovoľuje automatické zlepšovanie hodnôt priehľadnosti pre pixely v regióne  $T_U$  v trimape. GrabCut systém pozostáva z nasledujúcich krokov. V prvom kroku vypočítame  $k_n$  hodnoty komponentov pre každý pixel  $n$ . Krokom dva je nastavenie GMM parametrov. vzávislosti od pixelov a pravdepodobností z akou do nich patria. GMM parametre definujú jednotlivé komponenty, a teda aj model pre popredie a pozadie. V kroku tri je použitý

minimálny rez grafu. Štruktúra algoritmu garantuje konvergovanie riešenia, vďaka každému z krokov 1 až 3 iteratívnej minimizácie ktoré podmieňujú minimizáciu funkcie energie  $E$ , vzhlľadom na premenné  $k, \theta, \alpha$  [28].

### Grabcut algoritmus

#### Inicializácia

- Používateľ vytvorí štvoruholník  $R$  okolo objektu záujmu a tým inicializuje Trimap (Trimap je maska s tromi rôznymi hodnotami),  $T(T_F, T_U, T_B)$ ,  $T_U = R$ ,  $T_F = \emptyset$  a  $T_B = T_U^c$ .
- $\alpha_n = 0; n \in T_B$  a  $\alpha_n = 1; n \in T_U$ .
- GMM pre popredie a pozadie sa inicializujú z množín  $\alpha_n = 0$  a  $\alpha_n = 1$ .

#### Iteratívna minimalizácia

1. Prirad GMM komponenty pixelom: Pre každé  $n \in T_U$

$$K_n := \arg \min_{k_n} D_n(\alpha, k_n, \theta, z_n) \quad (2.8)$$

2. Vypočítaj GMM parametre z dát  $z$

$$\theta := \arg \min_{\theta} U(\alpha, k, \theta, z) \quad (2.9)$$

3. Odhadni segmentáciu, použi min-cut na vyriešenie:

$$\min_{\alpha} \min_{n: n \in T_u} \min_k E(\alpha, k, \theta, z) \quad (2.10)$$

4. Opakuj od kroku 1, kým riešenie nekonverguje [28].

#### Vstupy od používateľa

1. Editovanie: uprav niektoré pixely na  $\alpha_n = 0$  (vyznačenie pozadia) alebo  $\alpha_n = 1$  (vyznačenie popredia), obnov Trimap  $T$  a vykonaj krok 3.
2. Operácia vyhladenia: [voliteľná] zopakuj celý algoritmus iteratívnej minimalizácie.

## 2.2 Segmentácia objektu z videa

Za uplynulé roky prebehol pomerne významný a zásadný posun v problematike segmentácie objektu z videa. Tento posun je do veľkej miery spôsobený prístupmi pomocou neurónových sietí. Od roku 2016 existuje DAVIS súťaž, ktorá poskytuje dataset 50 videí pre segmentáciu objektov z videa. Jej autori každoročne organizujú seminár vrámci CVPR konferencie. Metódy delia do dvoch kategórii. Prvou je čiastočné učenie s učiteľom, ktoré výchadza z techniky učenia s učiteľom kde ku vstupným dátam máme aj informáciu o správnom výstupe (zaradení vstupu do kategórie / triedy). Keďže vytvoriť rozsiahlu databázu so vstupnými dátami + odpovedajúcimi správnymi výstupmi, je časovo náročné, vznikla technika čiastočného učenia sa s učiteľom (ang. semi-supervised machine learning) kde ku časti vstupných dát existujú výstupné dátá a určitá časť odpovedajúce výstupné dátu nemá. Takáto technika má dve hlavné výhody, prvou je časová náročnosť vytvorenia dostatočne veľkej databázy a odpovedajúcim vystupným dátam. Zásadnejším faktom je, že pri označených vstupných dátach používateľom (učiteľom) môže byť klasifikátor prostredníctvom týchto dát ovplyvnený ľudským vnímaním daného objektu. Druhou kategóriou sú klasifikátory učiace sa bez učiteľa, teda bez vystupnej informácie k daným vstupným dátam. Takéto algoritmy sa snažia hľadať podobnosti a blízkosť medzi jednotlivými vstupnými dátami a tvoriť z nich takzvané zhluky, ktoré kategorizujú dátu do tried. V DAVIS súťaži sú momentálne 3 najlepšími technikami pre techniky čiastočne sa učiace s učiteľom PReMVOS [22], OSVOS-S [23] (vylepšenie techniky OSVOS), OnAVOS [34], a 3 najlepšie techniky učiace sa bez učiteľa MotAdapt [30], PDB [31], ARP [16].

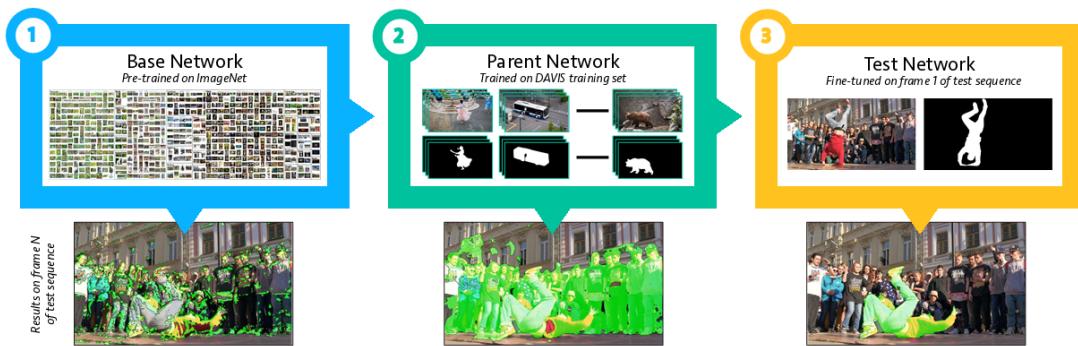
### 2.2.1 OSVOS: One-Shot Video Object Segmentation

One-shot Video Object Segmentation (OSVOS) je konvolučná neurónová sieť, ktorá rieši problém segmentácie objektu z videa, teda klasifikácie každého pixelu každej snímky videa do masky pozadia alebo popredia. Technika patrí do kategórie čiastočného učenia sa s učiteľom. Vstupom je maska jednej alebo viacerých snímkov videa. Na obrázku 2.3 môžeme vidieť príklad vstupných a výstupných dát OSVOS techniky.



Obr. 2.3: Vstupné a výstupné dátá pre techniku OSVOS. Segmentácia objektu na prvom snímku (červená maska) je použitá na naučenie modelu pre daný objekt, ktorý je segmentovaný na ostatných snímkach nezávisle (zelená maska), na každej 20 snímke z celkových 90 snímkov videa [7].

Predstavme si že chceme segmentovať objekt vo videu a jediná dostupná informácia je jedna snímka s maskou objektu. Človek sa pozrie na snímku, analyzuje daný objekt - vytvorí si jeho model a na ďalších snímkach objekt hľadá. Pre ľudí je takáto úloha jednoduchá a ľahko dokážu nájsť objekt vo videu veľmi presne aj ak je ovplyvnený rôznymi degradáciami obrazu. Ľudský mozog sa najskôr naučí aký objekt je na vstupnom obrazze a následne na ostatných obrazoch porovná naučený objekt s ostatnými a nájde ten najbližší. Týmto prístupom je inšpirovaná aj metóda OSVOS. Techniku popíšem v dvoch častiach. Prvou je adaptácia konvolučnej neurónovej siete na akýkoľvek objekt na jednej snímke a jeho prisluchajúcej maske. Druhou časťou je doladenie CNN na konkrétnom prípade pri jej používaní v praxi. Na obrázku 2.4 môžeme vidieť štruktúru OSVOS metódy.

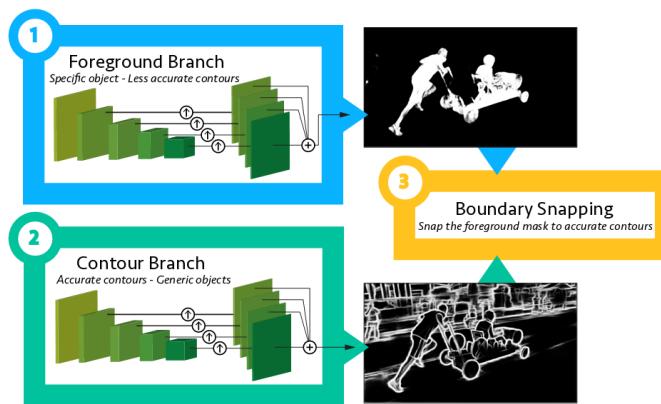


Obr. 2.4: Štruktúra OSVOS siete: 1.) Základná sieť je CNN predtrénovaná na ImageNet Datasets. 2.) Rodičovská sieť vychádza zo základnej a trénovaná na DAVIS datasets. 3.) Testovacia sieť je vrchnou vrstvou, ktorá vďaka nášmu vstupu - segmentácie objektu na jednom snímku videa, rapídne zvýši presnosť segmentovania objektu vo videu [7].

Základná sieť architektúry je natrénovaná na ImageNet datasete pre rozpoznávanie obrazcov. Rodičovská sieť je natrénovaná na vytvorených binárnych maskách objektov na DAVIS datasete k naučeniu generickej informácie ako segmenovať objekty, naučiť sa ich tvary atď. Použitý je Stochastic Gradientny zostup (SGD) s momentom 0.9 pre 50 000 iterácií. Dáta boli augmentované pomocou zrkadlenia o priblženia. Uroveň učenia (ang. learning rate) je nastavená na  $10^{-8}$ , a je postupne znižovaná. Na túto sieť autori článku odkazujú ako na neprepojený (ang. offline training) tréning, pretože sieť bola raz natrénovaná na segmentovanie objektu zo snímku, a už sa nebude pretrénovať pri testovaní. Posledná vrstva je autormi článku uvádzaná ako testovacia sieť. Táto sieť má najst objekt na ostatných snímkach videa, na základe aspoň jednej snímky a príslušnej masky k nej. V tejto vrstve sa pokračuje s finálnym tréningom (ang. fine-tuning) ktorý by sme mohli označiť aj ako lepšie naftovanie modelu z rodicovskej siete na konkrétnu video a objekt. Na tento tréning autori odkazujú ako na prepojený (ang. online training) tréning kedže je prítomný pri každom použití siete [7].

### Detekcia kontúr

V oblasti rozpoznávania objektov, kde bola aj základná sieť vytvorená nie je dôležitá priestorová presnosť riešenia, ale len rozhodnutie či sa objekt nachádza alebo nenachádza na obraze, čo je v kontraste s presnou lokalizáciou objektu ktorú očakávame v segmentácii objektu z videa. Hľadanie kontúr objektu na obraze vylepšuje nasledovná technika. Technika FBS (ang. Fast Bilateral Solver) [5] zachytí hrany ktoré patria pravdepodobne do pozadia, pomocou Gaussovo vyhladenia v 5D priestore obsahujúcim informáciu o farbe (3 dimenzie) + lokalizáciu na obráze (2 dimenzie). Vylepšenie spočíva vo vložení rozpoznaného pozadia do oblasti kontúr a použitia komplementárnej konvolučnej neurónovej siete v ďalšej vetve, ktorá sa bude učiť detektovať kontúry objektu. Túto architektúru môžeme vidieť na obrázku 2.5 [7].



