UNIVERZITA KOMENSKÉHO V BRATISLAVE

FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

ROZPOZNÁVANIE OBJEKTOV V OKOLÍ ZA ÚČELOM VIZUÁLNEHO MERANIA DOHĽADNOSTI

Diplomová práca

Študijný program: Informatika
Študijný odbor: 2508 Informatika
Školiace pracovisko: Katedra Informatiky
Školiteľ: RNDr. Andrej Lúčny, PhD.

Bratislava, 2012 **Bc. Marcel Ďuriš**

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Marcel Ďuriš
Študijný program: informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Rozpoznávanie objektov v okolí za účelom vizuálneho merania dohľadnosti

Cieľ: Na rozpoznávanie objektov na obraze sa používajú techniky založené na štatistickom spracovaní rôznych pohľadov na objekt za podmienok, keď je možné tento objekt ľahko izolovať z obrazu. Úlohou diplomanta je použiť podobnú metódu v aplikačnej oblasti meteorologického pozorovania pri meraní prevládajúcej dohľadnosti. Pozorované objekty sú však oveľa ďalej, než pri bežnom rozpoznávaní scény, pre ktoré je väčšina metód rozpoznávania objektov vytvorená.

Literatúra: dokumnetácia k OpenCV
Davies,E.R: Machine Vision, Elsevier 2004
séria článkov o počítačovom videní

Kľúčové slová: počítačové videnie, kognitívne videnie, umelá inteligencia, meteorologické pozorovanie

Vedúci: RNDr. Andrej Lúčny, PhD.
Katedra: FMFI.KI - Katedra informatiky

Dátum zadania: 08.10.2010

Dátum schválenia: 25.10.2010 prof. RNDr. Branislav Rovan, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Ďakujem vedúcemu tejto diplomovej práce za ochotu pomôcť a poskytnúť svoje odborné názory a skúsenosti, bez ktorých by táto práca nedospela do svojej finálnej podoby.

Abstrakt

Dohľadnosť je veličina významne ovplyvňujúca leteckú dopravu, a jej presné meranie podmieňuje bezpečnú prevádzku letísk. Zaužívané metódy však dohľadnosť nemerajú priamo podľa toho, ako ďaleko sú najvzdialenejšie viditeľné objekty, ale ju odhadujú na základe vlastností atmosféry v mieste merania. Rozhodli sme sa nahradiť zaužívané prístroje kamerou a jej výstupy následne analyzovať pomocou algoritmov počítačového videnia. V tejto práci sme analyzovali možnosti detekcie výrazných objektov v scéne a jej využiteľnosť pri rozpoznávaní objektov. Taktiež sme skúmali vplyv meteorologických podmienok na pozorovateľné vlastnosti rôznych objektov v okolí a na základe našich záverov sme navrhli algoritmy, ktoré umožňujú zistiť viditeľnosť daných objektov v konkrétnej scéne. Porovnali sme úspešnosť týchto algoritmov a výsledky na našej testovacej množine naznačujú ich praktickú využiteľnosť.

Kľúčové slová:
počítačové videnie, kognitívne videnie, umelá inteligencia, meteorologické pozorovanie

Abstract

Visibility is a physical quantity greatly influencing aviation and its exact measurement underlies secure operation of the airports. However, common methods don't measure visibility directly based on how far are the most distant visible objects but approximate it based on local properties of atmosphere. We decided to replace commonly used equipment by camera and its outputs analyze using computer vision algorithms. In this thesis we analyzed options of detection of salient object in scene and its feasibility for object detection. We also studied influence of physical phenomena on observable properties of various objects in vicinity and based on our conclusions we designed algorithms which enable us to determine visibility of given objects in concrete scene. We compared these algorithms and results on our test set indicate their practical usefulness.

Keywords:
computer vision, cognitive vision, artificial intelligence, meteorological observation

Predhovor

Spracovanie obrovského množstva údajov počítačmi si našlo nezastupiteľné miesto v mnohých oblastiach priemyslu a ľudského života ako takého, meteorológiu a letectvo nevynímajúc. Stroje automaticky získavajú a spracúvajú hodnoty najrôznejších veličín, a tak umožňujú, aby toto odvetvie dopravy mohlo byť prevádzkované bezpečnejšie a jednoduchšie.

Výsledky monitorovacích systémov sú ale podmienené presnosťou použitých senzorov, ale aj spôsobom merania a vyhodnocovania veličín. Zaujímavé postavenie ma veličina zvaná dohľadnosť, ktorá nám hovorí, ako veľmi vzdialené objekty ešte vidíme. Jej dôležitosť si vynútila rôzne automatické prostriedky na jej meranie, avšak ani jeden z takýchto zaužívaných spôsobov nemeria dohľadnosť priamo.

Merajú sa len vlastnosti atmosféry, ktoré významne ovplyvňujú správanie sa svetla, a na základe nich sa odhadne, ako ďaleko by sme mali dovidieť. Tento nedostatok sme sa rozhodli chápať ako výzvu a priestor na zlepšenie. Rozhodli sme sa nahradiť senzory vzdialené ľudskému vnímaniu takým, ktorý je zatiaľ najbližšie ľudskému oku, rozhodli sme sa nahradiť ich kamerou a jej výstup analyzovať algoritmami počítačového videnia, ktoré by dokázali určiť či sú dôležité objekty viditeľné.

Obsah

Úvod

Vzdialenosť, z ktorej sú objekty v okolí letiska pozorovateľné, ovplyvňuje schopnosť pilota možnosť bezpečne pristáť či vzlietnuť, a preto je potrebné pre plynulú a bezpečnú prevádzku letísk po celom svete túto vzdialenosť merať zohľadňujúc aktuálne podmienky s čo možno najväčšou presnosťou. Vychádzajúc z meteorologickej teórie, ale aj z viacerých predpokladov môžeme skonštruovať viacero rôznych typov prístrojov schopných odmerať s dobrou presnosťou také vlastnosti atmosféry, ktoré najviac vplývajú na schopnosť človeka pozorovať a rozpoznávať vzdialené objekty.

Kvôli technickým obmedzeniam však nedokážeme vykonávať takéto merania na veľkej ploche, zohľadňujúc rôzne fyzikálne aspekty, či dokonca vnímať svetelné spektrum tak, ako ľudské oko a následkom toho všetky dnes používané prístroje poskytujú len odhady dohľadnosti ako veličiny vnímanej človekom. Ich výsledky sú iste dobré a dostatočné pre súčasné potreby, ide to však aj lepšie. Aj keď existencia odporúčaní a metodík umožňuje expertom merať dohľadnosť priamo využitím odborných schopností a zraku, ani tento prístup však nie je pre automatické systémy vhodný kvôli obmedzeniam ľudského pozorovateľa.

Podľa našich vedomostí sa doposiaľ nikto úspešne nepokúsil merať dohľadnosť automaticky a priamo podľa toho, čo je skutočne vidno aj napriek tomu, že kamery dosahujú vysokú úroveň kvality a výpočtový výkon je tiež ľahko prístupný, čo naznačuje možnosť využiť prístupy počítačového videnia a stojového učenia na riešenie problému merania dohľadnosti o čosi lepšie. Našou snahou bolo navrhnúť algoritmus využívajúci kameru ako náhradu ľudského oka, ktorý by dokázal rozpoznať dôležité objekty v scéne a takýmto spôsobom prispieť k zlepšeniu relevantnosti dohľadnosti pre ľudského pozorovateľa.

V prvej kapitole zbežne načrtneme význam dohľadnosti v leteckej doprave a tiež poskytneme stručný úvod do problematiky merania dohľadnosti a vysvetlíme základnú teóriu využívanú súčasne používanými meracími zariadeniami. Spomenieme niektoré obmedzenia existujúcich zariadení a prístupov. Na záver zdôvodnime smer, ktorým sa bude uberať návrh našich riešení.

V druhej kapitole predstavíme niekoľko známych riešení v oblasti vyhľadávania

pútavych objektov scény a pokúsime sa zdôvodniť výber toho najvhodnejšieho a zároveň predstavíme niekoľko spôsobov jeho rozšírenia na viacero úrovní priblíženia.

Niekoľko skúmanych prístupov k rozpoznávaniu objektov predstavíme v tretej kapitole. Stručne zdôvonime nevhodnosť niektorých smerov návrhu. Taktiež vysvetlíme niekoľko sľubných prístupov a ozrejmíme motiváciu, ktorá viedla k ich návrhu.

Štvrtá kapitola pojednáva o realizácii navrhovaných riešení, spomína problémy, ktoré nastali, a tiež vysvetľuje metódy ich riešenia. V tejto kapitole tiež poskytneme niekoľko odporúčaní pre úspešné nasadenie našich algoritmov a načrtneme ďalšie možnosti rozšírenia systému.

Výsledky dosahované navrhnutými riešeniami, ako aj spôsob ich vyhodnocovania vysvetlíme v poslednej piatej kapitole. Okrem stručného popisu vstupov použitých na testovanie opíšeme aj niekoľko spôsobov vyhodnocovania úspešnosti algoritmov.

Kapitola 1

Dohľadnosť v meteorológii

Účelom tejto kapitoly je podrobne analyzovať cieľ práce a jeho špecifiká, a tak podrobne opísať problém, ktorý sme sa rozhodli riešiť. Predstavíme známe prístupy na riešenie problému, ich výhody aj nedostatky. Opíšeme obmedzenia platné v oblasti nasadenia a zdôvodnime potrebu riešenia problému pomocou počítačového videnia. Taktiež sa pokúsime čo najbližšie špecifikovať očakávané vstupy a výstupy systému, a tak opíšeme niektoré predpokladané úskalia spojené s riešením problému.

Táto kapitola bola zostavená za pomoci manuálov a odporúčaní publikovaných Svetovou meteorologickou organizáciou (WMO) v [22, Kapitoly 2 a 9] a Medzinárodnou organizáciou pre civilné letectvo (ICAO) v [16].

1.1 Dohľadnosť a jej význam

V meteorológii označuje dohľadnosť vzdialenosť, z ktorej je objekt alebo svetelný zdroj ešte rozoznateľný. Táto veličina do značnej miery ovplyvňuje najmä leteckú dopravu. V prípade zníženej dohľadnosti boli v minulosti lety presmerovávané na náhradné letiská, s čím je spojené veľké množstvo ďalších komplikácii. V dnešnej dobe je však možné, ak sú letisko aj lietadlo vhodne vybavené a posádka lietadla má vhodný výcvik, pristávať aj za zníženej dohľadnosti. Prevádzka letiska za takýchto podmienok je náročný proces, na ktorom sa zúčastňuje viacero riadiacich zložiek. Samotnému spusteniu prevádzky predchádza fáza prípravy, ktorá začína po dosiahnutí stanovených hodnôt dohľadnosti.

Dohľadnosť definovaná ako vzdialenosť, z ktorej sú objekty v okolí ešte pozorovateľné, by síce mohla byť postačujúca pre meranie ľudským pozorovateľom, avšak je náchylná na ovplyvnenie viacerými subjektívnymi faktormi. Podstané meteorologické veličiny, ako je napríklad priehľadnosť atmosféry, však môžu byť objektívne merané. Svetová meteorologická organizácia definuje dohľadnosť nasledovne

Definícia 1. *Meteorological Optical Range (MOR) je dĺžka dráhy v atmosfére potrebná na zníženie intenzity svetelného toku rovnobežného zväzku lúčov zo žiarovky na päť percent pôvodnej hodnoty. Žiarovka má mať farebnú teplotu*[1] *2700 K.*

Veličina MOR je však len zriedka používaná a pre potreby letectva sa bežne používa tzv. „visibility for aeronautical purposes“ (VIS-AERO) alebo „Runway Visual Range“ (RVR).

Definícia 2. *Dohľadnosť pre letecké účely, „visibility for aeronautical purposes“, je väčšia z:*

1. *Najväčšia vzdialenosť z ktorej je čierny objekt vhodných rozmerov umiestnený na zemi rozoznateľný voči jasnému pozadiu.*
2. *Najväčšia vzdialenosť z ktorej sú pozorovateľné svetlá o výkone 1000 cd voči neosvetlenému pozadiu.*

Veličina definovaná podľa 2 zahŕňa aj isté subjektívne faktory ako je efektívnosť ľudského oka, napriek tomu je to objektívna veličina a dá sa účinne merať. Dohľadnosť pre letecké účely je často používana bez prívlastku „pre letecké účely“ a môže dôjsť k zámene s oficiálne definovanou dohľadnosťou ako MOR. Ďalej môžeme definovať prevládajúcu dohľadnosť ako dohľadnosť (pre letecké účely), ktorá je dosiahnutá alebo prekonaná aspoň na polovici horizontu. Pre pilotov lietadiel je však vhodná aj nasledovne definovaná veličina.

Definícia 3. *Runway Visual Range (RVR) je vzdialenosť, z ktorej je pilot lietadla na strede pristávacej dráhy schopný pozorovať značky na pristávacej dráhe alebo svetlá označujúce okraj alebo stred pristávacej dráhy.*

Obdobne ako dohľadnosť z predchádzajúcej definície, aj túto veličinu je možné merať, respektíve počítať. Podľa odporúčaní je vhodné RVR merať najmä v čase, keď dohľadnosť klesne pod 1 500 m. Výška piatich metrov je braná ako rozumná hodnota výšky očí pilota nad pristávacou dráhou, napriek tomu môže najmä vo veľkých lietadlách táto hodnota dosahovať až desať metrov.

Z praktických dôvodov však nie je možne RVR merať priamo, lebo by bolo potrebné umiestniť prístroje vo výške niekoľkých metrov nad pristávacou dráhou, a teda je potrebné túto hodnotu určovať vzhľadom na to, čo by pilot uvidel z danej polohy. Medzi MOR a výkonom pristávacích svetiel na jednej strane a RVR ako závislej veličine platia zložité vzťahy a detaily o nich sa dajú nájsť napríklad aj v [16].

[1] Má vyžarovať svetlo porovnateľného odtieňa ako ideálne čierne teleso o teplote 2700 K.

1.2 Zaužívané prístupy k meraniu dohľadnosti

V tejto kapitole predstavíme teóriu podopierajúcu v praxi najpoužívanejšie prístupy k automatickému meraniu dohľadnosti a objasníme princíp fungovania prístrojov využívajúcich túto teoriu. Taktiež spomenieme podmienky ich správneho fungovania ako aj požiadavky kladené na presné meranie ľudkým pozorovateľom.

1.2.1 Základné vzťahy

Schopnosť pozorovateľa vidieť vzdialené objekty ovplyvňujú lokálne vlastnosti atmosféry spôsobené rôznymi fenoménmi. Dohľadnosť najviac ovplyvňujú útlm a rozptyl svetla vplyvom častíc v atmosfére (tzv. Rayleighov rozptyl). Základným vzťahom pre meranie dohľadnosti je tzv. *Bouguerov Lambertov zákon*, ktorý hovorí, že intenzita svetla exponenciálne klesá od prejdenej vzdialenosti x v závislosti od materiálu, ktorým prechádza.

$$F = F_0 e^{-\sigma x} \tag{1.1}$$

Využitím tohto vzťahu vieme dať do vzťahu pomer pôvodnej a konečnej intenzity, tzv. *transmission factor*, a vzdialenosť:

$$T = \frac{F}{F_0} = e^{-\sigma x} \tag{1.2}$$

Ak následne položíme $T = 0.05$ z definície 1, vieme určiť MOR v závislosti od *extinkčného koeficientu*:

$$T = 0.05 = e^{-\sigma P} \tag{1.3}$$

$$P = (1/\sigma)\ln(1/0.05) \approx 3/\sigma \tag{1.4}$$

Štandardnými úpravami vieme spojiť rovnice 1.2 a 1.4 a vyjadriť MOR od vzdialenosti x, ktorú prešlo merané svetlo, a *transmission factor* T, ktorý sme namerali:

$$P = x\ln(0.05)/\ln(T) \tag{1.5}$$

Meranie počas dňa

Pre pozorovania počas dňa môžeme využiť *Koschmiederov zákon*, ktorý dáva do vzťahu kontrast objektu meraný vzdialeným pozorovateľom a kontrast meraný z nízkej vzdialenosti.

Luminancia je fyzikálna veličina, ktorá nám hovorí, koľko svetelnej energie z plochého zdroja môžeme zachytiť na jednotku plochy, a teda udáva, ako jasný sa nám objekt

bude zdať. Luminančný kontrast môžeme pomocou luminancie definovať ako:

$$C = \frac{L_b - L_h}{L_h} \tag{1.6}$$

kde L_b je luminancia objektu a L_h je luminancia horizontu. V prípade, že pozorujeme čierny objekt na horizonte bude $L_b = 0$ a $C = -1$. Koschmiederov zákon môžeme zapísať ako:

$$C_x = C_0 e^{-\sigma x} \tag{1.7}$$

kde C_x je kontrast meraný vzdialeným pozorovateľom a C_0 je kontrast meraný z veľmi malej vzdialenosti. V prípade, že pozorujeme čierny objekt na horizonte a namerali sme $C_x = 0.05$, tento objekt sa nachádza vo vzdialenosti rovnej MOR.

Meranie počas noci

Obdoba Koschmiederovho zákona pre pozorovania v noci je *Allardov zákon*, ktorý dáva do vzťahu iluminanciu E známeho bodového zdroja intenzity I, extinkčný koeficient σ a vzdialenosť zdroja svetla:

$$E = Ix^{-2}e^{-\sigma x} \tag{1.8}$$

V prípade, že meraný zdroj svetla dosahuje prah viditeľnosti, vieme vyjadriť extinkčný koeficient koeficient σ, ktorý môžeme následne dosadiť do vzťahu 1.4:

$$P = x \ln(1/0.05) / \ln(I/(x^2 E_t)) \tag{1.9}$$

Využitie vyššie uvedených vzťahov nám umožní z príslušných nameraných veličín odhadnúť hodnotu vizuálnej dohľadnosti.

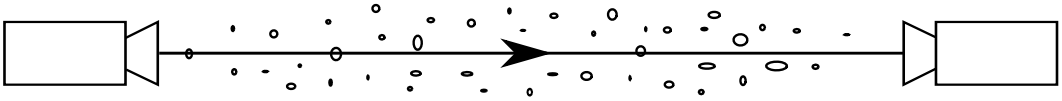
1.2.2 Automatizované meranie

Zariadenia využívane v praxi nemerajú dohľadnosť priamo, merajú vlastnosti atmosféry v mieste nasadenia, z ktorých sa potom vypočíta potrebná hodnota dohľadnosti. Je pochopiteľné, že nie je žiadúce posúvať meracie zariadenia tak, aby zachytávali vysielané lúče práve vo vzdialenosti, v ktorej podľa definície 1 dosiahne intenzita práve päť percent pôvodnej hodnoty. Meracie prístroje môžeme rozdeliť do dvoch širokých kategórii:

- Zariadenia merajúce extinkčný koeficient. Tieto zariadenia merajú útlm svetelného lúča známej intenzity
- Zariadenia merajúce koeficient rozptylu v malom objeme vzduchu. Pri istých fenoménoch môžeme absorbciu lúča časticami zanedbať a koeficient rozptylu dostatočne opisuje útlm lúča v atmosfére.

Prístroje merajúce intenzitu svetelného lúča

Transmisiometer meria extinkčný koeficient σ horizontálneho lúča smerujúceho od zdroja do senzora. Zdroj vysiela orientovaný a modelovaný lúč známej intezity. Z pomeru intenzity zachyteného lúča a pôvodnej intezity je následne vypočítaný extinkčný koeficient. Transmisiometre sú schopné vykonávať merania s vysokou presnosťou, ak sú dostatočne udržiavané. Pre správne fungovanie tohto prístroja je nutné zabezpečiť nemennú polohu obidvoch koncov zariadenia napríklad v prípade mrznúcej pôdy, silného vetra či iných nepriaznivých fenoménov. Obdobne je treba brať do úvahy starnutie zdroja svetla. Niektoré zariadenia sú schopné merať intenzitu lúča vysielaného zdrojom, a tak kompenzovať prípadné zmeny výkonu zdroja.



Obr. 1.1: Schéma fungovania transmisiometra.

Využívajúc vzťah 1.4, do ktorého dosadíme vzdialenosť prekonávanú vysielaným lúčom, je správanosť merania podmienená správnosťou *Bouguerovho Lambertovho zákona* ako aj predpokladom, že MOR nadobúda rovnaké hodnoty pozdĺž meracieho zariadenia ako aj medzi pozorovateľom a pozorovaným objektom. Pre zabezpečenie dostatočnej presnosti merania je všeobecne uznávanou praxou merať MOR v rozsahu 1 až 25 násobku základnej dĺžky transmisiometra.

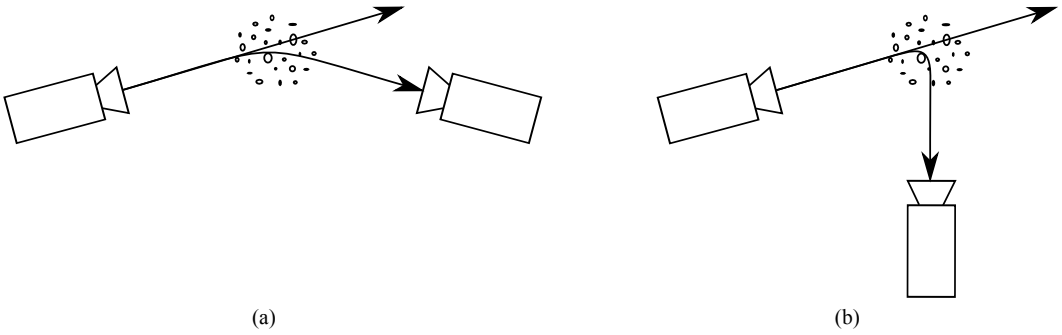
Pre účely merania RVR je však potrebné, aby meraná hodnota RVR nebola nižšia ako základná dĺžka zariadenia. Ak je RVR blízka základnej dĺžke zariadenia, intenzita meraného lúča totiž klesá na príliš nízke hodnoty. Na druhej strane, s klesajúcou základnou dĺžkou zariadenia je potrebné merať intenzitu zachyteného lúča s vyššou presnosťou. Ak chceme jediným transmisiometrom merať celý rozsah RVR, musí byť tento schopný merať MOR v rozsahu od 9.8 m až po 2 km.

Problémy v meraní transmisiometra môžu nastať, ak fenomén zodpovedný za zníženú dohľadnosť inak ovplyvňuje svetelné frekvencie využívane zariadením a inak tie, na ktoré je citlivé ľudské oko. Chyby pri meraní môžu nastať aj následkom rozptylu svetla v prípade, že svetlo pôvodne rozptýlené smerom von z lúča je následkom tohto javu naspäť lámané smerom k senzoru. Zariadenie môže taktiež vykonávať chybné merania v prípade, že meraný fenomén sa lokálne prejavuje inak pozdĺž meranej vzdialenosti ako v okolí celého letiska. Praktické skúsenosti však ukázali, že takéto situácie nie sú dostatočne časté, aby spôsobovali významné odchýlky.

Prístroje merajúce rozptyl svetla

Na rozdiel od transmissiometrov, tzv. *forward-scatter meter* a *back-scatter meter* nemerajú ako veľmi bol svetelný lúč utlmený v atmosfére, namiesto toho merajú intenzitu svetla odrazeného v malom objeme vzduchu. Častice v atmosfére svetelný lúč, ktorý ich zasiahne, nielen čiastočne absorbujú ale aj do istej miery rozptýlia v rôznych smeroch. Pri veľkej časti fenoménov zodpovedných za zníženie dohľadnosti sa na oslabení svetelného lúča podieľa práve rozptyl lúča časticami a absorbčná zložka útlmu je zanedbateľná a teda aj zariadenia založené na opísanom princípe sú schopné poskytnúť relevantné merania.

Podľa smeru, v ktorom zariadenie meria intenzitu odrazeného svetla, sa dá rozlíšiť *forward-scatter meter*, ktorý meria intenzitu svetla odrazeného časticami dopredu, a *back-scatter meter* merajúci intezintu svetla odrazeného v uhle väčšom ako 90 stupňov od pôvodného smeru lúča. Medzi dohľadosťou a intenzitou spätne odrazeného lúča sa však nepodarilo nájsť vzťahy dostatončne presné na praktické použitie, preto sa spätný odraz používa skôr na odhad vyskytujúceho sa fenoménu, ktorého znalosť je dôležitá pre vykonanie správneho odhadu založeného na doprednom lome svetla, nakoľko premenlivá veľkosť častíc rôznych fenoménov má rozdielny vplyv na odraz svetla.



Obr. 1.2: Schéma zariadení merajúcich intenzitu rozptýleného svetla. Na obrázku (a) je schéma zariadenia merajúceho dopredný rozptyl, na obrázku (b) spätný rozptyl.

Kým útlm svetla je približne rovnako ovplyvnený rôznymi fenoménmi, odraz je do značnej miery závislý od veľkosti a typu častíc spôsobujúcich daný fenomén. Svetlo inak odrážajú častice hmly, krúpy, dažďové kvapky, snehové vločky či dokonca piesok alebo dymové častice. Pre správne fungovanie takýchto zariadení je preto nutné správne určiť fenomén. Zároveň tieto zariadenia nie sú schopné fungovať, ak častice svetlo neodrážajú.

Nakoľko meraný odraz svetla nastáva iba v malom objeme, je potrebné aby meraná oblasť dostatočne reprezentovala meraný fenomén. Lokálne zmeny fenoménu sa tak môžu na presnosti takýchto zariadení prejaviť oveľa výraznejšie ako na meraní transmisiometrov.

