OULU 2015
C 536

ACTA UNIVERSITATIS OULUENSIS

C
TECHNICA

Daniel Herrera Castro

FROM IMAGES TO POINT CLOUDS

PRACTICAL CONSIDERATIONS FOR THREE-DIMENSIONAL COMPUTER VISION

UNIVERSITY OF OULU GRADUATE SCHOOL;
UNIVERSITY OF OULU,
FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING,
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ACTA UNIVERSITATIS OULUENSIS
C Technica 536

DANIEL HERRERA CASTRO

FROM IMAGES TO POINT CLOUDS

Practical considerations for three-dimensional computer vision

Academic dissertation to be presented, with the assent of the Doctoral Training Committee of Technology and Natural Sciences of the University of Oulu, for public defence in the Wetteri auditorium (IT115), Linnanmaa, on 14 August 2015, at 12 noon

UNIVERSITY OF OULU, OULU 2015

Acta Univ. Oul. C 536, 2015

Supervised by
Professor Janne Heikkilä

Reviewed by
Doctor Yasutaka Furukawa
Doctor Angel D. Sappa

Opponent
Doctor Ville Kyrki

ISBN 978-952-62-0852-7 (Paperback)
ISBN 978-952-62-0853-4 (PDF)

ISSN 0355-3213 (Printed)
ISSN 1796-2226 (Online)

Cover Design
Raimo Ahonen

JUVENES PRINT
TAMPERE 2015

Herrera Castro, Daniel, From images to point clouds. Practical considerations for three-dimensional computer vision
University of Oulu Graduate School; University of Oulu, Faculty of Information Technology and Electrical Engineering, Department of Computer Science and Engineering
Acta Univ. Oul. C 536, 2015
University of Oulu, P.O. Box 8000, FI-90014 University of Oulu, Finland

Abstract

Three-dimensional scene reconstruction has been an important area of research for many decades. It has a myriad of applications ranging from entertainment to medicine. This thesis explores the 3D reconstruction pipeline and proposes novel methods to improve many of the steps necessary to achieve a high quality reconstruction. It proposes novel methods in the areas of depth sensor calibration, simultaneous localization and mapping, depth map inpainting, point cloud simplification, and free-viewpoint rendering.

Geometric camera calibration is necessary in every 3D reconstruction pipeline. This thesis focuses on the calibration of depth sensors. It presents a review of sensors models and how they can be calibrated. It then examines the case of the well-known Kinect sensor and proposes a novel calibration method using only planar targets.

Reconstructing a scene using only color cameras entails di_erent challenges than when using depth sensors. Moreover, online applications require real-time response and must update the model as new frames are received. The thesis looks at these challenges and presents a novel simultaneous localization and mapping system using only color cameras. It adaptively triangulates points based on the detected baseline while still utilizing non-triangulated features for pose estimation.

The thesis addresses the extrapolating missing information in depth maps. It presents three novel methods for depth map inpainting. The first utilizes random sampling to fit planes in the missing regions. The second method utilizes a 2nd-order prior aligned with intensity edges. The third method learns natural filters to apply a Markov random field on a joint intensity and depth prior.

This thesis also looks at the issue of reducing the quantity of 3D information to a manageable size. It looks at how to merge depth maps from multiple views without storing redundant information. It presents a method to discard this redundant information while still maintaining the naturally variable resolution.

Finally, transparency estimation is examined in the context of free-viewpoint rendering. A procedure to estimate transparency maps for the foreground layers of a multi-view scene is presented. The results obtained reinforce the need for a high accuracy 3D reconstruction pipeline including all the previously presented steps.

Keywords: 3D reconstruction, camera calibration, depth map inpainting, free viewpoint rendering, point cloud merging, simultaneous localization and mapping

Herrera Castro, Daniel, Kuvista pistepilveksi. Käytännön näkökulmia kolmiulotteiseen tietokonenäköön
Oulun yliopiston tutkijakoulu; Oulun yliopisto, Tieto- ja sähkötekniikan tiedekunta, Tietotekniikan osasto
Acta Univ. Oul. C 536, 2015
Oulun yliopisto, PL 8000, 90014 Oulun yliopisto

Tiivistelmä

Kolmiuloitteisen ympäristöä kuvaavan mallin rakentaminen on ollut tärkeä tutkimuksen kohde jo usean vuosikymmenen ajan. Sen sovelluskohteet ulottuvat aina lääketieteestä viihdeteollisuuteen. Väitöskirja tarkastelee 3D ympäristöä kuvaavan mallin tuottamisprosessia ja esittää uusia keinoja parantaa korkealaatuisen rekonstruktion tuottamiseen vaadittavia vaiheita. Työssä esitetään uusia menetelmiä etäisyyssensoreiden kalibrointiin, samanaikaisesti tapahtuvaan paikannukseen ja kartoitukseen, syvyyskartan korjaamiseen, etäisyyspistepilven yksinkertaistamiseen ja vapaan katselukulman kuvantamiseen.

Väitöskirjan ensi osa keskittyy etäisyyssensoreiden kalibrointiin. Työ esittelee erilaisia sensorimalleja ja niiden kalibrointia. Yleisen tarkastelun lisäksi keskitytään hyvin tunnetun Kinect-sensorin käyttämiseen, ja ehdotetaan uutta kalibrointitapaa pelkkiä tasokohteita hyväksikäyttäen. Pelkkien värikameroiden käyttäminen näkymän rekonstruointiin tuottaa erilaisia haasteita verrattuna etäisyyssensoreiden käyttöön kuvan muodostamisessa. Lisäksi verkkosovellukset vaativat reaaliaikaista vastetta. Väitös tarkastelee kyseisiä haasteita ja esittää uudenlaisen yhtäaikaisen paikannuksen ja kartoituksen mallin tuottamista pelkkiä värikameroita käyttämällä. Esitetty tapa kolmiomittaa adaptiivisesti pisteitä taustan pohjalta samalla kun hyödynnetään eikolmiomitattuja piirteitä asentotietoihin.

Työssä esitellään kolme uudenlaista tapaa syvyyskartan korjaamiseen. Ensimmäinen tapa käyttää satunnaispisteitä tasojen kohdentamiseen puuttuvilla alueilla. Toinen tapa käyttää 2nd-order prior kohdistusta ja intensiteettireunoja. Kolmas tapa oppii filttereitä joita se soveltaa Markov satunnaiskenttiin yhteisillä tiheys ja syvyys ennakoinneilla. Tämä väitös selvittää myös mahdollisuuksia 3D-information määrän pienentämiseen käsiteltävälle tasolle. Työssä selvitetään, kuinka syvyyskarttoja voidaan yhdistää ilman päällekkäisen informaation tallentamista. Työssä esitetään tapa jolla päällekkäisestä datasta voidaan luopua kuitenkin säilyttäen luonnollisesti muuttuva resoluutio.

Viimeksi, tutkimuksessa on esitetty läpinäkyvyyskarttojen arviointiproseduuri etualan kerroksien monikatselukulmanäkymissä vapaan katselukulman renderöinnin näkökulmasta. Saadut tulokset vahvistavat tarkan 3D-näkymän rakentamisliukuhihnan tarvetta sisältäen kaikki edellä mainitut vaiheet.

Asiasanat: 3D kuvanmuodostus, kameran kalibrointi, pistepilvien yhdistäminen, syvyyskartan korjaaminen, vapaa katselukulmapohjainen renderöinti

Preface

A doctoral dissertation is a symbolic conclusion to a long journey into the world of research. This is especially the case with a compilation thesis, like this one, where the previous articles speak for themselves. They show parts of the progression along this journey and the contributions made. But there are parts of this journey that are not reflected in the publications or the dissertation though they were very important.

My journey into computer vision started before I chose to do so. I was deeply inspired during my bachelor studies by Dr. Pablo Alvarado, a.k.a. *el Doc*. The first, and for a long time the only, person I met with a doctoral degree. He demonstrated that knowledge, passion, and humility can go hand in hand. He introduced me to the computer vision field and opened a door for me that I did not know existed, and for that I am very grateful.

The doctoral journey was, like Finland, a somewhat lonely and exotic thing for me. The people who helped me along the way were like shining stars that turned an otherwise dark and scary night into a beautiful starry landscape. I would like to thank them all here for everything they have done for me. Unfortunately, there is not enough space in this book to list all the stars in the sky. And so, even though some may shine brighter than others, know that I'm truly thankful to all for lighting my way.

Perhaps, the brightest star has been my mentor, Dr. Juho Kannala. It's an inspiration to work with someone who speaks math better than English, even though his English is exceptional. He has been an excellent and patient guide throughout my journey, as well as a role model of what kind of doctor I would like to be. I'm convinced that I would be a worse researcher if I had not had him to check and challenge me. Thank you.

I'm also thankful to my Professor Janne Heikkilä who believed in me, even when I struggled, and provided a lot of valuable support. When I was too short-sighted to see only the tree, he always reminded me of the forest.

I'd like to thank the NVIDIA research team for their help and collaboration. In particular, Dr. Kihwan Kim, who went beyond the supervisor role and shared his friendship with me.

A journey of only academics, equations, and code would be bland and tasteless. I'm immensely grateful to those who added some spice and color to it. Miguel Bordallo and Matteo Pedone made the department a much warmer place for me. Special thanks to

Hannakaisa Aikio, who went beyond her duties to help me get established in Oulu. She has been a valuable friend both at work and outside.

Perhaps, one of the most important aspects of my journey has been lunch. Not only because it staves off the ever lurking hunger, but because it has kept my human mind sane in the midst of this computer world. I thank my early lunch group with Ville Ojansivu, Sami Huttunen, and Jari Hannuksela. The beginning was so much brighter because of your company. I cannot thank my new lunch group enough. Your charisma and friendship turns a simple lunch into an exotic discussion surrounded with dwarves, rangers, witches, and elves.

Finally, I would like to thank my family for their love and support. Though the distance may be large, it is comforting to know that there is always a place where I belong.

Oulu, March 2015 Daniel Herrera C.

Abbreviations

$\lvert\cdot\rvert$	Absolute value
$\lVert\cdot\rVert$	L^2-norm
2D	Two-dimensional
3D	Three-dimensional
α	Interpolation or transparency factor
$\boldsymbol{\alpha}$	Expert function parameters for a FoE formulation
$\mathbf{b}$	Background color vector
$\mathbf{c}$	Color vector
c	The speed of light
c_0, c_1	Kinect disparity to depth coefficients
$\mathcal{D}(\mathbf{x}_n)$	Camera distortion function
DIBR	Depth-image based rendering
DoF	Degrees of freedom
$\mathbf{f}$	Foreground color vector
f_x, f_y	Focal length parameters
FoE	Field-of-Experts
FVR	Free-viewpoint rendering
GPU	Graphics processing unit
$\mathcal{I}$	Image plane
$\mathbf{I}$	Image matrix
$\hat{\mathbf{I}}$	Warped image matrix
$\mathbf{j}$	Vector of linear filter coefficients
$\mathcal{K}(\mathbf{x}_n)$	Intrinsic projection function
$\mathbf{K}$	Intrinsic matrix of a pinhole camera
$k_1, k_2, \ldots, k_n$	Radial distortion coefficients
kdu	Kinect disparity units
λ	Relative weighting factor between data and regularization terms
$L(x,y,z,\theta,\phi)$	The plenoptic function.
$\mathbf{m}$	2D coordinates of a point on the image plane
MRF	Markov random field
ν	The frequency of light

$\hat{v}(\mathbf{x})$	Perspective division function
$\mathcal{N}$	Set of neighbourhoods in a Markov random field
$\phi(x;\alpha)$	Expert function for a FoE formulation
$\mathbb{P}^d$	Projective space of dimension d
$\mathcal{P}(\mathbf{x})$	Camera projection function
$\mathbf{P}$	Point cloud
$p(\cdot)$	Probability density function
$p_1, p_2, ..., p_N$	Tangential distortion coefficients or 3D point samples from a surface
QPBO	Quadratic pseudo-Boolean optimization
$\rho(x)$	Robust cost function
$\mathbb{R}^d$	Real space of dimension d
$\mathcal{R}$	Rigid 3D transformation
$\mathbf{R}$	3D rotation matrix
$R(\mathbf{x})$	Regularization prior term
RANSAC	Random sample consensus
SDF	Signed distance function
SLAM	Simultaneous localization and mapping
SVD	Singular value decomposition
$\mathbf{\Theta}$	Parameters of a probability distribution
τ	Threshold value
$\mathbf{t}$	3D translation vector
ToF	Time-of-flight
$U(x)$	Unary energy data term
(u_0, v_0)	Principal point of the camera
$\mathcal{V}$	Virtual image plane or the set of pixels with missing values
W	Weighting function
$\mathbf{x}$	3D coordinates of a point in Euclidean space
$\mathbf{x}_n$	2D coordinates of a point on the virtual image plane
$\mathbf{x}_{(\mathbf{k})}$	A vector containing the values of the pixel in neighborhood $\mathbf{k}$
$Z(\mathbf{\Theta})$	Partition function that normalizes a probability distribution

List of original publications

This thesis is based on the following articles, which are referred to in the text by their Roman numerals (I–VII):

I Herrera C. D, Kannala J & Heikkilä J (2012) Joint depth and color camera calibration with distortion correction. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(10): 2058–2064.

II Herrera C. D, Kim K, Kannala J, Pulli K & Heikkilä J (2014) DT-SLAM: Deferred Triangulation for Robust SLAM. International Conference on 3D Vision (3DV).

III Herrera C. D, Kannala J, Ladický L & Heikkilä J (2013) Depth map inpainting under a second-order smoothness prior. Proc Scandinavian Conference on Image Analysis (SCIA). Lecture Notes on Computer Science 7944: 555–566.

IV Herrera C. D, Kannala J, Sturm P & Heikkilä J (2013) A Learned Joint Depth and Intensity Prior using Markov Random Fields. International Conference on 3D Vision (3DV) 1: 17–24.

V Herrera C. D, Kannala J & Heikkilä J (2011) Generating Dense Depth Maps Using a Patch Cloud and Local Planar Surface Models. 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON): 1-4.

VI Kyöstilä T, Herrera C. D, Kannala J & Heikkilä J (2013) Merging overlapping depth maps into a nonredundant point cloud. Proc Scandinavian Conference on Image Analysis (SCIA). Lecture Notes on Computer Science 7944: 567–578.

VII Herrera C. D, Kannala J & Heikkilä J (2011) Multi-View Alpha Matte for Free Viewpoint Rendering. International Conference on Computer Vision / Computer Graphics Collaboration Techniques and Applications (Mirage). Lecture Notes on Computer Science 6930: 98–109.

The main responsibility for preparing articles I–V and VII was carried by the author of the dissertation. This includes the design of the algorithm, implementation, testing, and writing of the paper. The co-authors of these papers provided guidance and suggestions for the algorithm and experiments.

Paper II was a close collaboration between the University of Oulu and NVIDIA Corporation. Half of the work was done by the author under supervision of Dr. Kim and Dr. Pulli at NVIDIA. The other half was also done by the author under supervision of Dr. Kannala and Prof. Heikkilä at the University of Oulu. The author was responsible for the main ideas, implementation, experiments, and the vast majority of the writing.

Paper VI was implemented, tested, and written by Mr. Kyöstilä under supervision of the author. The algorithm and ideas were developed by the author with close collaboration with Dr. Kannala and Prof. Heikkilä.

Contents

1 Introduction

1.1 Background and motivation

The goal of computer vision is to extract useful information from images. It is a very attractive field of research because of the large amount of information contained in an image. Humans use visual information for a myriad of tasks. A single image can be used to recognize the identity of a person and his mood. Most fruits can be judged to be ripe or not by mere visual appearance. We can estimate the slipperiness of a surface or even reconstruct the structure of a building from images only. As a discipline, computer vision aims to give these very same abilities to computers.

Computer vision methods take many different forms depending on the information required and the imaged scene. Classification methods can assign global labels to an image, *e.g.* "*is scenery*", "*contains a person*", "*is happy*". Detection methods can locate the place in the image where an object is present, *e.g.* a car, a pedestrian. Registration methods determine the pose of a given object, *e.g.* face registration to determine the exact configuration of a face in an image. Reconstruction methods extract a model from noisy and incomplete data, *e.g.* three-dimensional reconstruction of a room from images. Each of these can be considered a separate area within computer vision and they often use very different techniques. In practice they also overlap and complement each other. For example, feature detection can be of great help for 3D reconstruction.

In a similar way, computer vision is not an isolated field. It borrows from some and contributes to others. Mathematics is the intrinsic language of computer vision. Problems and solutions are both expressed in mathematical terms. Due to the noisy nature of the world and its measurements algorithms are often probabilistic in nature and rely on concepts and methods from statistics. Geometry provides valuable insight and constraints that are used widely in computer vision and in this thesis. Machine learning plays a crucial role in many areas of computer vision. Many tasks, *e.g.* classification, can be solved by learning a function from the input data. The recent advances in deep convolutional neural networks mixed with image processing techniques have resulted in the most accurate recognition systems to date. The solutions and algorithms have to be implemented and tested which is a software engineering task. Also, computer graphics can be said to perform the inverse function to computer vision. Whereas computer vision tries to recover information from images, computer graphics uses models to generate images. Still, they share many common techniques since they both have images as a central concept.

So far, a generic solution that can extract and categorize all information available in an image of an arbitrary scene has remained out of reach. Nevertheless, impressive achievements have been made in many areas of computer vision. Face verification methods have surpassed human accuracy (Lu & Tang 2015). Image inpainting algorithms can replace foreground objects with an automatically generated yet realistic and convincing background (Roth & Black 2009). Three-dimensional reconstruction algorithms can produce city-wide 3D models from images directly obtained from an internet search (Agarwal *et al.* 2015a). A dense map of the environment can be reconstructed in real-time from an arbitrarily moving camera (Engel *et al.* 2014). Google's self-driving cars rely heavily on 3D reconstruction to estimate the world around them and have driven thousands of kilometres with no human assistance. Computer vision has seen a fast-paced and constant development in the last decade and continues to advance.

The advancements in computer vision come from many directions. Naturally, many of them come from basic research done within the field. A better understanding of epipolar and multi-view geometry has enabled better constraints for reconstruction (Hartley & Zisserman 2000, Hongdong & Hartley 2006, Ponce & Hebert 2014). The development of view-invariant methods for feature detection and description has significantly improved recognition, registration, and reconstruction algorithms (Lowe 2004, Heikkilä *et al.* 2009). Other fields have enabled many of the recent advancements. Hardware developments constantly increase the computational power available and enable new types of algorithms. Most of the top-performing algorithms would not have been possible ten years ago due to hardware constraints. The quantity and quality of the training data have had a tremendous impact on the quality of the algorithms. The recent advances in deep convolutional neural networks have been made possible by training the system with many thousands of labelled images. Such massive databases have only recently become available. New algorithms for numerical optimization enable computer vision systems to include more accurate constraints, more data, and better priors. New sensors have opened the way for more ambitious systems. Depth sensors produce a dense reconstruction of a scene with minimal computational cost and have enabled research on more advanced applications like autonomous navigation and large-scale dense mapping.

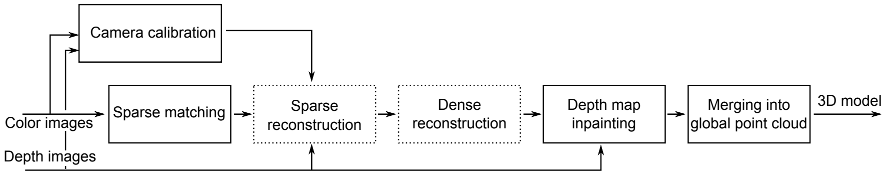


Fig 1. An example of a 3D reconstruction pipeline using depth maps and point clouds as the 3D model. The sparse and dense reconstruction stages can be avoided or simplified when using a depth sensor.

1.2 Scope of the thesis

Computer vision is a very broad field and a doctoral thesis can only cover a small part of it. This thesis is no exception and focuses on the area of 3D reconstruction. It aims to contribute knowledge and practical methods that can be used to improve reconstruction systems. The thesis describes the 3D reconstruction process as a pipeline that takes images as input and produces a 3D model as output. However, the exact stages of the pipeline and the structure of the output model are not fixed. There are many ways to approach the reconstruction problem. Instead of suggesting a concrete pipeline, this thesis proposes several improvements to different stages of it. As an example, Figure 1 shows a reconstruction pipeline that includes all the components mentioned in this thesis. The topics and problems considered in Papers I–VII may appear diverse, but they all support the idea of a reconstruction system. This section briefly describes these components and how they are related to the thesis.

Reconstructing the scene structure from images has been a central area of research since the beginnings of computer vision (Marr 1982, Faugeras 1993, Hartley & Zisserman 2000, Faugeras *et al.* 2001). The output model can be the goal itself or it can be the input to higher level applications like augmented reality or free viewpoint rendering. The reconstruction process itself is typically divided into sub-problems, as seen in Fig. 1. Camera calibration, feature detection and matching, sparse reconstruction, and all the other possible stages are complicated problems on their own and most are not considered solved yet. Impressive reconstruction systems have been built using the methods currently available for these components, but it is only natural to expect that improving the components will lead to even better systems.

An important part of the thesis deals with camera calibration. A calibrated camera is necessary to perform a metric reconstruction which is what we are after here. The contributions are centered around calibrating depth sensors, however, this is still closely related to the calibration of color images.

Naturally, the theory and geometry behind 3D reconstruction is central to this thesis. The contributions of the thesis focus on sparse reconstruction. Although dense depth maps are used in later parts of the thesis, dense reconstruction methods are out of scope. Efficient reconstruction methods that can be executed in real-time on a mobile device are more closely examined than offline batch processing methods.

Image inpainting may seem to be unrelated to 3D reconstruction. However, many applications require dense depth maps, whereas some systems produce only semi-dense reconstructions. Thus, depth map inpainting has also been explored in this thesis and several contributions have been made to this field.

The use of depth sensors simplifies some aspects of reconstruction systems but also brings new complications. For example, the high resolution and frame rate of new depth sensors makes it challenging to process and store all the information produced. This thesis approaches this problem by merging depth maps into a single point cloud, thus, the area of point cloud simplification and its theory are of interest to this work.

Free-viewpoint rendering is included in the scope of this thesis as an application of the reconstructed 3D model. In this context, matting (*i.e.* recovering the true color and transparency of a foreground object) is also explored. The thesis analyses the artefacts produced by transparency in free-viewpoint renderings and addresses them using matting techniques.

1.3 Contributions of the thesis

The main contributions of the thesis are listed below.

- An accurate, practical, and widely applicable calibration algorithm for depth and color camera pairs. It uses only a planar surface as the calibration target and avoids the need for detecting corners in the depth images.
- A distortion model that improves the reconstruction accuracy of the Kinect sensor.
- A publicly available Matlab toolbox that implements the calibration algorithm and the distortion model with a user interface to calibrate a Kinect sensor.
- An algorithm to efficiently utilize both, matches that have been triangulated and matches that have no depth, during camera pose estimation and bundle adjustment.
- A publicly available and open source implementation of a real-time SLAM system that utilizes this algorithm for pose estimation and bundle adjustment.

- An inpainting algorithm for depth maps that uses a second-order smoothness prior and graph cuts to fill missing areas.
- A publicly available Matlab toolbox that implements this inpainting algorithm using a second-order smoothness prior.
- An inpainting algorithm for aligned depth and intensity images using a learned Markov random field prior. The shape of the prior is learned from a database of *natural* images so that the inpainting produces visually plausible results that match the statistics of the natural images.
- An inpainting algorithm for depth maps that assumes piece-wise planar regions to quickly fill missing areas using the boundary pixels as cues.
- An algorithm for merging depth maps into a global non-redundant point cloud while retaining the input resolution.
- A multi-view matting algorithm to separate the observed colors into foreground and background layers with color and transparency information. This matting algorithm was then applied and tested in a free-viewpoint video application.