Obr. 2.5: Hlavná vetva detekcie popredia (1) je doplnená vetvou detekcie kontúr objektu (2), čo vylepšuje lokalizáciu objektu [7].

Measure	Ours	-BS	-PN-BS	-OS-BS	-PN-OS-BS				
$\mathcal{J}$	Mean $\mathcal{M} \uparrow$ <b>79.8</b>	77.4	<b>2.4</b>	64.6	<b>15.2</b>	52.5	<b>27.3</b>	17.6	<b>62.2</b>
	Recall $\mathcal{O} \uparrow$ <b>93.6</b>	91.0	<b>2.6</b>	70.5	<b>23.2</b>	57.7	<b>35.9</b>	2.3	<b>91.3</b>
	Decay $\mathcal{D} \downarrow$ 14.9	17.4	<b>2.5</b>	27.8	<b>13.0</b>	<b>-1.9</b>	<b>16.7</b>	1.8	<b>13.1</b>
$\mathcal{F}$	Mean $\mathcal{M} \uparrow$ <b>80.6</b>	78.1	<b>2.5</b>	66.7	<b>13.9</b>	47.7	<b>32.9</b>	20.3	<b>60.4</b>
	Recall $\mathcal{O} \uparrow$ <b>92.6</b>	92.0	<b>0.6</b>	74.4	<b>18.3</b>	47.9	<b>44.7</b>	2.4	<b>90.2</b>
	Decay $\mathcal{D} \downarrow$ 15.0	19.4	<b>4.5</b>	26.4	<b>11.4</b>	<b>0.6</b>	<b>14.3</b>	2.4	<b>12.6</b>
$\mathcal{T}$	Mean $\mathcal{M} \downarrow$ 37.6	<b>33.5</b>	<b>4.0</b>	60.9	<b>23.3</b>	53.8	<b>16.2</b>	46.0	<b>8.4</b>

Obr. 2.6: Tabuľka demonštrujúca dôležitosť všetkých troch zložiek OSVOS metódy: predtrénovanej rodičovskej siete, trénovania na testovacej sieti a detekcie kontúr [7].

V tabuľke 2.6 môžeme vidieť že všetky tri vyššie uvedené časti OSVOS architektúry hrajú významnú rolu v presnosti OSVOS techniky. V tabuľke sú vyhodnotené 3 rôzne merania: presnosť masky vzhľadom na správny počet zaradených pixelov ( $J$ ), presnosť kontúr ( $F$ ) a časová stabilita masiek ( $T$ ). OSVOS so všetkými tromi technikami (Ours),

OSVOS bez detekcie kontúr (-BS), OSVOS bez predtrénovanej rodičovskej siete a detekcie kontúr (-PN-BS), OSVOS bez trénovania na testovacej sieti a detekcie kontúr (-OS-BS) a v poslednom stlpci je presnosť základnej siete osvos architektúry. Malým písmom je pri jednotlivých hodnotách zvýraznené (modrou farbou) zníženie alebo (červenou farbou) nárast oproti technike OSVOS so všetkými blokmi (Ours). Výsledky sú vyhodnotené na validačnom datasete DAVIS. Vídime že predtrénovanie rodičovskej siete na všeobecný model, a trénovanie v testovacej sieti, pre lepšie napasovanie modelu na konkrétny objekt, hrá významnú rolu v OSVOS metóde, bez nich strácame 15,2 a 27,3 bodov v ( $J$ ). Detekcia kontúr pridala 2,4 bodu.

## 2.3 Rozmazanie pohybom

### 2.3.1 Charakteristika efektu rozmazania pohybom

Veľká časť dnešných digitálnych fotiek je zasiahnutá rozostrením. Budť sa jedná o rozostrenie spôsobené nízkou hĺbkou ostrosti (ang. Out of Focus Blur) alebo relatívnym pohybom objektov v scéne alebo kamery počas doby snímania, čo spôsobuje rozmazanie pohybom (ang. Motion Blur). Výsledkom relatívneho pohybu medzi kamerou a objektom snímania je dopad svetla z jedného bodu v scéne na niekoľko susedných bodov na snímke. Intenzitu svetla na obrázku z tohto bodu zo scény zdieľajú susedné pixely snímky vzhľadom na relatívnu dobu snímania a pohyb objektu. Takéto rozmazané miesta definujú PSF funkciu pre rozmazaný obrázok [38]. Zodpovedajúce rozmazanie pohybu je zvyčajne modelované ako lineárna degradácia obrazu.

$$I = L \otimes f + n, \quad (1) \quad (2.11)$$

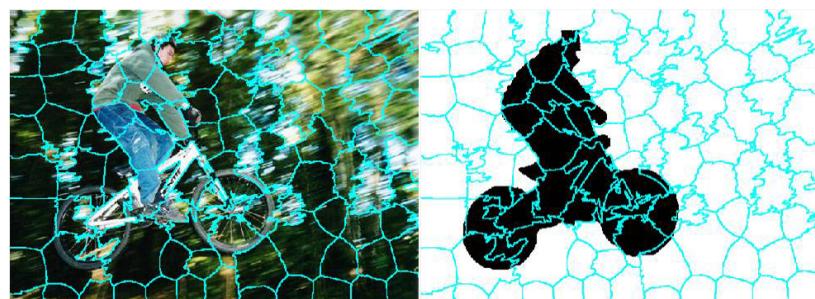
kde  $I$ ,  $L$  a  $n$  predstavujú degradovaný obraz, nerozmazaný (alebo latentný) obraz a šum,  $\otimes$  je operátor konvolúcie a  $f$  je neznáma lineárna funkcia PSF (Point spread function). Konvenčné prístupy pomocou slepej dekonvolúcie sú zamerané na odhad  $f$  pre obnovu  $I$  pomocou intenzity obrazu alebo gradientov. Problém obnovenia statického záberu obsahujúceho rozmazaný objekt nie je možné vyriešiť úplne pomocou slepej dekonvolučnej techniky, pretože pozadie nemusí podliehať rovnakému pohybu. PSF funkcia má jednotnú definíciu iba na pohyblivom objekte. Raskar a kol[1] navrhli riešenie ktoré sme si predstavili ako poslednu techniku v predchádzajúcej kapitole. Riešenie tohto problému existuje, za predpokladu že pozadie je známe [14].

Analýza efektu rozmazania pohybom je vrámci detekcie tohto problému veľmi dôležitá. Z hľadiska pohybu ktorým sa objekty spôsobujúce takúto degradáciu pohybujú môžeme rozmazanie pohybom rozdeľovať na lineárne a nelineárne. Nelineárne môže byť radiálne alebo rozostrenie priblížením alebo vzdialenosťou. Vo videu môže byť z pohľadu

pohybu objektu počas snímania rozmazanie pohybom homogénne (nemenné) alebo heterogénne (meniace sa v priebehu času). Z hľadiska lokalizácie môže byť rozmazanie globálne (ak sa otáča alebo posúva kamera) alebo lokálne pre objekty v scéne.

### Detekcia Rozmazania pohybom

V rámci skúmania techník detektie rozmazania pohybom som navrhol jednoduchú aplikáciu v prostredí Matlab, detekujúcu oblasti rozmazania pohybom na obrázku pomocou exrtahovania príznakov popisujúcich daný efekt a natrénovania jednoduchého klasifikátora na vyhodnotenie ich zaradenia. Použil som už spomínaný dataset zverejnený oddelením informatiky a počítačového inžinierstva Čínskej univerzity v Hong Kongu pozostávajúci s 300 obrázkov s rozmazaním spôsobeným pohybom a im prislúchajúcim maskám určujúcim ktorá časť obrázku je ostrá a ktorá rozmazaná. Obrazok bolo potrebné vhodne rozčleniť na menšie oblasti podobných vizuálne blízkych pixelov. Použil som rozdelenie obrázka na približne 100 superpixelov. Superpixely by sme mohli pomenovať aj ako nadpixely, teda oblasti/regióny zoskupujúce určité pixely ktoré sú si blízke, vizuálne podobné. Na rovnakých 100 superpixelov som rozčlenil aj výstupnú masku k obrázku. Získali sme tak dataset 30 000 superpixelov, ku ktorým prísluší konkrétné masky určujúce či je región rozmazaný alebo ostrý. Na obrázku 2.7 môžeme vidieť toto rozdelenie na superpixely.



Obr. 2.7: Vstupný obrázok (vľavo) som rozdelil na 100 superpixelov, pomocou funkcie `superpixels()` v prostredí Matlab, ktorá obsahuje niekoľko parametrov ako napríklad compactness kompaktnosť, Method= `slic( Simple Linear Iterative Clustering)`, NumIterations=100 (počet superpixelov na ktoré chcem obraz rozčleniť). Na obrázku vpravo som rozčlenil aj masku obrázku na základe členenia superpixelov na vstupnom obrazu.

Zo superpixelov sme extrahovali nasledovných 20 príznakov.

- Histogramové príznaky (*Mean, Variance, Skewness, Kurtosis, Energy, Entropy*) sme získali pomocou `chip_histogram_features.m` funkcie.
- Haralick-ové textúrne príznaky (*Angular Second Moment (Energy), Contrast, Correlation, Variance, Inverse Difference Moment, Sum Average, Sum Variance,*

*Sum Entropy, Entropy, Difference Variance, Difference Entropy, Information Measure of Correlation I, Information Measure of Correlation II, Maximal Correlation Coefficient) pomocou haralicksTextureFeatures.m funkcie.*

Vykonali sme normalizáciu dát a použili sme PCA Principal Komponent Analysis algoritmus na zníženie dimenzionality algoritmu. Dataset sme rozdelili na 80:10:10 pre trénovaciu, validačnú a testovaciu množinu. A natrénovali sme KNN, nelineárny SVM a lineárny klasifikátor. Najlepšiu presnosť sme však dosiahli len 64,7 percenta s lineárnym klasifikátorom aj po rôznych úpravách hyperparametrov klasifikátorov. Avšak, pre niektoré obrázky klasifikátor vyhodnooval popredie od pozadia pomerne presne. Na nasledujúcom obrázku môžeme vidieť obrázky ktorých klasifikácia je podstatne vyššia ako 64% aj obrázky so zlým zaradením.



Obr. 2.8: Na obrázku zľava: prvý stĺpec sú vstupné obrázky s vykreslenými superpixelmi. Druhý stĺpec sú vysledné masky vytvorené klasifikátorami. Stĺpec vpravo sú presné masky oblasti rozmažania (biela farba) a ostrosti (čierna farba). Na obrázku môžeme vidieť že na prvých troch snímkach pomerne dobre kategorizoval lineárny klasifikátor superpixely. Na poslednom obrázku demonštrujem ako veľmi sa na niektorých obrázkoch naopak mylil.

Na tejto jednoduchej úlohe som mal možnosť nahliadnuť do techník založených na rozpoznávaní obrazcov. Uvedomil som si že použitie klasifikátorov obmení zadanie a vytvorí nové podproblémy ktoré musíme riešiť. Prvým je správne rozdelenie priestoru

na oblasti ktorých hranice dobre kopírujú hrany objektov na obrázku. Rozdelenie pomocou superpixelov nie je najoptimálnejšie nakoľko aj na obrázku 2.7 môžeme nájsť superpixely ktoré sú súčasne aj v rozmazanej aj v ostrej oblasti. Druhým problémom bolo zlé rozdelenie datasetu na trénovaciu, testovaciu a validačnú množinu. Keď som túto aplikáciu navrhoval ešte som mal malú znalosť metód založených na rozpoznávaní obrazcov a mylne som predpokladal že použitím čo najväčšej testovacej množiny zaručím čo najväčšie natrénovanie klasifikátora. Ďalším pravdepodobne najdôležitým faktorom vpäťvajúcim na výsledok sú dobré príznaky popisujúce rozmazanie pohybom. Článok [2] ktorý som opisoval aj v úvodnej kapitole, vyšiel v roku 2018 až potom, ako som vytvoril danú aplikáciu. Podľa článku som si ale overil, že niektoré príznaky som použil veľmi správne, čo je jedným z dôvodov prečo mi niektoré obrázky klasifikovalo správne. Dôvod prečo klasifikátory na niektorých obrázkoch dobre rozpoznávajú rozmazanú časť od rozostrenej, by mohol byť taký, že ja som daný dataset rozdelil do troch množín (trenovacia, validačná, testovacia) podľa obrázkov, nie podľa superpixelov. To znamená že nejaký obrázok celý s jeho 100 superpixelami patril do trénovacej množiny a iný obrázok s jeho 100 superpixelami patril do testovacej množiny. Obrázky sa od seba dosť odlišujú a tak sa klasifikátor mohol naučiť segmentovať objekty len na určitý druh obrázkov. Tieto problémy sú ale implementačné, ktoré by som mohol odstrániť. Problém ktorý by som ale asi nevedel vyriešiť je ten, že v práci sa zameriavam na segmentáciu jedného konkrétneho objektu ovplyvneného rozmazaním pohybom. Problém ktorý by som ešte musel vyriešiť je rozpoznávanie len takého rozmazania ktoré je spôsobené objektom.

### **Od Rozmazania pohybom k Alpha Matting**

Matting si kladie za cieľ extrahovať objekt z obrazu tak aby započítaval zlomkové hodnoty priehľadnosti pixlov popredia. Vo väčšine prác zaobrajúcich sa touto tému, sa predpokladá že objekt v popredí je statický a čiastočná priehľadnosť vzniká z len čiastočného pokrycia pixlu objektom popredia, takže farba pixlu je vytvorená ako kombinácia popredia a pozadia. Príkladom môže byť prameň vlasov, ktorý je často príliš tenký aby úplne pokryl pixel a tak je farba pixelu zložením farby prameňa vlasov a pozadia (viď obrázok 2.9 vľavo). Ďalším združením čiastočnej priehľadnosti je rozmazanie objektu spôsobené pohybom. V tomto prípade je pixel obsadený objektom popredia len časť expozičného času, a pozadím po zvyšok expozičného času. (viď obrázok 2.9 vpravo [25]).



Obr. 2.9: Na obrázku (vľavo) môžeme vidieť typický príklad zadania pre Alpha Matting metódy - Segmentáciu vlasov. Na obrázku (vpravo) môžeme vidieť typický príklad obrázka s objektom rozmazaným pohybom.

### 2.3.2 Alpha Matting

Matting si kladie za cieľ extrahovať objekt z obrazu tak aby započítaval zlomkové hodnoty priehľadnosti pixlov popredia. Pre podrobnú analýzu vývoja matting metód pozri [36].

Alpha matting odkazuje na problém rozdelenia obrázku na popredie a pozadie, čo je konvexnou kombináciou na obraze:

$$I_z = \alpha_z * F_z + (1 - \alpha_z) * B_z, \quad (2.12)$$

kde  $I$  je daná farba pixlu,  $F$  je neznáme popredie,  $B$  je neznáme pozadie, a  $\alpha$  je neznáma hodnota priehľadnosti. Pre daný obraz, všetky 3 hodnoty,  $\alpha$ ,  $F$  a  $B$  sú neznáme a je nutné ich vyriešiť pre každý pixel. Známa informácia je 3D farebný vektor pixelu  $I_z$  (predpokladajúc že pracujeme s obrazom s 3D farebným priestorom) a neznáme premenné sú 3D farebné vektory  $F_z$  a  $B_z$  a skalárna alpha hodnota  $\alpha_z$ . Matting je teda nedostatočne definovaný problém, nakoľko 7 neznámych hodnot má byť vyriešených z 3 známych hodnôt. Väčšina matting prístupov sa spolieha na vstup od používateľa. Alpha Matting metódy pracujú s Trimapom ( $T_U, T_F, T_B$ ) kde  $T_U$  je neznáma časť,  $T_F$  je určite popredie a  $T_B$  je určite pozadie [36].

Najviac skúmané sú dva prístupy k problematike. Prvými sú metódy zameriavajúce sa na farebnú informáciu (ang. Color Sampling methods). Tieto metódy pre daný pixel hľadajú v blízkych oblastiach popredia a pozadia vzorky ktoré sú priemerujú a porovnávajú ich blízkosť s daným pixelom. Druhou kategóriou sú metódy definujúce iné príbuznosti pixelov pre Matting (ang. Affinities for Matting methods). Medzi metódy zameriavajúce sa na farebnú informáciu patria Ruzon and Thomasi Metóda [29], Bayesian Matting [9], Mishima Metóda [24], Knockout metóda, Robust Matting [35]. Do kategórie metód zameriavajúcich sa na nachádzanie príbuzností pixelov patria, Poisson

Matting [33], Random Walk Matting [12], Geodesic Matting, [3], Closed-form Matting [18], Spectral Matting [19].

### 2.3.3 KNN Matting

KNN matting vychádza z nelokálneho princípu [17], čo znamená že pixely blízke svojou hodnotou k danému pixlu nemusia byť v jeho blízkom okolí. Problém segmentácie jednotlivých vrstiev obrázka prevedieme do teórie grafov, a na hľadanie dobrého rozdelenia tohto grafu na komponenty predstavujúce jednotlivé vrstvy. Metóda používa nelokálny princíp na nájdenie príbuzností, podobností pixelov, tak aby boli vytvorené dobré grafové komponenty predstavujúce vrstvy na ktoré chceme obraz rozdeľovať.

#### Matica príbuznosti, Matica stupňa, a Matica Laplaciánu

**Matica príbuznosti**  $A$  (ang. Affinity Matrix) alebo v grafovej terminológii nazývaná aj matica susedností (ang. Adjacency Matrix) je matica veľkosti  $N \times N$ , kde  $N$  je počet pixelov obrázka, a jej hodnota je vypočítavaná ako  $A = [k(i, j)]$ ,  $i, j \in N$ . Takže  $A_{i,j} \simeq 1$  ak sú si body veľmi blízke, a  $A_{i,j} \rightarrow 0$  ak sú body vzdialené. Cieľom je transformovať priestor tak aby, ak sú dva body blízke, nachádzali sa v rovnakom komponente a v rovnakej vrstve. Ak sú dva body odlišné, nachádzali sa v rôznych komponentoch grafu a teda aj rôznych vrstvách obrázka.

**Matica stupňa**  $D$  (ang. Degree Matrix) je v teórii grafov diagnálna matica ktorá obsahuje informáciu o “stupni” každého vrcholu, teda počte susedností s inými vrcholmi. Z definície:

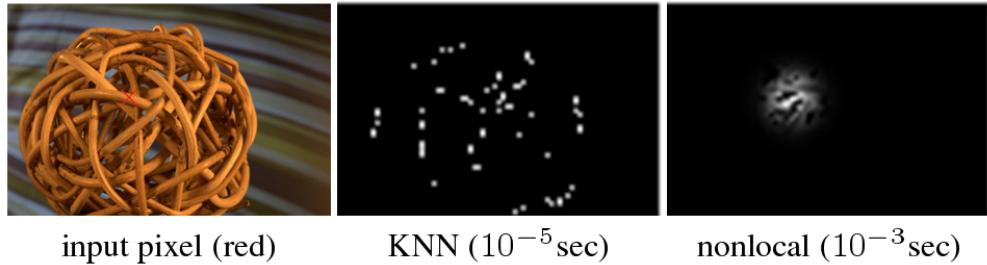
$$d_{i,j} := \begin{cases} \deg(v_i), & \text{ak } i = j \\ 0, & \text{inak} \end{cases} \quad (2.13)$$

kde stupeň  $\deg(v_i)$  je stupeň vrcholu.

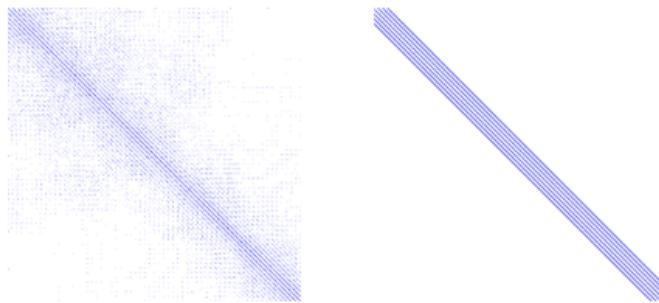
**Laplaciánová matica**  $L$  (ang. Laplacian Matrix) je matica reprezentujúca graf, ktorá sa používa k vyriešeniu rôznych vlastností grafu. V našom prípade k aproximácií riešenia. Z definície:

$$L = D - A \quad (2.14)$$

kde  $D$  je Matica stupňa a  $A$  je matica príbuzností. Na obrázku 2.10 môžeme vidieť rozdiel v nájdení nelokálnych susedov pixelu pomocou KNN metódy a pomocou nonlocal matting metódy. Na obrázku 2.11 môžeme vidieť matice príbuzností týchto metód [17].



Obr. 2.10: Porovnanie blízkych pixelov nájdené pomocou metódy KNN a metódy non-local matting. Nonlocal Matting používa okno so stredom v danom pixely pre hľadanie nelokálnych susedov (polomer 9). KNN matting zbiera susedov z celého obrázku, a zároveň má podstatne nižší výpočtový čas. ( $K = 81$  susedov). Sprava, vstupný obraz z označeným pixelem červeným krížikom, KNN matting pre 81 susedov, nonlocal matting s polomerom 9 [8].



Obr. 2.11: Matice pribuznosti (vľavo, s  $K=10$ ), ktorá nieje tak silne diagonálna ako jej naprotivná vypočítaná pomocou metódy nonlocal matting(vpravo, s polomerom 3) [8].

Príznakový vektor  $X(i)$  na pixely  $i$  je definovaný ako

$$X(i) = (\cos(h), \sin(h), s, v, x, y)_i \quad (2.15)$$

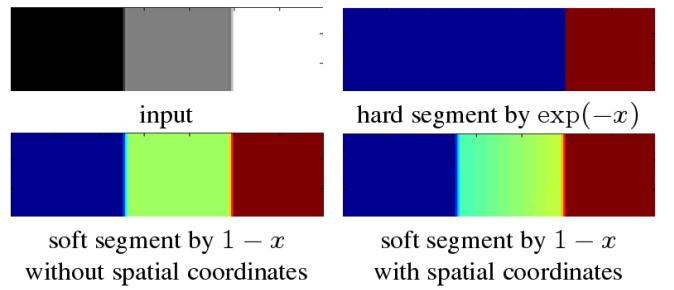
kde  $h$ ,  $s$ ,  $v$  su príslušné súradnice HSV farebného priestoru a  $(x, y)$  sú priestorové súradnice pixelu  $i$ . Príznakový vektor je ľahko rozšíriteľný aj do iného farebného priestoru. Vo vektorovom priestore nájdeme  $K$ -najbližších susedov pre daný pixel (v našom prípade 10).