Zhrnutie

Obidva spomínané typy zariadení však majú jeden výrazný nedostatok, nemerajú dohľadnosť priamo, iba ju odhadujú na základe informácii získaných o stave atmosféry v mieste pozorovania. Výrazné problémy tak môžu nastať, ak konkrétny fenomén inak ovplyvňuje svetelné spektrum využívané meracími prístrojmi a spektrum vnímané človekom. Ďalším významným problémom je, že prístroje vykonávajú merania na relatívne malej ploche a tak nie sú schopné zobrať do úvahy podmienky vo väčšej vzdialenosti od miesta merania.

1.2.3 Meranie pozorovaním

Na to, aby meranie ľudským pozorovateľom malo výpovednú hodnotu musia byť splnené viaceré podmienky. Meranie dohľadnosti musí byť vykonávané zaškoleným personálom, ten musí mať navyše „priemerný zrak“, nesmie používať pomôcky ako sú ďalekohľady alebo sa dívať cez okno. Oči pozorovateľa by mali byť približne vo výške približne jeden a pol metra nad zemou a dostatočne zvyknuté na tmu, ak je meranie vykonávané v noci.

Vhodné objekty

Samotnému meraniu predchádza príprava, počas ktorej sú zvolené vhodné pozorované objekty a ich poloha je zaznačená v meracom pláne spolu s ich vzdialenosťou. Pozorované objekty musia byť zvolené opatrne, najvhodnejšie sú takmer čierne až čierne objekty na horizonte. Aj keď za zamračeného počasia by bolo možne využiť aj tmavé objekty (napr. stromy) a chyba merania by nepresiahla niekoľko percent, počas slnečného dňa priamo osvetlená slnkom by už neboli vhodné.

V prípade, že je objekt pozorovaný voči terestriálnemu pozadiu, musí byť od neho dostatočne vzdialený, presnejšie by to mala byť aspoň polovica vzdialenosti tohto objektu od pozorovateľa. V oku pozorovateľa by tiež pozorovaný objekt nemal vytínať uhol menší ako pol stupňa, pretože takto pozorované objekty sa strácajú skôr. Zároveň by pozorovací uhol nemal prekročiť päť stupňov.

Na nočné pozorovanie sa najviac hodia bodové zdroje svetla. Je dôležité, aby boli mechanicky aj opticky stabilné, aby nevyžarovali príliš výrazne v jednom smere alebo pod malým uhlom. Viacero blízkych zdrojov svetla tvoriacich skupinu nie je vhodných na meranie dohľadnosti, nakoľko sa navzájom ovplyvňujú. Pozorovanie bodových zdrojov svetla je tiež ovplyvnené aj svetlom v okolí mimo zorného poľa pozorovateľa a teda je dôležité aby bolo meranie vykonávané na správnom a dostatočne tmavom mieste.

Pre nočné pozorovanie je zároveň nutné zohľadniť nielen vzdialenosť svetelného

(a) Vysoká dohľadnosť

(b) Znížená dohľadnosť

(c) Veľmi znížená dohľadnosť

(d) Nočné pozorovanie

Obr. 1.3: Príklady pozorovanej scény. Možno pozorovať miznúce objekty, výrazné zmeny oproti pozadiu, či dokonca zdroje svetla, ktoré sa objavujú až v noci.

zdroja ale aj jeho intenzitu a intenzitu osvetlenia pozadia. Pre určenie MOR je teda potrebné využiť *Allardov zákon* a vzťah 1.9.

Aproximácia merania

V prípade, že nie sú k dispozícii vhodné objekty a MOR presahuje vzdialenosti prístupných objektov a meranie pomocou zariadení nie je možné, dá sa vykonať aproximácia dohľadnosti. Ostro viditeľné a jasne sfarbené objekty indikujú vyššiu

hodnotu MOR ako je vzdialenosť objektu, naproti tomu nejasné farby a rozmazané hrany naopak znamenajú nižšiu hodnotu.

Nevýhody

Aj napriek tomu, že automatizované meranie dohľadnosti na základe merania rozptylu a absorbcie svetla neposkytujú globálny pohľad na dohľadnosť a sú schopné merať len na relatívne malej ploche, v porovnaní s ľudským pozorovateľom maju niekoľko nesporných výhod.

Dobre nakalibrované zariadenia poskytujú aj počas dlhej doby konzistentné merania s vysokým časovým rozlíšením a nie sú ovplyvnené okolitými vplyvmi. Meranie, či počítanie krátkodobých priemerov dohľadnosti nemá v prípade ľudského pozorovateľa význam nakoľko ten by len ťažko poskytol každú minútu presnú hodnotu. Zároveň je možné vykonávať automatizované merania aj na mieste, kde nie sú viditeľné objekty vhodné na pozorovanie.

1.3 Dohľadnosť a počítačové videnie

Cieľom práce je prispieť k automatizovaniu merania dohľadnosti. Súčasné technológie síce umožňujú automatické meradie dohľadnosti vhodné pre praktické použie, avšak stále majú isté rezervy a od dosiahnutia výsledkov blízkych ľudkému vnímaniu sú ešte veľmi vzdialené. Navzdory tejto skutočnosti nám nie je známe, že by na trhu existovalo riešenie, ktoré by sa snažilo spomínaný nedostatok používaných metód riešiť.

Našim cieľom je prispieť k odstráneniu tejto medzery na trhu a využiť technológiu, ktorá je svojimi vlastnosťami a schopnosťami zatiaľ najbližšie ľudskému oku - kameru. Za náš cieľ si však nekladieme dohľadnosť merať, snažíme sa navrhnúť spôsob akým by sa dali rozoznať objekty v okolí pri rôznych poveternostných podmienkach tak, aby to malo pre prípadné meranie dohľadnosti význam. Očakávaným výstupom je, či sa nám daný objekt podarilo detegovať alebo ho naopak rôzne fenomény nedovoľujú pozorovať.

Súčasný trh si vynútil množstvo metód a algoritmov, ktoré dokážu popísať a rozoznať rôzne objekty. Ako príklad široko známych algoritmov uveďme SIFT (Scale Invariant Feature Transform), popísaný v [12]. Tento algoritmus vyhľadáva významné body v gradiente scény, ktoré sú vzhľadom na svoje okolie v priestore zväčšení extrémne. K týmto bodom je následne vypočítaný kanonický smer a vzhľadom na ten sa vypočíta niekoľko histogramov orientácii na plochách v okolí tohto bodu. Príklad nájdených význačných bodov je na obrázku 1.4.

Algoritmus SURF (Speeded Up Robust Features), popísaný v [1], využíva obdobnú myšlienku ako SIFT, avšak líši sa v metóde vyhľadávania významných bodov a

konštrukcii ich popisov, na ktoré sa používajú integrálne obrazy. Príklad vyhľadania a párovania popisov nájdených bodov je na obrázku 1.5. Algoritmus HOG (Histograms of Oriented Gradient) [2] či (Dominant Orientation Templates) [7] využívajú prístup veľmi odlišný od skor spomínaných algoritmov, keď sa nesnaží popísať nednotlivé body scény, ale súvislú plochu si rozdelí do mriežky políčok, ku ktorým vypočíta popis a následne o prítomnosti objektu rozhodne na základe tejto informácie klasifikátor.

Ako však ukážeme, tieto algoritmy a im podobné známe metódy nie sú vhodné na riešenie nášho problému, napriek tomu nám boli cennou inšpiráciou pri návrhu nášho algoritmu.

Známe algoritmy rozpoznávajúce objekty v scéne by sa dali rozdeliť do dvoch hrubých skupín podľa toho, akým spôsobom sa zaujímavý objekt snažia popísať. Obidve majú svoje výhody a obmedzenia. Rozdelenie podľa spôsobu popisu objektu je teda možné na algoritmy využívajúce:

Riedke deskriptory (Sparse descriptors) Prvým krokom ich výpočtu je vyhľadanie význačných bodov, v druhom kroku sú tieto body popísané. Popis objektu tvorí vzájomná relatívna poloha význačných bodov a ich popis.

Husté deskriptory (Dense descriptors) Objekt známych rozmerov v scéne sa dá ohraničiť obdĺžnikom pevných rozmerov. Algoritmus popis vypočíta na základe obsahu celého obdĺžnika. Vyhľadanie prebieha posúvaním obdĺžnika po celej scéne a porovnávaním popisu pôvodného objektu a aktuálneho obdĺžnika.

Medzi algoritmy využívajúce riedke deskriptory patria napríklad SIFT alebo SURF. Husté deskriptory využívajú algoritmy HOG na rozpoznávanie chodcov a DOT schopný pracovať v reálnom čase.

Riedke deskriptory

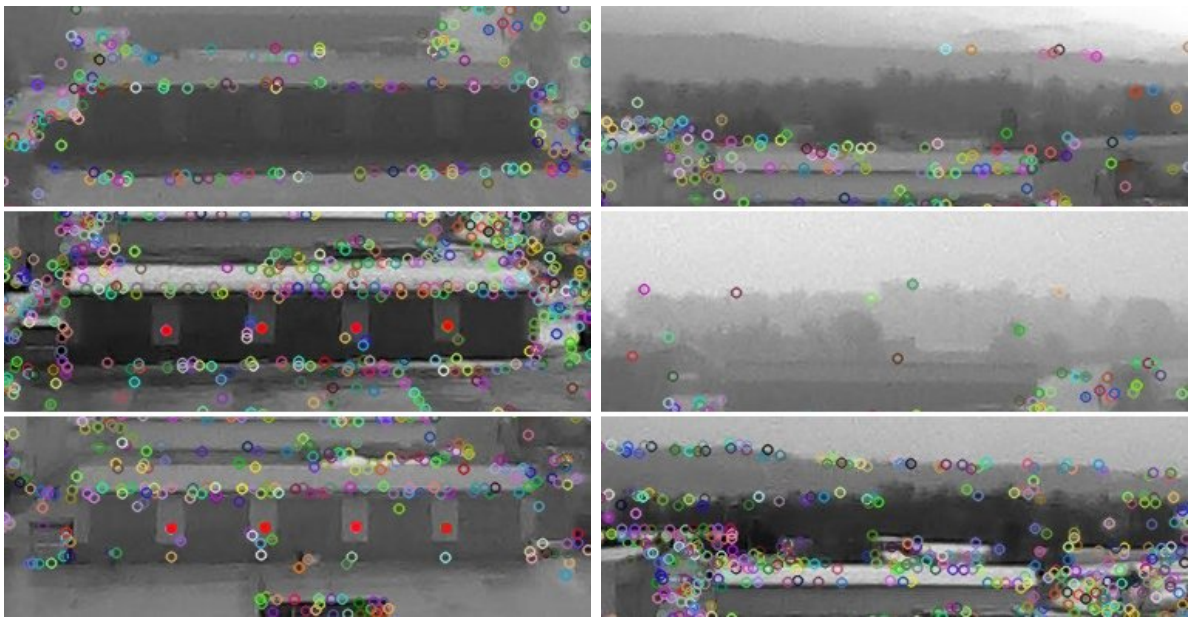
Ako naznačuje názov, tento spôsob tvorby deskriptora objektu sa nesnaží popísať celú plochu vyťatú objektom v scéne. Algoritmy využívajúce túto metódu sa najprv pokúsia nájsť zaujímavé body. Tie je nutné určiť tak, aby boli vyhľadateľné opakovane, nekladieme však požiadavku, aby boli nájdené presne v tom istom mieste objektu vždy. Stačí, aby boli dostatočne blízko na to, aby sa popis vypočítaný v tomto bode nelíšil od optimálnej odpovede. Kvalita vyhľadávania týchto bodov ovplyvňuje úspešnosť a rýchlosť celého algoritmu.

Algoritmus SIFT ako zaujímavé body vyberá tie, ktoré sú extrémami v priestore gaussiánov pôvodnej scény, preto by mal byť schopný vyhľadať aj body prislúchajúce zmenšenému, alebo naopak zväčšenému objektu, využívajúc relatívne zmeny kontrastu.

Algoritmus SURF motivovaný snahou o zrýchlenie vyhľadávania využíva na zistenie polohy význačných bodov integrálne obrazy a vyhľadáva maximálne determinanty matíc a narozdiel od SIFT-u neškáluje skúmanú scénu, ale svoje detektory.

V nájdených bodoch sú následne vypočítané popisy vzhľadom na ich bezprostredné okolie. Opäť, rôzne algoritmy využívajú rôzne metódy s líšiacou sa úspešnosťou. Dôležité je, že ak v popisovanom bode popíšeme kanonický smer, je možné tento bod popísať nezávisle od orientácie objektu, a tak popísať aj inak otočené objekty. Detekcia objektu následne spočíva v napárovaní nájdených popisov. Pri vyhľadávaní objektov z veľkej databázy je možné využiť Houghovu transformáciu, keď podobnosť nájdeného deskriptora v scéne a očakávaného deskriptora v databáze objektov zahlasuje za isté možné polohy objektu z databázy.

Algoritmy využívajúce túto metódu na tvorbu popisov sú schopné popísať a vyhľadať objekty pri miernych zmenách osvetlenia a kontrastu, taktiež sú schopné vyhľadávať inak otočené objekty. Schopnosť popísať objekty nezávisle od ich rotácie je pre naše potreby zbytočná, lebo len zriedka nastanú také meteorologické podmienky, že by bolo potrebné popísať inak zrotovanú budovu alebo iný objekt.

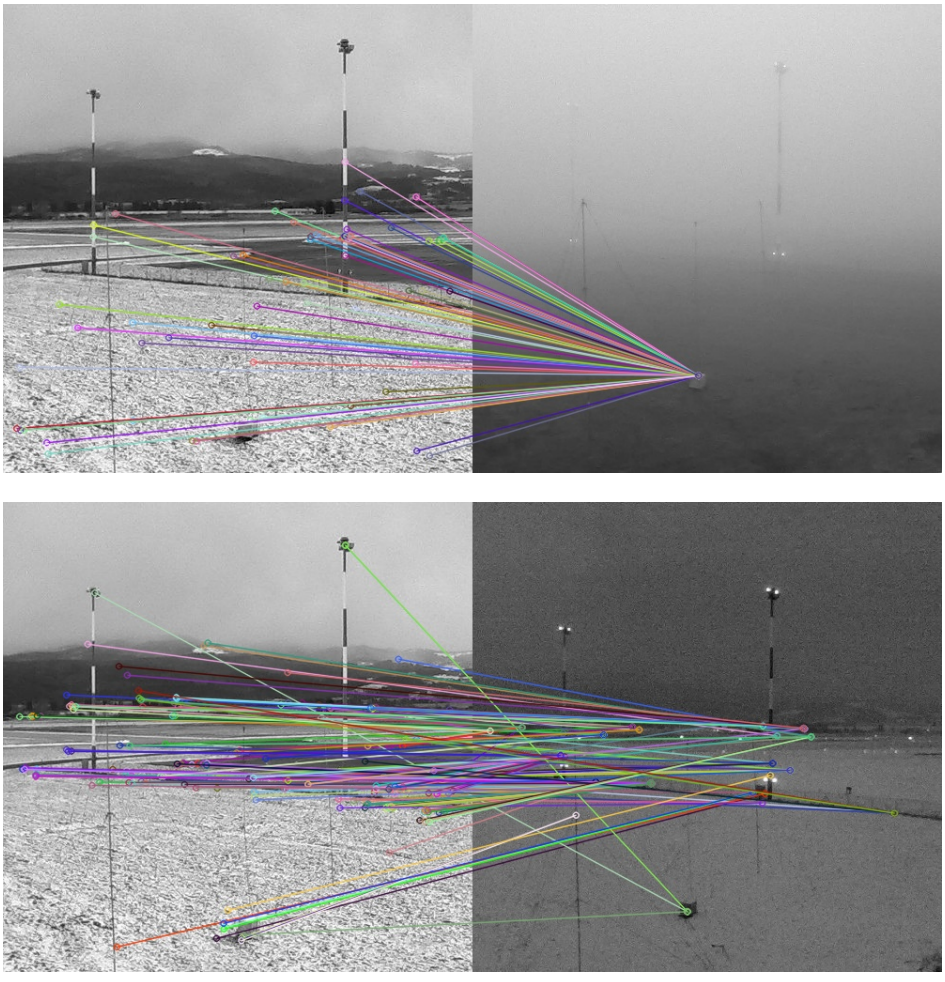


(a) Nevhodne nájdené význačné body (b) Rýchlo strácajúce sa význačné body

Obr. 1.4: Výsledky algoritmu SIFT, ktoré pokladáme za nedostatočné pre naše potreby. Na 1.4a algoritmus síce dokázal nájsť opakovane zlomok význačných bodov, tieto však prislúchali takým častiam objektu, ktoré mohli byť zakryté aj pri dobrej dohľadnosti. Na 1.4b zase možno badať rapídny úbytok význačných bodov aj napriek tomu, že kontúry objektu sú stále veľmi dobre pozorovateľné.

Medzi nevýhody všetkých nami skúmaných algoritmov patrí obmedzená úspešnosť vyhľadávania význačných bodov na hranách, respektíve ich neistá poloha na tejto hrane. Ani algoritmus SIFT ani SURF podľa našich experimentov neboli schopné s dostatočnou presnosťou opakovane vyhľadávať význačné body na hranách. Taktiež vplyvy fenoménov sa ukázali ako príliš náročné pre skúmané algoritmy a ich odolnosť voči zmenám kontrastu a tým pádom aj osvetlenia sa preukázala ako nedostatočná.

Algoritmy síce boli schopné vyhľadať niektoré objekty aj za rôzneho počasia, často sa však ich popis zmenil natoľko, že nebolo možné nájsť navzájom zodpovedajúce si deskriptory a tak napárovať deskriptory vzoru na skúmanú scénu. Zároveň tieto objekty boli málo významné pre popis celého objektu. (Mohli byť napríklad zakryté tieňom aj za pekného počasia.) Ako možno vidieť na 1.4, algoritmus SIFT dokázal vyhľadať



Obr. 1.5: Dva navzájom spárované deskriptory by mali zodpovedať približne tej istej oblasti. Výsledky párovania deskriptorov nájdených pomocou algoritmu SURF sú nedostatočné nielen kvôli často nájdenej náhodnej podobnosti, ale aj kvôli malému množstvu nájdených význačných bodov za zhoršených podmienok.

veľké množstvo význačných bodov, tieto však neboli dostatočne stabilné. Algoritmus SURF ich síce dokázal nájsť oveľa menej a párovanie prebehlo o trochu lepšie, bol však schopný popísať len obmedzené množstvo objektov nehovoriac o tom, že výsledné deskriptory obidvoch algoritmov boli natoľko odlišné medzi jednotlivými vstupmi, že párovanie deskriptorov jednotlivých algoritmov poskytlo nezmyselné výsledky ako vidieť napríklad na obrázku 1.5. Naše experimenty potvrdili nevhodnosť niekoľkých skúmaných algoritmov pre nasadenie v oblasti vizuálneho merania dohľadnosti.

Husté deskriptory

Zrieknúc sa možnosti popísať rôzne rotované objekty a taktiež obmedzením možností vyhľadania objektu v scéne, husté deskriptory popisujú ako veľmi je skúmaný výsek podobný hľadanému objektu. Kým riedke deskriptory sa snažili z popisu celej scény sofistikovanými postupmi vybrať popis hľadaného objektu, hustý deskriptor postupne počíta popisy pre všetky pozície a vyberie tú najlepšiu a dostatočne dobrú. Takýto postup môže byť pomalší, ako preukazuje algoritmus HOG [2], s využitím špeciálnej architektúry dostupných procesorov však dokáže algoritmus DOT [7] pracovať v reálnom čase.

Problémom pri nasadení týchto algoritmov sa javí trénovacia množina. Finálne rozhodnutie algoritmu HOG vykoná SVM, ktorá pre správne fungovanie potrebuje rozsiahlu a dostatočne pestrú vzorku vstupov, ktoré z praktických dôvodov bude obtiažné získať. Ak by bol obdobný prístup nasadený v praxi, bolo by nutné zbierať a ručne triediť vstupy, kým by nepokryli všetky prípustné fenomény, ktoré môžu nastať, čo by mohlo nepríjemne oddialiť spustenie systému. Algoritmus DOT naproti tomu potrebuje trénovanie za možno až laboratórnych podmienok pri extrahovanom skúmanom objekte. Výmenou za obtiažné trénovanie nám algoritmus poskytne pre naše potreby nepotrebnú schopnosť rozoznať objekt z rôznych pozorovacích uhlov a zároveň ostáva citlivý na narušenie siluety objektu.

Aj keď skúmané algoritmy na báze hustých deskriptorov nie sú priamo vhodné pre nasadenie v našej doméne, všeobecný prístup je pre naše potreby vhodnejší ako ten riedkych deskriptorov, nakoľko tie sa javia ako nedostatočne kvalitné pri popise hrán. Práve hrany sa javia ako najvhodnejšie na popis objektov za rôzneho počasia, nakoľko smer a poloha hrany sú aj pri mnohých meteorologických fenoménoch nemenné, čo sa nedá povedať o ich ostrosti a husté deskriptory kvôli rozšírenej množine predpokladov, ktoré poskytujú, považujeme za schopnejšie popísať zle odhaliteľné vlastnosti objektov. Náš výzkum v kapitole 3 sme sa teda rozhodli potreby smerovať k hustým deskriptorom.

Kapitola 2

Metódy konštrukcie mapy saliencie

Saliencia (z lat. salire - vyskakovať, vymykať sa bežnému, pútať pozornosť) je síce termín používaný v neurovede či psychológii, napriek tomu si zaslúžil pozornosť výskumníkov z oblasti počítačového videnia. Ich štúdie sa nezamerali len na odhady či meranie tejto veličiny, skúmal sa aj jej význam pri navigácii autonómnych robotov. Nás však zaujímalo, či sa dá saliencia využiť pri rozpoznávaní objektov za zhoršeného počasia.

Pred štúdiom významu a možností aplikácie saliencie v našej oblasti sme najprv skúmali existujúce metódy. V tejto oblasti bol vykonaný rozsiahly výzkum, výsledkom ktorého je viacero metód založených na odlišných myšlienkach. Metóda vhodná pre naše potreby by mala nielen identifikovať výrazné objekty, mala by byť aj dostatočne rýchla na vykonávanie meraní s požadovanou frekvenciou.

2.1 Neurálnou architektúrou inšpirovaný prístup

Často citovaný a porovnávaný prístup k detekcii výčnievajúcich regiónov popísali v [10]. Autori navrhujú systém založený na skúmaní scény zdola nahor, ktorý je zároveň prípustný aj z biologického hľadiska.

Zo vstupu je vypočítaných najprv niekoľko rôznych topografických máp vlasností, ktoré sú následne kombinované do výslednej mapy saliencie tak, že jednotlivé lokácie súťažia o salienciu, a tak sa na výstup dostávajú iba globálne významné oblasti.

Navrhovaný model skúma intenzitu, viacero kombinovaných farebných kanálov a smerové vlastnosti určené odpoveďami rôzne orientovaných Gáborovych filtrov. Zároveň je skúmaných viacero škál a zmeny spomínaných vlastností medzi nimi. Výsledkom je veľké množstvo čiastkových máp, ktoré obsahujú navzájom zle porovnávateľné hodnoty. Autori teda navrhli metódu normalizácie potlačujúcu mapy s veľkým množstvom obdobných odpovedí a naopak zvýrazňuje mapy s menším množstvom výraznejších extrémov. Normalizácia spočíva jednak v normalizácii do fixného rozsahu hodnôt a

následne prenásobenie hodnôt mapy rozdielom globálneho maxima mapy a priemeru lokálnych maxím mapy.

Pre takto normalizované mapy vieme sčítať zložky orientácie, farby a intenzity do jedinej mapy. Na tejto mape následne súťaží dvojvrstvová neurónová sieť, pričom prvá vrstva kumuluje potenciál blízkych pixelov, ktorý následne posúva druhej vrstve. Tá tvorí tzv. „winner-take-all“ sieť, v ktorej sa potenciál hromadí v niektorých neurónoch rýchlejšie ako v ostatných. Ak neurón druhej vrstvy má dostatok energie, oblasť významnosti je posunutá k nemu a následne sú jemu prislúchajúce pixely vstupu utlmené. Takýmto spôsobom sa dá v prehľadávanej scéne vyhľadať viacero oblastí s najviac zaujímavými objektami.

Experimentálne výsledky [10] ukazujú vysokú odolnosť voči šumu a v porovnaní s inými vtedy známymi metódami celkom dobré výsledky. Pre naše potreby však tento prístup poskytuje len polohu najviac významných objektov a neposkytuje v nijakej forme ich popis. Táto metóda síce nie je priamo pre naše použitie vhodná, poslúži však na porovnanie kvality iných skúmaných metód.

2.2 Polohovo invariantná saliencia

Podstatne odlišný prístup ku konštrukcii mapy saliencie navrhujú v [11]. Ich metóda je založená na meraní entropie v okolí bodu skúmanej bitmapy, ktorú následne využíjú ako mieru prekvapenia v danom bode.

Algoritmus pracuje v troch krokoch, pričom ako základnú meranú vlástnosť berie intenzitu bodov bitmapy.

Algoritmus polohovo invariantnej saliencie

1. **krok** Výpočet entropie v postupne zväčšujúcich sa okoliach.
2. **krok** Vyhľadanie tých rozsahov, v ktorých dosahuje entropia maximum.
3. **krok** Výpočet magnitúdy zmeny funkcie hustoty pravdepodobnosti ako funkcie škály. Skúmame rozsah zmeny intenzít vyskytujúcich sa v postupne zväčšujúcich sa okoliach.