1.4 Summary of the original articles

This thesis is based on seven previously published articles. The contributions listed above were first published in these articles. The original articles are reprinted in the appendix and their content is summarized below.

Paper I presents an algorithm to jointly calibrate a depth and color camera pair. The method is very practical because it uses only a planar calibration target, thus, avoiding complicated 3D calibration targets. It avoids the detection of depth discontinuities as calibration constraints due to the well-known noise in depth images around edges. Instead, it formulates the calibration constraints based on the coplanarity of depth pixels away from the depth discontinuities and the rigid transformation between depth and color cameras. It also proposes a novel distortion model for the Kinect sensor (Latta 2010) and evaluates the accuracy of the resulting algorithm calibrating several Kinect sensors.

Paper II introduces a complete real-time SLAM (simultaneous localization and mapping) system. Traditionally, keyframe-based SLAM systems have used an initialization step were an initial map is computed from two frames with a good baseline. After this initialization step, only detected features that had been previously triangulated are used for pose estimation and bundle adjustment. The main contribution of Paper II is an algorithm that mixes triangulated and non-triangulated features in a cost function that is minimized for pose estimation and bundle adjustment. This removes the need for an explicit initialization stage, allows the system to be robust to purely rotational motions where no feature can be triangulated, and improves accuracy by using all available information.

Paper III describes an inpainting algorithm for depth maps that enforces a second-order smoothness prior. A second-order smoothness prior is better than traditional first-order priors since it is invariant to camera 3D rotation. However, it is also harder to optimize. The proposed algorithm uses graph cuts to optimize a cost function based on the triple cliques resulting from the prior. When available, an aligned intensity image is used to weight the effects of the prior and favour depth discontinuities at intensity edges.

Paper IV proposes an inpainting algorithm for an aligned intensity and depth image pair following the idea that higher-order priors are more flexible. The high-order prior is based on Markov random fields. The images are processed by a bank of filters and the result is evaluated with a series of expert functions. The optimal bank of filters and the parameters of the expert functions are automatically learned from a database of natural images. This produces a generative model of the statistics of natural depth and intensity image pairs. The model is applied to inpaint real depth maps produced by the Kinect sensor and to perform super-resolution of depth maps.

Paper V presents a simpler inpainting algorithm for depth maps that assumes piece-wise planar surfaces. Many reconstruction systems produce semi-dense point clouds, *e.g.* Furukawa & Ponce (2010). However, many applications require dense depth maps. This inpainting algorithm takes an oriented semi-dense 3D patch cloud as input and produces a dense depth map for a reference image. It uses random sample consensus (RANSAC) to find candidate planes in the point cloud. To find planes that are relevant to the missing areas, only pixels on the boundary of the missing area are used as seeds for RANSAC. Once a list of candidate planes is obtained, a graph-cut-based optimization labels the missing pixels.

Paper VI describes an algorithm that takes a series of depth maps and merges them into a single non-redundant point cloud. Merging point clouds in 3D is problematic because of the need to choose a suitable clustering distance threshold. The proposed algorithm uses the 2D grid imposed naturally by the depth map resolution to find candidates to merge. Points are described using their position and uncertainty and merging only happens if their statistical model suggests they might belong to the same physical point. Thus, preserving the maximum resolution obtained during scanning but removing redundancies.

Paper VII proposes a multi-view transparency estimation algorithm in the context of free-viewpoint rendering. Traditional reconstruction systems do not take transparency into account which produces visible artefacts during free-viewpoint rendering. The algorithm proposed takes several images of a scene with their corresponding depth maps and separates each into foreground and background layers with corresponding colors and transparency values. The algorithm uses ray-tracing to gather samples from physical points. From these samples, a series of linear constraints are assembled to estimate the foreground color, the background color, and the observed transparency. A final graph-cut optimization enforces smoothness on the transparency values. Finally, the algorithm is implemented and tested as a free-viewpoint rendering algorithm.

1.5 Outline of the thesis

This thesis consists of an overview and an appendix. The appendix contains the original articles described in the previous section. The rest of the overview is organized as follows. Chapter 2 deals with geometric camera calibration, giving an overview of camera models and calibration techniques. Chapter 3 discusses 3D reconstruction techniques. Chapter 4 covers image inpainting and the different types of priors used. Chapter 5 reviews the theory behind point cloud simplification and depth map merging. A concrete application of 3D reconstruction is analysed in Chapter 6, free-viewpoint rendering. Finally, Chapter 7 presents conclusions.

2 Geometric camera calibration

Geometric camera calibration is the process in which the geometric properties of a camera are estimated. Camera calibration is an essential step for any 3D reconstruction pipeline. Knowledge of the geometric properties of the camera is necessary to make any geometric measurements of the scene (*e.g.* angles, length ratios, *etc.*).

Through calibration, a mapping is established between the scene points and the image points. This mapping consists of a forward- and a back-projection. The forward-projection maps a 3D scene point to its corresponding image point. For traditional color cameras, the back-projection is a one-to-many mapping that maps a 2D image point to a set of 3D scene points (usually a collinear set of points that pass through the optical center). For depth sensors, however, the back-projection is exact and matches an image point to a single 3D scene point.

Sections 2.1 and 2.2 review the color camera and depth camera models, respectively. The chapter then explores some of the algorithms used for calibrating these models in Section 2.3. Finally, Section 2.3.3 discusses Paper I, one of the major contributions of the thesis. It presents an accurate, practical, and widely applicable calibration algorithm for depth and color camera pairs, and shows results of this algorithm applied to the popular Kinect sensor. The algorithm uses only planar targets and does not require the detection of depth discontinuities, which would introduce noise in the calibration. Moreover, a novel distortion model is proposed for the Kinect that achieves a better accuracy than the state of the art.

2.1 Color camera models

Camera calibration is a two part process which consists of first determining the camera model and then estimating the parameters of this model. Selecting the model is a compromise. Models with fewer parameters are more robust to noise but may not be flexible enough to accurately capture the mapping. On the other hand, models with many parameters are more flexible but may lead to over fitting. Oftentimes, the model is manually selected based on knowledge about the camera and its structure.

A camera can be seen as a collection of pixels. Each pixel has a 2D coordinate in image space and is associated with an optical ray in 3D space. A pixel images the closest point along this optical ray. A color camera images the color of this point, whereas a depth camera images the distance along the optical ray. From the geometric point of view, the forward-projection of a color camera takes the 3D position of the scene point and results in a 2D pixel position. Depth sensors augment this by mapping a 3D scene point to a 3D quantity where the first two components indicate the 2D pixel position and the last is a function of the distance to the sensor. Oftentimes, depth sensors have similar optics to color cameras, so that we can use the same models to obtain the mapping of the 2D pixel position. Thus, this section first reviews the most common color cameras models and the next section then examines how these are extended for depth sensors.

The most generic model for a color camera describes each pixel with an independent optical ray. Each optical ray is represented as a 3D half-line. This can effectively model any camera that captures light travelling in a straight line between the camera and the opaque surfaces of a scene. This generic model can be useful, and indeed required for some types of cameras (*e.g.* modelling several cameras as a single sensor (Grossberg & Nayar 2001), non-central mosaics captured under rotational motion (Swaminathan *et al.* 2003)). It is possible to model and calibrate regular cameras using this model (Ramalingam *et al.* 2005). However, very often constraints are used to regularize the optical ray orientation and reduce the number of parameters.

Axial models assume that all optical rays intersect at a single line in space called the camera axis. For example, this is useful to model crossed-slit cameras (Feldman *et al.* 2003, Gupta & Hartley 1997). A more common and more restrictive constraint is to assume that all rays pass through a single point in space. These are called *central* models (Kannala *et al.* 2008). The most common central model is the perspective camera model. This model is now described in detail because it serves as a base to the more complicated depth sensor models presented in the following section.

2.1.1 The perspective camera

The perspective camera, also known as the pinhole camera model, consists of a plane and a point, as seen in Fig. 2. The plane represents the image plane and the point is the optical center of the camera. The optical rays travel from the observed surfaces in a straight line passing through the optical center and intersecting the image plane.

The perspective camera images straight lines as straight lines on the image plane. This projection is a linear mapping between $\mathbb{P}^3$ and $\mathbb{P}^2$. Thus, mathematically, the model can be described compactly in homogeneous coordinates by a single multiplication with a 3×4 matrix (Hartley & Zisserman 2000). However, for clarity, we will define the

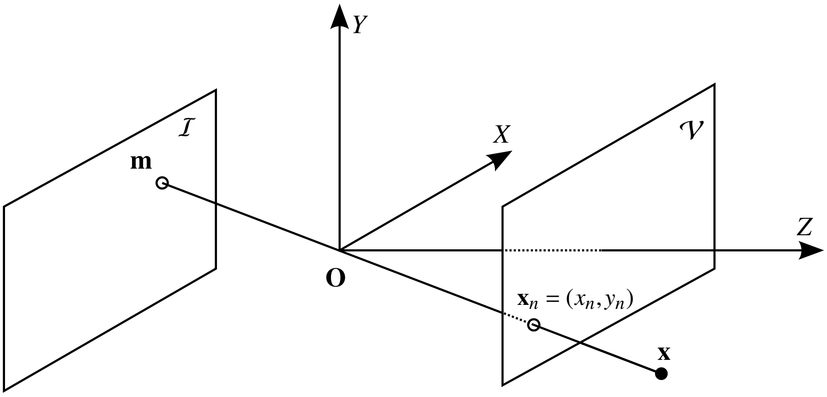


Fig 2. The perspective camera model. The scene point $\mathbf{x}$ is projected to point $\mathbf{m}$ on the image plane $\mathcal{I}$ by a camera located at $\mathbf{O}$. The plane $Z = 1$ is the virtual image plane $\mathcal{V}$. The mapping from $\mathcal{V}$ to $\mathcal{I}$ is defined by the intrinsic camera parameters.

model in inhomogeneous coordinates where it becomes a non-linear mapping from $\mathbb{R}^3$ to $\mathbb{R}^2$.

We can decompose the forward-projection into two parts: the extrinsic transformation $\mathcal{R}$, and the intrinsic transformation $\mathcal{P}_c$. This decomposition is useful because the extrinsic component depends solely on the position and orientation of the camera in the world and the intrinsic part is only a function of the internal characteristics of the camera. The forward-projection function is then a composition of the form:

$$\mathbf{m} = \mathcal{P}(\mathbf{x}) = (\mathcal{P}_c \circ \mathcal{R})(\mathbf{x}), \tag{1}$$

where $\mathbf{x} = [x, y, z]^\top$ is the scene point in world coordinates and $\mathbf{m} = [u, v]^\top$ is the measured position in pixel units.

The extrinsic transformation is a rigid Euclidean transformation, $\mathcal{R} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$, that translates points from the scene coordinate frame to a camera-centric coordinate frame. It rotates and translates the scene to align it with the camera coordinate frame. That is, after applying $\mathcal{R}$ the optical center is the origin of the coordinate frame, the Z-axis is perpendicular to the image plane, and the X-axis is aligned with the horizontal axis of the image plane. Such a rigid transformation can be defined by a function using a 3×3 rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$:

$$\mathbf{x}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \mathcal{R}(\mathbf{x}) = \mathbf{R}\mathbf{x} + \mathbf{t}. \tag{2}$$

For a perspective camera, the intrinsic projection function is composed of a perspective division $\hat{v}$ and an affine map

$$\mathcal{P}_c(\mathbf{x}_c) = (\mathcal{K} \circ \hat{v})(\mathbf{x}_c). \tag{3}$$

The perspective division is a constant function that divides a vector by its last element and discards this last element. This makes the forward-projection mapping highly non-linear.

$$\mathbf{x}_n = \begin{bmatrix} x_n \\ y_n \end{bmatrix} = \hat{v}(\mathbf{x}_c) = \begin{bmatrix} x_c/z_c \\ y_c/z_c \end{bmatrix}. \tag{4}$$

The affine map converts from normalized coordinates:

$$\mathbf{m} = \begin{bmatrix} u \\ v \end{bmatrix} = \mathcal{K}(\mathbf{x}_n) = \begin{bmatrix} f_x & \gamma \\ 0 & f_y \end{bmatrix} \begin{bmatrix} x_n \\ y_n \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \tag{5}$$

The focal lengths f_x and f_y determine the horizontal and vertical field-of-view respectively, γ determines the skew of the pixel axes, and the offsets u_0 and v_0 determine the location of the principal point (the projection of the optical axis) in the image. It is worth noting that in homogeneous coordinates, Equations 1 through 5 can be expressed succinctly as a single equation

$$\mathbf{m} \propto \mathbf{K}\,[\mathbf{R}|\mathbf{t}]\,\mathbf{x}, \tag{6}$$

where the equality is up-to-scale due to the use of homogeneous coordinates. The matrix $\mathbf{K}$ is known as the intrinsic matrix and contains the internal parameters of the pinhole model

$$\mathbf{K} = \begin{bmatrix} f_x & \gamma & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

2.1.2 Inverting the mapping

In practice the forward-projection $\mathbf{m} = \mathcal{P}(\mathbf{x})$ is a model of the physical process through which a camera images a point in the scene. It is often desirable to invert the process and obtain $\mathbf{x}$ from $\mathbf{m}$, especially in the context of 3D reconstruction. In order to obtain this backward-projection the individual components must be inverted, *i.e.*

$$\begin{aligned} \mathcal{P}^{-1}(\mathbf{m}) &= (\mathcal{R}^{-1} \circ \mathcal{P}_c^{-1})(\mathbf{m}) \\ &= (\mathcal{R}^{-1} \circ \hat{v}^{-1} \circ \mathcal{K}^{-1})(\mathbf{m}). \end{aligned} \tag{8}$$

Assuming a calibrated camera, *i.e.* all parameters in $\mathcal{P}$ are known, we observe that $\mathcal{R}^{-1}$ and $\mathcal{K}^{-1}$ are easy to obtain and produce a one-to-one inversion. However, inverting $\hat{\nu}$ is problematic because information is discarded when $\hat{\nu}$ is applied. At best, $\hat{\nu}$ can be inverted with one degree-of-freedom

$$\hat{\nu}^{-1}(\mathbf{x}_n) = \begin{bmatrix} z_c x_n \\ z_c y_n \\ z_c \end{bmatrix}, \tag{9}$$

where z_c is the unknown depth of the point. Thus, because the forward-projection maps all the points along an optical ray to the same pixel position, the backward-projection $\mathcal{P}^{-1}$ takes a pixel position and returns the ray containing all possible points that may have originated the measurement.

2.1.3 Distortion

The perspective camera model is an ideal model. Real cameras deviate from this model due to many factors, *e.g.* lens imperfections, lens decentering, manufacturing tolerances. Instead of trying to model each factor independently, it is common to model them jointly with a flexible distortion component that models the deviation from the ideal perspective camera. The distortion function $\mathcal{D}$ acts on the normalized coordinates after perspective division $\mathbf{x}_n$. The projection becomes

$$\mathcal{P}_d(\mathbf{x}_c) = (\mathcal{K} \circ \mathcal{D} \circ \hat{\nu})(\mathbf{x}_c). \tag{10}$$

The distortion function is often formulated in polar coordinates using the lens' optical center, also known as the principal point, as the origin. Using polar coordinates, the distortion is often separated into two main components: radial and tangential. Radial distortion alters the coordinates of the projected point based solely on the distance to the principal point. Radial distortion can produce either a barrel or a pincushion effect, as seen in Fig. 3. Tangential distortion also alters the coordinates of the projected point but it mostly depends on the polar angle.

Following Heikkilä (2000) we can separate the distortion function into its radial and tangential components:

$$\mathcal{D}(\mathbf{x}_n) = \mathbf{x}_n + \mathcal{D}_r(\mathbf{x}_n) + \mathcal{D}_t(\mathbf{x}_n). \tag{11}$$

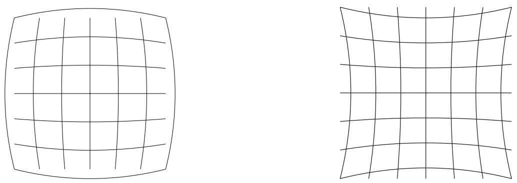


Fig 3. Radial distortion effects. Left: barrel distortion, lines bulge out away from the image center. Right: pincushion distortion, lines bend towards the center.

The radial component is approximated using the following expression

$$\mathcal{D}_r(\mathbf{x}_n) = (k_1 r^2 + k_2 r^4 + \ldots)\mathbf{x}_n, \tag{12}$$

where $r^2 = x_n^2 + y_n^2$ is the distance to the principal point and $k_1, k_2, \ldots$ are coefficients for radial distortion. Typically, one or two coefficients are enough to compensate for the distortion. The expression for the tangential component can be written as

$$\mathcal{D}_t(\mathbf{x}_n) = \begin{bmatrix} 2p_1 x_n y_n + p_2(r^2 + 2x_n^2) \\ p_1(r^2 + 2y^2) + 2p_2 x_n y_n \end{bmatrix}, \tag{13}$$

where p_1 and p_2 are the tangential distortion coefficients. These two components are flexible enough to model most deviations from the pinhole camera model for narrow-field-of-view cameras.

Usually, the distortion function (11) is a one-to-one mapping and therefore it can be inverted to obtain the backward-projection. However, the complex form of Eqs. (12) and (13) means that it cannot be inverted analytically. Instead, an iterative method is usually employed to invert $\mathcal{D}$ numerically. For example, the normalized coordinates $\mathbf{x}_n$ corresponding to each discrete image pixel may be stored in a look-up table.

2.1.4 Beyond perspective cameras

The distorted perspective camera model is a good approximation to most narrow field-of-view cameras and has been used successfully for many applications (Hartley & Zisserman 2000). However, the underlying perspective model is inherently limited to a narrow field-of-view. Figure 4 illustrates this limitation. As the angle between the optical ray and the optical center increases, the ratio x_c/z_c grows disproportionately. At an angle of 90°, the ratio reaches infinity and the model breaks down.

The limitation lies on the mapping from 3D camera coordinates to the virtual image plane. The perspective camera model uses a linear projection in projective

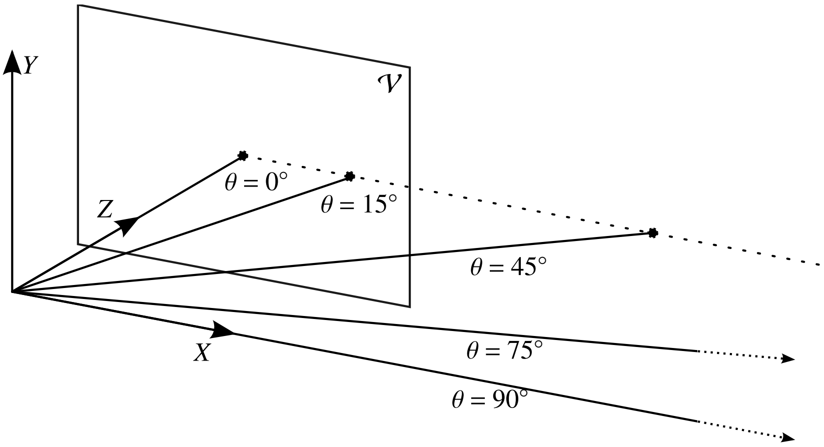


Fig 4. Limitation of the perspective camera model. As the angle θ of the optical ray with the optical axis increases, the coordinates tend to infinity. Note that at $\theta = 90°$ the optical ray is parallel to the virtual plane and never intersects. The pinhole camera model is unable to model cameras with a field-of-view of $90°$ or greater.

space for this mapping, *i.e.* perspective division. We can model cameras with a wider field-of-view by instead using a non-linear mapping function. For example, to model omnidirectional cameras, Kannala & Brandt (2006) parametrize the point by its spherical angle coordinates and apply a radially symmetric function based on a polynomial of the angles.

The distortion model must also be adjusted to suit this generic model. Equation (11) models the distortion as a function of the virtual plane coordinates. In the case of an omnidirectional camera, it has been recommended to model the distortion as a function of the spherical angle parameters.

Although these models are more general and support a wider range of cameras than the perspective model, their increased complexity makes them harder to calibrate and use. Therefore, many algorithms and applications still use the distorted perspective camera model. In particular, most depth sensors have a field-of-view considerably less than 180° and therefore use the perspective camera model as a base.

2.2 Depth sensor models

Depth sensors are similar to normal color cameras except that, instead of imaging the color of a scene point, they measure the distance of the point to the camera. The direct

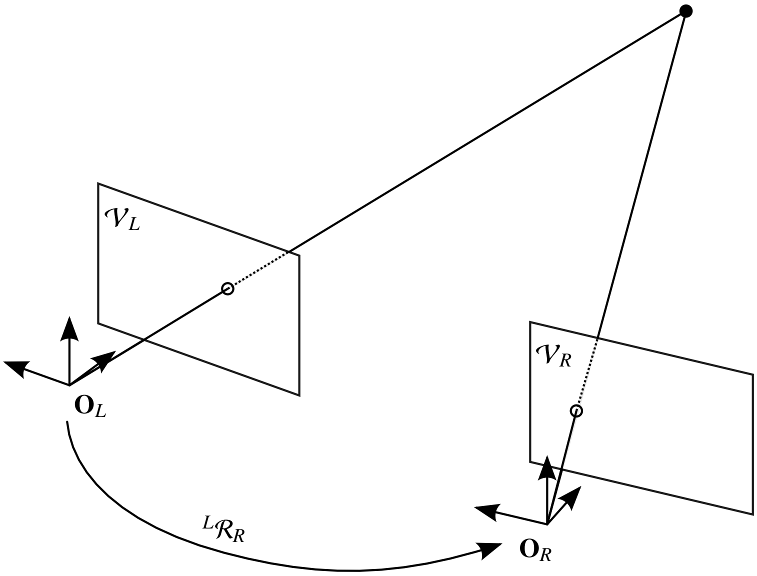


Fig 5. A stereo sensor consists of two cameras denoted *left* and *right*. Each camera can be modelled as a perspective camera centred at $\mathbf{O}_L$ and $\mathbf{O}_R$ respectively. The sensor is completely described by the intrinsic parameters for each camera and the relative rigid transformation between the camera centers ${}^R\mathcal{R}_L$. The 3D position of an observed point can be recovered by triangulation, *i.e.* intersecting the optical rays.

implication of this is that with a depth sensor we can measure the depth z_c and obtain a one-to-one mapping for $\mathcal{P}^{-1}(\mathbf{m})$.

The term *sensor* is used here because a depth sensor is often comprised of more than a camera. There are different principles by which depth sensors are constructed and each requires different hardware, *e.g.* a stereo system uses two color cameras, a structured light sensor uses a camera and a projector, a Time-of-Flight sensor uses an infrared light source and a special camera, *etc*. In the following, the principles of the most common depth sensors are outlined.

2.2.1 Stereo sensors

One of the most well-known ways of measuring the depth of a scene is to use a rig of two (or more) cameras that observe the scene from a different point-of-view. Such a rig, also known as a stereo rig, is illustrated in Fig. 5. Although a stereo rig consists solely of regular color cameras, we dub this a depth sensor because it is possible, through analysis of the images, to estimate the depth of the observed scene.

The stereo depth sensor is based on two principles to estimate depth of a scene. First, matches are found between the two images. This matching tells, either sparsely or densely, which pixel positions in the first camera observes the same scene point in the second camera. Thus, a match is described by a pixel pair $\mathbf{m}_1, \mathbf{m}_2$. This is, in fact, a very hard problem due to occlusion, low texture, repeating patterns, *etc*., and therein lies the main complexity of stereo depth sensors. Then, given a match and knowing that the two optical rays must intersect on the real point, it is possible to triangulate the position of the point in the scene and implicitly recover the depth.

A stereo rig is completely described by the model of two color cameras and the relative position between them. It has the advantage of a simple model that is easy to calibrate but suffers the cost of complicated algorithms for matching. Moreover, it is intrinsically limited by the texture of the scene and areas with little or repeating texture oftentimes cannot be reconstructed.

2.2.2 Time-of-Flight sensors

Time-of-Flight (ToF) sensors are active sensors that emit light and measure the time it takes for it to travel to the scene and back. By measuring this round-trip time and knowing the speed of light, it is then easy to calculate the distance to the scene. ToF sensors usually work with infrared light to avoid interfering with normal color cameras or the human eye.

ToF sensors can be divided into two categories, based on their working principle: pulse runtime sensors and continuous wave sensors. A pulse runtime sensor operates in the following way. It sends a pulse of light and starts a timer, then waits until it detects the reflection and stops the timer, thus directly measuring the round-trip time. This is an intuitive and direct way of measuring depth, but it requires very accurate hardware. Because the round-trip time t_{RTT} is measured directly, the depth can be simply computed as

$$z = \frac{t_{RTT} c}{2}, \tag{14}$$

where c is the speed of light.

A more common and more affordable type of ToF is the continuous wave sensor. It emits a continuous stream of modulated light, measures the phase shift in the observed reflections, and calculates the depth based on this phase shift. Given the frequency of the emitted light ν and the observed phase shift of the reflected light ϕ, the depth can be calculated as

$$z = \frac{\phi c}{4\pi\nu}. \tag{15}$$

The phase shift can be efficiently obtained by sampling the cross-correlation function of the emitted and received signals four times (Lange & Seitz. 2001). Due to the periodic nature of the phase shift, the working range of this type of sensor depends on the frequency and is usually around ten meters (Kahlmann & Ingensand 2006).

In both cases, the optics of the ToF sensor can usually be modelled by the distorted perspective camera model (10). This model then provides the optical ray and the calculated depth can be used to obtain the 3D point position corresponding to each pixel.

In practice, however, the formulas to estimate the depth (14) and (15) are only approximations. Real sensors exhibit many types of noise and distortions that must be accounted for to get an accurate reconstruction of the scene. For example, ToF sensors are susceptible to shot noise, mixed pixels, multiple reflections, and depth distortions due to reflectance and distance (Lindner & Kolb 2007, Fuchs & Hirzinger 2008).

Complex distortion models have been used to translate the measured round-trip time to the real depth. A common option is to use a B-spline to model deviations of the measured depth from the true depth (Fuchs & Hirzinger 2008). This allows an arbitrarily complex modelling of distortion function, limited only by the number of control points, at the expense of more calibration data. If reflectance distortions are to be taken into account it is possible to use a 2D B-spline (Lindner & Kolb 2007). However, proper sampling of the 2D reflectance-depth space for calibration becomes very costly. As a compromise, it has been suggested to model them separately by first normalizing the observed intensity based on the measured distance, and then applying two independent correction factors based on measured distance and normalized intensity (Lindner *et al.* 2010).

2.2.3 Structured light sensors

Structured light sensors are another type of active depth sensor. They also emit a light signal on the scene but the modulation is done spatially instead of temporally. It is very similar to a stereo sensor except that one of the cameras is replaced by a projector. In fact, the same model as in Fig. 5 is used because a projector can also be represented with the distorted perspective model.

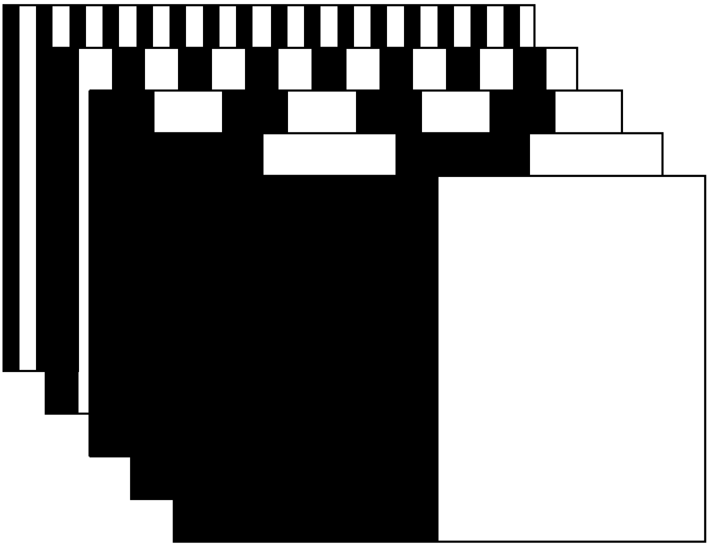


Fig 6. Binary coded patterns used for a structured lighting sensor. Each pattern is projected separately and an image is taken. After projecting n patterns each camera pixel will have an n-bit vector that uniquely describes the matching projector pixel.

Structured light sensors also work on the same principle as a stereo sensor. First, matches are made between the optical rays of the projector and the camera. Then, the scene points are triangulated from the pair of optical rays. Since we control the projected image, we can make sure that there is good texture in the scene, thus making the matching stage much easier.

There are different types of patterns that one can project depending on the accuracy and speed requirements. A common method is to project several *binary coded patterns* and capture one image for each pattern (Salvi *et al.* 2004). An example of a binary coded pattern is shown in Figure 6. This allows every projector pixel to be identified and matched producing a very dense and accurate reconstruction. However, it also requires the scene to be static during the capture.

An approach that has become very popular for both 3D reconstruction and gaming is to project an image with small unique patches that can be easily identified and matched. This allows the device to obtain projector-camera matches with only a single image, thus eliminating the need for a static scene. This is the approach taken by the first generation of the Microsoft Kinect (Freedman *et al.* 2008) which is able to reconstruct the scene at 30 fps. We will explore the details of the Kinect sensor both because of its popularity and because its calibration is one of the main contributions of this thesis.

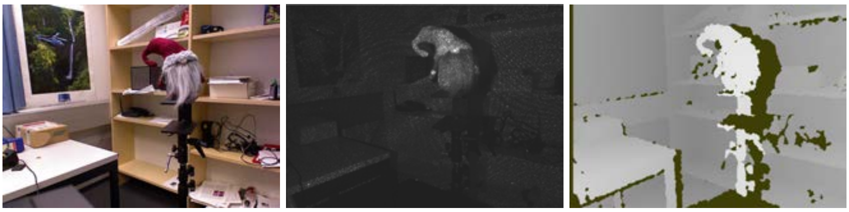


Fig 7. Sample Kinect images from a scene. From left to right: color, infrared, and depth.

The exact details of the Kinect are proprietary but much is known from patents (Latta 2010) and reverse engineering. An infrared light is passed through a diffraction grating that generates a unique speckle pattern, see Fig. 7. During construction, the Kinect memorizes the speckle pattern by projecting it onto a flat wall at a known distance. Then, during operation, it projects the pattern onto the scene and finds how the speckle pattern has shifted compared to the memorized image. A region growing algorithm is then used to infer a semi-dense depth map from the observed deviations.

In practice, the Kinect provides three types of images: color, infrared, and depth. Color images are taken with a regular color camera. Infrared images are taken with a second camera that observes only infrared light. These infrared images show a mix of the original scene texture and the projected speckle pattern. Finally, the depth images are derived from the infrared images using the internal algorithms of the Kinect.

The depth image, also known as *depth map*, is given in what we call *Kinect disparity units* (*kdu*). It has been shown that these units can be roughly translated to metric units using the formula

$$z(d) = \frac{1}{c_1 d + c_0}, \tag{16}$$

where d is the disparity in *kdu* and c_0 and c_1 are constants. The distorted perspective model can be used to describe the depth camera since it is a narrow-field infrared camera. However, it has been observed that even though the depth image is derived from the infrared image, they are not aligned, *i.e.* there is an offset between their coordinate frames. Thus, the perspective model parameters for the infrared and depth images may be different.

Naturally, Eq. (16) is only an approximation and distortions are always present. Due to the proprietary nature of the Kinect algorithms, it is hard to definitely state the source of these distortions. It has been observed, however, that some of these sources of error are systematic and can be corrected through calibration. This process is described in detail in Paper I and will be reviewed in Section 2.3.3.

2.3 Calibration methods

All the models reviewed in the previous sections have several parameters that vary from camera to camera. These parameters have to be estimated for each device. Even in the case of a simple model whose parameters may be estimated from manufacturer specifications, *e.g.* the focal length of the perspective model based on the camera's field-of-view, it is often best to calibrate them due to imperfections and tolerances during manufacturing. The following section first reviews the common methods used to calibrate the distorted perspective camera model of Eq. (10). Then, Section 2.3.2 reviews the extensions necessary to calibrate depth sensors.

2.3.1 Calibrating the pinhole model

We wish to estimate the parameters of the model based on image measurements. In the case of the distorted perspective camera model, the parameters to estimate are the camera pose (2), the distortion coefficients (11), and the intrinsic matrix coefficients (5).

We can use Eq. (10) to establish constraints between the scene and pixel coordinates of a point. However, the non-linear nature of the distortion model results in non-linear constraints that cannot be solved analytically. Therefore, it is customary to start by calibrating the distortionless model (1) in closed-form and using this as an initial guess for an iterative optimization of the full model (10).

It has been proven that if we do not assume anything about the scene the camera parameters can only be reconstructed up to a projective transformation (Hartley & Zisserman 2000). This is not enough for a 3D reconstruction pipeline because a projective transformation deforms the scene in undesirable ways. Thus, it is necessary to know something about the scene or cameras to eliminate this ambiguity during calibration. Mathematically, calibration is simplest if the coordinates of the scene points are known and they are not coplanar. In this case, the projection matrix can be estimated directly from linear constraints (Hartley & Zisserman 2000). It is, however, impractical to use a 3D calibration object with precisely known coordinates. A more practical and popular technique is to use a plane with known texture (*e.g.* a checkerboard) and take several images with the same camera, thus assuming that the intrinsic matrix is constant (Zhang 1999).

It is also possible to calibrate a camera without any knowledge of the scene, a procedure known as *self-calibration*. However, more constraints are needed in this case, for example, fixed intrinsic matrix or only varying focal length. In these cases, it is possible to perform self-calibration of a moving camera, either with general motion or purely rotational motion (Hartley & Zisserman 2000).

2.3.2 Calibrating depth sensors

Stereo sensors are straightforward to calibrate because they are composed of standard color cameras. It is possible to calibrate each camera independently with the techniques described in section 2.3.1. The relative position of the cameras can then be easily calibrated by simultaneously observing a simple calibration pattern (*e.g.* a checkerboard) with both cameras. For optimal results, a non-linear least squares minimization over all parameters (intrinsics, relative position between cameras, and position of the pattern) and all images can be performed. The images where the pattern is observed in only one camera will only constrain that camera's intrinsic parameters and the images where the pattern is observed in both cameras will constrain both the intrinsics and the relative position between cameras.

Calibrating a projector-camera rig follows a similar procedure as a stereo rig. However, it presents the added difficulty that a projector cannot perform measurements. A projector can be thought of as the dual of a camera, *i.e.* it follows the same model but emits light instead of measuring it. Therefore, a camera is needed to make measurements for the projector. A common procedure is to first independently calibrate the rig's camera and then use this camera to calibrate the projector. Once the camera is calibrated, a pattern is projected onto a surface of known geometry and position (*e.g.* a plane). Because the geometry and position are known, the camera can be used to establish correspondences between features on the 3D surface and features on the original 2D pattern, thus providing the necessary data for calibration (Gockel *et al.* 2004).

Time-of-Flight sensors produce two kinds of images: active brightness and measured depth. Both images are generated from the same underlying measurements and are thus taken from the same exact viewpoint and share the same projection parameters. This is very useful for calibration because it is hard to obtain accurate scene-to-image correspondences from a depth image. Therefore, one can use the same planar-pattern calibration technique as before to obtain most of the model parameters (Hansard *et al.* 2012). Yet, ToF sensors present the added difficulty that the measured depth must be corrected to obtain the true depth. The measured depth presents distortions that are a function of the reflectance and true distance (Lindner *et al.* 2010). In order to make the calibration practical, it is usually assumed that all pixels share the same distortion function. This simplifies calibration, yet it is still necessary to sample over different reflectances and distances to recover the distortion function. For example, Lindner *et al.* (2010) use a checkerboard pattern with varying brightness and place it at various distances to sample the reflectance-depth space.

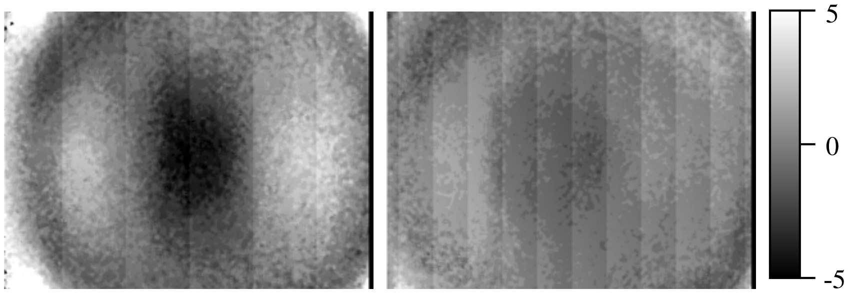
2.3.3 Contribution: calibrating the Kinect

The Kinect is a special type of structured-light depth sensor and its proprietary nature presents unique challenges for calibration. Even though the depth image is internally derived from the infrared image, they are not aligned. Since the processing algorithm is unknown and may change, it is desirable to calibrate the Kinect without using the infrared images. However, unlike Time-of-Flight sensors, the Kinect provides measurements in *Kinect disparity units* whose mapping to metric units must be calibrated as well. This introduces additional degrees of freedom in the model that easily make calibration with a single depth camera an under-constrained problem.

Several approaches have been proposed to calibrate the Kinect. It is possible to estimate the offset between the infrared and depth images, so that calibration is done with the infrared images and the obtained intrinsics are transformed to match the depth images (Smisek *et al.* 2013). This offset and the implied transformation, however, depend on the internal algorithms of the Kinect, which might change at any point. Moreover, this requires an external infrared illumination, which makes the method less practical. It is thus highly desirable to perform calibration directly from the depth images.

Unfortunately, intensity discontinuities are not visible in depth images which complicates the process of obtaining constraints for calibration. It is in theory possible to use depth discontinuities in a similar way. However, this requires a 3D calibration object with known structure which is not easy to fabricate. Moreover, depth discontinuities are often very noisy, as seen in Fig. 7. Paper I proposes a generic calibration method that overcomes these limitations.

We propose the use of a planar calibration target for calibration. Instead of enforcing point-to-point matches via discontinuity detection, the algorithm uses the constraint that all points on the target are coplanar. This constraint is less restrictive than a set of point-to-point matches, *e.g.* the camera can be rotated around the plane's normal without any effect. The use of several images observing the plane from different angles provides



(a) Plane at 0.56m

(b) Plane at 1.24m

Fig 8. Kinect error residuals of a planar scene, in disparity units, without distortion correction.

complimentary constraints that contribute to calibration. However, if the parameters of the disparity to depth equation (16) are unknown, the experiments performed for Paper I showed that the calibration is under-constrained even with multiple images observing the plane from different angles. The missing constraint are provided by a color camera rigidly attached to and jointly calibrated with the depth sensor.

This is particularly practical for the Kinect sensor because it contains both cameras and both need to be calibrated. Calibrating both cameras jointly is preferable to separate calibration because the cameras help constrain each other's calibration, thus obtaining a potentially better result. Additionally, it is possible to jointly calibrate another high-resolution camera rigidly attached to the Kinect to increase calibration accuracy.

It has been observed that, just like for ToF sensors, the measured depth for a Kinect pixel presents distortions. Figure 8 shows the reconstruction error obtained after calibration of a planar scene. Smisek *et al.* (2013) corrected this distortion with a per-pixel constant offset in metric units. Paper I showed that this distortion is more accurately corrected in disparity space. In this case, a per-pixel offset is added to the measured disparity, but it was found that the offset decays with increasing distance. The method presented in Paper I is able to calibrate this distortion model and obtains a more accurate calibration than previous approaches.

2.4 Discussion

The models presented in Sections 2.1 and 2.2 are the necessary first step to any geometric computer vision algorithm. Properly modelling the structure of the sensor and its measurement process is essential to using its measurements effectively. 3D reconstruction would not be possible without a proper modelling of how the 3D structure of the world is transformed into the observed image.

Although generic models were presented in Section 2.1 that can potentially cover almost any kind of camera, the more specific models discussed are more efficient and easier to calibrate and use. The model used will depend on the application and the expected variability of the sensor.

The proposed calibration method of Paper I (Section 2.3.3) is one of the main contributions of the thesis. It is an example of a combination of three key aspects of camera calibration: accuracy, flexibility, and practicality. In particular, using only a planar surface for depth camera calibration and avoiding depth discontinuities resulted in a very practical algorithm that still achieves high accuracy.

The algorithm is flexible to calibrate different depth sensor types because the infrared images are not used. For example, the Kinect version 2 is no longer a structured light sensor but uses Time-of-Flight instead, yet the same calibration algorithm can potentially be applied with minimum changes to the model.

Camera calibration is an important part of any 3D reconstruction pipeline. The proposed algorithm has been used for a wide variety of applications. For example, camera tracking (Tykkälä *et al.* 2014), analysis of plants from depth images (Chéné *et al.* 2014), hand gesture recognition (Dominio *et al.* 2014), gaze estimation (Funes Mora *et al.* 2014), and robotic manipulation of food items (Morales *et al.* 2014), among others.

3 3D reconstruction

The problem of 3D reconstruction consists of recovering the physical structure of the scene and the camera positions from a set of images. It has been a driving force in the development of computer vision since its early stages and still remains a challenging problem. It is known by different names depending on the field of research (*e.g.* *photogrammetry*, *structure from motion*, *SLAM*). Some fields tend to have different requirements, for example, in robotics, the reconstruction is often expected to be real-time to allow the robot to navigate using vision. This is often termed *simultaneous localization and mapping* (SLAM) because of the need to localize the robot and map the environment. On the other hand, in photogrammetry, the sequences are processed offline and high accuracy is expected. However, since the terms are also used interchangeably, this thesis makes no distinction between them.

The complexity of the problem varies greatly depending on the requirements and assumptions made. We can reduce the complexity of the problem by using a calibrated camera, limiting the structure of the scene (*e.g.* a planar scene), and/or by limiting the camera trajectory (*e.g.* pure rotations or translation in a plane). In the following, Section 3.1 explores the nature of the reconstruction problem and analyses some relevant special cases. Sections 3.4 and 3.5 look at the particular requirements of doing a sparse and a dense reconstruction respectively. The contribution of this chapter is an important part of the thesis and is discussed in Section 3.4.2 and Paper II. It consists of a novel method of merging triangulated and non-triangulated features inside a SLAM system. Both feature types are used during real-time pose estimation and background bundle adjustment to improve the pose estimate. Moreover, this allows the system to correctly handle pure camera rotations where no features have been triangulated.

3.1 A three-part problem

We look at the reconstruction problem here from a calibrated perspective, *i.e.* all cameras involved have been calibrated and the images have been warped to eliminate distortion. Thus, the equations to follow can ignore distortion without any loss of generality. Even though a calibrated setting simplifies the reconstruction problem, it is still complicated because we need to estimate the scene structure and camera positions simultaneously. In a connected scene, all point positions and camera poses are (at least indirectly) related to each other through the measurement Equation (6). The equation is revisited below with additional indices to show how a measurement m_{ij} depends on point i and camera j

$$\mathbf{m}_{ij} \propto \mathbf{K}_j \left[\mathbf{R}_j | \mathbf{t}_j \right] \mathbf{x}_i. \tag{17}$$

This shows that the reconstruction problem is a three-part problem involving the measurements, the camera poses, and the point positions. These form a tripartite graph, *i.e.* if two parts of the problem are known (*e.g.* the measurements and camera poses), each element of the third (*e.g.* the point positions) can be estimated independently. We explore these special cases first before addressing the more general problem of simultaneous estimation.

Although determining the measurements is an integral part of the 3D reconstruction problem, this chapter focuses on the estimation of camera pose and scene structure. Most of the 3D reconstruction algorithms rely on image correspondences to work. An image correspondence consists of two pixel positions in two different images that correspond to the same physical point in 3D. Obtaining these matches is a non-trivial problem and is still a rich area of research. Often, the process is divided in two steps: feature detection, where interest points are located in each image, and feature matching, where the interest points are matched between images. These matches may even be refined after the reconstruction is complete to iteratively refine the result (Furukawa & Ponce 2009). Feature detection and matching is beyond the scope of this thesis. A review of different methods can be found in (Szeliski 2010). In the following, we assume that these tasks are completed and the correspondences are available.

3.1.1 Triangulation

Given a set of images from a scene with known pose and calibration, we can estimate the position of each observed point independently. This is known as triangulation. Equation (17) produces two constraints for each camera that observes the point (Hartley & Zisserman 2000). Thus, two observations are sufficient to obtain the 3D position of a point. This is illustrated in Figure 5. Geometrically, triangulation consists of finding the intersection of the optical rays.

There are a few ways of performing the triangulation, as described in (Hartley & Zisserman 2000). It is ultimately a non-linear problem due to outliers and distortion. However, a recurring theme in computer vision is the solving of a series of non-linear constraints by first obtaining an approximation through linear means and then refining it using non-linear optimization. This also applies to triangulation.

To obtain the linear triangulation, we derive constraints from Equation (17). Because the point is expressed in homogeneous coordinates, it is tempting to fix the last coordinate of the point to one, thus obtaining a linear system of the form $\mathbf{A}[x,y,z]^\top = \mathbf{b}$ which can easily be solved through the pseudoinverse of $\mathbf{A}$. However, this process becomes numerically unstable when the optical rays of the measurements are close to parallel. This can happen either when the baseline between the cameras is small or when the point is very far away from the camera, situations that are both very common in most sequences. Therefore, it is preferable to estimate a homogeneous quantity. This results in a homogeneous linear system of the form $\mathbf{A}[x,y,z,w]^\top = \mathbf{0}$ which can be solved through the SVD decomposition of $\mathbf{A}$.

It is important to note that even though the homogeneous triangulation equations are numerically stable, this does not mean that the position of the point is well constrained. For example, in the extreme case where the transformation between cameras is a pure 3D rotation (*i.e.* no translation), the corresponding optical rays are parallel and no depth can be estimated. Any point along the optical ray will satisfy the constraints, including a point at infinity.

3.1.2 Camera pose estimation

The dual to triangulation is camera pose estimation with a known scene structure. Given a map that contains a set of points with known 3D position and a set of observations of these points in the images, we can estimate the position of each camera independently. In the calibrated case, we only need to estimate the rotation and translation of the camera (6 DoF). This is known as the Perspective-n-Point problem and can be solved minimally with only 3 points (Quan & Lan 1999).

A special case for pose estimation is when relative pose between the cameras is a pure 3D rotation. In this case, the true depth of the points can be ignored. The rotation can be recovered by assigning a constant depth to all measurements and solving through a simple SVD decomposition (Kanatani 1994).

3.2 Simultaneous structure and motion

It is also possible to estimate the camera pose without knowing the scene structure. In this case, there is no reference coordinate system as the one implied by a map. Thus, the pose of a camera is estimated relative to a reference camera. The relative pose can be estimated using only matches between corresponding points in both images. The geometric constraints between two calibrated images are encoded into the *essential matrix*, which implicitly contains the relative camera pose. A minimum of five correspondences is needed to estimate the essential matrix which decomposes into a rotation and translation (Hongdong & Hartley 2006). However, due to the well-known scale ambiguity for visual reconstruction, the scale of the obtained translation is unconstrained and can be chosen arbitrarily. The selected scale of this translation will determine the scale of the reconstructed scene.

This already provides the foundation to perform 3D reconstruction from a set of image correspondences. The camera pose is first estimated from the correspondences and then the 3D point positions can be obtained through triangulation. This is a straightforward solution when reconstructing a scene from two images. However, when more images are available, there are more constraints to take into account. The pose recovered from image correspondences is always up to scale. Thus, if the scene is reconstructed independently using two different image pairs, the scales will not match. In fact, as it is to be expected, a set of correspondences across three images produces stronger constraints than considering only the pairwise constraints between each pair of the same images.

This constraint is encoded in the trifocal tensor, as described in Hartley & Zisserman (2000). The trifocal tensor can be accurately estimated from image correspondences, just like the pairwise *essential matrix*. It implicitly contains the relative poses between all three images in a coherent scale. Once estimated, it can be readily decomposed into respective camera poses needed for triangulation. If more than three images are available, it is still necessary to match scales between the different triplets. Fitzgibbon & Zisserman (1998) presented a hierarchical approach that uses image triplets as the basic building block for 3D reconstruction over long image sequences.

3.3 Bundle adjustment

In most cases, it is necessary to use a linear approximation to obtain an initial solution. Distortion and other non-linearities are either ignored or approximated to solve the problems of triangulation and pose estimation efficiently. Moreover, the quantities are often estimated using only image pairs or triplets, even though there are many more images that can better constrain the solution. Therefore, once an initial reconstruction is obtained, it is necessary to perform a non-linear optimization of all parameters to obtain an optimal reconstruction. This is known as bundle adjustment (Hartley & Zisserman 2000). The quantity minimized is a robust function of the reprojection error. The general form of the bundle adjustment problem is

$$\underset{\mathbf{K}_k,\mathbf{R}_k,\mathbf{t}_k,\mathbf{x}_i}{\arg\min} \sum_k \sum_i \rho(|\mathbf{m}_{ik} - \mathcal{P}_k(\mathbf{x}_i)|^2), \tag{18}$$

where the projection function depends implicitly on the camera intrinsics $\mathbf{K}$ and extrinsics $[\mathbf{R},\mathbf{t}]$. The function $\rho(\cdot)$ is a robust function that limits the effect of outliers. Bundle adjustment requires a large-scale optimization framework. The optimization takes place over all point positions, camera poses, and often times also the camera intrinsics. Even medium-sized reconstructions contain hundreds of thousands of parameters to optimize.

Efficient optimization of such large scale problems has proven possible thanks to careful use of its sparse matrix structure (Triggs *et al.* 1999, Hartley & Zisserman 2000). Because not all points are observed by all cameras, the resulting Jacobian of the cost function has block diagonal sparsity. Through matrix manipulation, the scale of the problem can be considerably reduced. In particular, the Schur complement (Agarwal *et al.* 2015b, Zhange 2005) permits us to factor out the scene structure and solve only for the camera poses in an optimal way.

3.4 Sparse scene reconstruction

Structure from motion has made considerable progress and produced very impressive results. Snavely *et al.* (2007) published an open source implementation called *Bundler* that applies this pipeline to an unordered set of images. Their research was taken by Microsoft and expanded into a fully functional application called *PhotoSynth* that allows the user to explore a collection of photos using the recovered 3D structure and poses.

3.4.1 Online methods

Some applications of 3D reconstruction require real-time performance. For example, robot navigation and augmented reality require very low-latency estimations of the camera pose. This demands a careful balance of resources to satisfy the computational demands with current hardware. Pose estimation becomes a critical component of the system because applications such as augmented reality require a valid pose to render content. Offline methods can examine the entire image set and select the optimal frames for reconstruction, whereas online methods receive frames sequentially and must generate a valid camera pose for each incoming frame.

Initial approaches avoided bundle adjustment by using a filtering framework that collapses all previous measurements into the current *state* of the system (*e.g.* MonoSLAM (Davison *et al.* 2007) and Eade & Drummond (2007)). The state models the position of the points and cameras, as well as their first-order uncertainty through a covariance matrix. Each new measurement updates the state and reduces the covariance of the observed features. However, updating a large state is costly and thus such systems were limited to tracking a small number of features, which limits accuracy (Strasdat *et al.* 2010).

Klein & Murray (2007) presented one of the first real-time SLAM systems using bundle adjustment. They reduced the processing time required for each frame by estimating the scene structure only from a subset of key frames. Moreover, they separated processing into two parallel threads, one real-time for pose estimation and user interface, and the other for scene reconstruction and bundle adjustment, thus shifting the slower components to the background. The efficiency of this approach has been highlighted by recent developments. Forster *et al.* (2014) presented a visual odometry system based on a similar structure that is lightweight enough to run on a micro aerial vehicle with very limited hardware.

3.4.2 Contribution: deferred triangulation under varying baseline

A major contribution of this thesis is a SLAM system that builds upon the framework of Klein & Murray (2007). It extends the framework by including a reprojection term for features that have not yet been triangulated. This allows unrestricted camera motions and utilizes all available matches to estimate the camera pose and scene structure. A brief summary of the system is provided here, more details can be found in Paper II.

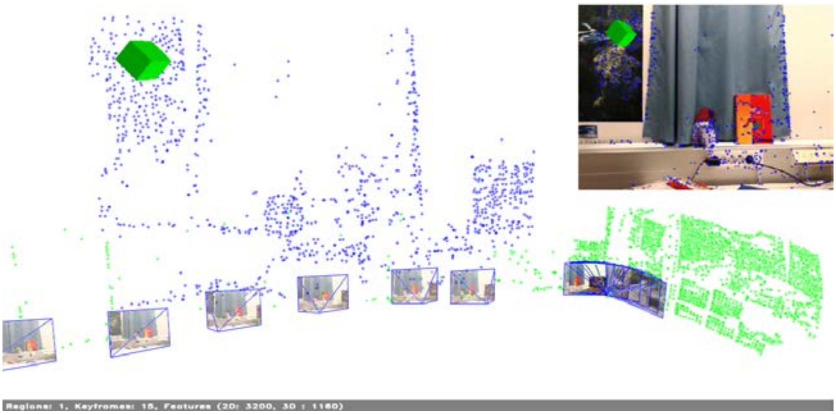


Fig 9. A sample reconstruction from Paper II, including a pure rotation. Blue points are triangulated points. Green points are non-triangulated points plotted at a unit depth.

Most key frame-based SLAM systems represent all the model points in 3D. However, when the baseline between key frames is small compared to the depth of the point, the triangulation ambiguity is high and the erroneous 3D points corrupt the map and degrade tracking. To cope, most systems avoid adding points when there is little baseline (*e.g.* when camera is only rotating). However, this ignores image correspondences that can in fact constrain the pose between images.

The framework proposed in Paper II tracks and maps both triangulated and non-triangulated features. All features contribute to estimating the camera pose and building the map. Triangulation is deferred until enough parallax is observed from at least two key frames. The key aspect of the system is the addition of a reprojection term to Eq. 18 for points that haven't been triangulated yet. Without depth, a point from one image reprojects to a line segment on another image. The new reprojection error measures the distance between the observed measurement and the reprojected line segment. This constraint is weaker than a direct point-to-point distance, but it still constrains the rotation and the direction of translation between cameras.

Because all image correspondences are tracked between frames, the system can detect pure rotations and estimate the camera pose even when no triangulated points are observed. Moreover, the system can effectively decide when a new key frame should be added based on whether it will contribute new features or enable new triangulations.

3.5 Dense scene reconstruction

The approaches presented so far reconstruct a scene based on feature matches. These matches are sparse and result in a sparse reconstruction of the scene. Although a sparse reconstruction can still be useful, specially for the recovered camera poses, it is often desirable to obtain a dense reconstruction of the scene. There has been very promising progress in this area. Seitz *et al.* (2006) provide a review and comparison of early multi-view dense reconstruction approaches. Their benchmark has been kept up to date and shows a comparison of many modern reconstruction methods. Although a thorough review of dense reconstruction methods is out of the scope of this thesis, some examples from the literature will be highlighted to give an impression of the state of the art.

A very popular semi-dense reconstruction software was released by Furukawa & Ponce (2010) called Patch-based Multi-view Stereo (PMVS). It uses the camera poses produced by a sparse reconstruction algorithm as input and focuses on the feature matching and triangulation aspects. It iteratively matches and triangulates features, starting with the most salient and easy to match. It uses the obtained depths as to aid the matching of less salient patches. In this way, the algorithm is able to cope with very weak or ambiguous texture. However, areas with no texture are not reconstructed and thus the algorithm produces a semi-dense reconstruction.

Vu *et al.* (2012) achieved very impressive results on large scale datasets by developing a two-stage pipeline. First, a semi-dense point cloud is generated by integrating photometric and visibility constraints. Then a mesh-based variational refinement captures small details. The reconstructed scenes can range from small objects with millimetre precision to city-wide reconstructions.

Recently, online dense reconstruction methods have become a possibility. Newcombe *et al.* (2011b) present a real-time system using GPU hardware. They bootstrap their reconstruction with the camera poses from a sparse reconstruction. Using hundreds of images from a video stream, they construct a photometric cost volume and minimise a global spatially regularised energy functional. This produces a dense depth map for selected key frames in the sequence. Once a depth map is generated, the system tracks the pose of the camera using the dense reconstruction.

Engel *et al.* (2014) published a method that avoids the need for a sparse reconstruction. It is a direct method in the sense that it does not extract features but works directly on pixel intensities throughout the system. Thus, it produces a full resolution depth map directly. They keep track of the estimated depth and uncertainty for each pixel, probabilistically refining them as new images are received. Depth maps are then linked together and their relative poses globally optimized to create a large scale reconstruction.

The arrival of consumer depth cameras, like the Microsoft Kinect, have contributed considerably to this field and have reduced the complexity of dense reconstruction. Image correspondences between depth cameras already contain 3D positions. This means that recovering the camera pose is a simple absolute orientation problem and there is no need to triangulate points. This leads to very fast and simple SLAM systems (*e.g*. Dryanovski *et al.* (2013)).

The quality of the reconstruction can also be significantly improved by merging overlapping depth images. Newcombe *et al.* (2011a) initially presented a system to merge several depth maps into a high quality surface volume. This was later extended by Chen *et al.* (2013) into a large scale SLAM system that uses only depth maps, thus demonstrating the high potential of depth cameras for this field.

3.6 Discussion

3D reconstruction has been one of the most active areas of research in computer vision. The fundamental mathematical framework has been established already for a long time, yet robust, flexible, and accurate systems are still hard to find. The recent move towards dense reconstruction systems reflects the improvements in computational capacity and depth sensor hardware. Yet, real-time, large-scale, and scalable sparse reconstruction is still a challenging problem.

The possible applications are many and impose different requirements. Offline reconstruction methods are nowadays mature and robust, but real-time systems are still a challenge. In particular, mobile platforms with limited computational complexity and battery life make real-time sparse reconstruction methods particularly attractive.

The system presented in Paper II fits in this category of real-time sparse reconstruction algorithms. Its contributions are aimed at making the reconstruction more robust and stable by using all available matches. Moreover, the system is presented to the community as open source software to foster cooperation. We expect other researches to use this system as a base for further research and improvements.

The scenes and videos used in Paper II are only small scale reconstructions. More tests and optimizations are, of course, needed to have a scalable algorithm that can reconstruct efficiently larger areas. Furthermore, the area seems to be moving towards depth sensors to improve reconstruction and lower the computational complexity, both in desktop and mobile. The algorithm of Paper II uses only a monocular color camera for reconstruction. However, extending the algorithm to use depth information provided by a depth sensor is a possibility and would increase the robustness of the method.

4 Depth map inpainting

A dense image is one in which all pixels have known values. Most cameras produce dense images, though not all. Depth cameras in particular often produce semi-dense images in which a small subset of pixels have no measurements. Depending on what the images are used for this might be unacceptable. Some applications, like free-viewpoint rendering (see Chapter 6), require a dense depth map. In such cases, it is necessary to infer (or guess) the values of the missing pixels from the rest of the image. This process is known as image inpainting.

The causes for missing pixels are many and depend on the technology of the camera. Stereo depth cameras have occluded areas that only one camera can see and thus no depth can be estimated. Most passive depth estimation methods require texture to work and thus produce no depth for textureless areas. Structured-light sensors overcome this by creating texture with specially crafted light, however, they still suffer from occlusions. Active depth cameras are often limited by the surface material. Highly specular or very dark surfaces may not be reconstructed and will produce missing pixels. Finally, inpainting is also very relevant to color images in cases where the image has been corrupted, *e.g.* old scanned photographs, or when we would like to remove an object from the scene.

In most cases, it is impossible to recover the true values for the missing pixels. However, the true values are often not necessary, a set of plausible values that are consistent with the known pixels is sufficient. For example, Figure 10 shows three possible inpaintings for a sample color image [1]. One is clearly wrong but the other two are equally plausible. In theory all pixels in the image determine what is a plausible solution. However, the pixels in the boundary around the missing pixel region have a much higher impact because discontinuities may produce very obvious artifacts.

Image inpainting in itself is an active field of research. It is an ill-posed problem and thus strong regularization or priors are needed. Solving the inpainting problem consists of two major things. First, one must find the right model that favours plausible solutions. This includes a noise model for the measurements and a suitable prior. Second, one must be able to optimize the image values under the chosen model to obtain a good solution.

[1] Because it is hard to evaluate a depth map with the naked eye, Fig. 10 shows color image inpainting examples.

Fig 10. A color image inpainting example. Three possible inpaintings for a missing region. The first inpainting looks clearly incorrect, whereas the last two are equally plausible.

There has been considerable progress in color image inpainting (Bertalmio *et al.* 2000). Many of the ideas used in color image inpainting have been applied to depth images as well. However, although similar, the problems of color and depth image inpainting are not identical. Section 4.1 discusses the differences between them. Still, a lot has been learned from the efforts of color image inpainting that are applicable to depth images.

The image inpainting problem is often cast as an energy minimization problem. An energy function is defined that captures our knowledge about the problem (the available measurements, the noise model, the priors) and it allows us to rank different solutions. The solution with the lowest energy is considered the best solution. Unfortunately, the complexity of the minimization depends on the shape of the energy function and can easily become untractable. This forces us to consider the shape of the prior carefully and select different optimization methods. Some of these priors and optimization methods are reviewed in Section 4.2. Alternatively, some algorithms avoid modelling the energy function and implicitly contain the priors in the algorithm itself. This is reviewed in Section 4.3.

The contributions of this chapter are discussed in Sections 4.2.3, 4.2.5, and 4.3.1, which discuss Papers III, IV, and V, respectively. Each presents a different approach to depth map inpainting. Paper III proposes an inpainting method with a 2nd-order prior that favours planar surfaces and does not suffer from a fronto-parallel bias. Paper IV uses a high-order prior based on natural image statistics. It learns the shape of the prior from a database of images and builds a generative distribution to model the image statistics. It extends previous approaches that have applied similar a prior to color images by expanding the prior to include aligned depth and color images. This allows the inpainting to take cues from the color image when deciding the location of inpainted depth discontinuities. Finally, Paper V presents a much simpler but also faster inpainting algorithm that also tries to enforce planarity in the scene without bias.

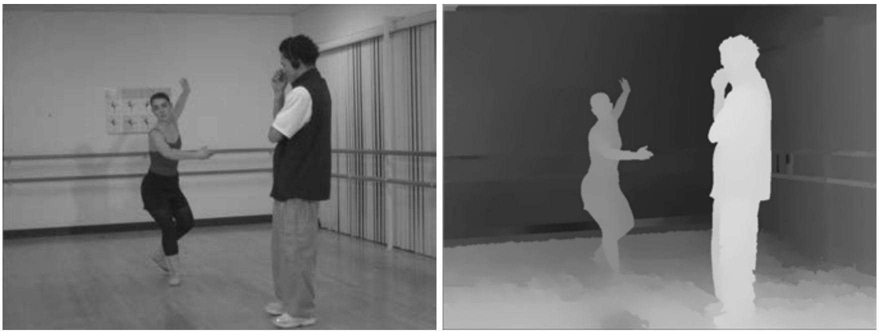


Fig 11. Intensity and depth images side-by-side. It is easy to determine which is which based on their structure.

It inpaints missing regions in a local fashion by extracting candidate planes from the boundary of the missing region.

4.1 Differences between color and depth inpainting

There are two major differences that separate color and depth image inpainting. First, the statistics of the images are themselves different. Second, there often exists more information in the depth map inpainting problem.

4.1.1 Image statistics

The biggest difference between inpainting color and depth images lies in the underlying structure of the images. In other words, the statistics of color and depth images are different. To see this, it is enough to place an intensity and a depth image side-by-side, as in Fig. 11. It is almost trivial to guess which is which because of their structure, *e.g.* depth images have larger continuous areas with smoother gradients.

The priors used to regularize the inpainting impose a predetermined structure on the result. Thus, since we expect different image structures, we can also expect that different priors should be used for colors and depth images.

4.1.2 Joint depth and intensity

The other major difference between color and depth image inpainting is the presence of additional information. Often times when inpainting depth images, we have a corresponding dense color image available. Time-of-Flight sensors inherently produce an intensity image alongside the depth image from the same exact viewpoint. Even stereo and structured light depth sensors can produce such aligned image pairs with minimum processing.

The intensity and depth channels of an image are clearly correlated. For example, depth discontinues very often result in intensity discontinuities. Thus, an aligned intensity image provides additional information that can be used to obtain a better inpainting solution. However, the exact nature of the correlation is non-trivial, *e.g.* depth discontinuities often produce intensity discontinuities but the opposite is less likely. Some approaches have tried to manually codify this relationship. For example, paper III uses a weighting function on the prior based on the gradient of the intensity channel. This paper is reviewed in more detail in Section 4.2.3. To avoid the manual codification of the intensity-depth correlation and the possible inaccuracies that it might produce, other approaches have attempted to learn the correlation from sample data. Paper IV explores this idea further and is described in Section 4.2.5.

4.2 Markov random field models

Markov random fields provide a useful framework for defining priors based on an energy function that clearly separates the noise model and the regularization prior. The energy function is split into a unary data term (U) and a regularization term (R). The data term operates individually per-pixel and models the measurement noise. The regularization term jointly considers the values of all pixels in a neighborhood and models the regularization prior.

Ideally, we would like to use a prior that reflects the statistics of the observed scene and captures all the interactions between the different parts of the scene. However, a compromise is often made and simpler priors are selected to lower the computational complexity and make the problem tractable. Moreover, modelling the statistics of natural images is still a work in progress (Hyvärinen *et al.* 2009). A general version of the energy function has the form

$$E = \sum_{p \in \mathcal{V}} U(x_p) + \lambda \sum_{\mathbf{k} \in \mathcal{N}} R(\mathbf{k}), \tag{19}$$

where λ controls the relative weight between the data and regularization terms. The set $\mathcal{V}$ contains all the pixel positions with measured values. Naturally, for an inpainting problem, $\mathcal{V}$ will have missing values. The data term will have no effect on these missing values and their final value will be solely determined by the regularization term.

However, the model allows the optimization of all values, not only the missing pixels, by taking the noise model into account inside of the data term.

The set $\mathcal{N}$ contains all the neighborhoods of the image. Each vector $\mathbf{k}$ containes the pixel positions of the pixels in one neighborhood. The number of elements $\mathbf{k}$ determines the order of the prior and, together with the shape of R, the type of prior. The order of the prior has a profound impact on what kind of structure the prior can enforce and the computational complexity of the optimization.

4.2.1 1st-order priors

Perhaps, the most common type of regularizer is a first-order prior. Such a prior works on neighborhoods of two pixels. It enforces smoothness in the resulting image by penalizing the difference between neighboring pixels, *e.g.*

$$R_{1\text{st}}(p,q) = |x_p - x_q|^2, \tag{20}$$

where x_p is the actual value for pixel p. In the case of a depth image where the values of x represent distances from the camera, the prior biases the result towards a fronto-parallel structure. This prior clearly does not capture the statistics of depth images. For example, the reconstructed structure depends strongly on the viewpoint since the alignment of a fronto-parallel surface changes as the camera rotates. Moreover, it unnecessarily penalizes slanted surfaces. However, it has been studied extensively due to its simplicity and very efficient optimization techniques exist.

The optimization of energies of this form has been studied under many contexts, *e.g.* stereo disparity estimation (Zhang & Seitz 2007), discrete object labelling, and image segmentation (P. *et al.* 2006). Levin *et al.* (2004) presented a colorization algorithm that uses a first-order prior similar to Eq. 20. They use user-drawn scribbles to fix the color of some pixels, then only the colors of the remaining pixels are optimized. The data term from Eq. 19 is not used and the simple form of the regularization term allows them to solve the problem in closed form. Although it was originally meant for colorization, color and depth share a similar relation to intensity. The algorithm has been directly applied to depth map inpainting (Silberman *et al.* 2012).

Both continuous and discrete optimization have been examined. Since depth maps are often quantized, discrete solutions are especially attractive due to their speed and efficiency. For example, a fronto-parallel plane sweeping approach provides initial guesses that are consistent with the prior and can be easily merged through graph cuts

(Gallup *et al.* 2007). A modern application of 1st order priors to depth image inpainting can be found in Chen & Koltun (2014). They demonstrate how such energies can be optimized very efficiently even with robust energy terms. However, even though Yanover *et al.* (2006) obtained global solutions to low-level vision problems with non-convex pairwise MRFs, their results indicate that pairwise models are incapable of producing very high-quality solutions for stereo problems, thus suggesting that a pairwise prior is insufficient.

A pairwise prior is still attractive due to its low computational cost during optimization. To overcome its fronto-parallel bias and limited expression power, it has been suggested to reformulate the problem as a discrete labelling. For example, if we assume that the scene is piecewise planar, we can obtain candidate planes from the initial point cloud and then assign a plane (and implicitly a depth) to each missing pixel. A simple way of modelling a discrete labelling problem uses the Potts energy model, where the prior has zero cost if the labels are equal or a constant cost λ_P if they are different, *i.e.*

$$R_{\text{potts}}(p,q) = \begin{cases} 0, & \text{if } l_p = l_q \\ \lambda_P, & \text{otherwise,} \end{cases} \tag{21}$$

where l_p is the label of a pixel.

Several depth reconstruction approaches have used a labelling formulation. Furukawa *et al.* (2009) used a strong Manhattan-world prior where only three orthogonal plane directions are allowed. This produces poor reconstructions of natural outdoor scenes but is a good approximation of man-made indoor scenes. Sinha *et al.* (2009) enforces a planarity constraint while allowing for arbitrary plane directions. They recover candidate planes using standard multi-view reconstruction methods and a final depth map is generated by labelling the pixels with the candidate plane labels. Although their approach is flexible in plane direction, it does not take non-planar surfaces into consideration. Gallup *et al.* (2010) extend this approach by including a non-planar label as well as a classifier to detect areas with non-planar texture.

4.2.2 2nd-order priors

Using neighborhoods of three pixels leads to a second-order prior. This is a much more expressive and flexible prior. For example, it can be used to enforce a smooth derivative with a regularization term of the form

$$R_{\text{2nd}}(p,q,r) = |x_p - 2x_q + x_r|^2, \tag{22}$$

where the neighborhood is defined as three consecutive pixels in either vertical, horizontal, or diagonal direction. Such a prior biases the solution towards flat surfaces. Although it is clear that real scenes are not always piecewise planar, assuming a planar structure for areas of constant color is often a good guess and produces plausible results. Moreover, the solution is in theory independent of the viewpoint since no plane orientation is favoured. Therefore, this prior models the environment considerably better than the first-order prior.

Unfortunately, minimizing the energy function under a second-order smoothness prior is also considerably harder. Local methods like gradient descent (Strecha *et al.* 2004) and level sets (Faugeras & Keriven 1998) struggle with long range interactions and often find poor local minima. Finding the true minimum through global methods is untractable because the energy function is non-submodular when triple cliques are used. Kohli (2007) provides an extensive review of the challenges and methods of minimizing higher order energy functions. A direct application of a smooth derivative prior for stereo reconstruction was presented by Woodford *et al.* (2008). It shows that *Quadratic Pseudo-Boolean Optimization* (QPBO) (Boros *et al.* 1991, Hammer *et al.* 1984, Ishikawa & Geiger 2006) can be used to minimize the energy function by merging proposal depth maps. However, in this case, fronto-parallel planes are not sufficient as proposals because of the non-submodularity of the energy function. Thus, generating the appropriate proposals becomes a challenging problem. Woodford *et al.* (2008) use a series of local stereo reconstruction methods with varying parameters to generate noisy depth maps. A series of plane proposals are obtained through RANSAC from these noisy depth maps. Finally, the proposals are merged using QPBO under a second-order prior, thus obtaining the best combination of these local algorithms on a per-pixel basis.

4.2.3 Contribution: a second-order joint prior using QPBO

In Paper III, this thesis presents a method to perform depth map inpainting under a second-order smoothness prior. The aim is to provide a plausible solution to the missing areas. In this case, two requisites are considered to determine a structure as plausible. First, a surface is expected to be continuous and smooth. This is enforced through the smoothness prior. Second, the discontinuities between surfaces are often aligned with color discontinuities. To accomplish this alignment, the smoothness prior is weighted in relation to the intensity gradient. Areas with high intensity gradient have a very small weight for the prior. Thus, when necessary, depth discontinuities are encouraged to be placed along nearby intensity discontinuities. This results in a regularization term of the form

$$R(p,q,r) = W(p,q,r)\rho(x_p - 2x_q + x_r), \tag{23}$$

where W is the weighting term that reduces the influence of the prior in neighbourhoods of high intensity gradient and $\rho(x) = min(|x|, \tau_r)$ is a robust function that truncates the energy to reduce the effect of outliers. This prior has also been called the weak plate model (Blake & Zisserman 1987).

Areas of missing values do not need a data term. However, as discussed in Chapter 2, the pixels on the boundary are often unreliable. To account for this, the missing region is expanded by a few pixels. The values of these pixels are included in the optimization and a data term is included to bias these pixels towards the measured values.

In a similar manner to Woodford *et al.* (2008), the algorithm uses QPBO to merge proposals and fill in the missing regions. The proposals are generated by sampling the boundary of the missing regions and extrapolating candidate planes. This results in plausible solutions that are sharper and more visually pleasing than previous methods. Details of the comparisons can be seen in Paper III.

4.2.4 Higher order priors: Field-of-Experts

Although a second-order prior is a considerable improvement over the simple first-order prior, it still cannot capture the complicated structure of a depth map. Occlusions create long reaching dependences between pixels that can span long distances. A higher-order prior is needed to properly describe such interactions. However, manually crafting a high-order prior is also difficult due to the high-dimensionality of images, their non-Gaussian statistics, and the need to model such long-reaching dependences. Thus, it is highly desirable to learn the prior from example images to avoid the need to carefully craft it by hand. Learning the prior not only avoids the effort involved in manually defining the prior, it also allows the system to learn complicated and non-obvious relationships inside the data that may have been missed by a person.

In order to learn the prior, we need a more flexible representation. We may generalize the notation of Eqs. (20) and 23 by the use of a bank of linear filters $\mathbf{j}_i$. We may express

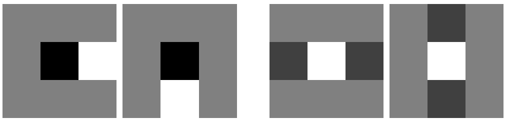


Fig 12. Linear filters for simple priors using a 3×3 neighborhood. Left: 1st-order prior. Right: 2nd-order prior.

our prior as

$$R(\mathbf{k}) = \sum_i \rho(\mathbf{j}_i^\top \mathbf{x}_{(\mathbf{k})}), \tag{24}$$

where the vector $\mathbf{x}_{(\mathbf{k})}$ contains the values of all pixels in the neighborhood $\mathbf{k}$. Figure 12 shows the filters that would produce Eqs. (20) and (23) using a 3×3 neighborhood. The order of the prior is now limited only by the size of the neighborhood and the number of active coefficients in the linear filters.

Roth & Black (2009) developed an MRF framework for learning high-order priors based on this formulation called *Field-of-Experts*. They apply a series of learned filters to overlapping neighborhoods in the image. The output of these filters is processed by a series of corresponding *expert* functions whose parameters are also learned. They follow a probabilistic approach and define the prior probability density for an image $\mathbf{x}$ as

$$p_{\mathrm{FoE}}(\mathbf{x};\boldsymbol{\Theta}) = \frac{1}{Z(\boldsymbol{\Theta})} \prod_{k\in\mathcal{N}} \prod_i \phi\left(\mathbf{j}_i^\top \mathbf{x}_{(k)};\boldsymbol{\alpha}_i\right). \tag{25}$$

Here, the vector $\boldsymbol{\Theta}$ contains all the learned parameters, *i.e*. the filters $\mathbf{j}_i$ and the expert function parameters $\boldsymbol{\alpha}_i$. The function $Z(\boldsymbol{\Theta})$ is a normalizing function that ensures it is a valid probability density. Function $\phi(\cdot)$ is the expert function. Initially, the expert functions where based on the student's T distribution, but an updated version of the framework by Schmidt *et al.* (2010) showed improved flexibility and performance using Gaussian Mixture Models. This probability density results in an energy of the form

$$E_{\mathrm{FoE}}(\mathbf{x};\boldsymbol{\Theta}) = \sum_{k\in\mathcal{N}} \sum_i \phi\left(\mathbf{j}_i^\top \mathbf{x}_{(k)};\boldsymbol{\alpha}_i\right) - \log Z(\boldsymbol{\Theta}), \tag{26}$$

which has a regularization term with the same form as Eq. 24. The normalizing function $Z(\boldsymbol{\Theta})$ has a complicated shape and is expensive to compute. During inference, it is constant and can be ignored, as was done for the methods in the previous sections. However, during learning, the normalizing function changes and must be taken into account.

The complexity of the normalizing function makes Maximum Likelihood impractical for learning. In order to avoid the normalizing function, the FoE framework uses

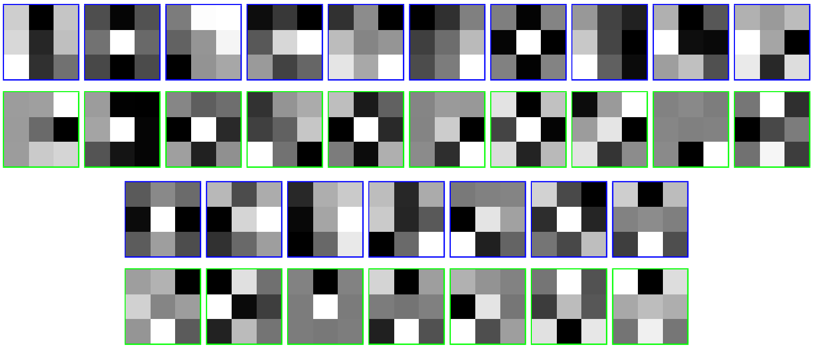


Fig 13. Obtained filters. Blue (1st and 3rd rows) are the intensity filters and green (2nd and 4th rows) are the corresponding disparity filters.

contrastive divergence (Hinton 2002) as a learning rule. Contrastive divergence uses random sampling from the current distribution to estimate the gradient of the model parameters. This results in an approximation to Maximum Likelihood that is much faster and tractable.

4.2.5 Contribution: a MRF-based learned joint prior

Paper IV applies the *Field-of-Experts* framework to the problems of depth map inpainting and depth map upsampling. It extends the framework to jointly model an intensity image and its depth map. The FoE framework has been extended to handle multi-channel images before. Both McAuley *et al.* (2006) and Zhang *et al.* (2007) presented extensions to model color images. However, these extensions increase the number of parameters considerably and previous approaches have had problems to learn them efficiently. For example, McCauley *et al*. only learned the parameters of the potential function and kept the filters fixed, whereas Zhang *et al*. resorted to alternatively optimizing the potential function parameters and the filters separately.

The method presented in Paper IV directly extends the formulation from Schmidt *et al.* (2010) to jointly model an intensity image and its disparity map. Disparity was chosen instead of depth because it better represents the accuracy of the depth sensor used. The joint formulation follows an analogous derivation as the original in Schmidt *et al.* (2010) and can thus also utilize contrastive divergence to learn both filters and the Gaussian mixture model parameters simultaneously.

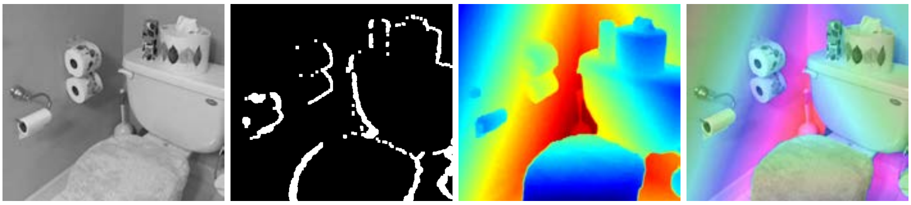


Fig 14. Inpainting results of Paper IV. From left to right: intensity image, mask of missing pixels, inpainted depth map, and depth map overlaid on the intensity channel.

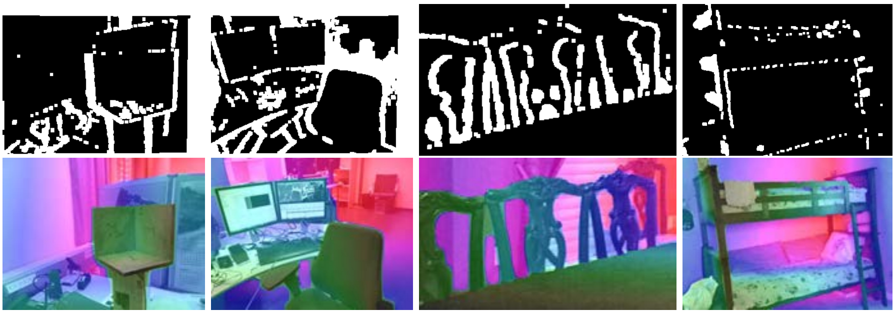


Fig 15. More inpainting results from Paper IV. Top: missing pixel mask. Bottom: depth map overlaid on the intensity channel.

Figure 13 shows the obtained 3×3 filters. Some filters can be interpreted as image derivatives similar to those of Fig. 12. Moreover, sometimes, the shape of the intensity and disparity filters is very similar, but this is not always the case. This suggests that some filters model the case where intensity and disparity edges align, while others model the cases where they do not. These filters demonstrate the complicated nature of natural image statistics.

The obtained prior was used to inpaint holes in semi-dense depth maps. The prior on the missing depth pixels is formulated as a conditional probability density function given the known depths and intensities. Because the missing depth pixels have intensity information and are surrounded by known depth values, this conditional probability provides a strong prior. The final inpainting is performed by sampling from this conditional distribution and obtaining the Bayesian minimum mean squared error estimation. Figures 14 and 15 show some sample inpainting results.

4.3 Non-energy based models

An energy function provides a clear and formal definition of the prior. However, not all inpainting approaches formulate the prior as an energy in the form of Eq. 19. The prior can be implicitly encoded into the inpainting algorithm.

An example of this is inpainting through bilateral filtering (Tomasi & Manduchi 1998). Bilateral filtering is an edge-preserving filtering technique. It can be applied repeatedly to an image with missing regions to propagate the information from the edges into the missing pixels. It has been applied to depth map inpainting by using an intensity image to guide the filtering (He *et al.* 2013). The prior in this case is implicit. The missing values are inpainted by diffusion from the edges, which implicitly enforces similarity between neighboring pixels, and the filtering is guided by the intensity gradient, which aligns depth and intensity edges.

Another important class of algorithms relies on copying existing patches to perform inpainting. It has been shown that most small image patches in a natural image tend to recur redundantly (Zontak 2011). This has been used to perform super-resolution (Glasner *et al.* 2009) and noise-removal (Dabov *et al.* 2006) on color images. It has also been applied to depth map inpainting and super-resolution (Ikehata *et al.* 2013). A similarity function is used to find patches that match the known values surrounding the missing regions and available intensity information. In the case of color images, several patches are merged to better match the surrounding areas. However, in the depth case, this leads to smoothing artefacts. Ikehata *et al.* (2013) use a sparse coding approach to minimize the number of patches to merge.

4.3.1 Contribution: local planar surface models

Paper V presents a depth inpainting algorithm that assumes a piecewise-planar structure. It builds upon the results of 3D reconstruction algorithms that produce an estimate of the surface normal. Algorithms like Furukawa & Ponce (2010) produce a cloud of planar patches (*i.e.* points with a 3D position and normal orientation). These patches can be projected onto one of the source images to produce a semi-dense depth map as seen in Fig. 16.

The algorithm extracts candidate planes and assigns plane labels to the missing pixels. It is closely related to the discrete reconstruction methods presented at the end of Section 4.2.1 (Furukawa *et al.* 2009, Sinha *et al.* 2009, Gallup *et al.* 2010).

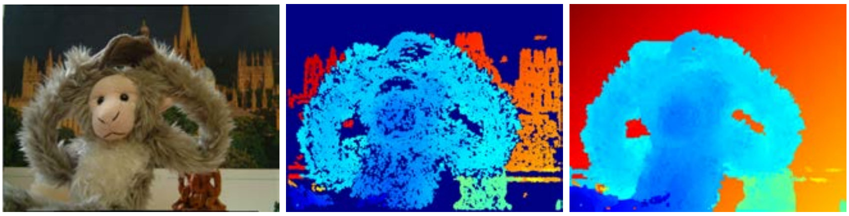


Fig 16. Inpainting results of using a local planar surface model (Paper V). Left: source image. Center: semi-dense reconstruction from Furukawa & Ponce (2010). Right: after inpainting.

However, the formulation is slightly different because it is aimed at inpainting and not full reconstruction. The biggest difference with the aforementioned methods is that Paper V takes a local approach to extract the plane candidates which increases the quality of the candidate planes given a fixed number of possible candidates. Moreover, because we only fill in the missing pixels, we cannot reduce the quality of the reconstruction by incorrectly assigning a non-planar region as planar.

The proposed algorithm uses the reconstructed normal and 3D position of pixels around the boundary of a missing region to generate candidate planes. Support for the candidate planes is determined from all the pixels with available depth. The best candidates are kept and a photometric consistency function is evaluated for each missing pixel. The final depth for each missing pixel is assigned by a single graph-cut labeling using the candidate planes as labels and the photometric consistency as a cost function.

This approach implicitly enforces a second-order prior because of its planarity constraint. However, it is less flexible than the approach presented in Paper III. Yet, for scenes with large planar structures or very small holes, it can produce plausible results with less computations, as seen in Fig. 16.

4.4 Discussion

This chapter presented the image inpainting problem and some of the most popular methods that have been used to approach it. The chapter focused on a formulation of the problem based in Markov random fields. This is not the only formulation possible but it provides a solid mathematical framework and efficient optimization techniques. In fact, many other formulations can be expressed in a MRF framework. Image inpainting is a large area of study in itself but this thesis is mostly concerned with its connection to 3D reconstruction and how it can be applied to depth maps.

Because of the different statistics between intensity and depth images, the inpainting method must be tailored to depth maps or it must learn the statistics from natural images. The algorithms from Papers V and III fall in the former category and that of Paper IV in the latter. The three algorithms explore the compromise between accuracy with a complicated prior and a faster alternative. The MRF formulation of Paper IV is the most flexible and promises better results, but it requires a lot of training data and long training and inpainting times. In contrast, the QPBO-based algorithm of Paper III uses a hand-crafted prior which is less flexible but allows for faster optimization and requires no training.

The idea of using an intensity and depth image jointly for inpainting (Papers III and IV) follows a trend in the proposed contributions of this thesis, namely to use all the information available simultaneously. Given that depth sensors can often provide an aligned intensity image, the information contained in it can help constrain the inpainting problem and improve the result.

5 From depth maps to point clouds

Point clouds are a very useful and common representation of 3D geometry. Sparse 3D reconstruction methods, as seen in Chapter 3, often directly produce a point cloud. Even advanced dense multi-view reconstruction methods often use point clouds as intermediate representations (*e.g.* Vu *et al.* (2012)). Moreover, the depth map produced by a depth sensor can directly be converted to a point cloud when the sensor is calibrated. These point clouds can be the final representation desired or they can be the input used to reconstruct a dense model (*e.g.* a mesh surface). Many of the applications and reconstruction algorithms where point clouds are used are themselves quite involved and require substantial amounts of memory and computation. Oftentimes, they do not scale well to increasing model sizes and resolution.

It is often a challenge to process these point clouds because of the massive amount of data they contain. Large scenes can easily become problematic to reconstruct due to computation and memory constraints. The recent advances in depth sensors have made this problem even more relevant. The Kinect sensor, for example, produces 12.3 million points per second (640 by 480 pixels at 30 fps). Many of these points are redundant, especially when sensor movement is small. Simply merging these individual depth maps into a combined point cloud results in too much data that is not suitable for most applications.

Depending on the application, one may approach the point cloud simplification problem in two ways. One can specify the desired size of the point cloud, thus emphasizing the memory and computational constraints, or one can specify an error bound on the geometric deviation between the initial and simplified clouds, thus emphasizing the accuracy of the model. Imposing hard limits on the size of the model is useful for real-time applications that have limited computational resources, whereas offline reconstruction methods often focus on higher accuracy.

This chapter reviews different ways of reducing the amount of data in the point cloud while trying to keep the maximum amount of information possible. Section 5.1 reviews different approaches that look at the point cloud as a whole and try to simplify it by comparing the result with the original point cloud. Section 5.2 looks at the case when the point cloud is constructed from a series of overlapping depth maps. The contribution of this chapter is described in Section 5.2.4 which consists of a novel point cloud simplification algorithm from Paper VI. The algorithm takes advantage of the input depth map resolution to produce a variable resolution point cloud that contains all the information from the input depth maps but with less redundancy.

5.1 Point cloud simplification

The problem of point cloud simplification can be formulated as follows (Moenning & Dodgson 2004): The input is a set of samples $\mathbf{P}_N = \{p_1, p_2, \ldots, p_N\}$ acquired from a smooth, compact two-manifold surface embedded in $\mathbb{R}^3$. We wish to convert it to a point cloud $\mathbf{P}_M$ so that $M < N$ subject to a user defined refinement condition. The simplified point cloud does not need to be a strict subset of the original. In fact, the simplification process may smooth out noise and result in a point cloud of better quality. Also, the points in the original set may be non-uniformly sampled.

5.1.1 Implicit surfaces

Initial approaches to geometry simplification used a mesh representation of the scene. The explicit connectivity between vertices in a mesh structure can be easily leveraged in these simplification methods and leads to methods that can efficiently process local neighbourhoods (Heckbert & Garland 1997). However, building a mesh structure from a point cloud is an error-prone and computationally intensive process. Topological distortions resulting from incorrect mesh construction can cause topology-preserving simplification algorithms to produce inferior results (Wood *et al.* 2002).

Pauly *et al.* (2002) showed that it is not only possible but also beneficial to perform the simplification directly on the point cloud before constructing the mesh. The basis of this approach is to recognize that a point cloud implicitly represents a connected surface. The explicit vertex connections from a mesh are replaced by spatial proximity of the sample points for sufficiently dense point clouds (Amenta *et al.* 1998).

Levin (2001) introduced a point-based surface representation called *Moving Least Squares* surfaces. The surface is approximated locally at each point in space by a local reference plane and a bivariate polynomial that encodes the offset from the plane. The plane and polynomial are both estimated from the nearest neighbours. This implicitly defines a closed surface around the point cloud.

Okabe *et al.* (2000) proposed an implicit surface based on Voronoi diagrams. They define a geodesic distance between points in the cloud based on *Fast Marching* (Mémoli

& Sapiro 2003). Using this distance measure, they construct a Voronoi diagram that implicitly defines a surface.

Once a surface has been defined the relation of the individual points with the surface can be evaluated. For example, a geodesic distance between points can be calculated and used to determine connectivity. This can be used to then simplify the point cloud. As mentioned by Pauly *et al.* (2002), the simplification algorithms on generic point clouds can be broadly grouped in the following three categories.

5.1.2 Iterative simplification

Iterative simplification methods remove one point at a time until the desired size or error bound is reached. This is similar to mesh-based simplification methods that create progressive meshes (Hoppe 1996). Early work (Alexa *et al.* 2001, Linsen 2001) performed simple point removal at each iteration. The points were ordered according to the error introduced by their removal. This straightforward formulation leads to a decimated point cloud that is a strict subset of the original which is prone to aliasing and often creates uneven sampling distributions.

Pauly *et al.* (2002) used an improved decimation operator that contracts point-pairs. The possible point-pair contractions are rated to select the one that produces the minimum error. The surface is approximated locally by a series of tangent planes and the error metric measures the deviation from the k-neareast neighbouring planes. The point-pairs contract to a single point which effectively resamples the surface.

It is also possible to iteratively add points instead of removing them. Moenning & Dodgson (2004) use an implicit surface based on Voronoi diagrams as mentioned in the previous section. They define a geodesic distance along the implicit surface and use this distance to iteratively add the point farthest away from the current simplified set.

5.1.3 Clustering-based simplification

Another approach to point cloud simplification is based on local clustering of points. Clustering has been extensively used in computer graphics (Rossignac & Borrel 1993) and computer vision (Achanta *et al.* 2012) to reduce the complexity of the input data. The standard approach to point cloud simplification is to subdivide the model's space into grid cells and replace all points inside a cell with a single representative point. Although this method can be very fast it has some serious drawbacks. When using cells of fixed size, the method cannot adapt to non-uniformities in the sampling distribution or different levels of complexity. Moreover, disconnected parts of the surface may be joined if spanned by a single cell. To avoid this, Pauly *et al.* (2002) first proposed clustering the points using an implicit surface on the point cloud. The clusters are built by collecting neighbouring samples with regards to the local sampling density.

Clustering can be achieved in mainly two ways: by region-growing, where points are iteratively aggregated to form clusters, and by splitting, where the point cloud is iteratively partitioned to create progressively finer clusters. Both methods create a set of clusters and each is replaced by a representative sample to create the simplified point cloud.

Region-growing methods typically choose a random point as the starting seed. Nearest neighbours are added to the cluster until it reaches a maximum bound. The bound can be a simple bound on the point count, which enforces a simple constraint on the resulting point cloud size, or a bound on the variation of some property. When the variation on the position is bounded, it results in a curvature-adaptive clustering where more and smaller clusters are created in areas of high complexity. The sequential nature of this type of clustering leads to fragmented clusters where many clusters did not reach the expected bound because their growth was restricted by adjacent clusters (Pauly *et al.* 2002).

Hierarchical clustering methods use a top-down approach. Starting with a single cluster containing the entire point cloud, clusters are recursively split using similar rules as the region-growing methods. If a cluster exceeds a certain bound (*e.g.* size or variation), it is split across the direction of maximum variance as in Brodsky & Watson (2000), Shaffer & Garland (2001). This results in a tree structure whose leaves are the final clusters.

More recently, Song & Feng (2008) proposed a global clustering approach that uses Voronoi diagrams to partition the model space. By choosing a set of representatives, they can partition the space and obtain the clusters. Their insight is that changes in the selection of the representative points only affects the geometry locally. Thus, they are able to efficiently calculate a global geometric deviation error and fine tune the local composition of the clusters.

5.1.4 Particle simulation

A third option to surface resampling is particle simulation. Turk (1992) first proposed this approach for mesh-based surfaces. It was later extended by Pauly *et al.* (2002) to use implicit surfaces and be applied directly to point clouds. An initial set of points is randomly distributed along the surface, then a repulsion force between points refines their position to create a good simplified representation of the original point cloud.

The initial distribution of points plays an important role in the final result. The points are placed randomly using the point density of the original cloud as a probability distribution. Once an initial distribution has been obtained, the process iterates between simulating the point movement due to the repulsion forces and projecting the points back onto the surface. The repulsion forces can be modulated based on the curvature of the model, resulting in more points allocated to more complex areas. The point projection uses the implicit surface to ensure that the points do not introduce any unnecessary artefacts.

5.2 Merging depth maps

The approaches discussed in Section 5.1 simplify a point cloud without taking into consideration the origin of the data. In our case, we are interested in point clouds originating from depth maps that are merged together. Point cloud simplification becomes a considerably important step when we have a depth sensor and we are able to estimate the sensors position. It is then straightforward to convert the depth maps to a series of point clouds that are aligned on the same Euclidean space. The algorithms presented in Section 5.1 could then be used on the merged depth maps, however, there are difficulties with this simple approach.

The biggest problem is memory and computational constraints. Due to the very large bandwidth of modern depth sensors, it is unrealistic to process a large number of frames simultaneously as it would result in too many points. Thus, it is necessary to use algorithms that can filter out redundancy and simplify the point clouds after receiving each depth map.

The perspective nature of most depth sensors means that points observed at different distances represent areas of different size. Points further away from the camera correspond to larger areas than those closer. Some reconstruction systems, like Henry *et al.* (2010), use a *surfel* representation to encode this size difference. However, it is not clear how to merge two surfels that have different sizes but the same position. This is a problem when a surface is observed from different distances. The different depth maps will have varying accuracy and resolution yet must be merged into a single point cloud.

This perspective nature also produces different uncertainties for the reconstructed points. Points closer to the depth sensor are more accurately localized than those farther away. Moreover, the uncertainty is directional. The uncertainty in the direction of the optical axis is much higher than the lateral uncertainty. However, as the camera moves, the direction of the optical axis changes and points are thus reconstructed with very different uncertainties.

5.2.1 Enforcing consistency

Depth maps contain a lot of redundant information that can be used to remove noise and detect outliers. A depth map can be back-projected to Euclidean space and reprojected to another image, thus generating a second depth map from the same viewpoint. Corresponding pixels can then be analysed to determine if they are consistent. If their depth is similar, they are probably separate measurements of the same surface. On the other hand, if the reprojected depth is smaller than the original depth, there is a clear inconsistency. In order to have made the original measurement, the optical ray must be empty from the camera center to the measured surface, yet the second depth map contains a point in between. Thus, only one measurement can be correct.

Merrell *et al.* (2007) presented a depth map merging algorithm that uses these visibility constraints to eliminate outliers. They produce a series of depth maps with corresponding confidence maps using plane-sweeping stereo (Gallup *et al.* 2007). Several depth maps are reprojected onto a reference view and a counting rule is used to determine if a reference point is consistent with the other depth maps. The depth and confidence are then updated using the depths that reproject closer to a given threshold. The consistent depth maps are then merged together into a single point cloud.

5.2.2 Voxelization

Voxelization has become a very popular alternative to surface reconstruction due to the high bandwidth of depth sensors. The most popular approach so far was introduced by Newcombe *et al.* (2011a): KinectFusion. They divide the Euclidean space into voxels using a predefined resolution. The voxel grid contains a discretized version of a

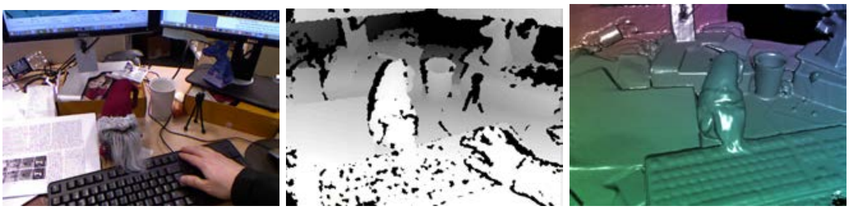


Fig 17. Sample KinectFusion results. Left: color image from Kinect. Middle: a single depth map from Kinect. Right: reconstructed surface using KinectFusion (Newcombe *et al.* 2011a).

signed distance function (SDF), *i.e.* each voxel contains the signed distance from the voxel's center to the reconstructed surface. This implicitly defines a surface that can be efficiently extracted from the grid.

More importantly, the SDF can be very efficiently updated when new measurements are available. Each depth map obtained is integrated into the grid by estimating the camera position and back-projecting the measured depths into the grid. Each measurement updates a group of voxels at the measured depth and along its optical ray (the SDF is truncated to reduce computation). In this way, the depth maps are quickly merged into a single surface and the amount of data is kept constant.

This approach produces very convincing results, see Fig. 17. It is able to eliminate most of the noise from the input depth sensor and fill holes by combining depth maps from different points of view. However, using a fixed resolution for the grid limits the amount of detail that can be modelled and truncates the scene to the grid edges. More recently, the approach has been extended to large-scale reconstruction by defining the working grid as sliding window on a larger volume (Whelan *et al.* 2012, Chen *et al.* 2013), but the fixed resolution still limits the amount of detail in any given area.

5.2.3 Multi-resolution voxelization

Although there have been many surface reconstruction methods proposed in the literature, few of them have considered the effect of multi-resolution sampling. It is common that different areas of the scene are observed with different resolution or scale. Areas of interest are often observed closely at high-resolution, whereas other areas are sampled more coarsely. Low-resolution samples usually correspond to a low-pass filtered version of the original surface (Klowsky *et al.* 2012), which results in a lower quality reconstruction when high-resolution and low-resolution samples are mixed as equals.

Fuhrmann & Goesele (2014) present a formulation similar to that of KinectFusion (Newcombe *et al.* 2011a) where they aggregate all samples to evaluate an implicit function, however, instead of a sample casting a single vote on a single voxel, a sample's influence is distributed according to their scale. Moreover, to account for the variable resolution, an octree is used instead of fixed-resolution grid. The scale of the sample is determined from the characteristics of the capture device. In general, samples further away from the device have a larger scale (lower resolution). The final surface can then be recovered from the zero-crossings in the octree with sub-voxel accuracy. This method has shown very good quality results. Although it is now tailored the dense surface reconstruction problem, it might be possible to adapt it to point cloud simplification.

5.2.4 Contribution: modelling uncertainty and adaptive point resolution

Paper VI presents a depth map merging algorithm that addresses some of the issues discussed so far. It takes in depth maps sequentially and merges them into a point cloud with the explicit purpose of eliminating redundant information. The algorithm presents two main contributions. First, it supports an adaptive 3D resolution by using the image space grid naturally determined by the input image resolution. Second, it models the position uncertainty to improve point merging.

The algorithm considers the depth maps sequentially. When a new depth map is received, the relevant area of the current point cloud is projected onto the image space. The new measurement is only compared to those points projecting onto its corresponding pixel. This effectively results in an adaptive 3D resolution. As the camera moves towards the surface, the point density becomes sparser and new points will not be merged together, thus maintaining the highest level of observed detail without introducing redundancy. Figure 18 shows the results of algorithm applied to a reconstruction of a scene. The algorithm applies less subsampling in areas with less input data.

The points are modelled with position and a 3D Gaussian uncertainty to take the sensor noise model into account. This not only gives an idea of the expected noise of the point cloud, but the algorithm is also able to determine if two measurements correspond to the same surface in a more formal probabilistic fashion. When a point is projected onto the same pixel as a measurement, the result of their potential merging is calculated. If the Mahalanobis distance between both of the original points and the merged result is below a set threshold, the points are said to intersect. They are then merged together using the best linear unbiased estimator which results in an improved

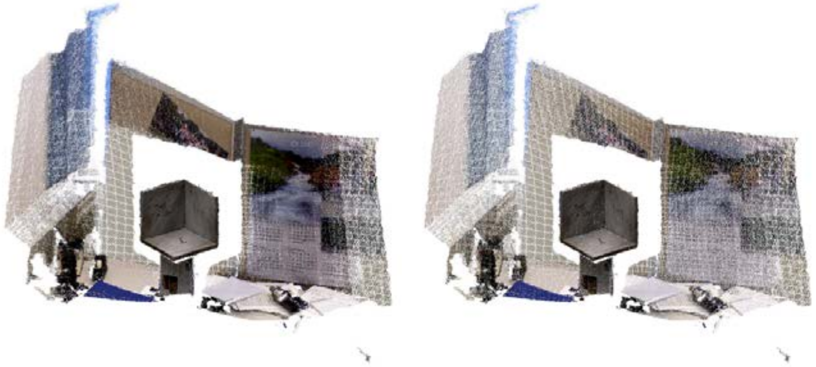


Fig 18. Results of Paper VI. Left: simple merging. Right: result from Paper VI. The merging algorithm corrects the non-uniform sampling behind the cube.

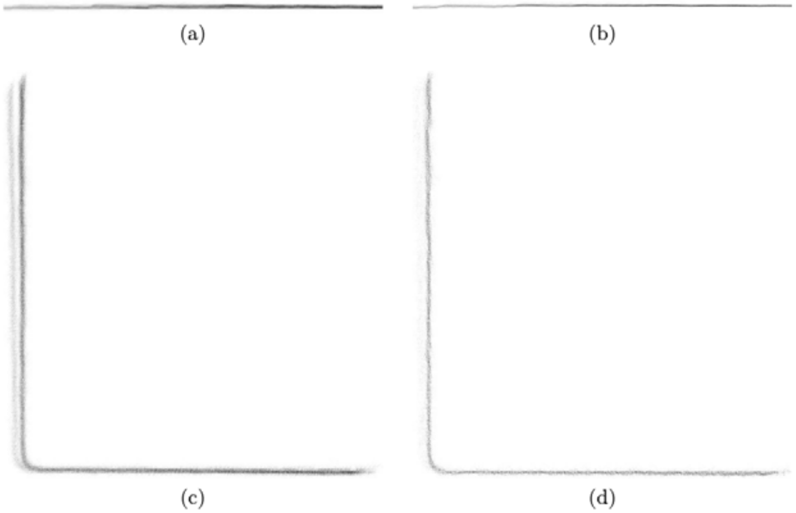


Fig 19. Noise-removal results of Paper VI. Left: planar structures after simple merging. Right: after merging the variation is reduced (thinner structures). Top: single plane from a whiteboard. Bottom: cross-section of a three-sided cube.

position and reduced uncertainty. Figure 19 shows how the noise of the model is reduced after merging. Paper VI contains more detailed results.

5.3 Discussion

Point clouds have proven to be a common way to represent 3D information and are therefore an interesting topic. They are closely related to depth maps since one can easily transform one into the other. However, they are not limited by image resolution as depth maps are and can thus grow without limit. This chapter explored some of the options to reduce the size of point clouds without reducing the quality of the 3D structure.

The final algorithm used will depend on the application. For example, volumetric approaches (Section 5.2.2) regularize the sampling of the data and limit resolution, which could be useful or negative depending on the application. In particular, Section 5.2 showed that if the information is known to come from depth maps, we can use this to improve the simplification algorithm.

The contributions presented in Paper VI show such improvement. By using the natural 2D resolution of the images as an upper bound on the 3D resolution of the point cloud, the algorithm can eliminate redundant data and noise without sacrificing quality. Although the complexity of the proposed algorithm is too high for a real-time implementation, similar ideas could be used to design a system that discards redundant information from a point cloud capturing system in real-time.

6 Application: Free-viewpoint rendering

The depth maps and point clouds produced by the methods discussed in the previous chapters have many potential applications, *e.g.* exploring photo collections (Snavely *et al.* 2007, Kushal *et al.* 2012), augmented reality (Reitmayr *et al.* 2010), virtual tours (Xiao & Furukawa 2014), and image-based rendering (Shum *et al.* 2007), among others. This chapter focuses on one particular application of 3D reconstructed geometry: *Free-viewpoint rendering* (FVR). The goal is to allow the user to view the scene from a novel viewpoint, one not directly captured. The system should be able to synthesize a visually plausible image from the captured images that corresponds to the novel viewpoint. This can produce a more interactive user experience for both images and videos. For example, panoramas can exhibit structure by interactively displaying parallax (Zheng *et al.* 2007) and replays for sports event can be viewed from arbitrary angles (Shum *et al.* 2007).

Free-viewpoint rendering is part of the field called image-based rendering (Shum *et al.* 2007). Image-based rendering covers techniques used to produce novel images from real images. It includes techniques such as view interpolation, view-dependent texture mapping, image morphing, light field rendering, and others. Many of these techniques have been used to synthesize an image that corresponds to a novel viewpoint. However, the simpler methods, like view-dependent texture mapping (Debevec *et al.* 1996), require some human interaction during processing and might limit the type of novel viewpoints that can be synthesized. For example, methods based purely on image morphing (Chen & Williams 1993) can only generate viewpoints form inside the visual hull of the captured camera centers.

There is a fundamental trade-off between space and complexity requirements in view synthesis methods. Light field-based methods can produce high quality images without reconstructing the scene but require a lot of data, whereas image warping-based methods use correspondences and scene structure to allow sparser sampling. Section 6.1 presents a brief summary of light field rendering methods. Section 6.2 explores the different types of image warping-based approaches and how the final image is synthesised. Section 6.3 explores the problem of transparency in FVR and the contributions made by Paper VII in this area. In particular, Section 6.3.2 presents an algorithm to estimate the transparency of foreground objects in a multi-view dataset. This information is then used in a transparency-aware free-viewpoint rendering algorithm to create images from novel viewpoints without transparency artefacts.

6.1 Light field-based rendering

Light field-based methods avoid modelling the structure of the scene in favour of a sampling-based approach. The distribution of light in an environment is described by the *plenoptic function* (Adelson & Bergen 1991). The plenoptic function is a five-dimensional quantity $L(x, y, z, \theta, \phi)$ that describes the flow of light at a given position (x, y, z) and direction (θ, ϕ) [2]. The plenoptic function contains all information necessary to render any image from any viewpoint. In fact, an image from a central camera is a discrete sampling of the plenoptic function at a fixed position, *i.e.* the camera center, where the directions sampled correspond to the directions of the optical rays. However, a dense sampling of the plenoptic function is often impractical.

The dimensionality can be reduced from 5D to 4D using a *light field*. The basic assumption behind it is that the intensity of a light ray does not change as it travels through empty space. We can define an arbitrary surface in front of the scene, *e.g.* an infinite plane, such that there is only empty space between the camera and the surface. Once the light rays cross the surface, their intensity will not change before they reach the camera. Thus, we can define the light field as a four-dimensional function $L(u, v, \theta, \phi)$ that describes the flow of light through a given position in the surface (u, v) with direction (θ, ϕ). The light field contains all necessary information to render any image from any viewpoint behind the surface.

The light field was first used for synthesising novel views by Gortler *et al.* (1996) and Levoy & Hanrahan (1996). They simultaneously proposed a method to capture a light field and synthesise novel views. Gortler *et al.* (1996) resample the light rays captured through images onto a regular grid, resulting in a very fast rendering algorithm. Buehler *et al.* (2001) later extended their work to use unstructured lumigraphs to avoid the resampling. The unstructed lumigraph rendering technique was recently improved on by Davis *et al.* (2012) and coupled with a SLAM system to provide feedback during capture.

Modern light field-based methods can produce very high quality renders (Davis *et al.* 2012, Kim *et al.* 2011). They are especially useful in 3D movie production since they can give editors fine control over disparity (Kim *et al.* 2011). It has also been explored

[2] Considering time, wavelength, and polarization adds more dimensions. They are omitted here for simplicity.

to use light fields as input to reconstruct high quality depth maps (Kim *et al.* 2013). However, many applications cannot use light fields as inputs due to their large size.

Sampling the 4D light field function is more practical than sampling the 5D plenoptic function. Yet, it is still a complicated task. A rather dense sampling is required to avoid artefacts. Gortler *et al.* (1996) suggested incorporating geometry information to compensate for parallax, thus reducing ghosting artefacts when the sampling density is not high enough. This captures a fundamental trade-off in view synthesis methods: simpler rendering methods require a prohibitively large amount of data to produce high quality images, whereas methods that use scene structure or correspondences can cope with sparser sampling.

6.2 Image warping-based rendering

In most scenes, much of the information contained in a light field is redundant. A Lambertian surface, for example, exhibits the same appearance when observed from different directions. Thus, all light rays originating from the same surface point will contain the same information. When these surfaces are imaged by several cameras, they will appear in different parts of the image due to scene movement and depth-induced disparity, but their appearance will stay the same. By modelling how the surfaces appear to move in image space, we can warp the captured images to synthesize a novel viewpoint.

There are two leading ways of modelling the image deformation between views. Section 6.2.1 explores how the warping can be modelled directly in the 2D image space by finding correspondences between images or calculating the optical flow. Alternatively, camera calibration and depth maps can be used to backproject the pixels to 3D space and reproject them onto a novel viewpoint, a technique known as depth-image based rendering and discussed in Section 6.2.2. Finally, regardless of the warping method used, once the different input images have been warped to match the novel viewpoint they need to be merged to synthesise the final image. This is reviewed in Section 6.2.3.

6.2.1 Image warping

Rendering a novel view using a few images of a scene is a long standing research topic. The first approaches tried to solve the problem purely in 2D image space. The seminal paper of Chen & Williams (1993) laid the basis for view interpolation based on image

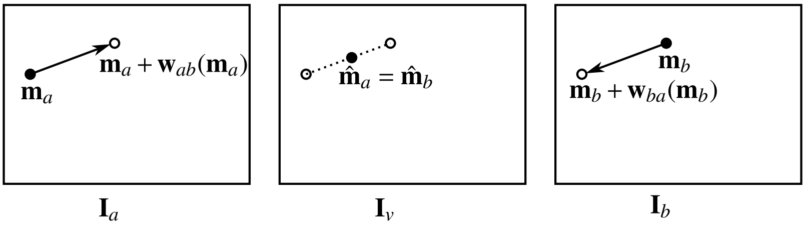

$\mathbf{m}_a + \mathbf{w}_{ab}(\mathbf{m}_a)$
$\mathbf{m}_a$
$\hat{\mathbf{m}}_a = \hat{\mathbf{m}}_b$
$\mathbf{m}_b$
$\mathbf{m}_b + \mathbf{w}_{ba}(\mathbf{m}_b)$
$\mathbf{I}_a$
$\mathbf{I}_v$
$\mathbf{I}_b$

Fig 20. Image warping. The position of a pair of corresponding pixels $\mathbf{m}_a$ and $\mathbf{m}_b$ is interpolated to obtain the warped positions $\hat{\mathbf{m}}_a$ and $\hat{\mathbf{m}}_b$ in the novel image $\mathbf{I}_v$.

correspondences. Their rendering formulation was later used for dynamic real-world recordings (Lipski *et al.* 2010, Lipsky *et al.* 2011). The basic algorithm takes two input images $\mathbf{I}_a$ and $\mathbf{I}_b$ and interpolate an image $\hat{\mathbf{I}}_{ab}$ that corresponds to a viewpoint between those of the input images. The algorithm requires a dense correspondence map between the input images. The mapping $\mathbf{w}_{ab}$ relates a pixel position $\mathbf{m}$ from image a to its corresponding position in image b

$$\mathbf{m}_b = \mathbf{m}_a + \mathbf{w}_{ab}(\mathbf{m}_a). \tag{27}$$

The mapping is directional and many-to-one due to occlusions and ambiguities, so a pair of mappings is needed for a pair of input images. The mapping $\mathbf{w}_{ba}$ gives the mapping in the other direction. Using these mappings, it is possible to linearly interpolate the position for the pixels for any viewpoint in between, see Fig. 20. For an interpolation factor $\alpha \in [0 \ldots 1]$, the interpolated position for a pixels in the source images are simply

$$\hat{\mathbf{m}}_a = \mathbf{m}_a + \alpha \mathbf{w}_{ab}(\mathbf{m}_a) \qquad \text{and} \tag{28}$$

$$\hat{\mathbf{m}}_b = \mathbf{m}_b + (1-\alpha)\mathbf{w}_{ba}(\mathbf{m}_b). \tag{29}$$

Using the warped pixel positions, we copy the pixel information to produce a warped image, $\hat{\mathbf{I}}_a(\hat{\mathbf{m}}_a) = \mathbf{I}_a(\mathbf{m}_a)$. Through this forward-warping process, we obtain two images $\hat{\mathbf{I}}_a$ and $\hat{\mathbf{I}}_b$ that have been warped to the desired viewpoint. However, these images are incomplete. Forward-warping and depth-induced disocclusions lead to cracks and holes, *i.e.* areas of $\hat{\mathbf{I}}_a$ that do not receive any pixel information from $\mathbf{I}_a$. Merging these images into a final image $\hat{\mathbf{I}}_{ab}$ is later analysed in Section 6.2.3.

Although simple and intuitive, this method has serious shortcomings. Obtaining a dense mapping between two images is a complicated task. Optical flow methods and feature descriptors have been used for this (Eisemann *et al.* 2008), yet it still remains an unsolved problem. Manual correction has been needed to achieve visually convincing results (Klose *et al.* 2011). Moreover, the real path that a pixel takes between viewpoints a and b is most of the time not linear. As was already pointed out by Chen & Williams (1993), the real path follows a projective transformation and is only linear in the special case of camera movement parallel to the image plane. Using a linear model is only an approximation used either to reduce computation or because the projective transformation cannot be recovered.

6.2.2 Depth-image based rendering

Taking into account the 3D structure of the scene can make the problem of obtaining a dense mapping much easier. General optical flow is a severely ill-constrained problem, whereas the epipolar geometry of calibrated cameras reduces the correspondence search area from the entire image to a line segment (Hartley & Zisserman 2000). More importantly, if we recover the depth of a pixel, we can easily obtain the true projective transformation that it undergoes as the camera moves.

Using 3D reconstruction methods and/or depth sensors, we can calibrate the cameras used for view interpolation and obtain a depth map for each image. This simplifies the warping and avoids the need for correspondences[3]. Each input image can be independently warped to the novel viewpoint by backprojecting the pixels to 3D space and reprojecting them onto the novel image plane, see Fig. 21. This is known as *depth-image based rendering* (DIBR).

Given a calibrated camera and a known depth d, we can backproject a pixel $\mathbf{m}$ to a unique point in 3D space, $\mathbf{x} = \mathcal{P}^{-1}(\mathbf{m}, d)$. Thus, a pixel from image a can be warped onto a novel viewpoint v by

$$\hat{\mathbf{m}}_a = \mathcal{P}_v(\mathcal{P}_a^{-1}(\mathbf{m}_a, d_a)). \tag{30}$$

The warping produced by Eq. 30 corresponds to the projective transformation induced by the scene geometry plus the cameras' distortion, which is a more accurate model than the linear image-based warping. Moreover, when using depth-based warping, pixels are transferred with depth and occlusions can be easily resolved using a *soft z-buffer* (see Section 6.2.3.

An extensive survey on free-viewpoint video with an emphasis on DIBR has been conducted by Smolic (2011). Starck *et al.* (2009) presented a combination of capture stages and algorithms for DIBR. The use of 3D structure implies either a static scene

[3] Some 3D reconstruction methods may require correspondences but not all. Depth sensors can avoid the error-prone correspondence matching problem.

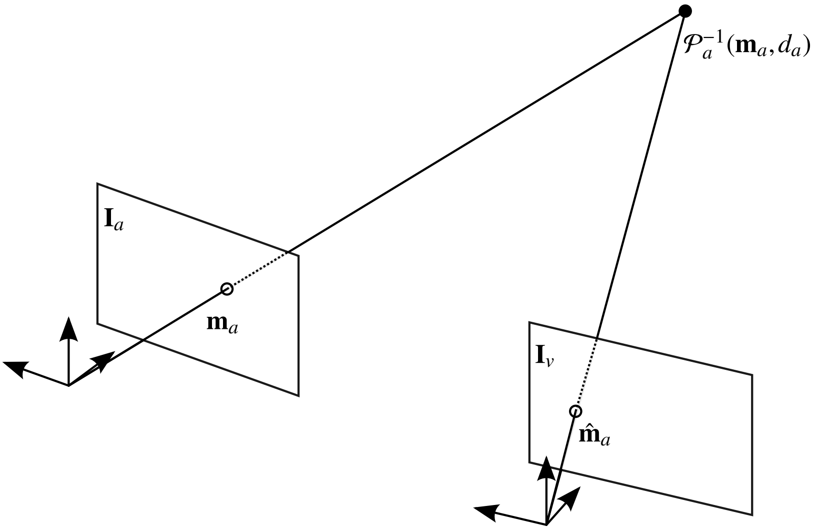


Fig 21. Depth-image based warping. All pixels are backprojected to 3D space and reprojected onto the novel image plane.

or synchronized cameras. Dense arrays of synchronized cameras have long been established as a way to automatically create depth-based free-viewpoint video (Zitnick *et al.* 2004). A high quality 3D reconstruction is critical to produce high-quality renders. High-resolution laser scans have been shown to improve the final quality considerably (de Aguiar *et al.* 2008).

Recently, is has proven to be successful to use a hybrid approach that combines correspondence-based and depth-based warping. Lipski *et al.* (2014) presented a method that backprojects points to 3D space and linearly interpolates correspondences in 3D space before reprojecting to the virtual viewpoint, as seen in Fig. 22. This eliminates ghosting artefacts and provides a smoother transition between images when the reconstructed geometry is not accurate. They demonstrated a system that generated production-quality images with no human interaction.

6.2.3 View synthesis

Once the input images have been warped to the novel viewpoint, they must be merged to create the final image. Using the blending factor α, we can obtain a straightforward blend of both warped images

$$\mathbf{I}_v(\mathbf{m}) = \alpha\hat{\mathbf{I}}_a(\mathbf{m}) + (1-\alpha)\hat{\mathbf{I}}_b(\mathbf{m}). \tag{31}$$

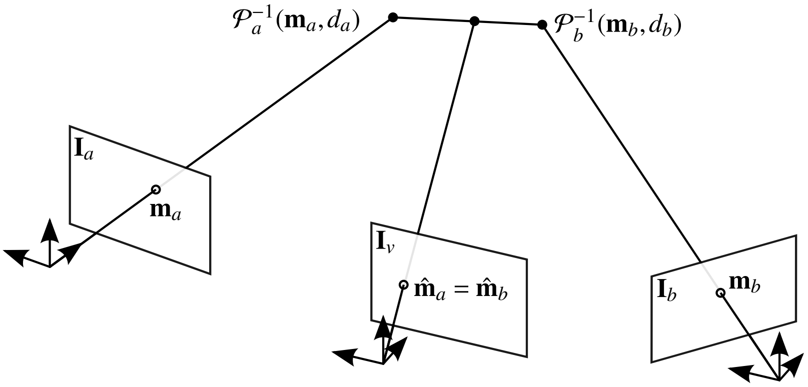

$\mathcal{P}_a^{-1}(\mathbf{m}_a, d_a)$
$\mathcal{P}_b^{-1}(\mathbf{m}_b, d_b)$
$\mathbf{I}_a$
$\mathbf{m}_a$
$\mathbf{I}_v$
$\hat{\mathbf{m}}_a = \hat{\mathbf{m}}_b$
$\mathbf{I}_b$
$\mathbf{m}_b$

Fig 22. Hybrid warping (Lipski *et al.* 2014). Pixels are backprojected to 3D space, their position is linearly interpolated in 3D space, and then reprojected onto the novel image plane.

Unfortunately, the warped images $\hat{\mathbf{I}}_a$ and $\hat{\mathbf{I}}_b$ are incomplete due to occlusions and missing regions. Most of the time, there will be information in at least one of the input images. In this case, we can fix α to 1 or 0 to use only the source pixel that has valid information. However, a sudden change in the blending weight often leads to visible artefacts due to mismatched exposures or white-balance between input images. To avoid this, a per-pixel weighting of the input views is used to enforce a smooth transition over the image. For example, Pulli *et al.* (1997) defined it as a combination of three weighting factors: view proximity, sampling density, and a weight to smoothly blend boundaries.

Not all pixels should be blended, however. Points that project to the same pixel may come from different surfaces. This is resolved through the use of a *soft z-buffer* on the destination image (Pulli *et al.* 1997). To account for noise on the depth component, points that project to the same pixel with a similar depth are merged, but if a point with a considerably larger depth is found, it is discarded.

Another source of rendering artefacts is missing pixels. Disocclusions may lead to areas in the novel image that are not observed in any source image. Inpainting methods can produce visually plausible results. Very small holes can be filled by copying the values from neighbours. The pixels copied must be those with a larger depth to avoid *fattening* the foreground object (Pulli *et al.* 1997). Larger holes can be filled with more advanced inpainting techniques, *e.g.* Criminisi *et al.* (2003).

The techniques described so far have been used to create very high-quality renderings that allow the artists to create cinematic effects after capture (Zheng *et al.* 2009). As an alternative to warping-based rendering, Fitzgibbon *et al.* (2003) proposed to formulate the novel view rendering as a color reconstruction problem while using an image-based

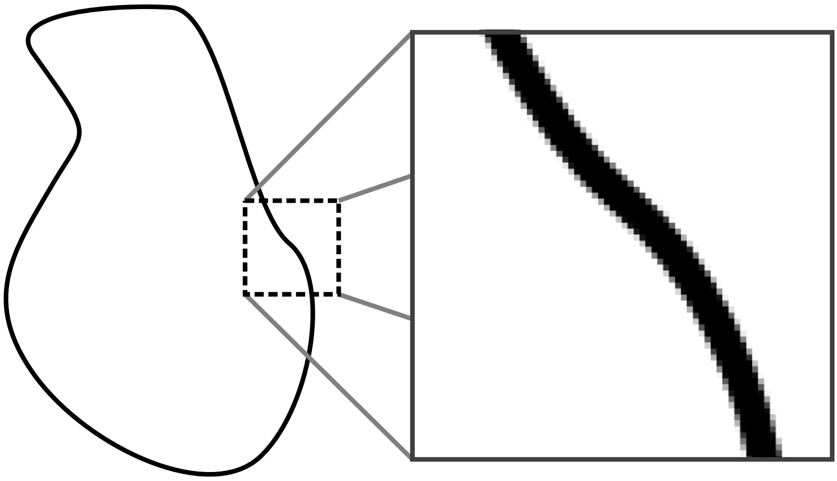


Fig 23. The discrete nature of cameras creates mixed pixels at object boundaries. A black object on white background creates pixels with intermediate gray values when the foreground object does not cover the entire pixel area.

prior. It is an interesting alternative because it avoids the reconstruction of depth and performs the rendering directly.

6.3 Transparency in FVR

Transparency is a major source of artefacts for free-viewpoint rendering. Naturally, these artefacts can be caused by transparent or translucent materials, but even when the scene contains only opaque surfaces, transparency artefacts are common due to mixed pixels. Pixels whose area is not covered entirely by the foreground surface result in a blend between foreground and background, as seen in Fig. 23. Transparency, due to both surface material and mixed pixels, is often view-dependent and when semi-transparent pixels are warped onto a novel image, they lead to ghosting. The outline of foreground objects can often be seen on the background layer, as in Fig. 24.

Most 3D reconstruction algorithms and depth sensors do not take transparency into account. Images have only a single color per pixel and depth maps only a single depth. Semi-transparent pixels may arbitrarily be assigned to the foreground or background object, depending on its appearance in the image. Either case may lead to ghosting, as seen in Fig. 24 because the warping does not take into account the view-dependent transparency. This is a challenging problem since the transparency can have different causes and the object boundaries are often complex (*e.g.* hair). To cope with this issue,

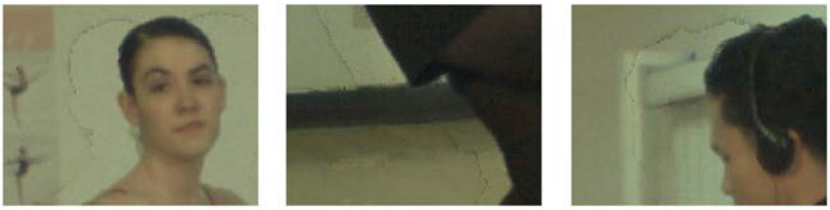


Fig 24. Ghosting artefacts due to semi-transparent pixels using the simple merging from Müller *et al.* (2008).

image matting algorithms can be used to estimate the true colour of the foreground object and its transparency.

6.3.1 Image matting

Image matting is the process simultaneously estimating the transparency and true color of a foreground object often without knowing the exact color of the background. The underlying idea is to model the color of a pixel **m** as a combination of foreground and background layers, **F** and **B** respectively. Because we model each pixel independently the index **m** is omitted. The color **c** of the pixel is modelled by

$$\mathbf{c} = \alpha\mathbf{f} + (1-\alpha)\mathbf{b}, \tag{32}$$

where α is the transparency factor. In the case of an RGB color image, a pixel provides three measurements and seven unknowns. Clearly, when using only a single image, matting is a severely ill-posed problem. Heavy priors are needed to solve it, as seen in the benchmark by Rhemann *et al.*rhemann2009.

For a single color image, the segmentation of foreground and background is ambiguous. Thus, most of the algorithms require manual segmentation of the image. The most common form of input is a trimap that segments the image into foreground, background, and unknown areas. The matting algorithm then computes the transparency value for the unknown area. Some algorithms accept a sparse labelling with only a few foreground and background scribbles. Nevertheless, these scribbles are often used to generate a proper trimap (*e.g.* Bai & Sapiro (2007), Juan & Keriven (2005), Rhemann *et al.* (2008)).

Single view matting methods can be roughly divided into three categories: color sampling, alpha propagation, and optimization methods. Color sampling methods take samples from nearby labelled regions and assume color smoothness to interpolate alpha values between the labelled regions (*e.g.* shared sampling (Gastal & Oliveira 2010)).

Alpha propagation methods assume that the alpha values are correlated with some local image statistics (*e.g.* closed form matting (Levin *et al.* 2007)). Optimization methods combine the previous two approaches to exploit their strengths (*e.g.* robust matting (Wang & Cohen 2007)).

Although very impressive results have been shown for single view methods (Rhemann *et al.* 2009), it is expected that multi-view methods would improve their results, since more information is available and several observations of the same point can be used. Moreover, the foreground and background labelling can be done automatically when depth information is available.

Hasinoff *et al.* (2006) propose a multi-view method that uses boundary curves in 3D space to estimate transparency. Their system detects mixed pixels assuming that the objects are opaque. Joshi *et al.* (2006) use a multi-view variance measure to estimate transparency. It internally computes a trimap based on the observed variance and propagates color statistics. Wexler *et al.* (2002b) estimate the complete light transfer function of a foreground object (*e.g.* a magnifying glass) from multiple views in a probabilistic fashion by assuming a planar background. In a similar work, Wexler *et al.* (2002a) approach the alpha matte estimation under a Bayesian framework but limit the model to planar layers and model the transparency as view independent, which is not suitable for mixed boundary pixels.

In the context of free-viewpoint rendering, Zitnick *et al.* (2004) presented an algorithm that estimates the alpha matte along depth discontinuities. It uses a variant of Bayesian matting (Chuang *et al.* 2001) to estimate colors and transparency for mixed boundary pixels. Although the stereo and rendering is multi-view, the matting is performed using a single view using a fixed-width boundary region, which limits the applicability in scenes with large semi-transparent regions. A popular approach is to discard the mixed pixel and only use the information from neighbouring views (*e.g.* Müller *et al.* (2008)), however, this discards information, suffers from unnaturally sharp boundaries, and still produces artefacts for complicated semi-transparent regions.

6.3.2 Contribution: multi-view alpha matte for FVR

Paper VII introduced a multi-view matting method that is able to recover the true color of foreground objects and the observed transparency. It uses linear constraints on RGB space obtained from all the views that observe a given point. Eq. (32) must be updated

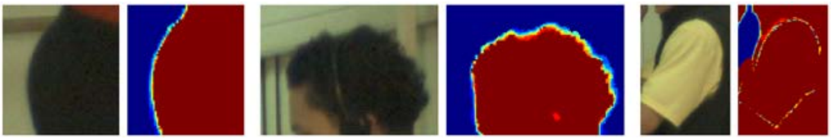


Fig 25. Close up of alpha maps produced by Paper VII. Left: Boundary pixels. Middle: Semi-transparent hair. Right: Incorrect estimation due to mixed pixels in the foreground with a similar background colour.

to reflect the contribution of each view. Thus, for a point in 3D space observed in view i

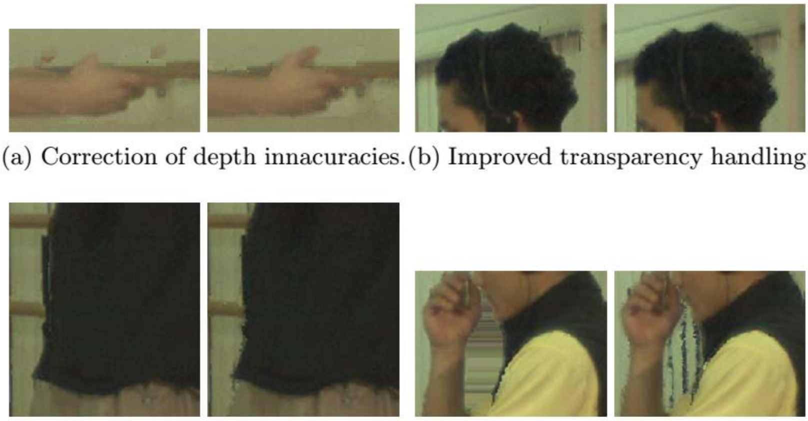
$$\mathbf{c}_i = \alpha_i \mathbf{f} + (1 - \alpha_i)\mathbf{b}_i. \tag{33}$$

The observed color, the transparency, and background are view-dependent but the true color of the foreground object is constant. Moreover, by intersecting the optical ray with the depth maps of neighbouring images, we are able to directly observe the background color. Thus, if a foreground point is observed by N views, it will have $3N$ constraints and only $3+N$ unknowns (the constant foreground color and one alpha value for each view).

The endpoints $\mathbf{f}$ and $\mathbf{b}_i$ of Eq. (33) define a line segment in RGB space. Because α_i and $\mathbf{f}$ are unknown, each constraint obtained defines a ray in RGB that starts at the background color $\mathbf{b}_i$ and passes through the observed color $\mathbf{c}_i$. The foreground color $\mathbf{f}$ must lie at the intersection of these constraints. Once the foreground color is determined, we can estimate each view's alpha value by determining the position of the observed color along the line segment. Paper VII describes heuristics to handle background colors similar to the foreground color or to each other.

The algorithm of Paper VII begins by calculating a background layer $\mathbf{B}$ for each image by taking samples from all neighbouring images. Once the background is determined, linear constraints in RGB space are assembled for each pixel, the foreground color is determined, and the alpha value is finally calculated. To remove the effect of noise, a final smoothing step is applied by enforcing a low alpha gradient where the intensity gradient is also low.

Figure 25 presents some of the alpha maps extracted for semi-transparent regions. The obtained foreground, background, and alpha layers were applied to the problem of free-viewpoint rendering to generate novel views. Figure 26 shows a comparison of the render results of Paper VII with the method of Müller *et al.* (2008). The proposed method is fully automatic and correctly recovers the information from semi-transparent materials and mixed pixels.



(a) Correction of depth innacuracies. (b) Improved transparency handling.

(c) Removal of line artifact on left border. (d) Naïve hole filling vs. recovered background.

Fig 26. Comparison of synthesised views from a novel viewpoint. Left: Müller *et al.* (2008). Right: Results of Paper VII.

6.4 Discussion

This chapter presented a concrete application of the 3D reconstruction methods and sensors explored in the previous chapters: free-viewpoint video. The extensive literature in the field shows that FVR is an attractive application and still in development. A major conclusion from the surveyed literature is that using 3D information significantly improves the quality and speed of FVR methods. The calibration and 3D reconstruction methods of Chapters 2 and 3 can, thus, be useful in the context of FVR. Dense depth maps are a default format for representing and transmitting the 3D structure, which validates the need for methods presented in Chapter 4.

Section 6.3 showed that transparency is unavoidable even in simple scenes and is a major source of artefacts in FVR. Traditional reconstruction methods do not take transparency into account and need to be augmented to model it. The contributions of Paper VII directly address this issue. The multi-view alpha matting approach proposed creates a multi-layer depth map with transparency. It has the potential to correctly handle a wider range of transparency cases than single-view approaches.

It remains an issue that transparency and 3D reconstruction are handled separately. After all, transparency directly affects the appearance of surfaces and 3D reconstruction methods often use this appearance as a basic constraint. A method that simultaneously estimates 3D structure and transparency is expected to improve over an ad hoc combination of individual reconstruction and transparency estimation methods.

7 Summary and conclusion

This thesis has presented novel algorithms as well as practical considerations for several components of a 3D reconstruction pipeline. The components examined were camera calibration, simultaneous localization and mapping, depth map inpainting, depth map merging, and transparency estimation. These cover a wide range of topics and the contributions may, at first, seem disconnected to each other. However, as described in previous chapters, they can all be applied to the general reconstruction problem and thus follow a common direction, *i.e.* to recover information about a scene from images using strong geometric and natural priors.

The first topic discussed was camera models and calibration in Chapter 2. A practical and accurate algorithm was proposed to jointly calibrate a depth and color camera pair. The algorithm uses only a planar calibration target, thus avoiding the need to detect noisy depth discontinuities in depth images. The algorithm was applied to the Kinect sensor and a novel distortion model was also introduced. The combination of the distortion model and the calibration algorithm improved the reconstruction accuracy of the device at close range. These developments were implemented and publicly released as a Matlab toolbox to be used by the computer vision community. From a practical standpoint, this was a considerable contribution since at the time there were no other comprehensive calibration methods for the Kinect.

Chapter 3 reviewed the general theory behind 3D reconstruction and presented a system for simultaneous localization and mapping for mobile platforms. The system introduced a method to enforce constraints on a camera's pose based on the observed features in an image. The method can simultaneously utilize features that have been triangulated and those that have not, thus improving accuracy over systems that can only use triangulated features. Moreover, it supports more generic camera motions, such as pure rotations without camera translation. This complete SLAM system was also released as open source for the computer vision community to use as a development and research framework.

Depth sensors and many reconstruction algorithms produce only semi-dense reconstructions and oftentimes applications require dense depth maps. Because of this, Chapter 4 examined the problem of image inpainting and presented three different depth map inpainting algorithms. Each algorithm uses a different prior to solve the inpainting problem. The first used a second-order smoothness prior which is invariant to changes in viewpoint. This eliminates the bias towards fronto-parallel planes of first-order algorithms. The second inpainting method uses an MRF-based prior that is learned from a database of natural images. The high-order nature of the prior makes the algorithm very flexible, however, it is also slow because of its complexity. The third approach assumes planar surfaces for missing areas and performs the inpainting by looking at the boundary around the missing region to find candidate planes. These three methods explore the compromise between prior representation power and complexity. Although the MRF-based prior obtained the best quantitative results some applications might require a faster approach.

The recent success of depth sensors for 3D reconstruction has brought new challenges to the field. One of them is the amount of data generated by these sensors. At high frame rates, a lot of the information is redundant because the sensor reconstructs the same surfaces over and over again. Chapter 5 reviewed the problem of point cloud simplification to address this issue. An algorithm was proposed that incrementally merges incoming depth maps into a global point cloud, removing redundancy but maintaining resolution. This algorithm is a promising way of reducing the amount of data that is needed for storing or transmitting a reconstructed point cloud without sacrificing quality.

The thesis also considered an application of 3D reconstruction in Chapter 6: free-viewpoint rendering. In particular, the sources of transparency in a scene and the artefacts it causes for free-viewpoint rendering were examined. From this, a novel algorithm was proposed to estimate transparency from a multi-view dataset. The algorithm successfully separated an image into a foreground and background layer with color and transparency information, allowing for a rendering with less-visible artefacts. The algorithm also showed that FVR is very susceptible to depth map and color image misalignments, which reinforced the importance of the joint depth and color image inpainting algorithms.

As is natural, the topics and contributions presented in this thesis suggest many opportunities for future research in all the touched areas. For example, the calibration algorithm presented is practical but still requires a printed checkerboard pattern. A calibration method that can use any arbitrary pattern (also known as self-calibration) would be even more practical. The SLAM system could benefit greatly from using inertial measurements to cope with low texture scenes. The inpainting methods presented are still far from being real-time and thus limit the possible applications. Moreover, as the recent developments in machine learning have shown, a more accurate and larger training database may improve the quality of the learned priors. Finally, the suggested method for transparency estimation considers reconstruction a separate problem. Since transparency clearly affects the quality of reconstruction, it is to be expected that a joint transparency and structure reconstruction method would achieve higher quality in both areas.

References

Achanta R, Shaji A, Smith K, Lucchi A, Fua P & Süsstrunk S (2012) SLIC superpixels compared to state-of-the-art superpixel methods. PAMI 34: 2274–2282.

Adelson EH & Bergen JR (1991) The plenoptic function and the elements of early vision, chapter 1. MIT Press.

Agarwal S, Furukawa Y, Snavely N, Curless B, Seitz S & Szeliski R (2015a) Building rome in a day. Communications of the ACM 54: 105–112.

Agarwal S, Mierle K & Others (2015b) Ceres solver. `http://ceres-solver.org`.

Alexa M, Behr J, Cohen-Or D, Fleishman S, Levin D & Silva T (2001) Point set surfaces. Proc. IEEE Visualization.

Amenta N, Bern M & Kamvysselis M (1998) A new Voronoi-based surface reconstruction algorithm. Proc. ACM SIGGRAPH.

Bai X & Sapiro G (2007) A geodesic framework for fast interactive image and video segmentation and matting. Proc. ICCV.

Bertalmio M, Sapiro G, Caselles V & Ballester C (2000) Image inpainting. Proc. ACM SIGGRAPH.

Blake A & Zisserman A (1987) Visual Reconstruction. MIT Press, Cambridge.

Boros E, Hammer PL & Sun X (1991) Network flows and minimization of quadratic pseudo-boolean functions. Technical Report RRR 17-1991, RUTCOR.

Brodsky D & Watson B (2000) Model simplification through refinement. Proc. Graphics Interface.

Buehler C, Bosse M, McMillan L, Gortler S & Cohen M (2001) Unstructured lumigraph rendering. Proc. SIGGRAPH.

Chen J, Bautembach D & Izadi S (2013) Scalable real-time volumetric surface reconstruction. ACM Trans. Graph. 32.

Chen Q & Koltun V (2014) Fast MRF optimization with application to depth reconstruction. Proc. CVPR.

Chen S & Williams L (1993) View interpolation for image synthesis. Proc. ACM SIGGRAPH.

Chuang Y, Curless B, Salesin D & Szeliski R (2001) A bayesian approach to digital matting. Proc. CVPR.

Chéné Y, Belin E, Chapeau-Blondeau F, Caffier V, Boureau T & Rousseau D (2014) Plant Image Analysis: Fundamentals and Applications, chapter Anatomo-functional bimodality imaging for plant phenotyping: An insight through depth imaging coupled to thermal imaging. CRC Press.

Criminisi A, Perez P & Toyama K (2003) Object removal by exemplar-based inpainting. Proc. CVPR.

Dabov K, Foi A, Katkovnik V & Egiazarian K (2006) Image denoising with block-matching and 3D filtering. Proc. SPIE Electronic Imaging.

Davis A, Levoy M & Durand F (2012) Unstructured light fields. Proc. Eurographics.

Davison A, Reid I, Molton N & Stasse O (2007) Monoslam: Real-time single camera slam. TPAMI .

de Aguiar E, Stoll C, Theobalt C, Ahmed N, Seidel HP & Thrun S (2008) Performance capture from sparse multiview video. Proc. SIGGRAPH.

Debevec PE, Taylor CJ & Malik J (1996) Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. Proc. ACM SIGGRAPH.

Dominio F, Donadeo M & Zanuttigh P (2014) Combining multiple depth-based descriptors for hand gesture recognition. Pattern Recognition Letters 50: 101–111.

Dryanovski I, Valenti R & Xiao J (2013) Fast visual odometry and mapping from RGB-D data. Proc. ICRA.

Eade E & Drummond T (2007) Monocular SLAM as a graph of coalesced observations. Proc. ICCV.

Eisemann M, De Decker B, Magnor M, Bekaert P, de Aguiar E, Ahmed N, Theobalt C & Sellent A (2008) Floating textures. Comput. Graph. Forum 27: 409—418.

Engel J, Schöps T & Cremers D (2014) LSD-SLAM: Large-scale direct monocular SLAM. Proc. ECCV.

Faugeras O (1993) Three-Dimensional Computer Vision. The MIT Press.

Faugeras O, Luong Q & Papadopoulo T (2001) The Geometry of Multiple Images. The MIT Press.

Faugeras OD & Keriven R (1998) Complete dense stereovision using level set methods. Proc. ECCV.

Feldman D, Pajdla T & Weinshall D (2003) On the epipolar geometry of the crossed-slits projection. Proc. IEEE International Conference on Computer Vision (ICCV), 988—995.

Fitzgibbon A, Wexler Y & Zisserman A (2003) Image-based rendering using image-based priors. Proc. ICCV.

Fitzgibbon A & Zisserman A (1998) Automatic camera recovery for closed or open image sequences. Proc. ECCV, 311—326.

Forster C, Pizzoli M & Scaramuzza D (2014) Fast semi-direct monocular visual odometry. Proc. ICRA.

Freedman B, Shpunt A, Machline M & Arieli Y (2008) Depth mapping using projected patterns. US Patent App. 11/899,542.

Fuchs S & Hirzinger G (2008) Extrinsic and depth calibration of ToF-cameras. Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Fuhrmann S & Goesele M (2014) Floating scale surface reconstruction. Proc. ACM SIGGRAPH.

Funes Mora K, Monay F & JM O (2014) EYEDIAP: a database for the development and evaluation of gaze estimation algorithms from RGB and RGB-D cameras. Proc. Symposium on Eye Tracking Research and Applications, 255–258.

Furukawa Y, Curless B, Seitz S & Szeliski R (2009) Manhattan-world stereo. Proc. CVPR.

Furukawa Y & Ponce J (2009) Accurate camera calibration from multi-view stereo and bundle adjustment. Proc. IJCV.

Furukawa Y & Ponce J (2010) Accurate, dense, and robust multiview stereopsis. TPAMI 32: 1362–1376.

Gallup D, Frahm JM, Mordohai P, Yang Q & Pollefeys M (2007) Real-time plane-sweeping stereo with multiple sweeping directions. Proc. CVPR.

Gallup D, Frahm JM & Pollefeys M (2010) Piecewise planar and non-planar stereo for urban scene reconstruction. Proc. CVPR.

Gastal E & Oliveira M (2010) Shared sampling for real-time alpha matting. Computer Graphics Forum 29.

Glasner D, Bagon S & Irani M (2009) Super-resolution from a single image. Proc. ICCV.

Gockel T, Azad P & Dillmann R (2004) Calibration issues for projector-based 3D-scanning. Proc. Shape Modeling Applications, 367–370.

Gortler S, Grzeszczuk R, Szeliski R & Cohen M (1996) The lumigraph. Proc. SIGGRAPH.

Grossberg M & Nayar S (2001) A general imaging model and a method for finding its parameters. Proc. IEEE International Conference on Computer Vision (ICCV), 108–115.

Gupta R & Hartley R (1997) Linear pushbroom cameras. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 19: 963—-975.

Hammer PL, Hansen P & Simeone B (1984) Roof duality, complementation and persistency in quadratic 0-1 optimization. Mathematical Programming 28: 121—155.

Hansard M, Lee S, Choi O & Horaud R (2012) Time-of-Flight Cameras: Principles, Methods and Applications. Springer.

Hartley R & Zisserman A (2000) Multiple View Geometry in Computer Vision. Cambridge University Press.

Hasinoff S, Kang S & Szeliski R (2006) Boundary matting for view synthesis. Computer Vision and Image Understanding 103.

He K, Sun J & Tang X (2013) Guided image filtering. TPAMI 35: 1397—-1409.

Heckbert P & Garland M (1997) Survey of polygonal surface simplification algorithms. Proc. Multiresolution Surface Modeling Course, SIGGRAPH.

Heikkilä J (2000) Geometric camera calibration using circular control points. PAMI 22: 1066–1077.

Heikkilä M, Pietikäinen M & Schmid C (2009) Description of interest regions with local binary patterns. Pattern Recognition 42: 425—-436.

Henry P, Krainin M, Herbst E, Ren X & Fox D (2010) RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. Proc. ISER.

Hinton GE (2002) Training products of experts by minimizing contrastive divergence. Neural Comput. 14: 1771—1800.

Hongdong L & Hartley R (2006) Five-point motion estimation made easy. Proc. ICPR.

Hoppe H (1996) Progressive meshes. Proc. ACM SIGGRAPH.

Horn B (1986) Robot Vision. The MIT Press.

Hyvärinen A, Hurri J & Hoyer P (2009) Natural Image Statistics: A probabilistic approach to early computational vision. Springer-Verlag New York Inc.

Ikehata S, Ji-Ho C & Aizawa K (2013) Depth map inpainting and super-resolution based on internal statistics of geometry and appearance. Proc. ICIP.

Ishikawa H & Geiger D (2006) Rethinking the prior model for stereo. Proc. ECCV.

Joshi N, Matusik W & Avidan S (2006) Natural video matting using camera arrays. ACM Trans. Graph. 25.

Juan O & Keriven R (2005) Trimap segmentation for fast and user-friendly alpha matting. Proc. VLSM.

Kahlmann T & Ingensand H (2006) Calibration of the fast range imaging camera swissranger for use in the surveillance of the environment. Electro-Optical Remote Sensing II 6396.

Kanatani K (1994) Analysis of 3-D rotation fitting. PAMI 16: 543–549.

Kannala J & Brandt S (2006) A generic camera model and calibration method for conventional, wide-angle and fish-eye lenses. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 28: 1335–1340.

Kannala J, Heikkilä J & Brandt S (2008) Geometric camera calibration. Wiley Encyclopedia of Computer Science and Engineering .

Kim C, Sorkine-Hornung A, Heinzle S, Matusik W & Gross M (2011) Multi-perspective stereoscopy from light fields. Proc. SIGGRAPH.

Kim C, Zimmer H, Sorkine-Hornung A, Pritch Y & Gross M (2013) Scene reconstruction from high spatio-angular resolution light fields. Proc. SIGGRAPH.

Klein G & Murray DW (2007) Parallel tracking and mapping for small ar workspaces. Proc. ISMAR.

Klose F, Ruhl K, Lipski C, & Magnor M (2011) Flowlab–an interactive tool for editing dense image correspondences. Proc. CVMP.

Klowsky R, Kuijper A & Goesele M (2012) Modulation transfer function of patch-based stereo systems. Proc. CVPR.

Kohli P (2007) Minimizing dynamic and higher order energy functions using graph cuts. Ph.D. thesis, Oxford Brookes University.

Kushal A, Self B, Furukawa Y, Gallup D, Hernandez C, Curless B & Seitz S (2012) Photo tours. Proc. 3DimPVT.

Lange R & Seitz P (2001) Solid-state time-of-flight range camera. IEEE Journal of Quantum Electronics 37: 390—397.

Latta SG (2010) Gesture keyboarding. US 2010/0199228 A1.

Levin A, Lischinski D & Weiss Y (2004) Colorization using optimization. ACM Transactions on Graphics 23: 689–694.

Levin A, Lischinski D & Weiss Y (2007) A closed-form solution to natural image matting. PAMI .

Levin D (2001) Mesh-independent surface interpolation. Proc. Advances in Computational Mathematics.

Levoy M & Hanrahan P (1996) Light field rendering. Proc. SIGGRAPH.

Lindner M & Kolb A (2007) Calibration of the intensity-related distance error of the pmd tof-camera. Intelligent Robots and Computer Vision XXV: Algorithms, Techniques, and Active Vision 6764.

Lindner M, Schiller I, Kolb A & Koch R (2010) Time-of-flight sensor calibration for accurate range sensing. Journal of Computer Vision and Image Understanding 114: 1318–1328.

Linsen L (2001) Point cloud representation. Technical Report 2001-3, Faculty of Computer Science, University of Karlsruhe.

Lipski C, Klose F & Magnor M (2014) Correspondence and depth-image based rendering a hybrid approach for free-viewpoint video. Circuits and Systems for Video Technology 24.

Lipski C, Linz C, Berger K, Sellent A & Magnor M (2010) Virtual video camera: Image-based viewpoint navigation through space and time. Comput. Graph. Forum 29: 2555—2568.

Lipsky C, Klose F, Ruhl K & Magnor M (2011) Making of who cares? HD stereoscopic free viewpoint video. Proc. CMVP.

Lowe D (2004) Distinctive image features from scale-invariant keypoints. IJCV 60.

Lu C & Tang X (2015) Surpassing human-level face verification performance on LFW with GaussianFace. Proc. AAAI.

Marr D (1982) Vision: A Computational Investigation into the Human Representation and Processing of Visual Information. W. H. Freeman and Company.

McAuley J, Caetano T, Smola A & Franz M (2006) Learning high-order MRF priors of color images. Proc. International Conference on Machine Learning, 617–624.

Merrell P, Akbarzadeh A, Wang L, Mordohai P, Frahm J, Yang R, Nistér D & Pollefeys M (2007) Real-time visibility-based fusion of depth maps. Proc. ICCV.

Moenning C & Dodgson N (2004) Intrinsic point cloud simplification. Proc. 14th GraphiCon.

Morales R, Badesa F, Garcia-Aracil N, Sabater J & Zollo L (2014) Soft robotic manipulation of onions and artichokes in the food industry. Advances in Mechanical Engineering 2014.

Mémoli F & Sapiro G (2003) Distance functions and geodesics on point clouds. Technical Report TR 1902, IMA, University of Minnesota, USA.

Müller K, Smolic A, Dix K, Merkle P, Kauff P & Wiegand T (2008) View synthesis for advanced 3D video systems. EURASIP Journal on Image and Video Processing 2008.

Newcombe R, Izadi S, Hilliges O, Molyneaux D, Kim D, Davison A, Kohli P, Shotton J, Hodges S & Fitzgibbon A (2011a) Kinectfusion: real-time dense surface mapping and tracking. Proc. ISMAR.

Newcombe R, Lovegrove S & Davison A (2011b) Dtam: Dense tracking and mapping in real-time. Proc. ICCV.

Okabe A, Boots B & Sugihara K (2000) Spatial Tessellations - Concepts and Applications of Voronoi Diagrams. John Wiley & Sons, Chicester, UK., 2nd edition.

P K, Torr P & Zisserman A (2006) An object category specific MRF for segmentation. In: Toward Category-Level Object Recognition, volume 4170 of *Lecture Notes in Computer Science*, 596–616. Springer Berlin Heidelberg.

Pauly M, Gross M & Kobbelt L (2002) Efficient simplification of point-sampled surfaces. Proc. IEEE conference on visualization, 163—-70.

Ponce J & Hebert M (2014) Trinocular geometry revisited. Proc. CVPR.

Pulli K, Cohen M, Duchamp T, Hoppe H, Shapiro L & Stuetzle W (1997) View-vased rendering: visualizing real objects from scanned range and color data. Proc. Eurographics workshop on rendering.

Quan L & Lan Z (1999) Linear n-point camera pose determination. PAMI 21: 774–780.

Ramalingam S, Sturm P & Lodha S (2005) Towards complete generic camera calibration. Proc. CVPR.

Reitmayr G, Langlotz T, Wagner D, Mulloni A, Schall G, Schmalstieg D & Pan Q (2010) Simultaneous localization and mapping for augmented reality. Proc. ISUVR.

Rhemann C, Rother C, Rav-Acha A & Sharp T (2008) High resolution matting via interactive trimap segmentation. Proc. CVPR.

Rhemann C, Rother C, Wang J, Gelautz M, Kohli P & Rott P (2009) A perceptually motivated online benchmark for image matting. Proc. CVPR.

Rossignac J & Borrel P (1993) Multiresolution 3D approximations for rendering complex scenes. Proc. Modeling in Computer Graphics: Methods and Applications.

Roth S & Black M (2009) Fields of experts. IJCV 82: 205–229.

Salvi J, Pagès J & Batlle J (2004) Pattern codification strategies in structured light systems. Pattern Recognition 37: 827–849.

Schmidt U, Gao Q & Roth S (2010) A generative perspective on MRFs in low-level vision. Proc. CVPR.

Seitz S, Curless B, Diebel J, Scharstein D & Szeliski R (2006) A comparison and evaluation of multi-view stereo reconstruction algorithms. Proc. CVPR, 519–526.

Shaffer E & Garland M (2001) Efficient adaptive simplification of massive meshes. Proc. IEEE Visualization.

Shum HY, Chan SC & Kang SB (2007) Image-Based Rendering. Springer.

Silberman N, Hoiem D, Kohli P & Fergus R (2012) Indoor segmentation and support inference from RGBD images. Proc. ECCV.

Sinha S, Steedly D & Szeliski R (2009) Piecewise planar stereo for image-based rendering. Proc. ICCV.

Smisek J, Jancosek M & Pajdla T (2013) 3d with kinect. Proc. Consumer Depth Cameras for Computer Vision, 3–25.

Smolic A (2011) 3D video and free viewpoint video – from capture to display. Pattern recognition 44(9): 1958–1968.

Snavely N, Seitz S & Szeliski R (2007) Modeling the world from internet photo collections. IJCV .

Song H & Feng H (2008) A global clustering approach to point cloud simplification with a specified data reduction ratio. Computer-Aided Design 40.

Starck J, Maki A, Nobuhara S, Hilton A & Matsuyama T (2009) The multiple-camera 3-D production studio. Circuits Systems Video Technology 19: 856–869.

Strasdat H, Montiel J & Davison A (2010) Visual SLAM: Why filter? Proc. ICRA.

Strecha C, Fransens R, & Van Gool L (2004) Wide-baseline stereo from multiple views: a probabilistic account. Proc. CVPR.

Swaminathan R, Grossberg M & Nayar S (2003) A perspective on distortions. Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 594—-601.

Szeliski R (2010) Computer vision: algorithms and applications. Springer London.

Tomasi C & Manduchi R (1998) Bilateral filtering for gray and color images. Proc. ICCV, 839–846.

Triggs B, McLauchlan P, Hartley R & Fitzgibbon A (1999) Bundle adjustment — a modern synthesis. Proc. International Workshop on VIsion Algorithms.

Turk G (1992) Re-tiling polygonal surfaces. Proc. SIGGRPAH.

Tykkälä T, Comport A, Kämäräinen J & Hartikainen H (2014) Live rgb-d camera tracking for television production studios. Journal of Visual Communication and Image Representation 25: 207–217.

Umeyama S (1991) Least-squares estimation of transformation parameters between two point patterns. TPAMI 13: 376–380.

Vu HH, Labatut P, Keriven R & Pons JP (2012) High accuracy and visibility-consistent dense multi-view stereo. TPAMI .

Wang J & Cohen M (2007) Optimized color sampling for robust matting. Proc. CVPR.

Wexler Y, Fitzgibbon A & Zisserman A (2002a) Bayesian estimation of layers from multiple images. Proc. ECCV.

Wexler Y, Fitzgibbon A & Zisserman A (2002b) Image-based environment matting. Proc. Eurographics workshop on Rendering.

Whelan T, Kaess M, Fallon M, Johannsson H, Leonard J & McDonald J (2012) Kintinuous: Spatially extended KinectFusion. Proc. RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras.

Wood Z, Hugues H, Desbrun M & Schröder P (2002) Isosurface topology simplification. Technical Report MSR-TR-2002-28, Microsoft Research.

Woodford OJ, Torr PHS, Reid ID & Fitzgibbon AW (2008) Global stereo reconstruction under second order smoothness priors. Proc. CVPR.

Xiao J & Furukawa Y (2014) Reconstructing the world's museums. IJCV .

Yanover C, Meltzer T & Weiss Y (2006) Linear programming relaxations and belief propagation – an empirical study. Mach. Learn. Res. 7: 1887—-1907.

Zhang K, Jin H, Fu Z & Liu N (2007) Optimal learning high-order markov random fields priors of colour image. Proc. ACCV, 482–491.

Zhang L & Seitz S (2007) Estimating optimal parameters for MRF stereo from a single image pair. PAMI 29(2).

Zhang Z (1999) Flexible camera calibration by viewing a plane from unknown orientations. Proc. IEEE International Computer Vision Conference (ICCV), 666–673.

Zhange F (ed) (2005) The Schur complement and its applications, volume 4 of *Numerical Methods and Algorithms*. Springer.

Zheng K, Colburn A, Agarwala A, Agrawala M, Curless B, Salesin D & Cohen M (2009) Parallax photography: Creating 3D cinematic effects from stills. Proc. Graphic Interface.

Zheng KC, Kang SB, Cohen M & Szeliski R (2007) Layered depth panoramas. Proc. CVPR.

Zitnick C, Kang S, Uyttendaele M, Winder S & Szeliski R (2004) High-quality video view interpolation using a layered representation. Proc. SIGGRAPH.

Zontak M (2011) Internal statistics of a single natural image. Proc. CVPR.

Original publications

I Herrera C. D, Kannala J & Heikkilä J (2012) Joint depth and color camera calibration with distortion correction. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(10): 2058–2064.

II Herrera C. D, Kim K, Kannala J, Pulli K & Heikkilä J (2014) DT-SLAM: Deferred Triangulation for Robust SLAM. International Conference on 3D Vision (3DV).

III Herrera C. D, Kannala J, Ladický L & Heikkilä J (2013) Depth map inpainting under a second-order smoothness prior. Proc Scandinavian Conference on Image Analysis (SCIA). Lecture Notes on Computer Science 7944: 555–566.

IV Herrera C. D, Kannala J, Sturm P & Heikkilä J (2013) A Learned Joint Depth and Intensity Prior using Markov Random Fields. International Conference on 3D Vision (3DV) 1: 17–24.

V Herrera C. D, Kannala J & Heikkilä J (2011) Generating Dense Depth Maps Using a Patch Cloud and Local Planar Surface Models. 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON): 1-4.

VI Kyöstilä T, Herrera C. D, Kannala J & Heikkilä J (2013) Merging overlapping depth maps into a nonredundant point cloud. Proc Scandinavian Conference on Image Analysis (SCIA). Lecture Notes on Computer Science 7944: 567–578.

VII Herrera C. D, Kannala J & Heikkilä J (2011) Multi-View Alpha Matte for Free Viewpoint Rendering. International Conference on Computer Vision / Computer Graphics Collaboration Techniques and Applications (Mirage). Lecture Notes on Computer Science 6930: 98–109.

Reprinted with permission from IEEE (I, II, IV, V) and Springer-Verlag (III, VI, VII).

Original publications are not included in the electronic version of the dissertation.

519. Partala, Juha (2015) Algebraic methods for cryptographic key exhange

520. Karvonen, Heikki (2015) Energy efficiency improvements for wireless sensor networks by using cross-layer analysis

521. Putaala, Jussi (2015) Reliability and prognostic monitoring methods of electronics interconnections in advanced SMD applications

522. Pirilä, Minna (2015) Adsorption and photocatalysis in water treatment : active, abundant and inexpensive materials and methods

523. Alves, Hirley (2015) On the performance analysis of full-duplex networks

524. Siirtola, Pekka (2015) Recognizing human activities based on wearable inertial measurements : methods and applications

525. Lu, Pen-Shun (2015) Decoding and lossy forwarding based multiple access relaying

526. Suopajärvi, Terhi (2015) Functionalized nanocelluloses in wastewater treatment applications

527. Pekuri, Aki (2015) The role of business models in construction business management

528. Mantere, Matti (2015) Network security monitoring and anomaly detection in industrial control system networks

529. Piri, Esa (2015) Improving heterogeneous wireless networking with cross-layer information services

530. Leppänen, Kimmo (2015) Sample preparation method and synchronized thermography to characterize uniformity of conductive thin films

531. Pouke, Matti (2015) Augmented virtuality : transforming real human activity into virtual environments

532. Leinonen, Mikko (2015) Finite element method and equivalent circuit based design of piezoelectric actuators and energy harvester dynamics

533. Leppäjärvi, Tiina (2015) Pervaporation of alcohol/water mixtures using ultra-thin zeolite membranes : membrane performance and modeling

534. Lin, Jhih-Fong (2015) Multi-dimensional carbonaceous composites for electrode applications

535. Goncalves, Jorge (2015) Situated crowdsourcing : feasibility, performance and behaviours

Book orders:
Granum: Virtual book store
http://granum.uta.fi/granum/

UNIVERSITY OF OULU P.O. Box 8000 FI-90014 UNIVERSITY OF OULU FINLAND

ACTA UNIVERSITATIS OULUENSIS

UNIVERSITY of OULU
OULUN YLIOPISTO

ISBN 978-952-62-0852-7 (Paperback)
ISBN 978-952-62-0853-4 (PDF)
ISSN 0355-3213 (Print)
ISSN 1796-2226 (Online)