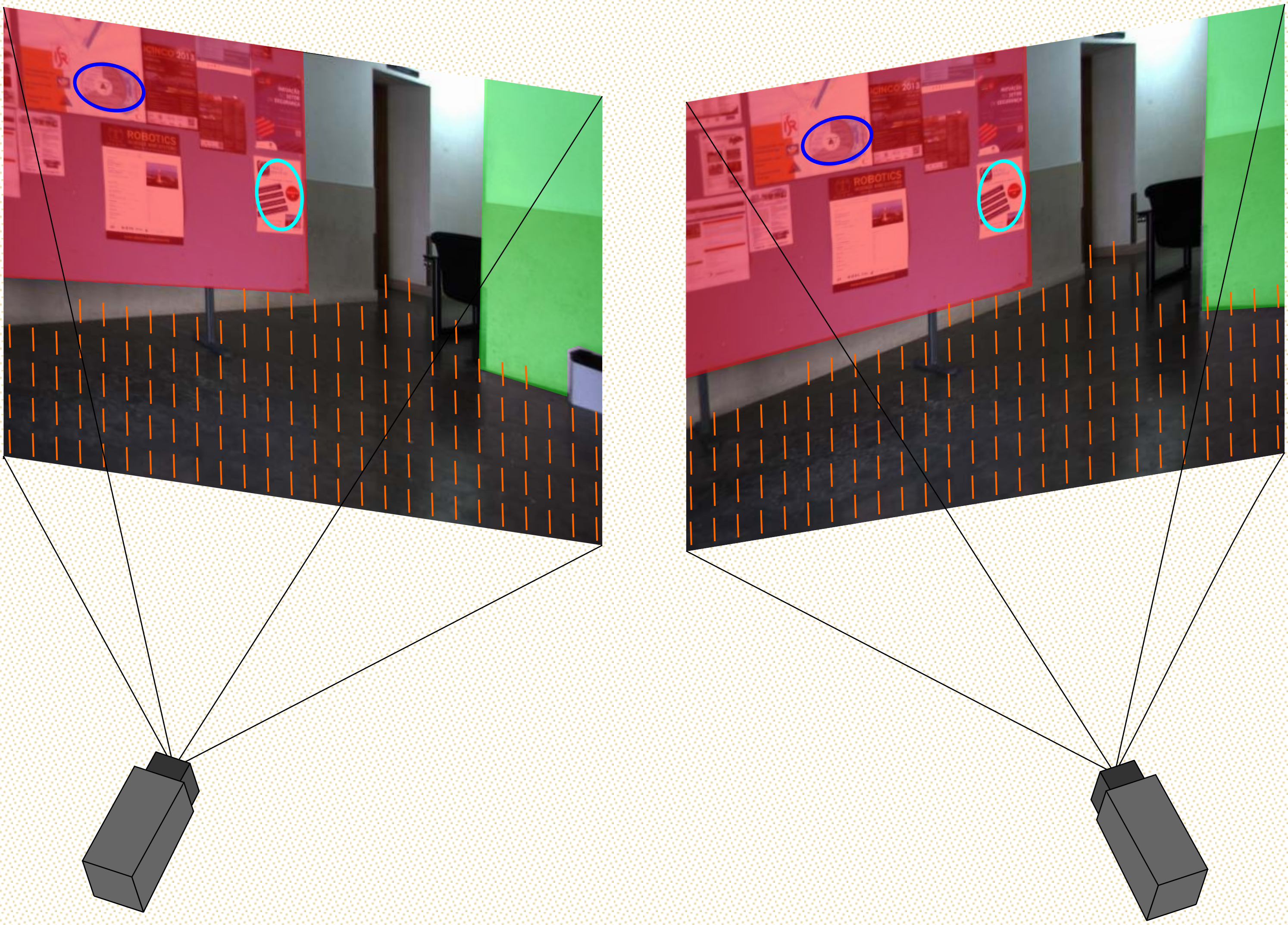


Carolina dos Santos Raposo

CALIBRATION, STRUCTURE-FROM-MOTION AND REGISTRATION BEYOND POINT FEATURES

Tese de doutoramento em Engenharia Electrotécnica e de Computadores, ramo de especialização em Automação e Robótica, orientada pelo Professor Doutor João Pedro de Almeida Barreto e apresentada ao Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Ciências e Tecnologia da Universidade de Coimbra

Setembro de 2016

• U C •

UNIVERSIDADE DE COIMBRA

FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

PhD Thesis

Calibration, Structure-from-Motion and Registration beyond Point Features

Carolina dos Santos Raposo
http://arthronav.isr.uc.pt/~carolina
carolinaraposo@isr.uc.pt

September 19, 2016

Calibration, Structure-from-Motion and Registration Beyond Point Features

By

Carolina Raposo

Advisor: Prof. Dr. João Pedro Barreto

Department of Electrical and Computer Engineering
Faculty of Sciences and Technology,
University of Coimbra

September 19, 2016

Acknowledgments

My first acknowledgement goes to my advisor João Barreto for teaching me a lot about geometry applied to computer vision and letting me choose my favourite research topics. Without his guidance and patience, I know this thesis would not have been possible.

Secondly, I want to thank my lab colleagues, and co-authors of some papers, Michel, Miguel, Pedro and Francisco for helping me whenever I needed and making my daily work a more enjoyable experience.

On a more personal level, I would like to sincerely thank my parents, sisters and brother, as well as my closest and most important friends: Miguel, Sampaio, Vanessa, Rita, Carla, Teresa, Mafalda, Nujo, Vitor... They truly helped my without even noticing with their good moods and valuable advices.

I acknowledge the Portuguese Science Foundation (FCT) for generously funding my PhD program through grant SFRH/BD/88446/2012, and the Institute for Systems and Robotics - Coimbra for supporting my research. This work has also been supported by the FCT and COMPETE program (co-funded by FEDER) under Projects PDCS2010 (PTDC/EEA-AUT/113818/2009), AMS-HMI12 (RECI/EEI-AUT/0181/2012) and VisArthro (PTDC/EEI-AUT/3024/2014).

Abstract

Geometric computer vision is strongly based in point primitives in problems of calibration, Structure-from-Motion (SfM), and registration. The reason for this is that points are the most fundamental primitive that is always present in images. There are other sorts of primitives that can be used, either ones that often arise in man-made environments and that are composition of points, such as lines and planes, or primitives that have a differential character such as affine matches or normals, encoding how surfaces vary locally. In this thesis, we explore these alternative primitives, showing that in the specific contexts of calibration, SfM and registration they can be advantageous with respect to the dominating trend.

The first line of work concerns the calibration of heterogeneous sensor arrangements. We start by developing a method for calibrating a camera-depth sensor pair, based on a novel 3D plane registration algorithm, that is able to provide results as accurate as the state-of-the-art using about 1/6 of the input images. This important improvement enables the extension of the calibration approach to the case of non-overlapping Field-of-View (FoV) through mirror reflections, whereas methods that require many more images easily lead to a prohibitive total number of frames to be acquired. By applying a similar extension to the state-of-the-art approach for calibrating camera-Laser RangeFinder (LRF) pairs, we achieve, for the first time, an algorithm that is able to calibrate any sensor arrangement - with or without overlapping FoV - comprising cameras, LRFs and depth sensors, as long as a camera is involved in the system.

Plane primitives are also used in the task of SfM and 3D modelling. In these contexts, they lead to advantages over point primitives that include being able to deal with situations of lack of texture, perceptual aliasing, high surface slant, wide-baseline and presence of dynamism in the scene, while providing visually pleasant reconstructions. Thus, and knowing that man-made environments are dominated by planar surfaces, we propose two pipelines that accomplish SfM using these primitives. Experiments clearly demonstrate that all these problems are efficiently tackled with planes, with the proposed pipelines significantly outperforming state-of-the-art point-based approaches in challenging situations.

The relations between an Affine Correspondence (AC) and the fundamental geometry have recently been derived [12]. This new result motivated our third topic of research, where we study how ACs constrain the homography and the epipolar geometry. We show that ACs, that are currently discarded after performing point association, contain extremely useful information that can reduce the combinatorics of SfM and enable fast and reliable segmentation of planes. This led to the development of a new monocular Visual Simultaneous Localization and Mapping (vSLAM) pipeline that provides a dense Piecewise Planar Reconstruction (PPR) of the scene and significantly outperforms another competing monocular SfM method.

The final subject of research of this thesis is 3D point cloud registration. This is a topic with important applications in object detection and recognition, tracking, Simultaneous Localization and Mapping (SLAM) and even medical endoscopy. We propose to solve the coarse alignment of point clouds in arbitrary initial positions by extracting pairs of oriented points, i.e. points with associated normals. Our method greatly benefits from a new smart indexing technique for extracting pairs of points proposed in the Super4PCS algorithm [79] that works solely with points. A comparison with this method shows that including normals leads to similar or higher accuracies in less than 1/5 of the time when working with noisy depth-camera scans. Speed ups of over 100× are achieved for noise-free datasets.

Resumo

A visão por computador geométrica é fortemente baseada em primitivas de pontos em problemas de calibração, estrutura por movimento e registo. A razão para isto é que os pontos são a primitiva mais fundamental que está sempre presente nas imagens. Existem outros tipos de primitivas que podem ser usadas, tanto as que surgem em ambientes criados pelo Homem e que são a composição de pontos, tal como linhas e planos, como as que têm um carácter diferencial, tal como correspondências afim e normais que codificam a maneira como as superfícies variam localmente. Nesta tese, exploramos estas primitivas alternativas, mostrando que nos contextos específicos de calibração, estrutura por movimento e registo podem ser vantajosas em relação à tendência dominante.

A primeira linha de trabalho diz respeito a calibração de conjuntos de sensores heterogéneos. Começamos por desenvolver um método para calibrar um par câmara-sensor de profundidade, baseado num algoritmo novo de registo de planos 3D, que consegue produzir resultados tão precisos como o estado-da-arte usando acerca de 1/6 das imagens de entrada. Esta melhoria importante permite a extensão do método de calibração para o caso em que os campos de visão dos sensores não se sobrepõem, através de reflexões de espelhos, enquanto que os métodos que necessitam de muitas imagens de calibração facilmente originam um número total de imagens a serem adquiridas que é proibitivo. Aplicando uma extensão semelhante ao método do estado-da-arte em calibração de pares câmara-telémetro laser, obtemos, pela primeira vez, um algoritmo que é capaz de calibrar qualquer conjunto de sensores - com ou sem sobreposição dos campos de visão - contendo câmaras, telémetros laser e sensores de profundidade, desde que uma câmara esteja envolvida no sistema.

Primitivas de planos são também usadas nas tarefas de estrutura por movimento e modelação 3D. Nestes contextos, estas primitivas têm vantagens em relação a primitivas de pontos que incluem serem capazes de lidar com situações de falta de textura, *aliasing* perceptual, declive das superfícies elevado, distância entre câmaras grande e presença de dinamismo na cena, enquanto produzem reconstruções visualmente agradáveis. Assim, e sabendo que os ambientes criados pelo Homem são dominados por superfícies planares, propomos dois esquemas de estrutura por movimento a partir destas primitivas. Os resultados experimentais mostram com clareza que todos estes problemas são eficazmente corrigidos usando planos, e que os esquemas propostos funcionam melhor que os algoritmos do estado-da-arte baseados em pontos em situações desafiantes.

As relações entre correspondências afim e a geometria fundamental foram recentemente derivadas em [12]. Este novo resultado motivou o nosso terceiro tópico de investigação, onde estudamos como as correspondências afim restringem a homografia e a geometria epipolar. Mostramos que as correspondências afim, que actualmente são descartadas depois da associação de pontos, contêm informação extremamente útil que pode reduzir a combinatória em estrutura por movimento e permitem uma rápida e fidedigna segmentação de planos. Isto levou ao desenvolvimento de um novo esquema monocular de SLAM (Localização e Mapeamento Simultâneos) visual que origina reconstruções densas da cena em planos e funciona significativamente melhor do que um método competitivo de estrutura por movimento monocular.

O último assunto de investigação desta tese é o registo de nuvens de pontos 3D. Este é um tópico com aplicações importantes em detecção e reconhecimento de objectos, seguimento, SLAM e até endoscopia médica. Propomo-nos a resolver o alinhamento grosseiro de nuvens de pontos em posições iniciais arbitrárias através da extracção de pares de pontos orientados, *i.e.* pontos com normais associadas. O nosso método beneficia bastante de uma nova técnica de indexação inteligente para extrair pares de pontos proposta no algoritmo Super4PCS [79] que trabalha somente com pontos. Uma comparação com este método mostra que incluir normais origina precisões semelhantes ou superiores em menos de 1/5 do tempo, quando trabalhamos com dados ruidosos adquiridos por câmaras de profundidade. Acelerações de mais de 100× são obtidas em datasets sem ruído.

Contents

List of Figures

List of Tables

Acronyms

SfM	Structure-from-Motion	i
FoV	Field-of-View	i
LRF	Laser RangeFinder	i
AC	Affine Correspondence	i
PC	Point Correspondence	5
SLAM	Simultaneous Localization and Mapping	ii
vSLAM	Visual Simultaneous Localization and Mapping	ii
4PCS	4-Point Congruent Sets	112
PPR	Piecewise Planar Reconstruction	ii
RANSAC	RANdom SAmple Consensus	2
DoF	Degree-of-Freedom	43
ICP	Iterative Closest Point	46
SURF	Speeded Up Robust Features	45
PEaRL	Propose Expand and Re-estimate Labels	54
UFL	Uncapacitated Facility Location	56
MRF	Markov Random Field	5
ZNCC	Zero-mean Normalized Cross-Correlation	57
PPSS	Piecewise Planar StereoScan	69
SIFT	Scale-Invariant Feature Transform	87
AP	Affinity Propagation	98
LCP	Largest Common Pointset	114
2PNS	2-Point-Normal Sets	112

RMS	Root Mean Square

Chapter 1

Introduction and Motivation

Images are 2D representations of a 3D reality. There is loss of information and recovering a 3D description of the observed scene is of great interest. This often requires intrinsic camera calibration in order to infer meaningful metric information from 2D images. When working with multiple sensors, extrinsic calibration is also necessary so that the sensor setup works as a whole. These are all well studied problems in computer vision as can be seen by the numerous publications about topics like calibration, SfM, 3D reconstruction, and registration. Although most of these works are strongly based in point primitives, there are other sorts of primitives that can be used, either ones that are a composition of points that often arise in man-made environments, such as lines and planes, or primitives that have a differential character such as affine matches or normals that encode how surfaces vary locally.

Many sensor calibration approaches work by extracting keypoints from images for finding both the intrinsic and extrinsic parameters. This is commonly done when calibrating cameras by using a planar checkerboard pattern [122, 133, 146]. Other sensor arrangements have also been calibrated by extracting keypoints, such as RGB-D cameras [20] and camera-LRF pairs [87]. However, attempts to perform the intrinsic and extrinsic calibration of heterogeneous sensor setups using other primitives have been done. In [52,53], a Kinect sensor is calibrated by registering 3D planes and the LRF-camera pair calibration in [135] is accomplished by registering 3D lines with 3D planes. These are relatively recent research directions and there still is an interest in pursuing them since the existing methods either require many images to perform well [52, 53] or do not work in general configurations (*e.g.* when the sensors' FoVs do not overlap) [53, 135].

SfM is another relevant research subject that consists in recovering the motion of a camera from the image information it acquires and simultaneously obtain some sort of 3D modelling of the observed scene. Typically, it is assumed that the observed scene is rigid and SfM is accomplished from a sequence of images acquired by a moving monocular camera. In the last few years the paradigm has been extended to either accommodate different sensor modalities, e.g. passive stereo [44] or, more recently, RGB-D [29], or to relax the rigidity assumption by considering multiple rigid motions [108,126,134], areas of non-rigid deformation [72], or even no rigidity at all [19,41,91]. Despite the diversity in acquisition setups and rigidity/non-rigidity assumptions, one factor remains the same: all the proposed approaches are point-based. The information that is extracted from images are keypoints that can potentially be associated, with the local patches surrounding them being used simply as a signature. The algorithms work globally over these points.

Methods and approaches for carrying SfM and 3D modelling in general that use image primitives other than keypoints is something that has been scarcely explored in the past, with most existing works being limited to the use of lines [10, 21, 111]. Bartoli and Sturm [10] propose a multi-view algorithm for recovering the structure and motion from line correspondences. Camera motion estimation is performed by computing trifocal tensors for triplets of consecutive images. Later, Schindler *et al.* [111] extend the work of Bartoli and Sturm [10] by introducing additional constraints on the orientations of lines in a 3D urban environment, leading to simpler representations of lines. More recently, Chandraker *et al.* [21] used line matches in two stereo pairs (4 images), to compute the camera pose in indoor office environments. Unlike the previous methods that required a minimum of 3 monocular views, this method requires only 2 stereo pairs for estimating the camera motion. Moreover, the authors use infinite lines, overcoming the problem of the quality of reconstruction being limited by the accuracy of detecting end-points.

Another important topic of research in computer vision is 3D point cloud registration, having several applications in object modelling, detection and recognition, tracking, and SLAM. Similarly to the case of SfM, 3D registration is typically solved using solely 3D points, either by performing a coarse alignment with RANdom SAmple Consensus (RANSAC)-based schemes [1, 79, 85, 128], or by performing fine alignments by minimizing error metrics such as the Euclidean distance between corresponding points [14, 96]. Few works solve the registration problem using other primitives. Examples include works that find correspondences between pairs of points augmented with normals and possibly colour information [22,27,138] for computing the rigid transformation to align point clouds in arbitrary initial positions. However, the majority of these algorithms are applied in object detection and recognition, and not only include time-consuming offline procedures but also require that the source and target point clouds have sufficient overlap.

The reason why most of the research on calibration, SfM and 3D registration so far focuses in using points is that they are the most fundamental primitive that is always present in images. Note that other primitives such has lines or regions are groups of points and can be divided into this simpler primitive. Nevertheless, and despite the understandable reasons for the dominance of point features, we believe that it is worth exploring alternative primitives, such as planes, regions and normals, that in specific contexts of application can be advantageous with respect to the dominating trend.

1.1 Thesis outline and contributions

In this thesis, we analyse until which extent the use of features other than keypoints may help in solving the problems of calibration, SfM, 3D reconstruction and registration. These problems are addressed using planes, affine regions and normals, and the main contributions are presented in the next chapters as follows:

Chapter 2 shows an elegant formulation for computing the rigid motion from plane matches. This is explored for calibration in this chapter and for SfM in man-made environments using RGB-D cameras and stereo pairs in Chapter 3. The calibration algorithm is further extended to calibrate multi-modal sensor arrangements with non-overlapping FoV. The detailed contributions are the following:

- A new optimal minimal solution for registering two sets of corresponding 3D planes that is, for the first time, explained in the dual projective space.
- A new method for calibrating a color-depth camera pair which outperforms the state-of-the-art method [53], requiring about 1/6 of the number of input frames and running in 1/30 of the time. This is accomplished by a new optimization step that prevents the calibration to suffer from a drift in scale, as well as a method for estimating a depth distortion model which significantly improves the calibration accuracy. The calibration toolbox corresponding to this algorithm can be accessed at `http://arthronav.isr.uc.pt/~carolina/kinectcalib/`.

- An accurate, easy to use method that combines different modules for accomplishing the extrinsic calibration of sensor setups comprising cameras, depth sensors and LRFs with non-overlapping FoV. The toolbox implementing this method is available in `http://arthronav.isr.uc.pt/~carolina/toolbox_multimodal/`.

The contents in this chapter led to the articles [99, 100].

Chapter 3 describes an hierarchical RANSAC scheme for computing the camera motion from planes extracted from the scene. This scheme is employed in two pipelines that combine the benefits of PPR and plane-based odometry by recovering both structure and motion from plane-primitives. It is shown that using planes as opposed to points not only enables to handle situations that cannot be handled by the latter, but also improves overall accuracy while providing visually pleasant reconstructions. Our contributions are:

- A robust scheme that performs plane registration to estimate the relative camera pose. It works in a hierarchical manner in the sense that it uses as many planes as possible to perform the computation and point correspondences are only extracted when strictly necessary. When 3 plane correspondences are available, it makes use of the plane registration algorithm proposed in Chapter 2. Solutions for the cases of 1 and 2 plane correspondences are also proposed.
- A pipeline that detects the scene planes from images acquired by an RGB-D camera and uses the proposed hierarchical scheme for initializing the camera motion. The initial estimate is refined in a photo-geometric optimization step that takes full advantage of the plane detection and simultaneous availability of depth and visual appearance cues. Experiments show that it is as accurate as state-of-the-art point-based approaches when the camera displacement is small, and significantly outperforms them in case of wide-baseline and/or dynamic foreground. This pipeline is available in `http://arthronav.isr.uc.pt/~carolina/kinect_odometry/`.
- A pipeline for PPR and camera motion estimation that takes as input a sequence acquired by a calibrated stereo rig. The planes are extracted in each stereo pair using SymStereo [4] and the camera motion is initialized using the new hierarchical scheme. The camera motion and the planes are refined simultaneously using a new formulation of a multi-modal fitting approach that allows linking, fusing, and back-propagating plane

hypotheses across stereo pairs. A new Markov Random Field (MRF) formulation, specific for sequential PPR, that handles low-textured regions and ensures coherence in visibility across views is proposed.

- A dataset containing several sequences acquired with 3 different stereo camera setups made available to the research community in `http://montecristo.co.it.pt/PPR_Rec/`.

This material led to the works [98, 104, 105].

Chapter 4 shows that ACs, that are currently discarded after performing point association, contain extremely useful information that can reduce the combinatorics of SfM and enable fast and reliable segmentation of planes, without the need of generating homography hypotheses. This led to the development of a new monocular SLAM pipeline that provides a dense PPR of the scene. In detail, our contributions are:

- The characterization of the family of homographies compatible with an AC and the subsequent demonstration that an AC puts 3 constraints on the epipolar geometry. These constraints can be directly used for estimating the essential and the fundamental matrices, from 2 and 3 ACs, respectively, by simply applying the 5-point [119] and the 7-point [47] algorithms. Using this approach, we propose a robust scheme for estimating the essential matrix from 2 ACs that greatly benefits from the reduced combinatorics and is extensively tested against the 5-point method [88] in real sequences. This provides experimental evidence that ACs are a viable alternative to Point Correspondence (PC)s for visual odometry and can be highly advantageous in the presence of many outliers as it happens in scenes with multiple moving objects and/or deformable surfaces.
- The derivation, for the first time, of the constraints that must be verified by a PC or AC to be compatible with the 2-parameter family of homographies associated with an initial AC. These are used as a metric for segmenting correspondences according to planes present in the scene. Comparative experiments show the benefits of this direct metric with respect to sophisticated global multi-model fitting approaches [71] that require the generation of hypotheses.
- The development of πMatch, a new monocular vSLAM pipeline that makes use of the proposed plane segmentation method. It is able to

decide about multiple motion situations (*e.g.* dynamic foreground, pure rotation of camera, *etc.*) and provides a dense PPR of the scene. Experiments show the superiority of our approach w.r.t. state-of-the-art monocular methods [44].

These contributions were published in the articles [101, 102].

Chapter 5 describes a 3D registration method that works by extracting pairs of oriented points, *i.e.* points with associated normals, for computing the rigid transformation. A comparison with the state-of-the-art method [79], that works solely with points, shows that including normals leads to similar or better results in less than 1/100 of the time in noise-free synthetic datasets. The computational speed up for real datasets is over 19 times.

This material was recently submitted to an international robotics conference [103].

Chapter 6 presents some concluding remarks.

1.2 Notation

Scalars are represented by plain letters, *e.g.* x, vectors are indicated by bold symbols, *e.g.* $\mathbf{x}$, and matrices are denoted by letters in sans serif font, *e.g.* R. Planes are represented by a 4D homogeneous vector that is indicated by an uppercase Greek letter, *e.g.* $\mathbf{\Pi}$. Sets of intrinsic parameters and point clouds are defined by uppercase calligraphic letters, *e.g.* $\mathcal{I}, \mathcal{P}$. The symbol $=$ will be used to denote strict equality and $\sim$ to represent equality up to scale between projective representations.

Chapter 2

Plane Registration in the Dual Space and Applications in Calibration

Plane registration in 3D has applications in areas such as computer vision, computer graphics and robotics. In particular, it has been used in the past for sensor calibration [53], where the authors propose a linear sub-optimal solution for this task. In this chapter we propose a new minimal solution for 3D plane registration that is used in the calibration of cameras and depth sensors. We start by giving an elegant formulation of the problem in the dual space (Section 2.1), and propose a fast and accurate method for the intrinsic and extrinsic calibration of a Kinect sensor (Section 2.2). This calibration approach is afterwards extended to handle situations of sensor arrangements with non-overlapping FoV. Similarly, the state-of-the-art solution for the calibration of camera-LRF pairs [135] is also extended to handle situations of disjoint FoVs. The outcome is a new method for calibrating multi-modal sensor arrangements - that include cameras, LRFs and depth sensors - with non-overlapping FoV that works effectively in scenarios for which there is no simple solution in the current state-of-the-art (Section 2.3).