Funkcia jadra:

$$k_{(i,j)} = 1 - \frac{\| X_{(i)} - X_{(j)} \|}{C} \quad (2.16)$$

sa používa na reprezentovanie afinity medzi dvoma pixlami, kde  $C$  je supremum (ang. least upper bound)  $\| X_{(i)} - X_{(j)} \|$ , aby sme hodnoty  $k_{(i,j)}$  normalizovali na  $[0,1]$ . Funkcia jadra ?? kladie rovnaký dôraz na celý interval  $[0,1]$  a predchádza tak vychyleniu k poprediu alebo pozadiu. Typický výber funkcie jadra v strojom učení býva,  $\exp(-x)$

a bol použitý aj v [17]. V [8] článku autori tvrdia že to nie je dobrý výber pre jemnú segmentáciu, práve naopak, vytvára ostru segmentáciu. Na obrázku 2.12 môžeme vidieť príklad segmentácie, kde sú tri vrstvy vďaka čiastočnej priehľadnosti. Na obrázku je KNN matting algoritmus s funkciou jadra  $\exp(-x)$ . Ako vidíme na obrázku táto funkcia produkuje ostrú segmentáciu. Ostrá segmentácia môže byť spôsobená ne-maximálnym potlačením Gausovského jadra, ktoré časť ne-popredia silne penalizuje [8].



Obr. 2.12: Hodnota  $\exp(-x)$  má sklon generovať ostrú segmentáciu, pretože jadro je znižované exponenciálne, vzhladom na farebné rozdiely. Kontrastne,  $1 - x$  hodnota bez priestorových súradníc, môže tvoriť jemnú segmentáciu bližšiu k ground truth). Navyše, použitím  $1-x$  hodnoty s priestorovými súradnicami, vytvárame alpha matting s jemnejšími prechodmi medzi susednými pixelmi. [8].

### Uplné riešenie:

Kým clustering Laplacian  $L_c = (D - A)^T(D - A)$  vedie k dobrému rozdeleniu grafu do blokov, Laplacian  $L=D-A$  je redšia matica ktorej riešenie je dosiahnutelné rýchlejšie (až 100 krát rýchlejšie v porovnaní s  $L_c$ ), bez znižovania kvality výsledkov, za podmienky že zabezpečíme dostatočné vstupy od používateľa. Toto môžeme považovať za kompromis medzi výpočtovým časom, množstvom vstupov od používateľa a kvality výsledkov [8]. Ak máme vstup od používateľa vo forme trimapu, úplne riešenie pre rozdelenie  $n = 2$  a viac vrstiev je:

$$(L + \lambda D) \sum_i^n \alpha_i = \lambda m \quad (2.17)$$

kde  $D = diag(m)$  je diagonálna matica binárneho vektora indexov  $m$  všetkých označených pixelov.  $\lambda$  je konšanta určujúca dôveryhodnosť vstupov od používateľa. Optimalizačná funkcia  $g(x)$  má uplné riešenie:

$$g(x) = x^T L x + \lambda \sum_{i \in m-v} x_i^2 + \lambda \sum_{i \in v} (x_i - 1)^2 \quad (2.18)$$

kde  $v$  je binárny vektor indexov označených pixelov pre danú vrstvu. [8] Ďalej  $g(x)$

môžeme upraviť:

$$x^T L x + \lambda \sum_{i \in m-v} x_i^2 + \lambda \sum_{i \in v} x_i^2 - 2\lambda v^T x + \lambda |v| \quad (2.19)$$

$$= x^T L x + \lambda \sum_{i \in m} x_i^2 - 2\lambda v^T x + \lambda |v| \quad (2.20)$$

$$= \frac{1}{2} x^T 2(L + \lambda D) x - 2\lambda v^T x + \lambda |v| \quad (2.21)$$

$$= \frac{1}{2} x^T H x - c^T x + \lambda |v| \quad (2.22)$$

kde  $\lambda |v|$  je konštanta. Všimnime si že  $H = 2(L + \lambda D)$  je pozitívna a semi-definitná pretože aj  $L$  je taká a  $D$  je diagonálna matica vytvorená binárnym vektorom  $m$ . [8] Parciálna derivácia podľa  $x$ :

$$\frac{\partial g}{\partial x} = Hx - c = 0 \quad (2.23)$$

z čoho optimálne riešenie vyvodíme ako:

$$H^{-1}c = (L + \lambda D)^{-1}(\lambda v). \quad (2.24)$$

Predpokladajme zjednodušený model, že grafové komponenty sú dobre identifikované, potom Laplácian  $L$  je približne blokovo-diagonálny, s každým blokom definujúcim jeden komponent. Ak máme 3 hlavné bloky ( $C_1, C_2, C_3$ ), kde  $L_{c1,c1}$  je podblok komponentu  $C_1$ , atď. Dá sa predpokladať že 3 najmenšie vlastné čísla a vlastné vektory  $(\lambda_i, v_i)$  Laplacianu  $L$ , každé zodpovedá inému komponentu. Čo nastane ak platí:  $Lv_i = \lambda_i v$  sú najnižší pár vlastných čísel a vlastných vektorov v celom priestore a  $L_{c1,c1}v_{c1} = \lambda_{c1}v_{c1}$  je najnižší par vlastného čisla, vlastného vektoru pre blok  $C_1$ , potom z tohto predpokladu množina troch najnižších vlastných čísel je množinou komponentov.

# Kapitola 3

## Implementácia

Táto kapitola opisuje proces vytvorenia aplikácie. V kapitole sú obsiahnuté softvérové a hardvérové požiadavky, moduly integrované do aplikácie, implementácia jednotlivých častí a návod na používanie aplikácie.

### 3.1 Softvérové požiadavky

- **Operačný systém:** Aplikáciu som vytvoril a testoval vo Windows 7 Professional (64 bit) a Linux Ubuntu 18.10.
- **Python 3.6:** Pri výbere programovacieho jazyka pre vytvorenie aplikácie sme uvažovali tri možnosti. Matlab z dôvodu Image Processing Toolbox knižníc a veľkej podpory knižníc pre spracovanie obrazu a videa. Alternatívou bola OpenCV knižnica v kombinácii s C++ alebo Pythonom. Nakoniec som sa rozhodol pre Python z dôvodu, že neurónová sieť OSVOS je implementovaná v jazyku Python. Rozhodovanie bolo aj na úrovni verzie Python jazyka. Do úvahy pripadala verzia 2.7 alebo 3.6. Pre obidve verzie sa zdali byť knižnice, ktoré som chcel používať v aplikácii dostupné. Rozhodol som sa pre python 3.6
- **Anaconda 3:** Je voľne šíritelná multiplatformová distribúcia pre jazyky Python a R so zameraním na vedecké výpočty v oblastiach strojového učenia, data science, prediktívnej analytiky a ďalších, ktorá sa zameriava na správu knižníc, balíkov a ich inštaláciu a integrovanie do programovacieho jazyka. S inštaláciou Anaconda 3 sa automaticky inštaluje aj Python 3.7. Pre fungovanie našej aplikácie je potrebné preinštalovať python na verziu 3.6.
- **Tensorflow:** Je voľne šíritelná knižnica, používaná najmä na strojové učenie a tvorenie a ovládanie neurónových sietí. Je vyvinutá firmou Google a pôvodne mala slúžiť len pre internú prácu vo firme Google. Tensorflow môže pracovať na

niekoľkých CPU alebo GPU (s CUDA alebo SYCL rozšíreniami) pre výpočty na GPU.

- **OpenCV 2:** Je voľne šíritelná multiplatformová knižnica pracujúca s obrazom zameraná predovšetkým na počítačové videnie a spracovanie obrazu v reálnom čase. Vytvára ju spoločnosť Intel. Knižnicu je možné využiť v prostredí jazyka C a C++ a s generátorom rozhrania SWIG v jazykoch Python a Octave.
- **Tkinter:** Tkinter je štandardné grafické používateľské rozhranie GUI (Graphical User Interface) pre jazyk Python. Je to objektovo orientovaná nadstavba pre Tcl/Tk. Inštalácia Python jazyka pre Windows obsahuje Tkinter. Pre Linux distribúciu Pythonu je potrebná samostatná inštalácia Tkinter knižnice.
- **PIL:** PIL je Python Imaging Library, knižnica obsahujúca metódy spracovania videa a obrázkov v jazyku Python. Podporuje množstvo obrazových a video formátov. V našej aplikácii bude zohrávať rolu prepojenia OpenCV a GUI rozhrania aplikácie.
- **Numpy, Matplotlib a Scipy:** Numpy je matematická knižnica pre Python obsahujúca výpočty na veľkých multi-dimenziónych poliach a maticách, navrhnutá so zameraním na časovú a pamäťovú efektívnosť výpočtov. Matplotlib je knižnica, rozšírenie Numpy knižnice, ktoré vizualizuje dátu počítané Numpy knižnicou. Matplotlib poskytuje objektovo orientované API k pripojeniu grafov a vizualizácií týchto dát do GUI rozhrania aplikácie. Scipy je voľne šíritelná Python knižnica, rozšírenie Numpy knižnice, obsahujúce funkcie pre optimalizáciu, lineárnu algebru, interpoláciu, spracovanie signálu a spracovanie obrazu.

## 3.2 Dataset

Pre kvalitné vyhodnotenie výsledkov problematiky je vždy dôležitá voľba datasetu, teda vstupných dát a k nim odpovedajúcim očakávaným výstupom. Dataset k mojej problematike ešte neboli vytvorené, existujú ale datasety pre niektoré podproblémy v mojom zadani.

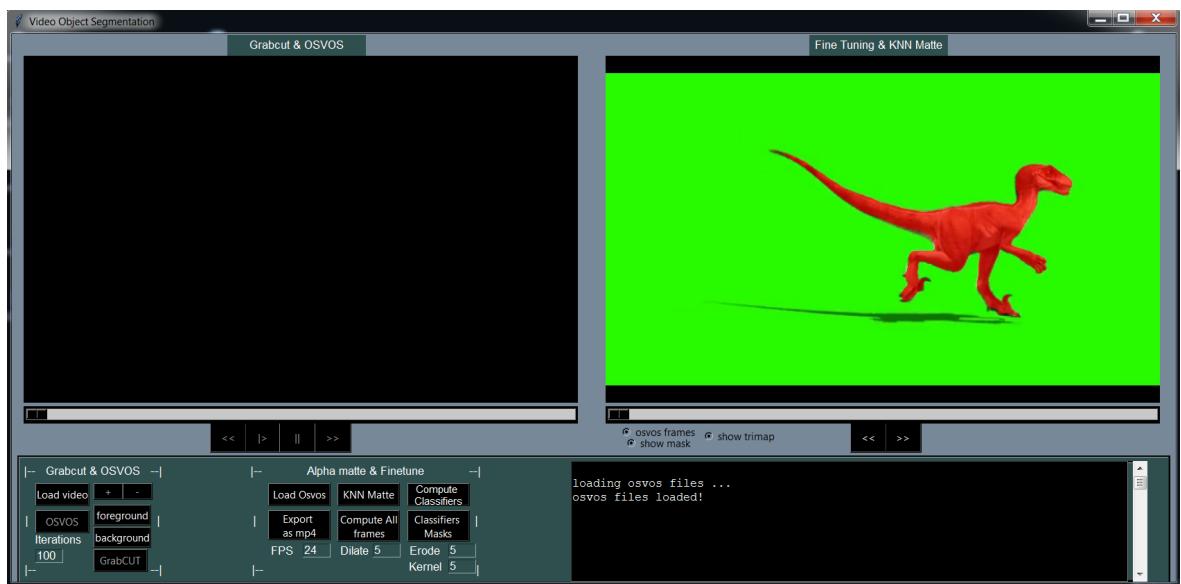
### DAVIS CHALLANGE: Densely Annotated Video Segmentation

DAVIS súťaž v segmentácii objektu z videa sa uskutočňuje od roku 2017, každý rok ako seminár vrámci Konferencie počítačového videnia a rozpoznávania obrazcov (ang. Conference on Computer Vision and Pattern Recognition - CVPR). Dataset obsahuje 50 HD a 50 Full HD video sekvencií, obsahujúcich niekoľko známych problémov a degradácií vznikajúcich vo videách, ako sú oklúzie, rozmazanie pohybom a zmeny vzhľadu. V každom videu je vytvorená presná maska objektu segmentácie pre každú

snímku vo videu. V rámci vyhodnocovania sa merajú 3 hlavné metriky, ktoré merajú priestorový objem segmentácie objektu, presnosť siluety kontúr objektu a časovú stabilitu segmentácie vrámci sekvencie snímok [27].

### 3.3 Grafické rozhranie a funkcia

Pre vytvorenie polo-automatickej segmentácie objektu z videa je potrebné vytvoriť aplikáciu s grafickým rozhraním na jednoduchú manipuláciu s videom, editáciou snímkov a vytvorením masky na danej snímke.



Obr. 3.1: Grafické rozhranie aplikácie.

Aplikácia obsahuje dve hlavné časti. Vo vrchnej časti sa nachádzajú dve prehrávacie okná, ľavé na prehrávanie vstupných videí a segmentovanie masky metódou GrabCut. V pravom okne sa bude pri tvorení masky metódou Grabcut zobrazovať vyznačená maska objektu, okrem toho môžeme v tomto okne prechádzať snímkami videa po OSVOS metóde a testovať a zlepšovať segmentáciu KNN matting metódou. V spodnej časti sa nachádza panel obsahujúci ovladanie aplikácie a vpravo je okno, ktoré nás informuje o spracovávaní dát a jednotlivých nastaveniach. Tlačidlá ovládania sú rozdelené do dvoch blokov. Prvým blokom sú tlačidla vľavo. Sú tu tlačidlá na načítanie videa, segmentáciu Grabcut metódou. Vytvorenie sekvencie snímok pre OSVOS metódu a spustenie OSVOS metódy. Pomocou tlačidla **load video** načítame vstupné video, ktoré sa nám zobrazí v ľavom okne. Na paneli výpisov sa po korektnom načítaní vypíšu informácie o videu (počet snímok za sekundu, rozlíšenie vstupného videa, kódeks videa a ak video už má vytvorenú (aspôň jednu) masku a zároveň má aj video uložené vo forme sekvencie snímok v priečinkoch OSVOS, panel vypíše túto informáciu.) Pomocou tlačidiel [**>**], [**|||**], [**>>**], [**<<**] môžeme video prehrávať a zastaviť na konkrétnej

snímke alebo posúvať snímku po snímke. Počas používania OSVOS som zistil, že limitáciou vstupných dát sú videá maximálne s rozlíšením 720P. Ak je video vo vyššom rozlíšení ako 800x600 pixelov tak sa v aplikácii zmenší do tejto veľkosti, samozrejme so zachovaním pomeru strán. Na 3.1 môžeme vidieť grafické rozhranie aplikácie.

### 3.4 Grabcut metóda

Pre hľadanie najlepšej snímky pre vytvorenie masky popredia, teda objektu, odporúčam nájsť snímku, na ktorej je najmenšie rozmazanie objektu pohybom. A to z dôvodu, že masku tvoríme binárnu a ak segmentujem oblasť, kde je aj rozmazanie pohybom, je dosť pravdepodobné, že na týchto miestach je farba pixelu čiastočne ovplyvnená popredím a čiastočne pozadím, čo môže spôsobiť, že metóda OSVOS sa pre dané farby pozadia a dané pixely bude myliť. Ak je v ľavom okne zvolená snímka, ktorá má mať masku, ľavým tlačidlom myši a pomocou funkcionality Drag&Drop vytvoríme štvoruholník okolo objektu záujmu. Je dôležité, aby celý objekt bol vnútri tohto štvoruholníka. Ak sa to nepodarí na prvýkrát, vyznačenie objektu takýmto štvoruholníkom opakujeme. Následne klikneme na tlačidlo **Grabcut**. Po tomto kroku môžeme iteratívne meniť segmentáciu grabcut metódy klikaním myšou na objekt v ľavom okne aplikácie. Červený kruh značí pozadie a žltý popredie. Tlačidlami **foreground** a **background** prepínam medzi označovaním popredia a pozadia. Následne v okne naľavo môžem vyznačovať kliknutím ľavým tlačidlom myši a tahaním, čo je popredie a čo pozadie. Tlačidlom **Grabcut** znova spustím metódu Grabcut pre dané masky. Metóda je v aplikácii implementovaná pomocou funkcie cv2.Grabcut a je nastavená na 5 iterácií algoritmu iteratívnej minimizácie. Na obrázku 3.2 môžeme vidieť používanie GrabCut algoritmu v aplikácii.



Obr. 3.2: Použitie GrabCut algoritmu v aplikácii.

Vždy pri kliknutí na tlačidlo Grabcut sa uloží daná binárna maska pod číslom snímky, na ktoré ju robím do priečinka OSVOS metódy. Takže ak som so segmentáciou masky spokojný, môžem pokračovať v prechádzaní videa a vytváraní ďalších masiek pre videosekvenciu alebo môžem použiť OSVOS technológiu.

### 3.5 Integrácia OSVOS siete

Pre fungovanie OSVOS siete je potrebné spustiť aplikáciu pod vytvoreným tensorflow prostredím. Postup ako aplikáciu spustiť:

- spustím **cmd.exe** ako správca pre Window, v Linux Ubuntu spustím shell
- prejdem do priečinka s aplikáciou
- Windows: zadám príkaz **tensorflow activate {názov tf prostredia }**
- Linux Ubuntu: zadám príkaz **source tensorflow {názov tf prostredia }**
- spustím aplikáciu pomocou príkazu **python main.py**

OSVOS sieť je integrovaná v aplikácii a spúšťa sa tlačidlom **OSVOS**. Nutnými podmienkami sú vstupy. Video, ktoré v aplikácii otvorím, si aplikácia zapamätá a pri spustení OSVOS technológie očakáva, že video bude uložené v správnom priečinku ako sekvencia obrázkov. Druhou nutnou podmienkou je aspoň jedna maska vytvorená metódou GrabCut alebo importovaná do priečinku s maskami. Ak sme už OSVOS aplikáciu pre video spúšťali, tak vstupy pre OSVOS sú už vygenerované čo nám aplikácia oznamí pri načítaní videa. Pod kolónkou [**Iterations**] je možné nastavovať počet iterácií OSVOS neurónovej siete pre lepšie naučenie sa daného videa.

### 3.5.1 priečinkový systém OSVOS tehchnológie

OSVOS sa nachádza v priečinku /OSVOS/. Vstupné a výstupné dátá sú v priečinku /OSVOS/DAVIS/. Binárne Masky objektov sa nachádzajú v priečinku /ANNOTATIONS/480P/{NÁZOV\_VIDEA}/ a majú názov odpovedajúci číslu snímky vo videu, na ktoré boli vytvorené. Sekvencia snímok videa je v /JPEGIMAGES/480P/{NÁZOV\_VIDEA}/ analogicky pod číselnými názvami uvadzajúcimi poradie snímok. Výsledné binárne masky: /SEGMENTATIONS/480P/OSVOS/{NÁZOV\_VIDEA}/BW\_MASK/ Masky na vstupných snímkach: /DAVIS/RESULTS/SEGMENTATIONS/480P/OSVOS/{NÁZOV\_VIDEA}/

## 3.6 KNN Matting

Tlačidlá na ovládanie KNN Matting metódy sa nachádzajú v bloku pomenovanom Finetuning & Alpha Matting. Tlačidlom **Load OSVOS** načítame sekvenciu snímok z priečinka, kde OSVOS výslednú segmentáciu vyexportoval. Tento priečinok nájdeme pod /OSVOS/DAVIS/Results/Segmentations/480p/OSVOS/{nazov\_videa}. V OpenFileDialog vyberieme len priečinok, v ktorom sa nachádzajú snímky po OSVOS segmentácii. Pomocou tlačidiel [«], [»] sa môžeme v snímkach posúvať. Pod zobrazovacím oknom je možnosť prepínať sa medzi segmentáciou po OSVOS, binárnu maskou prislúchajúcou danému obrázku a trimapom. Vlastnosti trimapu je možné nastavovať v časti Alpha matte & Finetuning pod tlačidlami nastavujeme počet iterácií pre Dilatáciu, Eróziu, a veľkosť kernelu. Morfologický operátor je štvorec. Tlačidlom **[KNN Matte]** je možné spustiť KNN matting metódu pre jednu snímku. Tlačidlom **[Compute all frames]** je možné spustiť KNN Matting pre všetky snímky sekvencie. Snímky sa automaticky ukladajú do podpriečinku, v ktorom sa nachádzajú snímky po OSVOS segmentácii. V aplikácií som testoval vytvárať menšie okná na hranici objektu. Pre tieto okná som vytváral trimap a spúštal na nich KNN Matting metódu. Motiváciou k takýmto lokálnym oknám na vyhodnocovanie segmentácie popredia a pozadia bol nápad vytvárať pre tieto okná Trimap Lokálne podľa veľkosti hrany medzi popredím a pozadím. Z časových dôvodov sa mi, ale takýto adaptívny trimap nepodarilo implementovať do aplikácie. Tlačidlá som v aplikácii ponechal, pretože v budúcnosti by som chcel implementovať túto funkcionality do aplikácie a výsledky segmentácie len častí obrázka sa v niektorých prípadoch zdajú byť sľubné. Tlačidlom **[Classifier Masks]** vykreslíme do okna masky tak, aby pokrývali časť objektu a časť pozadia. Tlačidlom **[Compute Classifiers]** necháme počítať KNN Matting pre všetky snímky na týchto lokálnych oknách.

# Kapitola 4

## Výsledky

V tejto kapitole popíšem a vyhodnotím výsledky aplikácie a metód použitých v nej. Kapitolu delím do dvoch podkapitol. V prvej časti zhodnotím silné a slabé stránky našej aplikácie. V akých aspektoch môže byť prínosná a čo sú jej limitácie. V druhej kapitole porovnáme presnosť našej segmentácie objektu s OSVOS a KNN Matting metódach so segmentáciou OSVOS metód na náhodne vybraných snímkach z datasetu a vyhodnotíme výsledky presnosti.

### 4.1 Vyhodnotenie

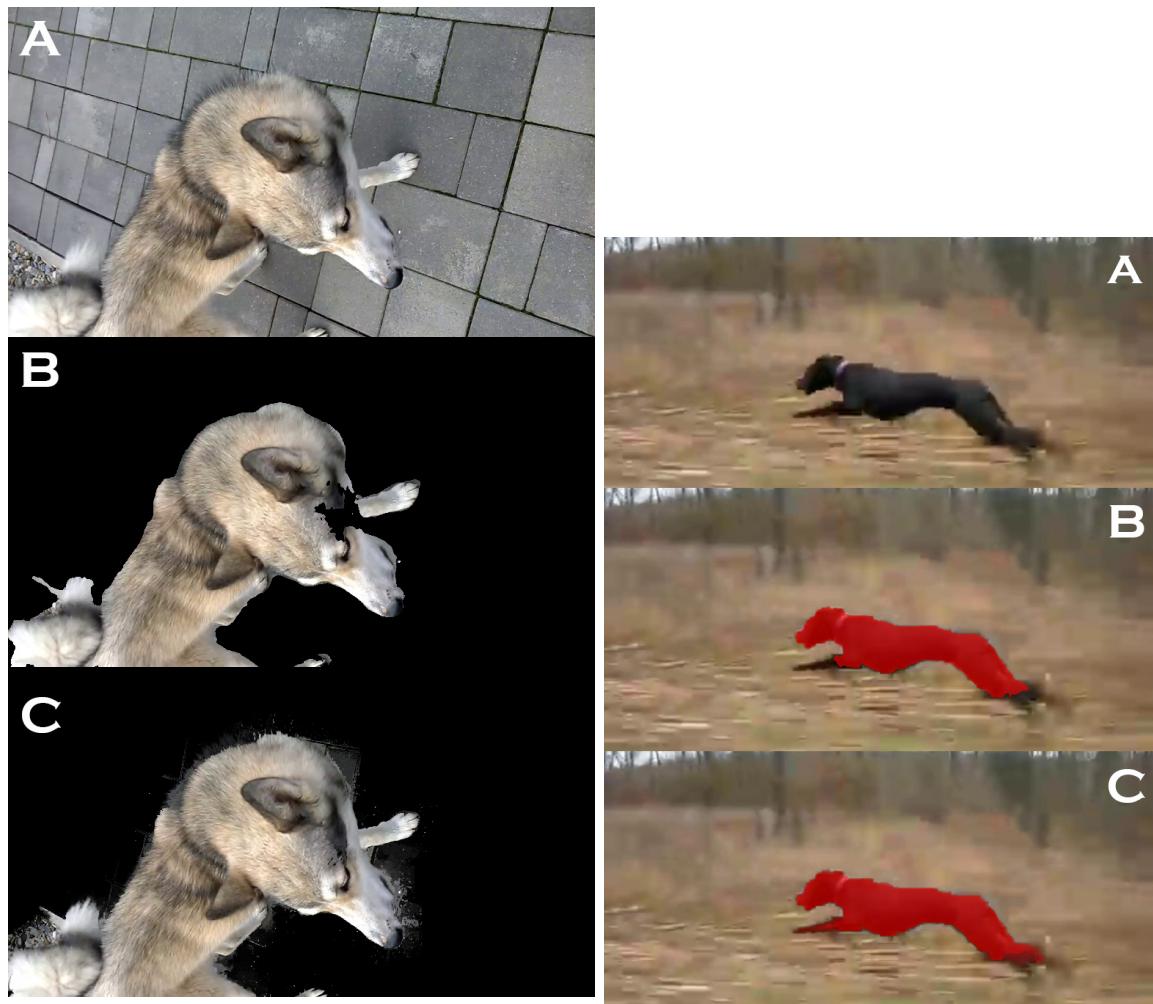
V nasledujúcim bloku popíšem 3 nepresnosti, ktoré OSVOS metóda produkuje a ktoré vie KNN Matting metóda odstraniť, a teda zvyšuje presnosť segmentácie.

#### 4.1.1 Zvýšenie presnosti segmentácie objektu

Počas testovania OSVOS siete som si všimol, že pre snímky, kde nastáva výrazné rozmažanie objektu pohybom má OSVOS metóda problém správne segmentovať objekt. Na obrázku 4.1 môžeme vidieť sekvenčiu snímkov, na ktorej je práve stredná snímka zle segmentovaná. Na snímke vidíme nohy bežiaceho psíka, ktoré niesú segmentované ako popredie. Na obrázku 4.2 môžeme vidieť túto snímku ako pôvodnú (a), po OSVOS segmentácii (b) a po KNN Matting segmentácií (c). Na obrázku 4.2a vidíme, že OSVOS metóda zle segmentovala časť hlavy psa. Takéto diery v segmentácii sa po OSVOS segmentácii vyskytujú a s KNN Matting metódou je možné ich opraviť a správne začleniť danú časť do popredia.



Obr. 4.1: Segmentácia bežiaceho psa OSVOS metódou. Na strednej snímke môžeme vidieť nepresnosť v segmentácii nôh psíka.



Vylepšenie segmentácie po OSVOS metóde  
 Pôvodný obrázok (a), Segmentácia po OS-  
 KNN matting metódou. (a) pôvodný obrázok,  
 (b) OSVOS segmentácia, (c) OSVOS + KNN  
 Matting segmentácia.

VOS metóde (b) je zvýraznená červenou  
 maskou, výsledná segmentácia po KNN  
 matting metóde (c).

Obr. 4.2: Príklady vylepšenia segmentácie objektov pomocou KNN Matting metódou.

Na obrázku 4.3 je v prvom riadku segmentácia po OSVOS metóde a v druhom upravená segmentácia gymnastky KNN Matting metódou. OSVOS metóda niekedy začleňuje do masky popredia aj širšie okolie okolo objektu, ktoré nieje úplne popredím.

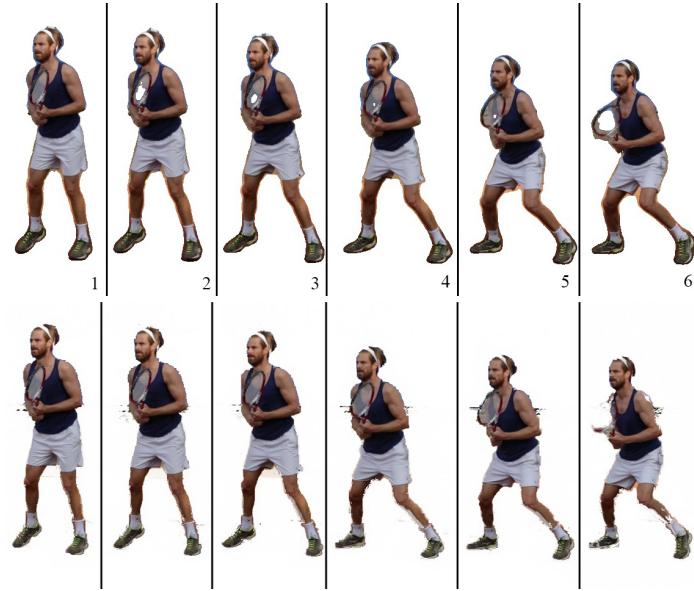
KNN Matting metóda funguje na tento typ problémov a odstraňuje oblasť, ktorá ne-patrí do popredia. Na obrázku 4.4 vidíme porovnanie OSVOS metódy s OSVOS+KNN Matting metódou kde KNN Matting lepšie segmentoval tenisovú raketu.

#### 4.1.2 Limitácie aplikácie

Na posledných dvoch obrázkoch 4.3 a 4.4 chcem demonštrovať aj limitáciu aplikácie. Na presnú segmentáciu objektu na jednej snímke je potrebné vytvoriť trimap. Trimap v aplikácii vzniká pomocou morfologických operácií dilatácie a erózie a veľkosť týchto operácií je možné nastavovať v aplikácii. Ak však spustím KNN Matting metódu pre celú sekvenciu, zníženie kvality segmentácie objektu je podmienené konštantným nastavením veľkosti trimapu. Na obidvoch obrázkoch bol trimap nastavovaný pre prvú snímku sekvencie (vľavo) a na ostatných snímkach je vidieť zníženie presnosti KNN Matting metódy.



Obr. 4.3: Demonštrácia zvýšenia presnosti segmentácie vďaka použitiu KNN Matting (druhý riadok) metódy po OSVOS metóde (prvý riadok).



Obr. 4.4: Segmentácia po OSVOS metóde (prvý riadok) a segmentácia po OSVOS+KNN Matting metóde. Obrázok demonštruje lepšie segmentovanie rakety hráča po KNN Matting metóde. Zároveň sekvencia dobre demonštruje zníženie kvality segmentácie objektu spôsobenej konštantným trimapom.



Segmentácia OSVOS metódou.



Segmentácia po KNN matte metóde

Obr. 4.5: Ukážka zlyhania KNN matting metódy.

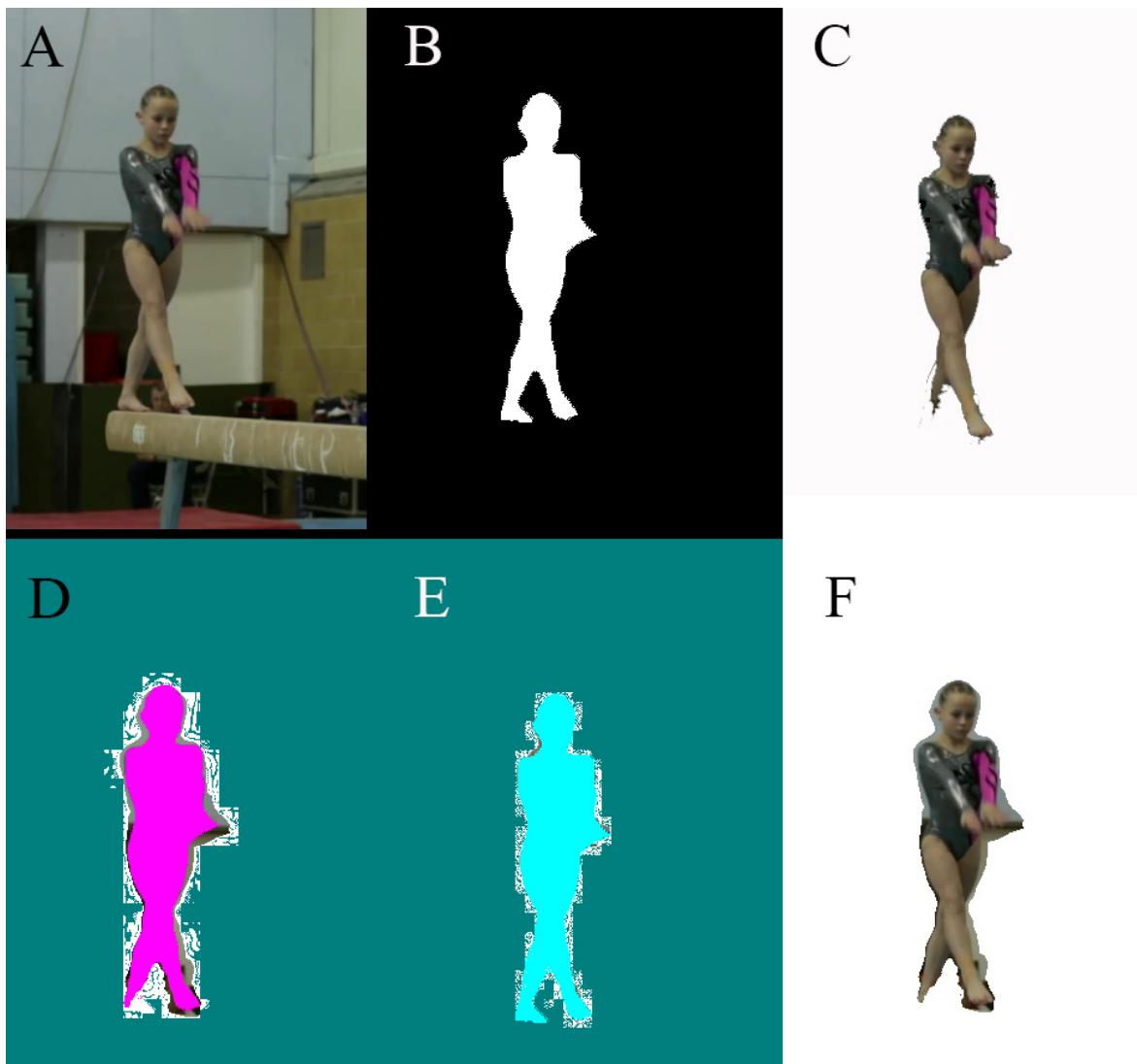
Druhým obmedzením, ktoré demonštrujem na obrázku 4.5b je vychádzanie z OSVOS metódy, ktorá nie vždy správne nachádza objekt. Pri výbere KNN Matting metódy som predpokladal, že ak OSVOS vysegmentuje objekt len čiastočne a zahrnieme do hľadania objektu pomocou KNN Matting metódy dostatočne veľkú časť obrázka

(tzn. vykonám silnú eróziu a dilatáciu), popredie a pozadie v trimape bude určite zodpovedať poprediu ktoré chcem segmentovať a pozadiu obrázka. KNN Matting dokáže nájsť presné hranice objektu. Po implementácii a testovaní som však zistil, že pre KNN Matting je dôležité, aby rozhodoval len o pixeloch na ktorých vzniká splývanie popredia a pozadia na obrázku.

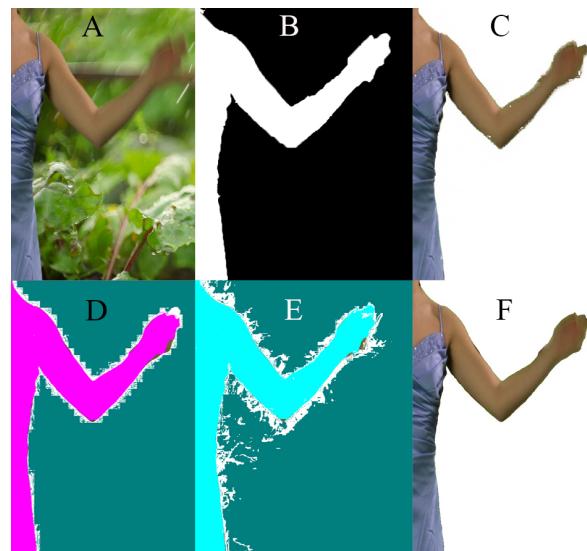
## 4.2 Testovanie

Pre vyhodnotenie presnosti metód som sa rozhadol z videí vybrať 4 snímky, pre ktoré vytvorím presné masky popredia. Následne použijem OSVOS metódu a OSVOS+KNN Matting metódu a porovnám zaradenie každého pixelu do popredia alebo pozadia. Na nasledujúcich 4 obrázkoch 4.6, 4.7, 4.8 a 4.9 môžeme vidieť pôvodný obrázok (a), presnú masku popredia (b), segmentáciu po OSVOS metóde (c), Masky popredia a pozadia pre OSVOS segmentáciu (d), Masky popredia a pozadia pre KNN Matting segmentáciu (e) a KNN Matting segmentáciu (f).

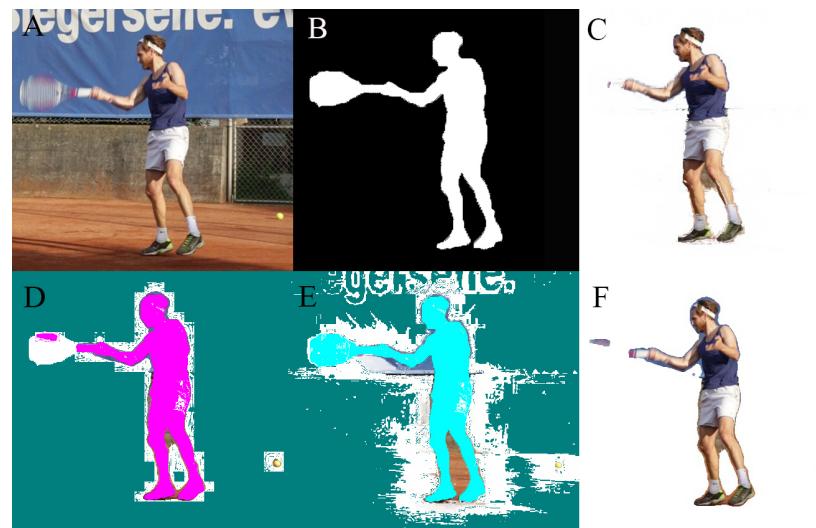
Dôležitou časťou sú najme časti (d) a (e) na týchto obrázkoch. Masky (d) a (e) na nasledujúcich obrázkoch znázorňujú presnú masku pozadia (tyrkysová farba). Fialová maska je presná maska popredia, ktorá prekrýva segmentáciu OSVOS a zvýrazňuje časti, ktoré OSVOS metóda zaradila do popredia a nepatria tam. Svetlo tyrkysová farba na obrázku 4.6 znázorňuje tiež masku popredia. V Okolí masiek a segmentácií OSVOS a KNN Matting metód môžeme vidieť biele oblasti. Toto sú oblasti pozadia z segmentácií (c) a (f). Sú to pixely, ktoré OSVOS a KNN Matting nesegmentovali ako pozadie, pretože pozadie má na výslednej segmentácii konštantnú farbu. Nesegmentovali to ale ani ako popredie. Sú to pixely veľmi blízke pozadiu, ale niesú jednoznačne označené ako pozadie.



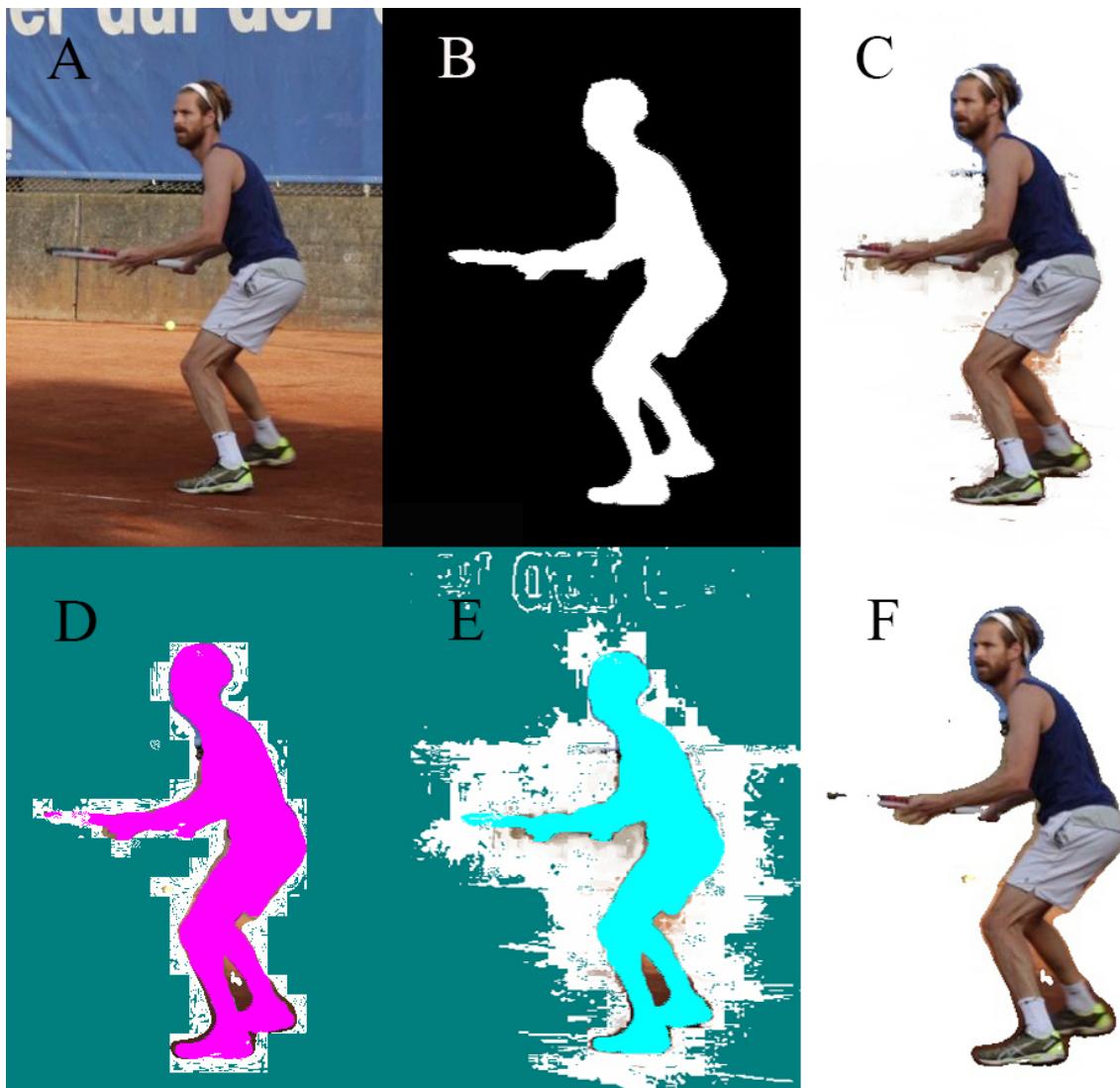
Obr. 4.6: Pôvodný obrázok (a), presná maska popredia (b), segmentácia po OSVOS metóde (c), Masky popredia a pozadia pre OSVOS segmentáciu (d), Masky popredia a pozadia pre KNN Matting segmentáciu (e) a KNN Matting segmentácia (f). Na obrázku môžeme vidieť, ako KNN Matting metóda zlepšila segmentáciu objektu na jeho okrajoch. Zároveň je ale faktom, že nenašla nohu objektu. Túto nepresnosť môžem odvôdniť tým že som trimap vytvoril tak, aby bol objekt viac dilatovaný, nakoľko som chcel dosiahnuť presnejšiu segmentáciu po okrajoch objektu.



Obr. 4.7: pôvodný obrázok (a), presná maska popredia (b), segmentácia po OSVOS metóde (c), Masky popredia a pozadia pre OSVOS segmentáciu (d), Masky popredia a pozadia pre KNN Matting segmentáciu (e) a KNN Matting segmentácia (f). Na obrázku je vidieť že obidve metódy majú porovnatelné výsledky. KNN matting nedokáže vždy zvýšiť presnosť riešenia.



Obr. 4.8: pôvodný obrázok (a), presná maska popredia (b), segmentácia po OSVOS metóde (c), Masky popredia a pozadia pre OSVOS segmentáciu (d), Masky popredia a pozadia pre KNN Matting segmentáciu (e) a KNN Matting segmentácia (f). Na obrázku v častiach (c) a (e) môžeme vidieť veľkosť KNN Matting alpha mapy s relatívne malou oblastou ktorú segmentuje.



Obr. 4.9: pôvodný obrázok (a), presná maska popredia (b), segmentácia po OSVOS metóde (c), Masky popredia a pozadia pre OSVOS segmentáciu (d), Masky popredia a pozadia pre KNN Matting segmentáciu (e) a KNN Matting segmentácia (f)

Obrázok	Veľkosť obrázka (počet pixelov)	Celková presnosť OSVOS metódy (%)	Celková presnosť KNN Matting metódy (%)	Presnosť OSVOS (len maska objektu) (%)	Presnosť KNN Matting (len maska objektu) (%)
Gymnastka	480 000	0.8862458333333334	0.8954875	0.8774541626458462	0.8977190981665058
Ruka	497664	0.8713541666666666	0.8460579828960906	0.8769159845282	0.8892394530898624
Hráč tenisu 1	409920	0.870830893832943	0.7920228337236534	0.872811574501141	0.8939311550225761
Hráč tenisu 2	409920	0.8602044301327088	0.8167788836846213	0.7558486897344116	0.8199257589818375

Obr. 4.10: pôvodný obrázok (a), presná maska popredia (b), segmentácia po OSVOS metóde (c), Masky popredia a pozadia pre OSVOS segmentáciu (d), Masky popredia a pozadia pre KNN Matting segmentáciu (e) a KNN Matting segmentácia (f)

V tabuľke môžeme vidieť presnosti pre segmentáciu objektu záujmu z obrázku. Pomerne dobré výsledky by som odôvodnil tak, že pre relatívne krátke sekvencie videí som vždy vytvoril štyri masky objektu, keďže tak dosahuje OSVOS metóda presnosť až 86%. Druhým faktorom pre pomerne presnú segmentáciu je aj fakt, že objekt sa nachádza len na malej časti obrázka. Viac ako polovica veľkosti obrázka je vždy určite pozadie ktoré metódy segmentujú správne. KNN Matting v niektorých prípadoch vylepšuje OSVOS metódu, ale nie je to výrazný posun. Táto neefektivita KNN Matting metódy môže byť zapríčinená trimapmi, ktoré tvorí morfologicky a ktoré by mali byť tvorené adaptívne len na oblasti, ktoré vie KNN Matting metóda dobre segmentovať (tzn. vlasy, transparentné oblasti, rozmazenie pohybom, atď.).

# Záver

V tejto práci som sa venoval segmentácii objektov vo videu, ktoré môžu byť rozostené rýchlym pohybom. Aj keď som nenašiel práce venujúce sa presne tejto téme, existuje pomerne veľké množstvo prác zaobrajúcich sa jej čiastočnými cieľmi, alebo prác, tejto téme blízkych. Naštudoval som rozsiahlu literatúru venujúcu sa detekcií rozmazania objektu pohybom na obraze a segmentácií objektu z obrazu a videa. Zanalyzoval som riešenia a navrhol som poloautomatickú metódu skladajúcu sa z troch metód ktorých prienikom vznikla aplikácia s pomerne nízkymi nárokmi na hardvér a perspektívnymi výsledkami,

Aplikácia používa metódu Grabcut, pre vytvorenie masky objektu záujmu na akejkoľvek snímke vo videu. Používa OSVOS konvolučnú neurónovú sieť na najdenie objektu na ostatných snímkach videa. OSVOS patrí medzi niekoľko momentálne najlepších prác v problematike segmentácie objektu z videa. Pre segmentáciu rozmazaných častí objektu na jeho okrajoch, som zvolil KNN matting metódu, ktorá nie je primárne určená na problém segmentácie objektu rozmazaného rýchlym pohybom, ale aj napriek tomu sa ukázala byť nádejnou pre zvýšenie presnosti segmentácie.

Výsledky ukazujú, že v niektorých prípadoch KNN vylepšilo segmentáciu objektu získaného metódou OSVOS. V ďalšom výskume chcem vylepšiť vytváranie trimapu, ktorý je vstupom pre KNN matting. Verím, že vyriešenie problému vytvárania trimapu adaptívne, vzľadom na predpokladaný objekt, môže výrazne zlepšiť výsledky v presnosti.

# Literatúra

- [1] Amit Agrawal, Yi Xu, and Ramesh Raskar. Invertible motion blur in video. In *ACM Transactions on Graphics (TOG)*, volume 28, page 95. ACM, 2009.
- [2] Usman Ali and Muhammad Tariq Mahmood. Analysis of blur measure operators for single image blur segmentation. *Applied Sciences*, 8(5):807, 2018.
- [3] Xue Bai and Guillermo Sapiro. A geodesic framework for fast interactive image and video segmentation and matting. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.
- [4] Xue Bai, Jue Wang, David Simons, and Guillermo Sapiro. Video snapcut: robust video object cutout using localized classifiers. In *ACM Transactions on Graphics (ToG)*, volume 28, page 70. ACM, 2009.
- [5] Jonathan T Barron and Ben Poole. The fast bilateral solver. In *European Conference on Computer Vision*, pages 617–632. Springer, 2016.
- [6] Yuri Y Boykov and M-P Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Proceedings eighth IEEE international conference on computer vision. ICCV 2001*, volume 1, pages 105–112. IEEE, 2001.
- [7] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation. In *CVPR 2017*. IEEE, 2017.
- [8] Qifeng Chen, Dingzeyu Li, and Chi-Keung Tang. Knn matting. *IEEE transactions on pattern analysis and machine intelligence*, 35(9):2175–2188, 2013.
- [9] Yung-Yu Chuang, Brian Curless, David H Salesin, and Richard Szeliski. A bayesian approach to digital matting. In *CVPR (2)*, pages 264–271, 2001.
- [10] Qingnan Fan, Fan Zhong, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Jumpcut: non-successive mask transfer and interpolation for video cutout. *ACM Trans. Graph.*, 34(6):195–1, 2015.

- [11] Dong Gong, Jie Yang, Lingqiao Liu, Yanning Zhang, Ian Reid, Chunhua Shen, Anton van den Hengel, and Qinfeng Shi. From motion blur to motion flow: A deep learning solution for removing heterogeneous motion blur. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [12] Leo Grady, Thomas Schiwietz, Shmuel Aharon, and Rüdiger Westermann. Random walks for interactive alpha-matting. In *Proceedings of VIIP*, volume 2005, pages 423–429, 2005.
- [13] Suyog Dutt Jain, Bo Xiong, and Kristen Grauman. Fusionseg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos. *arXiv preprint arXiv:1701.05384*, 2(3):6, 2017.
- [14] Jiaya Jia. Single image motion deblurring using transparency. In *Computer Vision and Pattern Recognition, 2007. CVPR’07. IEEE Conference on*, pages 1–8. IEEE, 2007.
- [15] Erik Kalalembang, Koredianto Usman, and Irwan Prasetya Gunawan. Dct-based local motion blur detection. In *Instrumentation, Communications, Information Technology, and Biomedical Engineering (ICICI-BME), 2009 International Conference on*, pages 1–6. IEEE, 2009.
- [16] Yeong Jun Koh and Chang-Su Kim. Primary object segmentation in videos based on region augmentation and reduction. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7417–7425. IEEE, 2017.
- [17] Philip Lee and Ying Wu. Nonlocal matting. In *CVPR 2011*, pages 2193–2200. IEEE, 2011.
- [18] Anat Levin, Dani Lischinski, and Yair Weiss. A closed-form solution to natural image matting. *IEEE transactions on pattern analysis and machine intelligence*, 30(2):228–242, 2008.
- [19] Anat Levin, Alex Rav-Acha, and Dani Lischinski. Spectral matting. *IEEE transactions on pattern analysis and machine intelligence*, 30(10):1699–1712, 2008.
- [20] Dingzeyu Li, Qifeng Chen, and Chi-Keung Tang. Motion-aware knn laplacian for video matting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3599–3606, 2013.
- [21] Renting Liu, Zhaorong Li, and Jiaya Jia. Image partial blur detection and classification. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

- [22] Jonathon Luiten, Paul Voigtlaender, and Bastian Leibe. Premvos: Proposal-generation, refinement and merging for video object segmentation. *arXiv preprint arXiv:1807.09190*, 2018.
- [23] Kevis-Kokitsi Maninis, Sergi Caelles, Yuhua Chen, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. Video object segmentation without temporal information. *arXiv preprint arXiv:1709.06031*, 2017.
- [24] Yasushi Mishima. Soft edge chroma-key generation based upon hexoctahedral color space, October 11 1994. US Patent 5,355,174.
- [25] Heesoo Myeong, Stephen Lin, and Kyoung Mu Lee. Alpha matting of motion-blurred objects in bracket sequence images. In *European Conference on Computer Vision*, pages 125–139. Springer, 2014.
- [26] Federico Perazzi, Anna Khoreva, Rodrigo Benenson, Bernt Schiele, and Alexander Sorkine-Hornung. Learning video object segmentation from static images. In *Computer Vision and Pattern Recognition*, 2017.
- [27] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 724–732, 2016.
- [28] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM transactions on graphics (TOG)*, volume 23, pages 309–314. ACM, 2004.
- [29] Mark A Ruzon and Carlo Tomasi. Alpha estimation in natural images. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 1, pages 18–25. IEEE, 2000.
- [30] Mennatullah Siam, Chen Jiang, Steven Lu, Laura Petrich, Mahmoud Gamal, Mohamed Elhoseiny, and Martin Jagersand. Video segmentation using teacher-student adaptation in a human robot interaction (hri) setting. *arXiv preprint arXiv:1810.07733*, 2018.
- [31] Hongmei Song, Wenguan Wang, Sanyuan Zhao, Jianbing Shen, and Kin-Man Lam. Pyramid dilated deeper convlstm for video salient object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 715–731, 2018.
- [32] Jian Sun, Wenfei Cao, Zongben Xu, and Jean Ponce. Learning a convolutional neural network for non-uniform motion blur removal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 769–777, 2015.

- [33] Jian Sun, Jiaya Jia, Chi-Keung Tang, and Heung-Yeung Shum. Poisson matting. In *ACM Transactions on Graphics (ToG)*, volume 23, pages 315–321. ACM, 2004.
- [34] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for video object segmentation. *arXiv preprint arXiv:1706.09364*, 2017.
- [35] Jue Wang and Michael F Cohen. Optimized color sampling for robust matting. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.
- [36] Jue Wang, Michael F Cohen, et al. Image and video matting: a survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(2):97–175, 2008.
- [37] Oliver Whyte, Josef Sivic, Andrew Zisserman, and Jean Ponce. Non-uniform deburring for shaken images. *International journal of computer vision*, 98(2):168–186, 2012.
- [38] Yitzhak Yitzhaky and Norman S Kopeika. Identification of blur parameters from motion blurred images. *Graphical models and image processing*, 59(5):310–320, 1997.