Entropia sa dá chápať ako miera predpovedateľnosti atribútov v okolí skúmaného bodu. Ak sú hodnoty intenzity blízke, entropia dosahuje nízke hodnoty a teda vieme tento atribút odhadnúť správne s vysokou pravdepodobnosťou. Cieľom druhého kroku je výber optimálnej škály, ktorá bude dostatočne odolná voči zmenám spôsobeným otočením alebo malými lokálnymi poruchami. Takéto nájdené rozsahy okolia skúmaných bodov zodpovedajú približne rovnako početným populáciám pixelov rôznych intenzít.

V treťom kroku nájdeme také okolia, ktoré sá s meniacou škálou výrazne menia. Naproti tomu, okolia, ktoré s rastúcou veľkosťou nemenia výrazne svoje vlastnosti sú pre nás nezaujímavé. Výsledná saliencia je súčin entropie a výraznosti vzhľadom na meniace sa rozsahy okolia vypočítanej v treťom kroku.

Tento algoritmus je možné rozšíriť tak, aby bol schopný podávať podobné výsledky aj po afinných transformáciach vstupu. Pre naše výpočty tak použijeme namiesto kruhového okolia eliptické s rôznymi orientáciami a excentricitami.

Vzhľadom na spôsob vyhľadávania význačných bodov sa dá všimnúť, že najviac význačné budú najmä body ležiace na hranách alebo v rohoch. Zároveň algoritmus neberie do úvahy vzájomnú polohu skúmaných bodov a preto môže odignorovať istú informáciu o pravidelnosti skúmanej scény. Nazdávame sa teda, že do istej miery je tento algoritmus možné nahradiť detektorom hrán. Ďalej predpokladáme vysokú výpočtovú náročnosť tohto algoritmu, ktorá by mohla brániť praktickému nasadeniu tejto metódy.

2.3 Saliencia založená na grafoch

Grafový algoritmus založený na myšlienke simulácie pohybu náhodného surfera bol navrhnutý v [6]. Táto metóda je jednoduchá a biologicky akceptovateľná najmä kvôli jednoduchej paralelizovateľnosti. Popíšme stručne základnú myšlienku algoritmu.

Algoritmus pracuje v dvoch krokoch, prvým je zostavenie aktivačnej mapy, ktorá je následne v druhom kroku algoritmu normalizovaná tak, aby boli zvýraznené pútavé regióny. Formovanie aktivačnej mapy je založené na analýze zvolenej vlastnosti vstupu, vstup je najprv predspracovaný lineárnym filtrovaním nasledovaným jednoduchými nelineárnymi filtrami, následkom čoho získame popisné vektory pokrývajúce analyzovaný vstup. V aktivačnej mape by vysoké hodnoty mali zodpovedať nezvyčajným bodom. Autormi jednoducho definovaná miera je založená na logaritmovanom pomere meraných hodnôt. Túto mieru následne využijeme na definovanie váhy hrany medzi jednotlivými uzlami mriežky tak, že mieru ováhujeme klesajúc po Gaussovej krivke.

Po normalizácii súčtu váh hrán odchádzajúcich z uzla na hodnotu 1 môžeme tieto váhy chápať ako prechodové pravdepodobnosti a uzly ako stavy skrytého markovovského modelu. Následne sa pýtame na rovnovážnu distribúciu, teda zlomok času, ktorý by strávil náhodný surfer v takto zostrojenom modeli, keby po ňom mal cestovať donekonečna. Táto veličina teda bude narastať v odlišujúcich sa regiónoch.

Nesmierne dôležitým krokom algoritmu je následná normalizácia aktivačnej mapy, nakoľko berúc do úvahy viacero analyzovaných vlastností vstupu, pri následnej kombinácii to jedinej mapy význačnosti by sa hmota akumulovaná v regiónoch viacerých aktivačných máp mohla relatívne rovnomerne rozdeliť po celej mape význačnosti. Hmotu sa teda

snažíme koncentrovať len do vybraných význačných regiónov.

Na riešenie tohto problému sa opäť osvedčil markovovský prístup podobný tomu pri konštrukcii aktivačnej mapy. Váhu hranám grafu ale priradíme na základe hodnoty aktivácie uzla, do ktorého hrana vchádza, opäť vhodne vyváženou Gaussovou funkciou vzdialenosti uzlov na mriežke.

Takto popísaný algoritmus dosahuje dobré výsledky v porovnaní s klasickými metódami merania saliencie a taktiež dokáže dobre predpovedať fixácie ľudského pozorovateľa. V porovnaní s metódou [10] dosahuje podľa autorov algoritmu pri predpovedaní ľudského správania lepšie výsledky.

2.4 Saliencia pomocou frekvenčnej reprezentácie

Predtým, ako sa pokúsime objasniť význam fázového spektra pri analýze saliencie, vysvetlíme stručne, čo je Fourierova transformácia a aké informácie o analyzovanom obraze nám môže poskytnúť.

Fourierova transformácia

Fourierová transformácia vyjadrí funkciu istej premennej ako funkciu frekvencie. Základná myšlienka myšlienka spočíva v tom, že niektoré funkcie sa pomocou základných zložiek dajú rozpísať do tzv. *Fourierovho radu* a tak vyjadriť pomocou amplitúdy, frekvencie a fázového posunu týchto základných zložiek:

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left[a_n \cos(nx) + b_n \sin(nx)\right] \tag{2.1}$$

$$= \frac{a_0}{2} + \sum_{n=1}^{\infty} \sqrt{a_n^2 + b_n^2} \cos(nx - \arctan\frac{b_n}{a_n}) \tag{2.2}$$

$$= \frac{a_0}{2} + \sum_{n=1}^{\infty} M_n \cos(nx - P_n) \tag{2.3}$$

kde M_n je amplitúda n-tej zložky a P_n je jej fáza.

Fourierovu transformáciu funkcie vieme vypočítať podľa vzorca:

$$F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i x \omega} dx$$

Nakoľko si Fourierova transformácia našla uplatnenie v mnohých odvetviach nielen fyziky a informatiky, bola patrične skúmaná a pre nás dôležité výsledky tohto výskumu sú možnosť rozšíriť Fourierovu transformáciu na dvojrozmerné a diskrétne funkcie a zároveň *State-of-the-Art* algoritmy, ktoré túto ale aj inverznú transformáciu počítajú.

Na naše ďalšie výpočty budeme potrebovať diskrétnu Fourierovu transformáciu dvojrozmerných funkcii. Ak položíme x za bod scény, n_1, n_2 za jeho súradnice, X za bod vo frekvenčnom spektre so súradnicami k_1, k_2, takúto transformáciu môžeme zapísať nasledovne:

$$X_{k_1,k_2} = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} e^{-\frac{2\pi i}{N_1} n_1 k_1} e^{-\frac{2\pi i}{N_2} n_2 k_2} x_{n_1,n_2}$$

Keďže sme sa presunuli zo spojitých funkcií na diskrétne a tie budú navyše pre naše potereby reprezentované bitmapami, integrál bol nahradený sumou na ohraničenom rozsahu. Na záver ešte dopňme inverznú transformáciu:

$$x_{n_1,n_2} = \frac{1}{N_1 N_2} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} e^{-\frac{2\pi i}{N_1} n_1 k_1} e^{-\frac{2\pi i}{N_2} n_2 k_2} X_{k_1,k_2}$$

Fourierová transformácia nám poskytuje výborný nástroj na analýzu scén, a to možnosť reprezentácie vo frekvenčnom spektre. Na vizualizáciu frekvenčného spektra sa často používa amplitúda jednotlivých zložiek. Príklady na 2.1 a 2.2 ukazujú amplitúdy frekvenčných spektier prislúchajúcich jednoduchým vstupom.

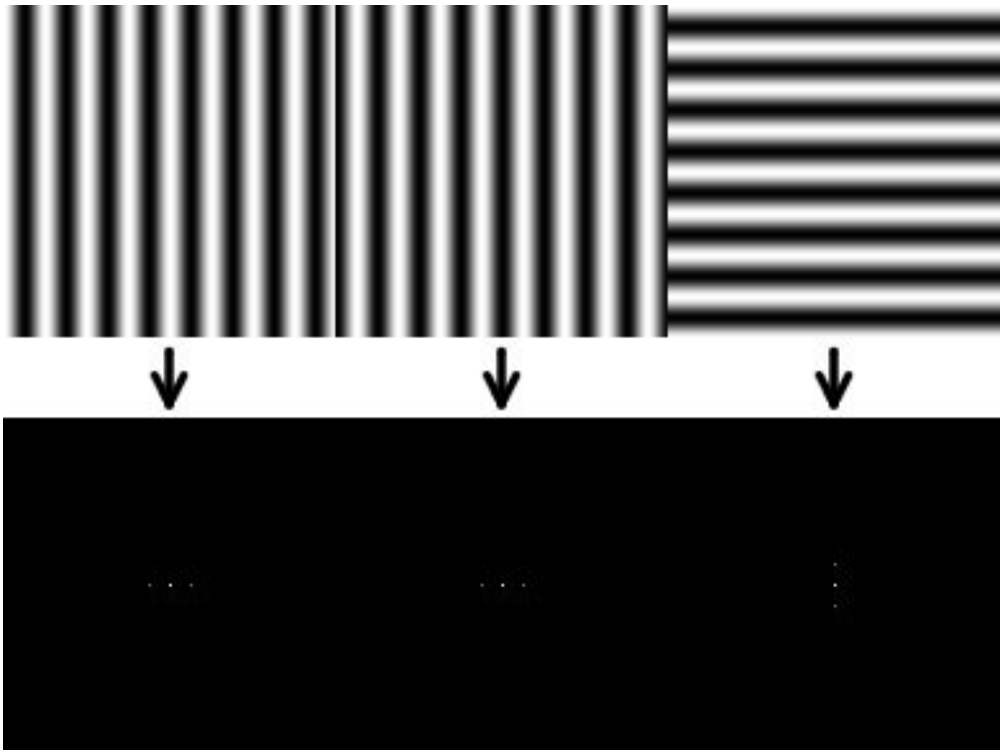
Spektrálne rezíduum

Využívajúc štatistické vlastnosti veľkého množstva analyzovaných scén môžeme využiť frekvenčné spektrum Fourierovej transformácie na konštrukciu mapy saliencie podľa [9]. Základnou ideou algoritmu je, že v priemernej scéne je amplitúda priemerovaná vo všetkých smeroch v závislosti od frekvencie približne opísaná vzťahom $A(f) \approx 1/f$. Autori spozorovali, že po priemerovaní veľkého množstva rôznych scén vykazovalo logaritmované amplitúdové spektrum podobné vlastnosti. Práve odchýlky spektra konkrétnej analyzovanej scény mali byť spôsobené vyčnievajúcou zložkou scény, ktorou sú salientnými objektami.

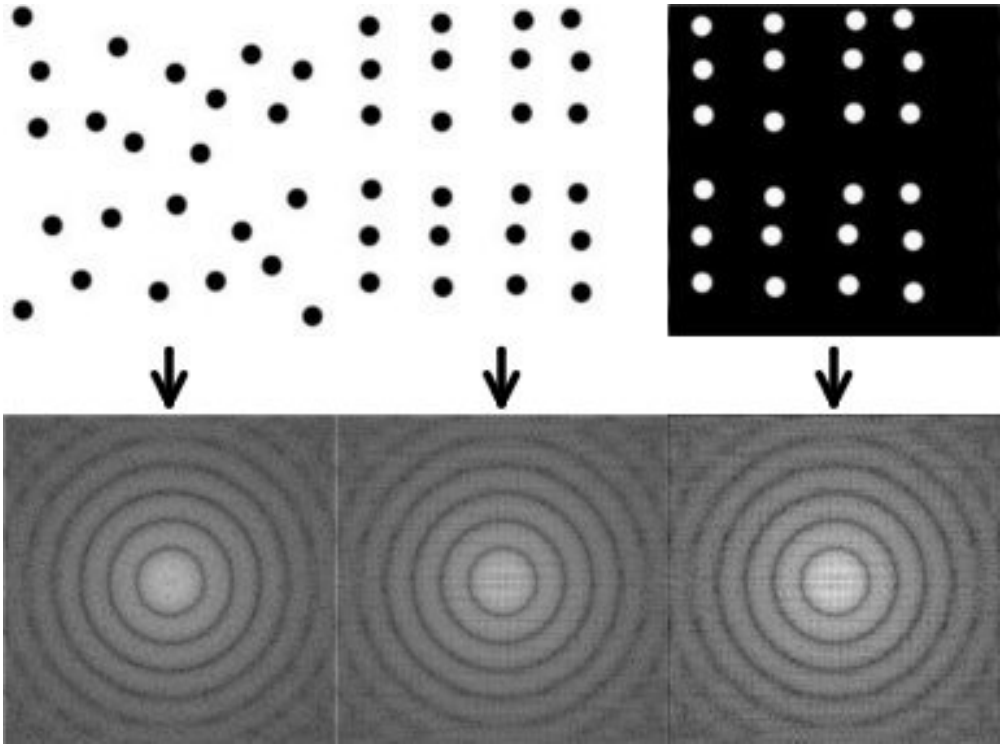
Mapa saliencie sa dá pomocou tejto myšlienky skonštruovať tak, že od logaritmovaného amplitúdového spektra odčítame jeho priemernu hodnotu z väčšieho okolia a následne vykonáme inverznú Fourierovu transformáciu zachovávajúc fázové spektrum. Nespornou výhodou takéhoto prístupu je absencia parametrov, ktoré by sa významnejšie prejavili na efektívnosti algoritmu.

Fázové spektrum

Inšpirovaní prístupom v predchádzajúcej kapitole skúmali v [5] význam fázového spektra pre mapu saliencie. Po dôkladnej analýze prišli autori k zisteniu, že dôležitou zložkou predchádzajúceho prístupu nie je spektrálne rezíduum ale fázové spektrum. Objavenie



Obr. 2.1: Fourierová transformácia jednoduchej vlny sa prejaví vo frekvenčnom spektre ako niekoľko výrazných frekvencií, ktoré sú stredovo symetrické podľa stredu sústavy frekvenčného spektra pričom spojnica týchto frekvencii je v smere vlny.



Obr. 2.2: Na prvý pohľad odlišné obrázky sú zložené z opakujúcich sa vzorov, ktoré sa prejavia na veľmi podobnej amplitúde frekvenčného spektra, ktorá dokonca nie je ovplyvnená ani inverziou farieb vstupu. Tento jav naznačuje, že výrazná časť informácie je kódovaná vo fázovom spektre. Zobrazujeme logaritmus amplitúdy frekvenčného spektra.

významu fázového spektra tak umožní zohľadniť nielen farebné zložky analyzovaného vstupu, ale aj v prípade potreby zohľadniť zmeny v čase využívajúc rozšírenie Fourierovej transformácie.

Aj keď nevieme o žiadnom matematickom dôkaze správnosti metódy založenej na fázovom spektre Fourierovej transformácie, intuitívne by sa dalo objasniť fungovanie metódy napríklad tak, že pravidelnosti v scéne sa prejavia pravidelnými zložkami a v mieste, kde sa vyskytuje nepravidelnosť sa týchto zložiek musí stretnúť viacero, aby túto nepravidelnosť zostavili. Fázové spektrum popisuje lokálnu informáciu v obraze a pomocou neho vieme odhadnúť či sa v danom mieste scény naraz stretáva málo alebo veľa rôznych základných zložiek, ktoré potom v týchto nepravidelných oblastiach vytvárajú tzv. *protoobjekty*.

Algoritmus spočíva z niekoľkých jednoduchých krokov podobných tým vo výpočte mapy saliencie pomocou spektrálneho rezidua. Najjednoduchší spôsob navrhnutý v [5] spočíva vo výpočte nasledujúceho:

$$F(x,y) = \mathfrak{F}\{f(x,y)\} \tag{2.4}$$

$$p(x,y) = P(F(x,y)) \tag{2.5}$$

$$M(x,y) = \|\mathfrak{F}^{-1}\{e^{i\cdot p(x,y)}\}\|^2 * g(x,y) \tag{2.6}$$

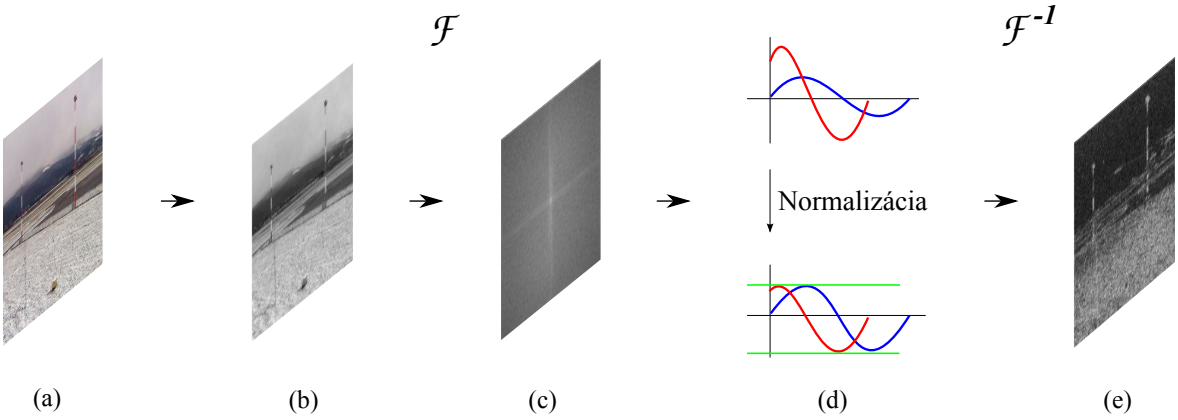
kde $f(x,y)$ je pôvodný obraz, P označuje fázovú zložku spektra. Rozdiel oproti metóde spektrálneho rezidua nastáva v rovnici 2.6 v ktorej chýba pričítanie odchýlky od priemeru. Pre tento prístup sa osvedčilo záverečnú mapu rozmazať konvolúciou s gaussiánom, nakoľko bola málo kohézna a tvorila akúsi mriežku.

Nakoľko výstupom Fourierovej tranformácie sú komplexné čísla, ktorých absolútna hodnota určuje amplitúdu a odklon od reálnej osi určuje fázu, predchádzajúci zápis sa dá zjednodušiť na:

$$M(x,y) = \left\| \mathfrak{F}^{-1}\left\{ \frac{\mathfrak{F}\{f(x,y)\}}{\|\mathfrak{F}\{f(x,y)\}\|} \right\} \right\|^2 * g(x,y) \tag{2.7}$$

Z tejto rovnice už je zrejmé, ako metódu implementovať, amplitúdu bodu vo frekvenčnom spektre normalizujeme na hodnotu 1 a následne vykonáme inverznú Fourierovú transformáciu takto upraveného frekvenčného spektra. Záverečné spracovanie mapy saliencie Gaussovým filtrom sa nám po našich experimentoch ukázalo ako nepotrebné v nami skúmanom a neskôr aj využívanom rozšírení tejto metódy, ktoré objasníme v nasledujúcich statiach. Pre metódu opísanú rovnicou 2.7 sa však záverečné spracovanie mapy saliencie vyhladením ukázalo ako potrebné z dôvodu nižšej kohéznosti vzniknutej mapy.

Výpočet mapy saliencie pomocou fázového spektra by sme mohli zhrnúť nasledovným postupom:



Obr. 2.3: Výpočet mapy saliencie pomocou fázového spektra. Vstup (a) skonvertujeme na čiernobiely (b), vypočítame Fourierovu transformáciu (c), normalizujeme amplitúdu na jednotkovú veľkosť (d). Inverznou transformáciou získame mapu saliencie (e).

Algoritmus na výpočet mapy saliencie pomocou fázového spektra

1. **krok: Načítanie vstupu** 2.3(a)(b) Spracuvávať budeme intenzitu vstupu, preto je potrebné prípadne farebnú vstupnú bitmapu skonvertovať na čiernobielu.

2. **krok: Výpočet Fourierovej transformácie** 2.3(c) Vypočítame Fourierovú transformáciu niektorým zo známych algoritmov. Získame tak reprezentáciu vo frekvenčnom spektre, z ktorého môžeme získať informácie o fáze a amplitúde jednotlivých zložiek.

3. **krok: Normalizácia** 2.3(d) Výstupom Fourierovej transformácie je bitmapa komplexných čísel, aby sme získali fázove spektrum, normalizujeme amplitúdu zložiek. To dokážeme vykonať tak, že každý komplexný pixel upravíme na jednotkový.

4. **krok: Inverzná transformácia** 2.3(e) Normalizovanú bitmapu prevedieme do pôvodného spektra inverznou Fourierovou transformáciou. Z amplitúdy výsledku už ľahko získame mapu saliencie.

Ako je ľahko vidieť z rovnice 2.7, táto metóda nedisponuje prakticky žiadnymi voľnými parametrami, a tak odpadá prípadný zdroj chýb pri nekvalitnom fungovaní mnohých algoritmov a to nesprávne zvolené voľné parametre. Táto metóda je zároveň jednoduchá na implementáciu, nakoľko si dovolíme predpokladať dostupnosť kvalitných algoritmov počítajúcich doprednú aj inverznú Fourierovu transformáciu a dokonca je schopná pracovať na vstupoch v nižšom rozlíšení v reálnom čase, na výstup zo snímoky vo vysokom rozlíšení si však môžeme počkať aj niekoľko sekúnd.

Hlavné výhody spomínaného prístupu však autori [5] nevidia v absencii parametrov ani v rýchlosti, ale v možnosti využiť výskum Fourierových transformácií [4] a rozšíriť vzťah 2.7 do vyšších dimenzii využitím kvaterniónov. Autori [4] ukázali, že je možné rozšíriť Fourierovu transformáciu z komplexných čísel na kvaternióny bez potreby zavádzania obrovského množstva ďalšej teórie a zložitých algoritmov.

Úvahy o Fourierovej transformácii viacrozmerných čísel vychádzajú z tzv. Cayleyho Dicksonovej formy kvaterniónu, v ktorej je kvaternión definovaný rekurzívne ako akési generalizované komplexné číslo, ktorého zložky sú samé komplexnými číslami. Napríklad kvaternión $q = a + bi + cj + dk$ sa dá prepísať do tejto notácie ako $q = A + Bj$, kde $A = a + bi$ a $B = c + di$. Tento zápis sa dá ďalej generalizovať na ľubovoľné jednotkové navzájom kolmé kvaternióny, ktorých špeciálnym prípadom sú aj čísla i a j pričom a A tvorí tzv. *simplexnú* a B *perplexnú* časť kvaterniónu, pričom je samozrejmé, že obe sú izomorfné s komplexnými číslami.

Rozdeliac vstupnú bitmapu do takejto simplektickej notácie môžeme vstup chápať ako dve zložky, dve samostatné bitmapy, tvoriace Cayleyho Dicksonovú formu zápisu farebnej bitmapy a samotnú Fourierovú transformáciu zapísať ako

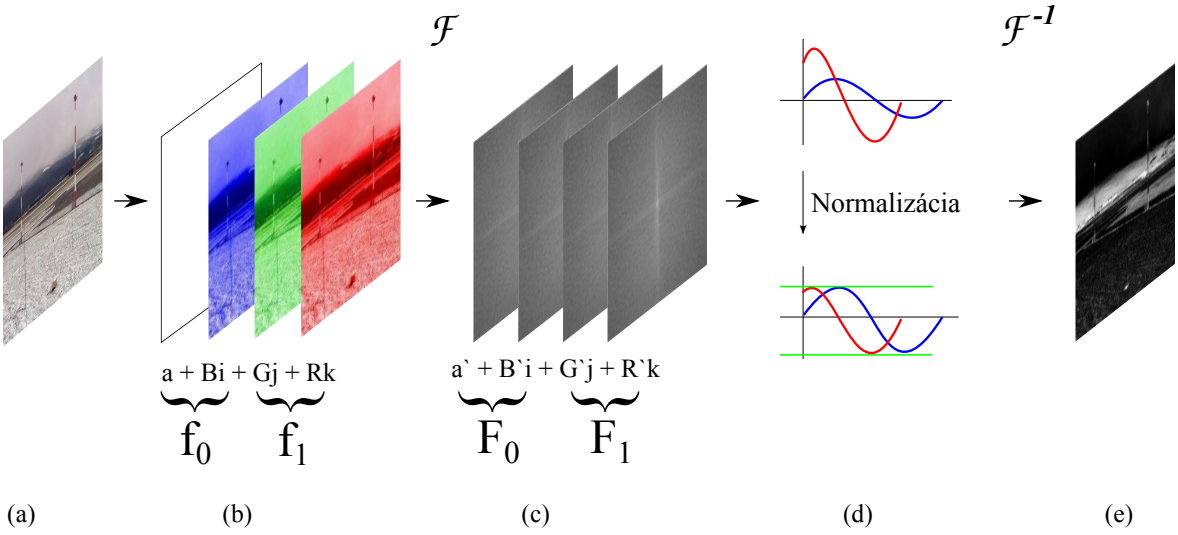
$$F(u, v) = F_1(u, v) + F_2(u, v)j \tag{2.8}$$

kde F_i sú transformácie, ktoré sú izomorfné s klasikými Fourierovými transformáciami jednotlivých zložiek a ktoré už môžeme implementovať za pomoci existujúcich knižníc. Výpočet aj takto zložitej Fourierovej transformácie na kvaterniónoch by sme mohli zredukovať na výpočet dvoch paralelných obyčajných Fourierových transformácii.

Aby sme neodbočili príliš od pôvodnej témy tejto práce, opísali sme metódu rozšírenej transformácie len zbežne a opomenuli sme napríklad vizualizáciu tohto spektra či aj úvahy o tom, ako sa voľba bázy (ktorej špeciálnymi prípadmi je báza i, j, k) prejeví na farebnej reprezentácii bitmapy. Všetky detaily ale aj ďalšie odkazy sa dajú nájsť v [4].

Ako sa dalo predpokladať zo smerovania nášho výkladu, ďalej postup podľa [5] využije práve vyššie spomínanu Fourierovú transformáciu, a tak rozšíri prístup fázového spektra o informácie o farebných zložkách vstupu. Pri efektívnom zápise nám však ostane jeden rozmer kvaterniónu nevyužitý a [5] navrhujú jeho obsadenie zmenou intenzity v priebehu času, čo by umožnilo výpočet dynamických máp saliencie berúcich do úvahy aj zmeny a pohyb vo video vstupe naproti nepohyblivým bitmapám. Takéto rozšírenie síce môže mať nesmierny význam pri spracovávaní videa, pre naše potreby sa však uspokojíme s nepohyblivými vstupmi nakoľko v našom prípade nás zaujíma súčasný stav dohľadnosti a nie jeho zmeny v priebehu času.

Autori [5] zároveň odporúčajú previesť vstupnú bitmapu do tzv. *oponentového* farebného modelu [21, Kapitola 4] odvolávajúc sa spracovanie obrazu v ľudskom mozgu



Obr. 2.4: Schéma výpočtu mapy saliencie - Vstup (a) sa rozdelí na jednotlivé farebné zložky, z ktorých je konštruovaná kvaterniónová reprezentácia (b). Následne je vykonaná Fourierová transformácia a získame frekvenčné spektrum (c), ktoré následne normalizujeme a inverznou Fourierovou transformáciou po úpravách amplitúdy výsledku získať výslednú mapu saliencie.

pomocou obdobného farebného modelu. Naše experimenty však ukázali, že takýto prevod sa nijako neprejaví na kvalite výslednej mapy saliencie a teda jednotlivé zložky reprezentácie pomocou kvaterniónu získame priamo z hodnôt RGB modelu nasledovne:

$$q = Bi + Gj + Rk$$

kde B, G a R sú modrá, zelená respektíve červená farebná zložka.

Z predchádzajúceho popisu je snáď zrejmé, že prvým krokom v konštrukcii mapy saliencie je výpočet dvoch Fourierových transformácii na komplexných bitmapách vytvorených z pôvodnej bitmapy ako Bi respektíve $G+Ri$. Nasleduje výpočet amplitúdy každého pixela takto získanej reprezentácie vo frekvenčnom spektre, aby sme následne vedeli každý kvaternión normalizovať na jednotkovú amplitúdu.

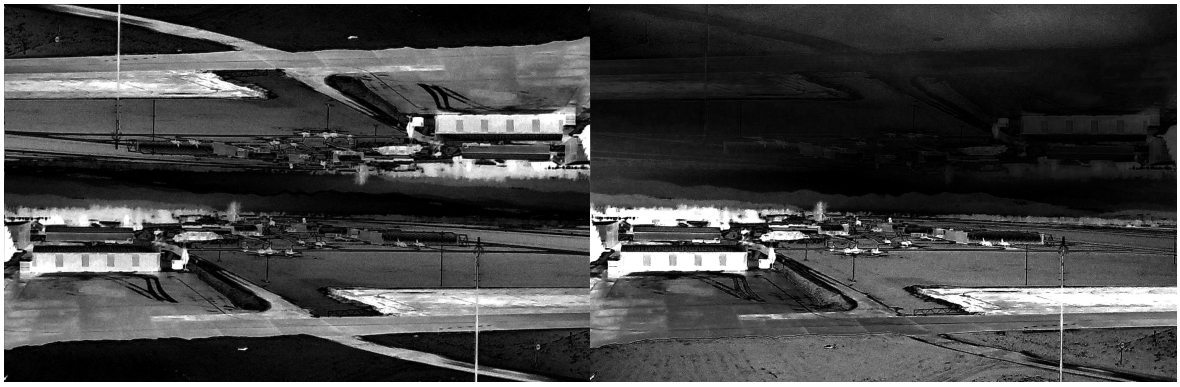
Samotnú mapu saliencie vieme zostrojiť ak obdobným spôsobom vykonáme inverznú Fourierovu tranformáciu na takto normalizovaných kvaterniónoch tvoracich bitmapu frekvenčného spektra a ako výsledok použijeme štvorec amplitúdy výsledku inverznej transformácie.

Samotný výpočet mapy saliencie pomocou kvaterniónového fázového spektra je až na formu Fourierovej transformácie totožný s jednoduchšou obdobou počítanou len za pomoci intenzity reprezentovanej pomocou komplených čísel. Výpočet je zhrnutý v 2.4.

Medzi mapami saliencie tvorenými pomocou kvaterniónov, ktoré zohľadňovali jednotlivé farebné zložky, a mapami, ktoré boli zostrojené len pomocou komplexných

čísel spracúvajúcich intenzitu vstupnej bitmapy, sme si všimli niekoľko zaujímavých rozdielov. Prvý z nich súvisí s artefaktami, ktoré sa síce menej výrazne prejavali na mape generovanej pomocou kvaterniónov, ale aj tak znehodnotili naše výstupy. Tieto artefakty vyzerali ako očakávateľný výstup otočený hore nohami a na mape saliencie počítanej z intenzity boli rovnakej intenzity ako samotná mapa saliencie kým na mape zohľadňujúcej farebné zložky dosahovali intenzitu približne na úrovni najmenej salientných objektov.

Prítomnosť artefaktov v našich výstupoch by sme mohli zdôvodniť predpokladom, ktoré mal nami použitý algoritmus, nakoľko pri použití algoritmu implementovaného v jazyku Octave sa tento problém neprejavil. Dokonca ani samotní autori tohto algoritmu na výpočet mapy saliencie nespomínali nijaké vzniknuté artefakty ani problémy s použitím Fourierovej transformácie. Aby sme dosiahli rozumné výsledky aj použitím našej knižnice, rozhodli sme sa vstup rozšíriť tak, aby bol symetrický. Aj následkom tohto rozšírenia sa výpočet mapy saliencie spomalil, dosiahli sme rozumné výsledky blízke tým, ktoré dosiahli [5]. Artefakty taktiež zanikli, ak sme Fourierovu transformáciu vykonali na vstupe, ktorý bol najprv rozšírený na rozmery vhodné pre algoritmus diskrétnej Fourierovej transformácie. Rozšírenie sa však prejavilo iba na fázovom spektre počítanom z intenzity.



(a) Fázové spektrum intenzity (b) Fázové spektrum farebného vstupu

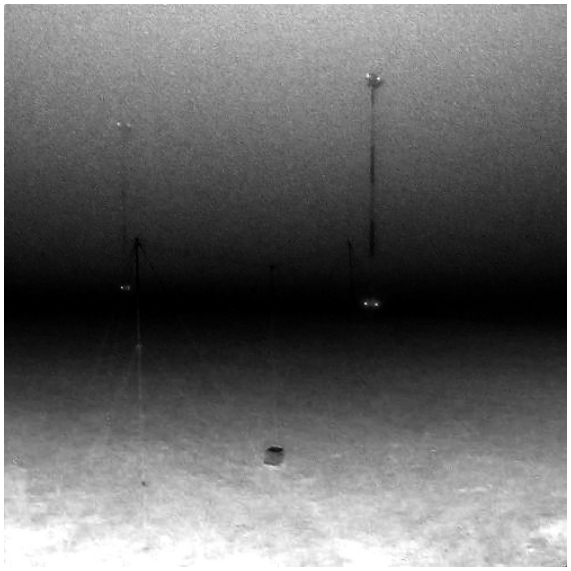
Obr. 2.5: Artefakty, ktoré vznikli pri tvorbe mapy saliencie

Ďalším prekvapivým zistením je, že vyššie spomínaným rozšírením sa zmenil charakter správania sa jednoduchej mapy saliencie počítanej z intenzity. Kým nerozšírená verzia generovala mapy so salientnými objektami, na symetrických vstupoch sme na výstupe dostali salientné hrany. Opäť, toto správanie sme potvrdili implementáciou v jazyku Octave a rovnako [5] ho nijako nespomínajú. Podobné výsledky však možno pozorovať v dávnejšom a inak zameranom výskume fázového spektra [15]. Mapa saliencia vytvorená pomocou kvaterniónov nebola okrem odstránenia artefaktov nijako inak ovplyvnená a naďalej na nej boli ako salientné vyhodnocované celé objekty a nie iba

hrany salientných objektov.

(a) Vysoká dohľadnosť

(b) Znížená dohľadnosť



(c) Veľmi znížená dohľadnosť

(d) Nočné pozorovanie

Obr. 2.6: Mapy saliencie zodpovedajúce vstupom 1.3

Aj keď stavia metóda tvorby mapy saliencie pomocou fázového spektra Fourierovej transformácie aj na dávnejších výsledkoch o význame fázy pri spracovaní signálov, ktoré skúmali napríklad aj [15], samotné využitie takejto reprezentácie signálu je relatívne nový a nazdávame sa, že aj celkom nepreskúmaný, prístup k analýze saliencie objektov scény. To, že o tento prístup majú výskumíci záujem sa dá potvrdiť aj niekoľkými článkami, ktoré boli vydané na túto tému počas písania tejto práce.

Výsledky pri vyhľadávaní významných objektov v scéne by sme mohli považovať za

kvalitné, avšak aké výsledky dosiahneme pri salientných detailoch, či malých objektoch na scéne, ktoré môžu byť zatienené salienciou objektov zaberajúcich v scéne väčšiu plochu? Obdobnú otázku si položili aj [8] a výsledkom ich snaženia bola metóda využívajúca ako základ údajne mapu saliencie konštruovanú pomocou kvaterniónov a zároveň zohľadňujúca nielen viaceré úrovne detailov, ale aj odlišnosť textúr. Ako bolo naznačené, malé objekty nemusia byť vždy správne vyhodnotené ako salientné kvôli tomu, že sa v rozsiahlej scéne stratí ich význačnosť.

Intuitívne riešenie tohto problému by spočívalo v priblížení si scény tak, aby sme vedeli spomínané detaily lepšie pozorovať. Na tento účel sa často používa prefiltrovanie bitmapy gaussiánom, pre naše účely však tento prístup nie je vhodný, nakoľko by spôsobené rozmazanie síce znížilo význačnosť rozsiahlych objektov, na strane druhej by spôsobilo zničenie nami požadovaných detailov, a tak by zavinilo znehodnotenie medzivýsledkov pre akékoľvek ďalšie spracovanie.

Na riešenie problému analýzy viacerých škál navrhujú [8] využiť anizotropnú difúziu navrhnutú [17], ktorá sa nám mimochodom osvedčila aj v mnohých iných aplikáciách. Vhodne zvolené parametre tejto formy spracovania umožnia postupné vyhladenie jemných farebných prechodov vo vnútri plochy vymedzenej objektami a postupné vyhladzovanie stále výraznejších a výraznejších hrán bez straty úrovne detailov. Podľa [8] je potrebné použiť viacero úrovní vyhladenia vstupu stále agresívnejším anizotropným filtrovaním tvoriace základ, z ktorého bude následne vypočítaných niekoľko máp saliencie. Na každej z nich sa prejavia v rôznej miere iné detaily, nakoľko tieto môžu byť zatienené výraznosťou iných objektov a následne odfiltrované, aby uvolnili miesto jemnejším vrstvám. Viacero vrstiev bude spojených do tzv. *regionálnych* máp saliencie jednoduchým geometrickým priemerovaním.

Ďalšie navrhované kroky výpočtu podľa [8] majú za cieľ odfiltrovať rozsiahle plochy pokryté zložitými textúrami, ktoré sa síce môžu javiť ako salientné, ale z globálneho pohľadu predstavujú iba súvislú plochu istých vlastností. Príkladom môže byť lúčny kvet, ktorý sa sám o sebe môže javiť v scéne ako výrazný, ale na lúke posiatej takýmito kvetmi sa neobyčajnosť a tym pádom aj saliencia jediného kvetu stratí a lúka sa stane iba podkladovou plochou. Na vyriešenie tohto problému navrhujú použiť histogram farieb vyskytujúcich sa na relatívne malej ploche. Ak sa histogram skúmanej plochy odlišuje výrazne od histogramov plôch s ňou susediacich, bude táto plocha a všetky objekty na nej uznané ako salientné. V prípade, že farebné histogramy budú naopak podobné, pravdepodobne sú si aj samotné plochy podobné svojou textúrou a ich význam vo výslednej mape bude potlačený.

Aj keď by mohla saliencia na viacerých úrovniach detailov znamenať veľký prínos pre ďalšie spracovanie, nemôžeme si dovoliť do bodky implementovať prístup, ktorý

(a) Vysoká dohľadnosť (b) Veľmi znížená dohľadnosť

Obr. 2.7: Našou metódou vytvorené mapy saliencie zodpovedajúce vstupom 1.3a a 1.3c. Môžeme si všimnúť, že v porovnaní s 2.6c sa neobjavujú rozsiahle salientné plochy na inak nevýrazných častiach vstupu ale inak sme dostali podobné výsledky ako v 2.6a.

navrhli [8], nakoľko by pre naše potreby mohla informácia o samotných textúrach tvoriť vitálnu informáciu pre následné rozonávanie objektov. Naše experimenty sme sa teda rozhodli ochudobniť o filtrovanie textúr pomocou histogramov postupom spomínaným vyššie.

Reprodukovateľnosť výsledkov [8] sa však ukázala nízka. Autori síce spomínajú využitie máp saliencie konštruovaných pomocou algoritmu Fourierovej transformácie na kvaterniónoch, ich medzivýsledky o tom podľa našich skúseností nesvedčia. Dôvodom je, že táto forma máp saliencie obsahuje salientné objekty a celé plochy, ktoré pokrývajú. V [8] ale hovoria o salientných hranách, ktoré vieme získať pomocou klasickej Fourierovej transformácie intenzity vstupu, čo potvrdzuje aj [15].

Ďalšie úvahy o viacškálovej saliencii sme brali s väčšou opatrnosťou. Napriek tomu sa nám nepodarilo s výraznejším úspechom aplikovať ani viacero úrovní máp saliencie konštruovaných za pomoci anizotropnej difúzie. Tento náš neúspech by sme mohli prisúdiť našej neschopnosti nájsť tie správne kombinácie argumentov, ale čiastočne aj nášmu postupne klesajúcemu záujmu o túto metódu, nakoľko sa javila ako príliš pomalá.

Pokúsili sme sa teda navrhnúť vlastný spôsob výpočtu saliencie detailných objektov. Základná myšlienka bola, že na dostatočne malej ploche, môže byť aj podstatný detail dostatočne salientný. Rozhodli sme sa teda vypočítať veľké množstvo máp saliencie z

políčok mriežky, na ktorú sme rozdelili vstup. Prekvapil nás však nekohézny výstup s veľmi ostrými prechodmi saliencie na hranách prislúchajúcim políčkam mriežky v mape saliencie.

Tento problém sme sa rozhodli riešiť zväčšením políčok mriežky, ich prekryvom a následným priemerovaním, čím zabijeme dve muchy jednou ranou. Jednak pomocou prekryvu políčok zohľadníme v mape saliencie viaceré polohy objektu v čiastkových mapách, čím rovnomerne zohľadníme vplyv okolia , ale následným priemerovaním ováhovaným stúpajúcou vzdialenosťou od štvorcového stredu príslušnej čiastkovej mapy saliencie sa zbavíme nekonzistentných salientných prechodov na hranách políčok mriežky. Výsledky nášho algoritmu sú na obrázku 2.7 a nazvali sme ho algoritmus lokálnej saliencie.

Aplikáciou nášho postupu sme vyriešili aj dovtedy nevšimnutý problém. Na vstupoch počas zhotovených počas nepriaznivých podmienok sa vyskytovali rozsiahle súvislé plochy ponorené do hmly. Tieto plochy však boli vyhodnotené ako salientné, pričom intenzita tejto nesprávnej saliencie postupne klesala so stúpajúcou vzdialenosťou od mysleného ohniska. Pri priemerovaní čiastkových máp saliencie sa tento prechod nepravej saliencie stratil a rozsiahla zahmlená plocha bola pokrytá približne rovnakou intenzitou saliencie.

Avšak ani výsledky pri konštrukcii viacúrovňových máp saliencie sa nejavili ako výrazne dôležité pre naše potreby, nakoľko boli príliš podobné tým z klasického globálneho prístupu. Aj keď sa nám nepodarilo získať užitočné výsledky pre viaceré úrovne detailov, nie sme sklamaní, nakoľko viacúrovňovú salienciu sme pokladali skôr za obohatenie algoritmu, nie jeho nutný prvok. Ďalšie zistenie je, že obdobné výsledky by sa dali dosiahnuť aj pri výpočtoch na malej ploche. Aj keď je zrejmé, že mapa saliencie počítaná len na časti vstupu nemôže byť rovnaká ako jej zodpovedajúca časť mapy z celého vstupu, neboli výsledky opísané v časti 3.2 ovplyvnené do významnej miery.

Touto metódou sa nám však podarilo vyhľadať salientné objekty, a tak predspracovať vstup do podoby, o ktorej dúfame, že poskytne dôležitú informáciu pri rozpoznávaní objektov. Pri návrhu algoritmu sme sa snažili využiť výsledky, ktoré sa nám podarilo dosiahnuť pomocou tejto metódy v skorých fázach rozpoznávania.

Kapitola 3

Návrh algoritmov rozpoznávania objektov za zníženej dohľadnosti

Cieľom nášho štúdia saliencie bolo nájsť čo možno najlepší spôsob ako predspracovať vstupný obraz tak, aby v ňom bolo možné čo najjednoduchšie identifikovať zaujímave objekty. V našom prípade ide o vopred známe objekty vhodné na meranie dohľadnosti. V tejto kapitole predstavíme niekoľko skúmaných spôsobov ako určiť, či je hľadaný objekt naozaj viditeľný. Zároveň sme predpokladali stabilnú kameru a prítomnosť objektov vhodných na meranie dohľadnosti pozorovaním.

Prípadný algorimus sme sa snažili navrhnúť tak, aby mal používateľ možnosť zvoliť okrem zaujímavých častí scény aj prah viditeľnosti, akúsi minimálnu mieru pri ktorej ešte považujeme objekt za viditeľný. Algoritmus by tak pre konkrétnu pozorovanú scénu a daný objekt vrátil hodnotu, ktorú by sme porovnali s týmto prahom a na základe toho rozhodli o tom, či je objekt viditeľný alebo nie. V nasledujúcich častiach popíšeme niekoľko prístupov, ktoré sme skúmali.

3.1 Detekcia objektov segmentovaním scény

Ako prvú sme skúmali možnosť priamo využiť mapu saliencie na zistenie prítomnosti hľadaného objektu. Po analýze máp saliencie vygenerovaných z našich testovacích vstupov sme zistili, že objekty bez ďalších detailov sa na mape prejavia ako súvislá plocha, dokonca často jasnejšía ako okolie skúmaného objektu. V niektorých prípadoch sa objekt síce javil na mape saliencie tmavší ako okolie, jeho obrysy však ostali zachované. K takejto inverzii dochádzalo často v prípade, že bola ako salientná vyhodnotená obloha nad horizontom. Blízke objekty, u ktorých bolo možné pozorovať do istej miery aj detaily, boli spravidla blízke a na mape saliencie vysekávali v siluete blízkeho objektu výraznú plochu mierne odlišnej intezity. V každom prípade však salientný objekt tvorený jednou

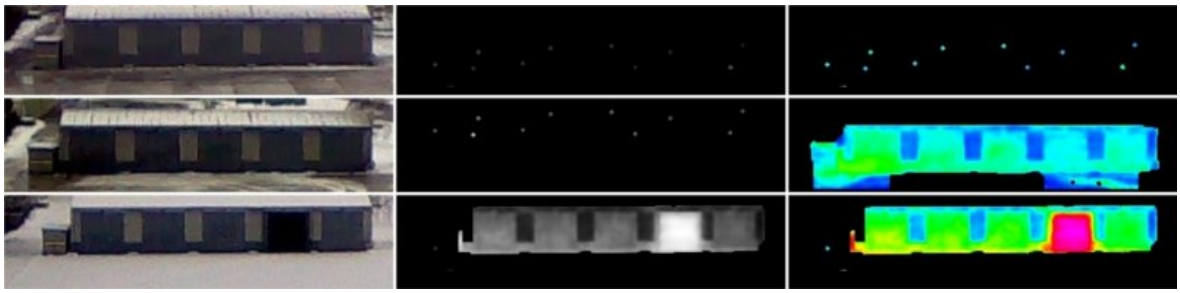
alebo viacerými súvislými plochami, v ktorých sa intenzita mapy saliencie výrazne nemenila.

Vyššie spomínané skutočnosti nás viedli k domnienke, že pozorovaný salientný objekt je dobre vyčleniteľný zo svojho okolia nielen človekom ale aj automaticky niektorým známym segmentačným algoritmom. Nakoľko naše vstupy pochádzajú zo stabilnej kamery, je možné zafixovať niekoľko význačných bodov o ktorých vieme povedať, či patria objektu alebo naopak pozadiu. Poloha a príslušnosť týchto objektov by poslúžila ako vstup pre segmentačný algoritmus.

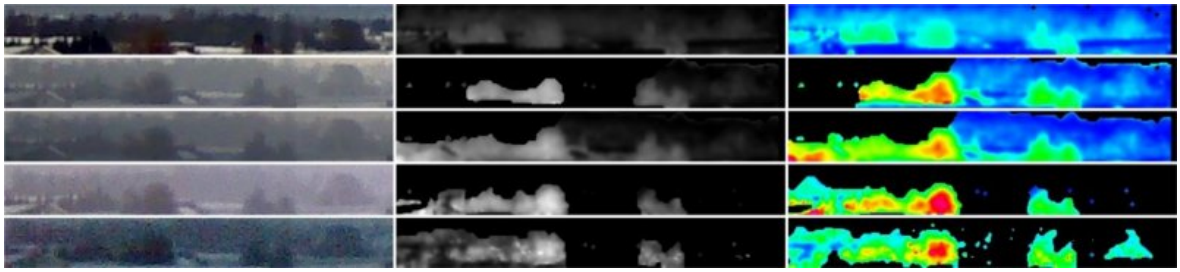
Aj keď, najmä pri objektoch blízkych horizontu, môže dôjsť k lokálnej inverzii mapy saliencie následkom vyhodnotenia oblohy ako významného objektu, táto inverzia na nami skúmaných vstupoch nastávala konzistentne s obrysmi objektov, objekt samotný bol na mape saliencie tmavší a vyššie spomínané zafixované body na mape saliencie mali síce inú intenzitu, tá však bola u všetkých bodov zmenená konzistentne.

Salientné objekty boli síce tvorené plochou intenzity odlišnej od svojho okolia, intenzita salientného objektu ani rozdiel intenzity objektu a jeho okolia však nebol zachovaný vo viacerých rôznych vstupoch, ba dokonca ako bolo spomínané, tento rozdiel bol niekedy aj záporný. To nás viedlo k vylúčeniu prahovania ako použiteľného prístupu, nakoľko práve meniace sa intenzity spôsobjú problém pri stanovovaní hodnôt prahov a prípadné inverzie vyradia z hry aj adaptívne prahovanie. Ďalším problémom boli iné salientné objekty v blízkom okolí skúmaného objektu, ktoré boli takto často zlúčené so skúmaným objektom. Nepraktickosť prahovania sme potvrdili aj experimentálne na niektorých testovacích sadách.

Obr. 3.1: Príklad sledovaného objektu počas rôznych podmienok. Možno pozorovať meniaci sa kontrast, miznúce detaily, zmeny na objekte, pohyb tieňov, ale aj zmenu ostrosti a farebného tónu.



(a) Garáž



(b) Les

Obr. 3.2: Výsledky použitia algoritmu grabcut boli nestabilné.

Pri opätovnom skúmaní možností segmentovania sme tiež došli k domnienke, že vhodne zvolené body v rozumnej vzdialenosti od okrajov pozorovaného objektu by mohli zaistiť istú odolnosť voči prípadným malým posunom objektu do tej miery, kým by zvolené fixné body pozadia a popredia objektu neprekročili hranice objektu, čím by sme mohli čiastočne poľaviť z požiadavky fixnej kamery, a teda by sa kládli menšie požiadavky na inštaláciu aj údržbu.

Zároveň, keďže saliencia zachovávala obrysy objektu vrámci možností poskytnutých ostrosťou konkrétneho vstupu, možno by nebolo potrebné skúmať samotný tvar objektu, ktorý by sme dostali zadarmo od saliencie, ale iba množstvo plochy, ktorú zaberá. V prípade, že by objekt nebol pozorovateľný na danom vstupe, segmentovanie by na základe mapy saliencie nemohlo správne určiť obrysy, a plocha pokrytá nesprávne vysegmentovaným objektom by bola príliš veľká alebo malá, lebo objekty nepozorvateľné kvôli hmle alebo obdobnému meteorologickému fenoménu sú na mape saliencie prekryté výrazne väčšou súvislou plochou vrámci ktorej sa intenzita mení len pozvoľne.

V našich úvahách sme nenašli iný nedostatok, ako problematické zoradenie vstupov podľa klesajúcej dohľadnosti, ktorá sa síce prejaví na postupnom rozmazávaní okrajov objektu, ťažko však predpokladať správanie sa ľubovoľného segmentačného algoritmu na takto zoradených vstupoch, najmä ak nám hrozí spomínaná inverzná saliencia.

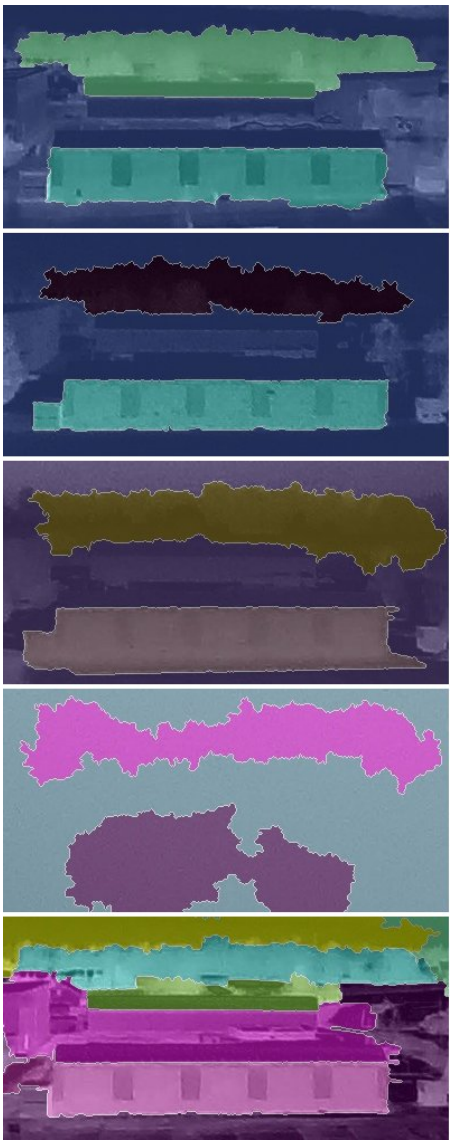
Aj napriek naším očakávaniam a nádeji v zostavenie rozumne fungujúcej miery, sme si vylámali zuby hneď na prvom kroku uvažovaného postupu. Nami použitý algoritmus

GrabCut[18] nebol schopných správne segmentovať isté objekty s neostrými hranami. Aj keď sme dokázali rozumne spracovávať blízke objekty ako sú napríklad budovy, tie vzdialené sa stali neriešiteľným problémom.

Algoritmus GrabCut nepoužíval iné parametre ako zoznam bodov s príznakom príslušnosti k objektu a citlivosť na hrany nastavoval dynamicky od konkrétnych vstupov. Ani po označeni veľkého množstva bodov, čím by sme potenciálne prišli o odolnosť voči afinným transformáciám, nedokázal tento algoritmus rozoznať ani ešte celkom salientné objekty ako vidno na obrázku 3.2.

Objekty boli často degradované iba na predom určené body prislúchajúce objektu. V prípade zníženej dohľadnosti a prekrytia skúmaného objektu súvislou nesalientnou plochou bol v súlade s našimi očakávaniami ako objekt vyhodnotený celý región záujmu prislúchajúci danému objektu.

Naše výsledky sa nám nepodarilo zlepšiť ani použitím algoritmu Watershed[13], ktorého výsledky sú na obrázkoch 3.4 a 3.3 a ktoré neprejavili významnejšiu závislosť na saliencii alebo viditeľnosti objektu.



Obr. 3.3: Algoritmus Watershed poskytol tak nestabilné výsledky, že by sme ich ťažko odlíšili od nenájdeného objektu.

Táto metóda tak dokázala fungovať len na niektorých objektoch, na tých pre ňu menej vhodných dokázala pracovať len na príliš úzkej množine vstupov, ktoré žiaľ neniesli vzhľadom na dohľadnosť nijaké významné spoločné vlastnosti. Fungovanie segmentačného algoritmu sa síce mierne zlepšilo ak sme zväčšili rozsah intenzít mapy saliencie ofarbením nepravými farbami, toto zlepšenie však nebolo postačujúce. Opustili sme teda ďalšie skúmanie tejto metódy.

Obr. 3.4: Výsledky algoritmu Watershed na inom vstupe. Obdobne ako na výsledkoch 3.3, aj na malom pozorovanom objekte sa výrazne prejavila nestabilita mnohých častí obrysu, ktorej napravenie spôsobilo veľké množstvo falošných výsledkov.

3.2 Detekcia charakteristických hrán

Aj keď sa nám predchádzajúcu metódu nepodarilo úspešne implementovať, rozhodli sme sa časť pozorovaní využiť iným spôsobom. Kým vtedy sme sa snažili vyťať súvislú plochu ohraničenú prechodom do odlišnej intenzity na mape saliencie, ďalej sme sa pokúsili využiť rovno hrany určené týmto prechodom. Aj keď mierne posunutie hrany v predchádzajúcom prístupe by nepredstavovalo výrazný problém, pre jej priamu detekciu to znamená významné komplikácie. Zároveň je potrebné riešiť úroveň zmeny intenzity, ktorú považujeme za hranu, ich prípadný posun spôsobený rozmazaním tak častým pri zníženej dohľadnosti, či naopak falošné hrany spôsobené šumom, či nevhodne nastavenými prahmi.

Analýzou vstupov pri odlišných podmienkach sme si všimli, že pri rôznych objektoch sa rôzne hrany podieľajú na rozoznateľnosti pozorovaného objektu inou váhou. Napríklad pri budovách mohol tieň zakryť detaily privrátenej steny, a tak sa stali okná, dvere, ale aj rohy tvorené privrátenou a bočnou stranou objektu nepozorovateľné. Naproti tomu bola kontúra strechy a často aspoň jednej strany budovy dobre rozoznateľná. Inak sa pri pozorovaní správali stĺpy a neprekvapuje, že sa javili ako vertikálne hrany. Dôležitý je aj fakt, že na viditeľnosti objektu sa podpísala aj jeho farba. V prípade hmly na pozadí sa môže biely objekt stať neviditeľným, aj keď poveternostné podmienky inak nezabraňujú jeho pozorovaniu. Vychádzajúc z týchto pozorovaní, dospeli sme k dvom dôležitým záverom.

Rozhodli sme sa brať do úvahy osobitne smer hrany, nakoľko pre rôzne vyzerajúce objekty môžu byť hrany v rôznych smeroch inak dôležité.

Ďalej sme sa vzdali akýchkoľvek ďalších pokusov o automatické určenie dôležitých hrán, nakoľko z jediného vstupu nie je možné bez dôkladnej znalosti prostredia, meteorologických fenoménov a správania sa osvetlenia určiť, ktoré vlastnosti objektu ho budú reprezentovať aj za zníženej dohľadnosti.

Nazdávame sa, že pre ľudského pozorovateľa nie je obzvlášť náročné riešiť spomínaný problém, naproti tomu, prípadné riešenia na báze umelej inteligencie by pri súčasnom stave tohto odvetvia potrebovali širokú trénovaciu množinu, ktorej získanie by mohlo predstavovať podstatné zdržanie pri prípadnom nasadení systému, ktorý by musel odpozorovať dostatočné množstvo fenoménov rôznych intenzít v iných častiach dňa pred tým, ako by začal poskytovať užitočný výstup. Nevylúčili sme však možnosť získať túto trénovaciu množinu napríklad pomocou simulácii. Rozhodli sme sa ale smerovať náš výskum iným smerom a voľbu vlasností dôležitých pre rozoznanie objektu prenechávame na používateľa pri kalibrácii systému.

3.2.1 Vyhľadanie dôležitých hrán

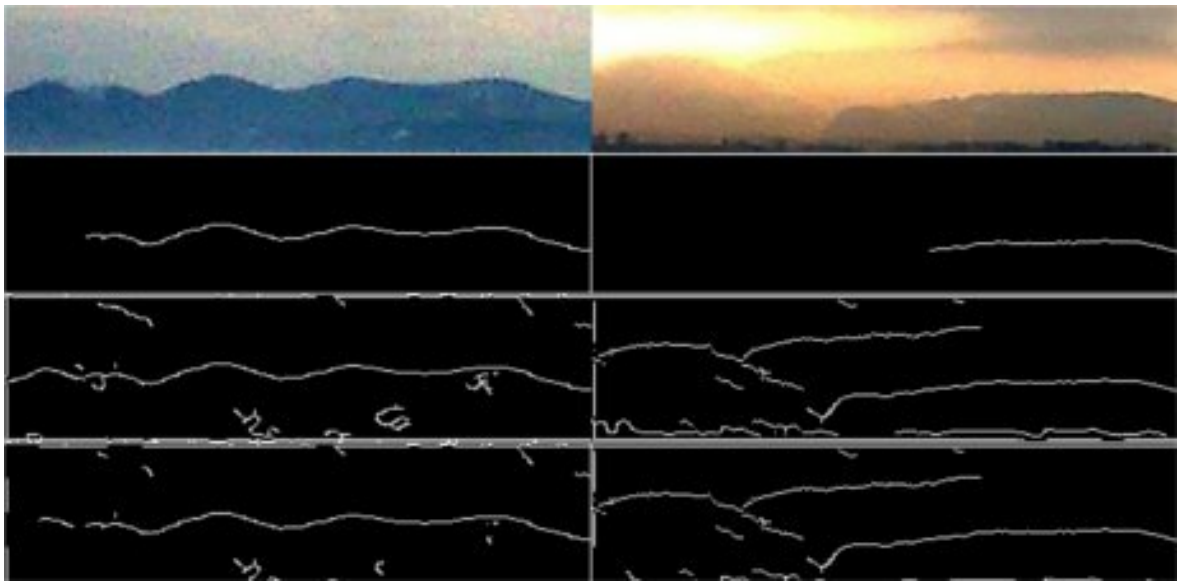
Na detekciu hrán sme sa rozhodli použiť Cannyho detektor hrán [14, Kapitola 4.2.5]. Medzi pre nás dôležité výhody patrí napríklad unikátna odpoveď na jednu hranu vo forme jeden pixel širokej čiary. Tento detektor bol totiž konštruovaný s dôrazom na to, aby neposkytol viac ako jednu odpoveď na jednu hranu. Hrany sa tak javia ako jeden pixel široké čiary, ktoré nám poslúžia na hrubé rozoznávanie objektov podľa ich tvaru.

Ani samotné použitie Cannyho detektora hrán sa neukázalo ako jednoduché. Detektor hrán používa na rozhodnutie o prítomnosti hrany tzv. hysterézne prahovanie [14, Kapitola 4.2.5], ktoré sa dá riadiť dvoma prahmi. Pri stanovovaní hodnôt týchto prahov sme dospeli k starému známemu problému a mali sme problém buď s chýbajúcimi hranami alebo naopak príliš výrazne prejavujúcim sa šumom. Aj problém stanovenia hodnôt prahov pre Cannyho detektor hrán sme sa rozhodli zamiesť pod koberec a riešiť ho tak, aby sme získali pre nás vhodné výsledky.

Nakoľko je výstupom nášho predspracovania mapa saliencie, ktorej výstupy sú síce do istej miery ťažko odhadnuteľné, platí však, že kontrastné, dobre pozorovateľné a blízke objekty sa od svojho okolia odlišujú výrazne vyššou intenzitou, naproti tomu vzdialené objekty pokrývajúce väčšiu časť zorného poľa sa prejavujú nielen menšou zmenou intenzity, ale aj menším množstvom im blízkych objektov a iných detailov, nakoľko tie sa strácajú v diaľke. Rozhodli sme sa teda zostaviť experiment, v ktorom by sme sa pokúsili spojiť do jednej dve množiny detekovaných hrán. Jedna by prislúchala blízkym výrazným a často aj nakopeným objektom s vysokým prahom Cannyho detektora, druhá vzdialeným, menej výrazným a viac rozprestreným objektom s nízkym prahom.

Ako však určiť, ktoré objekty sú blízke a ktoré vzdialené? Využijúc predpoklad, že u vzdialeného objektu vieme pozorovať oveľa menšie množstvo detailov, rozhodli sme sa vylúčiť hrany, ktoré naznačujú prítomnosť takýchto detailov, ktoré sa zase prejavili na zvýšení počtu hrán v okolí objektu. U spomínaných vzdialených menej výrazných objektov sa malé množstvo jemných a ostrých hrán prejavilo na tom, že aj pri nízkych prahoch hysterézneho prahovania bolo nájdených len málo hrán. Naproti tomu u blízkych objektov sa jemné prechody v mape saliencie prejavili na tom, že pri nízkych prahoch boli, dalo by sa povedať, že až vyplnené veľkým množstvom jemných a nestabilných hrán.

V našom experimente sme sa ako mieru detailnosti rozhodli použiť práve množstvo hrán nájdených na danej ploche. Vďaka vlastnostiam Cannyho detektora hrán sme si mohli dovoliť zrátať nenulové pixely a tie prehlásiť za príslušné hranám. Ak bola príliš veľká časť danej plochy pokrytá hranovými pixelmi, prehlásili sme danú oblasť za detailnú a ďalej sme pre ňu používali iba hrany nájdené pri vysokých prahoch. Ak naopak neprekročil počet hrán na jednotku plochy stanovený prah brali sme do úvahy aj



Obr. 3.5: Príklad detekcie výrazných aj menej výrazných hrán. V poradí zhora nadol sú pôvodný vstup, hrany detekované pri vysokých prahoch, hrany detekované pri nízkych prahoch a zlúčené už prefiltrované hrany. Detekcia menej výrazných hrán sa neprejaví na všetkých vstupoch.

málo výrazné hrany, ktoré síce neovplyvnili dobre viditeľné a výrazne salientné objekty, umožnili nám však do istej miery pozorovať aj tie menej salientné črty konkrétnej scény. Príklad fungovania tohto postupu je na obrázku 3.5.

3.2.2 Porovnávanie hrán so vzorom

Po spomínanom spracovaní vstupu by sme už mali disponovať dostatočne kvalitne vyextrahovanými hranami pozorovateľných objektov, potrebovali sme však zodpovedať otázku, či konkrétna množina hrán zodpovedá danému hľadanému objektu pričom dôležité hrany zadá do systému používateľ. Nie je prekvapivé, že sa výskumníci venovali aj tomuto problému, známemu aj pod názvom *Shape-Matching*. Istý prehľad niektorých známych metód ponúka [20]. Namiesto hrnutia sa do ďalšieho štúdia možných metód, rozhodli sme sa skúsiť najskôr implementovať vlastný spôsob namieru ušitý vlastnostiam našich vstupov.

Ako už bolo naznačené, u rôznych objektov sa môžu rôzne hrany podieľať na možnosti rozoznať tento objekt rôznou mierou. Tieto hrany pre konkréty objekt často vykazovali aj niektoré spoločné vlastnosti. Rôzne stĺpy nám umožnili detekovať prevažne vertikálne hrany, naproti tomu u budov a pohorí boli užitočnejšie horizontálne hrany. Pozorovali sme, že pri znižujúcej sa dohľadnosti sa u niektorých pozorovaných objektov isté hrany strácali skôr, respektíve boli horšie pozorovateľné pri určitých meteorologických

podmienkach. Príkladom môže byť strecha budovy, ktorá je dobre pozorovateľná, aj keď tieň zakrýva alebo posúva vertikálne obrysy, či isté detaily stĺpu, ktoré síce boli pozorovateľné pri dobrom počasí, avšak veľmi rýchlo sa strácali už pri miernom zhoršení podmienok.

Toto pozorovanie nás viedlo k domnienke, že by bolo rozumné rozlišovať medzi hranami v rôznych smeroch a prípadne im umožniť rôznou mierou podieľať sa na rozlišovaní daného objektu. Autori pracujúci s obdobnou myšlienkou [19] pracovali s celkovo štyrmi smermi - vertikálnym, horizontálnym a dvoma diagonálnymi. Naše experimenty však neskôr ukázali, že postačujúce je rozonávať, síce možno s mierne vyššou toleranciou, iba dva smery a to horizontálny a vertikálny.

Ďalším naším pozorovaním bolo, že vypočítané hrany nemajú vzhľadom na meniace sa podmienky stabilnú polohu, ktorá sa mohla vrámci rôznych vstupov líšiť o vzdialenosť rádovo v jednotkách pixelov. Tento posun mohol byť spôsobený jednak možnými pohybmi snímacieho zariadenia, ale aj samotným počasím. Bolo teda nutné navrhnúť spôsob, ako vyhľadávať aj mierne posunuté hrany, avšak môžeme si dovoliť predpokladať prítomnosť hrany v istej oblasti.

Obidva spomínané problémy s posunom hrany ale aj potrebou rozlišovať smer rieši tzv. *Gáborov filter*, ktorý môžeme pre naše potreby chápať ako zloženie sínusoidnej vlny a gaussiánu. Každá z týchto zložiek nám umožňuje riešiť jedenu zo spomínaných požiadaviek. Snáď nie je potrebné zdôrazňovať, že vhodnou voľbou parametrov je možné tento filter nastaviť tak, že bude reagovať len na hrany v určitom smere, danej hrúbky, či dokonca rozlišovať, či sa jedná o tenkú hranu podobnú tej z výstupu Cannyho detektora hrán alebo o prechod medzi dvoma rôznymi plochami.

Parametre Gáborovho filtra sa nám podarilo zvoliť tak, aby nielen rozlišoval medzi horizontálnymi a vertikálnymi hranami, ale ich aj mierne rozmazal. Rozmazanie sa ukázalo pri rošírení plochy patriacej hrane, ktorá bola zase dôležitá v neskorších fázach, ale aj pri zlúčení viacerých odpovedí na jednu hranu v dôsledku zašumených vstupov. Gáborov filter spolu s prahovaním nám tak umožnil ušetriť neskoršiu dilatáciu smerovaných hrán.

Na rozlíšenie smeru hrán sme použili dva Gáborove filtre s parametrami identickými až orientáciu sínusoidnej vlny. Výstupom teda boli dva súbory odpovedí na detekované hrany, pričom v jednom z nich je možné pozorovať len horizontálne a v druhej len vertikálne hrany. Na riešenie nášho problému hľadania zodpovedajúceho tvaru sme sa rozhodli použiť jednoduché šablóny zodpovedajúce očakávaným horizontálnym, respektíve vertikálnym hranám. Mieru zhody sme vypočítali z počtu pixelov, ktoré prislúchali niektorej z hrán jednak v šablóne, ale aj v aktuálnom medzivýsledku. Vo výsledku neboli zohľadnené pixely, ktoré boli v šablóne uvedené ako nehranové, a

to z dôvodu, že nás zaujíma iba prítomnosť hrany na danom mieste, hrany navyše nepovažujeme za problém. Ak však chýba nejaká hrana zo šablóny, ktorá nebola zdetekovaná, prejaví sa to na znížení hodnoty miery zhody.

Náš spôsob riešenia problému zodpovedajúcich tvarov sme teda riešili relatívne hrubým spôsobom. Zaujímala nás iba prítomnosť hrany na danom vstupe, pričom nijako nerozlišujeme, ktorému objektu prislúcha. Navyše sme pripustili do istej miery aj nepresnosti vo výpočtoch, najmä Cannyho detektora hrán. Prípadné problémy spôsobené hranami iných objektov, ktoré by sa vyskytli na mieste hrany skúmaného objektu, sa nám na našej relatívne širokej testovacej množine vstupov nepodarilo spozorovať. Vhodne umiesnené hrany takýchto cudzích objektov by mohli spôsobiť, že by ich náš algoritmus vyhodnotil ako viditeľné hrany skúmaného objektu, zrejme by však chýbali ostatné hrany a celková odpoveď miery zhody by bola nízka. Zdôrazňujeme ale, že kvôli nízkej pravdepodobnosti takejto situácie, nepovažujeme hrany navyše oproti šablóne ako problém.

3.2.3 Zhrnutie

Navrhovaný algoritmus využívajúci detekciu a následné porovnávanie spracovaných hrán sa teda dá zrhnúť do nasledovných krokov:

Algoritmus detekcie charakteristických hrán

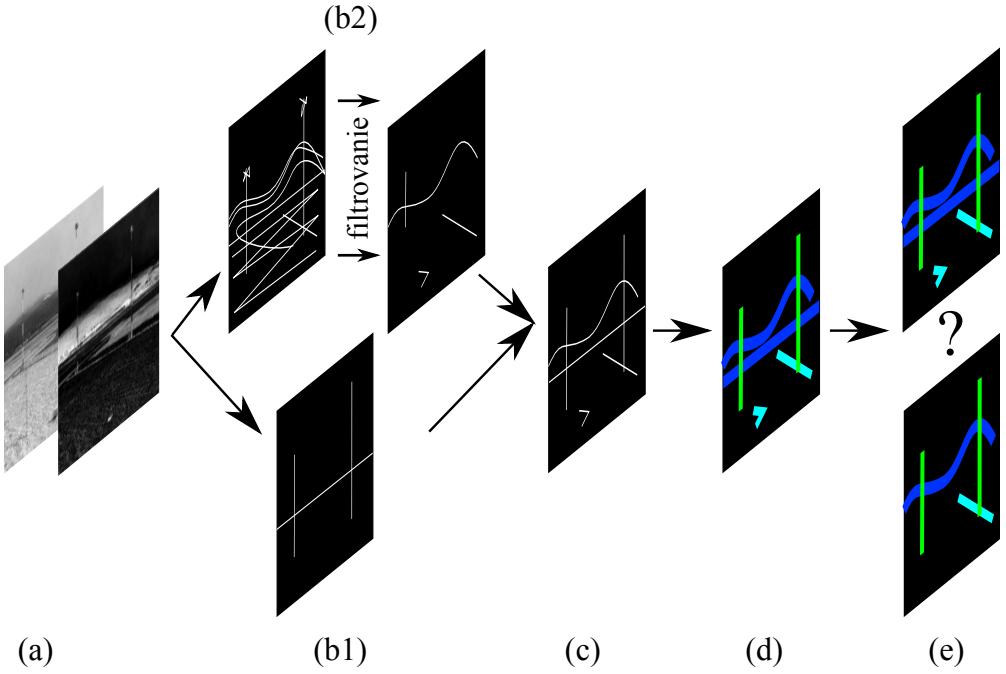
1. **krok: Výpočet mapy saliencie** 3.6(a)
   Použitím metódy fázového spektra na kvaterniónoch vytvoríme mapu saliencie.

2. **krok: Detekcia výrazných hrán** 3.6(b1)
   V mape saliencie vyhľadáme výrazné hrany pri vysokých prahoch hysterézneho prahovania Cannyho detektora hrán.

3. **krok: Pridanie osamotených nevýrazných hrán** 3.6(b2)(c)
   Okrem výrazných hrán budeme brať do úvahy aj objektov ako sú vzdialené pohoria, ktoré sú málo salientné kvôli farebnosti podobnej s okolím.

4. **krok: Filtrovanie hrán orientovaným Gáborovým filtrom** 3.6(d)
   Rozlíšime horizontálne a vertikálne hrany pomocou Gáborovho filtra, ktorý zároveň vykoná dilatáciu hrán, a tak zaistí čiastočnú odolnosť voči posunutiu.

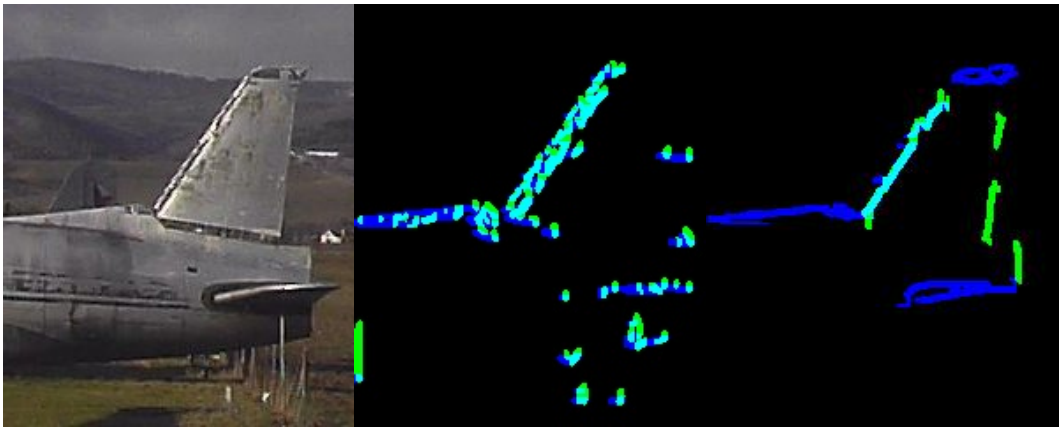
5. **krok: Porovnanie zhody so šablónou** 3.6(e)
   Výsledkom bude pomer počtu detekovaných a očakávaných hranových pixelov.



Obr. 3.6: Schéma výpočtu algoritmu detekcie hrán opísaného vyššie. Farebná notácia je rovnaká ako na obrázku 3.7.

Výsledkom tohto algoritmu na konkrétnom vstupe a v ňom zvolenom objekte je číslo, ktoré nám hovorí ako dobre sa nám podarilo nájsť zadané významné hrany objektu.

Po vykonaní predbežných testov nás potešili relatívne dobré výsledky tohto algoritmu. Pri skúmaní jeho možností sme odhalili, že lepšiu presnosť dosiahneme, ak vynecháme



(a) (b) (c)

Obr. 3.7: Príklady detekovaných hrán. Obrázok 3.7a je vstup, na 3.7b je výstup detekcie salientných hran, ktorý zlyhal na málo salientnom pozadí napravo od chvosta lietadla. Na 3.7c je výstup bez medzikroku výpočtu mapy saliencie.

výpočet mapy saliencie. Takýmto spôsobom sme vedeli pomocou metodiky testovania popísanej v kapitole 5 dosiahnuť zlepšenie približne o 10%. Na obrázku 3.7 vidíme rozdiel v nájdených hranách pri použití mapy saliencie a bez nej. Tieto problémy v detekcii zmejme spôsobovalo okolie objektov tým, že objekt sa nejavil dostatočne salientný voči svojmu pozadiu, a teda dekcia hrán zlyhala aj pri nízkych prahoch.

Aj napriek tomu, že vyššie načrtnutý algoritmus poskytuje celkom rozumné výsledky a poskytol nám rozumný základ na ďalší výskum, mali sme obavy, že objekty istých vlastností by mohli robiť problémy. Príkladom by mohli byť napríklad bodové zdroje svetla, ktorých hrany môžu byť ťažko detekovateľné respektíve ťažko odlíšiteľné od nesprávne identifikovaných hrán. Ďalej sme videli rezervy nášho prístupu detekcie málo výraznych hrán napríklad tých na okrajoch lesov, ktoré boli často v blízkosti iných objektov a teda len časť im príslušných hrán bola naozaj nájdená. Tieto nedostatky nás viedli k motivácii skúmať aj iné spôsoby detekcie hrán a následne navrhnuté algoritmy porovnať a pokúsiť sa z nich vybrať čo možno najlepší.

3.3 Mriežka odpovedí Gáborových filtrov

Stále v duchu vyhľadávania významných hrán v dopredu známom smere a mieste, rozhodli sme sa preskúmať možnosti Gáborovho filtra v oblasti ich detekcie. Pripomíname, že Gáborov filter vieme zložiť z priestorovej sínusoidnej vlny a Gaussiánu. Parametrami vieme upravovať fázu, frekvenciu a smer vlny a tiež šírku, výšku a eliptické natiahnutie Gaussiánu. Vopred známa očakávaná poloha hrán a chýbujúca potreba vysporiadania sa s prípadnými afinnými transformáciami by nám mohli umožniť navrhnúť hustý deskriptor v duchu algoritmu HOG, ktorý navrhli [2].

Ako už bolo spomínané, Gáborov filter môže pri vhodnej parametrov reagovať na hrany v požadovanej orientácii, hrúbke, šírke a aj ostrosti. Jeho schopnosti už boli skúmané a zaslúžil si pozornosť nielen odborníkov z oblasti počítačového videnia, ale aj expertov na vizualizáciu dát napr. [21, Kapitola 5]. Výskum, ktorý vykonal [3] naznačuje, že Gáborov filter je blízky spracovaniu informácie v skorých fázach ľudského videnia a preto by mohol byť aj biologicky plausibilným prístupom k riešeniu nášho problému. Taktiež sa stal cenným nástrojom na opis a analýzu textúr, nakoľko dokáže reprezentovať informáciu aj vo frekvenčnej doméne.

Náš výskum sa len letmo priblížil využitiu plného potenciálu Gáborovho filtra, možno však poslúži ako základ pre ďalšie skúmanie, či iné možnosti jeho využitia pri rozpoznávaní objektov za účelom merania vizuálnej dohľadnosti. Pre naše potreby budeme Gáborov filter chápať skôr ako elegantný nástroj schopný zastúpiť viacero známych a zaužívaných metód.

Aj keď gradient poskytuje cenné informácie o priebehu zmien intenzít v scéne, a tak poskytuje istú na svoje okolie normalizovanú informáciu, ba dokonca by sme vhodnou zámenou operátov zaužívaných na jeho konštrukciu mohli gradient parametrizovať, nazdávame sa, že Gáborov filter poskytuje kvalitnú alternatívu s bohatšími možnosťami rozšírenia napríklad na rôzne úrovne detailov či rôzne typy vlastností, na ktoré by sme sa chceli potenciálne zamerať.

3.3.1 Rozdelenie vstupu do mriežky

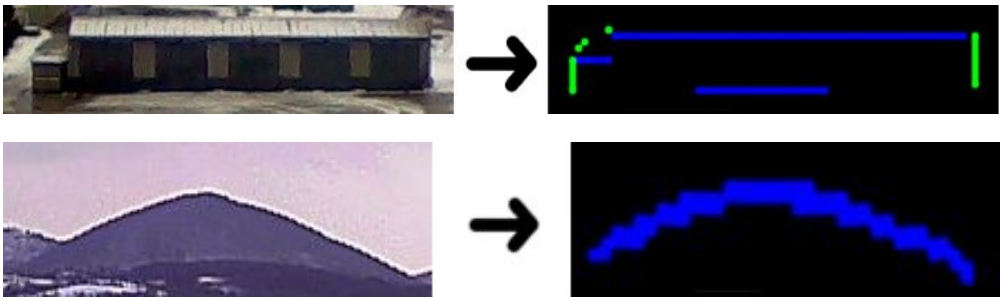
Základnou myšlienkou, ktorú sme sa rozhodli vypožičať si z algoritmu HOG [2], je vytvoriť popis hrán vyskytujúcich sa na istej malej ploche políčka mriežky, na ktorú sme rozdelili skúmaný vstup. Na tvorbu popisu jedného políčka sa následne použije histogram gradientu, ktorým dokázali autori algoritmu približne popísať hrany pomocou jedného vektora. Tieto vektory následne normalizovali vzhľadom na vektory z ich okolia v pôvodnej mriežke. Výsledny popis bol tvorený normalizovanými vektormi prekrývajúcich sa blokov, vrámci ktorých prebiehala normalizácia.

Z výsledkov, ktoré dosiahli [2], sa dá usúdiť, že aj tak komplikovaný útvar akým je ľudská postava sa dá všeobecne popísať a rozoznať len na základe vhodnej reprezentácie jeho siluety. Pokúsili sme sa teda navrhnúť algoritmus, ktorý by obdobným spôsobom reprezentoval význačné kontrúry objektu tak, aby sme mali dostatočnú toleranciu voči zmenám obrysov spôsobených vplyvom rôznych meteorologických fenoménov.

Našou snahou bolo navrhnúť čo možno najjednoduchší popis políčok, čo by sa mohlo v porovnaní s algoritmom HOG priaznivo prejaviť na rýchlosti nášho algoritmu a zároveň sme upustili od myšlienky využiť na záverečné rozhodnutie o prítomnosti objektu *Support Vector Machine*. Ako už bolo spomínané v predchádzajúcich kapitolách, nadobudnutie trénovacej množiny pre SVM by mohlo spôsobiť výrazné komplikácie pri nasadení systému a univerzálne natrénovaný klasifikátor považujeme pri súčasnej miere poznania za nereálny.

3.3.2 Výpočet popisnej hodnoty políčka

Obdobne ako [2] sme sa rozhodli rozdeliť skúmanú plochu do mriežky. Pre jednotlivé políčka je potrebné vypočítať dostatočne podrobný popis, ale zároveň si neželáme aby popisom políčka bol príliš dlhý vektor. Najjednoduchšia reprezentácia, ktorú sme ešte považovali za schopnú poskytnúť prakticky použiteľné výsledky, mala slúžiť ako miera prítomnosti hrán v ľubovoľnom smere. Realizovaná bola pomocou viacerých Gáborových filtrov orientovaných v rôznych smeroch a výsledkom bol súčet ich hodnôt. Predbežné experimenty však ukázali, že toto zjednodušenie je až prílišné a neposkytuje



Obr. 3.8: Príklady šablón budovy (hore) a pohoria (dole), modrá predstavuje očakávané horizontálne hrany, zelená vertikálne, tyrkysová by predstavovala prítomnosť obidvoch smerov v políčku. Aj keď sú detaily budovy pozorovateľné, do šablóny boli zaradené len hrany obrysov.

dostatočné informácie na odlíšenie skutočného objektu od šumu.

Pokúsili sme sa znovu využiť úvahy z predchádzajúceho prístupu a merať horizontálne a vertikálne hrany vyskytujúce sa na danom mieste a reprezentovať ich pomocou dvoch nezávislých popisov. Čiastočne využívajúc výsledky pri návrhu histogramov v algoritme HOG a ale aj našich vlastných pozorovaní, rozhodli sme sa ignorovať informácie o smere prechodu hrany z nízkej intenzity na vysokú. Dôvodom na toto rozhodnutie sú výrazne zmeny kontrastu, ktoré môžu spôsobiť, že objekt sa nebude vyskytovať pred pozadím výrazne meniacej sa intenzity.

Na výpočet hodnoty políčka sme skúmali niekoľko prístupov. Na základe našich experimentov sme usúdili, že dostatočne kvalitné výsledky vieme dosiahnuť, ak použijeme aritmetický priemer odpovedí Gáborovho filtra v danom políčku.

Narozdiel od [2] sme políčko mriežky v jednom smere popísali jedinou hodnotou a celkový popis políčka teda tvorí dvojrozmerný vektor. Z úvahy o možnosti vynechať informáciu o charaktere zmeny kontrastu na hrane nasledovalo aj presvedčenie, že aj samotná intenzita odpovede Gáborovho filtra nie je veľmi dôležitá, lebo ak povolíme kolísanie kontrastu do takej miery, že môže dôjsť k zmene smeru prechodu z vyššej intenzity na nižšiu, tak už aj samotný kontrast nie je veľmi dôležitý pokiaľ samozrejme vieme hranu rozoznať. Výsledkom tejto úvahy je teda nápad zjednodušiť dvojrozmerný vektor popisu políčka len na príznaky o prítomnosti hrany v danom smere. Tento krok sa dá zrealizovať jednoducho prahovaním odpovede Gáborovho filtra.

3.3.3 Zhrnutie

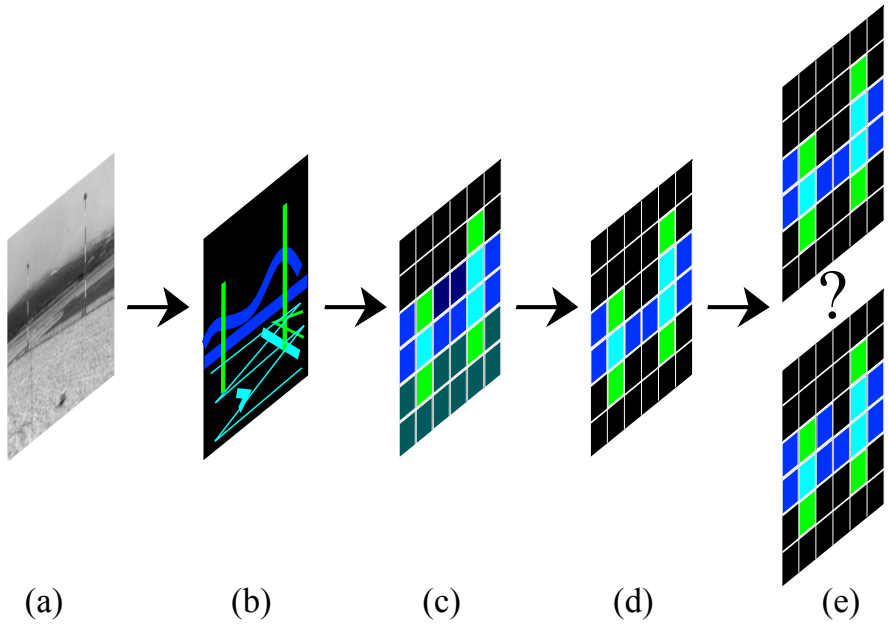
Tento algoritmus sme sa snažili navrhnúť tak, aby dokázal zohľadniť významné vlastnosti pozorovaného objektu, ktorými sú aj horizontálne a vertikálne hrany. Scénu sme sa rozhodli analyzovať pomocou jej rozdelenia do mriežky a využitím Gáborovho filtra

sme popísali jej jednotlivé políčka.

Výsledný deskriptor konštruovaný týmto algoritmom teda vyzerá ako dve matice, ktorých prvky indikujú prítomnosť hrany v horizontálnom, respektíve vertikálnom smere. Obdobne ako deskriptor z časti 3.2 aj veľkosť deskriptora tohto algoritmu je závislá od konkrétneho pozorovaného objektu a jeho veľkosť nijako neobmedzujeme ani nenormalizujeme. Príklad deskriptora je na obrázku 3.8. Obdobne ako algoritmus detekcie hran prezentovaný v predchádzajúcej časti, aj tento predpokladá, že zaujímavé hrany objektu zadá skúsený odborník tak, aby ich viditeľnosť charakterizovala viditeľnosť celého objektu.

Našim pôvodným zámerom bolo použiť vyššie opísaným spôsobom Gáborov filter spolu s mapami saliencie, ktoré sa nám osvedčili v prístupe z časti 3.2 nakoľko sme sa nazdávali, že analýza salientných objektov z tejto mapy bude menej náročná ako priama analýza vstupu. Prekvapilo nás však, že opísaný deskriptor konštruovaný za pomoci Gáborovho filtra podáva lepšie výsledky, ak mapu saliencie nepoužijeme. Tento výsledok považujeme za prekvapujúci najmä preto, lebo v mape saliencie sú skúmané objekty často výraznejšie oproti ich nesalientnému pozadiu.

Po analýze problematických vstupov sme prišli k záveru, že použitie Gáborovho filtra spolu s mapou saliencie dávalo kvalitné výsledky najmä na blízkych objektoch, ktoré sa často javili ako salientné. Avšak u vzdialených objektov sa opäť prejavila ich nízka saliencia a často boli naším algoritmom prehliadnuté. Výsledkom teda bolo, že spojenie načrtného algoritmu a mapy saliencie dávalo uspokojivé výsledky len na



Obr. 3.9: Schéma výpočtu algoritmu mriežky odpovedí Gáborových filtrov.

určitých triedach objektov. Naproti tomu použitie algoritmu na pôvodnom vstupe dokázalo poskytnúť uspokojivé výsledky na všetkých nami skúmaných vstupoch a v porovnaní s predchádzajúcim prístupom prinieslo ešte výraznejšie zlepšenie.

Algoritmus rozpoznávania objektov v okolí využívajúcu mriežku hodnôt Gáborových filtrov môžeme zhrnúť do nasledovných krokov:

Algoritmus mriežky odpovedí Gáborových filtrov

---

1. **krok: Filtrovanie vstupu Gáborovým filtrom** 3.9(b)
   Vstup spracujeme Gáborovým filtrom v horizontálnom aj vertikálnom smere, pričom analyzujeme hrany medzi oblasťami rôznych intenzít. V každom smere použijeme Gáborove filtre sínusiodného priebehu s posunom o pol periódy a ich odpovede sčítame. Výsledkom sú odpovede filtrov v dvoch smeroch.

2. **krok: Rozdelenie vstupu do mriežky** 3.9(c)
   Odpovede filtrov v horizontálnom a vertikálnom smere spriemerujeme vrámci neprekrývajúcich sa štvorcových políčok. Výsledkom tohto kroku sú dve matice, ktorých prvky zodpovedajú políčkam, na ktoré sme rozdelili odpovede Gáborovych filtrov.

3. **krok: Prahovanie** 3.9(d)
   Do úvahy budeme brať len prvky matice nad hodnotu indikujúcu prítomnosť hrany v políčku.

4. **krok: Porovnanie zhody so šablónou** 3.9(e)
   Ako výslednú metriku použijeme pomer počtu detekovaných hranových pixelov prekrývajúcich sa so šablónou a celkového počtu hranových pixelov v šablóne.

Predbežné experimenty naznačili sľubné využitie vyššie popísaného algoritmu. Okrem parametrizovania pomocou rôznych Gáborových filtrov je možné fungovanie algoritmu vyladiť pomocou prahu minimálnej odpovede Gáborovho filtra na políčko. Tento algoritmus dokázal fungovať veľmi dobre a podarilo sa mu zavše nájsť aj veľmi zle viditeľné objekty. Na základe našich experimentov sme ale usúdili, že dostatočne kvalitné výsledky vieme dosiahnuť pevne zvolenými parametrami Gáborovho filtra a stanovením prahu odpovede na hodnotu blízku minimálnemu potrebnému kontrastu na rozpoznanie kontúry.

Kapitola 4

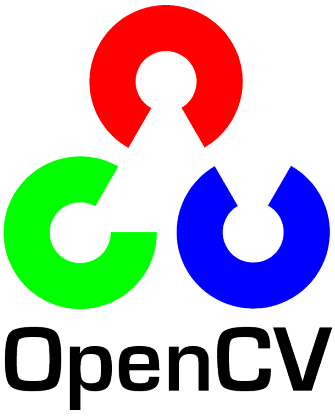
Implementácia navrhnutých algoritmov

V tejto kapitole podrobnejšie popíšeme detaily praktickej implementácie postupov spomínaných v predchádzajúcich kapitolách. Spomenieme problémy, s ktorými sme sa museli vysporiadať a objasníme metódy, ktorými sme sa ich rozhodli riešiť. Ďalej opíšeme podrobnosti použitia známych nástrojov na riešenie očakávaných problémov spojených s praktickou realizáciou teoreticky navrhnutých algoritmov.

4.1 Technológie implementácie

Pred voľbou samotného programovacieho jazyka, v ktorom by sme implementovali navrhované algoritmy, padlo rozhodnutie použiť knižnicu algoritmov a metód z oblasti počítačového videnia OpenCV. Táto knižnica implementuje veľké množstvo všeobecných algoritmov a pre potreby počítačového videnia všeobecne potrebných štruktúr, ako aj príslušných metód na prácu s nimi. Využitie tejto knižnice značne urýchlilo vývoj, nakoľko nebolo potrebné vyvíjať všeobecne známe postupy, jej použitie tiež umožnilo jednoducho pracovať so štruktúrami reprezentujúcimi vizuálnu informáciu a tak dalo možnosť spracovávať údaje na abstraktnej ale aj nízkej a technickej úrovni.

Knižnica OpenCV má dobrú reputáciu a teší sa veľkej obľube aj vďaka postykovaným možnostiam ale aj kvôli jej kvalitnej implementácii zabezpečujúcej rýchle a spoľahlivé spracovanie údajov. Knižnica poskytuje nepreberné množstvo známych algoritmov, dátových štruktúr a pomocných nástrojov dôležitých nielen pre počítačové videnie ale aj pre strojové učenie či klasifikáciu údajov. Knižnica tiež umožňuje naplno využiť hardvérový



potenciál počítačov pomocou viacvláknových rozšírení, či implementácii niektorých algoritmov na nasadenie na špecializovanom hardvéri grafických kariet.

Po rozhodnutí použiť knižnicu OpenCV bolo už viac menej priamočiare zvoliť ako jazyk implentácie C++, ktorý nie je potrebné bližšie predstavovať. Použitie knižnice OpenCV je vďaka existencii wrapperov možné aj s inými jazykmi ako je napríklad Java, ich možnosti však nedosahujú plný potenciál tejto knižnice a mohli by spôsobiť nepríjemné technické komplikácie.

Väčšina nášho vývoja prebiehala s použitím stabilnej verzie OpenCV 2.3.1, skoršie fázy sme implementovali s pomocou verzie 2.2.0. V čase ukončenia vývoja našej práce je prístupná beta verzia 2.4.0, čo svedčí o živom vývoji knižnice a jej bohatom potenciáli v budúcnosti. Vývoj prebiehal na operačnom systéme Ubuntu prevažne vo verziách 11.04 a 11.10. Inštalácie knižnice na tento operačný systém nebola triviálna, s použitím dostupných internetových zdrojov ju však bolo možné zvládnuť. Ako vývojové prostredie sme použili Netbeans vo verzii 7.0.1. Vývoj sme nezamerali na špecializovaný hardvér a cielili sme ho na bežnú platformu osobných počítačov.

4.2 Návrh systému

Samotná štruktúra aplikácie, ktorú sme na experimentálne účely vyvinuli nie je z pohľadu sofvérovej architektúry nijako zložitá. Až na výnimočné prípady sme si vystačili so štandardnými štruktúrami knižnice OpenCV ako sú rôzne typy matíc. Algoritmy navrhované v kapitole 3 sme teda mohli implementovať bez strát na prehľadnosti či rozšíriteľnosti kódu ako jednoduchý súbor funkcií, ktoré je možné viac menej nezávisle používať.

Narozdiel od samotných algoritmov, ktorých výskum bol cieľom tejto práce, sa podporné nástroje ukázali ako relatívne zložité a bolo potrebné navrhnúť viacero rozšíriteľných tried, ktoré umožnili abstrahovať napríklad nad konkrétnym spôsobom voľby hrán šablóny najvhodnejšieho pre daný algoritmus. V nasledujúcich častiach opíšeme problémy, ktoré sme museli vyriešiť, ako aj možnosti znovupoužiteľnosti nami navrhnutého systému pre prípadné rozšírenie.

4.2.1 Používateľské rozhranie

Návrh používateľského rozhrania bol sprevádzaný viacerými komplikáciami. Jednou z nich je, že knižnica OpenCV disponuje prostriedkami na zobrazovanie grafického výstupu a spracovanie vstupov z klávesnice či myši, ovládacie prvky poskytuje len v minimálnej miere a sofistikované grafické rozhranie by bolo zrejme potrebné implementovať pomocou iných prostriedkov.

Nepríjemnosť, na ktorú sme pri vývoji grafického rozhrania narazili, je, že rozhranie poskytované OpenCV nám neumožnilo jednoducho spracuvávať súvisle trvajúce akcie ako je napríklad ťahanie myšou a priebežné vykresľovanie obdĺžnika znázorňujúceho zvolenú oblasť. Pokusy spracovať takéto udalosti viedli k plnému vyťaženiu prostriedkov procesora a k značne oneskorenej odozve systému. Tento problém sa nám žiaľ nepodarilo riešiť len s pomocou prostriedov OpenCV. Vystačili sme si teda s ovládaním aplikácie len pomocou jednoduchých kliknutí myšou. Náš návrh ale umožňuje obísť tento problém nami zvolenej knižnice a na grafické rozrhanie využiť iné prostriedky bez straty funkcionality a s využitím výraznej časti existujúceho kódu.

Pre účely našich experimentov však boli možnosti knižnice OpenCV dostačujúce a pokúsili sme sa navrhnúť používateľské rozhranie tak, aby bol náš systém čo možno najľahšie zaintegrovateľný do iných systémov a vyabstrahovať logiku vykonávaných činností od konkrétnych možností spracovania používateľského vstupu.

Rozhodli sme sa navrhnúť abstraktnú triedu zastrešujúcu rôzne akcie vykonávané na zobazovanej scéne, táto trieda poskytuje funkcie na spracovanie stlačených klávesov a tlačidiel myši. Ďalej poskytuje možnosť získať odozvu systému v podobe bitmapy určenej na zobrazenie a po skončení používateľských akcii vrátiť vystup v podobe matice, ktorá už závisí od konkrétneho účelu konkrétnej implementácie.

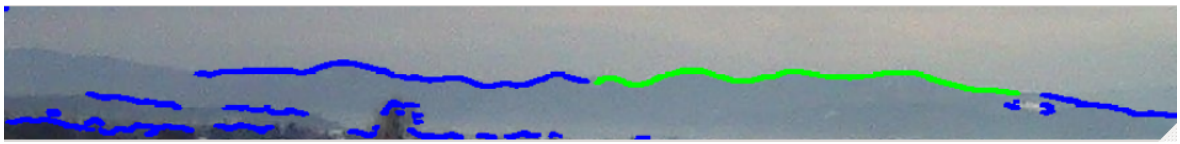
Implementovali sme niekoľko konkrétnych realizácii tejto abstraktnej triedy:

Výber regiónu záujmu Táto implementácia má používateľovi vybrať zaujímavý región na veľkom vstupe a tak zjednodušiť jeho ďalšie spracovanie zobrazením len relevantnej časti vstupu.

Výber detekovaných plôch Algoritmus popísaný v časti 3.2 rozlišuje horizontálne a vertikálne hrany, ktoré pred samotným porovnaním s tvarom zo šablóny spracuje do podoby malých súvislých plôšok, ktoré je možné zvoliť ako zaujímavé. Okrem toho táto implementácia umožňuje zvoliť len časť nájdenej plochy vo zvolených menších regiónoch.

Výber zaujímavých pixlov Pre účely algoritmu skúmaného v 3.3 je potrebné zvoliť políčka, na ktorých má Gáborov filter poskytnúť dostatočnú odpoveď na hranu. Ostané políčka mriežky v šablóne ignorujeme.

Výber bodov pre GrabCut Pred segmentovaním vstupu týmto algoritmom musíme zvoliť niekoľko bodov označených podľa toho, či sme si ich príslušnosťou do popredia alebo pozadia istí alebo nie. Na základe pozície týchto bodov segmentuje vstup a prípadne ho umožní segmentovať v ďalších iteráciách.



(a) Detekcia hrán

(b) Odpovede Gáborovho filtra

Obr. 4.1: Príklad použitia grafického rozhrania na výber zaujímavých hrán pre ich detekciu v mape saliencie (hore) a výber zaujímavých políčok mriežky odpovedí Gáborovho filtra.

Aby sme funkcionalitu implementovanú v spomínaných triedach vedeli využiť vytvorili sme jednoduchú triedu, ktorá tvorila fasádu medzi používateľom a realizujúcou triedou. Táto trieda už využívala konkrétne prvky grafického rozhrania poskytované OpenCV. Okrem inicializácie funkcii zachytávajúcich vstup od používateľa ma za úlohu tento vstup podať implementačnej triede ako buď udalosť myši alebo stlačenie klávesu. Po spracovaní vstupu implementačnou triedou zobrazí aktuálnu odozvu systému. Dúfame, že zmena prvkov grafického rozhrania si nevyžiada nič iné ako nahradenie funkcionality tejto triedy.

V závislosti od informácie, ktorú potrebujeme od používateľa, teda vytvoríme potrebnú triedu akcie a tú podáme fasáde. Po spracovaní vstupu sa môžeme rozhodnúť, či budeme požadovať ďalšie vstupy a proces opakovať, alebo budeme ďalej pokračovať spracovaním vložených údajov. Trieda fasády ale aj abstraktná trieda akcie totiž implementujú jednoduchý mechanizmus na určenie toho, či je vstup používateľa finálny alebo naopak plánuje vykonávať ďalšie akcie.

Využitím spomínaných tried sme boli schopní realizovať grafické rozhranie, ktoré umožňuje používateľovi jednoduchým spôsobom vytvoriť šablóny potrebné pre fungovanie algoritmov navrhovaných v kapitole 3. Pre mierne zvýšenie komfortu pri používaní nášho rozhrania sme umožnili v relevantných prípadoch prepínať medzi pohľadom na scénu a detekovanými zaujímavými vlastnosťami.

4.2.2 Testovacie nástroje

Pre účely vývoja a ladenia navrhovaných algoritmov ako aj kvôli nutnosti záverečného vyhodnotenia presnosti algoritmov bolo potrebné zostrojiť sadu nástrojov a funkcií, ktoré

by ich umožnili jednoducho spúšťať a vizualizovať ich výstupy za účelom vyhodnotenia úspešnosti. Za týmto účelom sme realizovali niekoľko funkcií sériovo spúšťajúcich skúmané algoritmy na zadanej množnie vstupov a popisov objektov, pričom výstupy zapisovali do súborov so zadaným formátom.

Jeden výstupný súbor tak obsahoval pre pozorovanú scénu a na nej skúmaný objekt niekoľko riadkov, pričom na každom z nich bolo poradové číslo konkrétnej skúmanej scény a hodnota odpovede analyzovaného algoritmu. Pre jedno spustenie testov bolo vygenerovaných toľko súborov, koľko bolo pozorovaných objektov.

Zvolený formát súborov umožnil jednoduchú vizualizáciu aj veľkého množstva odpovedí rôznych algoritmov na rôznych pozorovaných objektoch a ľubovoľne ich pri tejto vizualizácii kombinovať, čím sme mohli jednoduchšie pozorovať a porovnávať výhody ale aj nedostatky skúmaných postupov.

Kým spomínaná sada testov bola postačujúca pri vývoji a analýze výsledkov, záverečné vyhodnotenie úspešnosti a tiež skúmanie vhodných hodnôt prahov viditeľnosti si vyžiadalo komplikovanejšiu a perzistenciu výsledkov podporujúcu sadu testov. Bolo totiž potrebné pre neskoršiu analýzu uchovať hodnoty odpovedí testovaných algoritmov ako aj očakávané hodnoty viditeľnosti zadané ľudským pozorovateľom.

Na tento účel sme realizovali testovaciu triedu, ktorá využitím možností perzistencie knižnice OpenCV archivovala človekom zadané rozhodnutia o viditeľnosti daného objektu na konkrétnom vstupe, a ktorá zároveň umožňovala tieto údaje nielen zadať, ale aj uložiť a načítať z úložiska.

4.2.3 Formát ukladania údajov

Všetky nami skúmané algoritmy rozpoznávania objektov v okolí nutne potrebovali v nejakej forme udržiavať viac či menej zložité údaje o pozorovaných objektoch. Bolo teda nutné zabezpečiť ich perzistenciu medzi jednotlivými spusteniami algoritmov. Medzi potrebné údaje o objektoch patria ich poloha v scéne, ktorú vieme zapísať pomocou štyroch hodnôt: polohy na osi x a y, šírky objektu a jeho výšky. Algoritmy navrhnuté v častiach 3.2 a 3.3 potrebovali pre správne fungovanie aj používateľom zadanú šablónu objektu, či prahy hodnôt odpovedí, od ktorých je možné považovať objekt za viditeľný, pre potreby vykonania finálneho rozhodnutia o viditeľnosti objektu.

Použitý systém ukladania dát teda musel zvládnuť uložiť všetky spomenuté údaje a pokiaľ možno umožniť k nim jednoduchý prístup a manipuláciu s nimi. Knižnica OpenCV nám bola užitočná aj v tomto aspekte, nakoľko postkytuje možnosti perzistentného ukadania dát. Knižnica disponuje podporou pre dva úložné formáty a to *XML* a *YML*, pričom umožňuje ekvivalentne jednoduchú a pohodlnú prácu s obidvoma z týchto formátov. Jediný nedostatok, na ktorý sme narazili bola nemožnosť pomenovať jednotlivé

prvky zoznamov pri použití formátu *XML*. OpenCV ich ukladalo ako elementy „<_/>“, čo sme nepovažovali za vhodné.

Ako úložný formát sme teda zvolili formát *YML*, pričom v prípade potreby je konverzia medzi týmito formátmi jednoduchá. OpenCV poskytuje nástroje umožňujúce abstrahovať nad konkrétnym úložným formátom a umožnilo takmer bezproblémové ukadanie popisov objektov. Pri ukladaní sme sa museli ale vysporiadať s menšou chybou v knižnici, ktorá nenačítala údaje správne, ak boli jednotlivé popisy uložené v skrátenej notácii čo sme riešili tak, že sme skrátenu notáciu v súbore jednoducho nepoužili nakoľko nemá žiaden ďalší význam.

So zápisom popisujúcich údajov súvisí aj ich reprezentácia v programovacom jazyku. Pre naše potreby sa ukázalo ako dostatočné použiť klasickú C štruktúru, ktorá obsahovala nasledovné:

- Celočíselné súradnice objektu
- Celočíselné rozmery objektu
- Šablóna konkrétného algoritmu ako matica knižnice OpenCV
- Celočíselny rozmer políčka mriežky pre algoritmus z časti 3.3
- Prah minimálnej hodnoty odpovede algoritmu ako číslo v plávajúcej desatinnej čiarke

Tieto informácie umožňujú v konkrétnom vstupe identifikovať tzv. *Region of Interest* alebo tiež oblasť záujmu, v ktorej sa nachádza pozorovaný objekt, šablóna umožní porovnať aktuálne identifikované hrany objektu s tými, ktoré očakávame. Hodnota rozmeru mriežky má síce zmysel len pre použitie v algoritme Gáborových filtrov, ale umožní jej rozšírenie na rôzne úrovne detailov. Nakoniec prah minimálnej hodnoty určuje minimálnu odpoveď zhody šablóny s aktuálne roznaným útvarom, ktorú ešte považujme za viditeľnosť objektu.

Údaje ukladáme do súborov tak, že jeden súbor zodpovedá jednej analyzovanej scéne a obsahuje jeden alebo viac popisov pozorovaných objektov, ktoré sú uložené v koreňovom elemente súboru ako zoznam elementov popisujúcich jeden pozorovaný objekt. Načítanie a zápis do perzistentných dátových štruktúr umožňuje knižnica OpenCV tak pohodlne, že nie je potrebné dbať na poradie jednotlivých atribútov popisu objektu, dôležitý je iba ich názov. V zoznamoch je však dôležité dbať na poradie ich prvkov, lebo sme vynechali ľudsky čitateľný popis objektu a spätné určenie toho, o ktorý objekt sa jedná by inak ako na poradí v zozname bolo potrebné ručne vykonať na základe polohy objektu v scéne alebo samotnej šablóny. Taktiež je potrebné aby boli šablóny vrámci jedného súboru konzistentné a zodpovedali jednému zvolenému algoritmu.

Pre naše potreby spracovania takýchto databáz popisov objektov sme vyvinuli funkciu, ktorá prečíta súbor s daným názvom a popisy v ňom uložené preloží do vektora štruktúr vhodných pre ďalšie spracovanie. Taktiež sme vyvinuli súvisiacu funkciu, ktorá naopak daný vektor štruktúr uloží do súboru s daným názvom. Názov súboru čítaného týmito funkciami je jedným z argumentov konzolovej aplikácie, ktorú sme vyvinuli.

Pre potreby prípadného praktického nasadenia systému nevylučujeme potrebu ukladať popisy objektov aj v inej forme, ako do nami zvoleného formátu súborov. Pre potreby spracovania ďalšími funkciami nie je formát ukladania dát dôležitý, ak z neho dokážeme skonštruovať spomínané štruktúry jazyka C.

4.3 Konzolová aplikácia

Praktickou realizáciou cieľa tejto práce je konzolová aplikácia, ktorá slúži ako koncept využiteľnosti počítačového videnia pre potreby merania vizuálnej dohľadnosti. Táto aplikácia slúži len ako fasáda medzi používateľom a funkciami knižnice realizujúcej navrhované algoritmy a poskytuje jednoduchú možnosť vizualizácie výstupov algoritmov.

Konzolová aplikácia aplikácia umožňuje využiť dva algoritmy schopné poskytnúť prakticky relevantnú informáciu a získať ich odpoveď na konkrétnom vstupe. Aplikácia požaduje celkovo tri parametre, konkrétne typ algoritmu, cestu k súboru, ktorý obsahuje popis pobjektov podľa časti 4.2.3, a na záver cestu ku konkrétnej snímke scény. Výstupom je postupnosť odpovedí v poradí popisov objektov v súbore, ktorá hovorí, či bolo podľa daného popisu možné v zadanej scéne rozoznať popisovaný objekt. Aplikácia tiež podporuje nepovinny štvrtý argument, ktorého zadanie spôsobí, že namiesto vyhodnotenia viditeľnosti objektov sa zobrazí hodnota miery zhody nájdeného objektu s jeho šablónou.

Pre účely konštrukcie databázy popisov rozoznávaných objektov aplikácia tiež umožňuje namiesto typu algoritmu zadať typ popisov, ktoré hodláme vložiť do databázy. Ostatné povinné parametre sú zhodné ako v predchádzajúcom prípade t.j. cesta k súboru, do ktorého budú popisy uložené a cesta k snímke scény, pomocou ktorej zostrojíme šablóny pozorovaných objektov. Je vhodné aby bola daná snímka zhotovená za vysokej dohľadnosti a aby na nej boli všetky požadované objekty pozorovateľné. Nepovinný parameter umožňujúci voliť formu výstupu pri tomto použití nemá význam.

Na získanie presných inštrukcií ako používať konzolovú aplikáciu ju stačí spustiť bez parametrov.

4.3.1 Praktické aspekty nasadenia

V tejto časti popíšeme podmienky použitia konzolovej aplikácie a kroky súvisiace s jej nasadením potrebných na to, aby poskytovala relevantné informácie.

Okrem nutnosti dodržania odporúčaní na voľbu objektov vhodných na meranie vizuálnej dohľadnosti, z ktorých niektoré boli spomínané v kapitole 1.2.3, je potrebné zobrať do úvahy aj možnosti použitej kamery a vlastnosti pozorovaných objektov.

Úlohou experta počas prípadného nasadzovania systému je nielen zvoliť objekty vhodné na pozorovanie, ale aj vrámci týchto objektov určiť ich význačné prvky, podľa ktorých podľa ktorých bude tento objekt rozpoznávaný. Tieto prvky musia ostať viditeľné aj počas zhoršených podmienok, ak samozrejme považujeme za týchto podmienok objekt ako viditeľný. Ako vhodné považujeme hrany siluety objektu voči kontrastnému pozadiu, svetlá, prípadne iné, ale vyrazné a stabilné hrany prislúchajúce objektu.

Naopak ako nevhodné považujeme obrysy strechy, najmä ak môže byť zasnežená a je pozorovaná voči zemi. Obdobne sú nevhodné príliš biele, alebo naopak tmavé objekty pozorované voči zemi či asfaltovému pozadiu, nakoľko aj tieto môžu splynúť s pozadím, a tak spôsobiť nesprávne zníženú odpoveď navrhovaných algoritmov. Ďalej je nevhodná jemná textúra budov, ktorá sa môže stratiť v tieni v istej dobe dňa. Ako nestabilné sa tiež ukázali niektoré hrany rozhrania na rozhrani objektu a miesta jeho styku so zemou, nakoľko tieto boli až príliš ovplyňované snehom a námrazou, nevylučujeme však ich vhodnosť v istých prípadoch.

Ďalšou úlohou experta pri nasadzovaní systému, ktorú sme na neho delegovali, je stanovenie prahu viditeľnosti pre konkrétne objekty. Naše testy z kapitoly 5 ale naznačujú, že tieto prahy bude možné vybrať z niekoľkých predom stanovených intervalov.

Ako beh na dlhú trať tiež vyhodnocujeme samotné určenie dohľadnosti, nakoľko výstupmi našich algoritmov sú len jednoduché informácie o viditeľnosti daných objektov, či naopak nemožnosti ich pozorovania. Správne vyhodnotenie týchto výstupov požaduje metodiku zohľadňujúcu šum, nepresnosti vo výsledkoch našich a možno aj iných algoritmov, nárazové a krátkodobé zmeny viditeľnosti a ich relevanciu k prevládajúcej dohľadnosti. Výstupy našich algoritmov totiž môžeme chápať iba ako výstup nejakého senzora, ktorý je treba následne vyhodnotiť, a obdobný prístup nie je v leteckej meteorológii neznámy[1].

[1]Príkladom môže byť systém LLWAS popísaný v americkom patente číslo 5 221 924, ktorý sofistikovanými algoritmami filtruje a analyzuje výstupy snímačov rýchlosti vetra.

Kapitola 5

Vyhodnotenie úspešnosti navrhnutých algoritmov

V tejto kapitole zhrnieme výsledky navrhovaných algoritmov a vyhodnotíme ich úspešnosť na reálnych dátach. Zmienime sa nielen o presnosti výsledkov a výkonnostných aspektoch navrhovaných riešení, ale budeme analyzovať aj ďalšie možnosti využitia skúmaných algoritmov. Cieľom testovania bolo overiť reálnosť využitia spomínaných riešení na skutočných vstupoch z oblasti potenciálneho nasadenia, a tak sa čo najviac priblížiť podmienkam fungovania systému.

Spomenieme taktiež problémy, ktoré sme počas testovania odhalili a načrtneme postup ich riešenia.

5.1 Vstupy a metodika testovania

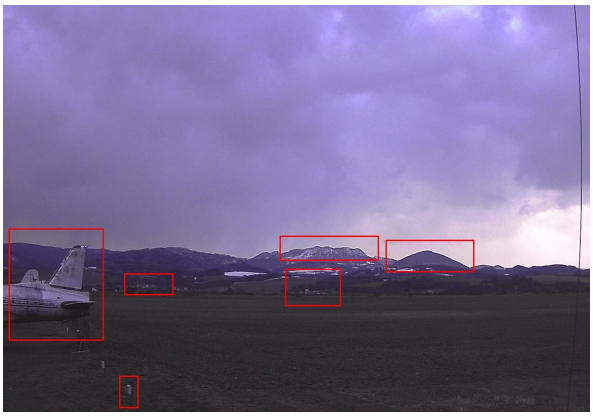
Pri návrhu algoritmov a vyhodnocovaní ich presnosti sme použili celkovo štyri množiny vstupov zhotovených na letiskách, nakoľko tie sú miestom praktického nasadenia systému. Vstupy sme v súlade s praktikami v oblasti strojového učenia rozdelili na akúsi trénovaciu a testovaciu množinu. Návrh algoritmov a vyhodnocovanie čiastkovej úspešnosti jednotlivých krokov algoritmu prebiehal na trénovacej množine zhotovenej na jednom slovenskom letisku. Záverečné vyhodnotenie úspešnosti sme vykonali na nezávislej testovacej množine z iného letiska.

Vstupy boli zhotovované so zreteľom na čo najpresnejšie zachytenie významných vlastností vizuálnej dohľadnosti a viacerých aspektov pozorovania význačných objektov. Trénovacia množnina bola zhotovená na letisku v období od prvého decembra 2010 do piateho februára 2011 v zimnom období pomocou mobilného telefónu Nokia 3120 classic odborným meteorológom a zachytáva rôzne podmienky, počas ktorých je potrebné vizuálnu dohľadnosť merať. Táto množina pokrývala viaceré problémy, ktoré môžu

(a) Trénovací pohľad 1

(b) Trénovaci pohľad 2

(c) Trénovací pohľad 3

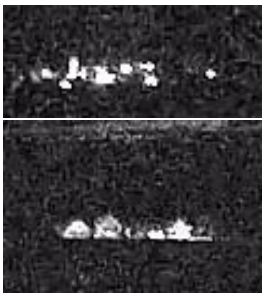


(d) Testovací pohľad

Obr. 5.2: Reprezentatívne príklady vstupov. Na testovacom pohľade sú zvýraznené polohy pozorovaných objektov. Pozorovali sme lietadlo, ktoré počas pozorovaní nezmenilo polohu, blízku značku vľavo dole, dve pohoria a dva zdroje svetla. Pohľad na objekty určené na pozorovanie v noci je na obrázku 5.1.

počas zhotovovania vstupov vzniknúť ako napríklad afinné transformácie meraných objektov spôsobene zmenou smeru pohľadu fotoaparátu, či dokonca zmenu perspektívy spôsobenú posunom miesta, v ktorom bola fotografia zhotovená.

V tejto trénovacej množine boli zaznamenané celkovo tri rôzne pohľady zachytávajúce objekty rozdielnych vlastností, na ktorých viditeľnosti sa rozličným spôsobom prejavovali meteorologické fenomény. Správne fungovanie na tejto množine predstavovalo výzvu, s ktorou sme sa v zadaní práce rozhodli vysporiadať a je podmienkou akýchkoľvek ďalších úvah o zmysle nasadenia systému.



Obr. 5.1

Vstupy boli manuálne upravené ich posunom, rotáciou a orezaním tak, aby sa

viac priblížili tým automaticky zhotovovaným a približne utriedené podľa klesajúcej dohľadnosti aby bolo uľahčené vyhodnocovanie čiastkových výsledkov algoritmov.

Trénovacia množina obsahovala zopár vstupov zhotovených počas výbornej a naopak veľmi nízkej dohľadnosti. Väčšina vstupov bola zhotovená za podmienok, pri ktorých síce boli vzdialené objekty pozorovateľné, často však na kraji ich viditeľnosti, čo nám umožnilo vyhodnocovať najmä úspešnosť na diskutabilných vstupoch a tak veľmi prísne vyhodnocovať úspešnosť navrhovaných riešení.

Záverečné testy prebiehali na slede fotografií, ktoré boli zhotovené automaticky priemyselnou kamerou v období od jedenásteho januára 2012 do dvadsiateho druhého marca toho istého roku s odstupom desiatich minút medzi snímkami. Táto množina predstavuje konkrétny príklad reálneho nasadenia algoritmov a vstupov, ktoré bude treba vyhodnotiť. Na tejto množine bolo zachytené veľké množstvo situácií, v ktorých musí systém fungovať ako sú napríklad denné a nočné pozorovania, sneh na pozorovaných objektoch, vysoká miera šumu počas nočných pozorovaní, či dokonca úplne alebo čiastočne zasnežená kamera.

Testovacie vstupy neboli nijako upravované ani triedené, nakoľko našim cieľom bolo vyhodnotiť úspešnosť na reálnych dátach, nie analyzovať zaujímavé okrajové prípady. Navyše, množina testovacích vstupov bola príliš rozsiahla na akékoľvek manuálne úpravy.

5.1.1 Metodika testovania

Pri vývoji našich algoritmov sme si postačili s jednoduchými testovacím nástrojmi, ktoré umožnili ľahko vizualizovať výstupné hodnoty niekoľkých skúmaných objektov a tie boli následne ručne analyzované vzhľadom na príslušné vstupy. Ako však bolo naznačené v časti 4.2.2, na rigorózne vyhodnotenie presnosti výsledkov navrhovaných algoritmov je takýto prístup nepostačujúci.

Cieľom algoritmov je povedať, či je objekt na danom vstupe viditeľný alebo nie. Overiť si odpoveď algoritmu však nie je možné realizovať plne automaticky a je nutné zadať manuálne očakávané odpovede pre všetky testovacie vstupy. Naše nástroje na záverečné testovanie tieto očakávané hodnoty umožňujú rýchlo a jednoducho zadávať za pomoci používateľsky prívetivého rozhrania, ktoré sa ukázalo ako veľmi užitočné pri ručnej klasifikácii veľkého množstva testovacích vstupov.

Tieto očakávané hodnoty sme obdobne ako databázu popisov ukladali v súbore formátu YML za pomoci prostriedkov knižnice OpenCV. V tomto súbore sme tiež uložili referenciu na testovacie popisy objektov, ako aj popis cesty k súborom, v ktorých sú uložené testovacie vstupy. Na záver sme kvôli zrýchleniu analýzy testov do súboru uložili aj číselné hodnoty odpovedí algoritmu prislúchajúceho danému súboru popisov

objektov.

V pozorovanej testovacej scéne sme zvolili celkovo šesť skúmaných objektov rôznych charakterov, aby sme testovaním pokryli čo možno najširší okruh možných situácii, s ktorými by sme sa mohli v praxi stretnúť. Pre každý skúmaný algoritmus sme k zvoleným objektom zostrojili príslušné šablóny a uložili ich do súborov vo formáte stručne opísanom v časti 4.2.3.

V našich analýzach sme sa zamerali na počet správne rozpoznaných prípadov viditeľnosti objektu. Taktiež sme za pomoci senzitivity a špecifickosti skúmali pomer chýb prvého a druhého typu, ktoré nastali a ten vizualizovali za pomoci ROC kriviek. Keďže spoločným parametrom oboch navrhnutých algoritmov je prah minimálnej odpovede viditeľného objektu, analyzovali sme spomínané hodnoty vzhľadom na hodnotu tohto parametra.

Skúmali sme odpovede obidvoch algoritmov pre každý objekt samostatne, aby sme boli schopní porovnať úspešnosť konkrétneho algoritmu na danom objekte. Pre celkové porovnanie úspešnosti algoritmov sme použili súhrnné počty odpovedí.

5.2 Výsledky testov

Po vypočítaní hodnôt odpovedí algoritmov sme sa pokúsili čo možno najrozumnejšie vizualizovať získané dáta tak, aby sme mohli čo možno najjednoduchšie vyhodnotiť ich úspešnosť. Okrem jednoduchého znázornenia počtu správnych odpovedí od zvoleného prahu viditeľnosti sme sa rozhodli použiť aj tzv. *Receiver Operating Characteristic* alebo ROC krivku.

Z počtu správne klasifikovaných vstupov vzhľadom na pozorovaný objekt na grafoch 5.3a a 5.3b môžeme pozorovať nielen rozdiely medzi algoritmami samotnými, ale aj v úspešnosti klasifikácie konkrétnych objektov vrámci jednotlivých algoritmov.

Medzi najzaujímavejšie pozorovania radíme veľkú citlivosť obdivoch algoritmov na blízky chvost lietadla, ale aj problémy mriežky Gáborových filtrov s rozpoznávaním zdrojov svetla určených len na nočné pozorovanie. Druhý problém by sme prisúdili náročnému určeniu očakávaných hodnôt pre pozorovania počas dňa a následným nevyspytateľným výsledkom, ktoré dosahuje algoritmus v tomto období. V niektorých prípadoch algoritmus správne určil viditeľnosť ťažko pozorovateľných budov, ktoré sú počas noci dobre pozorovateľné kvôli ich nasvieteniu. Naše problémy môžu byť paradoxne spôsobené vysokou citlivosťou algoritmu. Nakoľko však tieto objekty nie sú vhodné na denné pozorovanie, neprikladáme príslušným odpovediam veľkú váhu.

Obdobné problémy nastali aj s chvostom lietadla, ktorý bol pre zmenu náhodne pozorovaný počas noci, a aj keď by ľudský pozorovateľ označil aj v zašumenej nočnej

snímke ako viditeľný, obidva algoritmy dokázali tento problém s rozumnými výsledkami rozhodovať iba pri nízkych prahoch.

Tieto výsledky zároveň ilustrujú dôležitosť zohľadnenia ďalších faktorov pri meraní dohľadnosti automatickými systémami. Aby sme zohľadnili výsledky algoritmov v pozorovaní objektov na relevantných vstupoch, rozhodli sme sa analyzovať aj mierne upravené štatistiky výsledkov, v ktorých sme umelo nastavili odpovede algoritmov pre objekty pozorované v nevhodnom období, t.j pre lietadlo sme ignorovali kladné odpovede počas noci a pre zdroje svetla naopak počas dňa, a tieto sme nastavili na zápornú odpoveď.

Po zohľadnení doby zhotovenia snímky pri rozpoznávaní objektov sme získali prekvapivo dobré výsledky. Ako vidno na grafoch 5.3c a 5.3d, pri pozorovaní niektorých objektov sme si mohli dovoliť použiť vyššie prahy a zároveň sme neboli ovplyvnení odpoveďami na vstupoch, na ktorých sme neočakávali správne odpovede algoritmu, čo môžeme ilustrovať pozorovaním zdrojov svetla určených na nočné pozorovanie.

Na porovnanie celkových výsledkov algoritmov sme zlúčili výsledky na jednotlivých objektoch a obdobne sme analyzovali filtrované aj nefiltrované výsledky. Ako vidieť na grafoch 5.3c a 5.3d u oboch navrhnutých prístupov došlo k zlepšeniu výsledkov, pričom k výraznejšiemu zlepšeniu došlo u mriežky Gáborových filtrov.

Nakoľko v oblasti nasadenia je dôležité poznať okrem chybovosti využívaných prostriedkov aj charakter ich chýb, analyzovali sme výsledky algoritmov aj pomocou ROC krivky použitím obdobného prístupu ako pri analýze úspešnosti algoritmov. Výsledné grafy sú na obrázkoch 5.4 a 5.5.

Spomínané vlastnosti pozorovaných objektov vzhľadom na dobu pozorovania sa prejavili aj na tvare ROC kriviek a opäť malo filtrovanie výsledkov výraznejší vplyv na mriežku Gáborových filtrov. Obidva algoritmy poskytli kvalitné výsledky, pričom mriežka Gáborových filtrov dokázala predbehnúť algoritmus detekcie hrán len na malom množstve objektov.

Oba algoritmy dokázali po filtrovaní na základe doby dňa poskytnúť zarážajúco dobré výsledky na blízkom chvoste stíhačky, ale aj na bodových zdrojoch svetla, na ktorých dosahovali veľmi nízku mieru falošných detekcií.

Na základe súhrnných výsledkov sme získali pre oba algoritmy ROC krivky, ktoré naznačujú potrebu rozlišovania nočného a denného pozorovania, nakoľko filtrovanie výsledkov spôsobilo výrazné zlepšenie pomeru falošných detekcií a prehliadnutých objektov pre oba algoritmy.

Výsledkom našich analýz teda je, že oba navrhnuté algoritmy dokážu poskytnúť kvalitné výsledky so zaujímavým pomerom nerozpoznaných objektov a falošných detekcii. Ďalším dôležitým výsledkom je, že oba algoritmy dokážu pracovať s univerzálne zvolenou

hodnotou prahu viditeľnosti, čím sa dá zjednodušiť prípadné nasadenie systému, ktoré by malo spočívať hlavne v konštrukcii šablóny pozorovaného objektu. Univerzálne zvolená hodnota prahu však nie je ani pre jeden algoritmus či pozorovaný objekt záväzná alebo obmedzujúca a v prípade potreby je možné tento parameter upraviť tak, aby výsledky lepšie vystihovali potreby spoľahlivosti v konkrétnom nasadení.

V nasledujúcej tabuľke zhŕňame výsledky algoritmov.

Algoritmus	Úspešnosť	
	nefiltrovaná	filtrovaná
Detekcia hrán	86.46%	95%
Mriežka Gáborových filtrov	79.97%	93.95%

Tabuľka 5.1: Počet správne správnych odpovedí pri prahu rovnom 0.2.

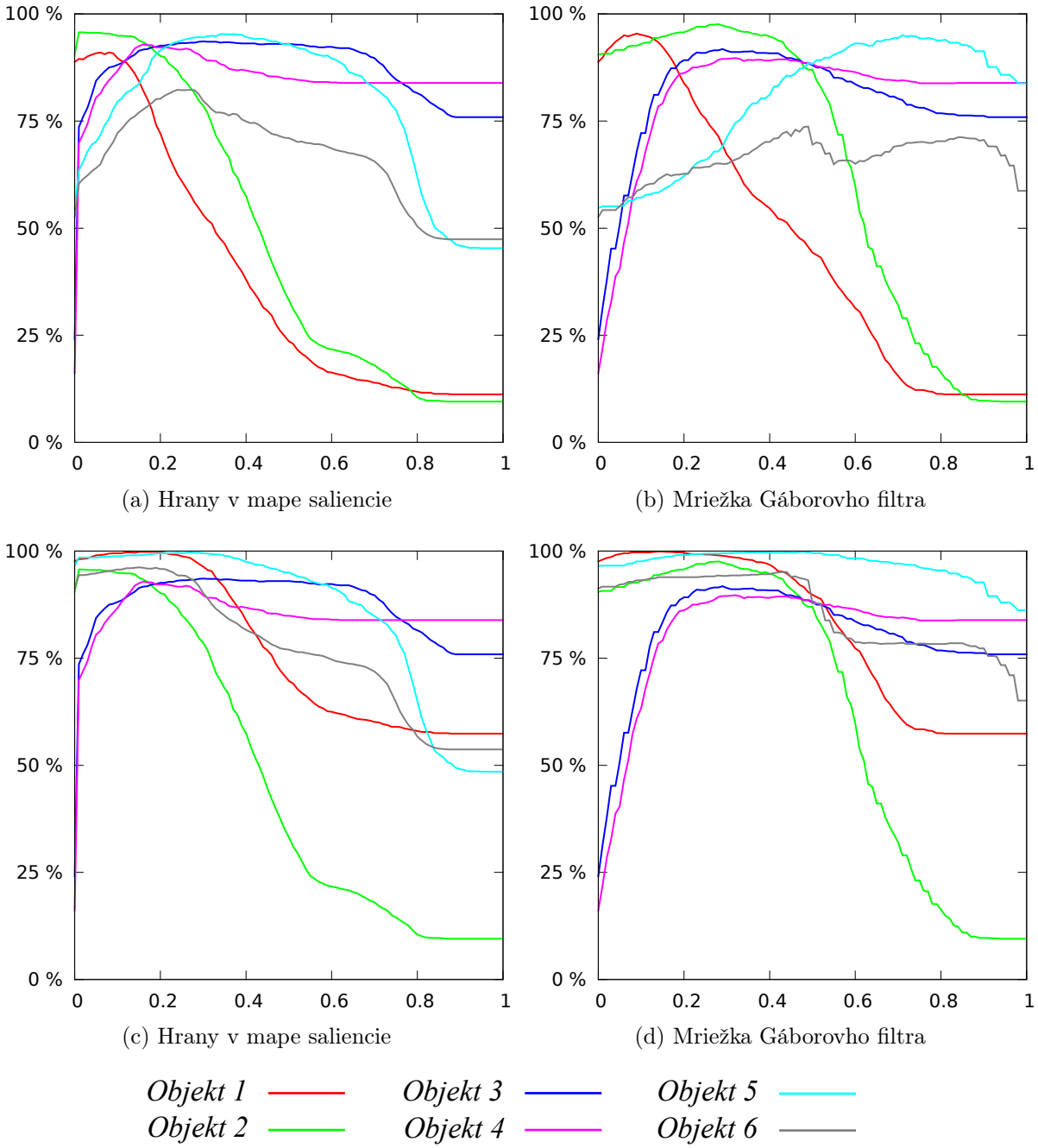
5.3 Výkonové aspekty

Praktické nasadenie našich algoritmov je podmienené ich náročnosťou na výpočtový výkon, ktorá zároveň ovplyvňuje okrem možnosti ďalších analýz výstupov algoritmov aj cenu nasadenia systému. Potešilo nás, že pri vývoji ani testovaní algoritmov sme nenarazili na výraznejšie obmedzenia v tomto smere. Algoritmy dokázali aj vďaka efektívnosti implementácie knižnice OpenCV pracovať veľmi rýchlo aj na relatívne nemodernom hardvéri, pričom okrem optimalizácie výkonu samotných algoritmov vidíme priestor na získanie ďalšej výpočtovej sily vo využití knižníc umožňujúcich paralelizáciu výpočtov či využitie výkonu grafických kariet.

Výkonovo náročnejší zo skúmaných algoritmov je ten využívajúci detekciu hrán v mape saliencie, na konštrukciu ktorej je potrebné využiť drahú Fourierovu transformáciu, ktorá môže na rozsiahlych objektoch výrazne zvýšiť výpočtové nároky algoritmu. Túto skutočnosť však nepovažujeme za výrazný problém, nakoľko väčšina pozorovaných objektov zaberá len malú plochu snímky a rozsiahly objekt by mohol okrem príliš nízkej vzdialenosti od kamery znamenať aj nevhodnú voľbu pozorovaných objektov.

Algoritmus využívajúci mriežku Gáborových filtrov dokázal spracovať jeden testovací vstup o šiestich pozorovaných objektoch za približne jednu tretinu sekundy. Detekcia hrán zabrala približne polovičný čas. Časová zložitosť našich algoritmov je najviac ovplyvnená zložitosťou Fourierovej transformácie či Cannyho detektorom hrán.

Pamäťová náročnosť obidvoch skúmaných algoritmov nie je vysoká, nakoľko okrem konštantného faktoru pamäte potrebnej na uchovanie medzivýsledkov nekládli na pamäť ďalšie nároky a dokázali pracovať nanajvýš s desiatkami megabajtov operačnej pamäte.



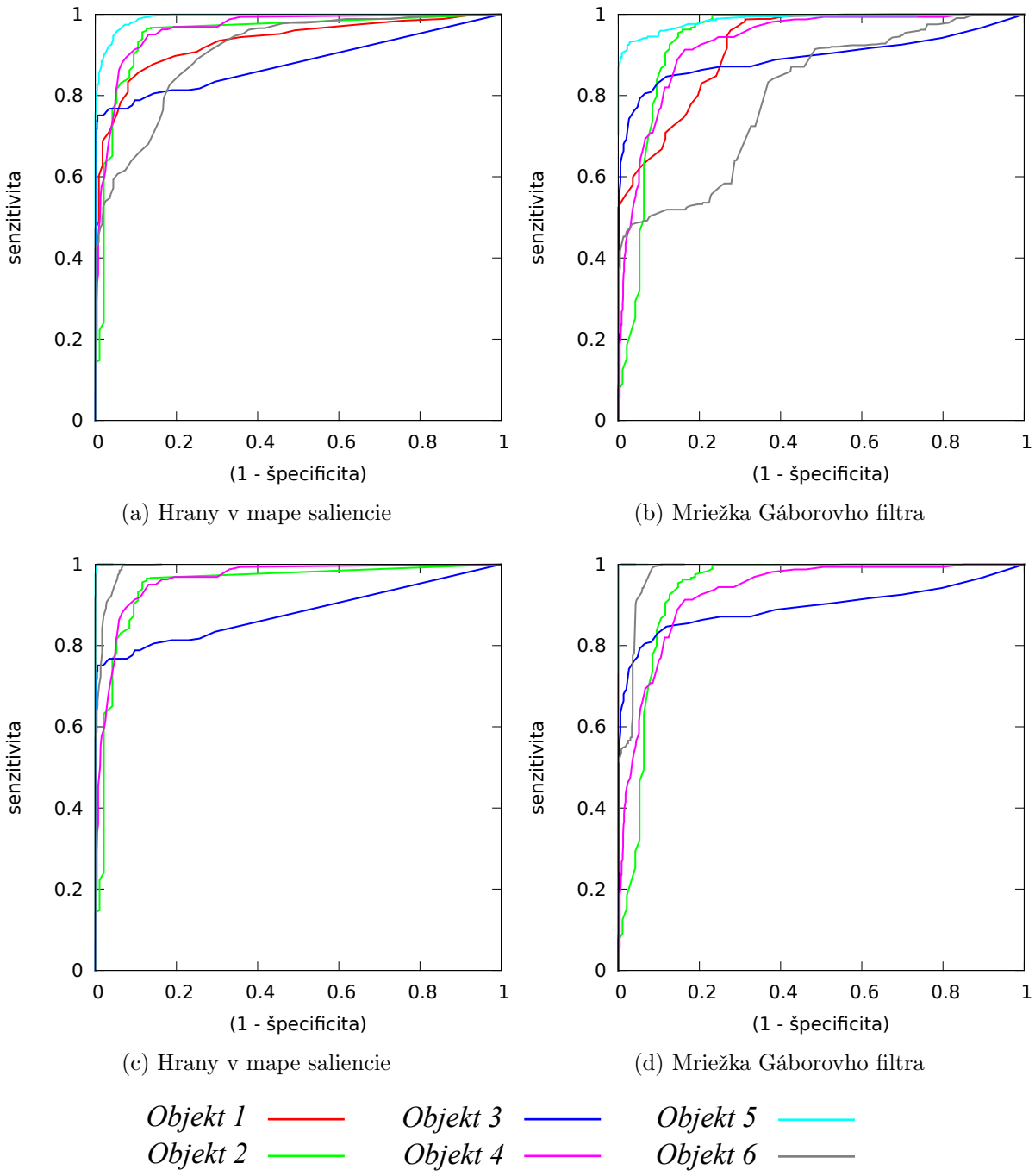
(a) Hrany v mape saliencie

(b) Mriežka Gáborovho filtra

(c) Hrany v mape saliencie

(d) Mriežka Gáborovho filtra

Obr. 5.3: Percentuálna miera úspešnosti oboch algoritmov v závislosti od hodnoty prahu viditeľnosti. Grafy znázorňujú počet správne klasifikovaných vstupov pre každý pozorovaný objekt osobitne. V hornom rade sú výsledky ignorujúce dennú dobu. V dolnom rade vidno zlepšenie po filtrovaní odpovedí. Pri niektorých objektoch sa však problém denného a nočného pozorovania neprejavil. Tieto grafy svedčia o možnosti stanoviť hodnotu prahu viditeľnosti na univerzálnu hodnotu.



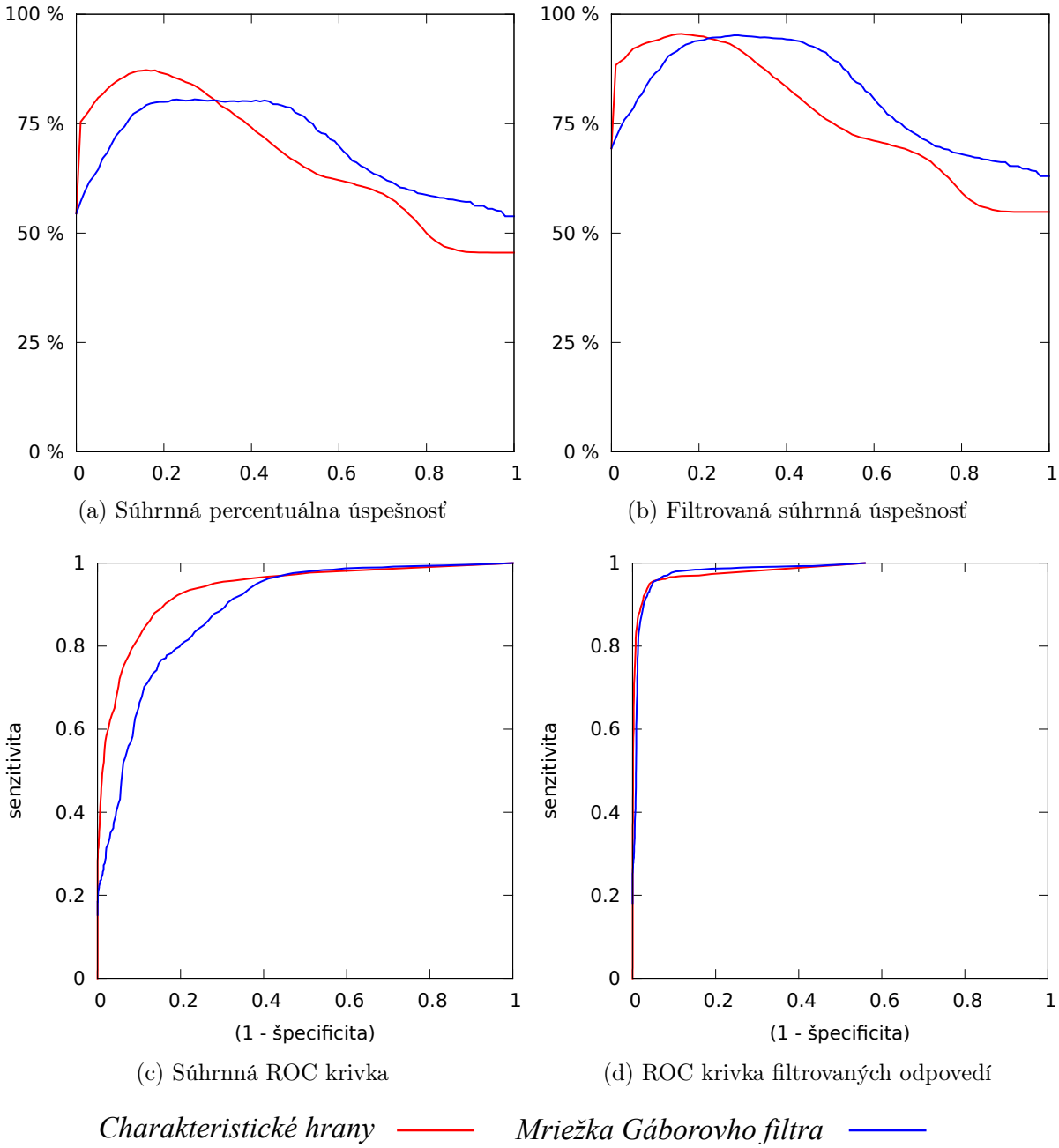
(a) Hrany v mape saliencie

(b) Mriežka Gáborovho filtra

(c) Hrany v mape saliencie

(d) Mriežka Gáborovho filtra

Obr. 5.4: Pomer prehliadnutých a chybne nájdených objektov zobrazený pomocou ROC krivky. Algoritmus využívajúci mriežku odpovedí Gáborovych filtrov sa zlepšil najmä kvôli jeho nepredvídateľnému správaniu na objektoch prislúchajúcich plošným zdrojom svetla počas dňa. Tieto krivky nám môžu pomôcť stanoviť hodnotu prahu pre nami požadovaný pomer nesprávane prehliadnutých či zaregistrovanych objektov.



(a) Súhrnná percentuálna úspešnosť

(b) Filtrovaná súhrnná úspešnosť

(c) Súhrnná ROC krivka

(d) ROC krivka filtrovaných odpovedí

Obr. 5.5: Súhrnné výsledky oboch algoritmov nerozlišujúce medzi jednotlivými objektami. V hornom rade je percentuálny počet správne klasifikovaných objektov, v dolnom rade sú súhrnné krivky oboch algoritmov. V ľavom stĺpci sú nefiltrované výsledky, napravo výsledky zohľadňujúce dobu pozorovania. Tieto grafy svedčia o možnosti stanoviť hodnotu prahu viditeľnosti na univerzálnu hodnotu.

Záver

Nazdávame sa, že sa nám podarilo poskytnúť nielen dostatočne zrozumiteľný popis teórie, na ktorej sú založené súčasné rozšírené spôsoby merania dohľadnosti, ale aj zdôvodniť ich nedostatky, a tak podoprieť vhodnosť riešenia metódami počítačového videnia. Ako sme však ukázali ani doteraz známe zaužívané algoritmy sa nejavia ako vhodné na riešenia nášho problému, a teda vznikla potreba navrhnúť vlasné riešenie, ktoré by bolo dostatočne robustné a spoľahlivé.

Smer, ktorý sme sa rozhodli preskúmať a ktorý nám mal poskytnúť významnú pomoc v rozpoznávaní objektov, bolo využitie saliencie. Analýza salientných objektov si zaslúžila pozornosť mnohých odborníkov a výsledkom ich snaženia je viacero prístupov, ako odhadnúť, ktorý objekt je pre scénu najviac významný a najviac upúta pozornosť. Výberu vhodného algoritmu z plejády známych prístupov sme preto venovali náležitú pozornosť a z niekoľkých analyzovaných prístupov sa ako najvhodnejší ukázal ten využívajúci kvaternióny na analýzu fázovej zložky vo frekvenčnom spektre prislúchajúcom pozorovanej scéne. Jednoduchšia verzia tohto algoritmu využívajúca klasickú Fourierovu transformáciu a čiernobielu reprezentáciu scény poskytla výrazne odlišné výsledky menej vhodné na ďalšie spracovanie.

Možnosti tohto prístupu nekončia pri konštrukcii mapy saliencie a načrtá sa možnosť analýzy na viacerých úrovniach, pričom nami navrhnuté jednoduché riešenie dokázalo napraviť niektoré odhalené nedostatky spomínanej metódy konštrukcie mapy saliencie.

Snažiac sa využiť mapu saliencie pri detekcii objektov skúmali sme niekoľko možných spôsobov segmentovania mapy saliencie za účelom výpočtu čo najjednoduchšej miery viditeľnosti objektov, avšak výsledky tohto nášho snaženia neboli dostatočne dobré, a preto sme sa rozhodli využiť ďalšie vlastnosti pozorovaných objektov akými sú aj polohy a orientácie hrán.

Po analýze pestrej množiny vstupov sme dospeli k záveru, že pomocou vertikálných a horizontálnych hrán dokážeme dostatočne dobre popísať objekt na to, aby sme ho následne vedeli rozoznať a detekovať. Samotná mapa saliencie nám mala pomôcť oslobodiť sa od vplyvov rôznych fenoménov, a tak nám zjedošiť prácu na istým spôsobom normalizovaných vstupoch.

Prvý z nami navrhnutých algoritmov využíval na detekciu hrán v mape saliencie viacero hodnôt prahov Cannyho detektora hrán. Ďalší algoritmus sa pokúsil využiť Gáborov filter a tak rozpoznávať vertikálne a horizontálne hrany na malej ploche. Oba algoritmy dokázali rozpoznať rozumným spôsobom hrany v scéne a ich porovnanie s očakávanými polohami hrán tak umožnilo vypočítať mieru viditeľnosti objektu.

Úspešne sme teda navrhli dva algoritmy, pričom obidva z nich dosahovali dobré výsledky, čím sa načrtla možnosť ich ďalšieho zlepšovania. K nášmu prekvapeniu však u oboch týchto algoritmoch spočívalo zlepšenie práve vo vynechaní výpočtu mapy saliencie. Tento náš výsledok môže naznačovať nevhodnosť nami zvolenej metódy konštrukcie mapy saliencie, ale aj nevhodnosť saliencie ako takej pri detekcii objektov.

Vzhľadom na vlastnosti našich vstupov sme si mohli dovoliť predpoklady nemysliteľné v iných nasadeniach a to zamerať sa na konkrétnu oblasť vstupu. Saliencia tak mohla poskytnúť informácie z globálneho hľadiska, ktoré sme následne nezobrali do úvahy, ba dokonca nám mohla uškodiť, ked objekt mierne splýval s okolím, čo spôsobilo jeho malé vyčnievanie a následne nízke hodnoty saliencie. Naproti tomu sa nám vyššie popísanými spôsobmi podarilo detekovať hranu v pôvodnom vstupe.

Aj napriek tomu, že sa nám nepodarilo efektívne využiť salienciu pri rozpoznávaní objektov, nemôžeme povedať, že by sme zlyhali pri celkovom návrhnu algoritmov, nakoľko tie dokázali s rôznou mierou úspešnosti fungovať pri rozličných podmienkach. Výsledky našich testov naznačujú praktickú využiteľnosť oboch algoritmov a nevylučujú ich nasadenie s podpornými systémami, ktoré by ešte viac zlepšili ich výkon a umožnili ich využitie na meranie aktuálnych hodnôt vizuálnej dohľadnosti priamym meraním oveľa bližším ľudskému vnímaniu ako pomocou doteraz zaužívaných spôsobov.

Úspešným návrhom algoritmov sa však naša práca neskončila, okrem ich ďalšieho rozvíjania by stálo za zváženie aj preskúmanie iných smerov. V našej práci sme napríklad opomenuli neurónové siete, ktoré by mohli inšpirovať veľa zaujímavých prístupov. Zároveň je pre úspešné určenie dohľadnosti z odpovedí našich algoritmov potrebné určiť prevládajúcu dohľadnosť zohľadňujúc vzdialenosť ale aj polohu pozorovaných objektov, či najrôznejšie záludné scenáre, ktoré nám počasie vie pripraviť.

Literatúra

[1] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.

[2] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 886–893, Washington, DC, USA, 2005. IEEE Computer Society.

[3] J. G. Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *Journal of the Optical Society of America A: Optics, Image Science, and Vision*, 2(7):1160–1169, 1985.

[4] Todd A. Ell and Stephen J. Sangwine. Hypercomplex fourier transforms of color images. In *International Conference on Image Processing*, pages 137–140, 2001.

[5] Chenlei Guo, Qi Ma, and Liming Zhang. Spatio-temporal saliency detection using phase spectrum of quaternion fourier transform. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:1–8, 2008.

[6] Jonathan Harel, Christof Koch, and Pietro Perona. Graph-based visual saliency. In *Advances in Neural Information Processing Systems 19*, pages 545–552, 2007.

[7] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Pascal Fua, and Nassir Navab. Dominant orientation templates for real-time detection of texture-less objects. In *CVPR*, pages 2257–2264, 2010.

[8] Michal Holtzman-Gazit, Lihi Zelnik-Manor, and Irad Yavneh. Salient edges: A multiscale approach. *ECCV 2010 Workshop on Vision for Cognitive Tasks*, 2010.

[9] Xiaodi Hou and Liqing Zhang. Saliency detection: A spectral residual approach. In *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR07). IEEE Computer Society*, pages 1–8, 2007.

[10] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(11):1254–1259, November 1998.

[11] T. Kadir, A. Zisserman, and M. Brady. An affine invariant salient region detector, 2004.

[12] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60:91–110, November 2004.

[13] Fernand Meyer. Color image segmentation. In *Image Processing and its Applications, 1992., International Conference on*, pages 303 – 306, 1992.

[14] Mark Nixon and Alberto S. Aguado. *Feature Extraction & Image Processing, Second Edition.* Academic Press, 2 edition, January 2008.

[15] A. V. Oppenheim and J. S. Lim. The importance of phase in signals. *Proceedings of the IEEE*, 69(5):529–541, 1981.

[16] International Civil Aviation Organizoation. *Manual of Runway Visual Range Observing and Reporting Practices (DOC 9328).* ICAO, third edition, 2005.

[17] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(7):629 –639, jul 1990.

[18] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "grabcut": interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23(3):309–314, August 2004.

[19] Thomas Serre, Lior Wolf, Stanley Bileschi, Maximilian Riesenhuber, and Tomaso Poggio. Robust Object Recognition with Cortex-Like Mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):411–426, March 2007.

[20] Remco C. Veltkamp. Shape matching: Similarity measures and algorithms. In *Proceedings of the International Conference on Shape Modeling & Applications*, SMI '01, pages 188–, Washington, DC, USA, 2001. IEEE Computer Society.

[21] Colin Ware. *Information Visualization: Perception for Design.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[22] WMO. *Guide to Meteorological Instruments and Methods of Observation (Wmo-8).* World Meteorological Organization, seventh edition, 2008.

Zoznam elektronických príloh

Na priloženom CD nosiči sa nachádzajú:

- Elektronická verzia práce
- Zdrojové kódy aplikácie
- Príklady vstupov
- Popisné súbory šablón navrhnutých algoritmov
- Príklady rozpoznávania objektov
- Príklady máp saliencie