Notation: In this chapter, entities in LRF reference frame are represented using $'$ and in depth camera reference frame using $\mathring{}$, e.g. $\mathbf{\Pi}'$ and $\mathring{\mathbf{\Pi}}$. With abuse of notation, the pose and the homography will be denoted by the same symbol, whenever it is convenient.

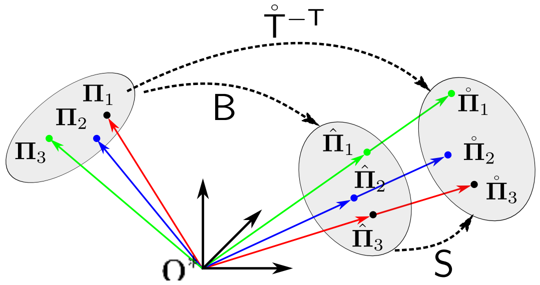

$\mathring{\mathsf{T}}^{-\mathsf{T}}$
B
S
O
$\boldsymbol{\Pi}_1$
$\boldsymbol{\Pi}_2$
$\boldsymbol{\Pi}_3$
$\hat{\boldsymbol{\Pi}}_1$
$\hat{\boldsymbol{\Pi}}_2$
$\hat{\boldsymbol{\Pi}}_3$
$\mathring{\boldsymbol{\Pi}}_1$
$\mathring{\boldsymbol{\Pi}}_2$
$\mathring{\boldsymbol{\Pi}}_3$

Figure 2.1: The relative pose estimation can be cast as a point registration problem in the dual projective space $\mathcal{P}^{3*}$ that maps points $\boldsymbol{\Pi}_i$ into points $\mathring{\boldsymbol{\Pi}}_i$. The factorization $\mathring{\mathsf{T}}^{-\mathsf{T}} \sim \mathsf{SB}$ allows the rotation and translation components to be computed separately.

2.1 Plane registration in the dual space

The rigid transformation $\mathring{\mathsf{T}}$ between two reference frames, with

$$\mathring{\mathsf{T}} = \begin{bmatrix} \mathring{\mathsf{R}} & \mathring{\mathbf{t}} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{2.1}$$

where $\mathring{\mathsf{R}}$ is the rotation matrix and $\mathring{\mathbf{t}}$ the translation vector, can be estimated by solving a 3D plane registration problem. The goal is to find the transformation $\mathring{\mathsf{T}}$ that satisfies the condition

$$\mathring{\boldsymbol{\Pi}}_i \sim \underbrace{\begin{bmatrix} \mathring{\mathsf{R}} & \mathbf{0} \\ -\mathring{\mathbf{t}}^\mathsf{T}\mathring{\mathsf{R}} & 1 \end{bmatrix}}_{\mathring{\mathsf{T}}^{-\mathsf{T}}} \boldsymbol{\Pi}_i, i = 1, 2, 3, \tag{2.2}$$

where $\boldsymbol{\Pi}_i \sim [\mathbf{n}_i \quad 1]^\mathsf{T}$ and $\mathring{\boldsymbol{\Pi}}_i \sim [\mathring{\mathbf{n}}_i \quad 1]^\mathsf{T}$ are the planes represented in each reference frame, in homogeneous coordinates, with normal vectors $\mathbf{n}_i$ and $\mathring{\mathbf{n}}_i$, respectively.

Knowing that points and planes are dual entities in $3D$ - a plane in the projective space $\mathcal{P}^3$ is represented as a point in the dual space $\mathcal{P}^{3*}$, and vice-versa - Equation 2.2 can be seen as a projective transformation in $\mathcal{P}^{3*}$ that maps points $\boldsymbol{\Pi}_i$ into points $\mathring{\boldsymbol{\Pi}}_i$. As shown in Figure 2.1, the transformation $\mathring{\mathsf{T}}^{-\mathsf{T}}$ can be factorized into a rotation transformation B, mapping points $\boldsymbol{\Pi}_i$ into points $\widehat{\boldsymbol{\Pi}}_i$, and a projective scaling S that maps points $\widehat{\boldsymbol{\Pi}}_i$ into points $\mathring{\boldsymbol{\Pi}}_i$:

$$\mathsf{B} = \begin{bmatrix} \mathring{\mathsf{R}} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}, \quad \mathsf{S} = \begin{bmatrix} \mathsf{I} & \mathbf{0} \\ -\mathring{\mathbf{t}}^\mathsf{T} & 1 \end{bmatrix}, \tag{2.3}$$

where I is the 3×3 identity matrix. B can be computed from $K = 2$ point-point correspondences but S requires $K = 3$ correspondences to be estimated. An easy two-step process to perform the registration is presented:

1. Since the length of a vector is not changed by rotation, $\mathbf{n}_i$ and $\mathring{\mathbf{n}}_i$ are normalized, and an algorithm derived from [55] for computing a transformation between two sets of unitary vectors is applied. From Equation 2.2, it is known that this transformation is a pure rotation, and thus the translation component is not computed.

2. The computation of the projective scaling S that maps points $\widehat{\mathbf{\Pi}}_i$ into points $\mathring{\mathbf{\Pi}}_i$, with $\widehat{\mathbf{\Pi}}_i = \mathsf{B}\mathbf{\Pi}_i$, $i = 1, 2, 3$, is done similarly to [135]. We can write $\mathring{\mathbf{\Pi}}_i \sim \mathsf{S}\widehat{\mathbf{\Pi}}_i$, yielding

$$\mu_i \begin{bmatrix} \mathring{\mathbf{n}}_i \\ 1 \end{bmatrix} = \begin{bmatrix} \mathsf{I} & \mathbf{0} \\ -\mathring{\mathbf{t}}^\mathsf{T} & 1 \end{bmatrix} \begin{bmatrix} \mathring{\mathsf{R}}\mathbf{n}_i \\ 1 \end{bmatrix}, \tag{2.4}$$

where μ_i is an unknown scale factor. From the top three equations we can write

$$\mu_i = \frac{\mathring{\mathbf{n}}_i^\mathsf{T} \mathsf{R} \mathbf{n}_i}{\mathring{\mathbf{n}}_i^\mathsf{T} \mathring{\mathbf{n}}_i},$$

and replacing in the bottom one, it comes that

$$\mathring{\mathbf{n}}_i^\mathsf{T} \mathring{\mathbf{n}}_i \mathbf{n}_i^\mathsf{T} \mathring{\mathsf{R}}^\mathsf{T} \mathring{\mathbf{t}} - \mathring{\mathbf{n}}_i^\mathsf{T} \mathring{\mathbf{n}}_i + \mathring{\mathbf{n}}_i^\mathsf{T} \mathring{\mathsf{R}} \mathbf{n}_i = 0. \tag{2.5}$$

Each pair $\mathbf{\Pi}_i$, $\mathring{\mathbf{\Pi}}_i$ gives rise to a linear constraint in the entries of the translation vector $\mathring{\mathbf{t}}$, which can be computed by $\mathring{\mathbf{t}} = \mathsf{A}^{-1}\mathbf{c}$, with

$$\mathsf{A} = \begin{bmatrix} \mathring{\mathbf{n}}_1^\mathsf{T} \mathring{\mathbf{n}}_1 & 0 & 0 \\ 0 & \mathring{\mathbf{n}}_2^\mathsf{T} \mathring{\mathbf{n}}_2 & 0 \\ 0 & 0 & \mathring{\mathbf{n}}_3^\mathsf{T} \mathring{\mathbf{n}}_3 \end{bmatrix} \underbrace{\begin{bmatrix} \mathbf{n}_1^\mathsf{T} \\ \mathbf{n}_2^\mathsf{T} \\ \mathbf{n}_3^\mathsf{T} \end{bmatrix}}_{\mathsf{N}_i} \mathring{\mathsf{R}}^\mathsf{T}, \mathbf{c} = \begin{bmatrix} \mathring{\mathbf{n}}_1^\mathsf{T} \mathring{\mathbf{n}}_1 - \mathring{\mathbf{n}}_1^\mathsf{T} \mathsf{R} \mathbf{n}_1 \\ \mathring{\mathbf{n}}_2^\mathsf{T} \mathring{\mathbf{n}}_2 - \mathring{\mathbf{n}}_2^\mathsf{T} \mathsf{R} \mathbf{n}_2 \\ \mathring{\mathbf{n}}_3^\mathsf{T} \mathring{\mathbf{n}}_3 - \mathring{\mathbf{n}}_3^\mathsf{T} \mathsf{R} \mathbf{n}_{c3} \end{bmatrix}. \tag{2.6}$$

It comes in a straightforward manner that if the 3 normals do not span the entire 3D space, then N_i is rank deficient and the computation of the translation becomes underdetermined. This occurs when either all the normals are coplanar or all the planes are parallel.

2.2 Fast calibration of RGB-D cameras

Nowadays, the joint information provided by cameras and depth sensors has applications in areas including scene reconstruction, indoor mapping, and mobile robotics. The Kinect is a camera pair capable of providing such information. Its depth sensor consists of a projector that emits a dot pattern which is detected by an infrared (IR) camera. The Kinect has been used for multiple purposes including 3D modelling of indoor environments [50], and SfM [115]. Most of these applications require the camera pair to be calibrated both intrinsically and extrinsically. The intrinsic calibration consists in determining the parameters that enable to convert measurement units into metric units. The extrinsic calibration consists in locating the sensors in a common coordinate frame, for them to function as a whole.

The Kinect is a device for the consumer market of games and entertainment. The intrinsic parameters of both depth and color cameras, as well as their relative pose, are pre-calibrated in factory and recorded in the firmware. Average values for these parameters are known by the community and commonly used in robotic applications [20]. However, it is well known that these parameters vary from device to device, and that the factory presets are not accurate enough for many applications [50, 115]. This justifies the development of calibration methods for the Kinect, or of methods to refine and improve the accuracy of the factory presets.

Authors have tried to independently calibrate the intrinsics of the depth sensor and color camera, and then register both in a common reference frame [109, 141]. As pointed out by Herrera *et al.* [53], the depth and the color camera must be calibrated together both because the accuracy in the color camera propagates to the depth camera, and because all available information is being used.

Depth sensors may present depth distortions which decrease their accuracy. This is the case of the Kinect device which has shown radially symmetric distortions [115] that are not corrected in the manufacturer's calibration. Herrera *et al.* [52] firstly proposed an algorithm that calibrates not only the cameras' intrinsics, but also the parameters that convert kdu into meters. Zhang and Zhang extend this work by considering point correspondences between color and depth images, showing improved accuracy. However, neither methods deal with the depth distortion.

Smisek *et al.* [115] observed that the Kinect exhibited residuals for close range measurements, and were the first to propose considering both distortion in the projection and in the depth estimation. The depth distortion was estimated for each pixel by averaging the metric error, after carrying the intrinsic and extrinsic calibrations of the device. More recently, another depth distortion correction procedure was proposed by Herrera *et al.* [53], which leads to improved accuracy. They initially estimate the intrinsics and the plane pose from homographies computed using plane-to-image correspondences. The extrinsic calibration is carried by registering the 3D planes estimated in color and depth camera coordinates. Due to the high number of parameters to be optimized, they use an iterative refinement step that optimizes the parameters alternately. Unfortunately, in order to effectively model the depth camera parameters, including the distortion term, it requires many images (≥ 20). Also, its iterative optimization step is highly time consuming. Due to its high accuracy, we build on this contribution and downsize the calibration pipeline to enhance usability.

As in [53], the color camera is calibrated from plane-to-image homographies which enable to know the pose of the calibration plane in the color camera reference frame. Concerning the depth camera, we use the preset values to reconstruct 3D points, and compute the calibration plane pose using a standard fitting algorithm. Computation of the extrinsic calibration is accomplished by performing plane registration. While Herrera carries the registration using a sub-optimal linear algorithm, we use the minimal solution proposed in Section 2.1 in a hypothesize-and-test framework. This provides better initializations of the relative pose, which facilitate the subsequent steps. All parameters are optimized in a bundle adjustment step which considers metric information in order to avoid a drift in the disparity to depth conversion. We use the depth distortion model presented in [53], which has shown to yield good accuracy. However, unlike Herrera's method, we estimate the model's parameters in an open-loop, making our approach much less time consuming.

This pipeline leads to improvements in usability without sacrificing the final accuracy. The improvements are both in terms of decreasing the number of input images by a factor of 6, and reducing the computational time by a factor of 30. Our method, as Herrera's method, can be used with more than one color camera. However, in this work, we only consider the Kinect's color camera in the experiments.

2.2.1 Background

2.2.1.1 Projection model

In this work, the intrinsic parameters of the color camera are modelled as in [49], where both radial and tangential distortions are considered. Let $\mathbf{X}_c = [X_c, Y_c, Z_c]^\mathsf{T}$ be a 3D point in the camera reference frame. The normalized image projection of

$\mathbf{X}_c$ is $\mathbf{x}_n = [x_n, y_n]^\mathsf{T}$, with $x_n = X_c/Z_c$ and $y_n = Y_c/Z_c$. Including lens distortion, it comes that

$$\mathbf{x}_k = (1 + k_{c1}r^2 + k_{c2}r^4 + k_{c5}r^6)\mathbf{x}_n + \mathbf{d}_x, \tag{2.7}$$

where $r^2 = x_n^2 + y_n^2$ and $\mathbf{d}_x$ is the tangential distortion. The pixel coordinates $\mathbf{x}_c = [x_c, y_c]^\mathsf{T}$ of the projection of $\mathbf{X}_c$ on the image plane are

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} f_{cx} & 0 \\ 0 & f_{cy} \end{bmatrix} \begin{bmatrix} x_k \\ y_k \end{bmatrix} + \begin{bmatrix} c_{cx} \\ c_{cy} \end{bmatrix}, \tag{2.8}$$

where $\mathbf{f}_c = [f_{cx}, f_{cy}]$ are the focal lengths, and $\mathbf{c}_c = [c_{cx}, c_{cy}]$ is the principal point. We refer to the set of intrinsic parameters of the color camera by $\mathcal{I} = \{\mathbf{f}_c, \mathbf{c}_c, \mathbf{k}_c\}$.

The pixel coordinates of the projection of the 3D point $\mathbf{X}_d$ in depth camera coordinates can be obtained using a model similar to the color camera's one. The parameters $\mathbf{f}_d$ and $\mathbf{c}_d$ are the focal length and the principal point of the depth camera, respectively. Since considering the distortion in the depth camera does not improve accuracy significantly, $\mathbf{k}_d$ is set to zero.

2.2.1.2 Depth measurements

The Kinect's depth sensor consists of an IR camera which detects a constant pattern emitted by a projector. It delivers depth information in disparity units (kdu) which must be converted into metric units (meters). This can be done by using a scaled inverse of the format

$$z = \frac{1}{c_1 d_u + c_0}, \tag{2.9}$$

where c_0 and c_1 are part of the depth camera's intrinsics. Depth z is obtained from d_u, which is the undistorted disparity, *i.e.*, after performing distortion correction. The Kinect's depth sensor presents a depth distortion which has been modelled by Herrera *et al.* [53]:

$$d_u = d + \mathsf{D}(x_d, y_d)e^{\alpha_0 - \alpha_1 d}, \tag{2.10}$$

where d is the disparity returned by the Kinect in pixel $[x_d, y_d]$, D contains the spatial distortion pattern, and $\alpha = [\alpha_0, \alpha_1]$ models the decay of the distortion effect. We refer to the set of intrinsic parameters of the depth camera by $\mathring{\mathcal{I}}^* = \{\mathbf{f}_d, \mathbf{c}_d, \mathbf{k}_d, c_0, c_1, \mathsf{D}, \alpha\}$.

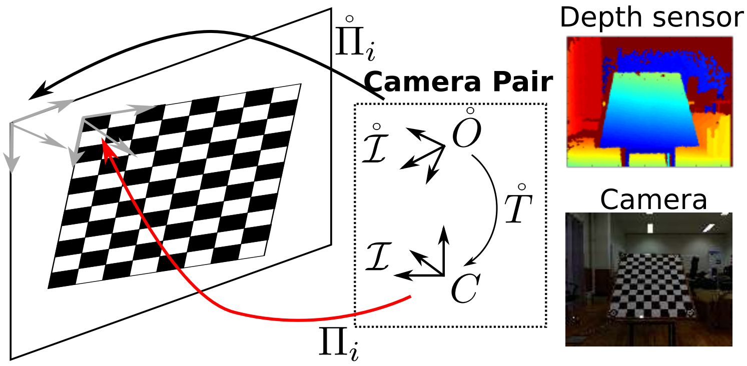


Figure 2.2: The color and depth cameras are related by a rigid transformation $\mathring{\mathsf{T}}$. Both sensors observe the same planar surface, allowing the computation of the extrinsic calibration.

2.2.1.3 Herrera's method

Herrera *et al.* [53] recently proposed a new method for calibrating a color-depth camera pair, as well as a new explicit distortion correction term for the Kinect device, which significantly improves accuracy. They use a setup identical to Figure 2.2, where all cameras observe a planar checkerboard pattern from multiple views, which are used for calibrating the sensors. A diagram of Herrera's method is shown in Figure 2.3, where it can be seen that the initialization steps can be performed by two different methods, yielding two versions of the method to which we refer by Herrera and Herrera I. The remaining steps do not depend on how the initial estimate was obtained, and constitute the non-linear minimization.

Initial estimation

The color camera intrinsics can be initialized using Zhang's method [146]. The checkerboard corners are extracted from the intensity images and, using known corner positions in the checkerboard reference frame, both the intrinsic parameters and the plane to image homographies can be estimated. This leads to the initialization of $\mathcal{I}$ and $\mathbf{\Pi}_i$, for all input images i.

The same method can be applied to estimate the depth camera parameters and homographies using plane corners [53]. From these initial parameters, it is possible to obtain an estimate for the expected depth of each selected corner. The corresponding measured disparities can be used for determining an initial guess for c_0 and c_1, using Equation 2.9. Thus, by setting D and α to zero, an initialization of

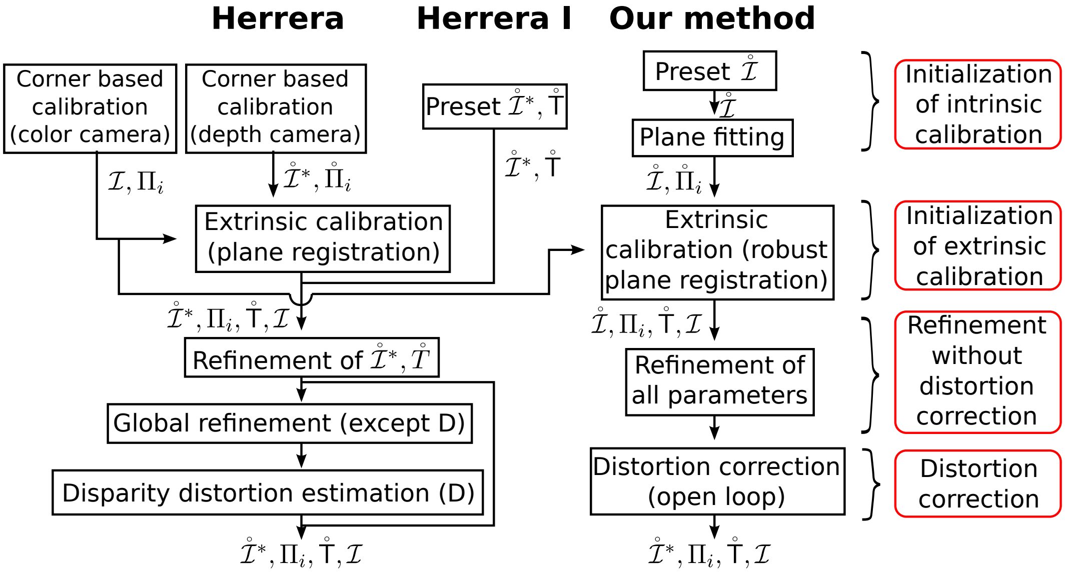

Herrera
Herrera I
Our method
Corner based calibration (color camera)
Corner based calibration (depth camera)
Preset $\mathring{\mathcal{I}}^*, \mathring{\mathsf{T}}$
Preset $\mathring{\mathcal{I}}$
$\mathring{\mathcal{I}}$
Plane fitting
Initialization of intrinsic calibration
$\mathcal{I}, \Pi_i$
$\mathring{\mathcal{I}}^*, \mathring{\Pi}_i$
$\mathring{\mathcal{I}}^*, \mathring{\mathsf{T}}$
$\mathring{\mathcal{I}}, \mathring{\Pi}_i$
Extrinsic calibration (plane registration)
Extrinsic calibration (robust plane registration)
Initialization of extrinsic calibration
$\mathring{\mathcal{I}}^*, \Pi_i, \mathring{\mathsf{T}}, \mathcal{I}$
$\mathring{\mathcal{I}}, \Pi_i, \mathring{\mathsf{T}}, \mathcal{I}$
Refinement of $\mathring{\mathcal{I}}^*, \mathring{T}$
Refinement of all parameters
Refinement without distortion correction
Global refinement (except D)
Disparity distortion estimation (D)
Distortion correction (open loop)
Distortion correction
$\mathring{\mathcal{I}}^*, \Pi_i, \mathring{\mathsf{T}}, \mathcal{I}$
$\mathring{\mathcal{I}}^*, \Pi_i, \mathring{\mathsf{T}}, \mathcal{I}$

Figure 2.3: Calibration algorithms: two versions of Herrera's method (named Herrera and Herrera I), and our method.

$\mathring{\mathcal{I}}^*$ and $\mathring{\mathbf{\Pi}}_i$ is obtained. This initialization procedure is used in Herrera's method, as depicted in Figure 2.3. However, it produces a very rough initial estimate, especially if the number of calibration planes is small. Thus, since there exist publicly available values for the intrinsics of the Kinect device, in method Herrera I these are used, and the extrinsic calibration step is skipped since estimates for $\mathring{\mathcal{I}}^*$ and $\mathring{\mathsf{T}}$ are known.

Extrinsic calibration

In method Herrera, it is necessary to explicitly compute the relative pose between the sensors $\mathring{\mathsf{T}}$. From Figure 2.2, it is evident that the checkerboard and calibration plane reference frames are not aligned, and thus there is not a common reference frame between the two sensors. This means that is it not possible to find $\mathring{\mathsf{T}}$ by simply chaining transformations $\mathbf{\Pi}_i$ and $\mathring{\mathbf{\Pi}}_i$. However, $\mathring{\mathsf{T}}$ can be found through plane registration, since it is known that both planes are coplanar. Herrera *et al.* use a linear sub-optimal algorithm to carry this estimation.

Non-linear minimization

The non-linear minimization of Herrera's method consists of 3 steps, as shown in the diagram of Figure 2.3. It aims to minimize the weighted sum of squares of the measurement reprojection errors over all parameters ($\mathcal{I}$, $\mathring{\mathcal{I}}^*$, $\mathring{\mathsf{T}}$, and $\mathbf{\Pi}_i$ for all calibration images). For the color camera, the error is the Euclidean distance between the measured corner position $\widehat{\mathbf{x}_c}$ and its reprojected position $\mathbf{x}_c$ (first term of Equation 2.11). For the depth camera it is the difference between the measured disparity $\widehat{d}$ and the predicted disparity d. The errors are normalized using each measurement variance σ^2. It comes that the cost function is

$$c = \frac{\sum ||\widehat{\mathbf{x}}_c - \mathbf{x}_c||^2}{\sigma_c^2} + \frac{\sum (\widehat{d} - d)^2}{\sigma_d^2}. \tag{2.11}$$

The optimization process is divided into three steps: firstly, only $\mathring{\mathcal{I}}^*$ and $\mathring{\mathsf{T}}$ are optimized to account for the fact that they are poorly initialized; secondly, Equation 2.11 is minimized over all the parameters, except for D; lastly, D is optimized independently for each pixel. The two last steps are repeated until convergence is reached.

2.2.2 Proposed calibration method

We propose a new calibration method that consists of four main consecutive steps: an initial estimation of the intrinsic and extrinsic parameters, a non-linear minimization, and a depth distortion model estimation. Figure 2.3 shows a block diagram of our method, which presents a simpler framework than Herrera's. Our optimization procedure consists of only one step, and a depth distortion model is estimated using the optimized parameters.

2.2.2.1 Initialization of intrinsic calibration

For the color camera, the initial estimation of $\mathcal{I}$ and $\mathbf{\Pi}_i$ for all calibration images is done as described in the previous subsection, for which we use Bouguet's toolbox [17]. We redefine the intrinsic parameters of the depth camera as $\mathring{\mathcal{I}} = \{\mathbf{f}_d, \mathbf{c}_d, \mathbf{k}_d, c_0, c_1\}$ because we do not consider depth distortion terms. They are initialized using preset values, which are publicly available for the Kinect [20].

2.2.2.2 Initialization of extrinsic calibration

For each input disparity map i, the plane corners are extracted, defining a polygon. For each point $\mathbf{x}_d$ inside the polygon, the corresponding disparity d is used for computing a depth value z_d using Equation 2.9, where $d = d_u$ since the measured disparities are used. The correspondences (x_d, y_d, z_d) are used for computing 3D

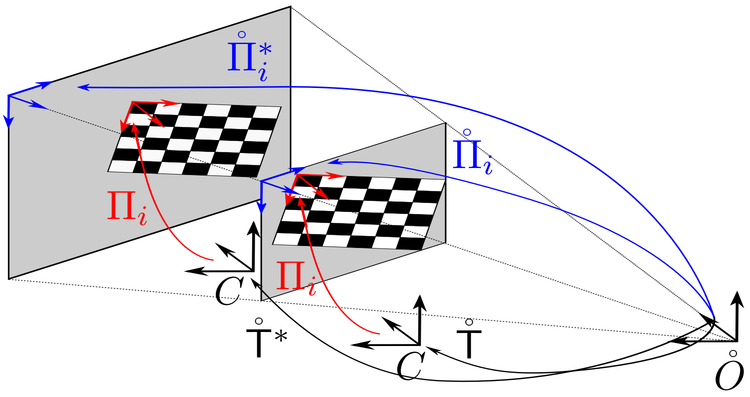


Figure 2.4: The problem of occurring a drift in scale. The pose of grid in the color camera reference frame is fixed, while the depth camera may observe the calibration plane at different depths.

points $\mathbf{X}_d$ originating a 3D point cloud. To each 3D point cloud, a plane is fitted using a standard total least squares algorithm, yielding $\mathring{\mathbf{\Pi}}_i$.

The plane registration algorithm proposed in Section 2.1 provides the extrinsic calibration of a camera and a depth sensor in the case of $K = 3$ correspondences. For $K > 3$ pairs of planes, each triplet of plane-plane correspondences gives rise to one solution, and the best estimation can be found using an hypothesize-and-test framework:

1. For each possible triplet of pairs of planes $\mathbf{\Pi}_i$, $\mathring{\mathbf{\Pi}}_i$, a transformation $\mathring{\mathsf{T}}$ is estimated.

2. For each solution $\mathring{\mathsf{T}}$, the depth camera coordinates $\mathring{\mathbf{\Pi}}_i^*$ for all $\mathbf{\Pi}_i$ are computed using Equation 2.2, and the euclidean distance d_i in the dual space between the computed $\mathring{\mathbf{\Pi}}_i^*$ and $\mathring{\mathbf{\Pi}}_i$ is determined.

3. Each solution is ranked by $rank(\mathring{\mathsf{T}}) = \sum_i \max(t, d_i)$, where t is a predefined threshold. The correspondences for which $d_i < t$ are considered as inliers.

4. Find $\mathring{\mathsf{T}}$ for which $rank(\mathring{\mathsf{T}})$ is minimum.

After obtaining an initial estimation for the transformation $\mathring{\mathsf{T}}$, and a set of inlier correspondences, a bundle adjustment procedure is performed.

2.2.2.3 Non-linear minimization

It was experimentally observed that under poor initialization and a small number of images, Herrera's method tends to drift in depth. After careful analysis, we came up with an hypothesis for this observation. From Equation 2.9, it can be seen that if c_0 and c_1 are affected by a scale component, an error in the extrinsic calibration will occur, while the reprojection error does not change. Figure 2.4 depicts the problem, where it can be seen that a scale drift will cause the pose of the calibration plane w.r.t. the depth camera $\mathring{\mathbf{\Pi}}_i^*$ to be incorrectly estimated, while its pose relative to the color camera $\mathbf{\Pi}_i$ is not affected. This automatically originates an erroneous estimation of the relative pose, represented by $\mathring{\mathsf{T}}^*$.

Thus, we change the cost function 2.11 by adding a term that accounts for the difference between the Euclidean distances between points of an object λ and the measured distances between those points $\widehat{\lambda}$. Our objective function is, then,

$$\min_{\mathcal{I},\mathring{\mathcal{I}},\mathring{\mathsf{T}},\mathbf{\Pi}_i} e = \frac{\sum ||\widehat{\mathbf{x}}_c - \mathbf{x}_c||^2}{\sigma_c^2} + \frac{\sum (\widehat{d} - d)^2}{\sigma_d^2} + \beta|\widehat{\lambda} - \lambda|^2, \tag{2.12}$$

where β is a weighting factor which should be sufficiently high. This information could be included as a hard constraint. However, since we do not know how accurate the measurements are, we decided to include it as a penalty term. This means that our algorithm requires at least one image of an object with known dimensions, for avoiding the calibration to drift in scale.

2.2.2.4 Depth distortion model estimation

The optimized intrinsic and extrinsic calibrations can be used for estimating the depth distortion model of Equation 2.10. Note that it can be rewritten as

$$d_u = d + \mathsf{W}(x_d, y_d)e^{-\alpha_1 d}, \tag{2.13}$$

where $\mathsf{W}(x_d, y_d) = \mathsf{D}(x_d, y_d)e^{\alpha_0}$.

For a pair of disparity maps where a given pixel $\mathbf{x}_d$ belongs to the calibration plane in both maps, there are two correspondences $(\tilde{d}_1, d_1)$ and $(\tilde{d}_2, d_2)$, where d is the measured disparity and $\tilde{d}$ is the expected disparity computed by knowing the plane equation. Using the two correspondences, we can write the system of equations

$$\begin{cases} \tilde{d}_1 - d_1 = \mathsf{W}(x_d, y_d)e^{-\alpha_1 d_1} \\ \tilde{d}_2 - d_2 = \mathsf{W}(x_d, y_d)e^{-\alpha_1 d_2} \end{cases} \tag{2.14}$$

and find α_1 by

$$\alpha_1 = \frac{\ln \frac{\tilde{d}_1 - d_1}{\tilde{d}_2 - d_2}}{d_2 - d_1}. \tag{2.15}$$

For every possible pair of correspondences, we compute an estimate for α_1 and consider their average as the final result.

Knowing α_1, W can directly be estimated for the pixels which belong to a known plane. For pixel (x_d, y_d), if more than one value is found, the average of all values is considered. Although it is not possible to find individual estimates for α_0 and D, this method allows to recover the whole depth distortion function. Like Smisek *et al.* [115], we perform the estimation in open-loop. However, since we use Herrera's model, we obtained better accuracy.

2.2.3 Experimental results

Two sets of experiments were conducted in order to compare the accuracy of Herrera's method, which has been released as a toolbox, and our method. The first one uses the dataset included in the toolbox, and shows extensive results with a varying number of calibration images. The second set uses a small number of images acquired by another Kinect, in order to further validate the results.

2.2.3.1 Herrera's dataset

The dataset comprises image-disparity map pairs for 70 distinct plane poses, with the images being both acquired by the Kinect's color camera and an external high resolution camera. We selected 10 image-disparity map pairs acquired by the Kinect (validation set) and used the rest of the data as input to the original Herrera algorithm, that was executed with and without distortion correction (DC). Figure 2.5 shows the reprojection error measured for the validation set, where it can be seen that the latter is substantially more accurate than the former. We will consider this last calibration result as being close to the ground truth, and refer to it as *pseudo ground truth*, given the large amount of data and the use of a high resolution camera. However, it is merely indicative, since we do not know how exact the calibration is.

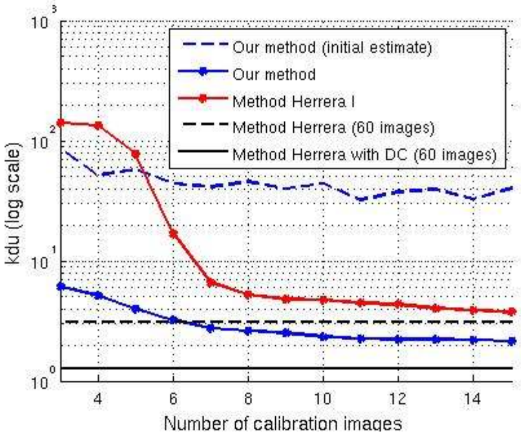
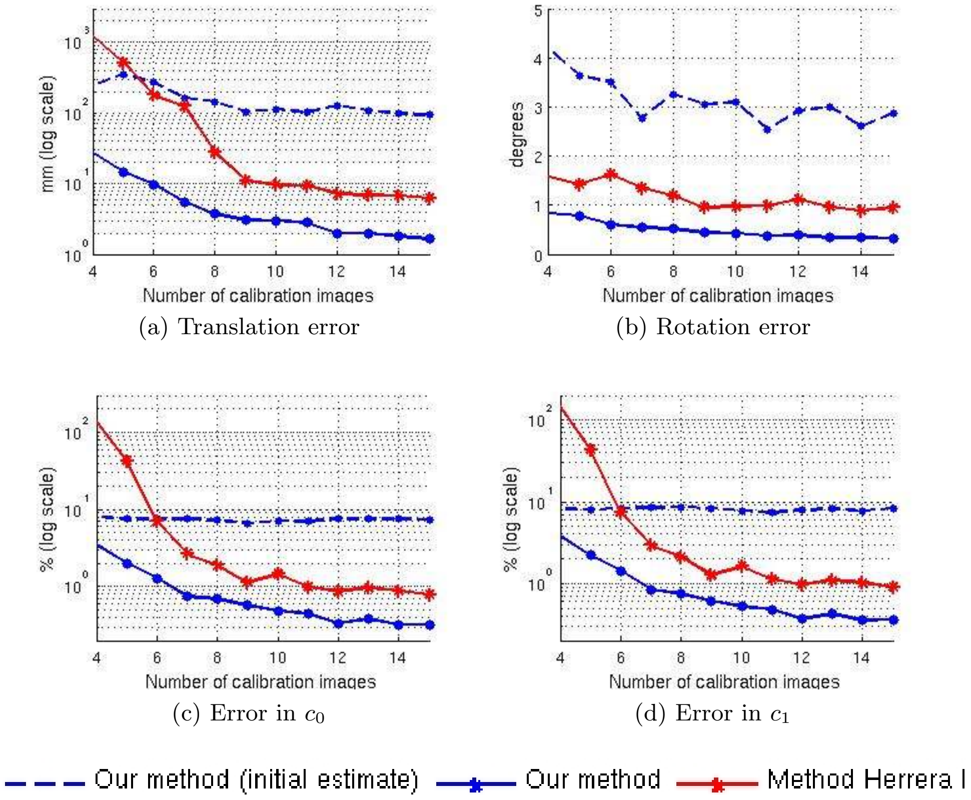


Figure 2.5: Average RMS reprojection errors in kdu obtained with the validation set of 10 images. All calibrations were performed without distortion correction (DC), except for one using a dataset with 60 plane poses (pseudo ground truth).



(a) Translation error

(b) Rotation error

(c) Error in c_0

(d) Error in c_1

Figure 2.6: Errors relative to the pseudo ground truth obtained without performing distortion correction.

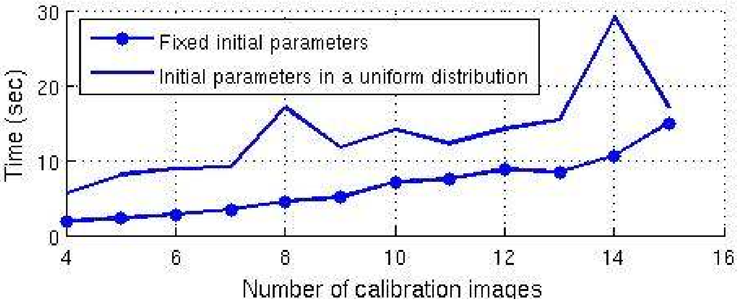
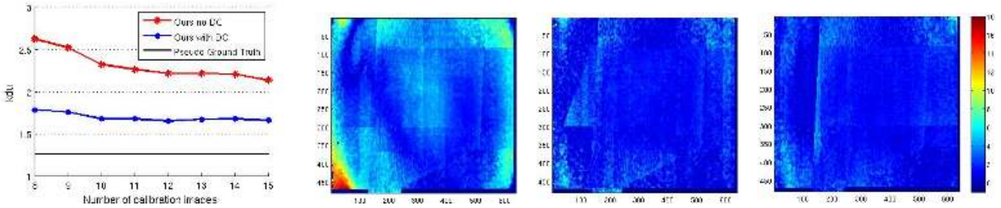


Figure 2.7: Average run times obtained with our method, for increasing number of calibration images.

The estimations by the different methods are compared against this one. From the test set we selected 20 image-disparity map pairs acquired by the Kinect. These pairs were grouped in sets of $K = 4, 5, \ldots, 15$, and we randomly picked 50 sets for each value of K. For each group of input images, we ran the calibration using Herrera I and our method. The initial values were sampled from a uniform distribution with amplitude equal to 5% of the original value. The idea was to evaluate the robustness to poor initialization.

For each trial, we evaluated the result in terms of reprojection error, using the 10 validation images, and in terms of extrinsics, by comparing with the pseudo ground truth. Figures 2.5 and 2.6 show the average errors for increasing number of K input images. Results clearly show that under the same conditions, our method systematically outperforms Herrera's method, which is not capable of producing acceptable results with small datasets ($\leq$ 8 calibration images). Our method, on the other hand, yields good results with only 6 calibration images. Although the initial estimates are very poor, both methods are capable of converging during the optimization phase. Figure 2.7 shows the average run times of our method, when using fixed initial parameters or parameters sampled from a uniform distribution. When the parameters are not fixed, a poorer initial estimation may be obtained, leading to higher run times in the optimization step. However, this is a low time consuming method since it never exceeds 30 seconds. Using the results obtained with calibration sets of more than 7 images, we estimated the depth distortion model with 2 images of a wall at different depths. The average RMS reprojection errors for the validation images are shown in Figure 2.8a. It can be seen that the model was correctly estimated since the reprojection errors significantly decreased. This can be confirmed in Figure 2.8 where the average reprojection errors obtained in each



(a) Average results (b) Our without DC (c) Our with DC (d) Pseudo GT

Figure 2.8: Results obtained with Herrera's dataset. (a) Average and per-pixel RMS reprojection errors obtained with the validation set for (b) our method without distortion correction (DC), (c) our method with DC, and (d) the pseudo groud truth.

	Our method	Method Herrera I
No DC	0.495°	0.743°
With DC	0.369°	0.602°

Table 2.1: Average angular error between all possible pairs of 10 reconstructed planes.

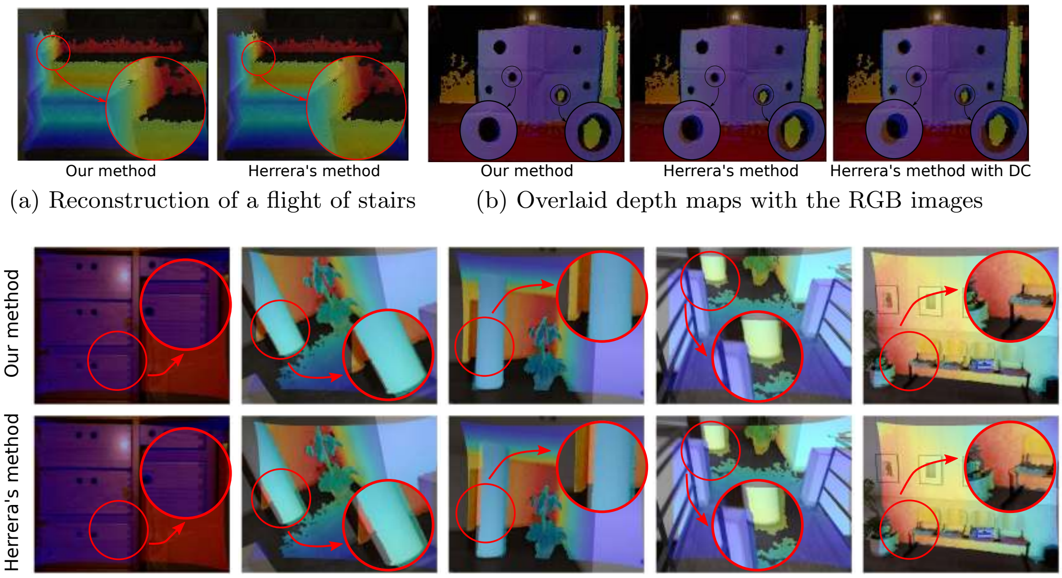
	Our method	Method Herrera I
No DC	1.54 kdu	4.08 kdu
With DC	1.20 kdu	3.61 kdu

Table 2.2: Average RMS reprojection errors obtained with the validation set of 6 images acquired by our Kinect.

pixel, for the 10 validation images, are shown. It can be seen that before correcting the distortion, a radial pattern of the residuals is observed. After applying the distortion correction, the reprojection errors significantly decrease, and the pattern obtained becomes very similar to the pseudo ground truth's. The estimation of the model with 2 images takes about 10 seconds, so that the overall run time is of about 30 seconds for 15 calibration images. Herrera's method, however, is much more time consuming, taking about 3 minutes with 20 images.

2.2.3.2 Our dataset

In this set of experiments, we acquired a dataset of 14 images, of which 8 were used for calibration and 6 for validation. We used the 8-image dataset for calibrating the camera pair with ours and Herrera I method, both with and without distortion correction. Note that our estimation of the depth distortion model is done with the 8 images of the calibration set. The quality of the depth camera's intrinsic calibration is assessed by reconstructing the planes of a flight of perpendicular stairs, and computing the angles between all possible pairs of planes (Figure 2.9a). These are compared with 90° if the planes are orthogonal, and 0° if they are parallel. Results in Table 2.1 show that, although both methods perform well, ours yields smaller



(a) Reconstruction of a flight of stairs

(b) Overlaid depth maps with the RGB images

(c) Overlaid depth maps with the RGB images

Figure 2.9: (a) Reconstruction of a flight of stairs yielded an average angular error between all possible pairs of planes of 0.495° with our method and 0.743° with Herrera's method. Note that a slight misalignment is observed when the depth map is overlaid with the RGB image using Herrera's solution, confirming it is less accurate than ours. This inaccuracy is evident in (b), where the depth maps are overlaid with the RGB image of an object with holes. While our method provides a good alignment, Herrera's method does not, and performs worse when distortion correction (DC) is applied. (c) Examples of RGB images - acquired in scenarios with very different depths and object shapes - overlaid with the corresponding depth maps using the calibration obtained with our method (top row) and Herrera's method (bottom row), both without DC.

angular errors in average. Applying distortion correction leads to a more accurate reconstruction in both cases. Average Root Mean Square (RMS) reprojection errors were computed for the validation set and results are shown in Table 2.2. As expected, our method outperforms Herrera's since the calibration set is not large enough for it to produce good results.

Although using distortion correction leads to an improvement in the accuracy for both methods, Figure 2.9 shows that in Herrera's method, it leads to a poorer extrinsic calibration. The 3D points computed from the disparity image are represented in color camera coordinates, to which colors are assigned. A correct calibration should align the intensity image with the depth map. Results with our method show that

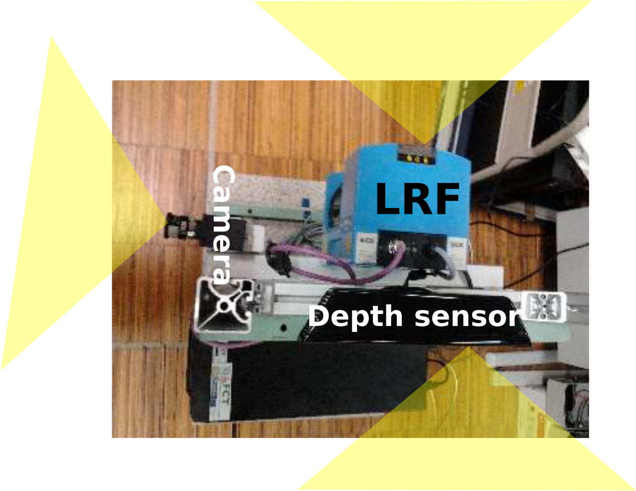


Figure 2.10: Example of a heterogeneous sensor network with non-overlapping FoV placed in a moving platform.

the misalignment is very slight, while for Herrera's method it is significant, and is larger when using distortion correction. This indicates that Herrera's method is not able to properly model the depth distortion with small datasets.

2.3 Calibration of multi-modal sensor arrangements with non-overlapping FoV

Many applications in robotics and intelligent transportation systems (ITS) require the use of multiple sensors that can be of the same modality (homogeneous sensor networks) [90, 93, 110] or of different modalities (heterogeneous or hybrid sensor setups) [15, 26, 61]. These setups must be calibrated both intrinsically and extrinsically. The literature in extrinsic calibration is vast and includes methods for finding the relative pose between sensors of different modalities [53, 135]. However, most of these solutions require the FoV to overlap and cannot cope with situations in which sensors are observing different, disjoint parts of the scene (see Figure 2.10).

Regarding homogeneous sensor setups, stationary camera networks are used in surveillance and object tracking [93], while multi-camera rigs allow for the coverage in vehicles of the whole surrounding environment [90]. In [6], Auvinet *et al.* describe a system that uses multiple active cameras for reconstructing the volumes of bodies in motion from the acquired depth maps. Its main application is in gait analysis which has become an increasingly interesting area of research. LightSpace [140] combines depth cameras and projectors to provide interactivity on and between surfaces in everyday environments, allowing a convincing simulation of the manipulation of physical objects. In [76], planar mirrors are employed for image-based camera localization and 3D scene reconstruction. The usage of mirrors relates with multi-camera systems since each mirror allows the camera to capture an extra view, working as another camera. The literature also reports heterogeneous sensor setups comprising both combinations of color cameras[1] and LRF, and combinations of color and depth cameras. Color camera and LRF networks have recently been used in object classification for the construction of maps of outdoor environments [26], by integrating visual and shape features. In [97], these features are combined for pedestrian detection in urban environments. With the intent of building a reliable and real-time multi-target tracking system, the fusion of laser and visual data is performed in [144]. In [61], multiple color and depth cameras are used for generating high-quality multi-view depth, allowing for the construction of 3D video.

For most of these applications the relative pose between sensor nodes must be known in advance in order for the acquired multi-modal information to be fused and the platform to work as a whole. There are several methods for performing both intrinsic and extrinsic calibration of color cameras, LRFs, and depth cameras. A brief overview of current approaches to calibrate either sensors of the same modality, or mixtures of two modalities is now provided. Table 2.3 summarizes the results of this overview clearly showing that the majority of existing solutions are unable to handle the problem of generic extrinsic calibration between sensors of different modalities with non-overlapping FoV.

1. *Color Camera Calibration:* The literature is vast but explicit methods using a known checkerboard pattern are specially popular because they are stable, accurate, and the calibration rig is easy to build. Bouguet's camera calibration toolbox [17] implements Zhang's method [146] that, given 3 or more images, estimates the intrinsic parameters, as well as the poses of the checkerboard with respect to the camera. These poses can be used to find the relative rigid displacement between different camera nodes (extrinsic calibration) by simply assuring that some planes are simultaneously observed across different nodes. However, there are camera networks for which the FoVs of the different nodes do not overlap. This happens either in surveillance, where many times a broad region must be covered with a small number of cameras [59, 93], or in

[1] The term *color camera* is used when referring to regular cameras, either RGB or grayscale, in order to better distinguish from depth cameras.

Table 2.3: Methods for calibrating multi-sensor systems.

		Color Cam.	LRF	Depth Cam.
Overlap	Bouguet [146]	X		
	Vasconcelos [135]	X	X	
	Herrera [53]	X		X
NonOverlap	Rodrigues [107]	X		
	Bok [15]	X	X	
	Our Contribution	X X	X 	 X

robotics, whenever cameras are placed to obtain an omni-directional view of the scene around the vehicle [90]. A possible solution in these cases is to use mirrors for computing the pose with respect to an object that is outside the FoV [54, 70, 107, 121]. The idea has been first used in [70] to calibrate a camera network with the pose of the object being estimated from a minimum of 5 mirror reflections. Sturm *et al.* [121] proved that such relative pose could be determined from a minimum of 3 images and Rodrigues *et al.* [107] introduced a minimal, closed-form solution for the problem that outperforms the methods suggested in [70, 121]. An exhaustive experimental evaluation showed that in practice 5 to 6 reflections are more than enough to obtain very stable and accurate results.

2. *Color Camera - LRF Calibration*: Zhang and Pless [145] proposed a practical method for the extrinsic calibration of a color camera and a LRF that uses at least 5 images of a known checkerboard pattern. Later on, Vasconcelos *et al.* [135] described a minimal solution for the problem leading to a robust algorithm that clearly outperforms the method in [145]. These solutions only deal with the overlapping case. More recently, an algorithm for calibrating a color camera and a LRF whenever their FoVs do not intersect was proposed [15]. The method makes assumptions about the relative pose between the checkerboard and the environment's structure that may be difficult to satisfy in small or cluttered spaces. Moreover, due to these assumptions, the sensor platform must move in order to acquire calibration data. In case of large platforms, such as ground vehicles, this method is not appropriate since it would be extremely difficult to acquire the required calibration data in different

positions and orientations. In this section, the method [135] is extended to the non-overlapping case, providing a simple solution that works for any color camera - LRF configuration that can be attached to either a small or a large platform.

3. *Color Camera - Depth Camera Calibration*: Scene reconstruction from a color - depth camera pair measurements requires the system to be calibrated, both intrinsically and extrinsically. Kinect cameras have a standard calibration from factory that is not accurate enough for many situations. As mentioned in Section 2.2, Herrera *et al.* [53] have recently modelled the Kinect's depth camera distortion and proposed a method for calibrating a depth camera and additional color cameras whose FoV overlap. Its main strength is in an explicit depth distortion term. Unfortunately it requires many images (over 20) and it is not prepared for handling non-overlapping situations. We tackle the first issue by making use of our new calibration method for color camera - depth camera pairs proposed in Section 2.2, that has proven to perform accurately with only 6 to 10 calibration images. In this section, this new approach is extended to the non-overlapping case, solving the calibration problem for any possible sensor configuration.

This section revisits the problem of the extrinsic calibration of multi-sensor arrangements that can comprise color cameras, LRF, and/or depth cameras. It builds on recent results for estimating the pose of an object observed by a color camera through planar mirror reflections [107] and proposes a systematic, practical approach for calibrating mixtures of color cameras, LRFs, and depth cameras with non-overlapping FoV. A thorough experimental assessment of the solution that enables to decide about the number of mirror reflections N and object views M or K that are needed to reach a certain accuracy level is presented, as well as an experiment of the calibration of the sensor platform in Figure 2.10 with an application in SfM that evinces the usefulness of heterogeneous sensor systems. The method uses a checkerboard pattern as calibration object and handles situations for which there is no simple, effective solution in the state-of-the-art.

2.3.1 Camera pose estimation from mirror reflections

It can be shown that the image acquired by a camera looking at a planar mirror is equivalent to the image that would be acquired by a virtual camera placed behind the mirror plane. In this case, the virtual and real cameras have the exact same

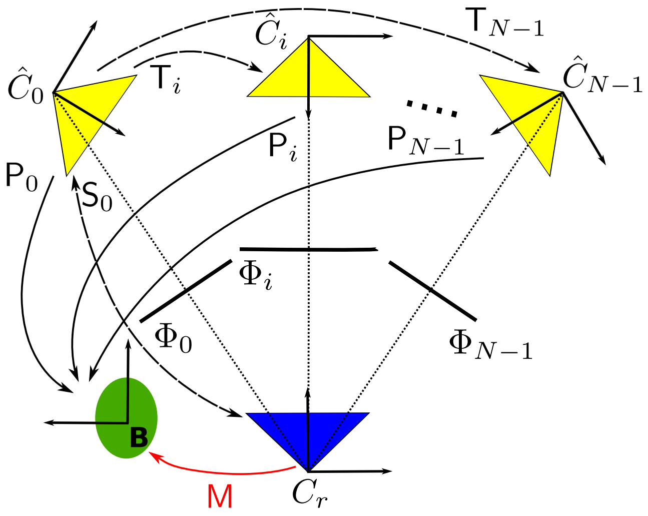


Figure 2.11: Object $\mathbf{B}$ is seen by camera C_r through N planar mirror reflections $\mathbf{\Phi}_i$, originating N virtual cameras $\widehat{C}_i$. Our goal is to find the pose M of the real camera C_r with respect to object $\mathbf{B}$.

intrinsic parameters and their local reference frames are related by a symmetry transformation S with respect to the mirror plane. Rodrigues *et al.* [107] propose to freely move a planar mirror in front of a camera in order to obtain images of an object that lies outside the FoV. It was shown that, given $N \geq 3$ images, it is possible to estimate the rigid displacement M between camera and object (Figure 2.11), as well as the plane coordinates of the N mirrors. Since their algorithm will be extensively used to accomplish the extrinsic calibration of sensors with non-overlapping FoV, this section overviews its steps.

2.3.1.1 Review of the algorithm presented in [107]

Figure 2.11 shows an object $\mathbf{B}$ being observed by camera C_r through N planar mirror reflections. Each virtual camera $\widehat{C}_i$ is originated by the mirror plane $\mathbf{\Phi}_i$, which is uniquely defined by its unitary normal vector $\vec{\mathbf{n}}_i$, and the scalar euclidean distance d_i, with $i = 0, \ldots, N-1$. The pose of the object P_i in each virtual camera reference frame is determined by either applying the *PnP* algorithm [46], in case $\mathbf{B}$ is a known 3D object, or by estimating and factorizing a planar homography [47], in case $\mathbf{B}$ is a plane surface. For the sake of convenience we always use a planar checkerboard pattern as calibration object. With abuse of notation, the pose and the homography will be denoted by the same symbol, whenever it is convenient.

Given the $N \geq 3$ object poses P_i, Rodrigues *et al.* [107] choose a reference virtual view and determine the position of the corresponding mirror plane. Let $\widehat{C}_0$ be the reference camera. The first step is to compute the relative pose T_i of the remaining virtual views, which can be easily accomplished by applying the following formula:

$$\mathsf{T}_i = \mathsf{P}_i^{-1}\mathsf{P}_0, i = 1, 2, \ldots, N-1. \tag{2.16}$$

It can be shown that each rigid motion T_i gives rise to two independent linear constraints on the parameters of the mirror plane $\mathbf{\Phi}_0$ that can be stacked for the $N-1$ motions, originating a system of linear equations. The least squares solution can be found by applying SVD, and $\mathbf{\Phi}_0$ is computed. The symmetry transformation S_0, that relates the reference frames of virtual camera $\widehat{C}_0$ and real camera C_r, is given by:

$$\mathsf{S}_0 = \begin{bmatrix} \mathsf{I} - 2\overrightarrow{\mathbf{n}}_0\overrightarrow{\mathbf{n}}_0^\mathsf{T} & 2d_0\overrightarrow{\mathbf{n}}_0 \\ \mathbf{0} & 1 \end{bmatrix}. \tag{2.17}$$

This symmetry matrix is involutory, meaning that $\mathsf{S}_0 = \mathsf{S}_0^{-1}$. From P_0 and S_0, the pose M of the object $\mathbf{B}$ comes in a straightforward manner as:

$$\mathsf{M} = \mathsf{S}_0\mathsf{P}_0. \tag{2.18}$$

Note that due to the mirror reflection, if the reference frame C_r is right-handed, then the reference frame $\widehat{C}_0$ is left-handed, and vice-versa, making the multiplication in Equation 2.18 to be defined this way. To improve robustness, this algorithm is performed N times independently, each time considering a different virtual camera as the reference frame. Then, the average of all estimations of M is considered.

A singular configuration occurs whenever all the mirror planes intersect into a single line in 3D. This can be caused by either rotating the mirror around a fixed-axis, or when the reflection planes are all parallel (the intersection line is at infinity).

2.3.1.2 Calibration of cameras with non-overlapping FoV

As explained in [107], the fact that it is now possible to determine the pose of an object M that is outside the camera's FoV enables the extrinsic calibration of cameras that do not observe overlapping regions of the scene.

Consider the situation of Figure 2.12 where cameras C_F and C_B, mounted on a platform, observe object $\mathbf{B}$ directly and through mirror reflections, respectively. The extrinsic calibration D can be carried by computing M_F and M_B, and then finding $\mathsf{D} = \mathsf{M}_F^{-1}\mathsf{M}_B$. M_F can be computed as a planar homography using a standard

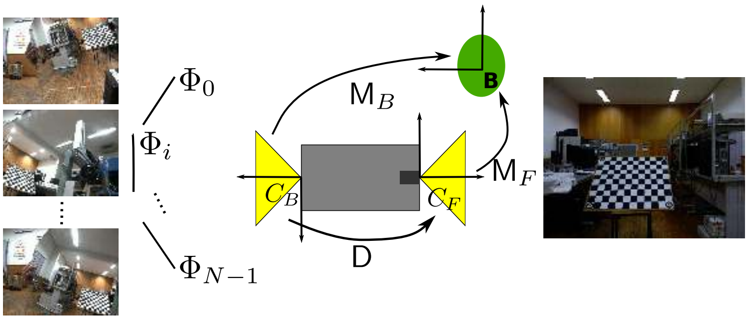
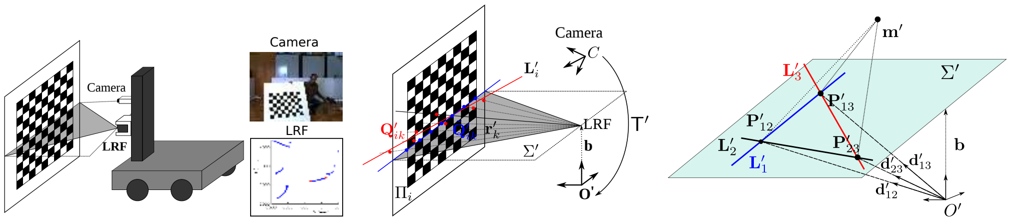


Figure 2.12: The extrinsic calibration D is accomplished by showing the object directly to camera C_F (checkerboard image on the right) and showing it through mirror reflections to camera C_B (checkerboard images on the left).



(a) Camera and LRF mounted on a platform

(b) Relative pose T' between camera and LRF

(c) Line intersections in the scan plane Σ'

Figure 2.13: (a) The extrinsic calibration in [135] is carried by moving a checkerboard pattern in front of both color camera and LRF. (b) The color camera C that observes plane $\mathbf{\Pi}_i$ and the LRF O' that sees line $\mathbf{L}'_i$ are related by a transformation T'. (c) Lines $\mathbf{L}'_i$ lie in the scan plane $\mathbf{\Sigma}'$ and intersect in points $\mathbf{P}'_{ij}$, which define the directions $\mathbf{d}'_{ij}$ with the origin of the LRF reference frame O'.

approach [146], while estimating M_B is performed using the algorithm from [107].

In practical terms, according to [107], for $N = 6$ mirror views it is possible to achieve subpixel accuracy, corresponding to a rotation error slightly below 1° and a translation error of approximately 3.5%.

2.3.2 Extrinsic calibration of a color camera and a LRF

Let us now consider the problem of finding the extrinsic calibration T' between a color camera C and a LRF O' as illustrated in Figure 2.13.

Vasconcelos *et al.* [135] have recently shown that a color camera and a LRF can be calibrated from a minimum of $M = 3$ images of a planar grid. The problem of finding T' is cast as the problem of registering a set of planes $\mathbf{\Pi}_i, i = 1, 2, 3$, expressed in color camera coordinates, with a set of 3D lines $\mathbf{L}'_i, i = 1, 2, 3$ in the

LRF coordinate system. They show that there are 8 solutions, with the correct one being selected by an additional plane-line correspondence.

We start by reviewing the algorithm [135] and then extend the method to the case of non-overlapping FoV by using mirror reflections.

2.3.2.1 Review of the algorithm presented in [135]

Consider M planes $\mathbf{\Pi}_i \sim [\mathbf{n}_i^\mathsf{T} \quad 1]^\mathsf{T}, i = 1, 2, \dots, M$ expressed in color camera coordinates and the corresponding lines $\mathbf{L}'_i$, expressed in LRF coordinates, where $\mathbf{L}'_i$ is the locus of intersection between $\mathbf{\Pi}_i$ and the scan plane $\mathbf{\Sigma}'$, as shown in Figure 2.13b.

Vasconcelos *et al.* [135] show that the relative rotation R' can be determined by solving the system of non-linear equations

$$\begin{cases} \alpha_{12}\mathbf{d}_{12} = \mathsf{R}'^\mathsf{T}(\mathbf{P}'_{12} + \mathbf{m}') \\ \alpha_{13}\mathbf{d}_{13} = \mathsf{R}'^\mathsf{T}(\mathbf{P}'_{13} + \mathbf{m}') \\ \alpha_{23}\mathbf{d}_{23} = \mathsf{R}'^\mathsf{T}(\mathbf{P}'_{23} + \mathbf{m}') \end{cases} , \tag{2.19}$$

where $\mathbf{d}_{ij}$ is the direction of the line where planes $\mathbf{\Pi}_i$ and $\mathbf{\Pi}_j$ intersect, $\mathbf{P}'_{ij}$ is the point in plane $\mathbf{\Sigma}'$ where lines $\mathbf{L}'_i$ and $\mathbf{L}'_j$ meet, $\mathbf{m}'$ is an unknown vector and α_{ij} are unknown scalars that assure algebraic equality. The authors observed that the system of equations 2.19 corresponds to solving the *P3P* problem [46] for determining the relative pose between a color camera and an object from 3 object-image point correspondences. Figure 2.13c shows the nature of this *P3P* problem, where the virtual perspective camera is centered in point $\mathbf{m}'$ where planes $\mathbf{\Pi}'_i$ intersect, $\mathbf{d}_{ij}$ play the role of image points and $\mathbf{P}'_{ij}$ of object points. *P3P* enables to find the relative orientations R'^T and the position $\mathbf{m}'$ of the intersection of the 3 planes in LRF coordinates.

Finally, to find the translation $\mathbf{t}'$, it is shown in [135] that it suffices to compute $\mathbf{t}' = \mathsf{A}^{-1}\mathbf{c}$ with A and $\mathbf{c}$ defined as in Equation 2.6 after substituting $\mathring{\mathbf{n}}_i$ by $\mathbf{n}'_i$, for $i = 1, 2, 3$, where $\mathbf{n}'_i$ refers to the normal to plane $\mathbf{\Pi}'_i$, expressed in LRF coordinates.

A discussion about the singular configurations of this method is given in [135]. In general terms, whenever the lines where the checkerboard planes intersect are parallel, or the checkerboard planes intersect in a point that lies in the danger cylinder (refer to [135]) a singular configuration occurs.

The registration procedure originates a total of $R \leq 8$ rigid transformations T'_i that align 3 planes with 3 coplanar lines. For sets with $M > 3$ plane-line correspondences, the best solution is chosen in a hypothesize-and-test framework to select

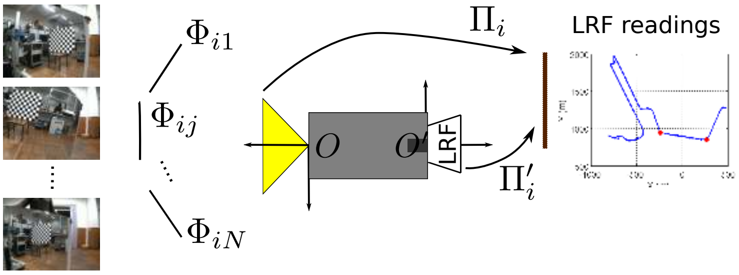


Figure 2.14: Calibration of a color camera - LRF setup in case of non-overlapping FoV. It is shown how to determine the pose $\mathbf{\Pi}_i$ of the checkerboard in color camera coordinates from N mirror reflections $\mathbf{\Phi}_{ij}$. The line segment between the red dots in the LRF readings plot corresponds to the calibration plane.

inliers, which are then used in a bundle adjustment step to refine the solution. This is done by simultaneously minimizing the reprojection error and the distance to LRF depth measurements. According to [135], a total of $M = 5$ checkerboard planes are sufficient for achieving good accuracy.

2.3.2.2 Calibration of a color camera - LRF pair with non-overlapping FoV

As shown in the scheme of Figure 2.14, the checkerboard pattern is placed in front of the LRF and, for each pose $\mathbf{\Pi}'_i, i = 1, \ldots, M$, the color camera observes the pattern through N mirror reflections $\mathbf{\Phi}_{ij}, j = 1, \ldots, N$. The coordinates of the checkerboard plane $\mathbf{\Pi}_i$ in the color camera reference frame can be found using the method described in Section 2.3.1. By applying this algorithm for retrieving each plane pose, the extrinsic calibration between the color camera and the LRF can be directly obtained by using the method from [135]. Note that, in order to be possible to perform such a calibration, a minimum of 9 images are necessary, since the algorithm presented in [107] (reviewed in Section 2.3.1) requires $N \geq 3$ mirror reflections and the algorithm from [135] (reviewed in Section 2.3.2.1) requires $P \geq 3$ plane poses. However, and as discussed before, in practice more images are required to obtain robust, accurate results.

The first step for performing the extrinsic calibration is to find an initial estimation. In this case, it is done similarly to the method from [135], having the difference that planes $\mathbf{\Pi}_i$ are determined from the algorithm described in Section 2.3.1 and not directly from plane-to-image homographies. A refinement step is then applied to minimize the reprojection errors in the color camera and LRF. The minimization is performed over the parameters T', inlier planes $\mathbf{\Pi}_i$ and corresponding mirror poses $\mathbf{\Phi}_{ij}$, and the color camera's intrinsic parameters K,

$$\min_{\mathsf{T}',\mathbf{\Pi}_i,\mathbf{\Phi}_{ij},\mathsf{K}} e = e_{LRF} + ke_{CAM}, \tag{2.20}$$

where k is a weighting parameter. The LRF residue e_{LRF} is the sum of the squared distances between the points $\tilde{\mathbf{Q}}'_{ik}$, obtained by mapping planes $\mathbf{\Pi}_i$ into the LRF reference frame and intersecting with the directions $\mathbf{r}'_k$ in the scan plane $\mathbf{\Sigma}'$, and the points $\mathbf{Q}'_{ik}$ that are reconstructed from the depth readings (refer to Figure 2.13b):

$$e_{LRF} = \sum_i \sum_j ||\mathbf{Q}'_{ik} - \tilde{\mathbf{Q}}'_{ik}||^2. \tag{2.21}$$

In the present case of a non-overlapping configuration, since the checkerboard images result from mirror reflections, the reprojection error of the color camera is computed differently, when compared to the algorithm proposed in [135]. The virtual camera relative to the mirror pose $\mathbf{\Phi}_{ij}$ is computed by using transformation S_0 in Figure 2.11, which is a reflection about a plane. Since the virtual cameras "observe" the checkerboard directly, they are used for computing the reprojection errors of the plane-to-image homographies, yielding e_{CAM}. Note that the computation of the color camera residue depends on the intrinsic parameters K, so that this formulation allows to refine both the intrinsic and the extrinsic calibrations. The main steps of the proposed method are outlined in Algorithm 1.

2.3.2.3 Experimental results for non-overlapping FoV

The method proposed by Vasconcelos *et al.* [135] is able to achieve robust, accurate results for the LRF-color camera pair calibration in an overlapping configuration from $M \geq 6$ checkerboard images. In the case of non-overlapping FoV, besides the rigid displacement T' between the sensors, the pose of the checkerboard $\mathbf{\Pi}$, which is observed through mirror reflections, in color camera coordinates, must be estimated. Rodrigues *et al.* [107] show that in practice $N \geq 6$ virtual views are required for the estimation to be reasonably accurate.

In this section a real experiment with ground truth is carried, not only to validate the approach presented in Section 2.3.2.2, but also to assess the number of $M \times N$ images that are necessary in practice to obtain a certain degree of accuracy.

A setup that has overlapping FoV was used in order to enable calibration with the algorithm of Section 2.3.2.1 that works as ground truth. A dataset of $M = 12$

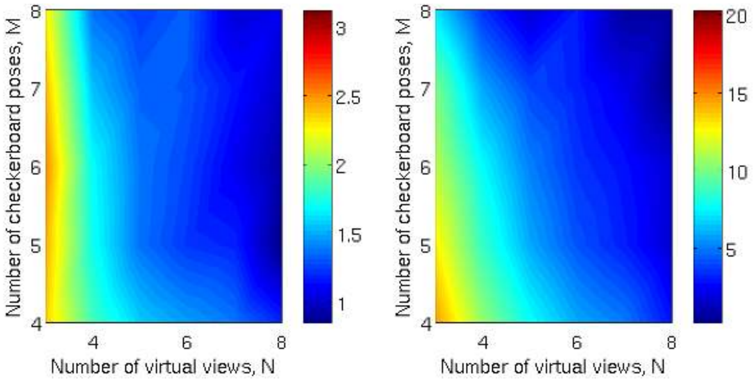
Algorithm 1: New method for the calibration of a color camera - LRF pair with non-overlapping FoV

```
Inputs: Scan plane $\mathbf{\Sigma}'$, lines $\mathbf{L}'_i$ and object poses in relation to the virtual
  cameras $\mathsf{P}_{ij}, i = 1, \dots, M, j = 0, \dots, N-1$
Output: Extrinsic calibration $\mathsf{T}'$, mirror poses $\mathbf{\Phi}_{ij}$, and planes $\mathbf{\Pi}_i$

  1. Compute each plane pose $\mathbf{\Pi}_i$ using the method from [107] (reviewed in
     Section 2.3.1).
  2. Use the algorithm presented in [135] (reviewed in Section 2.3.2.1) for
     initializing the extrinsic calibration $\mathsf{T}'$ from the obtained planes $\mathbf{\Pi}_i$ and
     the input lines $\mathbf{L}'_i$.
  3. Refine the estimated transformation $\mathsf{T}'$, planes $\mathbf{\Pi}_i$, mirror poses $\mathbf{\Phi}_{ij}$, and
     color camera intrinsics $\mathsf{K}$ in a bundle adjustment step as defined in
     Equation 2.20.
```

direct image-LRF cuts was collected for calibrating the sensor pair. Without moving the color camera with respect to the LRF, we collected a second dataset where the checkerboard is observed by the color camera through mirror reflections in order to mimic the non-overlapping situation. A total of $M = 12$ checkerboard poses with each pose being observed by $N = 12$ mirror reflections was acquired.

In order to find the number of checkerboard poses M and mirror reflections N required to obtain accurate results, $M = 4, \dots, 8$ and $N = 3, \dots, 8$ were considered, and for each of the 30 possible combinations of M and N, 50 calibrations sets of M checkerboard poses and N corresponding image reflections were randomly selected. The average rotation and translation errors with respect to the ground truth obtained with these calibration sets are shown in Figure 2.15. It can be seen that using $M < 6$ checkerboard poses or $N < 6$ mirror reflections frequently leads to translation errors over 4%. This can be explained by the fact that, in average, not even the original methods perform more accurately with less images. Note that, as observed in experiments performed with the original methods, the rotation error is always relatively low (below 2.4°). Moreover, using the minimum number of mirror reflections ($N = 3$) provides poor results, as reported in [107]. However, the method was able to achieve very accurate results, with translation errors of approximately 1% and rotation errors up to 1° with datasets of 7 or more checkerboard poses and mirror reflections. A good compromise would be to use a dataset of 36 images, consisting of $M = 6$ checkerboard poses and $N = 6$ mirror reflections, as the obtained average



(a) Rotation error (°)

(b) Translation error (%)

Figure 2.15: Extrinsic calibration errors obtained with the LRF-color camera pair in a non-overlapping configuration, for varying number of checkerboard poses and virtual views.

errors are of about 3.5% in translation and 1.26° in rotation. Thus, in overall terms, the results are satisfactory and prove that, for carefully chosen checkerboard and mirror orientations, the extrinsic color camera - LRF calibration can be accurately achieved using small datasets.

2.3.3 Extrinsic calibration of a color camera and a depth camera

In this section the calibration of a color camera - depth camera pair when their FoVs do not overlap is considered. A similar approach to the one proposed in section 2.3.2 is considered: using mirror reflections for enabling the color camera to observe a calibration target placed in the FoV of the depth camera.

As described in Section 2.2, Herrera *et al.* [53] recently proposed a method for solving this problem in the case of overlapping FoV, using as depth camera the Kinect. They showed that jointly calibrating both sensors improved accuracy, as opposed to a separate calibration, and proposed a new depth distortion model for the depth camera. Unfortunately, their method requires $K > 20$ images of a checkerboard to provide good results. The straightforward extension to the non-overlapping case would lead to the need of collecting at least 120 images because of the mirror reflections. Thus, we consider the modified version of Herrera's method proposed in Section 2.2, which is able to achieve comparably accurate calibration results using only about 6 images, favouring calibration both in overlapping and non-overlapping situations. This section presents the extension of this new method to the non-

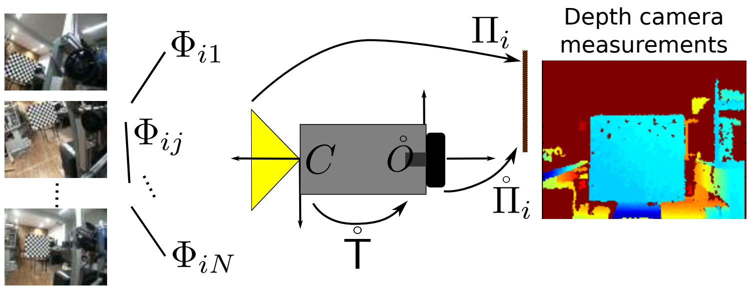


Figure 2.16: Calibration of a color-depth camera pair in a non-overlapping configuration, similar to the color camera - LRF situation of Figure 2.14.

overlapping case.

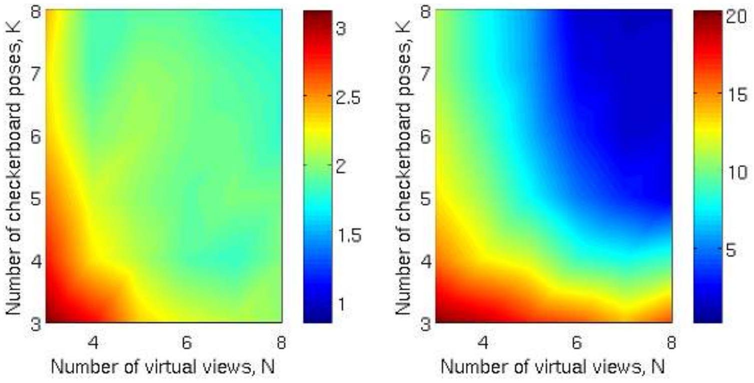
2.3.3.1 Calibration in the case of non-overlapping FoV

Calibrating a depth and a color camera with non-overlapping FoVs is done in an analogous manner as for the LRF case (Section 2.3.2.2). Figure 2.16 shows that the checkerboard pattern is moved in front of the depth camera and, for each plane pose $\mathbf{\Pi}_i$, the color camera observes the pattern through N mirror reflections $\mathbf{\Phi}_{ij}$. Again, the checkerboard poses with respect to the color camera are computed using the algorithm reviewed in Section 2.3.1, and the extrinsic calibration between the color camera and the depth camera is carried using the new method described in the previous section. However, and as previously explained in Section 2.3.2.2, since mirror reflections are being used for estimating the checkerboard poses $\mathbf{\Pi}_i$, the first term of the error function in Equation 2.12, which corresponds to the color camera reprojection error, must be changed. The virtual camera corresponding to mirror pose $\mathbf{\Phi}_{ij}$ is computed using transformation S_0 in Figure 2.11, and used for finding the reprojected pixel positions $\mathbf{x}_c$ that appear in Equation 2.12. Moreover, since the mirror positions $\mathbf{\Phi}_{ij}$ are being taken into account, these are also refined, along with $\mathcal{I}, \mathring{\mathcal{I}}, \mathring{\mathsf{T}}$ and $\mathbf{\Pi}_i$.

For a non-overlapping configuration, the extrinsic calibration of a color camera and a depth camera requires a minimum of 9 images since the algorithm in Section 2.3.1 needs $N \geq 3$ mirror reflections and the present method requires $K \geq 3$ checkerboard poses.

2.3.3.2 Experimental results for non-overlapping FoV

The method presented in Section 2.2, that works with sensors whose FoVs overlap, reported accurate results for datasets of 6 image-disparity maps pairs. As in Section



(a) Rotation error (°) (b) Translation error (%)

Figure 2.17: Extrinsic calibration errors obtained with the color camera - depth camera pair in a non-overlapping configuration, for varying number of checkerboard poses and virtual views.

2.3.2.3, the purpose of this experiment is both to validate the approach and to assess the number of checkerboard poses and mirror reflections required to produce a certain accuracy, when working with a non-overlapping configuration. We performed similarly as in Section 2.3.2.3, concerning the sensor setup and dataset acquisition. Our ground truth is the result of the calibration performed with the method of Section 2.2, with our dataset.

A total of 50 calibration sets were randomly selected for each combination of $K = 3, \ldots, 8$ checkerboard poses and $N = 3, \ldots, 8$ mirror reflections. These sets were used as input to the method from Section 2.3.3.1 and the average rotation and translation errors are presented in Figure 2.17. The overall conclusions are the same as in Section 2.3.2.3. For datasets using less than 6 checkerboard and mirror poses, the translation errors tend to be higher than 4% and the rotation errors slightly over 2°. Increasing the number of acquired images to 36 ($K = 6$ and $N = 6$) leads to average errors of 3.6% and 1.9° in translation and rotation, respectively. This is an acceptable result for many applications. However, when higher accuracy is required, increasing the number of checkerboard poses and the number of mirror reflections up to 8 originates results with errors smaller than 1.5% in translation and 1.6° in rotation.

Remark that in general, the errors obtained with the color camera - depth camera pair are slightly larger than the ones obtained with the color camera - LRF pair. This can be explained by the fact that in the first there are more parameters to be optimized, requiring more images to achieve the same accuracy. However, this

Table 2.4: Angular (e_α) and distance (e_d) errors between reconstructed lines and planes observed from 5 different views. Angular errors between the floor plane and the wall planes reconstructed by the depth camera (e_{g1}) and the LRF (e_{g2}).

View	1	2	3	4	5
$e_\alpha(°)$	0.11	0.33	0.34	0.16	0.47
$e_d(\%)$	0.97	1.72	1.24	1.33	1.50
$e_{g1}(°)$	0.1967	0.4512	0.2116	0.3769	0.1615
$e_{g2}(°)$	0.1951	0.4538	0.7642	0.4201	0.2053

correspondences between different views are known, the minimal solution proposed in [98] was applied for computing the platform motion. In this case there are only two correspondences of non-parallel planes (the floor and the walls), so it is necessary to extract one point correspondence for computing all 6 degrees-of-freedom. The extracted point correspondences are shown in Figure 2.19 with identifying colors. This example is particularly interesting because the complete lack of texture in the wall planes hampers the reconstruction using RGB cameras, and thus sensors that provide depth measurements are required. Figure 2.19 shows the segmented regions corresponding to each plane (top row), and the obtained 3D model after concatenating the individual reconstructions using the estimated platform motion (bottom row). Qualitatively, by observing the alignment between the individual 3D models, particularly in the area surrounding the door entrance, it can be seen that the registration was well performed. The green ellipse corresponds to the area of misalignment between planes, which, as can be seen, is very slight. This indicates that the platform motion estimation is accurate, which could not have been possible if the surface planes or the extrinsic calibration had been poorly recovered. In quantitative terms, this reconstruction was assessed by computing the angular errors between the floor plane and the reconstructed wall planes, shown in the last two rows of Table 2.4. Errors below 0.8° were always achieved, being another indicative that the reconstruction, and thus the extrinsic calibration, is accurate. Note that each plane was recovered using one of the sensors independently and only a good extrinsic calibration would provide small errors. The results confirm that this kind of sensor setup can indeed be used for obtaining accurate reconstructions of the scene, even in situations of lack of texture.

In general, the good results obtained prove that the proposed method is practical, effective and useful, solving a problem that so far did not have a simple solution.

2.4 Conclusions

In this chapter we present a novel minimal, optimal solution for registering 3D planes across different reference frames by formulating the registration problem in the dual projective space. It is integrated in a new fast and accurate method for the intrinsic and extrinsic calibration of a Kinect sensor that achieves high accuracy using only 6 to 10 image-disparity pairs of a planar checkerboard pattern. We build on the recent work of Herrera *et al.* [53] that uses a large number of input frames and multiple iterative minimization steps for obtaining very accurate calibration results. Besides using the new solution for 3D plane registration, we propose other modifications to this estimation pipeline that dramatically improve stability, usability, and runtime: (i) including a metric constraint during the iterative refinement to avoid a drift in the disparity to depth conversion; and (ii) estimating the parameters of the depth distortion model in an open-loop post-processing step. Comparative experiments show that our pipeline can achieve a calibration accuracy similar to [53] while using less than 1/6 of the input frames and running in 1/30 of the time.

Finally, this chapter builds on recent results in object pose estimation using mirror reflections to provide an accurate and practical solution for the extrinsic calibration of mixtures of color cameras, LRFs, and depth cameras with non-overlapping FoV. The method is able to calibrate any possible sensor combination as far as the setup includes at least one color camera. The technique is tested in challenging situations not covered by the current state-of-the-art, proving to be practical and effective.

Chapter 3

SfM using Plane Primitives in RGB-D and Passive Stereo

Although multi-view stereo and odometry from RGB-D sequences have been intensive fields of research in the last few decades, current methods still have difficulty in handling situations of weak or repetitive texture, variable illumination, non-lambertian reflection, and high surface slant [40]. In this context, it makes sense to explore the fact that man-made environments are usually dominated by large plane surfaces to improve the accuracy and robustness of 3D reconstruction. This is the key idea behind the so-called Piecewise-Planar Reconstruction (PPR) methods that use the strong planarity assumption as a prior to overcome the above mentioned issues [3, 37, 38, 40, 81, 114, 139]. In addition, piecewise-planar 3D models are perceptually pleasing and geometrically simple, and thus their rendering, storage, and transmission is substantially less complex when compared to conventional point-cloud models [2, 116].

Although the aforementioned works are mainly multi-view stereo methods, the usefulness of plane primitives is not limited to multi-view stereo reconstruction as shown by recent works in SLAM for RGB-D cameras that estimate the motion from plane correspondences [123]. Taguchi *et al.* highlight that plane features are much less numerous than point features, favouring fast correspondence and scalability, and that the global character of plane-primitives helps avoiding local minima issues [123]. Also, man-made environments are often dominated by large size planes that enable correspondence across wide baseline images and, since plane-primitives are mostly in the static background, the motion estimation is specially resilient to dynamic foreground.

This chapter describes two pipelines that combine the benefits of PPR and plane-based odometry by recovering both structure and motion from plane-primitives. One algorithm receives as input an image sequence acquired by a calibrated RGB-D camera (Section 3.2), while the other works with a calibrated passive stereo rig (Section 3.3), and both output the camera motion and the 3D planes in the scene. Assuming that we have a sensor, that can be either an RGB-D or a stereo camera, and that we can detect planes, in Section 3.1 we describe how to associate and compute the relative camera motion from these planes.

3.1 Relative motion from plane primitives

Consider two consecutive cameras C_i and C_{i+1} and two sets of plane detections. Let $\mathbf{\Pi}_k^{(i)} \sim [\mathbf{n}_k^\mathsf{T} \quad 1]^\mathsf{T}$ and $\mathbf{\Pi}_k^{(i+1)} \sim [\mathring{\mathbf{n}}_k^\mathsf{T} \quad 1]^\mathsf{T}$, with $k = 1 \ldots K$ be putative plane correspondences across the two pairs. Our objective is to use these plane correspondences to estimate the relative pose $(\mathsf{R}_i, \mathbf{t}_i)$ between the two cameras. In [45], it was first shown that two sets of 3D planes can be registered in a closed-form manner from a minimum of 3 correspondences as long as their normals span the entire 3D space. More recently, Taguchi *et al.* [123] used this registration algorithm as a starting point for their plane-based SLAM method for RGB-D cameras. They studied the singular configurations and showed how to use reconstructed 3D points to disambiguate the motion whenever the information provided by planes was insufficient. We revisit this registration problem and show how to disambiguate the motion by directly using image point correspondences, in order to avoid having to reconstruct points and introduce extra sources of noise.

3.1.1 Relative pose from 3 plane correspondences

Whenever there are 3 plane correspondences that span the entire 3D space, the relative pose between cameras i and $i+1$, $\mathsf{R}_i, \mathbf{t}_i$, is estimated using the minimal, optimal solution proposed in Section 2.1, where R_i and $\mathbf{t}_i$ are represented by $\mathring{\mathsf{R}}$ and $\mathring{\mathbf{t}}$, respectively. In this case, N_i in Equation 2.6 has rank 3 and the translation is fully determined.

3.1.2 Relative pose estimation in case N_i has rank 2

The matrix of the normal vectors N_i can have rank 2 whenever there are only two corresponding planes available or the three planes have a configuration such that their normals are coplanar. An example of this situation happens when at least two planes are parallel. The rotation R_i is estimated using Horn's algorithm [55] since two corresponding planes suffice. However, there is a 2D space for the translation, and thus there is one remaining Degree-of-Freedom (DoF) to be estimated. Given an image point correspondence $\mathbf{x}^{(i)}, \mathbf{x}^{(i+1)}$ between the two cameras C_i and C_{i+1}, the translation $\mathbf{t}_i$ can be fully determined by stacking the epipolar constraint ${\mathbf{x}^{(i+1)}}^{\mathsf{T}} \mathsf{E}_i \mathbf{x}^{(i)} = 0$, where $\mathsf{E}_i = [\mathbf{t}_i]_\times \mathsf{R}_i$ is the essential matrix, to the two linear constraints in Equation 2.6.

3.1.3 Relative pose estimation in case N_i has rank 1

Whenever there is a single plane correspondence or the putative plane correspondences are all parallel, the registration leads to the computation of 2 DoF for the rotation. In this case N_i has rank 1, and thus 1 DoF for the translation can be estimated. We show for the first time that in this case the relative pose can be determined from a minimum of 3 additional image point correspondences $\mathbf{x}_k^{(i)}, \mathbf{x}_k^{(i+1)}, k = 1 \ldots 3$. Related to this problem is the work described in [34], where a minimal solution for the case of two known orientation angles is given. Our problem differs from it because we have an extra constraint for the translation.

Our reasoning is explained in the 3D space instead of the dual space as in Section 2.1. Both cameras C_i and C_{i+1} are independently rotated so that their z axes are aligned with the plane normal, through transformations P_i and P_{i+1}. This implies that the rotated cameras become related by an unknown rotation around the z axis, $\mathsf{R}_u(\theta)$, and a translation $\mathbf{t}_u = [t_x \quad t_y \quad t_z]^\mathsf{T}$, where t_z can be computed as follows. In the rotated configuration, Equation 2.2 becomes

$$\begin{bmatrix} 0 \\ 0 \\ z_2 \\ 1 \end{bmatrix} \sim \begin{bmatrix} \mathsf{R}_u & \mathbf{0} \\ -[t_x \quad t_y \quad t_z]\mathsf{R}_u & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ z_1 \\ 1 \end{bmatrix}. \tag{3.1}$$

Thus, t_z can be determined by $t_z = -\frac{z_1/z_2 - 1}{z_1}$. The remaining 3 DoF ($\theta$, t_x and t_y) can then be determined from 3 point correspondences using the epipolar constraint. The essential matrix E_i has a simplified form as in [34], allowing the epipolar constraint to be written as $\mathsf{W}[t_x \quad t_y \quad 1]^\mathsf{T} = \mathbf{0}$, where the 3×3-matrix W depends on θ, which can be computed using the hidden variable method. This originates up to 4 solutions for the motion in the rotated configuration, T_u. The real motion T_i can then be

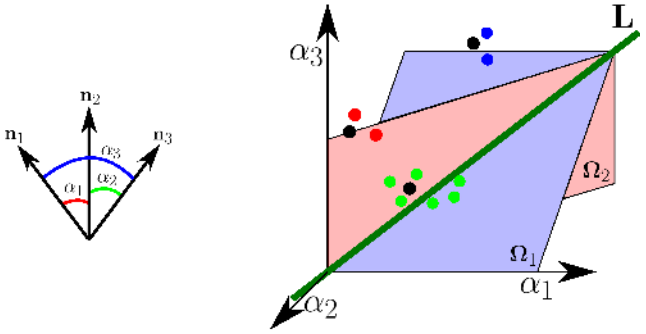


Figure 3.1: A descriptor is computed for the plane triplets and used in a nearest-neighbours approach for finding putative matches between the planes. Similarities between angles in the descriptor give rise to different hypotheses, depicted by the points near planes $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ and line L.

retrieved by simply computing $\mathsf{T}_i = \mathsf{P}_{i+1}^{-1}\mathsf{T}_u\mathsf{P}_i$.

3.1.4 Robust algorithm for computing the relative pose

Our relative pose estimation algorithm uses an hierarchical RANSAC scheme that works by considering the maximum number of planes present in the image pair, and only using point correspondences when strictly necessary. It first attempts to compute the pose from 3 plane correspondences, using subsequently less plane correspondences in case of failure, meaning that it tries to carry the registration with 2 planes and 1 point, and if this fails, with 1 plane and 3 points.

The method starts by building a descriptor (refer to Figure 3.1)) for matching triplets of planes, which consists of the 3 angles between the plane normals sorted by increasing value, in both cameras. Putative matches are established using a nearest neighbours approach. Remark that the descriptor implicitly establishes plane correspondences between elements in the triplet and that typically there is a relatively small number of triplets for each view. In case the angles in the descriptor are sufficiently different from each other, the descriptor establishes plane correspondences directly. However, if two of the angles are similar, two possible sets of element-wise correspondences are considered. This is the case in Figure 3.1 where the point in the descriptor space is close to plane $\mathbf{\Omega}_2$ that defines $\alpha_1 = \alpha_2$ (and identical for plane $\mathbf{\Omega}_1$ that defines $\alpha_2 = \alpha_3$). Similarly, if all three angles are close, six possible hypotheses for matches must be considered. This is the case when the point is close to the line L that defines $\alpha_1 = \alpha_2 = \alpha_3$.

For each triple correspondence, a solution is computed using the procedure in subsection 3.1.1. For each model, patches in camera C_i are projected to camera C_{i+1} using the homography induced by the plane containing the patch. Patches with photo-geometric error below a pre-defined threshold are considered as inliers, and used for computing a score ϵ.

The pose estimation is performed in a RANSAC framework. If there are no matching triplets of planes or the number of inliers for the computed solutions originates too low scores, the algorithm attempts to use 2 plane correspondences. A descriptor consisting of the angle between the 2 plane normals is considered for both cameras and matches are established using a nearest-neighbours approach. Since there is only one angle, each match gives rise to two hypotheses. A local feature detector (Speeded Up Robust Features (SURF) [11]) is used for extracting point features and solutions are computed in a RANSAC framework from two planes and one point correspondences (subsection 3.1.2). The models' inliers are computed as in the previous stage. Similarly, if there are no acceptable corresponding pairs of planes, the motion is estimated using one plane and three point-correspondences, as described in subsection 3.1.3. Note that in theory the scoring metric might fail if the planes surfaces lack texture. An hybrid score metric that mixes planes and points raises other type of issues, such as normalization. The metric used in this work always provided acceptable results, and thus it was kept unaltered.

3.2 SfM in case of RGB-D cameras

Visual odometry is the process of estimating the motion of a robot using the input of a single or multiple cameras attached to it. It has important applications in robotics, for control and navigation in the absence of an external reference system. Typical visual odometry systems can be split into 3 steps: (1) feature tracking/matching between images; (2) estimation of the camera motion inside a random-sample based procedure for robustness against outlier matches, and (3) optimization using bundle-adjustment for refining the camera poses. Research has been made in order to tackle this problem using RGB cameras [56, 66]. However, these methods face significant challenges including the reconstruction of textureless regions. RGB-D sensors, such as the Microsoft Kinect and the Asus Xtion Pro Live, cope with this issue since they provide the 3D geometry and the visual appearance of the scene simultaneously. Odometry systems that operate with such information are, therefore, different from the monocular systems, since depth information can be explored for providing reliable camera poses and 3D reconstructions.

Several researchers have focused on the problem of odometry and SLAM for RGB-D sensors [29, 51, 63, 118, 124]. Endres *et al.* [29] proposed a two-fold SLAM system for RGB-D sensors. On the front-end, the spatial relation between adjacent RGB-D images is established by extracting and matching image features. The matches are then used to estimate the relative transformation between sensor poses using a RANSAC-based procedure. The back-end of the SLAM system optimizes the pose observations with a graph-optimization procedure to keep long-term reliable reconstructions. A similar work to [29] was presented by Henry *et al.* [51]. Their approach uses sparse feature matches to compute an initial pose estimate using RANSAC, which is refined using an Iterative Closest Point (ICP) procedure.

Steinbruecker *et al.* [118] proposed a photo-consistency approach that aims to find the best transformation between two sequential RGB-D frames. For robustness against large image displacements, the optimization is carried from a coarse-to-fine image resolution. This approach was then generalized by Kerl *et al.* [63] by including a probabilistic derivation and by showing how motion priors can be used to further improve the performance of [118]. Kerl *et al.* also study the performance of different outlier weighting functions, concluding that weighting outlier pixels with t-distribution in conjunction with motion priors leads to the best performance.

Most of these methods run in real-time and provide accurate estimations for high frame rate acquisitions and moderate sensor velocity. However, they are not able to properly cope with large displacements between consecutive frames. In [118], it has been experimentally shown that the performance of the method degrades as the frame interval increases, which is equivalent to decreasing the frame rate of the acquisition, or increasing the sensor velocity.

In this section, we propose a new odometry method which uses both depth and color information for extracting planes from the scene, and the relative pose estimation between consecutive frames is cast as a plane registration problem. In the absence of the minimum number of required planes, 2D point correspondences are extracted for finding the remaining DoF. This procedure allows the method to cope with large baselines, since it only requires that there exists a few plane and/or point correspondences.

The algorithm uses the hierarchical scheme described in Section 3.1 for estimating the camera motion. As a final step, we perform the refinement of the initial estimation by minimizing the photometric error. The algorithm estimates the entire motion of the sensor using only the information acquired from pairs of consecutive frames, and no prior knowledge is considered. This is an important difference in our method since all the motion estimation is performed pairwise, whereas in the state-of-the-art methods temporal information is often explored to enforce smoothness in the trajectories.

Closely related with this work is the paper by Taguchi *et al.* [124] that, to the best of our knowledge, is the first work that proposes plane-based SLAM for RGB-D sensors. It uses both points and planes as primitives, and the registration of 3D data in different coordinate systems provides the relative pose estimation. Although our method also relies on planes and points for achieving the pose estimation, registration is performed pairwise and not in relation to a global map. Moreover, we only use points if strictly necessary, as opposed to Taguchi's method. Also, our points are not reconstructed, being more robust to measurement errors. Another key difference is the refinement step, where in [124] a bundle-adjustment procedure to minimize error between points (and between points and planes) is performed, whereas in our method photo-consistency is used.

We validated our approach on three image sequences from dataset [120], as well as on a sequence acquired at high resolution by our Kinect device. For all the sequences, the performance of our approach was compared to the state-of-the-art method presented in [63]. We found that we achieve similar accuracy for small camera displacements, significantly outperforming [63] in the presence of wide baselines.

3.2.1 Pipeline

This section describes a new method for estimating an RGB-D sensor's motion from the acquired color image-depth map pairs. For each pair of RGB-D images, two main consecutive steps are performed: an estimation of the sensor's relative pose between the two frames, and a refinement of this initial estimation. The reconstruction of the whole trajectory of the sensor is achieved by using only the pairs of consecutive frames, and does not take into account any prior information.

3.2.1.1 Relative pose estimation

The initialization step starts by segmenting the planes present in both RGB-D images, which is performed by using the method proposed by Taylor and Crowley [125]. Then, the relative pose estimation is initialized using the algorithm described in Section 3.1, by considering the image patches centred in each segmented plane. The transformation with the highest score ϵ is selected for further refinement.

3.2.1.2 Pose refinement with photo-consistency

The relative pose estimation carried using the segmented planes can be affected by noise in the plane segmentation step. To refine the initial estimation, we use an intensity-based registration procedure. In the perspective case, two images $\mathbf{q}_f$ and $\mathbf{q}_s$ of two planes $\mathbf{\Pi}^{(i)}$ and $\mathbf{\Pi}^{(i+1)}$, respectively, are related by an homography $\mathbf{q}_f \sim \mathsf{H}\,\mathbf{q}_s$ of the form:

$$\mathsf{H} = \mathsf{K}\left[\mathsf{R}_i + \mathbf{t}_i \frac{\tilde{\mathbf{n}}^\mathsf{T}}{d_f}\right]\mathsf{K}^{-1}, \tag{3.2}$$

where K represents the camera intrinsics, $d_f = 1/||\mathbf{n}||$ the distance of the plane to the origin of the reference frame, and $\tilde{\mathbf{n}}$ represents the unitary normal vector. By performing a normalization of $\mathbf{q}_f \sim \mathsf{H}\,\mathbf{q}_s$ to non-homogeneous coordinates, we can define a 2D warping function $\mathbf{w}(\mathbf{q}_f\,;\,\mathbf{p}) = \Psi\Big(\mathsf{K}\left[\mathsf{R}_i + \mathbf{t}_i \frac{\tilde{\mathbf{n}}^\mathsf{T}}{d_f}\right]\mathsf{K}^{-1}\mathbf{q}_f\Big)$, with Ψ denoting the normalization to non-homogeneous coordinates, and $\mathbf{p}$ being the warping parameter vector that encodes 3 parameters for camera rotation, 3 for translation, and 3 for the plane structure.

Given the 2D warping function $\mathbf{w}(\mathbf{q}_f;\mathbf{p})$, it is possible to define a cost function describing the sum of squared differences between the pixels of a planar patch in the reference, $\mathtt{I}_f$, and incoming, $\mathtt{I}_s$, images:

$$\gamma = \sum_{\mathbf{q}_f\,\in\mathcal{N}} \Big[\mathtt{I}_s(\mathbf{w}(\mathbf{q}_f\,;\,\mathbf{p})) - \mathtt{I}_f(\mathbf{q}_f)\Big]^2, \tag{3.3}$$

with $\mathcal{N}$ denoting a plane integration region. Since an initialization $\mathbf{p}$ of the parameter vector is already known from previous steps, we iteratively solve for $\delta\mathbf{p}$ increments on the warp parameters, with Equation 3.3 begin approximated by

$$\gamma = \sum_{\mathbf{q}_f\,\in\mathcal{N}} \Big[\mathtt{I}_s(\mathbf{w}(\mathbf{q}_f\,;\,\mathbf{p} + \delta\mathbf{p})) - \mathtt{I}_f(\mathbf{q}_f)\Big]^2 \approx$$

$$\approx \sum_{\mathbf{q}_f\,\in\mathcal{N}} \Big[\mathtt{I}_s(\mathbf{w}(\mathbf{q}_f;\mathbf{p})) + \nabla\mathtt{I}_s\frac{\partial\mathbf{w}}{\partial\mathbf{p}}\delta\mathbf{p} - \mathtt{I}_f(\mathbf{q}_f)\Big]^2. \tag{3.4}$$

By differentiating γ with respect to $\delta\mathbf{p}$, we obtained a closed form solution for $\delta\mathbf{p}$:

$$\delta\mathbf{p} = \mathcal{H}^{-1}\sum_{\mathbf{q}_f\in\mathcal{N}} \left[\nabla\mathtt{I}_s\frac{\partial\mathbf{w}(\mathbf{q}_f\,;\,\mathbf{p})}{\partial\mathbf{p}}\right]^\mathsf{T}\Big(\mathtt{I}_f(\mathbf{q}_f) - \mathtt{I}_s(\mathbf{w}(\mathbf{q}_f\,;\,\mathbf{p}))\Big), \tag{3.5}$$

with $\mathcal{H}$ being a 1st order approximation of the Hessian matrix [9], and the parameter vector being additively updated $\mathbf{p}^{j+1} \leftarrow \mathbf{p}^j + \delta\mathbf{p}$ at each iteration j. For robustness against a noisy camera motion initialization, we use a coarse-to-fine registration framework. We build an image pyramid by down-sampling the original image by factors of 2 (we use 3 pyramid levels). We start by optimizing the parameters at the coarsest level. After convergence (or a maximum number of iterations is reached), the resulting parameters are used to initialize the next pyramid level. The algorithm proceeds until the original image resolution is reached.

As explained in [8, 78], if only one plane is available it is impossible to estimate the 9 parameters of $\mathbf{p}$ from the 8 non-linear constraints of the homography. In such cases we fix the initial depth of the plane, and optimize the remaining 8 parameters of the warping function. In cases of multiple plane optimization, where the camera extrinsic parameters are the same for all the segmented planes, we adopt two different warping functions [78] that enable to estimate the camera motion globally for all the features being tracked:

$$\mathsf{H}^{(1)} = \mathsf{K}(\mathsf{R}_i + \frac{\mathbf{t}_i}{d_{f1}}\tilde{\mathbf{n}}_1^{\mathsf{T}})\mathsf{K}^{-1}, \qquad \mathsf{H}^{(k)} = \mathsf{K}(\mathsf{R}_i + \frac{\mathbf{t}_i}{d_{f1}}\frac{d_{f1}}{d_{fk}}\tilde{\mathbf{n}}_k^{\mathsf{T}})\mathsf{K}^{-1}, k > 1. \tag{3.6}$$

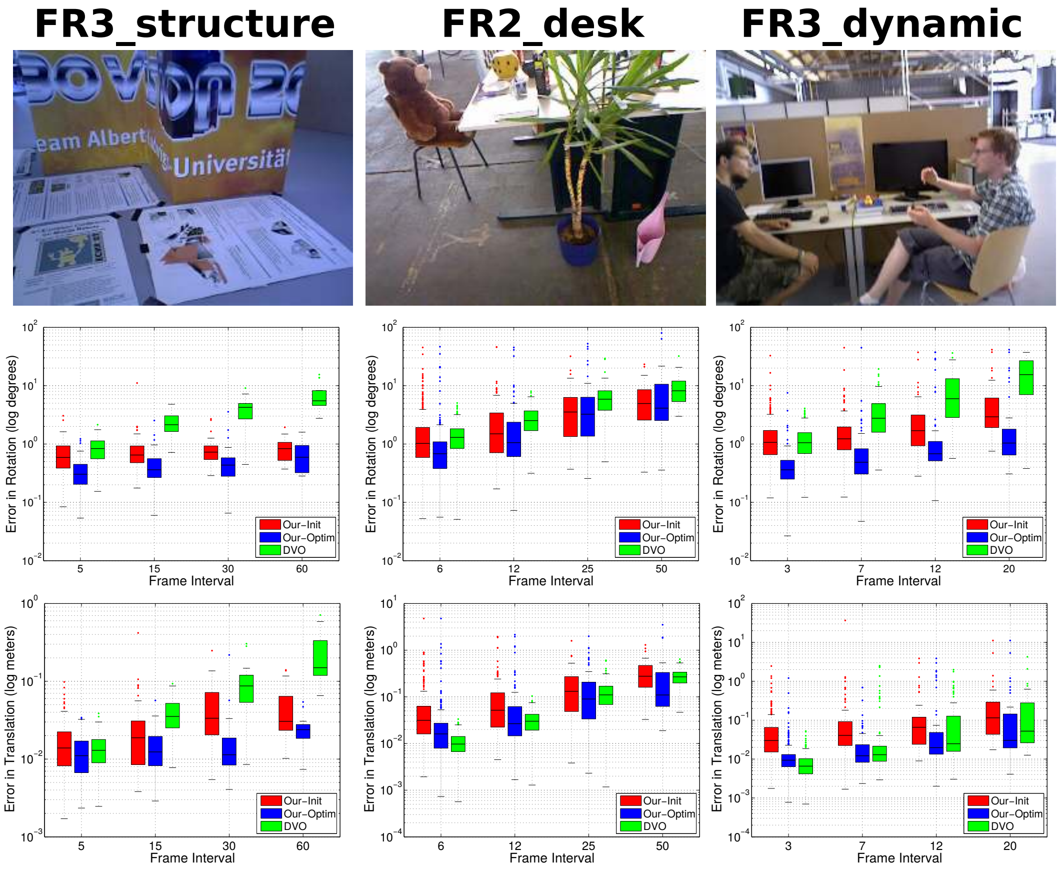
With such parametrization, we end up with a total of $6 \times 3k - 1$ parameters to optimize per frame pair. The parameter updates are computed using the Schur complement to explore the sparsity of the system. For further details on how to compute the parameters, we refer the reader to [8, 78].

3.2.2 Experimental results

In this section we conduct two sets of experiments to validate the proposed method in real scenarios. The first set uses a benchmark dataset with ground truth trajectories [120], while in the second we perform a loop-closure experiment with large baseline between frames.

3.2.2.1 Benchmark validation

The quantitative evaluation of our method is performed on 3 sequences from the TUM RGB-D dataset [120]. For simulating different camera velocities, we conduct the experiments by leaving out intermediate frames. We evaluate the pairwise camera motion estimations by computing the angular difference between the estimated and ground truth rotation matrix, and the norm of the difference between the estimated and ground truth translation vector. For comparison we use the dense visual



No. frames	75
% w/ pts	6.8%
No. planes	3.6
% outliers	3.9%

No. frames	124
% w/ pts	88.4%
No. planes	4.1
% outliers	7.6%

No. frames	110
% w/ pts	24.9%
No. planes	4.3
% outliers	12.3%

Figure 3.2: Benchmark validation. The first row shows a sample image of each sequence, and the second and third rows show the rotation and translation error per frame, respectively. The graphics show the performance of the different methods for different frame intervals. The last row shows some statistics regarding each dataset. We show the average number of frames per sequence, the average percentage of cases using points for computing the camera motion, the average number of planes used for registration, and, finally, the average percentage of outlier observations of our optimization step.

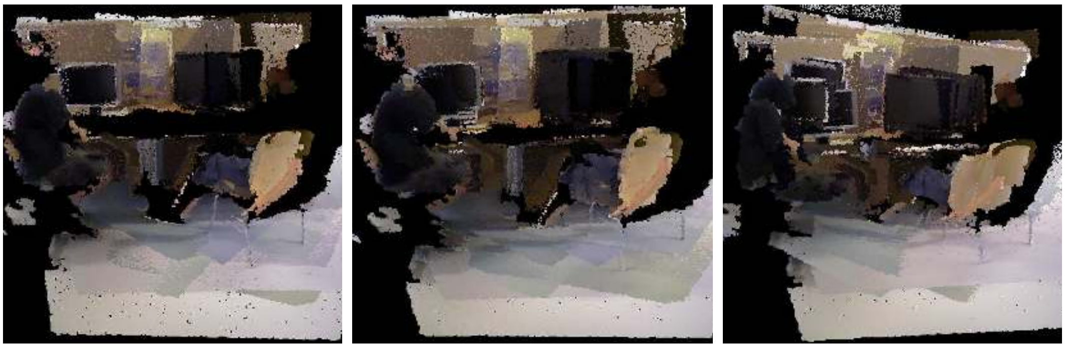
odometry (DVO) algorithm proposed by Kerl *et al.*[1].

Figure 3.2 shows the results for this controlled set of experiments. Dataset **FR3_structure** is dominated by large support textured planes without any occlusion. We can observe that the DVO algorithm shows good performance for the smallest frame interval. As we increase the baseline between frames, its performance starts to degrade due the larger number of outlier image pixels used for the global image registration. Our method presents an almost constant performance for all the baselines. In this particular dataset, the optimization by plane registration enables to estimate the camera rotation with a median error of less than 0.5 degrees, which is within the measurement error of the sensors used to compute the ground truth camera poses [120].

Dataset **FR2_desk** was acquired in a typical office environment, where planar surfaces present low texture (e.g. tables, monitor and floor). This places some challenges to our local photo-consistency optimization step. We observed this by the larger number of outliers in the box-plots when compared with the DVO algorithm, where photo-consistency is performed using all the available image pixels. By inspection of the results, we observed that the outlier estimations are mainly due to small support planes that, in conjunction with the noise from the initial estimation, do not allow enough overlap between views to successfully perform the registration. Overall, our algorithm performs better than the DVO for large baselines, being consistently better in rotation across all the baselines tested.

Finally, in the **FR3_dynamic** we validate our algorithm in a dynamic scene with two persons moving and partially occluding the surrounding environment. In this sequence, the camera has been rotated along the principal axes, with a minimal translation amplitude ($\approx$ 5 mm between camera poses). We can observe that our method clearly outperforms the DVO algorithm in terms of rotation accuracy across all the baselines, and in translation for the large baseline sequences. Despite DVO's poor performance in rotation, the algorithm is capable of providing good translation estimations. We believe this is a consequence of the motion *priors* used in the optimization step. Since the translation vector is always very small, the probabilistic filter, due the absence of reliable observations, probably favors the current state keeping the translation vector almost unchanged across frames. Figure 3.3 shows the 3D reconstruction obtained with the two methods for the sequence with 3 frames of interval, where DVO provides the lowest error in rotation. Since our algorithm is based on planes, which typically remain static, the camera motion recovery is less

[1] We use the source code provided by the authors at `https://github.com/tum-vision/dvo`



(a) Ground truth (b) Our-Optim (c) DVO

Figure 3.3: 3D reconstruction for the *FR3_dynamic* dataset. The figure shows the 3D reconstruction computed with (a) ground truth camera poses, (b) our method with optimization, and (c) the DVO method. The quality of the 3D reconstruction indicates that our algorithm is more robust than DVO for scenes with dynamic motion.

error prone and less influenced by the dynamic motion in the scene. This can be clearly seen by the accuracy of the 3D reconstruction where our method reconstructs the 2 existing monitors, while the DVO's reconstruction presents ”phantoms” due to the poor inter-frame registration.

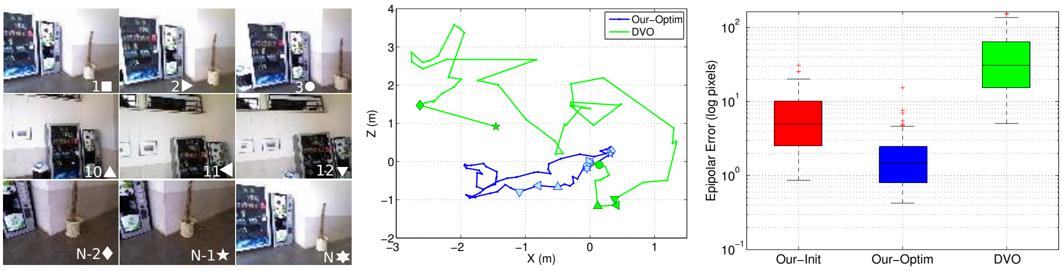
Note that none of the individual estimates where computed using only points. In every pair of frames of all three sequences, the algorithm was able to identify at least one plane correspondence.

3.2.2.2 Loop-closing experiment

In this experiment we navigate with a hand-held Kinect in a corridor to perform a loop-closed trajectory. This dataset is extremely challenging with difficult illumination conditions (see Figure 3.4a for some sample images), low texture, and fast camera motion (the images were acquired at 3Hz with a resolution of 1280×1024).

Figure 3.4b shows the trajectory estimation for the different algorithms tested. Our algorithm enables to keep a reliable trajectory estimation, with a consistent smooth transition between frames. The DVO method diverges after the first couple of frames, providing an erroneous trajectory. We believe this happens due to the large baseline between frames, which results in a large number of outlier pixels introduced in the DVO registration process.

Finally, we show in Figure 3.4c the epipolar geometric error to provide a quanti-



(a) Sample Images (b) Trajectory estimation (c) Epipolar Error

Figure 3.4: Loop closing experiment with N = 59 images. (a) shows some images of the sequence, (b) shows the estimated trajectories, and (c) compares the epipolar error of the different methods. The trajectory obtained with our method almost closes the loop.

tative error of the pairwise motion estimations. We use SURF to establish putative matches, which are filtered using a RANSAC procedure with the fundamental matrix. The inlier points are used to compute the Sampson distance for each method. We observe that our optimization procedure greatly improves our initial estimations. The epipolar errors obtained with the DVO algorithm justify the erroneous trajectory provided by this method.

Our algorithm was fully implemented in Matlab, taking in average 3 seconds per image pair, while the DVO algorithm runs at 30Hz. We believe that an optimized C++ implementation of our algorithm can achieve more than 10Hz.

3.3 SfM in case of passive stereo

Unlike the previous section where plane detection is trivial from the depth information provided by RGB-D cameras, in this section we propose to perform SfM in the case of passive stereo, where plane extraction is not as straightforward. The proposed pipeline combines the benefits of PPR and plane-based odometry by recovering both structure and motion from plane-primitives. It receives as input an image sequence acquired by a calibrated stereo rig and outputs the camera motion and 3D planes in the scene. These planes are segmented in each stereo pair, and the final piecewise-planar model is obtained by simply concatenating the PPR results from consecutive frames.

The pipeline builds on the work of Antunes *et al.* [3] in PPR from semi-dense depth estimation using the SymStereo framework, which proved to outperform competing methods for the case of two calibrated views [4]. We start by running a simplified version of Antunes's algorithm in each input stereo pair and use these initial plane detections to compute the relative pose between consecutive frames using the hierarchical scheme described in Section 3.1.

The next step is the joint refinement of camera motion and initial plane detections to obtain a coherent piecewise-planar model of the scene. In general, independent stereo detections of the same 3D plane are slightly different and must be merged into a single hypothesis before proceeding to bundle adjustment [40]. Moreover, it often happens that the same plane is wrongly reconstructed in a faraway view and correctly detected in a closer view, which means that the first plane hypothesis must be discarded and replaced by the second. We show that linking, fusing, and back-propagating plane hypotheses across stereo pairs can be conveniently formulated as a multi-model fitting problem that is efficiently solved using global energy minimization [25, 58, 71]. Thus, we propose to carry the joint refinement of motion and structure using the Propose Expand and Re-estimate Labels (PEaRL) framework [58] that alternates between a discrete optimization step, whose objective is to re-assign plane hypotheses to stereo pairs, and a continuous bundle adjustment step that refines the reconstruction results using the symmetry-energies arising from the initial semi-dense depth estimations [3, 5].

As a final step, the 3D plane surfaces detected in the scene are segmented in each stereo pair through dense labelling of the pixels. This can be accomplished using a standard MRF formulation, as proposed in [5, 40], with a data-term that quantifies label likelihood based on left-right photo consistency, and a smoothness term for regularization. We verified that these prior methods have difficulties in handling low-textured regions, where photo-consistency is ambiguous, and do not take into account coherence in visibility across successive frames. This section proposes a new MRF formulation, specific for sequential PPR, and that largely solves the above mentioned issues.

Our work relates with previous methods for PPR [37, 38, 40, 81, 114, 139] that operate in a batch manner by first applying point-based SfM to estimate the relative pose between monocular views [116], and then reconstructing the plane surfaces from all images in simultaneous. Unlike these methods, the algorithm herein described carries the 3D modelling in a sequential manner using a sliding window approach to concatenate the contributions of consecutive stereo pairs. This is an important difference that enables applications in visual odometry and SLAM. Thus, and since the method enables to estimate the relative pose between successive stereo pairs, it relates with prior works on visual odometry [28, 44, 62, 89]. We ran comparative experiments against the broadly used LIBVISO2 algorithm [44] that confirm the benefits of using plane-primitives, as opposed to image point matches, to recover the camera motion. In particular our method outperforms point-based methods in several circumstances such as wide baseline, repetitive appearance, low texture and specularities.

Since one relevant contribution of the proposed pipeline is the dense plane labelling of image pixels using MRF, it is worth reviewing previous formulations for the same purpose. In [114], Sinha *et al.* present an MRF formulation where the data term includes not only photo-consistency cues but also cues of geometric proximity between points and lines and free space violation. Inspired by this idea, the modification to the data term of the MRF proposed in this section consists in using the previously computed semi-dense labelling information [3] to decrease the cost of certain pixel assignments, which avoids violation of free-space and enables the labelling of low texture regions where photo-consistency alone leads to ambiguous results. In [40], Gallup *et al.* use a multi-view plane linking step that enforces global consistency across overlapping views. However, and since in our case the MRF optimization is independently performed in each stereo pair for the sake of scalability, inconsistencies may occur with different labels being assigned to the same structure in different views. This issue is tackled in a novel post-processing step that modifies the dense labelling of each view independently, by using the information from the dense labelling of another view that overlaps with the first.

3.3.1 Background

This section gives a brief review of background concepts that are useful for better understanding the proposed pipeline. We shortly discuss the PEaRL algorithm for geometric multi-model fitting, where the fitting is formulated as an energy-based optimization problem. Then, an overview of the two-view PPR framework proposed in [3] is provided.

3.3.1.1 Energy-based multi-model fitting

The authors of [58] discussed that methods that greedily search for models with most inliers while ignoring the overall classification of data are inappropriate for multi-model fitting, and formulating the fitting as a labelling problem with a global energy function is preferable. Following this, they propose the PEaRL algorithm that consists in 3 steps:

1. Propose an initial set of models (labels) $\mathcal{P}$ from the observations.
2. Expand the label set for estimating the spatial support (inlier classification).
3. Re-estimate the inlier models by minimizing some error function.

Given an initial model set $\mathcal{P}$, the multi-model fitting is cast as a global optimization where each model in $\mathcal{P}$ is interpreted as a label l. Consider that $d \in \mathcal{D}$ is a data point and that l_d is a label in $\mathcal{P}$ assigned to d. The objective is to compute the global labelling $\mathbf{l} = \{l_d | d \in \mathcal{D}\}$ such that the following energy is minimized:

$$E(\mathbf{l}) = \underbrace{\sum_{d \in \mathcal{D}} D_d(l_d)}_{\text{data term}} + \lambda_S \underbrace{\sum_{d,e \in \mathcal{N}} V_{d,e}(l_d, l_e)}_{\text{smoothness term}} + \underbrace{\lambda_L \cdot |\mathcal{F}_l|}_{\text{label term}}, \tag{3.7}$$

where $\mathcal{N}$ is the neighbourhood system considered for d, $D_d(l_d)$ is some error that measures the likelihood of point d belonging to l_d, and $V_{d,e}$ is the spatial smoothness term that encourages piecewise smooth labelling by penalizing configurations $\mathbf{l}$ that assign to neighbouring nodes d and e different labels. The label term is used for describing the data points using as few models as possible, with $\mathcal{F}_l$ being the subset of different models assigned to the nodes d by the labelling $\mathbf{l}$ (refer to [58]). In order to handle outlier data points in $\mathcal{D}$, the outlier label $l_\emptyset$ is used.

Energies containing only data and smoothness terms can be minimized using α-expansion [18]. In case all the terms of Equation 3.7 are taken into account, the energy can be optimized using an extension of α-expansion proposed in [58]. On the other hand, if the smoothness term is not considered, the problem becomes an Uncapacitated Facility Location (UFL) instance, which can be solved very efficiently using the message passing inference algorithm [71].

The third step of PEaRL consists in re-estimating the model labels l in $\mathcal{P}$, given the non-empty set of inliers. The new set of labels is then used in a new expand step, and the algorithm iterates between discrete labelling and model refinement until the energy of Equation 3.7 stops decreasing.

3.3.1.2 Pixel-wise plane labelling

Given a finite set of plane hypotheses contained in the scene, the final step of many existing PPR algorithms is to assign one of these planes to each pixel of the input images. For this purpose, a standard MRF formulation involving only the data and

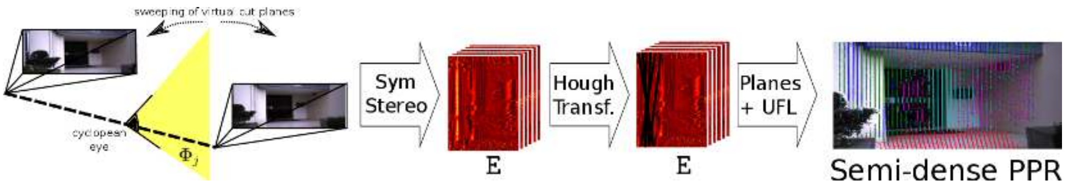


Figure 3.5: Two-view semi-dense PPR as described in Section 3.3.1.3. The inlier set of planes, along with the corresponding energies, is the input to the proposed pipeline. In the semi-dense labelling, each pixel of the cyclopean eye is assigned a plane label, each of which is identified with a color.

smoothness terms in Equation 3.7 is employed. The nodes $\mathbf{p}$ are the image pixels, and the labels $l \in \mathcal{P}$ are the plane hypotheses, where the label set $\mathcal{P}$ contains the scene planes $\mathcal{P}_0$ and the infinite plane Π_∞. The data term is defined as

$$D_{\mathbf{p}}(l) = \begin{cases} \min(\rho(l), \rho_m) & \text{if } l \in \mathcal{P} \\ \alpha\rho_m & \text{if } l = l_\emptyset \end{cases}, \tag{3.8}$$

where $\rho(l)$ is the photo-consistency between the pixels in the two views put into correspondence by the plane associated to label l, ρ_m truncates the cost and $\alpha < 1$. For measuring the photo-consistency the matching cost Zero-mean Normalized Cross-Correlation (ZNCC) is used. The smoothness term is defined as in [40].

3.3.1.3 Two-view semi-dense PPR

The first step of the proposed pipeline consists in obtaining a semi-dense PPR of the scene for each stereo pair. The method described in [3] was chosen for this purpose, mainly due to two characteristics: the fact that it was specifically designed for using two views, while other existing methods receive multiple images as input; and because superior results in terms of accuracy were reported when compared to other PPR methods [40,114]. The 3 major steps of the pipeline presented in [3] and illustrated in Figure 3.5 are as follows.

Step 1 Stereo-Rangefinding along virtual cut planes: Antunes et al. [4] have introduced a new stereo cost function, dubbed SymStereo, that is particularly well suited for estimating depth along a virtual cut plane passing in-between cameras. The approach uses a symmetry-based metric for obtaining an energy function that encodes the likelihood of each point in the virtual plane being a 3D point of the scene. In other words, the contour where the cut plane meets the scene should be a ridge of maxima in the energy function.

Step 2 Detection of plane hypotheses: Knowing that the intersections of the virtual planes with the planes in the scene are lines, the energy computed in Step 1 is used as input to a Hough transform for extracting line segments. Each pair of lines provides a plausible plane hypothesis.

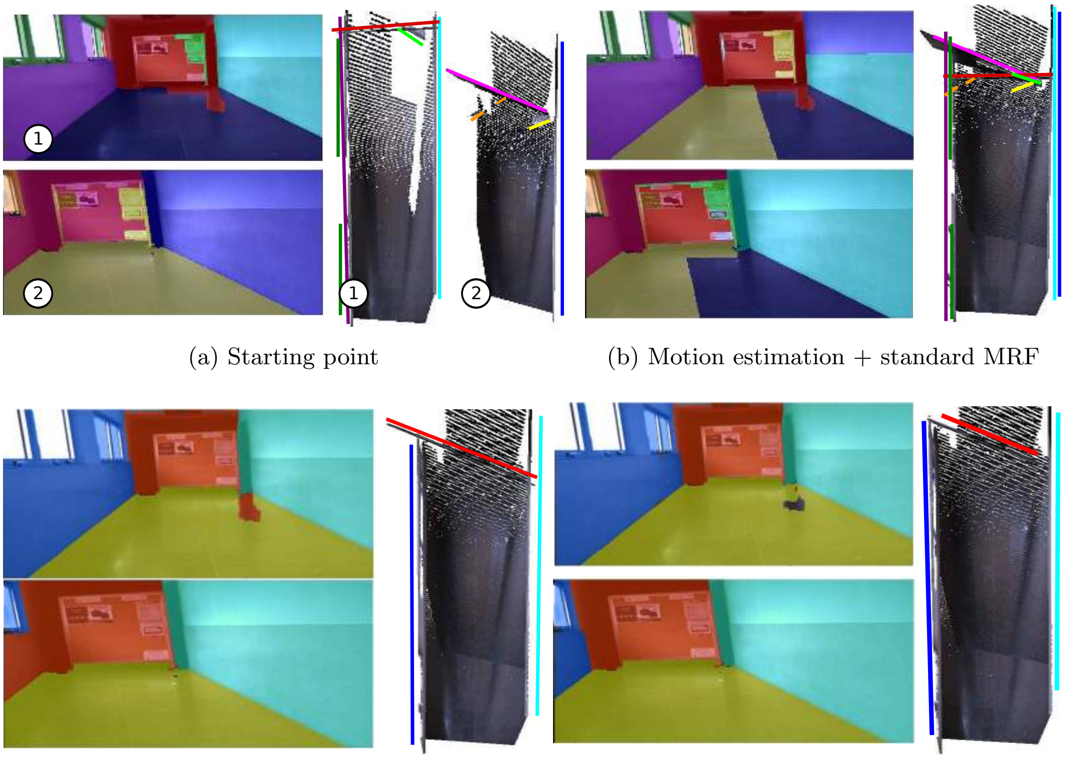
Step 3 Discrete-Continuous optimization: Let us assume a pencil of virtual cut planes $\mathbf{\Phi}_j$ intersecting the baseline in its midpoint. This can be thought of as an image created by a virtual camera that is located between the cameras (cyclopean eye), where each pixel is originated from the back-projection ray $\mathbf{d}_{j,r}$, corresponding to the intersection between the epipolar plane $\mathbf{\Psi}_r$ and the virtual plane $\mathbf{\Phi}_j$. In [3], the multi-plane fitting problem of assigning a plane label to each pixel of the cyclopean eye is formulated using the PEaRL algorithm, with an energy formulation as the one in Equation 3.7. Since we use their method in an initialization stage, we downsized the energy formulation by ignoring the smoothness term. In this case, the problem is reduced to an UFL instance and the solver of [71] can be used. This modification provides a less accurate but sufficiently good semi-dense PPR of the scene, being about 40% faster.

3.3.2 Overview of the problem and proposed solution

We propose a structure and motion framework that is able to automatically recover the camera positions and orientations along with a PPR of the scene from a stereo sequence. The explanation is given for a two-stereo pair sequence, being extended to longer sequences in a straightforward manner.

The starting point is a semi-dense PPR obtained for each stereo pair using a simplified and computationally more efficient version of the method proposed in [5], as summarized in Section 3.3.1.3. Due to its simplicity, problems such as spurious plane detections, inaccuracies due to e.g. low texture and slant, may occur. The reconstruction in Figure 3.6a presents some of these issues: the red plane in view 1 is inaccurately recovered due to its long distance to the camera, there is over-segmentation of the frontal plane, and the pink plane in view 2 is incorrectly segmented due to low texture and poor illumination. The output of this step is a set of plane hypotheses, with each reconstructed line contour assigned to one of these hypotheses.

The relative motion between consecutive stereo pairs is determined by registering plane hypotheses. This requires an algorithm for associating and registering planes across frames, as well as strategies to identify situations where plane information is insufficient and must be complemented with point correspondences. The method



(a) Starting point

(b) Motion estimation + standard MRF

(c) PEaRL + standard MRF

(d) PEaRL + proposed dense labelling

Figure 3.6: Segmentation and reconstruction results obtained in different scenarios: (a) Planes detected in each view independently, where a standard MRF is used only for visualization purposes. (b) Result obtained when applying a regular MRF after the motion initialization. (c) PEaRL refinement followed by a regular MRF. (d) Result obtained when applying the proposed pipeline with the new MRF formulation. In all cases, lines with colors corresponding to the detected vertical and nearly vertical planes are shown in the top view of the 3D models.

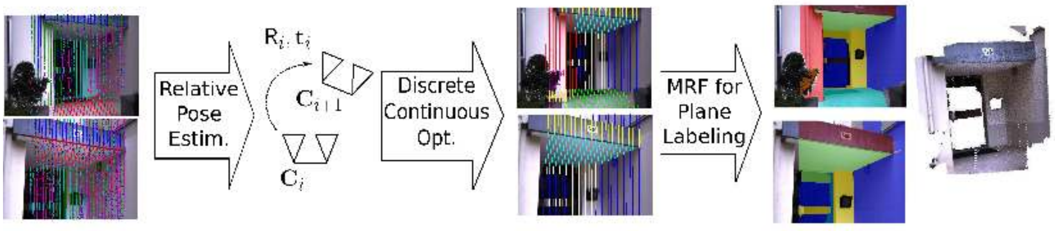


Figure 3.7: Different stages of the proposed pipeline. The planes detected in each view are used for the relative pose estimation. The pose and the plane hypotheses are refined in a discrete-continuous optimization step and a final MRF is used for dense labelling. Scene planes are identified with colors.

for carrying this step is described in Section 3.1. Given the relative motion between frames, the plane hypotheses arising from each pair can be represented in a common reference frame and dense pixel labelling can be carried using a standard MRF formulation as defined in Section 3.3.1.2. The problem is that the same 3D plane can give rise to multiple labels not only when it is detected in both views but also due to inaccuracies in the plane and relative motion estimation. Figure 3.6b illustrates this situation, where it can be seen that there is over segmentation in the plane labelling, and the reconstruction contains spurious planes.

As a consequence, a mechanism for merging plane hypotheses and back-propagating information across views, while simultaneously refining the camera motion and plane positions is required. This is achieved using the PEaRL algorithm reviewed in Section 3.3.1.1. It consists of a discrete optimization - planes detected in cameras C_i and C_{i+1} are assigned to pixels of the cyclopean eye, by minimizing an energy function in the format of Equation 3.7 - followed by the joint continuous optimization of the chosen planes and the relative pose. Further details are given in Section 3.3.3.

Having the planes and camera motion accurately determined allows using a standard MRF to obtain a correct dense labelling. However, and as observed in Figure 3.6c, this is not always the case as can be noticed in the labelling of the left wall and the floor. These problems have two main reasons: low-textured surfaces may cause photo-consistency to fail, and since the optimization is carried individually for each stereo pair, it might occur that the labelling becomes inconsistent across frames, leading to visibility issues. They are tackled in a final dense labelling step that includes an MRF segmentation followed by a novel post-processing step. Details on this new method are given in Section 3.3.4. By concatenating the individual reconstructions, it is possible to obtain a dense PPR for the complete sequence. Using this new method, the result in Figure 3.6d is obtained, where all the occlusions have been corrected.

As depicted in Figure 3.7, the solutions proposed for tackling all the aforementioned problems are concatenated in a pipeline that takes as input the semi-dense labelling of each stereo pair and a set of plane hypotheses, and outputs the correct dense labelling of a sequence of stereo pairs.

3.3.3 Discrete-continuous bundle adjustment

This section describes the optimization step that is carried for jointly refining the motion and the PPR. The previous single stereo PPR and relative pose estimation steps yield two sets of planes defined in the reference frames of cameras C_i and

C_{i+1}, $\mathbf{\Pi}_k^{(i)}, k{=}1\ldots K_i$ and $\mathbf{\Pi}_k^{(i+1)}, k{=}1\ldots K_{i+1}$, respectively, and an estimate for the relative pose R_i, $\mathbf{t}_i$ between the cameras. The optimization is achieved using the PEaRL algorithm (Section 3.3.1.1). The initial set of plane models $\mathcal{P}_0$ for PEaRL is the union of the $(K_i{+}K_{i+1})$ planes detected in each stereo pair separately. Then, consider the cyclopean eye relative to camera i, whose back-projection rays are denoted by $\mathbf{d}_{j,r}^{(i)}$, where r indexes a particular epipolar plane (refer to Section 3.3.1.3). The objective is to estimate the point on $\mathbf{d}_{j,r}^{(i)}$ that most likely belongs to a planar surface. This problem is cast as a labelling problem, in which the nodes of the graph are the back-projection rays $\mathbf{d}_{j,r}^{(i)} \in \mathcal{D}$, and to which we want to assign a plane label $l_{\mathbf{d}_{j,r}^{(i)}}$. The set of possible labels is $\mathcal{F} = \{\mathcal{P}_0, l_\emptyset\}$, where $l_\emptyset$ is the discard label and is used for identifying non-planar structures. This labelling problem is solved by minimizing an energy function E in the form of Equation 3.7, where the data and smoothness terms are modified such that they sum over the whole stereo sequence, becoming

$$\underbrace{\sum_i \sum_{\mathbf{d}_{j,r}^{(i)} \in \mathcal{D}} D_{d_{j,r}^{(i)}}(l_{\mathbf{d}_{j,r}^{(i)}})}_{\text{Data Term}} \text{and} \underbrace{\sum_i \sum_{\mathbf{d}_{j,r}^{(i)}, \mathbf{e}_{j,r}^{(i)} \in \mathcal{N}} V_{\mathbf{d}_{j,r}^{(i)}, \mathbf{e}_{j,r}^{(i)}}(l_{\mathbf{d}_{j,r}^{(i)}}, l_{\mathbf{e}_{j,r}^{(i)}})}_{\text{Smoothness Term}},$$

respectively, where $\mathcal{N}$ is the 4×4 neighbourhood of $\mathbf{d}_{j,r}^{(i)}$ and V is the spacial smoothness term. The data term $D_{\mathbf{d}_{j,r}^{(i)}}$ for the back-projection ray $\mathbf{d}_{j,r}^{(i)}$ is defined as

$$D_{\mathbf{d}_{j,r}^{(i)}}(l) = \begin{cases} \min(1 - \mathtt{E}_j^{(i)}(r, x_l), \tau) & \text{if } l \in \mathcal{P}_0 \\ \tau & \text{if } l = l_\emptyset \end{cases}$$

where the coordinate x_l is the column defined by the hypothesis l, corresponding to the intersection of $\mathbf{d}_{j,r}^{(i)}$ with the plane indexed by l. Using the camera pose, we can transform the planes detected in the stereo rig $i+1$ to the stereo rig i, and vice versa. This allows us to use all the structure information available simultaneously and reconstruct planes in a particular view even if they were detected by a different camera. The smoothness term V is used to describe the relationships between nodes. No penalization is assigned to neighbouring nodes receiving the same plane label, while in the case of one node obtaining the discard label, a non-zero cost is added to the plane configuration $\mathbf{l}$. For each camera i, the smoothness term V is defined as in [3], which encourages label transitions near crease or occlusions edges. For further details refer to that work.

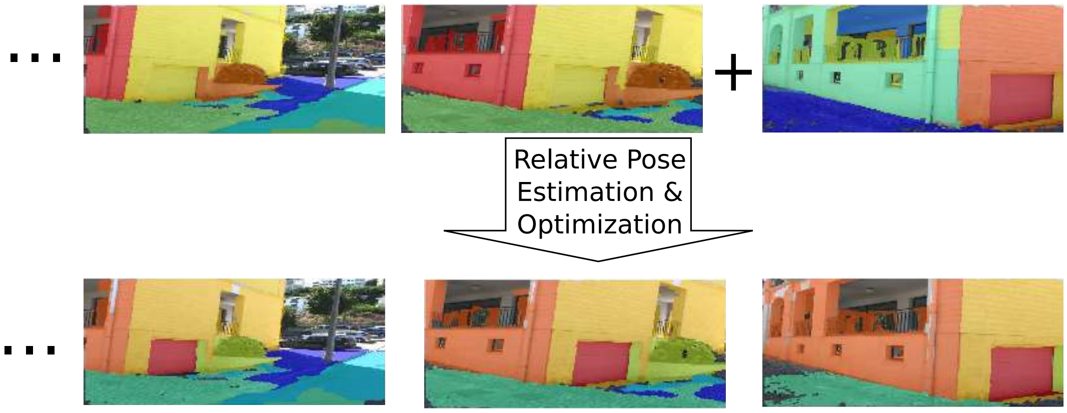


Figure 3.8: Back-propagation of planes across stereo pairs: a closer view of the door plane allows its correct detection and propagation to previous stereo pairs.

The output of this step is a set of planes shared by cameras C_i and C_{i+1}. Given the inliers of a particular plane label l, the corresponding energies $\mathtt{E}^{(i)}$ that come from SymStereo can be recomputed to enhance the likelihood measure with respect to a particular range of slant values [5]. These energies are used in the third step of PEaRL. Let $\mathbf{\Pi}_l$ be the plane associated to l to which has been assigned a non-empty set of inliers $\mathbf{D}(l) = \{\mathbf{d} \in \mathcal{D} | l_\mathbf{d} = l\}$. All the inlier planes $\{\mathbf{\Pi}_{l_k}\}$ and the relative pose R_i, $\mathbf{t}_i$ are refined simultaneously by minimizing the error function:

$$\{\mathsf{R}_i^*, \mathbf{t}_i^*, \{\mathbf{\Pi}_{l_k}^*\}\} = \min_{\mathsf{R}_i,\mathbf{t}_i,\{\mathbf{\Pi}_{l_k}\}} \sum_i \sum_k \sum_{\mathbf{d}_{j,r}^{(i)} \in \mathbf{D}(l)} \left(1 - \mathtt{E}_j^{(i)}(r, x_{\mathbf{\Pi}_{l_k}})\right) + \delta e_{ph}, \tag{3.9}$$

where $x_{\mathbf{\Pi}_{l_k}}$ is the column defined by the intersection of $\mathbf{d}_{j,r}^{(i)}$ with $\mathbf{\Pi}_{l_k}$, δ is a parameter that is zero whenever the optimization is carried out using 3 shared planes that span the 3D space and larger than zero otherwise, and e_{ph} is the photo-consistency error computed in a planar patch. The new set of plane labels $\mathcal{P}_1 = \left\{\mathbf{\Pi}_{l_k}^*\right\}$ is then used in a new expand step, and we iterate between discrete labelling and plane refinement until the α-expansion optimization does not decrease the energy E.

A sliding window approach is applied where at most one relative pose is refined. The exchange of planes between cameras, illustrated in Figure 3.8, has an important role in the 3D modelling process since it allows planar surfaces that are only properly detected in subsequent frames to be back-propagated and accurately reconstructed in previous images. Remark that plane information is only exchanged between different cameras inside the sliding window. In order to have plane propagation across distant views, a final PEaRL step using a significantly larger sliding window is applied to the whole sequence.

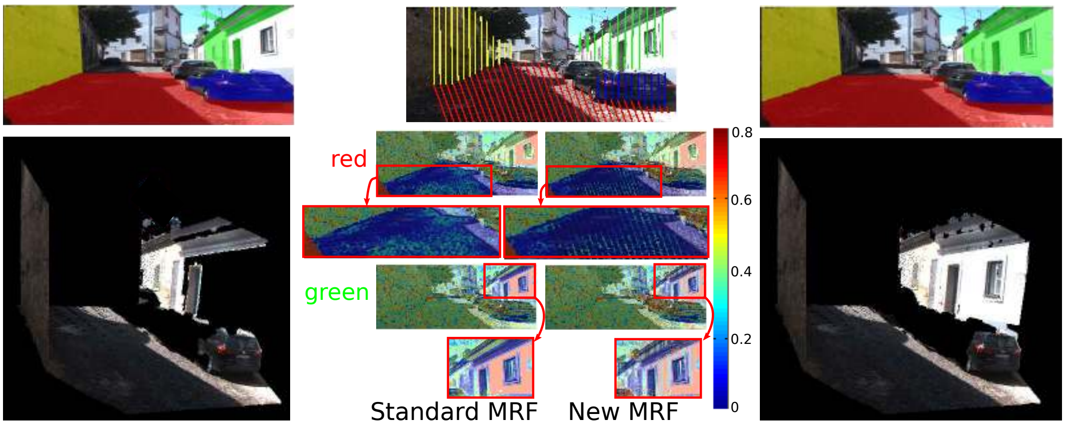
3.3.4 Dense labelling formulation

Commonly, the planes are densely segmented in each stereo pair using a standard MRF labelling as described in Section 3.3.1.2. Since its data cost solely relies on photo-consistency cues, this formulation tends to provide inaccurate labellings in cases of lack of texture or presence of non-planar surfaces at long distances. The result was not only the reconstruction of non-planar objects such as trees and pedestrians, but also the existence of occlusions and areas that failed to be reconstructed. Two new improvements to the described dense labelling framework are proposed for overcoming these issues. The first one consists in changing the data cost in the MRF formulation, whereas the second is a post-processing step that ensures coherence across views.

3.3.4.1 Updated data term

The standard MRF formulation described in Section 3.3.1.2 does not handle cases of textureless regions since the photo-consistency measure (ZNCC) becomes infinity in those regions due to the null variance. This often leads to pixels in textureless regions being discarded because they are assigned the highest cost ρ_m, although the correct plane hypothesis is in the label set. Figure 3.9a shows such an example where a large part of the white wall is not reconstructed. This is an important issue as it occurs frequently in outdoor scenarios. One possible solution to the problem would be to increase the penalty for plane change in the smoothness term, which will tend to extend neighbour planes to the textureless region. However, this has many disadvantages including the possibility of reconstructing non-planar objects and not recovering small planes.

Since the SymStereo framework [4] is able to handle low texture, the planes in these regions are correctly detected. Also, the semi-dense labelling in textureless areas is correct, as shown in Figure 3.9b. Although this information has not been used in the dense labelling step, it is relevant since it already solves part of the problem. Thus, it is proposed to incorporate the semi-dense labelling information in the MRF formulation by changing its data cost with the intent to enforce coherence between the semi-dense and dense labellings. The smoothness term then plays the role of extending the labelling to neighbouring pixels. This results in a more robust framework that includes not only photo-consistency cues but also, indirectly,



(a) Standard MRF (b) Semi-dense labelling (c) New MRF

Figure 3.9: (a) Due to the lack of texture, the white wall is not fully reconstructed when using the standard MRF formulation. (b) Data cost of each pixel, for two different labels corresponding to a scene plane, obtained with a standard MRF (left column) and new the MRF formulation (right column). The color corresponding to each plane in the semi-dense labelling (top image) is indicated next to the data cost matrices. The color bar corresponds to the matching cost. (c) The new MRF assigns the correct label to the textureless pixels.

symmetry ones that come from SymStereo [4].

The data cost in Equation 3.8 is modified by decreasing the cost of pixel $\mathbf{p}$ and its neighbours being assigned label f if $\mathbf{p}$ was assigned that label during the semi-dense labelling. In other words, let us first consider the set of all pixels $\mathbf{p}_i$ that were assigned label f in the semi-dense labelling, and define the $n \times n$ neighbourhood of each pixel as $\mathcal{N}_{\mathbf{p}_i}^{f}$ and the union of all these neighbourhoods as $\mathcal{U}^f$. If a given pixel $\mathbf{p}$ belongs to $\mathcal{U}^f$, the cost of that pixel being assigned label $l = f$ is decreased by a constant value λ. Also, if a pixel was discarded in the semi-dense labelling, i.e. if $f = l_\emptyset$, it and its neighbours have a constant data cost for all plane labels. The new data cost is then defined as

$$D_{\mathbf{p}}(l) = \begin{cases} \max\left(\min(\rho(l), \rho_m) - \lambda, 0\right) & \text{if } l \in \mathcal{P} \wedge \mathbf{p} \in \mathcal{U}^l \\ \alpha \rho_m & \text{if } l = l_\emptyset \vee \mathbf{p} \in \mathcal{U}^{l_\emptyset} \\ \min(\rho(l), \rho_m) & \text{otherwise} \end{cases}, \qquad (3.10)$$

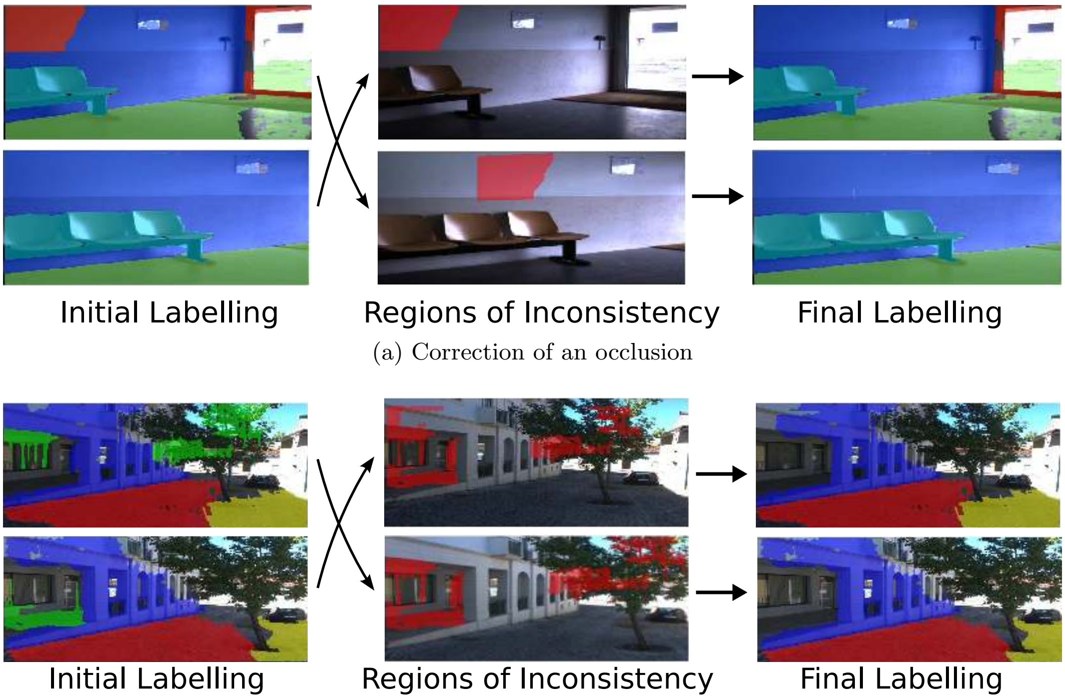
where λ is the parameter that controls the decrease in the cost and $\alpha < 1$. Another parameter to be controlled is the size n of the neighbourhood that corresponds to the thickness of lines of pixels whose data cost is decreased. This parameter must be tuned by taking into account the density of virtual cut planes, meaning that the higher the density, the thinner the lines can be. The remaining parameters are defined as in Section 3.3.1.2.

Remark that if a pixel of the cyclopean eye is assigned label $f \neq l_{\emptyset}$, its location in image I is computed from the corresponding plane equation. On the other hand, pixels assigned the discard label $l_{\emptyset}$ are not reconstructed and thus their location in the reference image I is not known. This issue can be tackled by making use of the symmetry energies delivered by SymStereo, as they provide the matching likelihood of pairs of pixels in the stereo views. For each pixel of the cyclopean eye assigned $l_{\emptyset}$, the maximum value of the energy is considered, providing a match. This allows the pixel to be reconstructed and then back-projected on the image I, allowing its location to be estimated.

Figure 3.9b shows the semi-dense labelling and the data costs computed with the standard MRF, in the left column, and the new MRF, in the right column, for each detected plane. It can be seen that, for each label, the data costs obtained with the new formulation are different from the original ones since lower values have been assigned to pixels with that label in the semi-dense labelling. In particular, for the green plane (last row), almost all pixels have a very high cost using the standard MRF, whereas with the new one, many pixels are assigned significantly lower costs, originating the reconstruction of the whole surface, as shown in Figure 3.9c. Also, pixels that had previously been discarded are assigned a high cost for all labels, as can be seen in the image area near the cars.

3.3.4.2 Label consistency across views

In image sequences, a correct labelling of all frames is the one in which corresponding areas in different views have the same label, i.e., represent the same plane. The discrete optimization step already greatly solves this problem since it selects the plane set that better describes the scene from multiple views. However, as shown in Figure 3.6c, a standard MRF for computing the dense labelling may originate inconsistencies in small areas of the images, leading to problems such as occlusions and the reconstruction of non-planar objects. In this section a post-processing step that enforces consistency across two consecutive views is proposed. The dense labelling of each view is modified independently, by using the information from the labelling of the other view. We refer to the image of camera C_i by I_i, and to its dense labelling by D_i. The procedure is described for the labelling of I_1, D_1, being applied similarly for D_2. It is as follows:



(a) Correction of an occlusion

(b) Vegetation removal

Figure 3.10: New post-processing step applied after the MRF labelling to ensure label consistency across frames. The initial dense labelling of each of the two frames (images on the left) is modified by finding the areas of disagreement between labels (images in the middle) and changing them so that corresponding areas in the scene have the same label (images on the right). This allows to correct problems of (a) occlusion and (b) the reconstruction of non-planar objects such as vegetation.

1. Reconstruct the points in image I_1 from the assigned plane labels and represent the dense labelling of I_1 in C_2, referred to as D_1^2.

2. Find the areas of inconsistency between D_1^2 and D_2 by finding the pixel locations where the assigned labels are different. In Figs. 3.10a and 3.10b these areas are shown in red in the middle images.

3. The labels of pixels that belong to these areas must be modified so that there is coherence across views. The proposed approach for performing this modification treats the discard and non-discard labels differently. Two labels are considered for each of these pixels, l_1^2 and l_2, that correspond to D_1^2 and D_2. If $l_2 = l_\emptyset$, then l_1 is also set to $l_\emptyset$, where l_1 is the label corresponding to l_1^2 in image I_1. Thus, the discard label $l_\emptyset$ works like an absorbing element. The alternative would be to consider $l_\emptyset$ as a neutral element, which is not appropriate

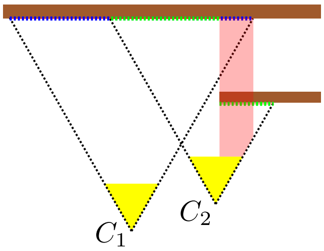


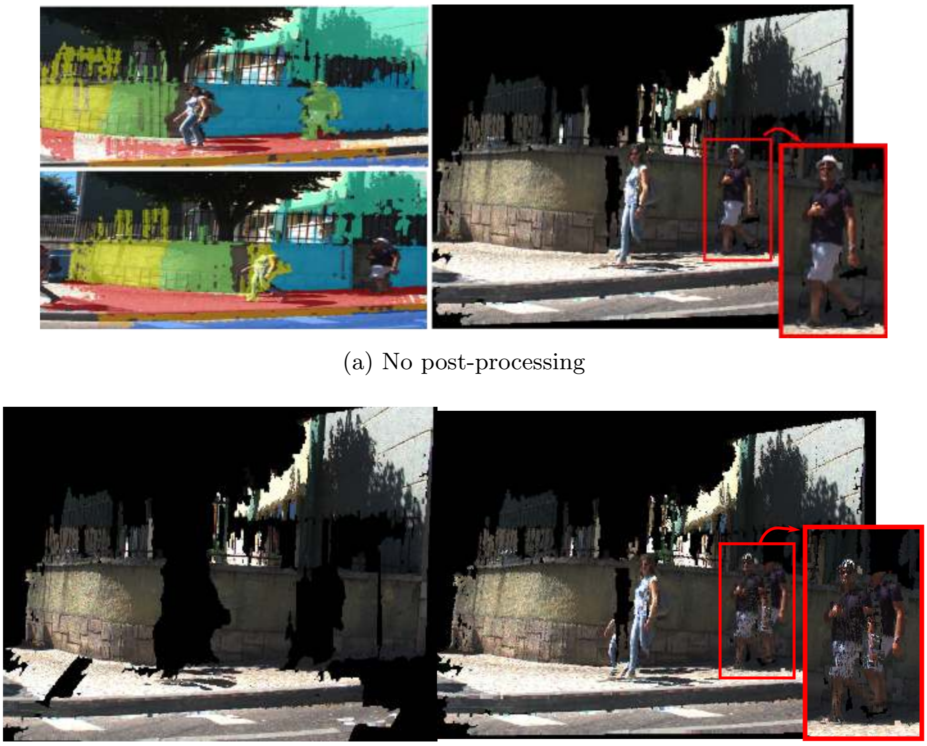
Figure 3.11: Example of a case where a legitimate occlusion originates label inconsistency. The proposed approach detects that this occlusion is a legitimate one and does not change the labels of the corresponding pixels.

as justified next.

4. Otherwise, if $l_2 \neq l_1^2 \wedge l_2 \neq l_\emptyset \wedge l_1^2 \neq l_\emptyset$ the point that is being analysed is reconstructed using both l_2 and l_1^2. The distances of both 3D points to camera C_2 are computed and l_1 is set to the label that yields the shortest distance. The reasoning is that, in general, closer surfaces are reconstructed more accurately than farther ones.

Note that there are cases in which having label inconsistency between views is correct. Figure 3.11 illustrates such an example. Two cameras observe a scene that contains two walls. While camera C_1 only observes the farther wall, as shown by the blue line, camera C_2 views both of them since they are inside its Field-of-View (FOV), depicted by the green line. The area in red shows the region where there is overlap of the cameras' FOVs, originating label inconsistencies since different surfaces are being observed. However, the points that belong to the closer wall viewed by camera C_2, when projected on the image plane of camera C_1, fall outside its FOV. Thus, our post-processing step does not modify the labellings in this case, being able to distinguish between legitimate and non-legitimate occlusions.

The described procedure allows to achieve consistency in all pixels of two views, enabling the correction of possible reconstruction errors. Figure 3.10 depicts two different common errors that can occur: occlusions and reconstruction of non-planar objects. In Figure 3.10a, although all label assignments are correct for the first view, some pixels that belong to the wall were incorrectly assigned to the door plane in the second view. This generates an occlusion since the area that is incorrectly reconstructed cannot be observed by any of the cameras as it is occluded by the wall. Correcting the labelling so that it becomes coherent across frames yields an accurate reconstruction without occlusions. Figure 3.10b depicts an example where a non-planar object (vegetation) is assigned to an existing plane when it is observed



(a) No post-processing

(b) $l_\emptyset$ as absorbing element

(c) $l_\emptyset$ as neutral element

Figure 3.12: Reconstruction results obtained when (a) not using the proposed post-processing step, (b) using the post-processing step as it is proposed, and (c) using the post-processing step with the alternative of considering the discard label as a neutral element.

from a long distance, and only correctly discarded in a closer view. By ensuring consistency across frames, all of the pixels belonging to the tree become discarded, originating a reconstruction without non-planar objects. Remark that this only happens because the discard label is being considered as an absorbing element. As mentioned before, it could instead work as a neutral element, and discarded pixels in one view that originated inconsistencies would be assigned the other label. In Figure 3.12, concatenated reconstructions are shown for three different scenarios. It can be seen in Figure 3.12b that using the proposed post-processing step - discard label as an absorbing element - originates the complete removal of pedestrians, but has the disadvantage of also removing the surrounding parts corresponding to the wall, caused by the fact that the pedestrians are moving. If the discard label was considered as a neutral element (Figure 3.12c), the outcome would be the reconstruction of the same pedestrian in both views, becoming visible in different positions along the same plane. Although this can be an interesting application in certain cases, considering $l_\emptyset$ as a neutral element would cause the vegetation in Figure 3.10b to be reconstructed in both views, which is a very poor approximation. Since this is a situation that happens very often in urban scenes, the choice of considering the discard label as an absorbing element is more appropriate.

3.3.5 Experimental results

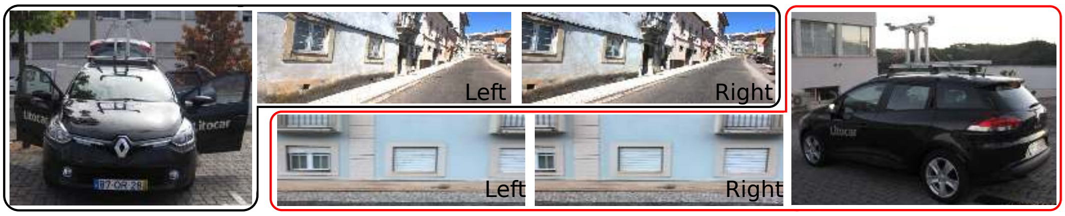
This section reports several experiments that validate the Piecewise Planar StereoScan (PPSS) framework and compares its performance against conventional point-based methods, more specifically against the LIBVISO2 [44] that is a widely used method for visual odometry using passive stereo. We started by running tests using the publicly available KITTI dataset [43] but soon realised that its sequences are not well suited to evaluate methods that rely in plane primitives as opposed to points. Although KITTI comprises sequences acquired in urban environment, the building facades providing the plane surfaces to be used by PPSS are in general too far away to be properly reconstructed. The reason for this is that the baseline between stereo cameras is small and only enables accurate depth estimation up to 13 meters [39]. The short baseline favours point-based methods because it provides large image overlap and prevents changes in perspective that can hamper matching. Unfortunately, it proved to be unsuitable to evaluate a method relying in plane primitives. Nevertheless, and for the sake of completeness, section 3.3.5.1 presents some results using the KITTI dataset.

In the absence of a suitable public dataset, we decided to collect our own sequences using three distinct camera setups whose characteristics are summarized in the table of Figure 3.13a. Setup S1 is a Bumblebee camera from PointGrey that was used to collect indoor sequences. The other two setups consist of a pair of synchronized cameras mounted on the roof of a vehicle whose images were carefully calibrated and rectified using standard methods. In setup S2 the cameras were mounted in a forward looking position, while in S3 the cameras were pointing to the right side of the vehicle (Figure 3.13b). Please note that the sequences acquired by S2 are challenging for most stereo methods because the image of the building facades are highly slanted.

The implementation of the PPSS pipeline for these experiments was done in MATLAB and the current computation time in a regular PC is around 60s for pair of stereo frames. This means that, for a sequence of N frames, the recovery of camera motion, detection of plane surfaces in the scene, and dense labelling of all images

Setup	S_1	S_2	S_3
Brief Description	PtGrey Bumblebee	Vehicle Forward	Vehicle Lateral
Resolution	1024×384	1142×410	1187×436
Baseline	$24cm$	$80cm$	$80cm$
FPS	-	7.5	7.5
Max. Range	10.3m	30.3m	31.6m

(a) Specifications of the acquisition setups



(b) Cameras mounted on a car and respective acquired stereo pair

Figure 3.13: (a) Information about image resolution, stereo camera baseline, acquisition rate and maximum range for accurate depth estimation for the 3 acquisition setups. The maximum range is computed by considering a depth error of 2% and a disparity error of 0.6 pixels.(b) Vehicle setups S_2 and S_3 with a corresponding acquired stereo pair enclosed in a black and red box, respectively.

takes about $(N-1) \times 60$s, which is way above the near real-time performance of point-based methods for visual odometry. It is true that the two approaches are not directly comparable in the sense that PPSS provides a complete, visual pleasing 3D model of the scene while conventional VO outputs a sparse 3D point could. Nevertheless, it is acknowledged that computation time is for the moment a weakness that must be improved in future work namely by exploring parallel computing (e.g. GPU). This is out of the scope of this article, whose objective is to introduce and provide experimental evidence on the advantages of using plane primitives in SfM applications.

3.3.5.1 Experiments with the KITTI dataset

A set of 3 sequences of the KITTI dataset containing buildings was selected for testing both PPSS and LIBVISO2 [44]. It was observed that using the original sequences, LIBVISO2 performed more accurately than our method, originating average errors of approximately 2% in translation and 0.1° in rotation, as opposed to 6% in translation and 0.5° in rotation for PPSS. As discussed before, the reason for

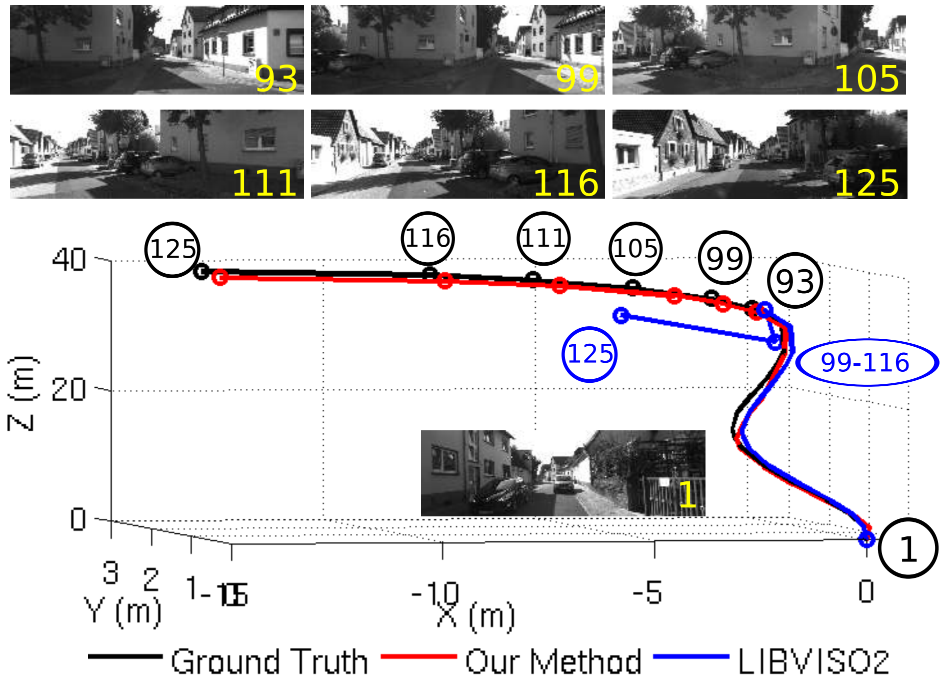


Figure 3.14: Results on a sequence of the KITTI dataset [43], using a sampling of frames that originates an average distance between consecutive frames of about 4 meters. LIBVISO2 outperforms our method when there is significant overlap between frames (as a consequence of straight movement) but diverges otherwise, while our method still provides acceptable results. It can be seen that LIBVISO2's performance degrades between keyframes 99 and 125, where a large rotation component is observed. It did not even provide an estimate for the camera motion between frames 99 and 116.

this is that KITTI clearly favours a point-based approach due to its small baseline and high frame rate.

In order to assess the performance with smaller amounts of data, and create difficulties to LIBVISO2, we sampled the sequences by considering only every fifth frame (corresponds to about 4 meters between consecutive frames). We observed that LIBVISO2 has good performance when the combination of camera motion and scene structure results in images with much overlap, even if the camera translation is high, yielding similar errors as the ones obtained with the complete sequences (about 2% in translation and less than 0.1° in rotation). However, otherwise it easily diverges, as shown in Figure 3.14 where it was not able to estimate the camera motion near the curve. As expected, reducing the number of frames did not influence the performance of PPSS as it properly handles situations of wide baseline. Thus, the average errors in the motion estimation were nearly the same as the ones obtained for the complete sequences.

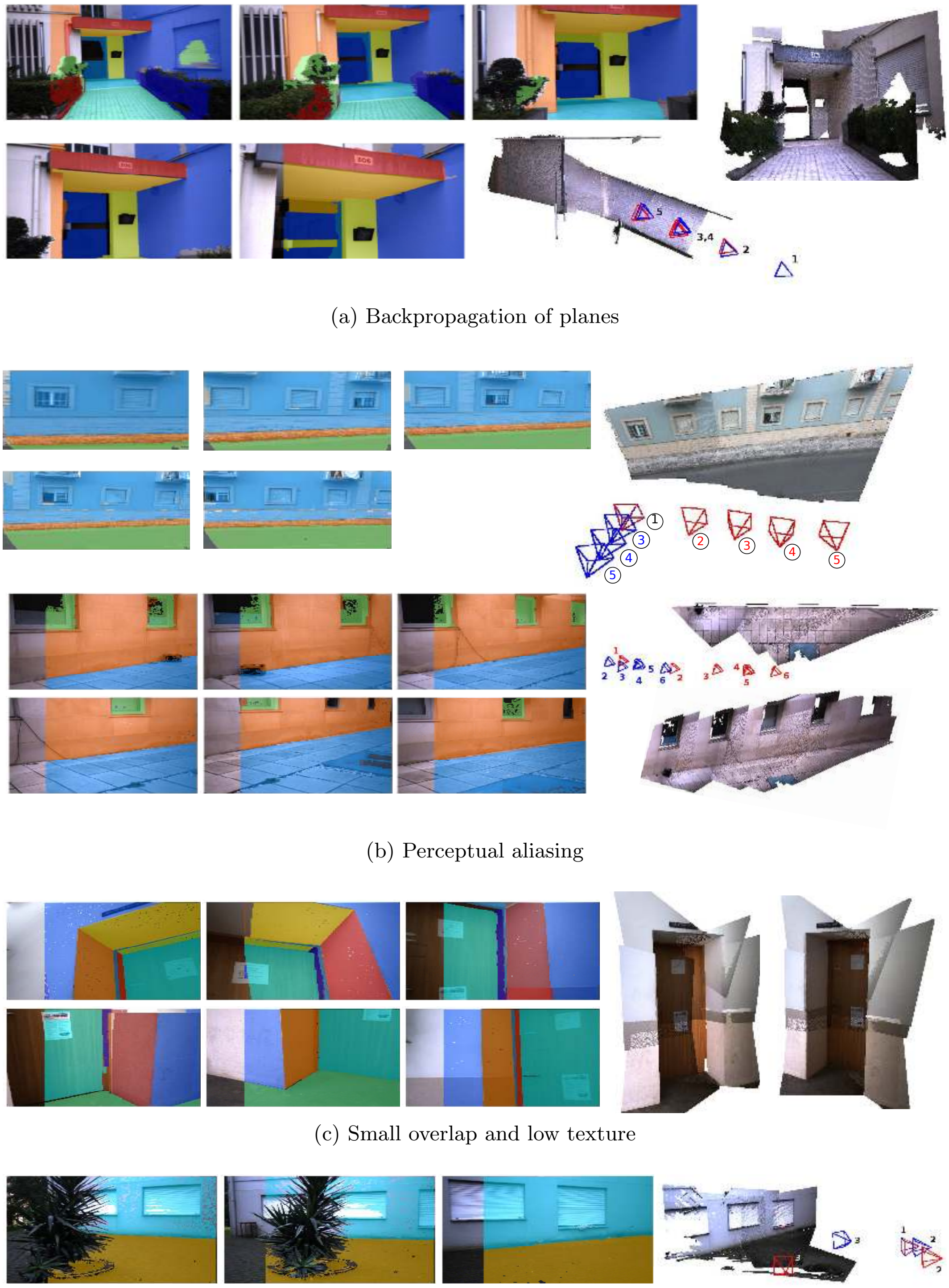
3.3.5.2 Experiments to evince the benefits of PPSS

This section presents a set of experiments on short stereo sequences selected with the intent to show the different features and advantages of the proposed method. A comparison with the point-based method LIBVISO2 [44] is given whenever it manages to provide a motion estimation. In those cases, the images of the 3D reconstructions include camera symbols in red and blue, if they were computed using PPSS or LIBVISO2, respectively. The left images of the stereo pairs are shown with the overlaid MRF labelling, where each color identifies one plane. The same color across images corresponds to the same plane. The sequence of images is sorted from left to right and top to bottom, and the cameras are numbered accordingly.

Figure 3.15a shows a 5-frame stereo sequence that was acquired with significant overlap in order to illustrate the exchange of planes between frames. It can be seen that in the first stereo pairs, the top plane of the entrance has very small image support, and thus cannot be recovered. Moreover, the back plane (containing the door) is poorly estimated since it is only observed from a long distance. These two planes are correctly reconstructed in the last frame, and back propagated to the initial frames, providing an accurate reconstruction of the whole scene. Our method and LIBVISO2 provided very similar results.

A 5- and a 6-stereo pair sequence of scenes with the presence of perceptual aliasing were acquired with stereo cameras S_3 and S_1, respectively. Results in Figure 3.15b show that our method was able to provide accurate reconstructions of the scenes. In the first example, it properly distinguished between the road (green) and side walk (orange) planes. In the second, it managed to accurately align the floor tiles. This is due to the fact that the algorithm requires only one correct point correspondence to be extracted since there are always at least two non-parallel plane correspondences between stereo pairs for estimating the camera motion. On the contrary, LIBVISO2 performed poorly on these sequences, not being able to estimate some relative poses and providing inaccurate results for the remaining ones. This is a consequence of the perceptual aliasing as most of the extracted point matches are incorrect.

A sequence of six stereo pairs with minimum overlap was acquired, originating a detailed reconstruction of a door (Figure 3.15c). It can be seen that the white low-textured walls and the small interior planes were accurately recovered. LIBVISO2

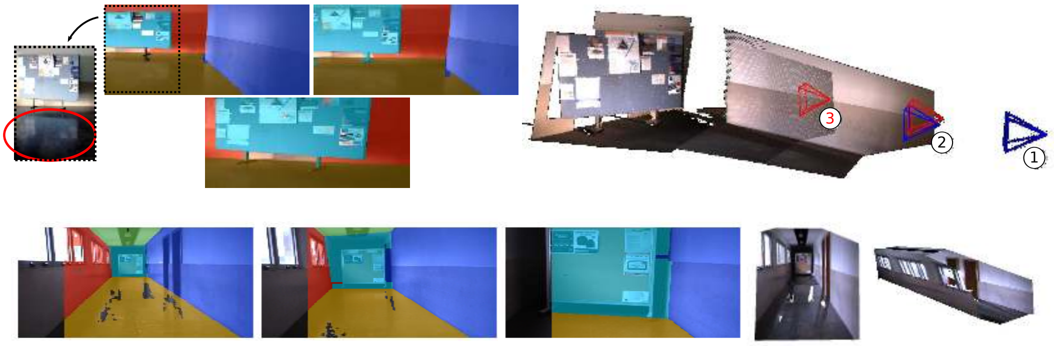


(a) Backpropagation of planes

(b) Perceptual aliasing

(c) Small overlap and low texture

(d) Vegetation removal



(e) Image specularities and high slant

Figure 3.16: Example cases to illustrate (a) the backpropagation of planes, the presence of (b) perceptual aliasing, (c) small overlap and low texture, (d) non-planar objects (vegetation), and (e) image specularities where the proposed plane-based method performs accurately, while the point-based method LIBVISO2 [44] fails. Red and blue camera symbols show the relative pose estimated using the proposed method and LIBVISO2, respectively. In (e), the image specularities are shown without the overlaid label and encircled in red for better visualization.

failed to find sufficient point correspondences for estimating the camera motion and thus camera symbols are not shown. Our approach computed the camera motion using correspondences of two planes and one point, as there are no triplet correspondences in consecutive stereo pairs.

The outdoor example in Figure 3.15d shows that our method is able to correctly distinguish between planar and non-planar objects, which can be used to automatically remove trees and vegetation from the final 3D model. The fact that the vegetation occupies a large part of the camera's field of view leads to a large percentage of incorrect point matches. Thus, LIBVISO2 provided very poor results for the estimation of the relative pose. As an example, camera 3 appears to be in an impossible position since it is in the vegetation's location.

Figure 3.16e shows results on the reconstruction of a scene with specular reflections. The acquired images (with camera S_1) show that the floor plane works like a mirror, reflecting the objects above it. