Computation of disparity in stereo images using three-dimensional recursive search

Rožić, Miroslav

University of Zagreb

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Miroslav Rožić

COMPUTATION OF DISPARITY IN STEREO IMAGES USING THREE-DIMENSIONAL RECURSIVE SEARCH

DOCTORAL THESIS

Zagreb, 2019.

University of Zagreb

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Miroslav Rožić

COMPUTATION OF DISPARITY IN STEREO IMAGES USING THREE-DIMENSIONAL RECURSIVE SEARCH

DOCTORAL THESIS

Supervisor: Professor Tomislav Pribanić, PhD

Zagreb, 2019.

Sveučilište u Zagrebu
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Miroslav Rožić

IZRAČUN DISPARITETA U STEREOSLIKAMA S POMOĆU TRODIMENZIONALNOGA REKURZIVNOGA PRETRAŽIVANJA

DOKTORSKI RAD

Mentor: prof. dr. sc. Tomislav Pribanić

Zagreb, 2019.

Doktorski rad izrađen je na Sveučilištu u Zagrebu Fakultetu elektrotehnike i računarstva, na Zavodu za elektroničke sustave i obradbu informacija.

Mentor: prof. dr. sc. Tomislav Pribanić

Doktorski rad ima: 87 stranica

Doktorski rad br.: _____

About the Supervisor

Tomislav Pribanić was born in Zagreb in 1971. He received BSc, MSc and PhD degrees in-electrical engineering from the University of Zagreb, Faculty of Electrical Engineering and Computing (FER), Zagreb, Croatia, in 1996, 2001 and 2005, respectively.

From February 1996 he has been working at the Department of Electronic Systems and Information Processing at FER. He was a visiting researcher at INRIA Rhone Alpes, Grenoble, France in 2004 and 2018. In addition he was a visiting researcher at Fraunhofer institute IGD, Darmstadt, Germany during 2006. In December 2009, December 2014 and May 2018 he was promoted to Assistant Professor, to Associate Professor and to Professor, respectively. He participated in three scientific projects financed by the Ministry of Science, Education and Sports of the Republic of Croatia and two international projects. He has been a project leader of three international and eight national projects. He published more than 50 papers in journals and conference proceedings in the area of computer vision and image processing, biomedical signal processing and human motion analysis. In 2016 he has founded Advanced Shape Reconstruction and Registration Laboratory (SHARK Lab) on Faculty of Electrical Engineering and Computing.

Prof dr. sc. Pribanić is a member of IEEE, Croatian Academy of Engineering, IFMBE, EMBS, CROMBES, Centre of Research Excellence for Data Science and Cooperative Systems and Center of Excellence for Computer Vision at the University of Zagreb. He participated in five conference international programs committees; he serves as a reviewer for more than a dozen international journals. He received the "Josip Lončar" silver medal from FER for outstanding Master theses in 2001. In 2006 he was awarded "Vera Johanides", the award of the Croatian Academy of Engineering for the Young Scientists, for an outstanding research performance (an emphasis on the applicability) in the last five years. In 2012 he received the biennial "Best Paper Award", among all papers published in 2010 and subsequently considered till 2012, in a prestigious journal Pattern Recognition based on the following criteria: originality of the contribution, presentation and exposition of the manuscript, and citations by other researchers. For his innovations "First world structured light scanner on the smartphone", "Efficient Software Synchronization of Camera and Projector (ESCAPE)" and "3D system for detection of the spinal deformity" he has received silver, bronze and gold medal on International innovation exhibition ARCA 2015, ARCA 2016 and ARCA 2017, respectively.

O mentoru

Tomislav Pribanić rođen je u Zagrebu 1971. Diplomirao je, magistrirao i doktorirao u polju elektrotehnike na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu (FER), 1996., 2001. odnosno 2005. godine.
Od veljače 1997. godine radi na Zavodu za elektroničke sustave i obradbu informacija FER-a. Bio je gostujući istraživač u Francuskoj, Grenoble, na INRIA institutu u 2004. i 2018. godini. Također bio je gostujući istraživač u Njemačkoj, Darmstadt, na Fraunhofer institutu IGD tijekom 2006. godine. U prosincu 2009. godine izabran je u zvanje docenta, u prosincu 2014. izabran je u zvanje izvanrednog profesora, a u svibnju 2018. izabran je u zvanje redovitog profesora. Sudjelovao je na tri znanstvena projekta Ministarstva znanosti, obrazovanja i sporta Republike Hrvatske i dva međunarodna projekta. Vodio je tri međunarodna, te osam domaćih projekata. Objavio je više od 50 radova u časopisima i zbornicima konferencija u području obrade slike, računalnog vida, te obrade biomedicinskih signala i analize ljudskog pokreta. U 2016. osnovao je FER-ov Laboratorij za naprednu 3D rekonstrukciju i 3D registraciju površina (engl. Advanced Shape Reconstruction and Registration Laboratory, SHARK Lab).
Prof. dr. sc. Pribanić član je stručne udruge IEEE, Akademije tehničkih znanosti Hrvatske, IFMBE, EMBS, CROMBES, Znanstvenog centra izvrsnosti za znanost o podacima i napredne kooperativne sustave te Centra izvrsnosti za računalni vid, Sveučilišta u Zagrebu. Sudjeluje u pet međunarodnih programskih odbora znanstvenih konferencija, te sudjeluje kao recenzent za desetak inozemnih časopisa. Godine 2001. primio je srebrnu plaketu “Josip Lončar” FER-a za posebno istaknutu magistarsku disertaciju. Godine 2006. dobio je nagradu “Vera Johanides” Hrvatske akademije tehničkih znanosti (HATZ) mladom znanstveniku za ostvaren osobni znanstveni napredak i zapaženu znanstvenu aktivnost na području tehničkih znanosti u proteklih pet godina. U 2012. dobio je nagradu za najbolji rad u konkurenciji svih radova objavljenih unazad dvije godine u prestižnom časopisu "Pattern Recognition" po kriteriju znanstvenog doprinosa, citiranosti i prezentaciji. Za svoje inovacije “First world structured light scanner on-the smartphone”, “Efficient Software Synchronization of Camera and Projector (ESCAPE)” te “3D system for detection of the spinal deformity” nagrađen je srebrnim, brončanim i zlatnim odličjem na međunarodnoj izložbi inovacija ARCA 2015, ARCA 2016 odnosno ARCA 2017.

Acknowledgments

A doctoral thesis is more than a one man's work - it is, as much as its author, touched by, reviewed, pushed and pulled ever towards its conclusion by many people. Even more so in the situation where the author is not a full-time academic researcher and pursues the doctoral degree in his free time.

Therefore, I would like to thank my academic advisors foremost:
professor Tomislav Pribanić, my supervisor, for everlasting patience and support; my undergraduate and long time mentor, professor Mario Cifrek, for advice and encouragement.

Furthermore, I would like to thank my employers and superiors: Damir Ježić, PhD at Mikroprojekt, Andrej Božić at Zagrebačka Banka, and Josip Ćesić, PhD at Gideon Brothers; who helped, encouraged, and supported me in my doctoral studies, providing me with the time and space to pursue this goal.

Finally, I would like to thank my wife Jelena and my parents Mirko and Ljubica for never-ending support and patience, for never losing faith, and for pushing me when I needed it the most.

For my daughter Katja, without whom it would not have made any sense.

Zahvale

Doktorski je rad više od individualnog poduhvata - on je, poput svog autora, diran, revidiran, guran i vučen od strane mnogih prema svojem završetku. Tim više u situaciji u kojoj autor nije istraživač u punom radnom vremenu i koristi svoje slobodno vrijeme za doktorski rad.

Stoga bih se prvenstveno htio zahvaliti mojim profesorima i mentorima: mom doktorskom mentoru prof. dr. sc. Tomislavu Pribaniću, na vječnom strpljenju i podršci; mom diplomskom i dugogodišnjem mentoru prof. dr. sc. Mariu Cifreku, na savjetima i ohrabrenju.

Nadalje, htio bih se zahvaliti mojim poslodavcima i nadređenima, dr. sc. Damiru Ježiću u Mikroprojektu, Andreju Božiću u Zagrebačkoj Banci te dr. sc. Josipu Ćesiću u Gideon Brothers koji su pomagali, ohrabrivali i pružali mi podršku u doktorskom studiju, pružajući mi vrijeme i mjesto za ostvarenje ovog cilja.

Naposljetku, htio bih zahvaliti supruzi Jeleni i roditeljima Mirku i Ljubici za bezgraničnu podršku i strpljenje, što su uvijek vjerovali u mene te me gurali kada je najviše trebalo.

Za moju kćer Katju, bez koje sve to ne bi imalo smisla.

Abstract

Disparity computation is a crucial step in the process of stereo 3D reconstruction, which has been a heavily investigated topic and one of the important problems in the field of computer vision. Many methods and approaches have been devised to provide a dense disparity map for a stereo image pair, with approaches roughly divided into local methods, which determine the correct disparity by choosing the value with an optimum matching cost in a local region, or global methods, which attempt to optimize a global energy equation for the entire scene. The global methods, while generally yielding better results, are more computationally intensive and are generally not suitable for real-time or embedded applications. The aim of this research was to propose new methods for disparity computation which would yield greater accuracy of the computed disparity map and faster computation by combining existing stereo algorithms with the three-dimensional recursive search algorithm (3DRS). The 3DRS algorithm was originally devised for motion estimation, frame rate up-conversion and de-interlacing in high definition televisions, and it can generate a coarse correspondence map (depending on the use, either the optical flow or disparity map) within a very short amount of time. In this dissertation, two methods based on the combination of the 3DRS algorithm with the local Winner-take-all method or the global Dynamic Programming method are presented. The proposed methods are evaluated using the Middlebury image set. It is shown that the proposed hybrid methods significantly outperform the original methods in execution time while maintaining or improving upon the accuracy of both methods. In addition, the hybrid methods are shown to exhibit greater robustness by reducing the dependency on specific external parameters.

Keywords: Stereo matching, disparity, 3DRS, Hybrid method, Winner-take-all, Dynamic programming

Prošireni sažetak

Izračun dispariteta u stereoslikama s pomoću trodimenzionalnoga rekurzivnoga pretraživanja

Uvod

Računalni vid interdisciplinarno je područje znanosti sa ciljem da računalima omogući visoku razinu razumijevanja okoline korištenjem digitalnih slika. Kao inženjerskoj disciplini, zadaća mu je proizvesti autonomne sustave koji mogu izvršavati iste vrste zadataka za koje je sposoban čovjekov sustav vida. Unutar računalnog vida, jedno od najistraživanijih je područje *guste trodimenzionalne stereo rekonstrukcije*, cilj kojeg je izgraditi trodimenzionalni model scene na osnovu jedne ili više slika. Metode se ugrubo mogu podijeliti na *aktivne*, koje koriste izvor svjetla kako bi u scenu ubacile informaciju, i pasivne, koje grade informaciju o dubini tražeći korespondencije između dva ili više različitih pogleda na scenu, izravno oponašajući način na koji čovjek percipira svoj okoliš binokularnim vidom.

Problem izračuna korespondentnih značajki u parovima slika je također predmet istraživanja u području estimacije pokreta, gdje se mapa kretanja značajki između slika (zvana još i *optički tok*) računa za niz slika u vremenu. Ugrubo se metode za računanje optičkog toka mogu podijeliti u *metode minimizacije ostatka*, korištene uglavnom u području kompresije video signala gdje su njihova svojstva korisna za minimizaciju rezidualnog signala dobivenog izračunom razlike između dvije susjedne slike u video signalu te *metode procjene stvarnog pokreta* svojstvo kojih je visoka korelacija između stvarne kretnje u sceni i vektora pokreta izračunatih u optičkom toku, što je poželjno kod primjena za interpolaciju slika u digitalnom televizijskom signalu.

Jedan od tržišno najuspješnijih algoritama za procjenu stvarnog pokreta jest algoritam *trodimenzionalnoga rekurzivnoga pretraživanja* (3DRS), koji je našao široku primjenu u digitalnim televizijskim aparatima zahvaljujući svojoj jednostavnosti, niskoj cijeni ugradnje te mogućnostima izračuna vrlo kvalitetnih procjena pokreta u sceni. Uspješno je upotrjebljen za obradu signala visoke razlučivosti u stvarnom vremenu, kao i za izračun grubih mapa dispariteta iz parova slika. S obzirom na performanse algoritma, njegovu široku u potrebu i jednostavnost, za cilj je ovog istraživanja odabrano ispitati može li se algoritam trodimenzionalnoga rekurzivnoga pretraživanja upotrijebiti za računanje guste mape dispariteta kombiniranjem sa postojećim metodama za izračun guste mape dispariteta, te unaprijediti performanse rečenih metoda smanjenjem njihove složenosti i poboljšanjem kvalitete njihovih rezultata.

Načela stereo rekonstrukcije

U poglavlju 2 opisana su osnovna načela stereo rekonstrukcije, počevši od definicije geometrijskih primitiva potrebnih za definiciju projektivne geometrije, te modela *camere obscure* (eng.*pinhole*), kojim opisujemo projektivno preslikavanje iz trodimenzionalnog koordinatnog sustava scene u dvodimenzionalni koordinatni sustav slike. Model kamere definiramo matricom koju nazivamo *intrinzična matrica kamere* te definiramo *ekstrinzične parametre kamere*, kojim definiramo odnos koordinatnog sustava kamere i vanjskog koordinatnog sustava scene, preko trodimenzionalnog vektora translacije i matrice rotacije. Definiramo sustav stereo kamera pomoću dva modela *camere obscure* između kojih je uspostavljen prostorni odnos putem vektora translacije i matrice rotacije. Razliku u koordinatama preslikanih objekata u dvije slike kamera nazivamo disparitetom, koji je obrnuto proporcionalan udaljenosti objekata od kamere (dubini). Geometriju preslikavanja objekata u sceni u koordinatni sustav kamera koja uspostavlja odnos slika objekata u jednoj u odnosu na drugu kameru nazivamo *epipolarnom geometrijom*. Epipolarnom su geometrijom definirani pravci u ravninama slika (zvani *epipolarnim pravcima*) na kojima se mogu nalaziti korespondentne točke između dvije slike. Epipolarna geometrija opisana je *fundamentalnim i esencijalnim matricama* stereo sustava. Naposljetku, definira se postupak *epipolarnog ispravljanja*, kojim se epipolarni pravci u obje slike stereo sustava čine paralelnima i poravnatima sa recima piksela u senzoru slike, što uvelike olakšava postupak izračuna dispariteta. Kao preduvjet za istu, opisan je postupak otklanjanja distorzije u slici koja nastaje zbog efekata koje unose realne leće kamera, uz opis radijalno-tangencijalnog modela distorzije slike.

Trodimenzionalno rekurzivno pretraživanje

U trećem se poglavlju opisuje algoritam trodimenzionalnoga rekurzivnoga pretraživanja (3DRS) u svojoj primarnoj namjeni procjene pokreta u video signalu. Temeljni način rada algoritma jest usporedba i uparivanje blokova u slici, gdje se slika dijeli u mrežu blokova kojima algoritam pokušava dodijeliti vektor pomaka koji uparuje svaki blok sa pozicijom istog ili najsličnijeg bloka u drugoj slici. Ovo algoritam 3DRS ostvaruje koristeći dvije osnovne pretpostavke rada algoritma: da su objekti u sceni veći od blokova u koji je podijeljena slika te da objekti u sceni imaju inerciju, odnosno da zadržavaju brzinu kretanja kroz vrijeme. Na osnovu navedenih pretpostavki, algoritam 3DRS koristi prostorno-vremensko susjedstvo u polju vektora kao kandidate za novoizračunati vektor na određenoj poziciji, uz dodatak ažuriranja, odnosno dodavanja dodatnog pomaka iz predefiniranog skupa na pojedine kandidate, kao sredstvo da algoritam pronađe ispravni vektor pomaka. Blok se evaluira izračunavanjem sume apsolutnih razlika između izvorišnog bloka u prvoj slici i bloka na koji pokazuje vektor-kandidat u drugoj slici, pri čemu blok s najmanjom vrijednosti postaje pobjednik. Ovakav se postupak

evaluacije kandidata ponavlja za svaki blok u slici, pri čemu se svi blokovi u slici obilaze u meandrirajućem redoslijedu iteraciju za iteracijom, mijenjajući smjer pri svakoj iteraciji. Svojstvo 3DRS algoritma da nakon konačnog broja iteracija rezultira ispravnim poljem vektora neovisno o početnom položaju nazivamo *konvergencijom*. Dodatno je, uz svojstva, dan pregled temeljnih poboljšanja osnovnog algoritma te različite implementacije.

Prethodna i vezana istraživanja

U poglavlju 4 izložen je pregled dosadašnjih istraživanja u području pasivne trodimenzionalne stereo rekonstrukcije i izračuna dispariteta. Metode se ugrubo mogu podijeliti na lokalne, koje za svaki piksel u slici pokušavaju izračunati optimalnu vrijednost cijene uparivanja (eng.*matching cost*) i globalne, koje na osnovu pretpostavki o glatkoći mape dispariteta formiraju globalnu funkciju energije uparivanja za cijelu scenu, koju potom minimiziraju računajući optimalni disparitet za sve piksele odjednom. Globalne metode daju veoma kvalitetne rezultate, no iznimno su zahtjevne za računanje. Kompromis između kvalitete i brzine izvršavanja daju metode bazirane na dinamičkom programiranju, poput poluglobalnog uparivanja (eng. *semi-global matching*). Recentnu veliku popularnost i kvalitetu rezultata pokazuju metode temeljene na tehnikama dubokog strojnog učenja (eng. *deep learning*), no kod primjene istih pokazalo se da kvaliteta rezultata iznimno ovisi o kvaliteti ulaznog skupa podataka na kojoj se model trenira.

Dvije klasične metode koje su detaljno opisane su lokalni postupak optimizacije "pobjednik uzima sve" (eng. *Winner-take-all*) te klasično dinamičko programiranje. Kod lokalnih metoda dominantni utjecaj na kvalitetu rezultata, osim same funkcije cijene uparivanja, ostvaruje način na koji se cijena uparivanja akumulira, odnosno agregira unutar lokalne regije podrške. Opisani su različiti načini agregacije: klasični kvadratni, promjenjivi, pomični te prilagodiv (adaptivni) prozor. Kako je agregacija cijene često najzahtjevniji korak u izračunu dispariteta, često se za ubrzanje agregacije upotrebljava tehnika *integralnih slika* koje omogućuju zbrajanje bilo koje pravokutne pod-regije slike u $O(1)$ složenosti jednostavnim matematičkim operacijama između vrijednosti piksela u uglovima regije.

Dinamičko programiranje globalni je postupak optimizacije koji formira globalnu funkciju energije kao zbroj cijene uparivanja i funkcije glatkoće, pridružujući penalizacijsku vrijednost lokacijama u kojima se nalazi diskontinuitet u mapi dispariteta. Osnovne su pretpostavke da je zadovoljen uvjet poretka (eng. *ordering constraint*), odnosno da je unutar svake linije relativni poredak piksela koji odgovaraju istim objektima u sceni nepromijenjen između dvije slike (gledajući s lijeva na desno, objekti ne mijenjaju svoj relativni poredak, već samo pozicije u slici) te jedinstvenosti (eng. *uniqueness*)- da značajka u jednoj od slika ima najviše jedan par u drugoj slici. Pod tim uvjetima, moguće je izraziti za svaku liniju u slici funkciju dispariteta kao put u ravnini (x,d) gdje je x koordinata u liniji a d disparitet. Put se sastoji od tri moguća koraka koji odgovaraju situacijama sigurnog uparivanja (zadržavanje dispariteta), prekrivanja

(okluzije) značajke s lijeva, koje povećava disparitet te prekrivanja (okluzije) s desna, koja smanjuje disparitet. Ovako formuliran problem moguće je rastaviti u odluku o ispravnom koraku u svakoj točki puta. Ovaj je problem ekvivalentan problemu naprtnjače, rješivom korištenjem dinamičkog programiranja. U završnom je dijelu poglavlja izložen kratki pregled nekoliko funkcija cijena uparivanja istraženih u više relevantnih radova: sume apsolutnih razlika, sume kvadriranih razlika, normaliziranih suma apsolutnih i kvadriranih razlika te cenzus transformacije.

Materija i metode

U petom su poglavlju predložene nove, hibridne metode za izračun guste mape dispariteta s pomoću algoritma trodimenzionalnoga rekurzivnoga pretraživanja. Za algoritam 3DRS su definirani uvjeti rada - uz zadovoljenje pretpostavke da su ulazne slike epipolarno ispravljene, rad 3DRS-a možemo ograničiti isključivo na dimenziju paralelnu s epipolarnim pravcima. Algoritam 3DRS je ujedno proširen dodatnim mogućnostima: podrškom za korištenje više različitih funkcija cijena osim osnovne sume apsolutnih razlika te mogućnosti proizvoljnog određivanja veličine bloka za izračun cijene neovisno o veličini samog bloka unutar slike za koji se traži vektor dispariteta. Na osnovu ovakvog algoritma 3DRS definirane su nove hibridne metode. U hibridnim je metodama prvi korak uvijek izračun grube mape dispariteta algoritmom 3DRS, gdje je svim pikselima unutar pojedinog bloka dodijeljen isti disparitet. U drugom se koraku gusta mapa dispariteta izračunava putem jedne od klasičnih metoda: metodom "pobjednik uzima sve" (lokalnom optimizacijom) ili dinamičkim programiranjem (globalnom optimizacijom). Pritom se prostor pretraživanja dispariteta sužava ponovnom primjenom pretpostavke 3DRS-a da su objekti u sceni veći od blokova. Svakom se bloku pridružuje jedan ili više intervala mogućih dispariteta iz susjedstva blokova za koje je estimacija dobivena algoritmom 3DRS. Kako bi se uzela u obzir moguća varijacija, umjesto fiksne vrijednosti se svakom bloku pridružuje interval dispariteta, pri čemu su konačni intervali dispariteta dobiveni unijom svih intervala iz susjedstva. Kod lokalnog se algoritma "pobjednik uzima sve" gusti se algoritam izvršava nad blokom te se za svaki piksel u navedenom bloku dodjeljuje konačni disparitet iz intervala dobivenih iz susjedstva. Kod globalnog se algoritma formira prostor dispariteta (x,d) za svaku liniju koristeći intervale dispariteta za blokove kojima linija prolazi, osiguravajući pritom da je moguće konstruirati put kroz prostor dispariteta od ruba do ruba, koji je moguće optimirati dinamičkim programiranjem.

Rezultati i diskusija

Poglavlje 6 iznosi rezultate evaluacije algoritama. Kao preduvjet postavljena je metodologija. Točnost rezultata utvrđena je postotkom neispravnih piksela u mapi dispariteta, pri čemu se is-

pravnost piksela utvrđuje odstupanjem od vrijednosti utvrđenih stvarnih dispariteta (eng. *ground truth*) za iznos veći od određenog praga (prag od 1 piksela je određen za potrebe evaluacije, gdje je vrijednost preuzeta iz relevantnih istraživanja). Performanse se algoritma utvrđuju mjerenjem te naknadnom usporedbom vremena izvođenja algoritama na referentnoj računalnoj platformi. Za ispitivanje algoritama odabran je *Middlebury* skup stereo slika sa poznatim utvrđenim disparitetima i mapama koji određuju specifične zone slika. Kako je Middlebury skup u širokoj primjeni kao etalon u području istraživanja, njegova primjena omogućuje jednostavnu usporedbu rezultata sa rezultatima drugih algoritama. Za potrebe evaluacije metoda u ovom su radu korištene četiri standardne slike iz Middlebury skupa - "Tsukuba", "Venus", "Teddy" i "Cones".

Inicijalno su mjerenja usmjerena na algoritam 3DRS radi utvrđivanja njegovih svojstava i podobnosti za ciljane primjene u računalnom vidu. Za više je različitih dimenzija slika izmjerena ovisnost vremena izvođenja algoritma 3DRS o veličini bloka. Utvrđeno je da vrijeme izvođenja pada sa porastom veličine bloka, do veličine bloka od 4 piksela, nakon čega je vrijeme izvođenja konstantno i ovisno jedino o veličini ulazne slike, neovisno o veličini bloka. Mjerenjem vremena izvođenja u ovisnosti o broju prolaza utvrđeno je da vrijeme izvođenja algoritma 3DRS raste linearno s brojem prolaza. Mjerenje točnosti provedeno je u ovisnosti o veličini bloka, broju prolaza algoritma, te veličini skupa iz kojeg algoritam 3DRS preuzima vrijednosti ažuriranja vektora. U ovisnosti o broju iteracija algoritma utvrđeno je da algoritam daje rezultat blizak optimalnom, bez značajnih poboljšanja daljnjim iteracijama, nakon samo dvije iteracije algoritma, što je u skladu sa načinom na koji se u algoritmu 3DRS osigurava propagacija novoestimiranih vrijednosti. U ovisnosti o veličini bloka, rezultat postiže svoj maksimum za blokove veličine četiri piksela i veće, što empirijski pokazuje granicu unutar koje je moguće osigurati dovoljan kontekst za optimalno raspoznavanje značajki prilikom uparivanja. U ovisnosti o veličini skupa ažuriranja pokazalo se da za maksimalnu vrijednost ažuriranja u iznosu 16 piksela algoritam 3DRS daje optimalan rezultat - uz manje vrijednosti maksimalnog ažuriranja algoritam pokazuje oštar pad brzine konvergencije sa padom veličine skupa (do izostanka konvergencije), dok uz vrijednosti ažuriranja veće od 16 piksela točnost rezultata lagano opada pod utjecajem šuma koje unose veći pomaci dispariteta. Točnost algoritma 3DRS ispitana je i u ovisnosti o odabranoj funkciji cijene, gdje se pokazalo da funkcija cijene temeljena na cenzus transformaciji slika i Hammingovoj udaljenosti daje najkvalitetnija rješenja u odnosu na druge ispitane funkcije cijena.

Hibridni algoritam u kojem je postupak optimizacije "pobjednik uzima sve" vođen algoritmom 3DRS evaluiran je za četiri odabrane slike iz Middlebury skupa. U odnosu na raniju evaluaciju algoritma, uvođenjem cenzus transformacije kao funkcije cijene dovelo je do unaprjeđenja kvalitete rezultata, dok su dodatna podešenja algoritma 3DRS rezultirala dodatnim ubrzanjem. Ispitana je ovisnost kvalitete izlazne mape dispariteta i brzine izvršavanja o vri-

jednosti parametra *r* koji definira širinu osnovnog intervala dispariteta izdvojenog iz rezultata algoritma 3DRS. Rezultati su pokazali da navedeni interval može biti smanjen na ± 1 piksel dispariteta bez negativnih utjecaja na kvalitetu. Analiza je pokazala da je postotak pogrešnih piksela kod predložene metode manji nego u obje originalne metode (algoritmu 3DRS i optimizaciji "pobjednik uzima sve"), te da je predložena metoda ostvarila 6,45 puta poboljšano vrijeme izvođenja u odnosu na originalni algoritam "pobjednik uzima sve".

Metoda dinamičkog programiranja daje sveukupno najbolje rezultate u ovom istraživanju po pitanju točnosti, no uz vrijeme izvođenja od preko 1,5 sekunde u prosjeku. Kod kombinacije algoritma 3DRS i optimizacije temeljene na dinamičkom programiranju, utvrđena je manja ovisnost kvalitete rezultata o kvaliteti grube mape dispariteta dobivene algoritmom 3DRS u odnosu na rezultate hibridnog algoritma temeljenog na optimizaciji "pobjednik uzima sve". S tim u sprezi, analiza je pokazala da je kod hibridnog algoritma koji kombinira algoritam 3DRS i dinamičko programiranje postotak pogrešnih piksela povećan, no ne više od 0,3% u prosjeku - čime možemo ustvrditi da je globalno nivo kvalitete rezultata zadržan, uz 6,22 puta poboljšano vrijeme izvođenja u odnosu na originalni algoritam dinamičkog programiranja.

Zaključak

U ovom je radu istražen problem stereo vida i trodimenzionalne rekonstrukcije sa fokusom na izračun dispariteta u stereoslikama. Kroz pregled dosadašnjih istraživanja izdvojen je veliki broj različitih rješenja koja su računalno vrlo zahtjevna. Glavna ideja ovog rada bila je iskoristiti grubi algoritam za uparivanje blokova sa dokazano brzim vremenom izvođenja kao temelj hibridnih algoritama koji bi postigli bolje rezultate od izvornih. Odabran je algoritam trodimenzionalnoga rekurzivnoga pretraživanja kao algoritam odgovarajućih svojstava koji je našao široku primjenu u potrošačkoj elektronici, no ne i u području računalnog vida. Klasični pristup "pobjednik uzima sve" i optimizacija dinamičkim programiranjem odabrani su kao kandidati za hibridne metode. Kako je algoritam 3DRS uglavnom u literaturi istražen u kontekstu primjene u potrošačkoj elektronici, njegova su svojstva ispitana empirijski. Dobiveni rezultati pokazali su kako za tipične radne uvjete algoritam 3DRS proizvodi grube mape dispariteta u konstantnom vremenu koje je ovisno isključivo o dimenziji ulaznih slika i broju iteracija neovisno o zadanoj veličini bloka. Dodatno su rezultati pokazali kako algoritam 3DRS konvergira ka rješenju bliskom optimalnom za određene postavke u najviše dvije iteracije uz dovoljno velik skup vektora ažuriranja. Na taj je način pokazana mogućnost algoritma 3DRS da rezultira grubom procjenom dispariteta relativno visoke točnosti u predvidivom, kratkom vremenu neovisnom o rasponu dispariteta koji je potrebno pretražiti. Ispitivanjem različitih funkcija cijena utvrđeno je da korištenje funkcije cijene temeljene na cenzus transformaciji rezultira drastičnim poboljšanjem izlazne mape dispariteta u odnosu na originalnu implementaciju. Naposljetku, korištenjem algoritma 3DRS kao inicijalnog grubog estimatora i primjenom načela istog kako

prostorno susjedstvo čini kvalitetne kandidate/prediktore dispariteta određene regije, formirane su dvije nove metode za izračun dispariteta u stereoslikama koje koriste algoritam 3DRS kao grubi inicijalni korak te klasični algoritam kao finalni korak koji daje gustu mapu dispariteta: 3DRS-om vođen algoritam "pobjednik uzima sve" te 3DRS-om vođeno dinamičko programiranje. Predložene metode unaprjeđuju ili zadržavaju točnost originalnih metoda ubrzavajući pritom vrijeme izvršenja za $6,2$ do $6,5$ puta u prosjeku. Buduća bi se istraživanja u ovom području trebala fokusirati na iskorištavanje drugih korisnih svojstava algoritma 3DRS poput estimacija dvodimenzionalnih vektora, poboljšanje točnosti hibridnih metoda, ili poboljšanje performansi uz krajnji cilj definiranja arhitekture za ugradnju algoritma u ugradbeni računalni sustav za rad u stvarnom vremenu.

Ključne riječi: **Stereo uparivanje, disparitet, 3DRS, Hibridna metoda, Pobjednik-uzima-sve, Dinamičko programiranje**

Contents

Chapter 1

Introduction

Computer vision is an interdisciplinary scientific field with the goal of enabling computers to obtain high-level understanding based on digital images. As an engineering discipline, it is tasked with devising autonomous systems which can perform task the human visual system is capable of [1]; from processing raw images, extracting features, performing segmentation, obtaining information about the structure of objects from either motion or stereo images, leading to higher levels of abstraction such as recognizing objects [2]. Concepts and methods of computer vision have been applied to a wide variety of real world uses, such as optical character recognition (OCR), machine inspection, photogrammetry, face detection, automotive driver assistance systems (ADAS), robotics, etc. One of the extensively researched problems in computer vision is that of stereo 3D reconstruction, where the goal is to obtain a three-dimensional model of a scene based on an image. Most often it is desired to obtain a dense map of the scene depth. The methods for dense stereo 3D reconstruction can be roughly divided into active and passive methods. Active methods are all those which use a light source, either to combine one or more cameras with a structured light source to encode information in the scene [3], or use a light source to determine the distance by measuring the time of flight for photons, such as LIDAR or a time-of flight camera. Passive methods, on the other hand, are based on extracting depth information from correspondences between multiple views of the scene (but most commonly between the two images in a stereo image pair), mimicking the way humans perceive their surroundings using binocular vision.

1.1 Motivation

Computing correspondent features in two images has also been the subject of extensive study in the field of motion estimation, where a motion map (also called the *optical flow*) of objects is generated for a sequence of images [4]. In this application, the correspondences are computed not between the two images of the same scene (as it is for stereo matching), but for consecutive

images within a sequence to determine the movement of features in the scene. The methods used for motion estimation can be roughly divided in two major categories [5]: the residue-minimizing methods, which are mostly used in video compression algorithms, and true motion estimators, used in video up-conversion and deinterlacing. The residue-minimizing methods usually perform a full-search block matching strategy in order to find the global minimum of the matching cost for each block. From the standpoint of image compression, this is desirable in order to minimize the residual signal when subtracting the two frames using the estimated displacement vectors, but the resultant vector field may differ significantly from the actual motion field. For frame rate conversion, where a high correlation with the true object motion is desired for the motion vector field, it is required to estimate the actual motion field in order to achieve visually pleasing results in frame rate interpolation.
One of the most successful true motion estimation methods is the Three-Dimensional Recursive Search (3DRS) [6], widely used due to its simplicity, low cost implementation, and the ability to provide good true motion estimates. The method has been successfully applied for real-time processing of 60 Hz high definition video and frame-rate conversion with frame rates up to 400 Hz, with specialized ASIC implementations [7]. As true motion vectors are desired for disparity estimation, 3DRS has been shown to be able to extract coarse disparity maps from image pairs [8] [9]. Given the comparative performance of 3DRS, its widespread use and overall simplicity, the goal of this research was to establish whether 3DRS can be utilized to enhance the existing dense stereo estimation methods and improve on their performance by decreasing their computational complexity and improving the delivered quality of the resultant computed disparities. For this purpose, a local method (Winner-take-all) and a global method (Dynamic programming) for disparity computation is selected to be combined with 3DRS.

1.2 Overview

This dissertation is organized as follows.
Chapter 2 provides an overview of the background of stereo 3D reconstruction, lays out the fundamental terms and provides the description of the approaches and methods this thesis is focused on.
Chapter 3 provides a comprehensive overview of the three-dimensional recursive search algorithm, its definition and its properties. The algorithm is presented in the light of its original purpose of true motion estimation, with the related work and most common enhancements of the basic algorithm.
In chapter 4 a brief overview of the related work and prior art on the selected local and global method, as well as matching costs, is presented.
Chapter 5 presents the main contributions of this dissertations with descriptions of the proposed

local and global hybrid methods combining the 3DRS algorithm with the selected local and global passive stereo methods. The results and performance of the hybrid methods are presented and compared with the performance and results of the original methods in chapter 6, accompanied by the discussion of the results and outlines of the key improvement points.
Finally, chapter 7 presents the conclusion of the thesis and a proposal for potential future research directions.

Chapter 2

Stereo reconstruction

2.1 Projective geometry

2.1.1 Primitives

In order to understand stereo imaging and 3D reconstruction it is necessary to form a geometric description of how the 3D world is mapped to the 2D image space, which can be modeled through *projective geometry*[10]. The relation which maps a set of points in a three-dimensional space to a set of points in a two dimensional plane (the image) is called a *projective transform*[11]. Such transforms can be conveniently expressed using *homogeneous coordinates*. The homogenous coordinates of a point in a projective space of dimension n are expressed as a $n+1$-dimensional vector, with the restriction that the points whose values are proportional are equivalent points.

2D Points

Points in two-dimensional (2D) space can be denoted using a pair of values $X = (x, y) \in \Re^2$ or by a 2-dimensional vector. Represented using homogenous coordinates[2], the 2D points are expressed as $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{w}) \in \mathbf{P}^2$. The space $\mathbf{P}^2 = \Re^3 - (0,0,0)$ is called the 2D *projective space*. A homogeneous vector can be converted back into an inhomogeneous vector by dividing it through by the last element $\tilde{w}$, yielding:

$$\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{w}) = \tilde{w}(x, y, 1) = \tilde{w}\bar{x}, \tag{2.1}$$

where the vector $\bar{x} = (x, y, 1)$ is called the *augmented vector*. Homogeneous points where $\tilde{w} = 0$ are called *ideal points* or *points at infinity*, which do not have an equivalent inhomogeneous representation.

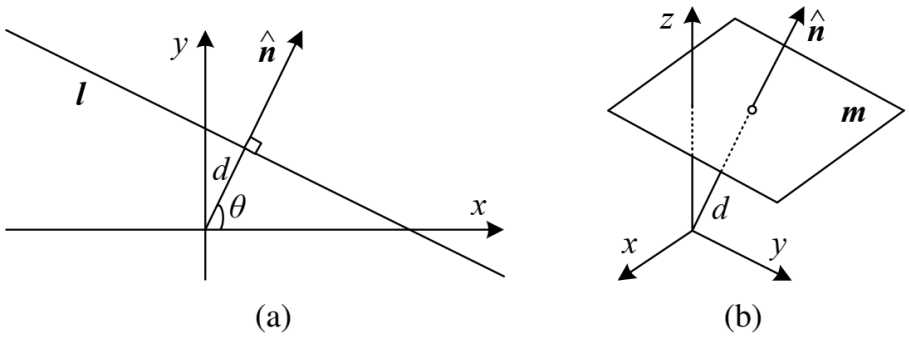


Figure 2.1: (a) 2D line and (b) 3D plane equation expressed with a normal vector $\tilde{\mathbf{n}}$ and distance d. Adapted from [2]

2D Lines

2D lines can also be represented using homogeneous coordinates $\tilde{l} = (a,b,c)$. The corresponding line equation is

$$\tilde{\mathbf{x}}\tilde{l} = ax + by + c = 0 \tag{2.2}$$

The vector of the line equation 2.2 can be normalized so that $l = (\hat{n_x}, \hat{n_y}, d) = (\hat{\mathbf{n}}, d)$ with $\|\hat{\mathbf{n}}\| = 1$, in which case $\hat{\mathbf{n}}$ is the *normal vector* perpendicular to the line and d its distance to the origin (exception to this is the *line at infinity*, $\tilde{l} = (0,0,1)$), as shown in figure 2.1 (a).

Using homogeneous coordinates, the intersection of two lines can be computed:

$$\tilde{\mathbf{x}} = \tilde{l}_1 \times \tilde{l}_2 \tag{2.3}$$

Similarly, a line joining two points can be written as

$$\tilde{\mathbf{l}} = \tilde{x}_1 \times \tilde{x}_2 \tag{2.4}$$

2D Conics

Conic sections can be defined using a *quadric equation*

$$\tilde{\mathbf{x}}^\top Q \tilde{\mathbf{x}} = 0 \tag{2.5}$$

3D Points

Points in three-dimensional (3D) space can be denoted using a triplet of values $X = (x,y,z) \in \Re^3$ or by a 3-dimensional vector, also using homogeneous coordinates $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{z}, \tilde{w}) \in \mathbf{P}^3$. It is occasionally convenient to use an augmented vector to define a 3D point $\bar{x} = (x,y,z,1)$, with

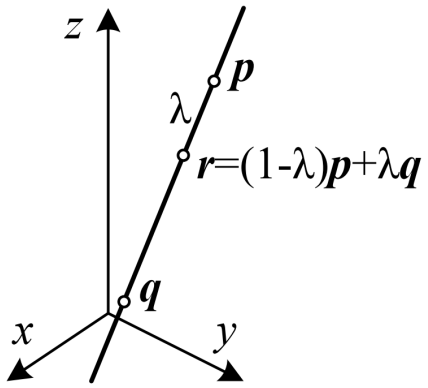


Figure 2.2: (a) 3D line equation. Adapted from [2]

$\tilde{\mathbf{x}} = \tilde{w}\tilde{\mathbf{x}}$.

3D Planes

A 3D plane can be represented as homogeneous coordinates $\tilde{m} = (a,b,c,d)$ with a corresponding plane equation:

$$\tilde{\mathbf{x}}\tilde{m} = ax + by + cz + d = 0 \tag{2.6}$$

which can be normalized as:

$$\begin{aligned} l = (\hat{n_x}, \hat{n_y}, \hat{n_z}, d) = (\hat{\mathbf{n}}, d) \\ \|\hat{\mathbf{n}}\| = 1 \end{aligned} \tag{2.7}$$

In eq. 2.7, $\hat{\mathbf{n}}$ is the normal vector perpendicular to the plane and d is the distance from the plane to origin, as shown in figure 2.1 (b).

3D Lines

A line in 3D space can be represented by two points $(\mathbf{p}, \mathbf{q})$. All other points in the line can be expressed as a linear combination of the two points:

$$\mathbf{r} = (1-\lambda)\mathbf{p} + \lambda\mathbf{q} \tag{2.8}$$

as shown in the figure 2.2.

3D Quadrics

An analog of a conic section in 3D space is a quadric surface.

$$\tilde{\mathbf{x}}^\top Q\tilde{\mathbf{x}} = 0 \tag{2.9}$$

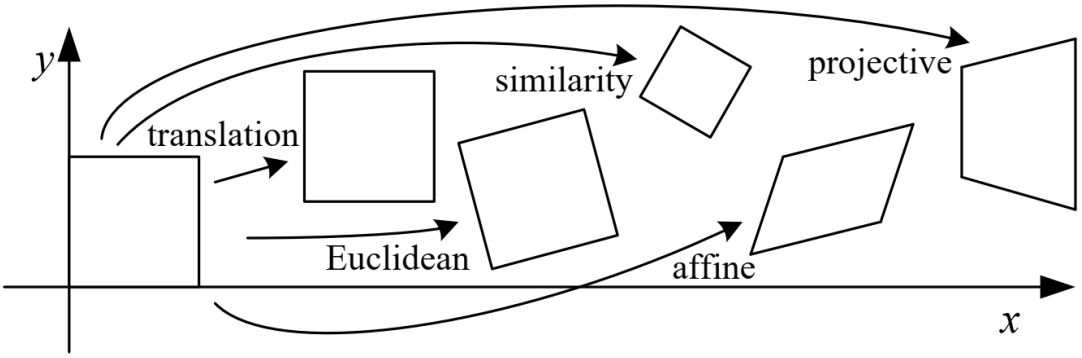


Figure 2.3: Basic set of coordinate transformations, with degrees of freedom increasing with x. The transformation is shown for 2D but the concepts extend to 3D. Adapted from [2]

2.1.2 Transformations

Having defined the primitive geometric elements in a projective space, geometrical transformations of various properties, as shown in figure 2.3 can be performed on them.

Translation

The simplest of the transforms is translation, which shifts the geometric primitive in space while retaining its shape, can be defined as $x' = x + \mathbf{t}$, or

$$x' = \begin{bmatrix} \mathbf{I} & \mathbf{t} \end{bmatrix} \bar{x} \tag{2.10}$$

where $\mathbf{I}$ is the (2x2) identity matrix. Alternatively it can be defined as

$$\bar{x}' = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \bar{x} \tag{2.11}$$

where $\mathbf{0}$ is the zero vector. Although with the (2×3) matrix such as in the Eq. 2.10 a more compact notation can be obtained, with a (3×3) matrix as in Eq. 2.11 it is possible to chain transformations with matrix multiplication. $\bar{x}$ is the augmented vector, which can be replaced at any time with the full homogeneous vector. In the 3D case, identical equation as the Eq. 2.10 applies, with the $\mathbf{I}$ being a (3×3) matrix.

Rigid(Euclidean) transform

A *rigid-body* or *Euclidean* transformation preserves the Euclidean distances. It is a combination of translation and rotation, which can be defined as $x' = \mathbf{R}x + \mathbf{t}$, or

$$x' = \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \bar{x} \tag{2.12}$$

where $\mathbf{R}$ is an orthonormal rotation matrix

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{2.13}$$

where θ is the angle of rotation. For the rotation matrix, $\mathbf{R}\mathbf{R}^\top = \mathbf{I}$ and $|\mathbf{R}| = 1$. For the 3D case, $\mathbf{R}$ is a (3×3) matrix, but its construction is more complex than the 2D case and dependent on how the rotation in 3D space is described. Popular approaches use *quaternions* or the *Rodrigues transform* as used in the OpenCV library[11], shown in Eq. 2.14. Let $\vec{r}$ be a three-dimensional vector $\vec{r} = [r_x, r_y, r_z]$ which implicitly defines θ as the magnitude of rotation by the length (or magnitude) of $\vec{r}$. It is possible to convert from the axis-magnitude representation to a rotation matrix R:

$$\mathbf{R} = \cos\theta \cdot \mathbf{I}_3 + (1 - \cos\theta) \cdot \vec{r} \times \vec{r}^\top + \sin\theta \cdot \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix} \tag{2.14}$$

Similarity transform

A similarity transform, or *scaled rotation*, can be expressed as $x' = s \cdot \mathbf{R}x + \mathbf{t}$, where s is an arbitrary scale factor. It can also be expressed as (for 2D) as

$$x' = \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \bar{x} = \begin{bmatrix} a & -b & t_x \\ b & a & t_y \end{bmatrix} \bar{x} \tag{2.15}$$

Unlike the equations 2.12 and 2.13, it is no longer required that $a^2 + b^2 = 1$. In 3D space, the transformation is the same with the vectors being three-dimensional and R being the (3×3) rotation matrix. The similarity transform preserves the angles between lines.

Affine transform

The affine transformation is defined as $x' = \mathbf{A}\bar{x}$. For the 2D case A is an arbitrary (2×3) matrix

$$\mathbf{A} = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \tag{2.16}$$

Under the affine transformation parallel lines remain parallel. In the 3D case the matrix A is an arbitrary (3×4) matrix.

$$\mathbf{A} = \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \end{bmatrix} \tag{2.17}$$

For the 3D affine transformation, parallel lines and planes remain parallel.

Projective transform

The projective transform, also known as a *perspective transform* or *homography*, is defined on homogeneous coordinates as:

$$\tilde{x}' = \tilde{\mathbf{H}}\tilde{x} \tag{2.18}$$

where $\tilde{\mathbf{H}}$ is an arbitrary (3×3) matrix in 2D, and a (4×4) matrix in 3D space. As $\tilde{\mathbf{H}}$ is homogeneous, two matrices which differ only by scale are equivalent. The resultant homogeneous coordinate $\tilde{x}'$ must be normalized in order to obtain the inhomogeneous result x'.

2.1.3 Hierarchy of transformations

The transformations listed above form a nested set of *groups*, signifying that they are closed under composition and have an inverse transform which is the member of the same group. Each simpler group is a subset of the more complex group below it, with the more complex group offering more degrees of freedom in the transformation. The hierarchy of transformations for 2D and 3D cases can be found in figures 2.4 and 2.5, respectively.

2.2 The camera model

A camera defines a mapping between the 3D world and a 2D image. For the purpose of this work *central projection cameras* [10], which form the image by projecting the points in the 3D space through a central point, are of interest. A *finite camera* has the central projection

Transformation	Matrix	# DoF	Preserves	Icon
translation	$\left[\begin{array}{c\|c} \boldsymbol{I} & \boldsymbol{t} \end{array}\right]_{2\times3}$	2	orientation	
rigid (Euclidean)	$\left[\begin{array}{c\|c} \boldsymbol{R} & \boldsymbol{t} \end{array}\right]_{2\times3}$	3	lengths	
similarity	$\left[\begin{array}{c\|c} s\boldsymbol{R} & \boldsymbol{t} \end{array}\right]_{2\times3}$	4	angles	
affine	$\left[\begin{array}{c} \boldsymbol{A} \end{array}\right]_{2\times3}$	6	parallelism	
projective	$\left[\begin{array}{c} \tilde{\boldsymbol{H}} \end{array}\right]_{3\times3}$	8	straight lines	

Figure 2.4: Hierarchy of 2D coordinate transformations. Each transformation also preserves the properties listed in the rows below it. the (2×3) matrices can be extended to (3×3) with a third $[\mathbf{0}^\top 1]$ row to enable homogeneous coordinate transformations. Adapted from [2]

Transformation	Matrix	# DoF	Preserves	Icon
translation	$\left[\begin{array}{c\|c} \boldsymbol{I} & \boldsymbol{t} \end{array}\right]_{3\times4}$	3	orientation	
rigid (Euclidean)	$\left[\begin{array}{c\|c} \boldsymbol{R} & \boldsymbol{t} \end{array}\right]_{3\times4}$	6	lengths	
similarity	$\left[\begin{array}{c\|c} s\boldsymbol{R} & \boldsymbol{t} \end{array}\right]_{3\times4}$	7	angles	
affine	$\left[\begin{array}{c} \boldsymbol{A} \end{array}\right]_{3\times4}$	12	parallelism	
projective	$\left[\begin{array}{c} \tilde{\boldsymbol{H}} \end{array}\right]_{4\times4}$	15	straight lines	

Figure 2.5: Hierarchy of 3D coordinate transformations. Each transformation also preserves the properties listed in the rows below it. the (3×4) matrices can be extended to (4×4) with a fourth $[\mathbf{0}^\top 1]$ row to enable homogeneous coordinate transformations. Adapted from [2]

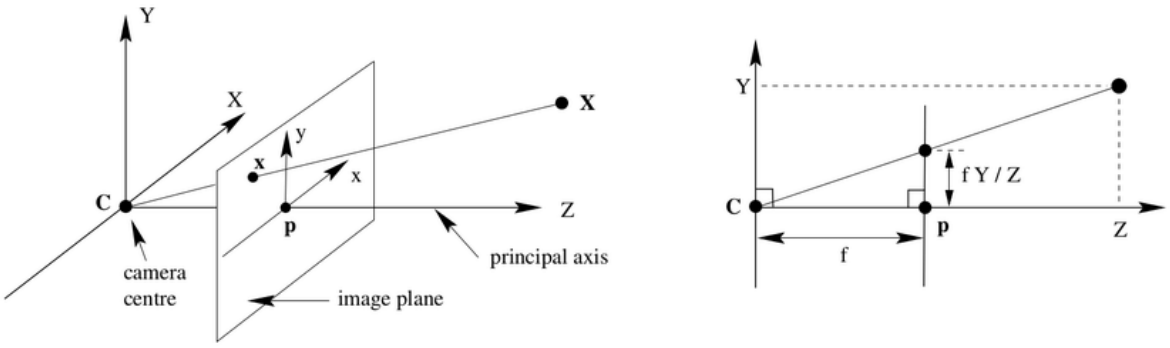


Figure 2.6: The pinhole camera geometry, adapted from [10]

point defined at finite coordinates. A central projection camera is a specialization of the *general projective camera*, for which a model can be defined through a matrix defining the projective transformation.

2.2.1 Pinhole camera

The most basic central projection camera model is the *pinhole camera model*. In a pinhole camera model, the center of projection is placed in the origin of a Euclidean coordinate system, and a plane $Z = f$, which is called the *image plane* or the *focal plane*, is defined as shown in figure 2.6. In a pinhole camera model, a point in 3D space (denoted with the coordinates $X = (X,Y,Z)^\top$) is mapped to a point at the focal plane in which a straight line passing through the coordinate system origin and the point intersects with the plane. The projection mapping can be described as

$$(X,Y,Z)^\top \rightarrow (fX/Z, fY/Z)^\top \tag{2.19}$$

which is a mapping from the Euclidean 3-space $\Re^3$ to Euclidean 2-space $\Re^2$. The center of the projection is called the *camera center*, or the *optical center*. A line constructed so that it is perpendicular to the image plane and passing through the camera center is the *principal axis* of the camera (also called the *optical axis*), which intersects the image plane at the camera's *principal point*. A plane parallel with the image plane passing through the camera center is called the *principal plane* of the camera. By using homogeneous coordinates, it is simple to express the camera transformation as a matrix:

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.20}$$

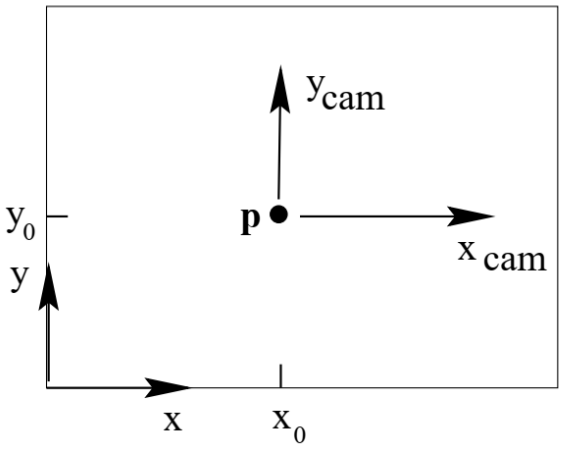


Figure 2.7: Offset of the principal point in the image plane and the image coordinate system, adapted from [10]

in which case equation 2.20 can be written as

$$\vec{x} = P\vec{X} \tag{2.21}$$

where P is the *camera projection matrix*. The expression assumes that the principal point is also the origin of the coordinates in the image plane, which may not be the case in practice. In that case it is necessary to account for the offset as shown in the figure 2.7 by adding additional parameters (p_x, p_y) which extend the camera matrix P defined in eqs. 2.20 and 2.21 to:

$$P = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.22}$$

and the transformation from eq. 2.19 to

$$(X,Y,Z)^\top \rightarrow (fX/Z + p_x, fY/Z + p_y)^\top \tag{2.23}$$

If the matrix K is defined as

$$K = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.24}$$

it is possible to express equation 2.21 as

$$\vec{x} = K[I|0]\vec{X} \tag{2.25}$$

The matrix K is called the *camera calibration matrix* or the camera *intrinsics matrix*. In the general case it is also considered, in practice usually due to construction of the actual imaging device, that the focal lengths in x and y directions are not equal. Additionally a parameter s, or *skew* accounts for the possibility of the image plane of the device not being mounted perfectly perpendicular to the principal axis. Therefore the final form of the matrix is

$$K = \begin{bmatrix} f_x & s & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.26}$$

Considering that the parameter s is usually zero, the result is a simple model in which a point $\vec{X}$ in the physical world with the coordinates $(X,Y,Z)\top$ is projected to the image point with the coordinates (x_{img}, y_{img}) computed as:

$$\begin{aligned} x_{img} &= f_x \cdot \frac{X}{Z} + p_x \\ y_{img} &= f_y \cdot \frac{Y}{Z} + p_y \end{aligned} \tag{2.27}$$

2.2.2 Camera orientation

The camera coordinate system is defined in respect to the center of projection, the principal / optical axis and the image plane, however in general, the points in 3D space will have a different coordinate system. To define the mapping between the real-world coordinates and the points in the image, a rotation and a translation operation is required. In general the mapping of the point in the camera coordinate system $\tilde{X}_{img}$ can be defined as:

$$\tilde{X}_{cam} = R(\tilde{X} - \tilde{C}) \tag{2.28}$$

where $\tilde{X}$ is an inhomogeneous 3-vector representing the world coordinates, $\tilde{C}$ is a similar vector representing the coordinates of the camera center, and R is a 3x3 rotation matrix between the camera and real world Euclidean 3D space. Combining this with 2.25, the result is:

$$\vec{x} = KR[I| - \tilde{C}]\vec{X} \tag{2.29}$$

as a general mapping of a pinhole camera.

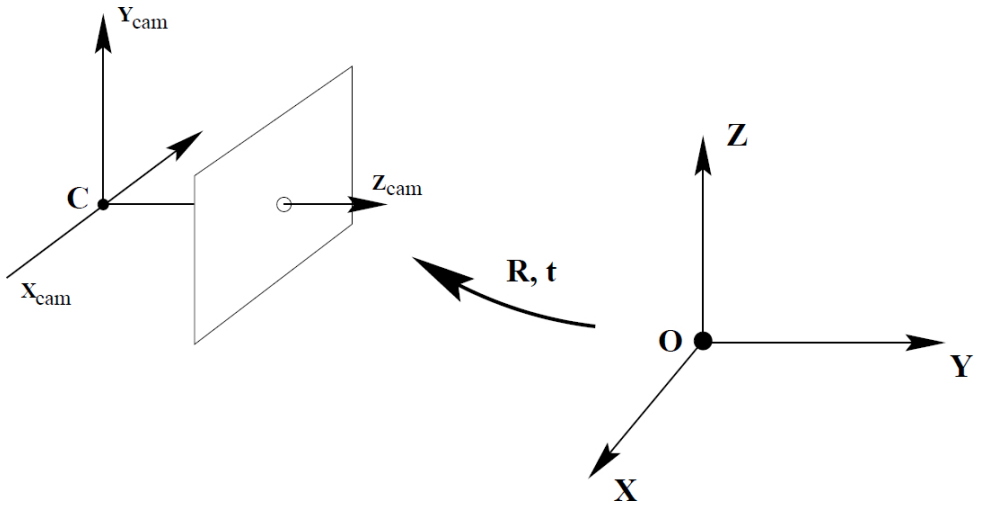


Figure 2.8: transformation between the real world and camera Euclidean space, adapted from [10]

The rotation matrix R and the position vector $\tilde{C}$ are called *extrinsic camera parameters*, which describe the *pose* of the camera. The camera center can also be made non-explicit for convenience, in which case the world-to-camera transformation can be defined as $X_{img} = R\tilde{X} + t$, where $t = -R\tilde{C}$. In this case the camera matrix is:

$$P = K[R|t] \tag{2.30}$$

where $t = -R\tilde{C}$. The transformation is shown in figure 2.8.

2.3 Stereo camera system

From the earliest inquiries into visual perception it has been known that humans perceive depth based on the differences in appearance between the left and right eye. The word "Stereo" comes from the Greek word for "solid" - stereo vision is how humans perceive solid shapes [2]. Under simple imaging configurations, with the both cameras or eyes looking straight ahead, the *disparity*, which is the amount of horizontal motion of objects in a scene between two views is inversely proportional to the distance from the point of observation. It is possible to show this principle on a simplified ideal Stereo setup example given in figure 2.9. In an ideal stereo setup, let us assume the perfect conditions for imaging: that the two cameras have image planes which are exactly co-planar with each other, with exactly parallel optical axes that are located at a well defined distance, and that they have lenses with exactly equal focal lengths $f_1 = f_2 = f$. Additionally let us assume that the principal points of the cameras $(c_x^{left}, c_x^{right})$ have been calibrated so that they have the identical pixel coordinates in the image of the left and right camera, respectively. Additional assumptions should apply: the imagers of the cameras are perfectly row-aligned. Furthermore, the assumption is that it is possible to find a point $\vec{P}$ in the physical

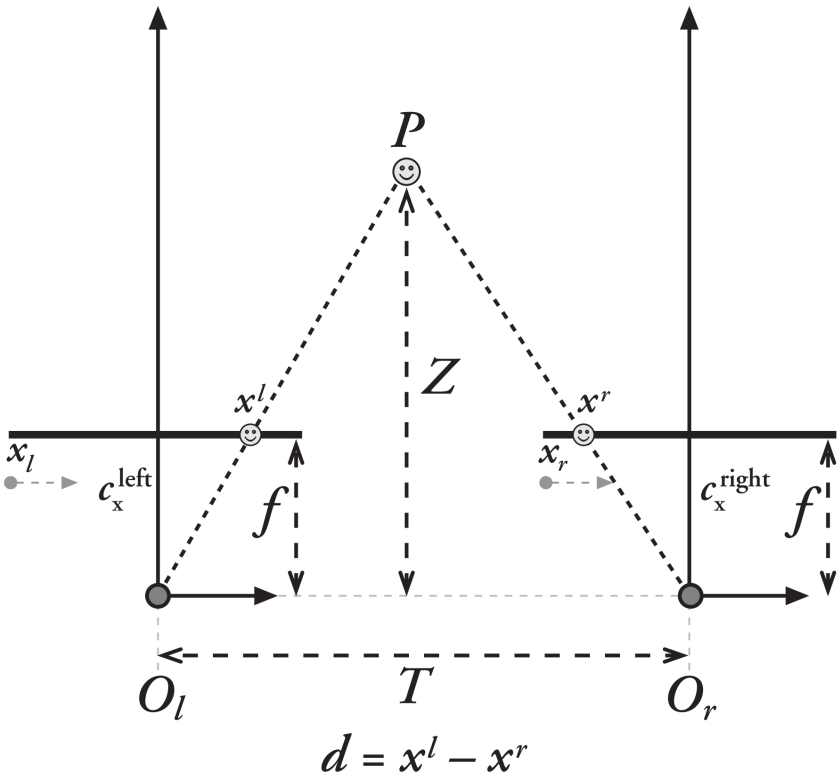


Figure 2.9: A model of an ideal stereo imaging setup, adapted from [11]

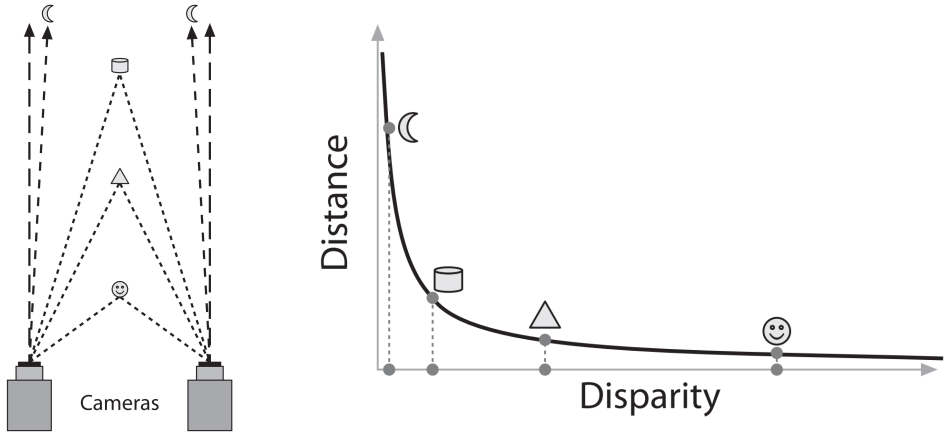
world with the left and right image views $\vec{p}_l$ and $\vec{p}_r$ with the respective horizontal coordinates x_l and x_r. For this extensively simplified case, it is apparent that the depth is inversely proportional to the disparity between the two images $d = x_l - x_r$, as it can be easily derived from the geometry through similar triangles:

$$\frac{T-(x_l-x_r)}{Z-f} = \frac{T}{Z} \implies Z = \frac{f \cdot T}{x_l - x_r} \tag{2.31}$$

As depth is inversely proportional to disparity, the consequence is that stereo vision systems have high depth resolution only for objects relatively near the camera, as shown in figure 2.10. In a practical situation, many of the assumptions of the ideal case are not applicable. For the understanding of how the cameras map the scene to their images and find the required correspondence, it is important to understand the geometry of a dual view camera system.

2.3.1 Epipolar geometry

The intrinsic projective geometry between the two camera views is called *epipolar geometry*. It is the geometry describing the intersection of the image planes with the pencil of planes having the line joining the centers of projection for the two cameras as the axis[10]. The line joining the two cameras' centers of projection is called the *baseline* of the stereo system. For the two



(a) Various objects at various distances from the stereo pair.

(b) Disparity in relation to distance from the camera

Figure 2.10: Relationship between depth and disparity; adapted from [11]

cameras, the centers of projection O_l and O_r with their corresponding projective planes Π_l and Π_r are defined. The point P in the physical world projects its image onto Π_l and Π_r as p_l and p_r, respectively. Epipolar geometry describes the relation between the points p_l and p_r. Observing the two camera centers O_l and O_r and the point P, it is evident that the three points in 3D space define a 3D plane. This plane contains the rays which are back-projected from the two points, and which are coplanar, is significant in establishing correspondence. This plane, Π_e, is called the *epipolar plane*. The epipolar plane interesects with the two imaging planes Π_l and Π_r. The lines obtained by intersecting the epipolar plane with the image planes are called *epipolar lines*.

It is also visible, as shown in the figure 2.11, that the baseline intersects with the two camera planes and defines points in the image planes Π_l and Π_r which are called the *epipoles* named e_l and e_r, respectively. For each of the two cameras, the epipoles are essentialy the images of the other camera's center of projection. For a stereo system, given all possible points P visible by both cameras, there is a one-parameter family (a pencil) of epipolar planes defined by the baseline, for which there is a corresponding family of epipolar lines in the two image planes, all of which interesect at the epipole. For a given point P and its location p_l in Π_l, it is possible to determine that the location of p_r lies in the plane Π_e, which in turn places p_r on the corresponding epipolar line in Π_r. The image of all possible locations of a point visible in one image is the line which passes through the corresponding point and the epipole of the other image. Given a feature in one image, its matching view in the other image must lie along the corresponding epipolar line. This is known as the *epipolar constraint*[11].

The properties of epipolar geometry are extremely important for computing stereo correspondence, given that:

- the epipolar constraint ensures that the corresponding point in the other imager resides on

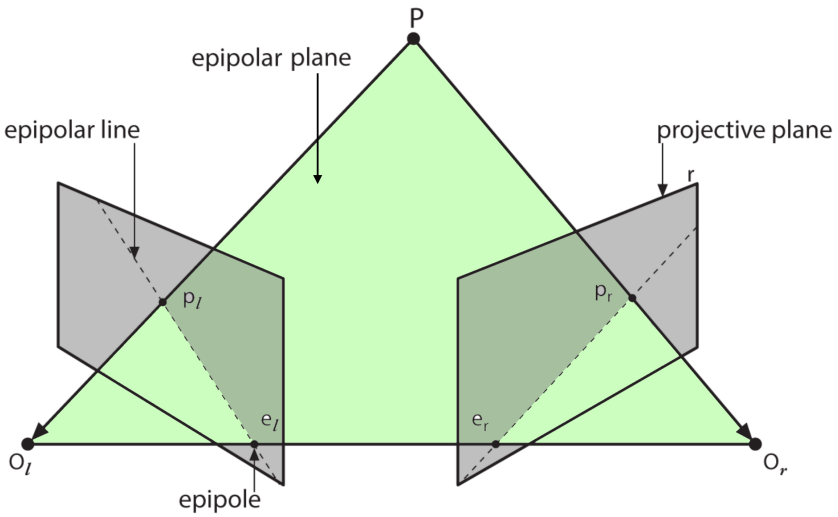


Figure 2.11: Epipolar geometry, intersection of epipolar plane with the image planes, adapted from [11]

the corresponding epipolar line. Therefore the correspondence becomes a one-dimensional instead of a two-dimensional search, vastly reducing the computational effort required to compute the correspondence and improving the accuracy.

- in the majority of conditions[12], the order is preserved on the epipolar line, if points A and B are visible in both images and occur horizontally in an order, the same order shall be preserved in the other imager. This effectively defines the search direction along the epipolar line, further simplifying the search for the corresponding feature. Even if a point is not visible in the other image due to occlusion, the order will still be preserved.

Essential and Fundamental Matrix

The geometry of the stereo system is defined by two matrices, which are called the *essential matrix* E and the *fundamental matrix* F.

Essential matrix

The essential matrix is a matrix which captures the essential geometric relation of the two camera models in physical space without taking into account their imaging properties. It is an Euclidean transform defined by rotation $\mathbf{R}$ and translation $\mathbf{t}$ which transforms the left camera model into the right camera model, as shown in figure 2.12.

Given a point P, the goal is to derive a relation that connects the projected points p_l and p_r of P on the two image planes in the respective camera coordinate system. Let P_l and P_r be the point P expressed in each camera's coordinate system, and let us choose the coordinate system for the left camera as the frame of reference. The location of the projected point is p_l in the image plane and the center of projection for the right camera is located at $O_r = \mathbf{t}$. The coordinates of

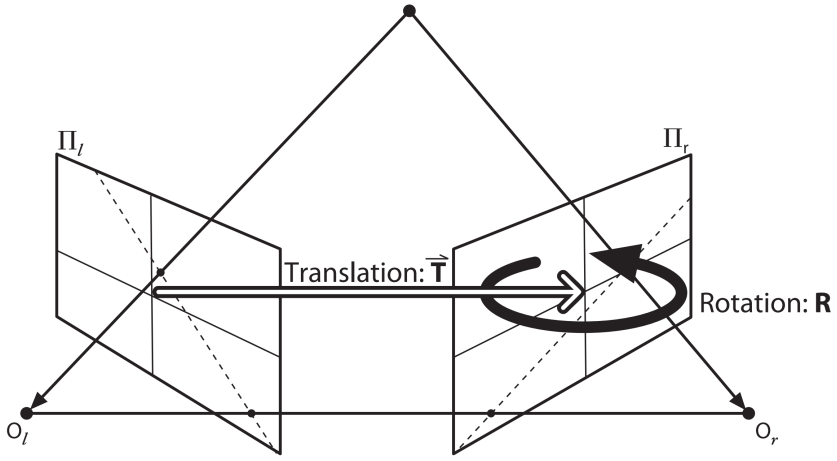


Figure 2.12: Epipolar geometry, transformations making up the essential matrix E, adapted from [11]

the 3D point P seen in the right camera is P_r in the right camera's coordinate system, where P_r can be expressed as

$$P_r = \mathbf{R} \cdot (P_l - O_r) \tag{2.32}$$

Introducing the epipolar plane into the equation, all points x on a plane with the normal vector $\vec{n}$ passing through the point a are constrained so that the following applies:

$$(x-a) \cdot \vec{n} = 0 \tag{2.33}$$

As the epipolar plane contains the points P_l and O_r , one can easily obtain the normal vector for the epipolar plane $\vec{n}$ through the cross product: $\vec{n} = P_l \times \mathbf{t}$. A plane equation containing all possible points P_l while passing through O_r and containing both vectors would be:

$$(P_l - O_r)^\top (O_r \times P_l) = 0. \tag{2.34}$$

Based on equation 2.32, it can be written that $(P_l - O_r) = R^{-1} P_r$. If it is also known that $R^\top = R^{-1}$, the following result is obtained:

$$(R^\top P_r)^\top (O_r \times P_l) = 0. \tag{2.35}$$

The cross product above can then be rewritten as matrix multiplication, if matrix S is defined so that:

$$(O_r \times P_l) = SP_l \implies S = \begin{bmatrix} 0 & -O_{rz} & O_{ry} \\ O_{rz} & 0 & -O_{rx} \\ -O_{ry} & O_{rx} & 0 \end{bmatrix} \tag{2.36}$$

Combining equations 2.35 and 2.36 yields:

$$(P_r)^\top RSP_l = 0 \tag{2.37}$$

Based on equation 2.37 it can be expressed that:

$$E = RS \implies (P_r)^\top EP_l = 0 \tag{2.38}$$

where E is the *essential matrix*. It can be observed that this describes a relation of 3D points, not their image coordinates. However if the equations 2.27 describing the camera projection are applied, it is possible to divide eq. 2.38 with $\frac{Z_l Z_r}{f_l f_r}$ to obtain:

$$p_r^\top E p_l = 0 \tag{2.39}$$

This equation does not completely specify the relationship between the two points as E is a rank-deficient matrix (the (3×3) essential matrix has a rank of 2). There are five parameters in the essential matrix: three for the rotation component, and two for the direction of translation (as the scale is not set). Therefore the essential matrix has only five degrees of freedom. Additional constraints on the essential matrix are:

- the determinant of the essential matrix is zero as it is rank-deficient.
- a 3x3 matrix is an essential matrix if and only if two of its singular values are equal, and the third is zero[10], as the matrix S is skew-symmetric and R is a rotation matrix.

Fundamental matrix

The essential matrix contains all the information about the geometry of two cameras relative to one another but it does so in *physical coordinates*. From the standpoint of stereo reconstruction, the relation of *image coordinates* which require the introduction of camera intrinsics is of interest. The relation from equation 2.21 is used, specifying that q is the point in image coordinates, p in physical coordinates, and K is the camera intrinsics matrix.

$$q = Kp \implies p = K^{-1}q \tag{2.40}$$

Inserting this into the essential matrix equation 2.39

$$q_r^\top (K_r^{-1})^\top E K_l^{-1} q_l = 0 \tag{2.41}$$

which yields the relation that the *fundamental matrix* F is

$$F = (K_r^{-1})^\top E K_l^{-1} \tag{2.42}$$

so that:

$$q_r^\top F q_l = 0 \tag{2.43}$$

The fundamental matrix expands upon the essential matrix, but while the essential matrix is relating the physical coordinates, the fundamental matrix relates the image coordinates of two cameras. Given a pair of images, to each point x in one image exists a corresponding epipolar line in the other image. The epipolar line is the projection of the ray from the point x through the camera center of the first camera, as seen by the other camera. This is a projective mapping from points to lines described by the fundamental matrix.

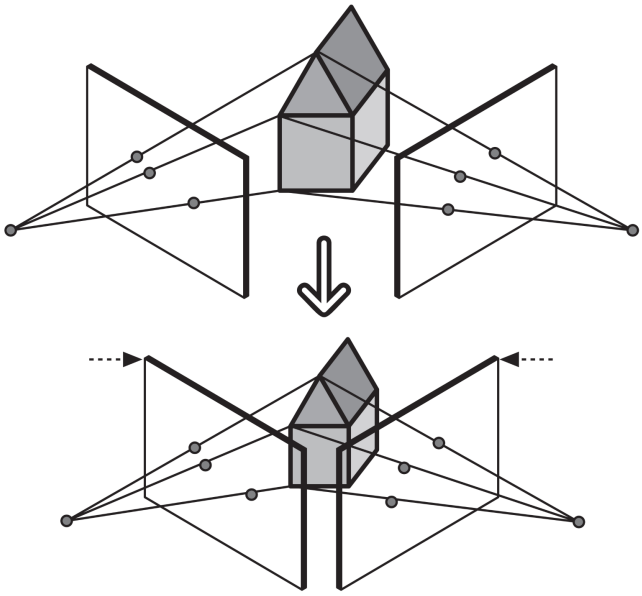
The fundamental matrix is a unique (3×3) rank 2 homogenous matrix which, for two images acquired by cameras with non-coincident centers, satisfies the equation $x'^\top F x = 0$ for all corresponding points $x \leftrightarrow x'$.

The fundamental matrix has the following properties:

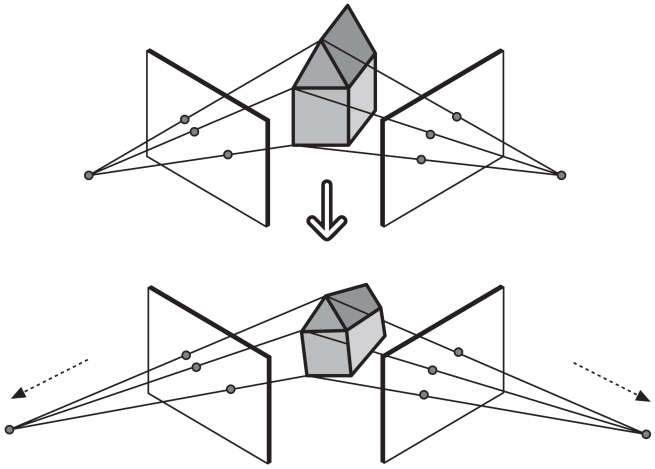
- **Transpose:** if F is the fundamental matrix of the pair of cameras (P, P'), then $F^\top$ is the fundamental matrix of the pair in the opposite order (P', P).
- **Epipolar lines:** For any point x in the first image, the epipolar line in the second image is $l' = Fx$. Likewise, $l = F^\top x'$ is the epipolar line in the first image corresponding to x' in the second.
- **The epipole:** for any point x other than the epipole e the epipolar line $l' = Fx$ contains the epipole e'. As e' satisfies $e'^\top (Fx) = (e'^\top F)x = 0$ for all x. It follows that $e'^\top F = 0$, or e' is the *left null-vector* of F. Likewise, $Fe = 0$, that is, e is the *right null-vector* of F.
- F has **seven degrees of freedom**. A (3×3) homogeneous matrix has eight independent ratios (there are nine elements and the common scaling is not significant); however, it is also valid that $\det F = 0$, which removes one degree of freedom.
- F is a **correlation** - a projective map mapping a point to a line. However, as a point in the first image defines a line in the second. If l and l' are corresponding epipolar lines, then any point x on l is mapped to the same line l'. This means that there is no inverse mapping, and F is not of full rank. For this reason, F is not a proper correlation (which would be invertible).

The importance of the fundamental matrix is that it can be computed for a stereo camera pair from a small number of known correspondences (7 correspondences are necessary at a minimum). Given a set of correspondences $x_i \leftrightarrow x'_i$, the fundamental matrix satisfies the condition $x'^\top_i F x_i = 0 \forall i$. With the x'_i and x_i defined, the equation is linear in the entries of F. With at least 8 correspondences it is possible to solve linearly for entries of F up to scale (and a non-linear solution is available for 7 point correspondences). Given more than 8 correspondences a solution can be determined using the least-squares method. This is the general principle of computing the fundamental matrix.

Once a fundamental matrix is computed, it is simple to compute the individual camera



(a) Objects of different size can appear the same depending on their distance from the camera, if object size is not known.



(b) Different projections can appear identical with different focal lengths and principal points, if camera intrinsics are not known.

Figure 2.13: Illustration of stereo reconstruction ambiguity, adapted from [11]

matrices (in their canonical form) from the fundamental matrix, up to a projective ambiguity as shown in the figure 2.13. With a fundamental matrix and the computed camera matrices, it is possible to reconstruct the 3D space via triangulation up to a projective ambiguity even without calibrated cameras. By computing the fundamental matrix and extrating the rotation matrix R and the camera matrices K_l and K_r, the stereo system is essentially *calibrated*.

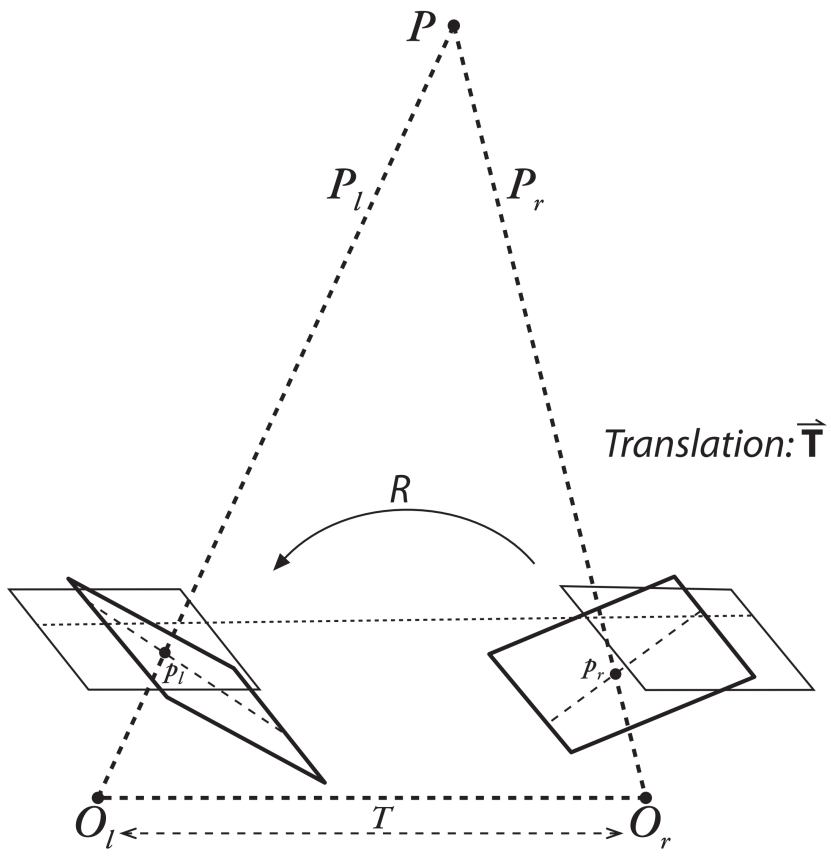


Figure 2.14: Adapting the two camera views into one viewing plane; adapted from [11]

2.3.2 Rectification

In an ideal stereo setup shown in the figure 2.9, the cameras are exactly aligned and all imperfections are abstracted away, so that the point correspondences can be computed optimally and the depth component triangulated. In the practical reality, many of the assumptions required for this model to be of use do not apply. Therefore additional steps must be applied which can bring a real-world camera into a geometry resembling the ideal arrangement as shown in figure 2.14.

Lens distortions

One part to consider in rectification is that the cameras are usually not ideal pinhole cameras, but use real lenses, which exhibit distortions that affect the projection of the images. Therefore, the rectification step must also include an *undistort* step in order to remove the lens distortions and bring the camera close to the pinhole model. The two common lens distortions which are addressed in the rectification procedure are *radial* (as shown in figure 2.15) and *tangential* distortion. The radial distortion arises from the fact that the lenses are usually, due to cost, constructed from elements with spherical surfaces (which are less costly to manufacture), which

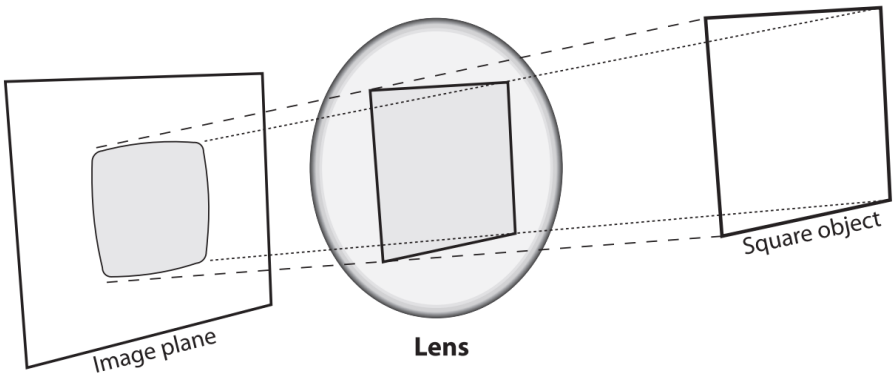


Figure 2.15: Radial distortion on a lens; adapted from [11]

introduces distortion the further the object is from the optical axis, as the optical characteristics change towards the edge of the lens. The distortion is usually 0 at the optical center, and increases towards the periphery. In practice it can be usually characterized by a Taylor series expansion around $r = 0$, where r is the radius around the optical center of the lens. As $f(r) = 0$ for $r = 0$ it can be assumed that for the general form $f(r) = a_0 + a_1 r + a_2 r^2, + a_3 r^3 + ...$, a_0 is zero, considering only the even powers as the distortion model is symmetric around the optical axis. With these constraints, the corrected image coordinates are defined as:

$$\begin{aligned} x_{corrected} &= x(1 + f_1 r^2, + f_2 r^4 + f_3 r^6) \\ y_{corrected} &= y(1 + f_1 r^2, + f_2 r^4 + f_3 r^6) \\ r &= \sqrt{(x - x_c)^2 + (y - y_c)^2} \end{aligned} \tag{2.44}$$

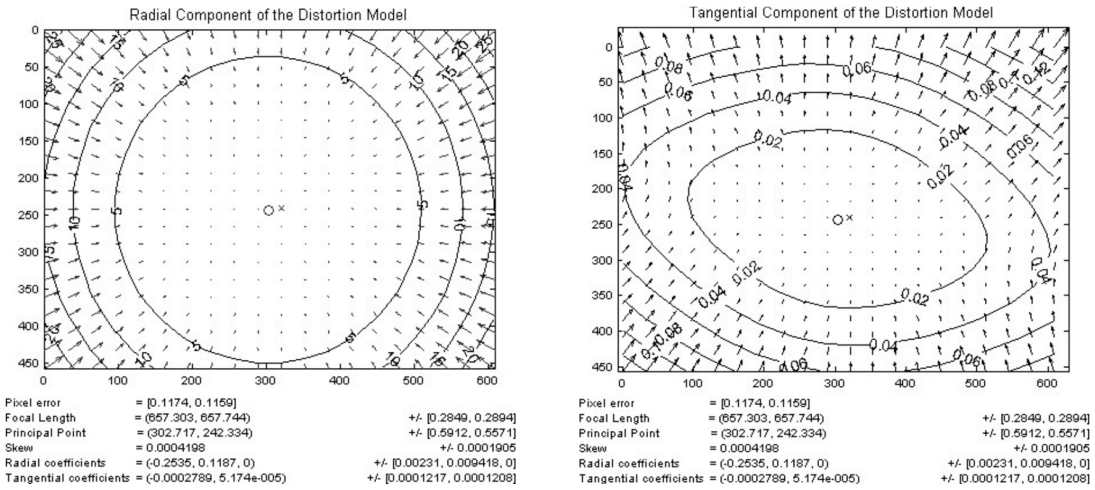
The tangential distortion is, on the other hand, a result of the assembly of the entire camera system, as it is a result of a lens not being exactly parallel to the imaging plane. The tangential distortion can be corrected through following equations[11]:

$$\begin{aligned} x_{corrected} &= x + (2p_1 y + p_2(r^2 + 2x^2) \\ y_{corrected} &= y + (p_1(r^2 + 2y^2) + 2p_2 x) \\ r &= \sqrt{(x - x_c)^2 + (y - y_c)^2} \end{aligned} \tag{2.45}$$

The distortion effects for an actual lens/camera are shown in figure 2.16

Stereo rectification

After correcting for the individual image distortions, the goal of *stereo rectification* is to mathematically align the two camera views into one viewing plane so that the pixel rows between the cameras are aligned with the epipolar lines and perfectly aligned with each other. This allows for the stereo correspondence algorithm to be able to search for correspondences constrained



(a) Radial distortion for an actual lens.

(b) Tangential distortion for an actual lens.

Figure 2.16: Example of actual lens distortion maps, adapted from [11]

only to one image row[11], which increases robustness and reduces the computational load. As the epipoles are the intersection point of all epipolar lines of their respective cameras this essentially means that the epipoles are mapped to infinity by the transformation which rectifies the images. This is the foundation of *Hartley's algorithm*[13],[10][11] which can yield uncalibrated stereo using only the fundamental matrix.

Hartley's algorithm attempts to find homographies that map the epipoles to infinity while minimizing the computed disparities between two images. This bypasses the need to compute the camera intrinsics for the cameras, as this information is implicitly contained in the point matches. Therefore, with a computed fundamental matrix, which can be computed using 8 or more points, the stereo system can be calibrated and rectified. While the advantage is the ability to work simply upon matching points in the scene, the disadvantage is the inability to determine scale. With a computed fundamental matrix F, the algorithm proceeds as follows:

1. using the fundamental matrix F, compute the epipoles via relations $e_r^\top F = 0$ and $Fe_l = 0$
2. seek a homography H_r which will map the first epipole to the 2D homogenous point at infinity $(1,0,0)^\top$.
3. Find the matching projective transformation H_l that minimizes the least-squares distance

$$\sum_i d(H_l \cdot x_i, H_r \cdot x_i') \tag{2.46}$$

4. Resample the left image according to the projective transformation H_l and the right image according to the projective transformation H_r.

Hartley's algorithm is an example of uncalibrated rectification which works up to a projective ambiguity as shown in figure 2.13. In order to obtain the actual distances, it is required to work with calibrated cameras. Bouguet's algorithm[11] is an example how to obtain cali-

brated rectification. Given the rotation and translation $(R, \mathbf{t})$ relating the two stereo images (and combined in the essential matrix E), Bouguet's algorithm attempts to minimize the amount of change which the reprojection introduces into the images during rectification. The algorithm achieves this by splitting the rotation matrix R into two matrices, one for each image (r_l and r_r, respectively). As each camera rotates only half of the total rotation amount, their principal rays are rotated parallel to the vector sum of their original direction. Such rotation makes a co-planar alignment, but does not align the pixel rows. In order to compute the matrix R_{rect} that will map the left camera's epipole to infinity and align the epipolar lines horizontally, the process starts with the unit-normalized direction of the epipole, which is directly along the translation vector $\mathbf{t}$, taking the principal point (c_x, c_y) as the origin:

$$\vec{e_1} = \frac{\mathbf{t}}{\|\mathbf{t}\|} \tag{2.47}$$

The second vector must be orthogonal to the first, but is otherwise unconstrained. A direction orthogonal to the principal ray (along the image plane) is a good choice. To obtain the orthogonal vector the normalized cross product between $\vec{e_1}$ and the principal ray is used.

$$\vec{e_2} = \frac{1}{\sqrt{t_x^2 + t_y^2}}(-t_y, t_x, 0)^\top \tag{2.48}$$

A third vector orthogonal to $\vec{e_1}$ and $\vec{e_2}$ is obtained using a cross product

$$\vec{e_3} = \vec{e_1} \times \vec{e_2} \tag{2.49}$$

Combining the three vectors in equations 2.47, 2.48 and 2.49 the matrix which maps the epipole to infinity is obtained:

$$R_{rect} = \begin{bmatrix} \vec{e_1}^\top \\ \vec{e_2}^\top \\ \vec{e_3}^\top \end{bmatrix} \tag{2.50}$$

The matrix R_{rect} rotates the left camera around the center of projection so that the epipolar lines become horizontal and the epipoles are at infinity. The row alignment of the two cameras is then achieved by setting the rotation matrices for left and right images to:

$$\begin{aligned} R_l &= R_{rect} \cdot r_l \\ R_r &= R_{rect} \cdot r_r \end{aligned} \tag{2.51}$$

Additionally, the rectified camera matrices are computed:

$$
\begin{aligned}
P_l = K_{rect,l} \cdot P_l' &= \begin{bmatrix} f_{rect,l} & \alpha_l & c_{x,l} \\ 0 & f_{y,l} & c_{y,l} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \\
P_r = K_{rect,r} \cdot P_r' &= \begin{bmatrix} f_{rect,r} & \alpha_r & c_{x,r} \\ 0 & f_{y,r} & c_{y,r} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}
\end{aligned}
\tag{2.52}
$$

The projection matrices project a 3D point in homogeneous coordinates to a 2D point in homogeneous coordinates as follows:

$$
P \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ w \end{bmatrix} \tag{2.53}
$$

where the screen coordinates can be calculated as $(x_s, y_s) = (x/w, y/w)$.

Points in two dimensions can also be reprojected into 3D given their screen coordinates and the camera intrinsics matrix. The reprojection matrix is:

$$
Q = \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 0 & f \\ 0 & 0 & -\frac{1}{t_x} & \frac{c_x - c_x'}{t_x} \end{bmatrix} \tag{2.54}
$$

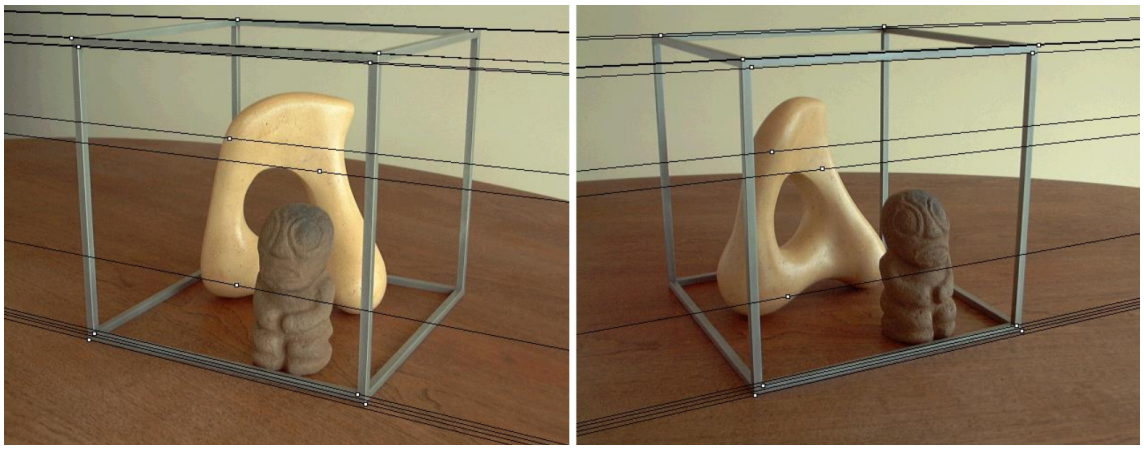
The parameters are here all from the left image, except c_x' which is the principal point x-coordinate in the right image. If the cameras are coplanar and the principal rays intersect at infinity, the term in the fourth column and fourth row is zero.

Finally, with the rectified images, the 3D space can be reconstructed with known disparities. With a two-dimensional homogeneous point and its associated disparity d, it is possible to

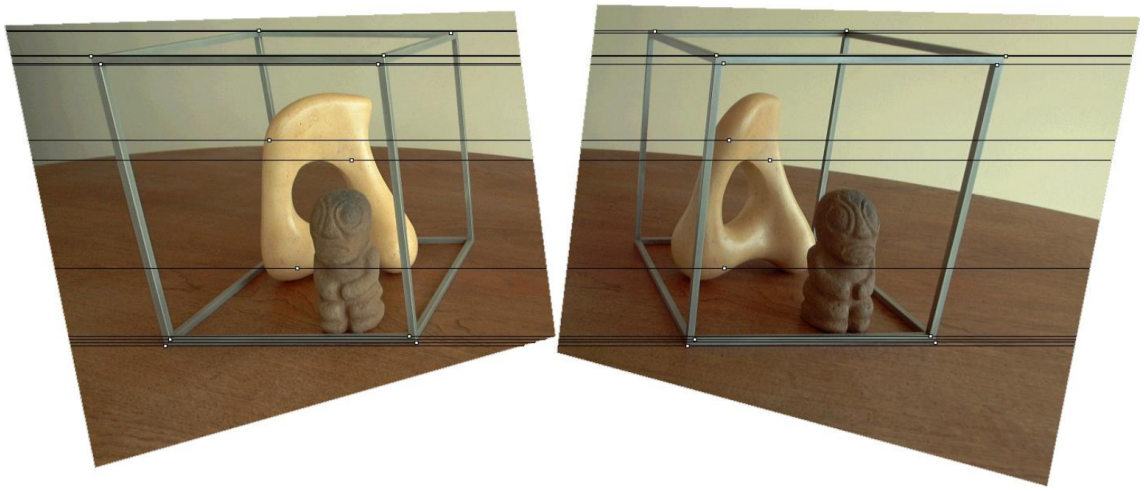
project the point into three dimensions using:

$$Q \cdot \begin{bmatrix} x \\ y \\ d \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} \tag{2.55}$$

Many other techniques have been proposed for camera calibration and rectification, such as [14] and [15]. An example of a stereo pair before and after rectification can be seen in Figure 2.17.



(a) Original stereo image pair



(b) Rectified stereo image pair

Figure 2.17: Illustration of stereo epipolar rectification, adapted from [14]

2.4 Closure

In this chapter an overview of the overall geometric problem of 3D reconstruction is presented. Only the essential terms and topics have been covered. More information can be found in relevant articles and textbooks [13], [10], [2], [11]. Finally, it was shown that stereo image pairs which have undergone epipolar rectification represent the best foundation for disparity computation given that they constrain the search problem to individual image rows. In this thesis, the assumption for all algorithms will be that the input images are properly rectified using the principles outlined in this chapter.

Chapter 3

Three-dimensional recursive search

3.1 Motion estimation

The problem of finding visual correspondences occurs frequently in digital image and video processing, applied in areas ranging from consumer electronics to machine vision. For two images of the same resolution, the goal is to establish a map of areas within a source image which are similar to areas within a reference image, and assign a vector map which points from the position of each defined area within the source image to the position of the corresponding area within the reference image. When this mapping technique is applied to a sequence of images, the resulting pattern of vectors is called the *optical flow*[4], and is used extensively in image processing, from visual effects to robotic vision. The optical flow provides information about the motion of pixels (and objects) within a visual scene as seen in Figure 3.1.

Estimating (and compensating for) the motion in a sequence of images is a technique commonly applied within video codecs such as MPEG-2 or MPEG-4.[5] Apart from estimating motion, the optical flow techniques can be applied to stereo images to provide a *disparity map* or a *depth map*, where the vectors estimate the distances (disparities) of the corresponding blocks in two stereo images, which are directly proportional to the depth of objects within a

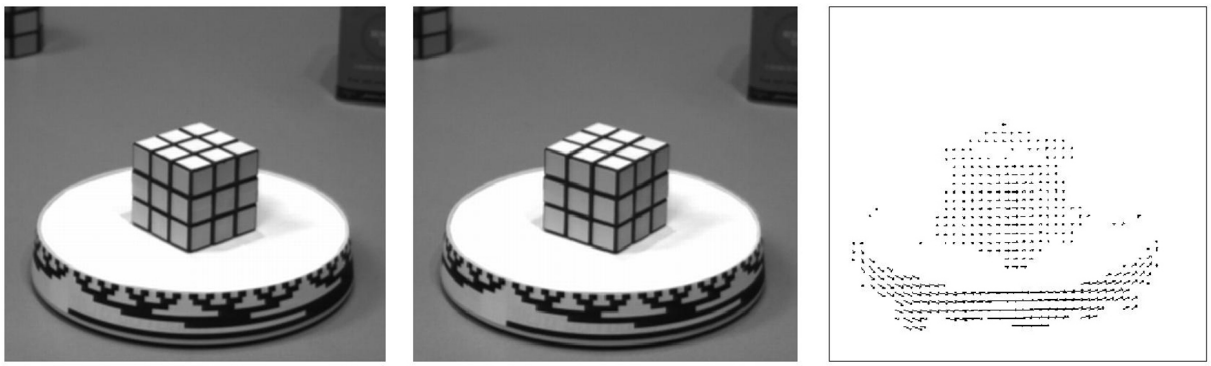


Figure 3.1: The optical flow within a sequence of images. From left to right: first image in a sequence, second image in a sequence, computed motion vectors. Adapted from [4]

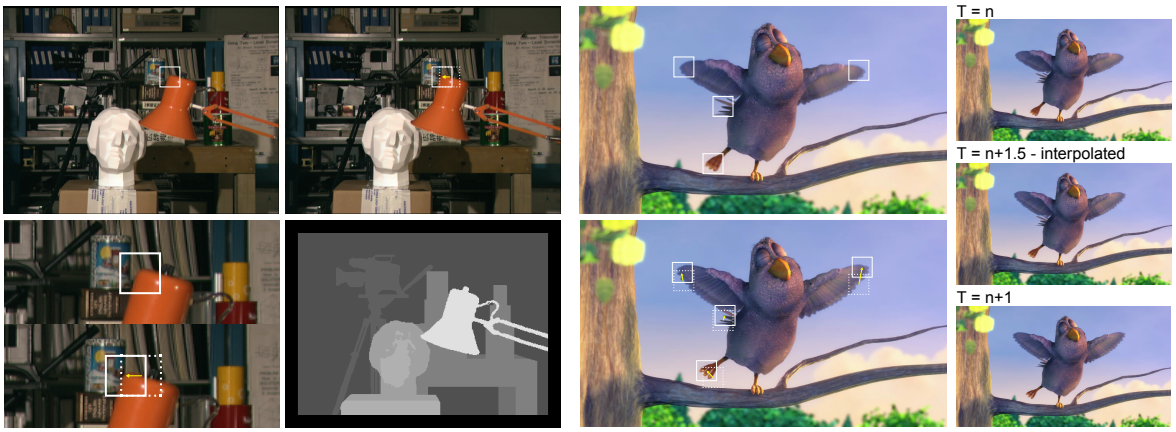


Figure 3.2: Various applications of optical flow; Left: disparity estimation; Stereo images, detail and disparity map. Right: motion estimation and motion interpolation.

three-dimensional scene.

While many algorithms and techniques for the computation of the optical flow exist, a considerable amount of them are neither suitable nor easily implementable within an embedded computer or a real-time system. One of the more demanding embedded applications is the frame rate conversion in consumer televisions, where the high definition video received at 25 or 50 frames per second (fps) is up-converted to 100, 200, or 400 fps in modern televisions. To achieve this, the image processor in the television must compute the optical flow of a high-definition image at up to 60 frames per second, providing a motion flow map for each pair of frames, which then allows the interpolation of missing frames, yielding the required frame rate, as illustrated in 3.2. The constraints of consumer electronics demand that the algorithm is as simple as possible in order for it to operate within the cost and power constraints.

the 3-Dimensional Recursive Search algorithm (3DRS)[6] is one of the first affordable and widely deployed motion estimators in consumer electronics devices. The 3DRS algorithm was first developed by Gerard De Haan and Henk Huijgen at the Philips Research Laboratories in 1989 and has subsequently gained wide adoption in motion estimation for frame rate conversion due to its simplicity, speed, and the ability to estimate true motion within the image sequence.

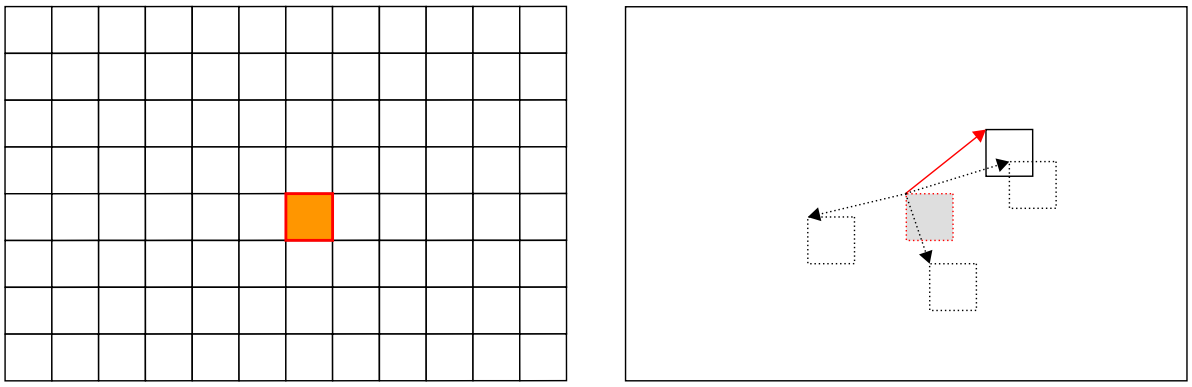


Figure 3.3: An illustration of block matching. Left: Source image divided in blocks. Right: reference image with multiple matching candidates.

3.2 Description

3.2.1 Overview

The 3DRS algorithm is a *block-matching algorithm*; the input source image is divided in blocks, for each of which the location of a best-matching block is searched in the reference image. In case of motion (or disparity), the best-matching block will be located at a different spatial position in the reference image than in the source image. To each of the blocks, a two-dimensional vector defining the distance between the original block position and the matching block position is assigned, as shown in Figure 3.3. This vector is called the *motion vector*. The map of vectors for all of the image blocks is the optical (or motion) flow of the image.

The blocks are matched through a *matching cost function*, which provides the measure of similarity for the matched blocks. The most common matching functions used are the *sum of absolute differences* (SAD) or the *sum of squared differences* (SSD), as they are simple and easily implementable. Other matching functions, such as the *normalized cross-correlation*(NCC) are also used for some applications, although it is significantly more complex than the aforementioned two. Finding the proper matching block involves finding the minimum value of the matching cost function. To find the minimum, an optimization method is usually applied, however the optimization can be a computationally intensive process, with many steps and solution candidates to be evaluated, which makes the implementation prohibitive for the limited resources of an embedded system. The 3DRS algorithm reduces the number of required candidate blocks (or candidate vectors) by making the following two assumptions:

1. Objects are larger than blocks.
2. Objects have inertia.

As shown in Figure 3.4, the objects in question are the features in the actual image, the movement (or disparity) of which is being estimated. By stating that the objects are "larger than blocks", it is being assumed that the vectors of the neighboring blocks provide a good

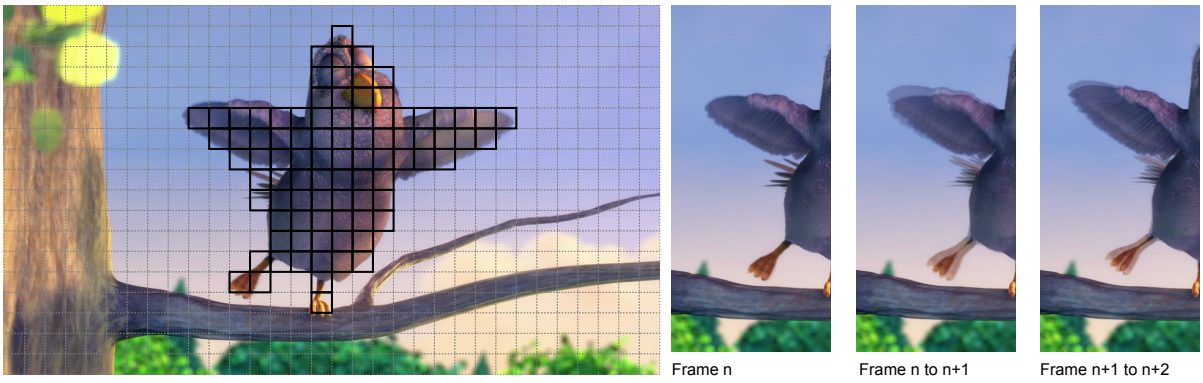


Figure 3.4: An illustration of 3DRS assumptions. On the left, an object is mapped to multiple blocks in the image. On the right, the progress of motion over multiple frames is shown, illustrating the concept of inertia in the image.

predictor, or assumption, about the block being estimated. By stating that objects "have inertia", an assumption is made that, in a running sequence of images, a *temporal predecessor*, meaning a previously estimated motion vector, is also a good candidate for the new vector.[16] By applying these assumptions to the estimation process, it is possible to significantly reduce the number of candidates evaluated to find the best-matching block vector, greatly speeding up the algorithm execution. However, by using only the available vectors, the algorithm would not yield good results, as it is required to evaluate alternate solutions, and account for changes which appear in the image sequence, such as appearing objects or acceleration. To account for this, *update vectors* are added to a subset of selected predictors. The update vector modifies the existing predictor value in order to try out a different solution. The update vectors can be generated based on a random distribution, but usually they are selected from a predefined set of update vectors. An example of the update vector set is:

$$U_v = \left\{ \begin{bmatrix} \pm 2^k \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ \pm 2^k \end{bmatrix} \right\} \; k = 0, 1, 2, 3, ..., k_{max} \tag{3.1}$$

The variation introduced by the update vectors ensures that the algorithm will progress towards a solution regardless of the initial conditions, especially in the typical initial case where the vector field contains only zero vectors.

For each block, 3DRS selects the output motion vector from from a candidate vector set that is based on prediction vectors from a spatio-temporal neighborhood, with the addition of updates in order to evaluate alternate solutions, as seen in Figure 3.5. For a block at a given location, the matching cost function is evaluated for each predictor, and the predictor with the best matching cost is selected (in a *winner-take-all* manner) and assigned as the new block vector at the given location. The important takeaway here is that this procedure is performed

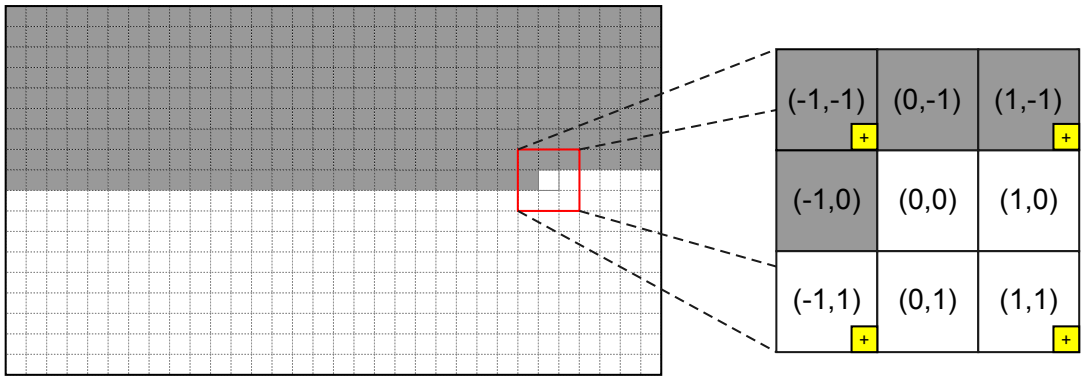


Figure 3.5: The spatio-temporal neighborhood used in the 3DRS estimation. The vector for the block at (0,0) is currently being estimated. Vectors for the blocks at (-1,-1), (0,-1), (1,-1) and (-1,0) have already been estimated for this image. The remainder of the blocks used are from a previous estimation, making them temporal predictors. The predictors coming from the blocks marked with a '+' are additionally modified with update vectors.

sequentially for every block and location in the image. The newly estimated value, which is assigned to the current block location, also becomes a part of the candidate set for the next location, directly influencing the estimate for the next location.

The order in which the block locations are evaluated is defined by the *scanning direction*. The most common order is from top-left to bottom-right, with each line beginning on the left. This order defines which of the predictors are spatial, and which have also a temporal component, as seen in Figure 3.5. It also defines the direction in which the newly estimated, "good" values propagate within the image. If a good estimate is found at the bottom of the image, it will require multiple iterations before it propagates to the top. To improve the propagation of estimates evenly throughout the image, it is beneficial to vary the scanning direction, as shown in Figure 3.6 One example is to change the scanning order from bottom-right to top-left on each new image, another one is to alternate the scanning direction (from left-to-right to right-to left and vice versa) for each line, which is called *boustrophedonic scanning*, or, more simply, *meandering*. Combining both of the approaches yields propagation of estimates in all four directions, helping to achieve the quick spread of good estimates throughout the image. For the correct estimates to propagate even faster, multiple scans can be done for one image, alternating the direction and the meander of the scan on each pass.

3.2.2 Elaboration

The 3DRS algorithm works by analyzing the blocks in two images:

- The source image I_S
- The reference image I_R

as shown in Fig.3.7.

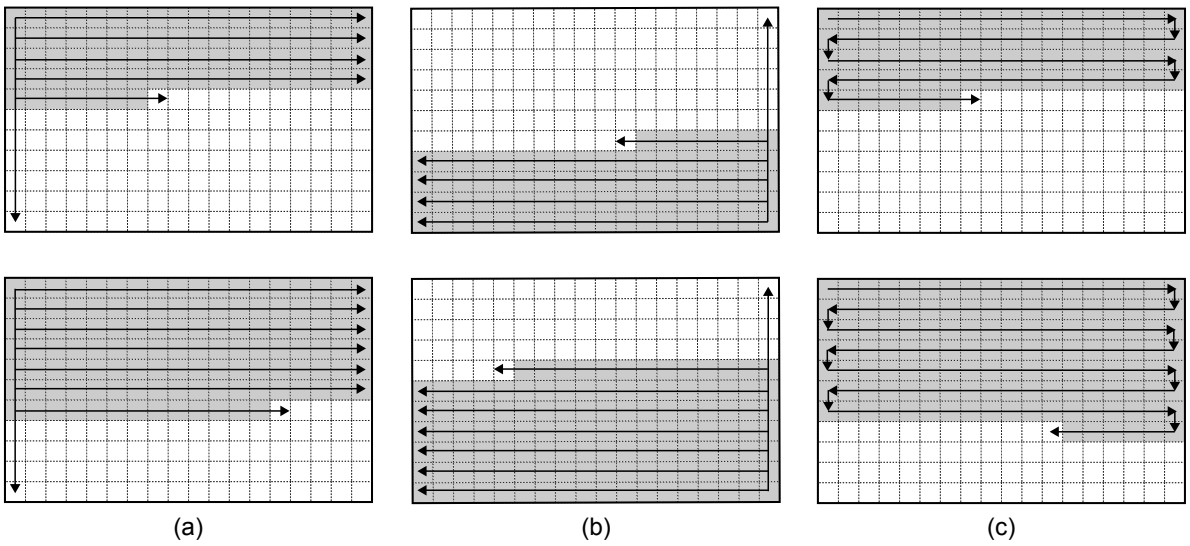


Figure 3.6: Various scanning orders for 3DRS, shown in time progression from the upper to the lower image; (a): Top to bottom, left to right; (b): Bottom to top, Right to left; (c): Boustrophedonic/meandering. The darker areas illustrate the progression of current estimates throughout the image.

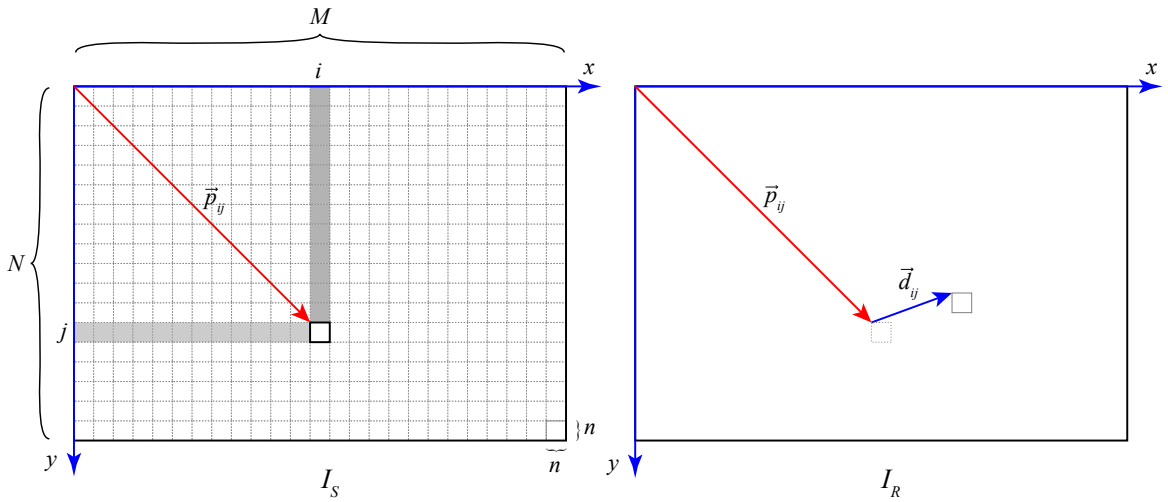


Figure 3.7: 3DRS image space and coordinate system

A cartesian coordinate system is defined on each image with the origin in the upper left corner and the coordinates $(0,0)$ pointing to the upper left pixel in the image, as shown in Figure 3.7 The x is the horizontal axis and the y is the vertical axis, with values increasing towards the right and bottom, respectively.

The source image I_S is divided into a rectangular grid of $M \times N$ blocks of $n \times n$ pixels as shown in the figure. For each block B_{ij}, i denotes the row index and j denotes the column index of the block. A motion vector $\vec{d}_{ij}$ is assigned to each block.

The pixel position vector $\vec{p}_{ij}$ for each block in I_S is defined as

$$\vec{p}_{ij} = n \cdot \begin{bmatrix} i \\ \\ j \end{bmatrix} \tag{3.2}$$

This vector points from the origin to the the upper-left pixel of each block.

The 3DRS algorithm consists of the following steps:

1. For each block B_{ij} in I_S, form a set of predictors:

$$P = \begin{cases} \vec{d}_{ij} \\ \vec{d}_{i+k\,j} \\ \vec{d}_{i\,j+l} \\ \vec{d}_{i+k\,j+l} + \vec{u}_v \end{cases} \qquad k = -1,1;\ l = -1,1 \tag{3.3}$$

   In eq. 3.3, $\vec{u}_v$ is the update vector, and a new one is selected either randomly of from a predefined set, for each of the predictors it is applied to.

2. Evaluate the matching cost for each of the predictors, and assign to $\vec{d}_{ij}$ the predictor with the minimum matching cost.

$$\vec{d}_{ij} = arg \min_{\vec{d}_P \in P} (MC(\vec{p}_{ij}, \vec{p}_{ij} + \vec{d}_P)) \tag{3.4}$$

   where $MC(\vec{s},\vec{r})$ is the aggregated matching cost for the compared blocks.

$$MC(\vec{s},\vec{r}) = \sum_{y=0}^{n-1} \sum_{x=0}^{n-1} C(I_S(\vec{s} + \begin{bmatrix} x \\ \\ y \end{bmatrix}), I_R(\vec{r} + \begin{bmatrix} x \\ \\ y \end{bmatrix})) \tag{3.5}$$

3. Repeat the previous two steps for every block in the image based on the selected scanning direction.

3.2.3 Enhancements

The basic algorithm provides solid performance in the real-time generation of motion maps. However, numerous improvements have been made to the original algorithm to enhance its performance. An important characteristic is the quality of the output, where the goal is to obtain the highest possible quality of the motion vectors in the least amount of algorithm passes (ideally, only a single pass). The quality of the output is defined by how well the obtained motion vectors match the true motion within the scene. As the motion vectors are obtained from a finite predictor set, it can be argued that not every predictor within the set has the same quality. The spatial predictors obtained in the same round are generally considered reliable, as they are the result of the most recent evaluation. However, the temporal predecessors have been calculated in the previous frame and have a lesser chance of corresponding to the actual situation, while they still may be good predictors. The updated predictors could be considered even less reliable, as they are a result of a predictor combined with a nonpredictable update, which might lead to a better solution, or, instead, push the algorithm in a suboptimal solution. It has been shown that these differences, although subtle, do affect the final output of the algorithm.

To account for these differences, a concept of *penalties* is introduced into the 3DRS algorithm. A penalty is a value which is added to the calculated cost *MC* for a given predictor $\vec{d}_{ij}$. The value is shared for all predictors of a certain reliability - for example, all spatial non-updated predictors calculated in the current round are not penalized; all temporal predecessors are penalized with a shared value which can be called *temporal penalty* - and all updated predictors (temporal or not) are additionally penalized with an additional, higher value called the *update penalty*. The matching process now proceeds to find a minimum value of the matching cost, additionaly weighed with the penalty value w_{ij} assigned to the predictor.

$$\vec{d}_{ij} = arg \min_{\vec{d}_P \in P} (MC(\vec{p}_{ij}, \vec{p}_{ij} + \vec{d}_P) + w_{\vec{d}_P}) \tag{3.6}$$

Where the penalty values w_{ij} are defined as follows:

$$w_{\vec{d}_P} = w_{\vec{d}_{i+k,j+l}} = \begin{cases} 0, \, (k,l) = (-1,0),(0,-1) \\ w_t, \, (k,l) = (0,1),(1,0),(0,0) \\ w_u, \, (k,l) = (-1,-1),(-1,1) \\ w_t + w_u, \, (k,l) = (1,-1),(1,1) \end{cases} \tag{3.7}$$

It is to be noted that this description is valid only in case of top-to-bottom, left to right scanning direction. Other scanning directions require different assignment of temporal penalties depending on which values are newly calculated and which are the temporal predecessors.

The penalty values are usually selected empirically depending on the implementation details

and assessed quality of the motion vectors obtained. In one reference implementation, for 16-bit grayscale image values, it has been determined that a penalty of w_t=128 per pixel in a block and w_u=512 per pixel in a block (the values added to the matching cost are further multiplied by the number of pixels in the matched block) produces good results.

Another enhancement to 3DRS addresses the problem of *convergence*, which is the process in which the 3DRS vectors change from the current state to obtain the next good state. Depending on the size and members of the update set, it may take several blocks, iterations, or even frames before a vector converges from the current value to the one corresponding to the actual value, if it reaches the proper value at all. A greater motion vector may take several iterations to converge to zero, and vice versa. To enhance convergence speed, two enhancements, which involve the addition of new predictors, have been proposed[5]. To accelerate convergence towards a greater value, a predictor is added which results from a parametric model of the global motion. Good results are, however, obtained only when the global motion is an affine transform, such as zoom or rotation. Another, and more useful, enhancement for convergence is the inclusion of the *zero vector* $\vec{0}$ in the predictor set for all blocks. The zero vector provides two advantages: first, stationary areas appear in video sequences and the zero vector guarantees that the vector estimated for stationary areas will be exactly zero; second, having a zero vector ensures that the 3DRS algorithm does not need multiple iterations to converge from a much larger vector to zero if necessary. The convergence speed is generally affected by the set of update vectors. Larger vectors may bring faster updates, but generally introduce substantial noise in the vector field, which is not desirable in the common applications of 3DRS.

3.2.4 Properties

The properties discussed so far have introduced the 3DRS algorithm as an efficient way to calculate motion vectors in image sequences. 3DRS is not the only algorithm available with these capabilities, although its simplicity and efficiency have resulted in a wide employment of 3DRS in practical systems. However, an additional quality which provides a distinction from the majority of motion estimators is that 3DRS provides *true motion* estimation. A wide selection of motion estimation algorithms initially developed for motion-compensated coding, such as the *Full search block matching* algorithm, aims to obtain *minimal residue*, that is, minimize the residual signal obtained by subtracting the two frames using the computed displacement vectors. This is, however, unsuitable for purposes such as motion interpolation and disparity estimation, where the accurate displacement of *objects* is desired. 3DRS, with its underlying assumptions, provides the true motion estimation within a scene, which makes it highly usable for the aforementioned applications such as motion interpolation or disparity estimation.

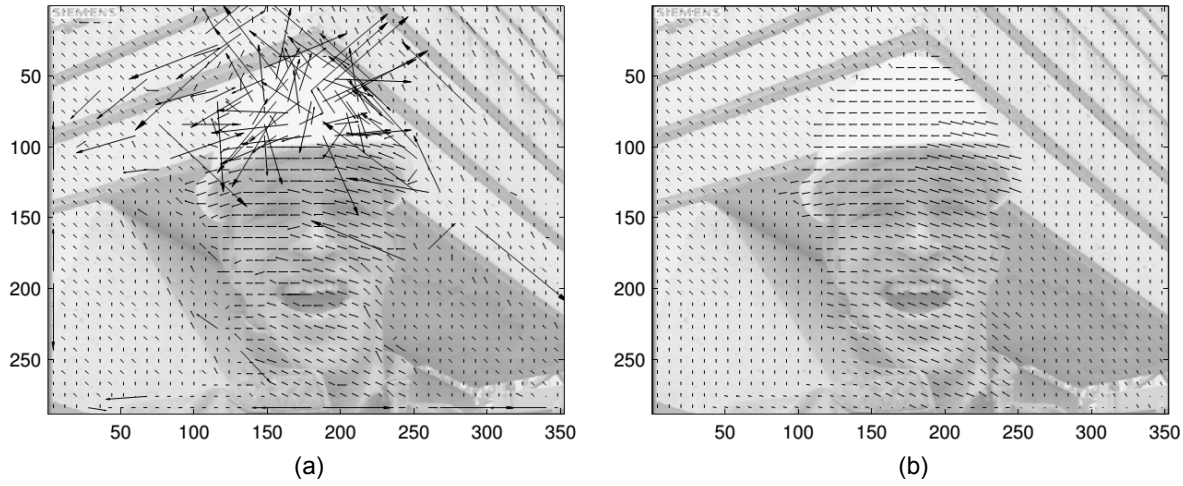


Figure 3.8: True motion estimation. The image (a) shows the frame 25 of the *foreman* sequence with motion estimated using the full block search. Image (b) shows the result of the 3DRS estimation for the same frame. Adapted from [5]

3.3 Implementation

Due to significantly reduced computational complexity in comparison to other motion estimation algorithms, such as Phase Plane Correlation [17], the 3DRS algorithm has been widely deployed in the areas of motion estimation and motion compensated video processing (de-interlacing and frame rate conversion).

3.3.1 Hardware

For the most typical applications, such as the motion estimation required to guide the frame interpolation in high definition televisions, the 3DRS algorithm is usually implemented in hardware, usually as a component of a larger TV system implemented in a VLSI integrated circuit, or ASIC[7]. 3DRS is particularly suited for embedded hardware implementations due to the simple logic and a reasonably small number of candidates to evaluate per block, which can be evaluated in constant time per block, yielding good predictability of the execution time. The most computationally challenging part of the algorithm is the computation of the Sum of Absolute Differences (SAD), which can be efficiently parallelized on the block level, making it especially suitable for hardware implementations. However, the usual problem in the VLSI implementations is the size and cost of the embedded memory required to store the source and the reference images. For this reason an external memory device such as an SDRAM chip (or multiple chips) is used to buffer the images. Alternatively, an on-chip *compressed* memory may be used. Both approaches, however, add substantial overhead and delays to random accesses to the image memory. SDRAM chips have a large overhead per access, which is usually compensated by transferring data in long bursts to reduce the percentage of overhead time vs. the active data

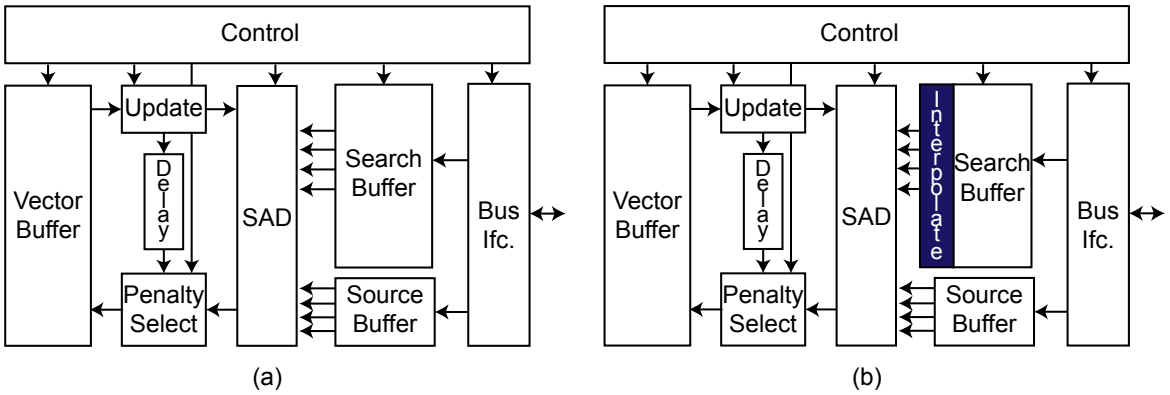


Figure 3.9: (a)Typical 3DRS Hardware architecture. A persistent vector memory is read out for predictors which are selectively updated, analyzed in the SAD, and the cost returned is penalized and matched to the vector, selected if best, and in the end written back to the vector memory. (b) Extension of architecture required for sub-pixel precise estimation.

time. Compressed memories can add an unpredictable delay in accessing the memory, depending on the compression algorithm. As the locations of the pixels used in the SAD calculation are defined by the vector being tested, random delays in accessing the individual pixel blocks would severely affect the algorithm's performance. The solution to this problem is to store a smaller memory buffer on-chip, which allows a constant-time, parallel access for multiple pixels in a block, providing good performance of the evaluation step. This, however, comes at an expense of reducing the search range within the confines obtainable by the available on-chip buffer size.

A typical hardware implementation architecture is shown in Figure 3.9. An additional advantage of the hardware implementation is that it can be easily adapted, at little hardware and almost no time cost, to work with sub-pixel precision and generate smooth motion maps, which would significantly degrade the performance of a software solution.

3.3.2 Software

The software implementation of 3DRS is straightforward, as the vector selection and update steps have constant complexity. The most challenging part in the implementation is the computation of the SAD cost function, as for a given block size N it involves N^2 subtractions and absolute calculations, repeated for every predictor at every block of the image. The complexity thus obtained is $O((m \times n)^2)$, for the image of $m \times n$ pixels.

A brute force solution to this problem which is available in the modern processors is to use the *SIMD-style operations* provided by the *multimedia instruction sets* which have been introduced to CPU architectures to shorten the computation time of multimedia algorithms [18] such as motion estimation. As the SAD is a common operation in video coding and motion

estimation, it is even directly implemented on some architectures. For example, on all of the current x86 or x64 processors, the MMX instruction set contains the PSADBW instruction, or *packed sum of absolute differences*, which computes the sum of absolute differences for 8 bytes in a single cycle. Multimedia instruction sets have also gained implementations in the embedded space with the newer ARM architectures and the *Neon* multimedia instruction set, which enables SIMD-style computation on embedded CPU devices. With modern multi-core architectures, another approach to accelerating the 3DRS execution is to parallelize the algorithm to execute it on multiple CPU cores in order to decrease the execution time. One of the primary challenges in such approaches is maintaining the estimation and convergence quality through the implementation of the meandering scan [19]

Chapter 4

Related work

4.1 Stereo correspondence

4.1.1 Classic methods

A popular taxonomy of methods for computing stereo correspondences was given in [12]. The depth information is obtained by computing disparity, which is the distance between the corresponding features in the stereo image pairs being evaluated. The taxonomy defines the 4 major components of stereo matching methods:

1. the computation of the matching cost
2. matching cost aggregation
3. disparity computation
4. disparity refinement

In terms of the disparity computation, the methods can be broadly categorized as local, which compute an optimum matching cost value for each pixel within a local region (also known as a "Winner-take-all" approach), and global, which make smoothness assumptions about the scene and then attempt to minimize a global energy function based on it. While it has been shown that the global methods based on Markov Random Fields, such as Graph cuts [20] or Belief Propagation [21] yield the best quality of the results[22], they are also typically very slow and computationally intensive [23]. The local methods, while faster and much more suitable for a compact or an embedded implementation, are still inferior to global methods in terms of result accuracy. Furthermore, even if they are less computationally intensive than the global methods, local methods are still sufficiently challenging for implementation in embedded systems in cases where real-time performance is required. A separate class of global optimization methods based on Dynamic programming [24], especially the hybrid methods based on Semi-global matching [25], which have been found to provide a compromise between the output quality and computation complexity, have been much researched and widely deployed.

4.1.2 Deep Learning

Recently, approaches employing convolutional neural networks (CNNs) and deep learning have been the focus of stereo matching research. These approaches have been shown to significantly improve the results of stereo estimation algorithms [26]. The approaches in this area can be roughly divided into several categories. In *matching cost learning*, where the CNN is employed to learn the matching cost instead of using hand-crafted methods.[27]. In these approaches, the disparity computation step is still based on classical methods, such as Semi-global matching. *Regularity learning* applies the piece-wise smoothness assumption as a constraint to the learning process[28] or employing semantic segmentation as a guide for stereo correspondence [29]. *End to end disparity learning* approaches directly infer the disparity map from stereo pairs [30], or compute the dense optical flow [31] [32]. Even with the advent and the superior results of these methods, several classes of new problems have emerged. The difficulty of finding the correct correspondence at inherently ill-posed regions such as occlusions, repeated patterns or texture-less regions remains a challenge [33]. For end-to-end networks, it is difficult to generalize a pre-trained deep stereo model in a new, different domain [34], as significant performance degradation is observed even in state-of-the-art methods such as [35]. In most cases, training is performed with synthetic datasets [30], and a re-training using samples from the target domain with ground truths is used to fine-tune the network. The impracticality of obtaining and maintaining such samples, as well as maintaining a substantially large training data set, remains a challenge in real-world applications of deep learning based stereo reconstruction.

4.2 Winner-Take-All optimization

4.2.1 Description

In local approaches, correlation between intensity values inside a matching window $N(p)$ assigned to a reference pixel p is analyzed, based on the assumption that the pixels within the matching window have similar disparities. The matching cost is aggregated by summing or averaging over a support region in the DSI $C(x,y,d)$ as shown in Figure 4.1, where a support region can either be two-dimensional in $x-y$ space at a fixed disparity, favouring fronto-parallel surfaces, or three-dimensional in $x-y-d$ space, supporting also slanted surfaces[2]. The generalized procedure of the WTA optimization[36] of a left disparity map in the case where *absolute difference* is used is as follows. For each of the disparity hypotheses, a per pixel cost $e(p,d)$ is calculated by using the image pair where the right image is shifted by the disparity d. An aggregated cost $E(p,d)$ is then computed via a summation of the per-pixel cost multiplied by a weighting function $w(p,q)$. The WTA optimization is finally performed for seeking the best

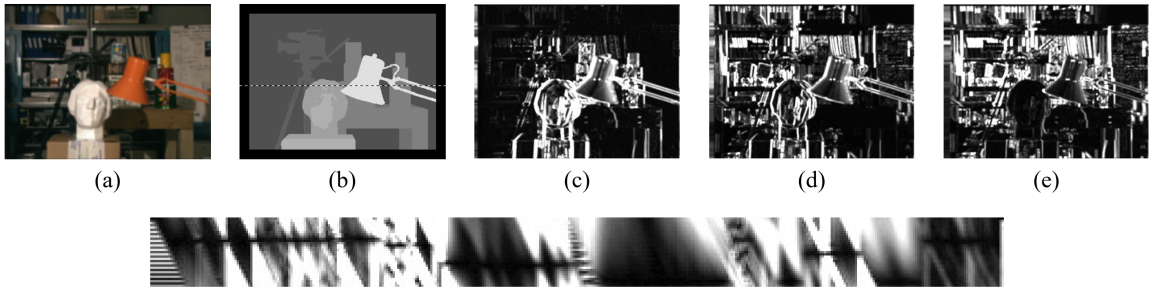


Figure 4.1: Example of a disparity space image. (a) Left image of the stereo pair "Tsukuba". (b) The disparity ground truth. (c)-(e) $x-y$ slices of the DSI for the stereo pair "Tsukuba" at disparities $d = 10, 16, 21$, respectively. (f) An $x-d$ slice for $y = 151$ (the dashed line in (b)). Matching regions are shown with a darker tone (intensity corresponding to matching cost). Adapted from [2]
.

disparity among all of the hypotheses:

$$e(p,d) = min(|I_l(x,y) - I_r(x-d,y)|, \sigma)$$

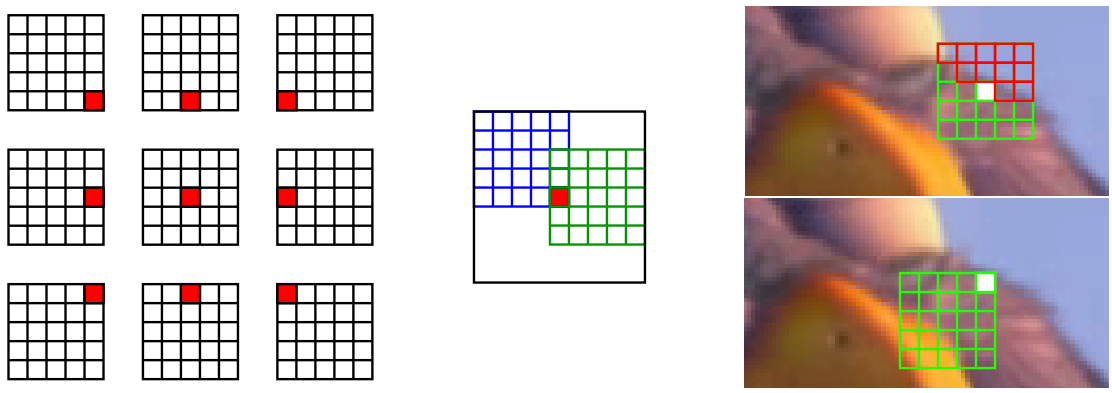
$$E(p,d) = \frac{\sum_{q \in N(p)} w(p,q)e(q,d)}{\sum_{q \in N(p)} w(p,q)} \tag{4.1}$$

$$d(p) = arg \min_{d \in [0,\ldots,D-1]} E(p,d)$$

where I_l and I_r are the left and the right images of the stereo pair. The per-pixel cost is additionally truncated with a threshold σ to limit the influence of outliers on the dissimilarity measure.

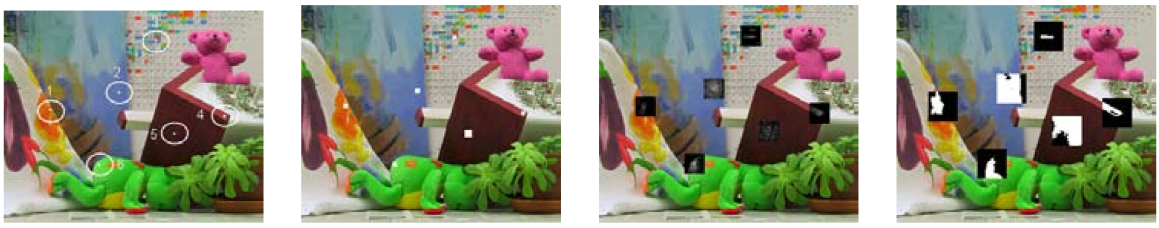
4.2.2 Related work

Many research efforts have been directed at improving the WTA algorithm in terms of speed and accuracy. Diverse matching cost functions which bring accuracy improvements at varying computational loads have been explored. However, the cost aggregation step remains the most important and the most time-consuming part of the calculation in the local approaches. Therefore, most of the efforts focus on defining the weighting function $w(p,q)$ which defines how the costs are aggregated within the window. The basic weighting function is a *square window* [37] or *box filter* where all of the neighbouring pixels are equally weighted. However, other approaches to aggregation have been devised to improve the matching accuracy, with the goal to match actual image features rather than raw pixels. *Shiftable windows*[38] [12] as shown in



(a) (b) (c)

Figure 4.2: Shiftable window. (a) Multiple possible configurations for a 5x5 aggregation window, with the actual evaluated pixel highlighted. (b) The area in which the windows are evaluated, centered at the pixel being evaluated. (c) The effect of shiftable windows: in the top image, a centered window for a pixel located at the border includes multiple pixels across the object border (a discontinuity in disparity) which may lead to a bad match. In the bottom image, a window shifted inwards from the object's edge matches only the pixels belonging to the same object as the evaluated pixel, which yields a better match.



(a) (b) (c) (d)

Figure 4.3: Cost aggregation examples, with weight for included pixels depicted as a gray scale between 0 (black) and 1 (white). (a) Selected matching points. (b)Variable windows. (c) Adaptive weights. (d) Segmentation based weights. Adapted from [2] with samples from [37].

Figure 4.2 aim to reduce the disparity noise at occlusion or border regions by evaluating multiple windows anchored at different points and selecting the one with the minimum cost for the evaluation, attempting to avoid discontinuity regions.

Similarly, windows with adaptive sizes [39] aim to address the fact that large windows are more efficient in low texture areas but tend to blur disparity images, whereas small windows yield more accurate disparity boundaries but generate noise in low texture areas. For each match, a separate window size is applied for optimum effect. Other research approaches have tried to implement an even more specific window configuration, such as a photometric similarity-based weighting function similar to a bilateral filter [40] or an image color segmentation based weighting function [41], which are used to obtain the best possible alignment of the support region and the object in the image while retaining sufficient information to resolve a match. Such support regions are shown in Figure 4.3.

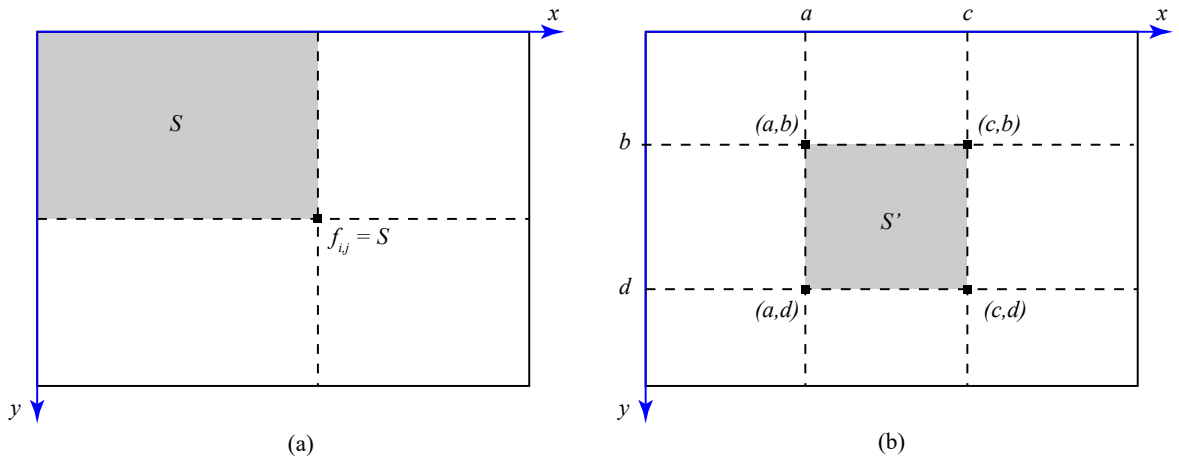


Figure 4.4: Integral images. (a) In an integral image, each element contains the sum of all of the elements of the original image which is contained in the rectangle S. (b) Any area sum can be calculated in constant time using only the four corner points of the rectangle.

For the classic square window and adaptive windows, a noteworthy enhancement is the use of integral images[39], which greatly accelerates the algorithm's execution in comparison with the direct aggregation. An *integral image*, or summed area table, is an image in which each element contains a sum of all elements in the original image bound by the rectangle which that element closes with the image's origin, as shown in figure 4.4 a) and computed with the formula:

$$f_{i,j} = \sum_{k=0}^{i} \sum_{l=0}^{j} g_{k,l} \tag{4.2}$$

where $g_{k,l}$ are pixels of the original image. Another desirable property of the integral image is that it can be computed incrementally with a constant computation cost per each pixel, yielding the complexity of $O(M \times N)$.

$$f_{i,j} = f_{i-1,j} + f_{i,j-1} - f_{i-1,j-1} + g_{i,j} \tag{4.3}$$

The greatest advantage brought by an integral image is that for any sub-block within the image, it is possible to compute the sum of all pixels in the block in constant time. If we define a rectangle R with two corner points, (a,b) and (c,d), the sum S' of pixels $g_{i,j}$ within that rectangle as shown in figure 4.4 (b) can be computed as:

$$S = \sum_{i=a}^{c} \sum_{j=b}^{d} g_{i,j} = f_{c,d} - f_{a,d} - f_{c,b} + f_{a,b} \tag{4.4}$$

A great part of the computational complexity, however, remains in the requirement to test all of the disparity hypotheses $d \in [0,...,D-1]$ for each pixel p.

4.3 Dynamic Programming

Dynamic programming (DP), introduced for edge-based stereo estimation with one of the first methods by Ohta and Kanade [42], has been identified as a sufficiently fast global approach to solving the stereo matching problem, used in real-time solutions [24].

4.3.1 Dynamic Programming algorithm

Global stereo matching approaches aim to minimize a global energy function,

$$E(d) = E_{data}(d) + \lambda E_{smooth}(d) \tag{4.5}$$

by establishing the disparity function d which minimizes E. In the equation 4.5, the data term E_{data} accounts for the matching cost of the image pair, whereas the smoothness term E_{smooth} accounts for the desired smoothness assumptions about the disparity image. E_{data} can be defined as

$$E_{data}(d) = \sum_{(x,y)} C(x,y,d(x,y)), \tag{4.6}$$

using the disparity space formulation, where C is the matching cost disparity space image (DSI). The smoothness term is usually restricted to measurement of the disparities between neighboring pixels

$$E_{smooth}(d) = \sum_{(x,y)} \rho(d(x,y) - d(x+1,y)) + \rho(d(x,y) - d(x,y+1)), \tag{4.7}$$

and assigns a penalty to the discontinuities in the disparity map. The 2D optimization of eq. (4.5) has been shown to be NP-hard [12] for common classes of smoothness functions. However, the 1-dimensional case can be computed in polynomial time with DP, providing a global minimum for individual scanlines.

The algorithm operates under assumptions of *uniqueness* - that a single feature in the left image maps to a single feature in the right image, and *monotonicity* (also referred to as the *Ordering constraint* [43]), that the relative ordering of pixels on a scanline remains the same between the two views. Based on those assumptions, the algorithm computes the disparity by calculating the minimum matching cost path through the DSI $(x, disparity)$ image for a pair of scanlines. The calculation of the shortest path is performed within the DSI, as shown in figure 4.5. Occlusions are explicitly handled by assigning a group of pixels from one image to a single pixel in another image, which corresponds to a discontinuity or a gap in the optimum path. The path which satisfies the ordering constraint traverses the DSI using three principal moves: a *match*, which moves in the x direction keeping a constant disparity, a *vertical oc-*

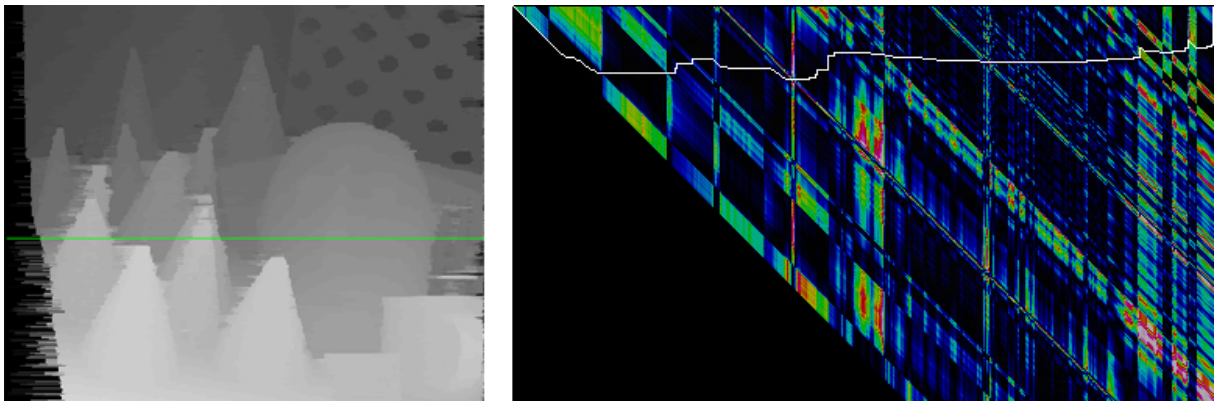


Figure 4.5: Computed disparity map and the corresponding DSI matching cost matrix structure. The computed path for the disparity scan-line highlighted in the left image is traced in white. The position x is assigned to the horizontal axis, while the disparity d is assigned to the vertical axis. The values in the structure represent the pairwise matching costs and are shown using a Jet color map.

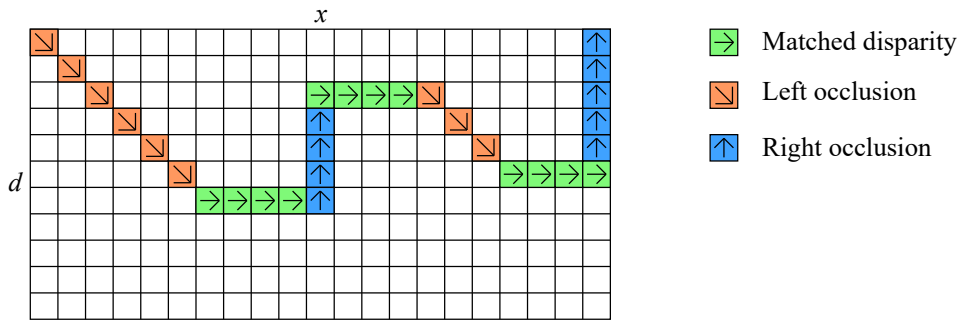


Figure 4.6: A path through a disparity space image $DSI(x,d)$ which satisfies the ordering constraint is composed of three principal moves: diagonal occlusion, vertical occlusion, and horizontal match. Which move is effected at a given pixel is determined by the computing the minimum matching cost, which depends on the previously computed cost and the match or occlusion cost. This relationship can be optimized through dynamic programming.

clusion corresponding to a right image occlusion and decreasing the disparity, and a *diagonal occlusion* which moves both in the x direction and increases the disparity, corresponding to a left occlusion, as shown in figure 4.6.

The DSI is generated by the Algorithm 1, in a common first forward pass of DP approaches which calculates the cost optimum. To obtain the actual disparities, the Algorithm 2 backtracks through the computed DSI along the optimum path, outputting a resultant pixel of the dense disparity map at each step.

Algorithm 1 DP DSI computation

Require: DSI image with the size $X_{max}, d_{max}+1$
Require: Match image with the size $X_{max}, d_{max}+1$
procedure COMPUTEDSI($I_l, I_r, y, DSI, Match$)
  for $i = 0$ to $X_{max}-1$ **do**
    $d_{high} \leftarrow i < d_{max}\ ?\ i : d_{max}$;
    $DSI(i,0) \leftarrow i * OccCost$;
    $DSI(i,d_{high}) \leftarrow i * OccCost$;
  end for
  for $i = 0$ to $X_{max}-1$ **do**
    $d_{high} \leftarrow i < d_{max}\ ?\ i : d_{max}$;
    for $j = d_{high}$ to 1 **do**
      $DiagCost \leftarrow DSI(i-1, j-1) + OccCost$;
      $VertCost \leftarrow DSI(i, j+1) + OccCost$;
      $MatchCost \leftarrow DSI(i-1, j) + C(I_l(i,y), I_r(i-j,y))$;
      $MinCost \leftarrow min(DiagCost, VertCost, MatchCost)$;
      $DSI(i,j) \leftarrow MinCost$
      if $MinCost = DiagCost$ **then**
        $Match(i,j) \leftarrow DiagMove$
      else if $MinCost = VertCost$ **then**
        $Match(i,j) \leftarrow VertMove$
      else if $MinCost = MatchCost$ **then**
        $Match(i,j) \leftarrow MatchMove$
      end if
    end for
  end for
end procedure

Algorithm 2 DP DSI backtracking

Require: Computed Match image with the size $X_{max}, d_{max}+1$

```
procedure BACKTRACK(D, y, Match)
    D_x ← D_max;
    x ← X_max − 1;
    while x > 0 and D_x > 0 do
        Move ← Match(x, D_x);
        if Move = DiagMove then
            Handle Left Occlusion
            x ← x − 1;
            D_x ← D_x − 1;
        else if Move = VertMove then
            Handle Right Occlusion
            D_x ← D_x + 1;
        else if Move = MatchMove then
            D(x, y) ← D_x;
            x ← x − 1;
        end if
    end while
end procedure
```

An important factor in the execution of Algorithm 1 is the *OccCost* parameter, or the occlusion cost, which is the cost assigned to occluded pixels. It greatly impacts the output results of the algorithm as it directly affects the decision whether the pixel is a match or occluded. The described DP method is repeated for each scanline to obtain the complete disparity map.

4.3.2 Related work

This work focuses on a pixel-based stereo DP algorithm as described by Cox et al. [43] and further expanded by Bobick and Intille [38] with the introduction of the Disparity Space Image (DSI) concept and Ground Control Points (GCPs) [44]. Veksler [45] proposed the use of tree-based structures for matching as opposed to scan-lines. Other tree-based methods were proposed afterwards [46]. Real-time implementations have been explored with CPU [24] and GPU [47] hardware. Coarse-to-fine DP approaches were also explored [48], as well as guided approaches [49] Methods have also been defined to address the scanline inconsistencies. Some of them include: using a tree-like structure which spans multiple scanlines [45], aggregating the matching cost across scanlines [50], reusing calculated paths [24] or performing a second DP pass in the vertical direction. While many research endeavors have tried to improve upon the accuracy of DP-based methods, these approaches have generally yielded an increase in computational complexity, which has been in turn addressed by high-performance hardware architectures [47]. Semi-global matching [25] (SGM), widely used in many real-time and real-world applications, makes extensive use of DP to compute the disparity for each pixel in the

image by estimating the disparity for multiple paths intersecting for every pixel of the image.

4.4 Matching cost

The proper selection of a matching cost function is vital for all passive stereo correspondence methods. In practice, the selected matching cost should provide the best possible matching accuracy under radiometric variations of input images, such as exposure differences, vignetting, varying lighting or noise [51]. In the hybrid approach, the matching cost quality is especially important for the coarse 3DRS step, as its output constrains the minimum cost path search area for the dense DP step. Evaluations [52] have shown that the Census [53] non-parametric matching cost provides the overall best performance.

4.4.1 Sum of absolute differences (SAD)

The sum of absolute differences is the most commonly used cost function [51]. For two points in images which are being matched, let W_i be a support area for cost computation. The cost function calculates the difference of corresponding pixels and extracts the modulus, summing all such collected values to produce a matching cost. For two images L and R, the matching cost is computed with the equation

$$C_{SAD} = \sum_{(x,y)\in W_i} |R(x,y) - L(x+d,y)| \tag{4.8}$$

A similar measurement is the *sum of squared differences*, where the squaring function is used in place of the modulus.

$$C_{SSD} = \sum_{(x,y)} [R(x,y) - L(x+d,y)]^2 \tag{4.9}$$

4.4.2 Zero-mean sum of absolute differences (SSD)

For the zero-mean sum of absolute differences, let $\overline{L_i)}$ be a mean value of all the pixels in the accumulation window W_i in the left image, and $\overline{R_i}$ in the right image. The zero-mean sum of absolute differences is defined as:

$$C_{ZSAD} = \sum_{(x,y)\in W_i} |[R(x,y) - \overline{R_i}] - [L(x+d,y) - \overline{L_{i+d}}]| \tag{4.10}$$

4.4.3 Census transform

The Census transform [53] [52] encodes structure in each pixel by forming a neighbourhood around each pixel p and defines a bit string $C(p)$ where each bit corresponds to a pixel in

the window around p. By comparing the pixel with the individual neighbors, a value of each bit is defined whether the selected central pixel is greater than or less than the neighbor, as denoted by equation 4.11. The bit string is then set to be the census-transformed value. The cost d_int is obtained by measuring the *Hamming distance* between the two pixels, which counts the number of different bits between the two values as shown in eq. 4.12. As with the sum of absolute differences, the matching cost can be aggregated in a support window.

$$\begin{aligned} C(p) &= \bigotimes_{q \in W} \xi(I_p, I_q) \\ \xi(I_p, I_q) &= \begin{cases} 1, if I_p < I_q, \\ 0, otherwise \end{cases} \end{aligned} \tag{4.11}$$

$$d_{int} = \underset{d}{argmin} \sum Hamming[C_l(p), C_r(p+d)] \tag{4.12}$$

The SAD, ZSAD and Census matching cost have been evaluated with the 3DRS algorithm to determine which one has the best characteristic in the implemented methods.

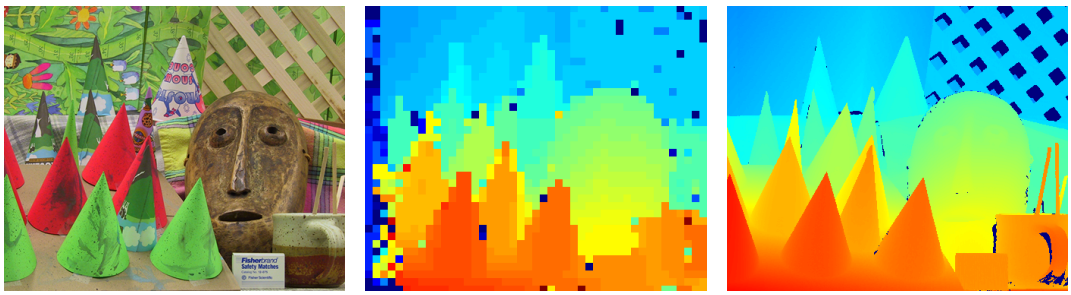
Chapter 5

Material and methods

5.1 3DRS Estimator

The 3DRS motion estimator is extremely applicable in coarse-to-fine hierarchical approaches due to its short execution time and the ability to produce coarse disparity maps. The initial assumption of many stereo matching methods is that the stereo image pairs have been previously rectified to satisfy the *epipolar constraint* [12]. Under these conditions the matching can be performed between image scanlines. Applied to the 3DRS estimator, this constraint removes the need for the estimation of the y component of the vector. Furthermore, as the disparity values, unlike motion vectors, are highly unlikely to be negative, the output of the disparity estimation can be clipped to address only the positive values. A 3DRS estimator with the described modifications is used as the initial step of two hybrid methods, one based on the Winner-take-all (WTA), and one based on the Dynamic Programming (DP) method. Figure 5.1 shows the output of the 3DRS estimator. The estimates are largely correct, with spurious outliers where the algorithm was not able to converge due to occlusions.

As the 3DRS algorithm will converge to a solution in constant time per image size, it can



(a) Left image of the stereo pair. (b) The 3DRS disparity map. (c) Ground truth disparities.

Figure 5.1: The result of the 3DRS disparity estimation. The estimator yields a coarse disparity map with spurious outliers located mostly in occlusion areas. The disparities are colored using the *Jet* color map.

be used to narrow down the disparity space locally within an image, ensuring that the actual disparity will be reached. It is to be expected that false positives or incorrect local optima where the algorithm might become trapped, would also be avoided. To reduce the disparity range, the idea is to re-apply the 3DRS assumption that objects are larger than blocks, and that the local neighborhood can be used as a predictor of the actual disparity range. To each of the predictors, a disparity range is assigned. These ranges are then combined to get the actual disparity ranges to be evaluated with the dense disparity estimator per each block. In general, this represents a local culling of the disparity space image.

For the purpose of obtaining the best possible disparity guidance out of the 3DRS motion estimator, two major enhancements have been implemented. Originally defined on the sum of absolute differences, the 3DRS algorithm was expanded with more capable matching costs such as the Census cost. Additionally the capability was added to the 3DRS estimator to use an arbitrary matching block size per each block in the image. This allows for smaller blocks to be matched with larger support regions, yielding higher output quality (albeit at the cost of decreased performance).

5.2 The hybrid 3DRS-WTA method

In the proposed hybrid 3DRS-WTA method [57], or 3DRS-guided-WTA (3GWTA), a coarse analysis is performed first using the 3DRS algorithm, with the image divided into blocks. This yields a coarse map of estimated disparities d_{ij} assigned to each of the blocks. For each block, a disparity range to be tested using the WTA method is obtained by re-applying the 3DRS assumption that the correspondence vectors of a block's spatio-temporal neighborhood can be good estimates of the final result. Based on the user-defined parameter r, an initial disparity range is assigned to the estimated block and the blocks in its 8-neighborhood. Finally, a union of individual disparity ranges of blocks within the 8-neighborhood is performed to obtain a list of disparity hypotheses to be tested within the WTA optimization step.

The parameter r limits the range of disparity hypotheses evaluated per each block within the 8-neighborhood to $[d-r,...,d+r]$, where d is the result of the 3DRS disparity estimation for each given block. In the optimum case where all of the 3DRS results in the 8-neighborhood are equal and the ranges perfectly match, the total number of hypotheses to test is $2*r+1$, equal to a single range. In the extremely unlikely worst case where the disparity ranges of within the 8-neighborhood do not intersect, the total number of disparity hypotheses to be evaluated is $9*(2*r+1)$.

The hybrid 3DRS-WTA method is outlined in Algorithm 3.

Algorithm 3 Coarse-to-fine WTA optimization

Require: r: integer > 0; ▷ disparity range per block
Require: b: integer > 0; ▷ 3DRS block size
Require: I_l: image; ▷ Left image
Require: I_r: image; ▷ Right image
Require: I_{dc}: array; ▷ 3DRS coarse output
Require: I_d: image; ▷ Disparity image

$3DRS(I_l, I_r, I_{dc}, b)$; ▷ Compute rough 3DRS map
for all block disparities $dc_{ij} \in I_{dc}$ **do**
  $rct := Rectangle(i*b, j*b, (i+1)*b, (j+1)*b)$;
  $L = NewList$;
  for $v = -1$ to 1 **do**
    for $h = -1$ to 1 **do**
      if $ValidBlock(i+h, j+v)$ **then**
        $AddToList(L, dc_{i+h,j+v})$
      end if
    end for
  end for
  $SortAscending(L)$
  $dmin := dmax := 0$;
  for all dc in L **do**
    $dmin_n := dc - r$;
    $dmax_n := dc + r$;
    if $dmin_n > dmax$ **then**
      $WTA(I_l[rct], I_r, dmin, dmax, I_d)$;
      $dmin := dmin_n$;
    end if
    $dmax := dmax_n$;
  end for
  $WTA(I_l[rct], I_r, dmin, dmax, I_d)$;
end for

5.3 3DRS-guided Dynamic Programming (3GDP)

In the same manner as the 3GWTA algorithm, the 3GDP algorithm, proposed in [54], uses the assumptions of the 3DRS algorithm to exploit the spatial neighbourhood of a coarse estimation to cull the DSI space and reduce the computational effort required for optimization.

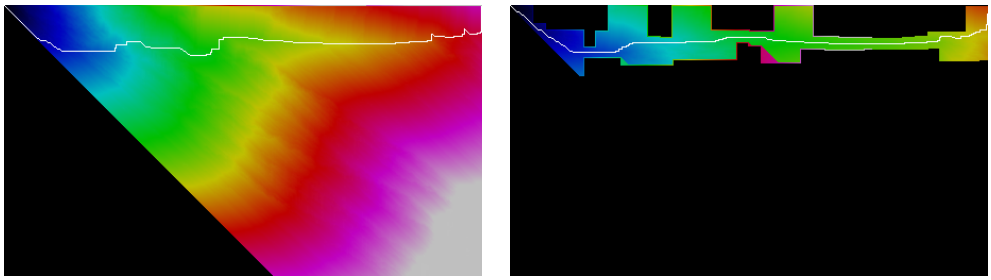
To compute the dense disparities from the coarse map the DP algorithm is applied. The computational cost of the DP method is directly proportional to the range of disparities being estimated. To reduce the computational cost, the coarse disparity map is used as a reference for piece-wise limitation of the disparity range within the DSI, which is divided into segments with width matching the width of the 3DRS blocks. Each segment has an assigned disparity range (d_s^{min}, d_s^{max}), which can be determined from the coarse disparity value plus or minus a range constant, an algorithm parameter. However, this does not account for the discontinuities in the optimum matching path. The use of varying estimated values obtained from the coarse disparity map, especially with the presence of unmatched outliers, may result in disconnected ranges, which would effectively prevent proper DSI traversal and backtracking in the DP step of the method. In order for the estimation to work, a fully end-to-end connected DSI structure must be ensured.

To achieve this, the 3DRS assumption that the spatial neighborhood of the estimated block provides a good predictor for that block is re-applied. By building the disparity range for a particular segment based on the values of neighboring segments with an additional clearance constant R_{Off}, it is ensured that each of the segments is connected to its neighboring segments while retaining space for potential deviations from the coarse block disparity.

To generate a connected DSI, the d_s^{min} and d_s^{max} are computed for each segment as shown in eq. 5.1.

$$\begin{aligned} d_s^{min}(x,y) &= min(d_{3drs}(x+i,y+j)) - R_{Off} \\ d_s^{max}(x,y) &= max(d_{3drs}(x+i,y+j)) + R_{Off} \\ i &= -1,0,1 \\ j &= -1,0,1 \end{aligned} \tag{5.1}$$

The resultant segmented DSI is shown in Figure 5.2 (b). Using the segmented DSI, the DP algorithm produces a dense disparity map. The comparison of the DP and 3GDP DSI structures demonstrates the reduction of required computational effort provided by the coarse 3DRS disparities.



(a) DP with the full disparity range.

(b) 3DRS-guided DP.

Figure 5.2: DSI structures and computed paths for DP and 3DRS-guided-DP algorithms. Color-shaded areas represent computed matching costs. The value of the cost is shown using the jet color map.

Chapter 6

Results and discussion

6.1 Methodology

6.1.1 Evaluation

In the evaluation of the algorithm, two aspects of quality for the implemented methods were measured. The first is the accuracy of the output results indicated by the percentage of incorrect pixels as calculated by equation 6.1

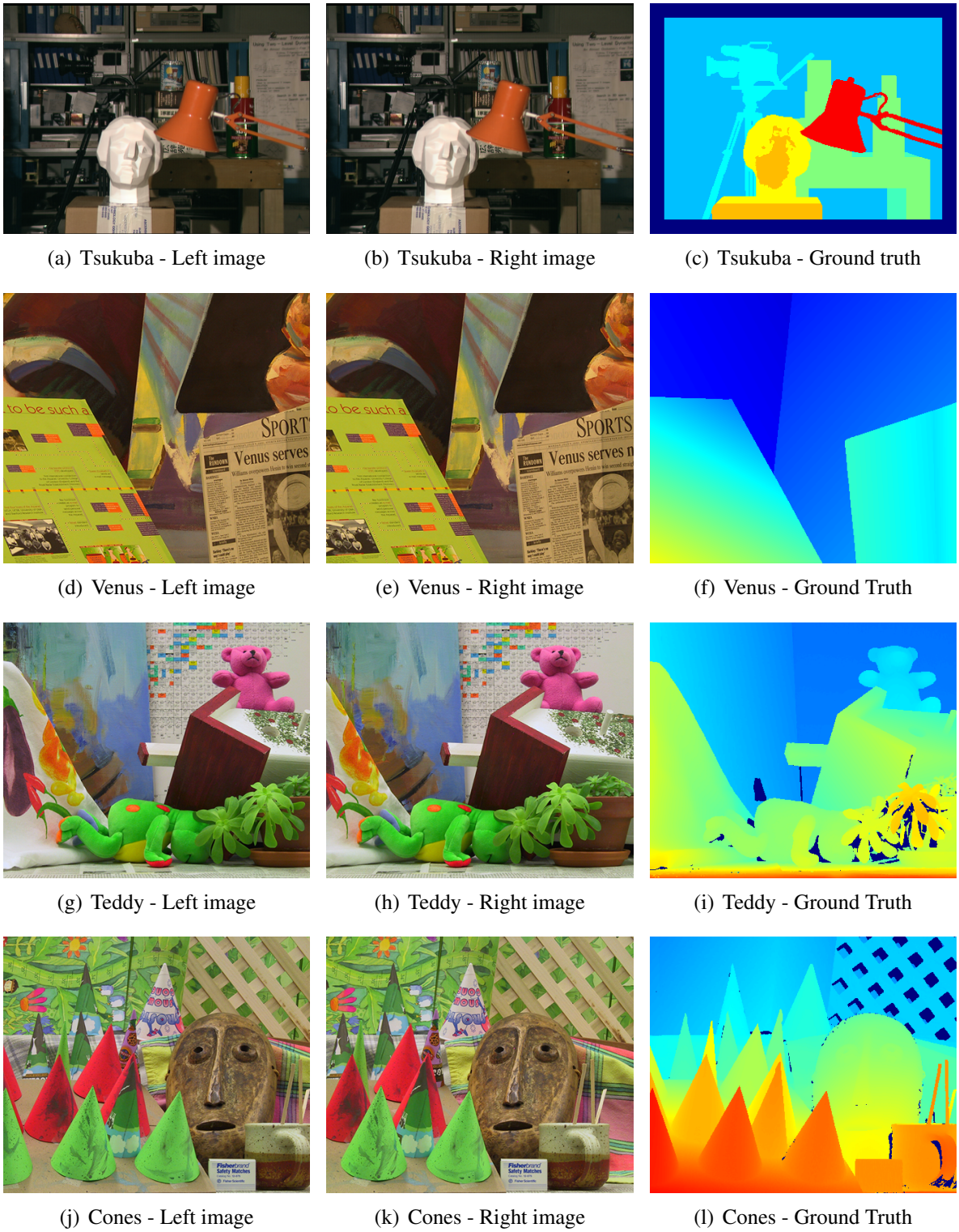
$$B = \frac{1}{N} \sum_{(x,y)} (|d_C(x,y) - d_T(x,y)| > \delta_D) \tag{6.1}$$

where d_C is the estimated disparity map, and d_T is the ground truth disparity map[12]. The threshold δ_D defines the accepted difference between a ground truth disparity and the computed disparity. Within the evaluation presented in this thesis a value of $\delta_D = 1$, as suggested in [12]. The second aspect is the algorithm run-time performance, indicated by the computation time, which is measured by querying the system performance counters.
The methods were tested on reference images from the Middlebury set [22] [55]: “Tsukuba”, “Venus”, “Teddy” and “Cones”, shown in the figure 6.1.

For the evaluation of computed disparities in comparison to the ground truth, three distinct areas of interest are considered in the ground truths of the Middlebury evaluation:

- All pixels (*All*) - every pixel of the disparity image is evaluated, whether it is occluded or not
- Non-occluded pixels (*nonocc*) - only the non-occluded pixels (with a certain match) are evaluated.
- Discontinuity regions (*disc*) - only the pixels in the discontinuity (occlusion) areas are evaluated.

The equation 6.1 for computing the percentage of incorrect pixels is expanded with a ground



(a) Tsukuba - Left image (b) Tsukuba - Right image (c) Tsukuba - Ground truth

(d) Venus - Left image (e) Venus - Right image (f) Venus - Ground Truth

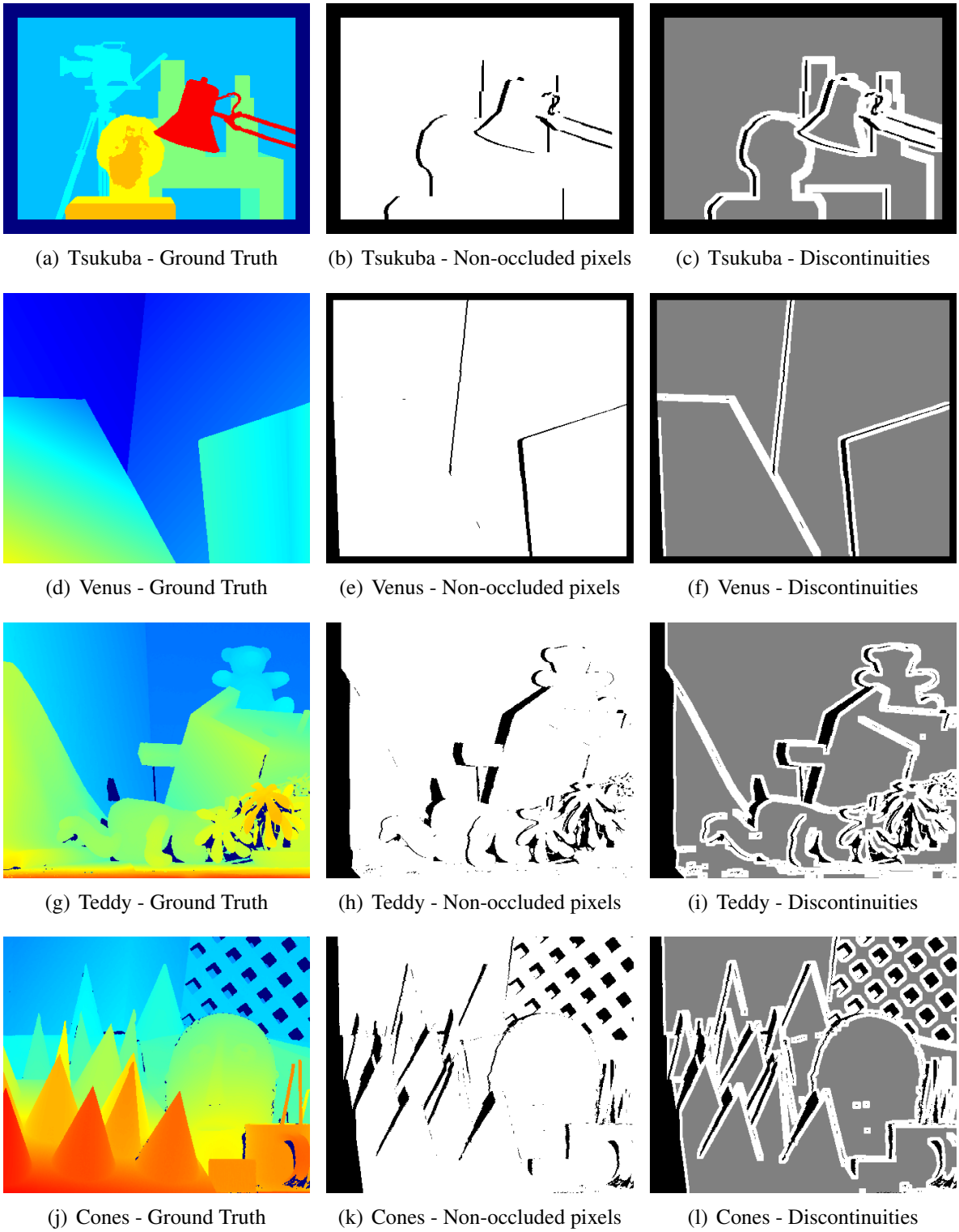
(g) Teddy - Left image (h) Teddy - Right image (i) Teddy - Ground Truth

(j) Cones - Left image (k) Cones - Right image (l) Cones - Ground Truth

Figure 6.1: Source images from the Middlebury set - Left image, right image, ground truth disparities. The disparities are scaled to full range $(0,255)$ and colored using the *Jet* color map. The stereo pair is shown in color.

truth validity function $M_{gt}(x,y)$.

$$B = \frac{\sum\limits_{(x,y)} \left(|d_C(x,y) - d_T(x,y)| \cdot M_{gt}(x,y) > \delta_D\right)}{\sum\limits_{(x,y)} M_{gt}(x,y)} \tag{6.2}$$



(a) Tsukuba - Ground Truth (b) Tsukuba - Non-occluded pixels (c) Tsukuba - Discontinuities

(d) Venus - Ground Truth (e) Venus - Non-occluded pixels (f) Venus - Discontinuities

(g) Teddy - Ground Truth (h) Teddy - Non-occluded pixels (i) Teddy - Discontinuities

(j) Cones - Ground Truth (k) Cones - Non-occluded pixels (l) Cones - Discontinuities

Figure 6.2: Evaluation masks from the Middlebury set - Ground truth, non-occluded areas, discontinuity areas. The masks are encoded as: white pixels - included, black and gray pixels - not included.

Where $M_{gt}(x,y)$ is a mask image through which a pixel is included in the evaluation if the value of M_gt is 1. For the 4 evaluation samples, the occlusion and discontinuity masks are shown in figure 6.2. In the course of the evaluation, for each of the samples the tested method is used

to obtain the disparity map, which is then evaluated using equation 6.2 for each of the defined cases. The value of 6.1 is recorded for the three particular cases (all pixels, nonoccluded, discontinuity areas).

6.1.2 Environment

The 3DRS, WTA, DP and the proposed hybrid methods (3GWTA, 3GDP) have been implemented within *StereoTest*, a visual evaluation environment developed for the purpose of evaluating stereo algorithms. The motivation for designing a new simulation environment was to reduce the dependency of the tested methods on external libraries as much as possible, with the ultimate goal of developing a real-time hardware implementation for the tested methods. All computation and image encoding is performed using integer and fixed-point arithmetic, with integer parameters. The environment does not implement a sub-pixel disparity refinement step after the optimization, as the goal of the evaluation was to compare the raw results of the disparity computation methods. The environment is coded in C# and operates under Microsoft's .Net Framework. All methods operate on 8-bit grayscale images.

The measurements have been performed on a 64-bit desktop computer equipped with an Intel Core i5-6600K processor and 32 gigabytes of available memory.

6.2 Results

6.2.1 3DRS results

With the behavior of Winner-take-all (WTA) and Dynamic Programming mostly well researched in previous work, the measurement was focused on the properties of the 3DRS algorithm in order to extract the parameters which would yield the highest quality guidance for the dense estimation steps.

The relationship between the 3DRS block size and the computation time per image is shown in Table 6.1 and Figure 6.3 . The *Prep+Calc* time involves the non-recurring steps in a 3DRS computation, such as preparation of resources and input images (memory allocation, transformation of inputs where necessary), while the *Calc* time measures only the time required to perform a single 3DRS pass, which is added for each iteration of 3DRS. The measurement was performed for a single 3DRS pass, using the Census cost.

Another important property of the 3DRS algorithm which affects the guidance of the later dense estimation step (WTA, DP) is convergence. Assuming that a low-confidence guidance map, which exhibits a greater estimation error, will adversely affect the final results of the hybrid method, establishes the goal to extract the greatest quality disparity map in the initial estimation step. Therefore, the accuracy of the 3DRS-produced coarse disparity map depending on the

Table 6.1: 3DRS execution time for various Middlebury set image sizes, measured in milliseconds, for varying 3DRS block size.

			Blocksize					
Set	**Image size**	**Action**	**1**	**2**	**3**	**4**	**5**	**6**
Tsukuba	384x288	Prep+Calc	69,80	48,67	45,65	45,78	44,48	41,07
		Calc	36,62	17,76	16,17	12,29	12,01	11,51
Teddy	450x375	Prep+Calc	107,43	74,54	69,46	68,43	66,80	63,42
		Calc	58,21	27,08	25,46	19,71	18,21	17,77
Baby2	620x555	Prep+Calc	217,73	148,94	139,22	135,17	135,04	128,29
		Calc	114,29	55,44	50,41	38,71	37,08	35,98
			Blocksize					
Set	**Image size**	**Action**	**7**	**8**	**9**	**10**	**11**	**12**
Tsukuba	384x288	Prep+Calc	44,21	42,55	44,25	44,19	44,59	42,70
		Calc	11,39	10,94	11,70	11,64	11,57	10,96
Teddy	450x375	Prep+Calc	66,82	66,56	67,28	66,57	67,47	67,70
		Calc	17,46	17,16	17,96	17,73	17,77	17,65
Baby2	620x555	Prep+Calc	131,99	129,74	133,78	132,25	134,26	132,37
		Calc	35,29	35,13	36,96	36,24	36,36	35,52

Table 6.2: Accuracy, expressed as the percentage of incorrect pixels for the produced disparity maps (lower is better) vs. the number of 3DRS passes (and the corresponding elapsed time). The time is measured in milliseconds.

		Number of Passes							
Set	**Measure**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Tsukuba	Time	42,75	54,88	65,86	78,78	90,39	103,13	116,83	127,64
	Bad Pixels	22,46%	17,94%	17,64%	17,56%	18,07%	18,41%	19,09%	19,30%
Venus	Time	67,59	83,81	100,06	120,49	137,30	153,76	171,65	189,84
	Bad Pixels	12,05%	4,43%	4,57%	4,72%	4,94%	5,00%	5,23%	5,23%
Teddy	Time	68,67	84,66	99,27	117,24	136,42	154,66	169,82	186,99
	Bad Pixels	33,95%	21,88%	21,33%	21,18%	21,33%	21,26%	21,47%	21,34%
Cones	Time	67,25	82,05	103,33	118,99	136,83	154,61	170,24	188,96
	Bad Pixels	31,84%	17,90%	17,15%	16,33%	16,29%	16,27%	16,28%	16,09%

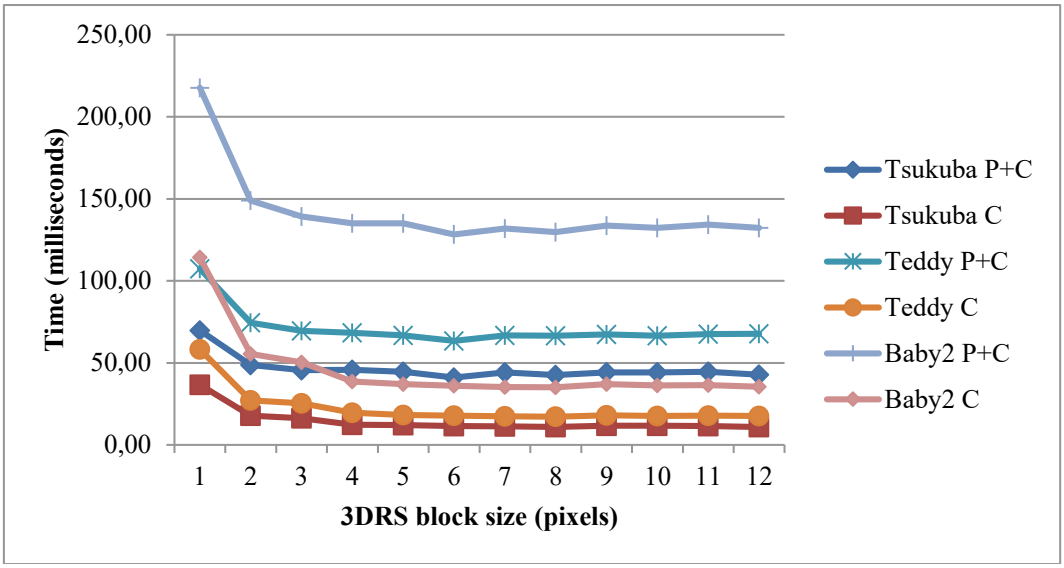


Figure 6.3: Execution times (in milliseconds) of 3DRS for the Middlebury set with varying 3DRS block sizes calculated - data from Table 6.1.

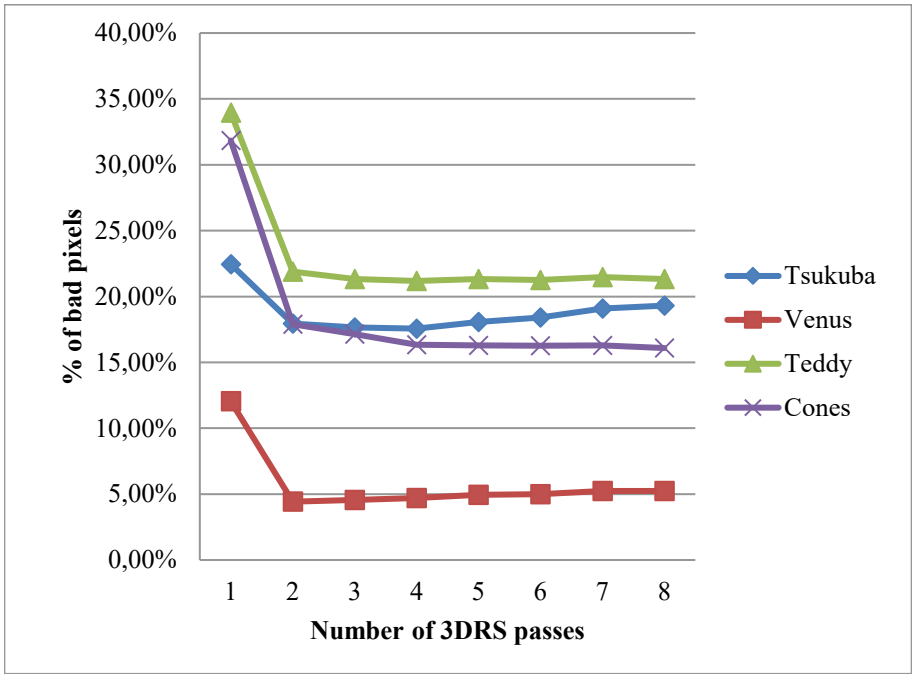


Figure 6.4: Disparity map accuracy, expressed as the percentage of bad pixels, in the 3DRS coarse disparity map shown in dependency to the number of 3DRS passes - data from Table 6.2.

number of 3DRS passe, the 3DRS block size and the size of the 3DRS update set was measured. The results are provided in table 6.2 and Figure 6.4. The percentage of incorrect pixels was calculated using equation 6.1, with $\delta_D = 1$, using the full image area with defined disparities (including the occluded areas). Additionally, table 6.2 and figure 6.5 show the relationship of the execution time for the 3drs algorithm depending on the number of passes.

Table 6.3 and figure 6.6 show the accuracy of the 3DRS algorithm simultaneously against the varying block size and number of passes. The data is computed for the "Cones" image of

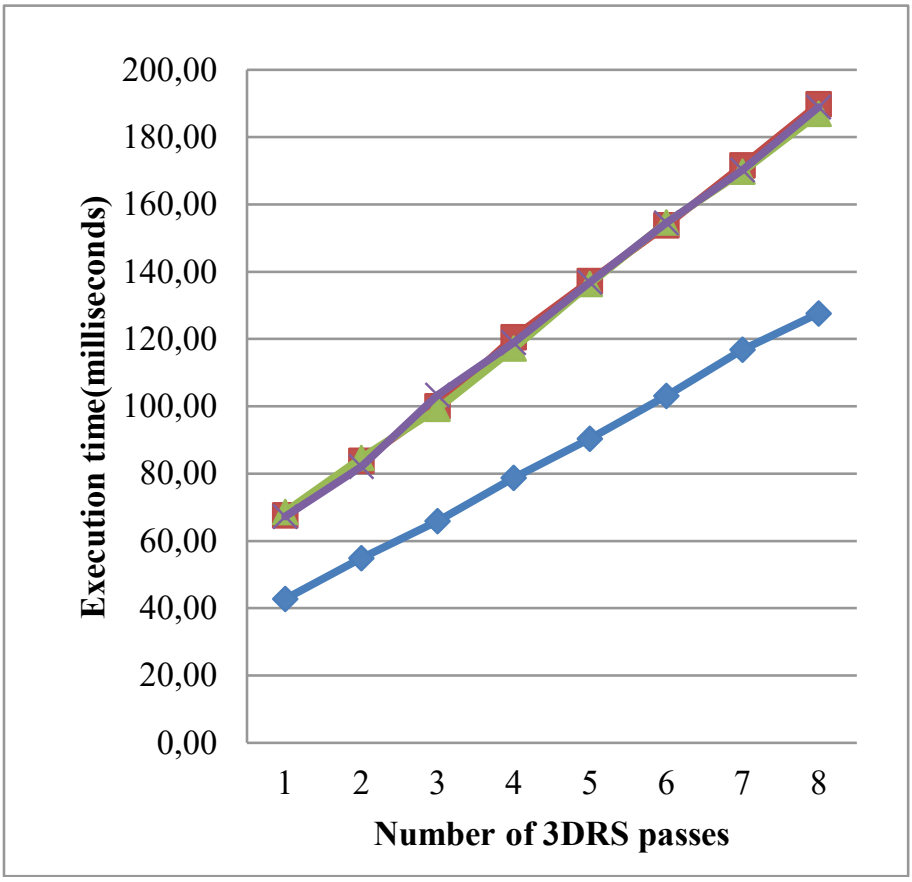


Figure 6.5: Algorithm execution time in dependency to the number of 3DRS passes - data from Table 6.2.

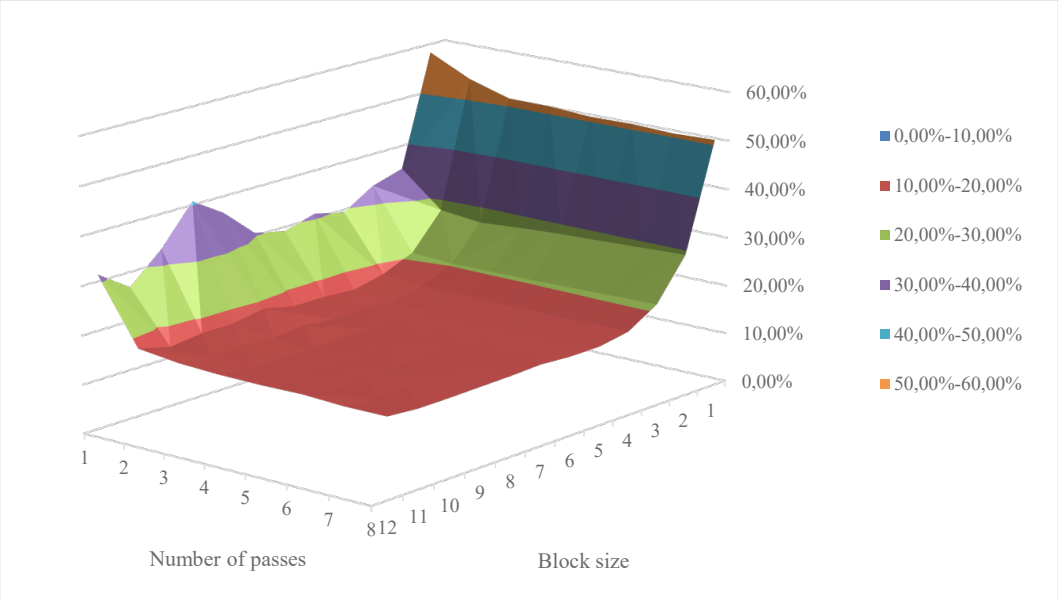


Figure 6.6: Accuracy of 3DRS against varying block passes and block size, simultaneosly - computed for the "Cones" image of the Middlebury set with the SAD matching cost; data from table 6.3. Accuracy is expressed as the percentage of errors. Lower is better.

Table 6.3: 3DRS accuracy, expressed as percentage of bad pixels (lower is better) shown in dependency to both the number of 3DRS passes and the block size.

	Blocksize											
No. of passes	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	58,19 %	35,13 %	33,26 %	29,42 %	30,93 %	29,09 %	30,60 %	36,53 %	40,70 %	32,82 %	26,91 %	31,38 %
2	54,11 %	28,02 %	20,82 %	18,16 %	16,73 %	17,19 %	17,14 %	18,57 %	17,43 %	17,40 %	16,62 %	18,24 %
3	51,56 %	27,11 %	20,28 %	17,41 %	16,03 %	16,69 %	16,36 %	16,88 %	16,37 %	16,79 %	16,28 %	17,25 %
4	51,53 %	27,71 %	20,37 %	17,34 %	15,96 %	16,52 %	16,41 %	16,33 %	16,05 %	16,29 %	15,92 %	16,86 %
5	50,95 %	28,20 %	20,44 %	17,34 %	16,02 %	16,33 %	16,47 %	16,33 %	15,94 %	16,22 %	15,76 %	16,76 %
6	51,10 %	28,50 %	20,59 %	17,31 %	16,09 %	16,31 %	16,47 %	16,24 %	16,01 %	16,19 %	15,63 %	16,89 %
7	50,75 %	28,86 %	20,78 %	17,31 %	16,12 %	16,24 %	16,47 %	16,30 %	16,19 %	16,12 %	15,79 %	16,49 %
8	51,04 %	29,21 %	20,90 %	17,23 %	16,15 %	16,13 %	16,54 %	16,26 %	16,15 %	16,02 %	15,99 %	16,51 %

the Middlebury set.

The effect of the 3DRS update set on the convergence property was explored by evaluating the algorithm concurrently with varying maximum update value and the number of passes, measuring the resulting accuracy. The results of the evaluation are given in table 6.4 and figure 6.7.

Table 6.4: 3DRS accuracy, expressed as percentage of bad pixels (lower is better) shown in dependency to the number of 3DRS passes and the update vector set.

	Number of passes							
Maximum update	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %
2	100 %	100 %	100 %	100 %	100 %	98,52 %	85,16 %	80,67 %
4	100 %	99,98 %	83,23 %	37,60 %	23,57 %	20,32 %	16,72 %	16,41 %
8	60 %	19,62 %	16,21 %	15,74 %	15,29 %	15,58 %	15,46 %	15,61 %
16	32,82 %	17,40 %	16,79 %	16,29 %	16,22 %	16,19 %	16,12 %	16,02 %
32	45 %	19,85 %	16,53 %	15,92 %	15,86 %	15,77 %	15,61 %	15,60 %
64	40,13 %	19,13 %	17,03 %	16,84 %	16,38 %	16,42 %	16,34 %	16,42 %

The 3DRS algorithm was also tested to evaluate its performance with the SAD, SSD, Zero-mean SAD, and Census cost functions. The comparison of the 3DRS results with different matching costs is shown in Figure 6.8, with a quantitative analysis provided in table 6.5, showing that the census cost achieves the lowest matching error rate. Based on the results, the Census cost was selected as the cost function to be used as the block matching cost in the 3DRS and WTA steps, as well as the pixel-wise matching cost in the Dynamic Programming step.

The 3DRS implementation does not include advanced optimizations such as hierarchical processing or penalties [56], but supports multiple iterations to improve algorithm convergence on static images.

6.2.2 Additional measurements

To assess the performance of the WTA and DP algorithms without the 3DRS guidance, the algorithms were evaluated and tuned to their optimal settings. Additionally the dependency of

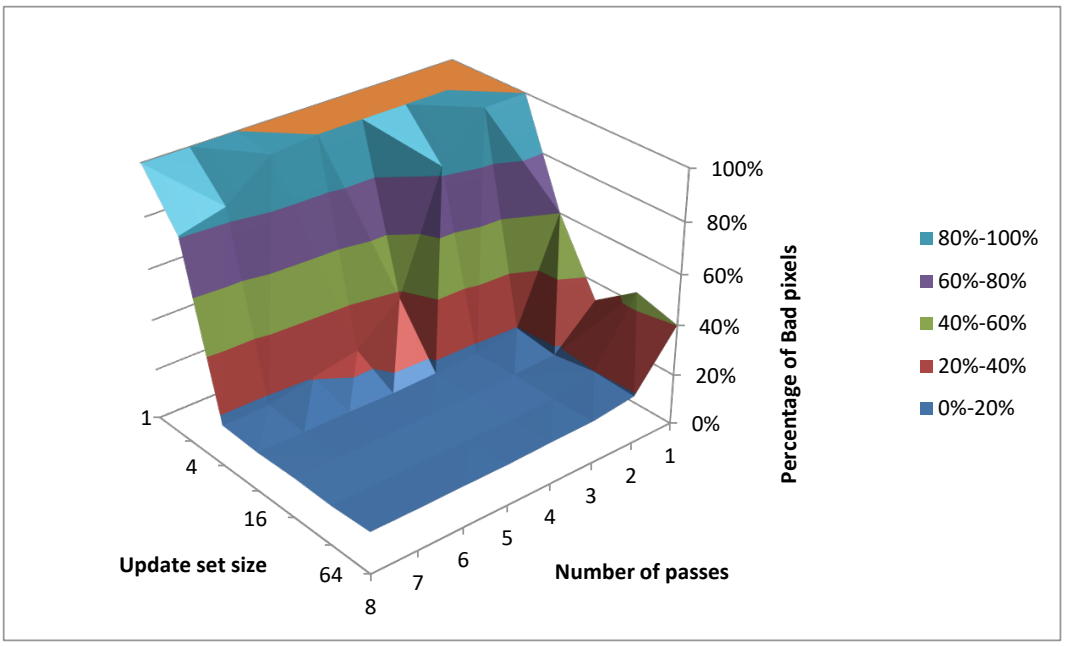


Figure 6.7: Accuracy of 3DRS against varying block passes and the maximum value of the update set, simultaneously - computed for the "Cones" image of the Middlebury set; data from table 6.4. Accuracy is expressed as the percentage of errors. Lower is better.

Table 6.5: Comparison of overall error rates for various matching costs using the 3DRS algorithm, considering all pixels. Lower is better.

Set	SAD	SSD	ZSAD	Census
Tsukuba	12,61%	12,44%	14,40%	21,44%
Venus	8,23%	7,81%	7,37%	6,85%
Teddy	30,26%	30,49%	24,58%	20,65%
Cones	26,70%	24,28%	22,50%	16,22%
Average	19,45%	18,76%	17,21%	**16,29%**

the algorithms on the input disparity range was measured and is shown in table 6.6 and figure 6.9

6.2.3 3DRS+WTA Experimental Results

The proposed local winner-take-all method guided by 3DRS was implemented in the StereoTest environment.

For the Middlebury set, an evaluation was conducted to measure the influence of the local range parameter r on the overall reduction of the disparity range to be checked by the WTA step. An overview of how the accuracy and the total number of disparity hypotheses (and execution time) correlate is shown in Table 6.7, as well as depicted in Figure 6.10.

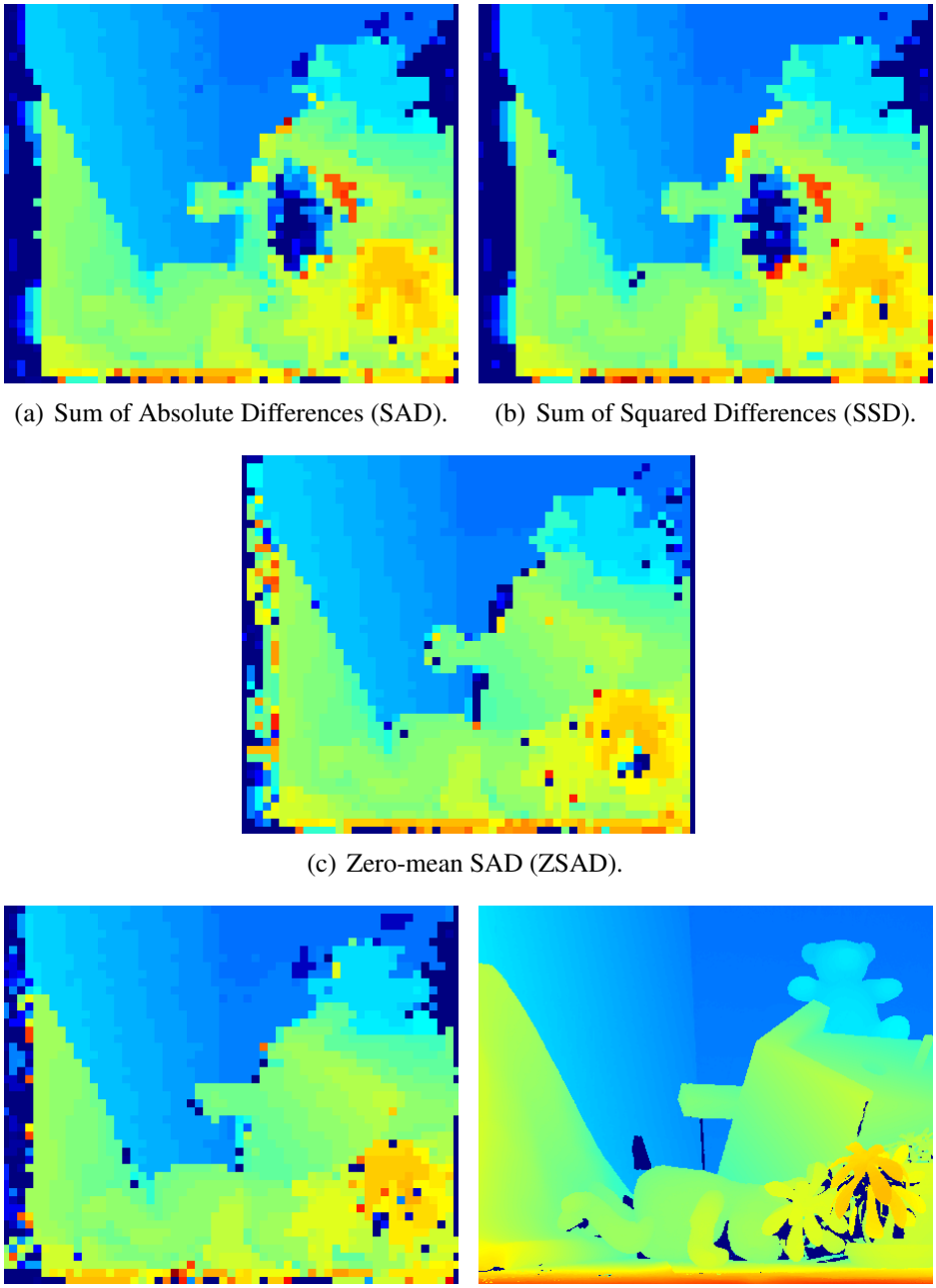
While most local methods expect the user to specify a certain disparity search range to optimize the performance, the proposed hybrid method will address any range of disparities which occurs in the input stereo pair with constant performance.

Table 6.6: Execution time of WTA and DP algorithms depending on the disparity range, in milliseconds. Slope of the curve at each point provided as illustration of (non)linearity.

	Execution time (ms)		**Slope** $\frac{time}{range}$	
Disparity range (pixels)	**WTA**	**DP**	**WTA**	**DP**
16	342,60	165,90	21,413	10,369
32	647,55	282,97	20,236	8,843
48	954,55	395,17	19,886	8,233
64	1260,52	488,20	19,696	7,628
80	1571,04	598,28	19,638	7,479
96	1877,33	688,09	19,556	7,168
112	2201,79	775,38	19,659	6,923
128	2492,01	867,21	19,469	6,775
144	2791,21	948,23	19,383	6,585
160	3087,44	1019,59	19,297	6,372
176	3393,45	1103,33	19,281	6,269
192	3703,28	1165,23	19,288	6,069
208	3990,41	1228,42	19,185	5,906
224	4291,67	1289,92	19,159	5,759
240	4582,14	1344,70	19,092	5,603
255	4866,31	1395,00	19,084	5,471

Table 6.7: Accuracy of generated disparities and the overall DSI space actually searched for the 3GWTA method, expressed as a percentage of incorrect pixels (lower is better) and percentage of searched space (lower is better), in dependency to the parameter R.

Parameter R	*Error rate*				*Space searched*			
	Tsukuba	**Venus**	**Teddy**	**Cones**	**Tsukuba**	**Venus**	**Teddy**	**Cones**
0	12,27%	3,45%	18,66%	14,17%	1,21%	0,97%	1,24%	1,23%
1	12,09%	3,51%	18,92%	14,05%	2,41%	1,87%	2,47%	2,42%
2	12,92%	3,62%	19,01%	14,37%	3,35%	2,74%	3,52%	3,48%
3	13,05%	3,81%	19,15%	14,40%	4,16%	3,55%	4,50%	4,45%
4	13,24%	3,83%	19,30%	14,43%	4,80%	4,27%	5,43%	5,38%
5	13,41%	3,87%	19,41%	14,42%	5,34%	4,96%	6,33%	6,27%
6	13,56%	3,91%	19,24%	14,44%	5,84%	5,59%	7,20%	7,13%
7	13,61%	3,92%	19,17%	14,49%	6,34%	6,20%	8,10%	7,98%
8	13,87%	3,95%	19,20%	14,46%	6,79%	6,71%	8,88%	8,81%



(a) Sum of Absolute Differences (SAD). (b) Sum of Squared Differences (SSD).

(c) Zero-mean SAD (ZSAD).

(d) Census Transform. (e) True Disparities.

Figure 6.8: 3DRS coarse disparity maps for the "Teddy" test image based on different matching costs, shown using the *Jet* color map.

The 3DRS-guided WTA method implementation follows Alg.3 directly, implementing the basic algorithm as denoted in Eq.4.1 with $w(p,q) = 1$. Traditional square windows, without shifting or specific filtering, are used to aggregate the cost values, for which SAD, SSD and Census cost functions can be used. Cost aggregation is accelerated with the use of integral images similar to the technique in[39]. A sub-pixel disparity refinement step was not performed after the optimization, as the goal of the evaluation was to compare the raw results of the optimization methods.

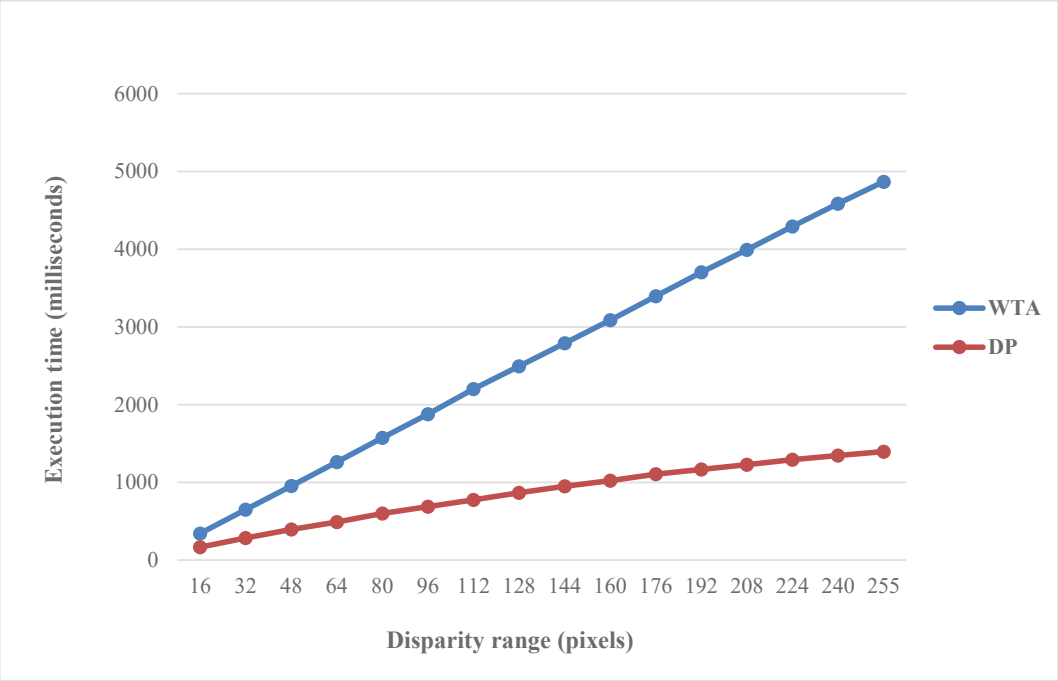


Figure 6.9: Correlation of the execution time of Winner-take-all and Dynamic programming methods and the designated disparity search range. Data from table 6.6. Time in milliseconds, search range in pixels.

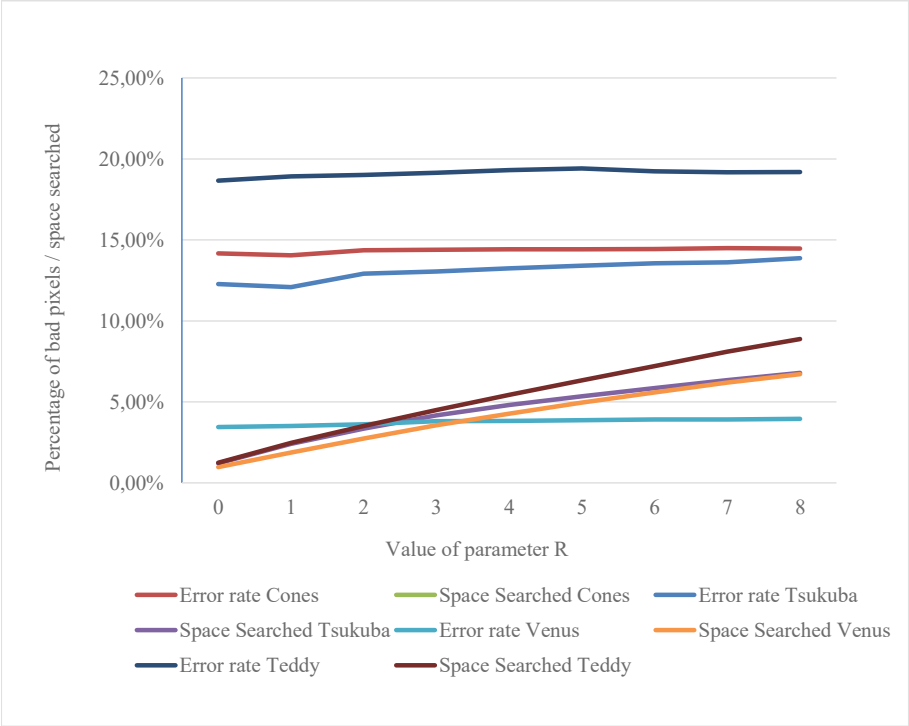
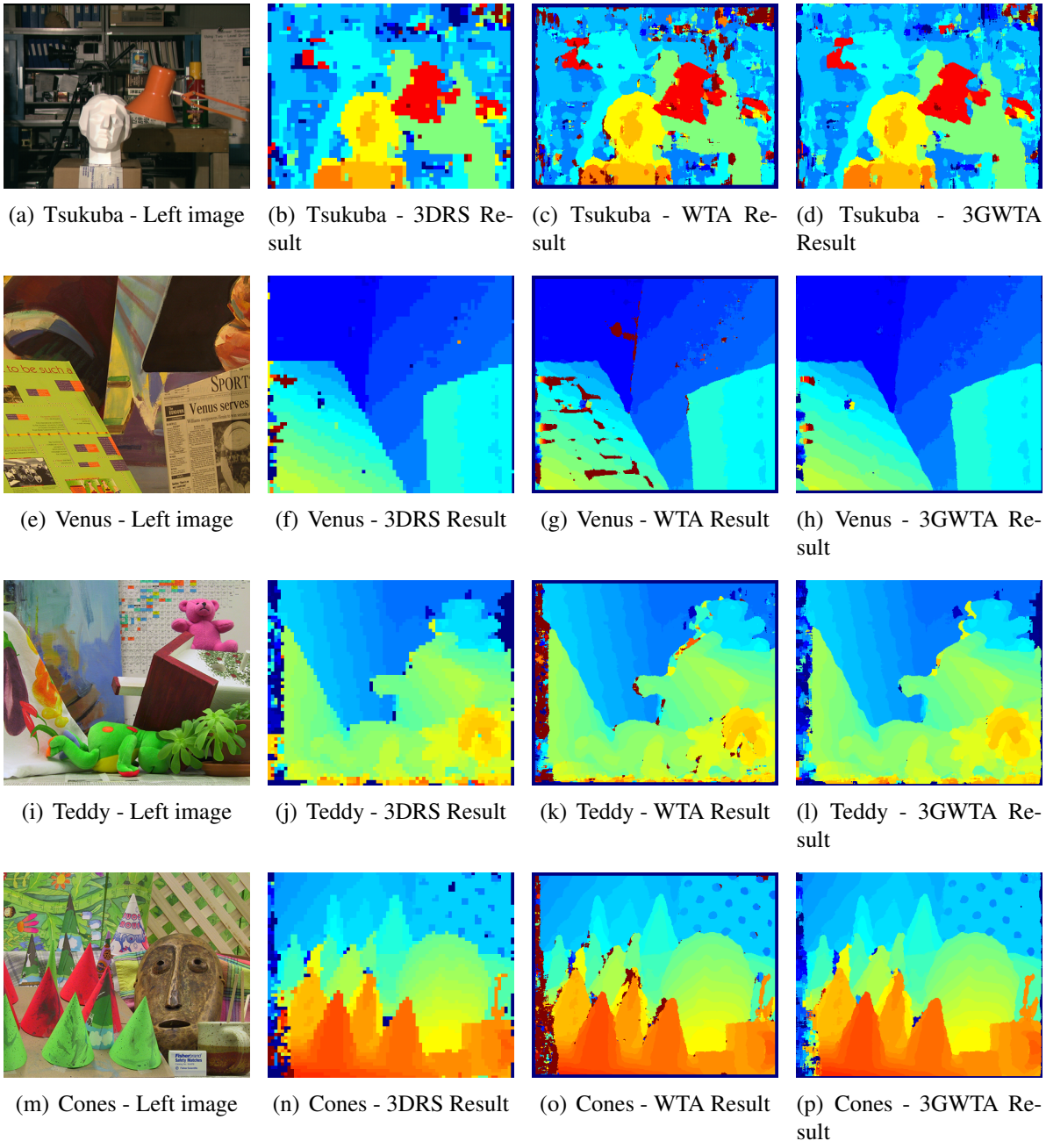


Figure 6.10: Correlation of accuracy (percentage of incorrect pixels) and the processed cost volume (total number of disparity hypotheses for the image) for the 3DRS-guided WTA algorithm. Lower values are better. Data from table 6.7.

The resultant disparity maps are shown in Figure 6.11.

The measured accuracy results in Table 6.8 and shown in Figure 6.12 show that the 3DRS



(a) Tsukuba - Left image (b) Tsukuba - 3DRS Result (c) Tsukuba - WTA Result (d) Tsukuba - 3GWTA Result

(e) Venus - Left image (f) Venus - 3DRS Result (g) Venus - WTA Result (h) Venus - 3GWTA Result

(i) Teddy - Left image (j) Teddy - 3DRS Result (k) Teddy - WTA Result (l) Teddy - 3GWTA Result

(m) Cones - Left image (n) Cones - 3DRS Result (o) Cones - WTA Result (p) Cones - 3GWTA Result

Figure 6.11: Algorithm results for WTA - Left reference image and resultant disparities for 3DRS, WTA, and 3DRS-Guided WTA methods. The disparities are scaled to full range $[0, 255]$ and colored using the *Jet* color map.

algorithm can provide coarse results with a level of accuracy comparable to the dense stereo optimizations such as the WTA, at a fraction of the computational complexity. In the initial evaluation of the algorithm [57], it was shown that the accuracy of the WTA method was typical for a fixed window implementation with the chosen window size, and it was shown that the proposed hybrid method either retained the accuracy of the WTA method or slightly improved it. In the implementation described in this work, the effects of the Census cost being applied in both the 3DRS and the WTA step of the algorithm yield significant improvements in quality.

Table 6.8: Accuracy of generated disparities for the 3GWTA method, expressed as a percentage of incorrect pixels (lower is better); Non-occ - non-occluded regions only; All - all pixels; Disc - discontinuity regions only.

Set	*3DRS*			*WTA*			*3GWTA*		
	Non-occ	**All**	**Disc**	**Non-occ**	**All**	**Disc**	**Non-occ**	**All**	**Disc**
Tsukuba	12,58%	13,90%	26,31%	14,42%	15,83%	26,50%	10,73%	12,09%	24,67%
Venus	2,74%	3,97%	23,63%	5,21%	6,48%	23,64%	2,20%	3,44%	23,00%
Teddy	12,09%	19,65%	28,70%	13,34%	21,25%	28,97%	10,95%	18,67%	26,74%
Cones	6,54%	14,98%	18,34%	8,99%	17,76%	16,82%	5,62%	14,05%	14,04%
Average		15,29%			16,60%			**13,85%**	

Table 6.9: Comparison of execution times for the 3DRS, WTA and 3GWTA method. Times are given in milliseconds. The speed-up ratio is given as a dimensionless quantity.

Set	t_{3DRS}	t_{WTA}	t_{3GWTA}	$\frac{t_{WTA}}{t_{3GWTA}}$
Tsukuba	141,22	3092,75	508,80	**6,08**
Venus	211,9	4616,71	627,35	**7,36**
Teddy	171,12	4762,93	774,82	**6,15**
Cones	172,36	4863,40	783,35	**6,21**
Average	174,15	4333,95	673,58	**6,45**

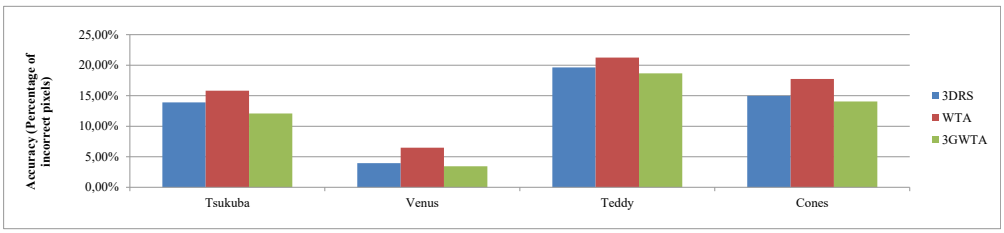
The computational time shown in Table 6.9 greatly favors the hybrid method in comparison to the classic WTA algorithm. The 3DRS execution time, although an order of magnitude faster, is provided for reference as it provides only coarse results.

6.2.4 3DRS-guided DP Experimental Results

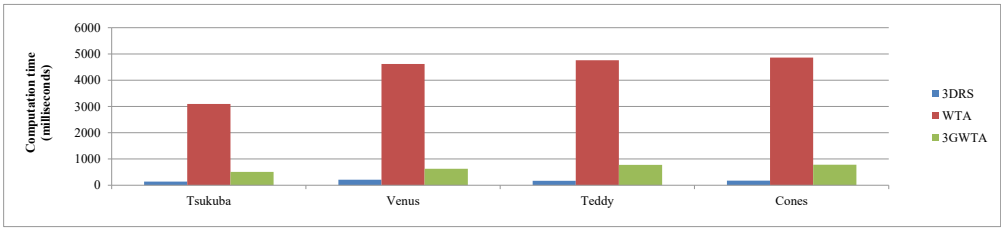
Table 6.10: Accuracy of generated disparities for the 3GDP method, expressed as a percentage of incorrect pixels (lower is better); Non-occ - non-occluded regions only; All - all pixels; Disc - discontinuity regions only.

Set	*3DRS*			*DP*			*3GDP*		
	Non-occ	**All**	**Disc**	**Non-occ**	**All**	**Disc**	**Non-occ**	**All**	**Disc**
Tsukuba	16,57%	17,94%	32,79%	5,91%	7,27%	23,29%	6,02%	7,38%	23,29%
Venus	3,12%	4,43%	23,99%	2,89%	4,18%	15,09%	3,45%	4,70%	15,05%
Teddy	14,56%	21,88%	33,51%	8,77%	16,61%	20,09%	8,98%	17,32%	21,11%
Cones	9,16%	17,90%	25,57%	5,22%	13,84%	13,58%	4,84%	13,98%	13,79%
Average		18,45%			11,40%			**11,66%**	

The DP implementation follows the algorithm described in Section 4.3.1. As the computation of each scanline is individual, this introduces scanline inconsistencies visible as streaking artifacts. One of the noted approaches [24] is to reuse the costs and computed path from a



(a) Accuracy for the computed disparities - data from Table 6.8.



(b) Method execution times - data from Table 6.9.

Figure 6.12: Comparison of accuracy for the computed disparities and method execution times for the images of the Middlebury set on 3DRS, pure WTA, and 3DRS-Guided WTA methods. Lower is better.

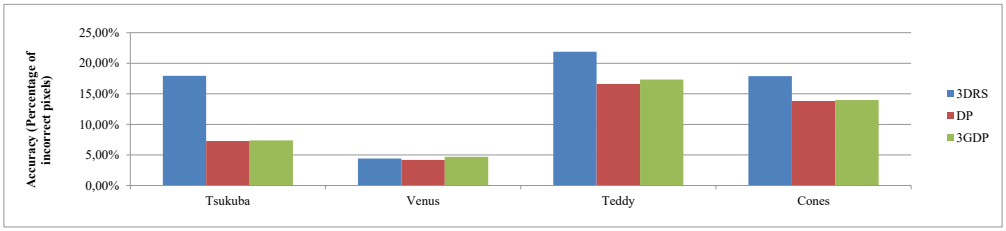
Table 6.11: Comparison of execution times for the 3DRS, DP and 3GDP method. Times are given in milliseconds. The speed-up ratio is given as a dimensionless quantity.

Set	t_{3DRS}	t_{DP}	t_{3GDP}	$\frac{t_{DP}}{t_{3GDP}}$
Tsukuba	55,29	851,74	135,50	**6,29**
Venus	84,45	1364,03	191,65	**7,12**
Teddy	81,01	1379,87	240,79	**5,73**
Cones	82,26	1370,45	235,15	**5,83**
Average	75,75	1241,52	200,77	**6,24**

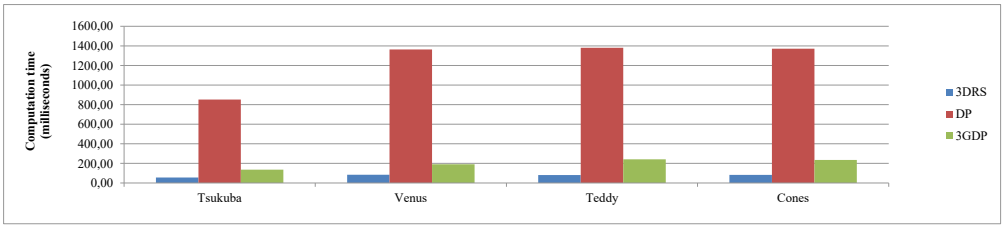
previous pass with an applied weighting factor, thus constraining the new path to roughly follow the previously computed path, contributing to vertical smoothness. The proposed approach therefore employs a similar vertical smoothing scheme, which adds a weighted cost from the previous calculated path to the cost of the current calculated path. It was found that satisfactory results are achieved with the weight parameter set to 180 (ranging from 0 to 255). The *OccCost* parameter was selected empirically based on the results. For the Census cost, it was observed that the value of $OccCost = 8$ performs well for all images. The reduced influence of the *OccCost* parameter on the final result is a result of employing the Census cost, which reduces the impact of radiometric differences.

For the R_{off} parameter used to constrain the DSI path, using a $R_{off} = 5$ has shown good results. There were no observable differences in quality for $R_{off} \geq 4$.

Based on these results, the final outputs of 3DRS, DP, and 3GDP are compared in terms of execution time and disparity map accuracy for both full image and non-occluded areas. Both



(a) Accuracy for the computed disparities - data from Table 6.10.



(b) Method execution times - data from Table 6.11.

Figure 6.13: Comparison of accuracy for the computed disparities and method execution times for the images of the Middlebury set on 3DRS, pure DP, and 3DRS-Guided DP. Lower is better.

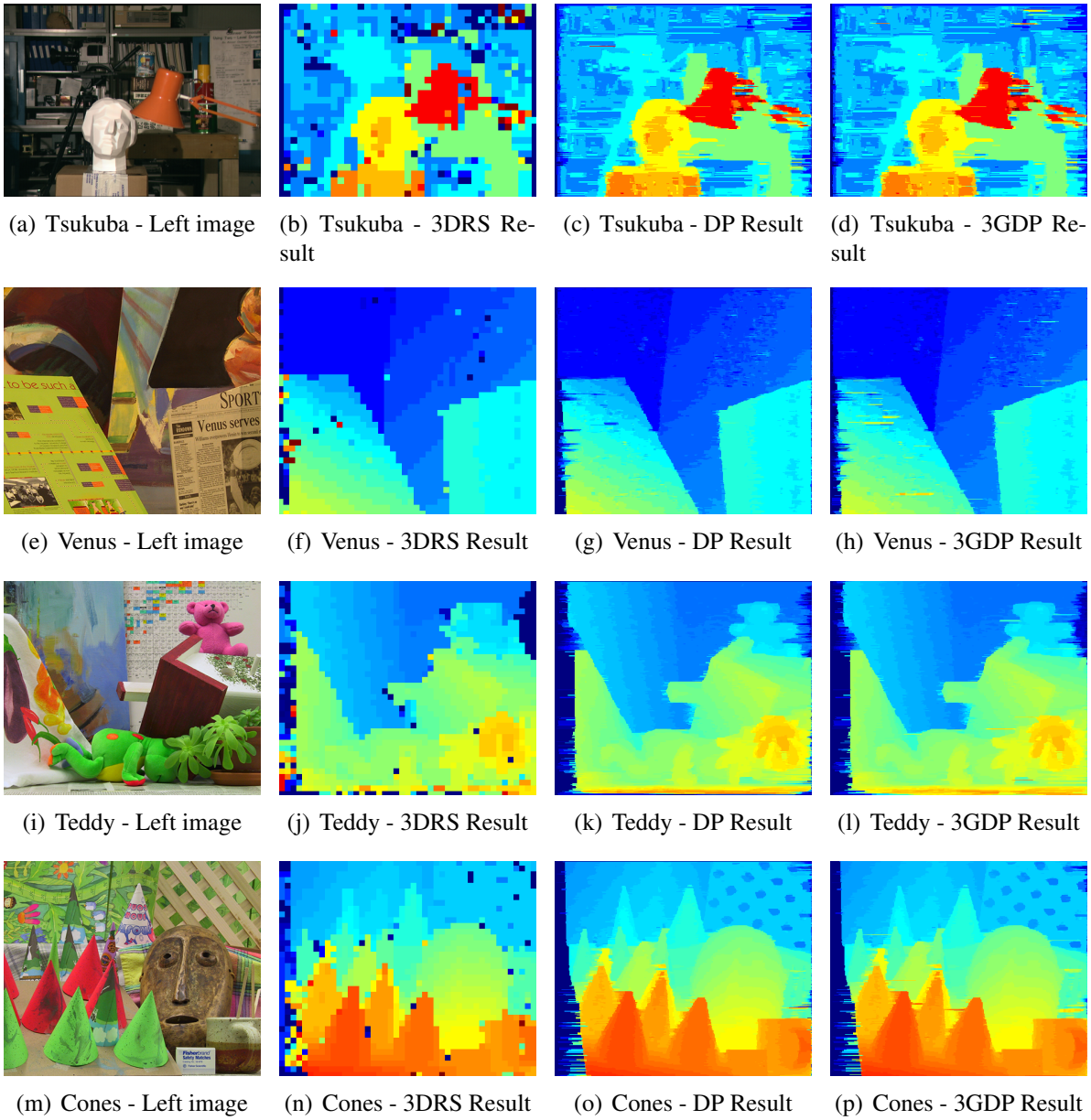
3DRS and DP passes employ the Census matching cost. The 3DRS method performs two passes in all cases (standalone and 3GDP). The measured accuracy of individual methods and the combined 3GDP method are provided in Table 6.10 and Figure 6.13 (a).

The execution time of methods is shown and compared in Table 6.11 and Figure 6.13 (b). The resultant disparity maps are shown in Figure 6.14.

6.3 Discussion

6.3.1 3DRS algorithm properties

As shown in the results (mostly in figure 6.6), the 3DRS algorithm can operate with constant time for practical block sizes (those yielding constant accuracy). As the 3DRS block size rises, the number of matching costs to calculate per vector increases. The actual number of pixels to be calculated for matching cost is the dominant component in those cases, but the overall number of the vectors in the image also decreases as less blocks cover the image, so the two motions balance out, yielding a constant execution time. For small block sizes (seen best in figure 6.3) an exponential increase of the execution time is observed for decreasing block sizes less than 4 pixels. This can be attributed to the exponential increase of the number of blocks and vectors to process, so that the overhead of processing the 3DRS algorithm steps becomes the dominant component in execution time. Nevertheless, for practical block sizes of 4 pixels and more (which also correlate with lowest measured error rates), the results suggest that the computation complexity of 3DRS does not change significantly with varying block sizes and is near-constant. It was also shown (figure 6.3 and 6.5) that the execution time of 3DRS, for all



(a) Tsukuba - Left image (b) Tsukuba - 3DRS Result (c) Tsukuba - DP Result (d) Tsukuba - 3GDP Result

(e) Venus - Left image (f) Venus - 3DRS Result (g) Venus - DP Result (h) Venus - 3GDP Result

(i) Teddy - Left image (j) Teddy - 3DRS Result (k) Teddy - DP Result (l) Teddy - 3GDP Result

(m) Cones - Left image (n) Cones - 3DRS Result (o) Cones - DP Result (p) Cones - 3GDP Result

Figure 6.14: Algorithm results - Left reference image and resultant disparities for 3DRS, Basic DP and 3DRS-Guided DP methods.

practical block sizes, depends solely on the input image resolution and the number of passes the 3DRS algorithm is applied on the image. Additionally, it is observed that the 3DRS execution time is far lesser than for the classical methods under the same conditions, and it grows linearly with the number of passes and image area (total number of pixels).

The convergence results in figure 6.4 show that a single 3DRS pass produces a disparity guidance with visibly lesser accuracy than after multiple passes, however after only two passes the accuracy improves to the point where subsequent passes do not further significantly improve it. These results match the expectations based on 3DRS theory, as the estimation starts from a zero-initialized state and requires several update cycles to reach a good estimate, which means

that a certain good estimate will be reached mid-image. As the vertical direction alternates between passes, the second pass will start from known good estimates and propagate them to the less good estimates from the first pass. After two meandering passes, it can be assumed that all vectors have a solid degree of confidence. Subsequent passes can improve the result further (although this is apparently image-dependent), but add constant time penalties. Varying the update vector set (as in figure 6.7) can marginally increase accuracy at the expense of the required number of passes, but can also introduce more noise, especially with increasing the update set. From the results, it appears that the maximum update value of 16 represents a sweet spot for the tested sample sets.

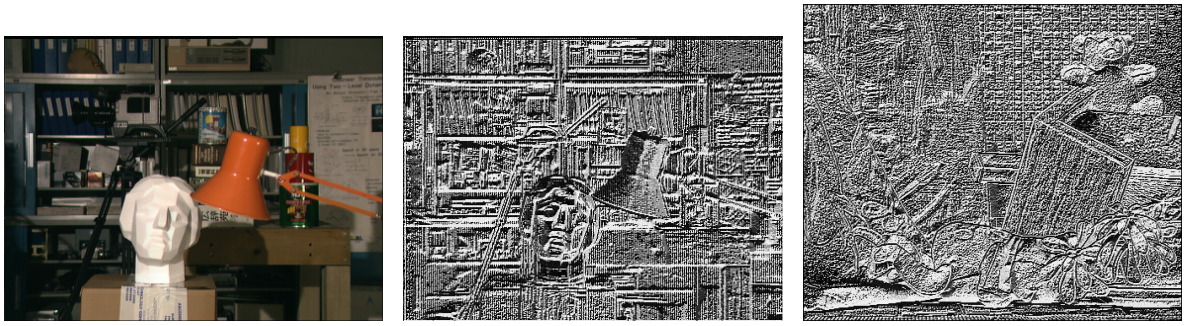
It is evident from the 3DRS estimation results that the disparity estimation obtained from 3DRS stabilizes and does not change significantly with further iterations after only two passes. Moreover, the disparity estimation enables the final accurate disparity computation via a dense method, independently from the input disparity range which is different for the tested pairs of images. In other words, unlike other conventional methods, the proposed methods do not require an explicit definition of the overall range of disparities within which the algorithm has to search for the optimum solution. Instead, the range is locally constrained with the 3DRS coarse result.

For these reasons, a two-pass 3DRS pre-estimation was selected as the coarse pre-computation step in the proposed hybrid methods.

6.3.2 Original methods and matching cost

The original methods (WTA and DP) have been evaluated for execution time dependent on the disparity range. It was shown in the results that there is a linear dependency between the disparity range and the execution time for WTA, and an almost linear, but slightly logarithmic dependency for dynamic programming. Nevertheless, the execution time grows almost proportionally to the estimated disparity range for both methods.

Several matching costs have also been evaluated with results shown in Table 6.5 and Figure 6.8. The results show that the Census cost provides the greatest matching accuracy, save for one case - the "Tsukuba" stereo pair where it actually degrades performance. An analysis was performed to determine the root cause and it has shown that the "Tsukuba" stereo pair exhibits slight, but noticeable vertical fixed pattern noise. This noise is effectively suppressed by other cost functions due to the slight difference in intensity which is also effectively averaged away by cost aggregation. However, for the Census cost, the encoding of the local intensity variation in respect to the individual pixel performs an effective amplification of the noise, as it can be seen in Figure 6.15. It can be concluded that the Census cost, although superior and suppressing radiometric differences, is sensitive to the presence fixed pattern noise in the image, which can present a problem with some cameras.



(a) Tsukuba - Left image (b) Tsukuba - Left Census tranform (c) Teddy - Left Census tranform

Figure 6.15: Census transform introducing noise into the "Tsukuba" image by amplifying a fixed pattern noise present in the original image but not immediately visible. The census transform for the "Teddy" image is shown for comparison.

6.3.3 3DRS-guided-Winner-take-all

Due to the application of the Census cost, the Winner-take-all results are significantly improved from the initial evaluation in [57], to the point where the accuracy of the results approaches the result of the Dynamic Programming method in some cases. Additionally the correlation of the range parameter r to the accuracy and the number of tested hypotheses (and consequently the execution time) was explored, where the results have shown (as it is shown in Figure 6.10 that significant reductions of search space (up to 99% of the Disparity space image volume) can be achieved without detrimental effects to the output quality, which is even improved by up to 1% by reducing the search space. This can be attributed to the effect that the constraining of the search space also constrains the range of outliers present in the occlusion areas, which decreases their overall effect on the output quality. The reduced search space also yields significant improvement in execution speed, with the hybrid 3DRS-WTA (3GWTA) algorithm improving the execution performance of the algorithm by 6,45 times on average, from times ranging up to five seconds to way below one second of execution, while consistently improving the error rates in all cases.

6.3.4 3DRS-guided-Dynamic programming results

As shown in Table 6.10, the pure DP method provides the overall best quality upon evaluation, by a small margin. However, the execution time is still significant and in ranges between 0,8 and 1,4 seconds in the reference implementation. It was shown that the 3GDP method produces very close results (off by on average 0,3% of the absolute score), but the execution time of 3GDP is 6,24 times faster on average. Therefore, with the hybrid 3GDP approach a more-than-six-fold speedup was obtained while maintaining the overall level of accuracy. The overall average disparity error of the proposed method is 11,66%, comparable with the results obtained by other scan-line DP-based approaches. As compared in Table 6.12, the 3DGP method obtains higher

accuracy than the basic DP method as benchmarked in [12], however, in DSI-based methods, it is still, in terms of quality, outperformed by SGM [25]. The execution speed, on average, exceeds the speed obtained by [58].

Table 6.12: Accuracy of generated disparities for the 3GDP method compared with other DP-based methods on the Middlebury stereo vision web site [22] (lower is better); Non-occ - non-occluded regions only; All - all pixels; Disc - discontinuity regions only.

Set	*DP[12]*			*SGM[25]*			*3GDP*		
	Non-occ	**All**	**Disc**	**Non-occ**	**All**	**Disc**	**Non-occ**	**All**	**Disc**
Tsukuba	4,12%	5,04%	12,0%	3,26%	3,96%	12,8%	6,02%	7,38%	23,29%
Venus	10,10%	11,00%	21,00%	1,00%	1,57%	11,30%	3,45%	4,70%	15,05%
Teddy	14,0%	21,60%	20,60%	6,02%	12,20%	16,30%	8,98%	17,32%	21,11%
Cones	10,50%	19,10%	21,10%	3,06%	9,75%	8,90%	4,84%	13,98%	13,79%
Average		13,93%			7,51%			**11,66%**	

6.3.5 Closing remarks

Although the proposed solutions do not always improve on the accuracy of other DSI-based algorithms, this can be attributed to several factors, such as the omission of the Disparity refinement step (as defined by the taxonomy of Scharstein and Szeliski [12]).

Comparing the results of the two method implementations in tables 6.8 and 6.10 , one can observe that the accuracy of the raw 3DRS estimation differs between the WTA and the DP hybrid methods, with the WTA estimation being more accurate. The cause for this is that the 3DRS algorithm was tuned differently for WTA and DP cases. For the WTA case, the 3DRS estimation was configured to extract a denser map with better quality, which significantly improved the WTA end-result, but the resultant 3DRS estimation time was considerably longer. For the 3GDP method, the additional precision of coarse disparity prediction was shown not to yield significant improvements of the disparity map quality, but did incur a substantial performance penalty. Therefore, for the DP case the 3DRS algorithm has been tuned to extract a coarser map in an approximately 2,5 times shorter period. This is a good example of how a 3DRS-based hybrid algorithm can be tuned for optimal performance.

Overall, a more-than-sixfold acceleration and memory footprint reduction for the WTA and DP methods can be singled out as the primary contributions of the proposed method to the state of the art, as it shows that the existing DSI based methods can be significantly accelerated by employing fast coarse true motion estimators, improving or retaining the quality of the disparity map.

Chapter 7

Conclusion

In this dissertation the stereo vision and 3D reconstruction problem was studied, with focus on the computation of stereo correspondences. Within the analyzed solution, several contributions can be isolated. An overview of the theoretical background was presented, laying the groundwork for the goal of reducing the computational complexity of stereo matching. Through the review of literature a considerable number of various solutions was identified, all of which have been shown to be computationally intensive. The main idea of this work was to employ a fast coarse block matcher, the three-dimensional recursive search algorithm which is commonly used in consumer electronics but not in computer vision, as a foundation for a hybrid solution which would outperform the standalone solutions. A classic winner-take-all local approach and a dynamic programming global approach was selected as a candidate for enhancement.
As the literature mostly covers the 3DRS algorithm from the perspective of motion estimation for motion picture enhancement in digital television, an analysis and evaluation of the 3DRS algorithm was conducted to examine its properties for computing disparities, with focus on its computational intensity and the robustness of the coarse disparity map. The measurements have shown that the 3DRS output is obtained in a constant time per frame which is directly proportional to the image dimensions, and is, for practical block sizes, invariant to the block size (and consequently the coarseness of the block grid), which is a useful property for a real-time application. Additionally, the convergence property of 3DRS was explored, where measurements have shown that two iterations produce a disparity map of near-maximum achievable quality, with further iterations offering little to no improvement or even worsening the disparity map quality in some cases. The two-iteration limit stems from the nature of convergence and the alternate-scan meandering, where the initial pass will converge mid-image, and the second scan will back-propagate the good estimates to the blocks where the algorithm had not yet converged in the initial pass. Finally, an enhancement to address the robustness of 3DRS for computer vision tasks was evaluated by implementing and evaluating different matching costs to address demanding imaging conditions. The cost function based on the Census transformation has

shown to yield the best quality of the disparity map.
In designing the hybrid method, the local approach was evaluated first, with the idea to reduce the disparity space which needs to be searched for the optimum disparity match. To obtain a high depth resolution it is desired to evaluate a large disparity range, which is computationally demanding. As the 3DRS convergence property was shown to be able to determine the correct disparity without an assumption of range, this property allows us to maximize the overall search range for the image while constraining it to a narrow range locally within the block grid of 3DRS estimation. An additional takeaway from the results is the requirement that the disparity space image should be connected with overlapping disparity ranges for neighboring blocks in the grid, in order to account for discontinuities in disparity naturally occuring in the occlusion areas. It was shown that re-applying the 3DRS assumptions and using the neighboring disparities as range predictors ensures that the disparity map retains the quality while reducing the overall number of hypotheses up to 99%.
Finally, a novel approach to stereo matching was proposed by combining a standard Dynamic Programming estimation with the 3DRS block-based motion estimator. The resultant method exhibits a nearly seven-fold increase in performance from the original DP method in the proposed implementation, while retaining the same level of quality. Overall, the use of 3DRS as a method of reducing and defining the DSI space has potential applications to other stereo matching methods, especially if large input disparity ranges are used. However, while the speed improvements have been substantial with multifold decreases in execution time, it is important to note that the improvements in accuracy have been mostly marginal or non-existent. Therefore additional effort should be invested in finding ways to additionally improve the quality of computed disparities. However, at minimum, it can be assumed that the new methods will retain the accuracy of the original methods, at increased speed.
Potential future work areas also lie in the exploration of other useful 3DRS properties, such as the natural ability of 3DRS to estimate 2D vectors, which might provide a guidance with inputs which do not exhibit perfect epipolar rectification. The future research should strive to further explore the hybrid methods aiming to improve the accuracy of the final result as well as further improve the execution speed, with the ultimate goal of defining an architecture for a real-time embedded hardware implementation.

Bibliography

[1] Huang, T., “Computer vision; evolution and promise”, in 19th CERN School of Computing, Vandoni, C., (ed.). CERN, 1996, pp. 21-25.

[2] Szeliski, R., Computer Vision: Algorithms and Applications, 1st ed. Berlin, Heidelberg: Springer-Verlag, 2010.

[3] Salvi, J., Fernandez, S., Pribanic, T., Llado, X., “A state of the art in structured light patterns for surface profilometry”, Pattern Recognition, Vol. 43, No. 8, 2010, pp. 2666 - 2680, dostupno na: http://www.sciencedirect.com/science/article/pii/S003132031000124X

[4] Barron, J., Fleet, D., Beauchemin, S., “Performance of optical flow techniques”, International Journal of Computer Vision, Vol. 12, No. 1, 1994, pp. 43-77, dostupno na: http://dx.doi.org/10.1007/BF01420984

[5] Braspenning, R. A., de Haan, G., “True-motion estimation using feature correspondences”, Vol. 5308, 2004, pp. 396-407, dostupno na: http://dx.doi.org/10.1117/12.525625

[6] de Haan, G., Biezen, P., Huijgen, H., Ojo, O., “True-motion estimation with 3-d recursive search block matching”, Circuits and Systems for Video Technology, IEEE Transactions on, Vol. 3, No. 5, Oct 1993, pp. 368-379, 388.

[7] de Haan, G., “Ic for motion-compensated de-interlacing, noise reduction, and picture-rate conversion”, Consumer Electronics, IEEE Transactions on, Vol. 45, No. 3, Aug 1999, pp. 617-624.

[8] Hendriks, E., Marosi, G., “Recursive disparity estimation algorithm for real time stereoscopic video applications”, in Image Processing, 1996. Proceedings., International Conference on, Vol. 1, Sep 1996, pp. 891-894 vol.2.

[9] Atzpadin, N., Kauff, P., Schreer, O., “Stereo analysis by hybrid recursive matching for real-time immersive video conferencing”, Circuits and Systems for Video Technology, IEEE Transactions on, Vol. 14, No. 3, March 2004, pp. 321-334.

[10] Hartley, R., Zisserman, A., Multiple View Geometry in Computer Vision, 2nd ed. New York, NY, USA: Cambridge University Press, 2003.

[11] Kaehler, A., Bradski, G., Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library, 1st ed. O'Reilly Media, Inc., 2016.

[12] Scharstein, D., Szeliski, R., "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms", Int. J. Comput. Vision, Vol. 47, No. 1-3, Apr. 2002, pp. 7–42, dostupno na: http://dx.doi.org/10.1023/A:1014573219977

[13] Hartley, R. I., "Theory and practice of projective rectification", International Journal of Computer Vision, Vol. 35, No. 2, Nov 1999, pp. 115–127, dostupno na: https://doi.org/10.1023/A:1008115206617

[14] Loop, C. T., Zhang, Z., "Computing rectifying homographies for stereo vision", Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149), Vol. 1, 1999, pp. 125-131 Vol. 1.

[15] Fusiello, A., Trucco, E., Verri, A., "A compact algorithm for rectification of stereo pairs", Machine Vision and Applications, Vol. 12, No. 1, Jul 2000, pp. 16–22, dostupno na: https://doi.org/10.1007/s001380050120

[16] Beric, A., de Haan, G., Sethuraman, R., van Meerbergen, J., "A technique for reducing complexity of recursive motion estimation algorithms", in 2003 IEEE Workshop on Signal Processing Systems (IEEE Cat. No.03TH8682), Aug 2003, pp. 195-200.

[17] Biswas, M., Nguyen, T., "A novel motion estimation algorithm using phase plane correlation for frame rate conversion", in Signals, Systems and Computers, 2002. Conference Record of the Thirty-Sixth Asilomar Conference on, Vol. 1, Nov 2002, pp. 492-496 vol.1.

[18] Slingerland, N. T., Smith, A. J., "Measuring the performance of multimedia instruction sets", IEEE Trans. Computers, Vol. 51, No. 11, 2002, pp. 1317–1332.

[19] Al-kadi, G., Hoogerbrugge, J., Guntur, S., Terechko, A., Duranton, M., Eerenberg, O., "Meandering based parallel 3drs algorithm for the multicore era", in Consumer Electronics (ICCE), 2010 Digest of Technical Papers International Conference on, Jan 2010, pp. 21-22.

[20] Kolmogorov, V., Zabih, R., "Computing visual correspondence with occlusions via graph cuts", in In International Conference on Computer Vision, 2001, pp. 508–515.

[21] Sun, J., Zheng, N.-N., Shum, H.-Y., "Stereo matching using belief propagation", IEEE Trans. Pattern Anal. Mach. Intell., Vol. 25, No. 7, Jul. 2003, pp. 787–800, dostupno na: http://dx.doi.org/10.1109/TPAMI.2003.1206509

[22] Scharstein, D., Szeliski, R., "Middlebury Stereo Vision Page", http://vision.middlebury.edu/stereo, [Online; accessed 31-Jan-2014]. 2014.

[23] Geiger, A., Roser, M., Urtasun, R., "Efficient large-scale stereo matching", in Proceedings of the 10th Asian Conference on Computer Vision - Volume Part I, ser. ACCV'10. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 25–38, dostupno na: http://dl.acm.org/citation.cfm?id=1964320.1964325

[24] Forstmann, S., Kanou, Y., Ohya, J., Thuering, S., Schmitt, A., "Real-time stereo by using dynamic programming", in Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on, June 2004, pp. 29-29.

[25] Hirschmuller, H., "Accurate and efficient stereo processing by semi-global matching and mutual information", in Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02, ser. CVPR '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 807–814, dostupno na: http://dx.doi.org/10.1109/CVPR.2005.56

[26] Ye, X., Li, J., Wang, H., Huang, H., Zhang, X., "Efficient stereo matching leveraging deep local and context information", IEEE Access, Vol. 5, 2017, pp. 18 745-18 755.

[27] Žbontar, J., LeCun, Y., "Computing the stereo matching cost with a convolutional neural network", in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015, pp. 1592-1599.

[28] Menze, M., Geiger, A., "Object scene flow for autonomous vehicles", in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2015. IEEE, Jun. 2015, pp. 3061–3070.

[29] Ladicky, L., Sturgess, P., Russell, C., Sengupta, S., Bastanlar, Y., Clocksin, W. F., Torr, P. H. S., "Joint optimization for object class segmentation and dense stereo reconstruction", International Journal of Computer Vision, Vol. 100, No. 2, 2012, pp. 122–133, dostupno na: https://doi.org/10.1007/s11263-011-0489-0

[30] Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T., "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation", CoRR, Vol. abs/1512.02134, 2015.

[31] Fischer, P., Dosovitskiy, A., Ilg, E., Häusser, P., Hazirbas, C., Golkov, V., van der Smagt, P., Cremers, D., Brox, T., "Flownet: Learning optical flow with convolutional networks", CoRR, Vol. abs/1504.06852, 2015.

[32] Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T., "Flownet 2.0: Evolution of optical flow estimation with deep networks", CoRR, Vol. abs/1612.01925, 2016.

[33] Pang, J., Sun, W., Ren, J. S. J., Yang, C., Yan, Q., "Cascade residual learning: A two-stage convolutional neural network for stereo matching", CoRR, Vol. abs/1708.09204, 2017.

[34] Pang, J., Sun, W., Yang, C., Ren, J. S. J., Xiao, R., Zeng, J., Lin, L., "Zoom and learn: Generalizing deep stereo matching to novel domains", CoRR, Vol. abs/1803.06641, 2018.

[35] Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., Bry, A., "End-to-end learning of geometry and context for deep stereo regression", CoRR, Vol. abs/1703.04309, 2017.

[36] Min, D., Lu, J., Do, M., "A revisit to cost aggregation in stereo matching: How far can we reduce its computational redundancy?", in Computer Vision (ICCV), 2011 IEEE International Conference on, Nov 2011, pp. 1567-1574.

[37] Tombari, F., Mattoccia, S., di Stefano, L., Addimanda, E., "Classification and evaluation of cost aggregation methods for stereo correspondence", 2008 IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1-8.

[38] Bobick, A. F., Intille, S. S., "Large occlusion stereo", International Journal of Computer Vision, Vol. 33, No. 3, 1999, pp. 181-200.

[39] Veksler, O., "Fast variable window for stereo correspondence using integral images", in Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on, Vol. 1, June 2003, pp. I-556-I-561 vol.1.

[40] Kuk-Jin Yoon, In So Kweon, "Adaptive support-weight approach for correspondence search", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 28, No. 4, April 2006, pp. 650-656.

[41] Tombari, F., Mattoccia, S., di Stefano, L., "Segmentation-based adaptive support for accurate stereo correspondence", in PSIVT, 2007.

[42] Ohta, Y., Kanade, T., "Stereo by Intra- and Inter-scanline Search Using Dynamic Programming", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 7, No. 2, Mar. 1985, pp. 139–154, dostupno na: http://dx.doi.org/10.1109/tpami.1985.4767639

[43] Cox, I. J., Hingorani, S. L., Rao, S. B., Maggs, B. M., "A maximum likelihood stereo algorithm", Computer Vision and Image Understanding, Vol. 63, 1996, pp. 542–567.

[44] Kim, J. C., Lee, K.-M., Choi, B.-T., Lee, S.-U., "A dense stereo matching using two-pass dynamic programming with generalized ground control points", in Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, Vol. 2, June 2005, pp. 1075-1082 vol. 2.

[45] Veksler, O., "Stereo correspondence by dynamic programming on a tree", in Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, Vol. 2, June 2005, pp. 384-390 vol. 2.

[46] Bleyer, M., Gelautz, M., "Simple but effective tree structures for dynamic programming-based stereo matching", in International Conference on Computer Vision Theory and Applications (VISAPP), 2008, pp. 415–422, vortrag: International Conference on Computer Vision Theory and Applications (VISAPP 2008), Funchal, Madeira - Portugal; 2008-01-22 – 2008-01-25, dostupno na: http://publik.tuwien.ac.at/files/PubDat_169059.pdf

[47] Congote, J., Barandiaran, J., Barandiaran, I., Ruiz, O., "Realtime Dense Stereo Matching with Dynamic Programming in CUDA", Andujar, C., Lluch, J., (ed.). San Sebastian, Spain: Eurographics Association, 2009, pp. 231-234, dostupno na: http://diglib.eg.org/EG/DL/LocalChapterEvents/CEIG/CEIG09/231-234.pdf

[48] Cai, J., "Fast stereo matching: Coarser to finer with selective updating", in Image and Vision Computing New Zealand 2007. Hamilton, New Zealand: Image and Vision Computing New Zealand, 2007, pp. 266–270, for more information, please refer to the conference website (see hypertext link) or contact the author., dostupno na: http://eprints.qut.edu.au/11228/

[49] Chang, X., Zhou, Z., Wang, L., Shi, Y., Zhao, Q., "Real-time accurate stereo matching using modified two-pass aggregation and winner-take-all guided dynamic programming", in 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on, May 2011, pp. 73-79.

[50] Wang, L., Liao, M., Gong, M., Yang, R., Nistér, D., "High-quality real-time stereo using adaptive cost aggregation and dynamic programming", in 3rd International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT 2006), 14-16 June 2006, Chapel Hill, North Carolina, USA, 2006, pp. 798–805, dostupno na: http://dx.doi.org/10.1109/3DPVT.2006.75

[51] Hirschmuller, H., Scharstein, D., “Evaluation of cost functions for stereo matching”, in Computer Vision and Pattern Recognition, 2007. CVPR ’07. IEEE Conference on, June 2007, pp. 1-8.

[52] Hirschmuller, H., Scharstein, D., “Evaluation of stereo matching costs on images with radiometric differences”, Pattern Analysis and Machine Intelligence, IEEE Transactions on, Vol. 31, No. 9, Sept 2009, pp. 1582-1599.

[53] Zabih, R., Woodfill, J., “Non-parametric local transforms for computing visual correspondence”, in Proceedings of the Third European Conference on Computer Vision (Vol. II), ser. ECCV ’94. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1994, pp. 151–158, dostupno na: http://dl.acm.org/citation.cfm?id=200241.200258

[54] Rožić, M., Pribanić, T., “Depth sensing with disparity space images and three-dimensional recursive search”, Automatika, Vol. 59, No. 2, 2018, pp. 131-142, dostupno na: https://doi.org/10.1080/00051144.2018.1503137

[55] Scharstein, D., Hirschmüller, H., Kitajima, Y., Krathwohl, G., Nesic, N., Wang, X., Westling, P., “High-resolution stereo datasets with subpixel-accurate ground truth.”, in GCPR, ser. Lecture Notes in Computer Science, Jiang, X., Hornegger, J., Koch, R., (ed.), Vol. 8753. Springer, 2014, pp. 31-42, dostupno na: http://dblp.uni-trier.de/db/conf/dagm/gcpr2014.html#ScharsteinHKKNWW14

[56] Heinrich, A., Bartels, C., van der Vleuten, R., Cordes, C., de Haan, G., “Optimization of hierarchical 3drs motion estimators for picture rate conversion”, Selected Topics in Signal Processing, IEEE Journal of, Vol. 5, No. 2, April 2011, pp. 262-274.

[57] Rožić, M., Đonlić, M., Pribanić, T., “A hybrid local method for stereo correspondence based on three-dimensional recursive search”, in MIPRO 2015 Conference Proceedings, August 2015, pp. 131-142.

[58] Salmen, J., Schlipsing, M., Edelbrunner, J., Hegemann, S., Lüke, S., “Real-time stereo vision: Making more out of dynamic programming”, in Computer Analysis of Images and Patterns, 13th International Conference, CAIP 2009, Münster, Germany, September 2-4, 2009. Proceedings, 2009, pp. 1096–1103, dostupno na: http://dx.doi.org/10.1007/978-3-642-03767-2_133

Biography

Miroslav Rožić was born in Zagreb on June 25th, 1982. He received his dipl.ing. degree in Computing from the Faculty of Electrical Engineering and Computing at the University of Zagreb, Croatia, in 2005. From 2006 to 2010 he was an FPGA Design Engineer and FPGA Development Lead at Intesis d.o.o. From 2010 to 2018 he worked as a Technical Director at Mikroprojekt d.o.o. During this time his primary focus was research and development in the field of image processing, camera technology and machine vision based on programmable logic devices. In 2018 he worked as the head of document management system application development at Zagrebačka Banka. From March 2019 he is a Senior Computer Vision engineer at Gideon Brothers, with focus on development of stereo vision systems for advanced visual perception. As a part of his scientific work he authored or co-authored five conference or peer-reviewed journal papers in international journal or conference proceedings. During his time at Mikroprojekt he presented at several international professional exhibitions and conferences, as well as participated in a joint project with the University of Split - "STRIPMed - Jačanje kapaciteta Sveučilišta u Splitu za istraživanje, razvoj i inovacije u području medicinske neuroelektronike" contributing in the field of computer vision. He is a member of IEEE.

Publications

Journal papers

1. Rožić, M., Pribanić, T., “Depth Sensing with disparity space images and three-dimensional recursive search ”, Automatika, Vol. 59, No. 2 , Aug. 2018., pg. 131-142.

Conference papers

1. Rožić, M., Đonlić, M., Pribanić, T., “A Hybrid Local Method for Stereo Correspondence based on Three-Dimensional Recursive Search”, MIPRO 2015 Conference Proceedings, 2015., pg. 155-160.
2. Mijatović, L., Dean, H., Rožić, M. “Implementation of algorithm for detection and correction of defective pixels in FPGA”, MIPRO 2012 Conference Proceedings, 2012., pg.

1731-1735.

3. Rožić, M., Rokov, J., Mlinarić, H. “Implementation of a RISC Computer Architecture in Programmable Logic”, MIPRO 2008 Conference Proceedings, 2008., pg. 208-213.
4. Cifrek, M., Mrvoš, S., Žufić, P., Rožić, M. “An Electronic Version of Fitts Tapping Task”, Abstracts 7th Alps-Adria Conference in Psychology, 2005., pg. 43-44.

Životopis

Miroslav Rožić je rođen 25.06.1982. u Zagrebu. Diplomirao je na fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu 2005. godine, smjer računarstvo. Od 2006. do 2010. radio je u tvrtci Intesis d.o.o. prvo kao dizajner FPGA sklopova, zatim kao vodeći FPGA razvojni inženjer. Od 2010. do 2018. Radio je u Mikroprojekt d.o.o. kao tehnički direktor. Tokom ovog razdoblja fokus mu je bilo istraživanje i razvoj u području obrade slike, digitalnih kamera i računalnog vida temeljenih na programirljivim logičkim sklopovima. Tokom 2018. godine je radio u Zagrebačkoj Banci kao voditelj aplikativnog razvoja sustava za upravljanje dokumentacijom. Od ožujka 2019. je Senior Computer Vision Engineer u tvrtci Gideon Brothers, sa fokusom na razvoj sustava stereo vida za naprednu vizualnu percepciju. U sklopu svog znanstvenog rada kao autor ili koautor objavio je pet radova na konferencijama ili u časopisu s međunarodnom recenzijom. Sudjelovao je ili prezentirao na više stručnih konferencija i sajmova, a za vrijeme rada u Mikroprojekt d.o.o. sudjelovao je na projektu "STRIPMed - Jačanje kapaciteta Sveučilišta u Splitu za istraživanje, razvoj i inovacije u području medicinske neuroelektronike" doprinoseći u temama računalnog vida. Član je strukovne udruge IEEE.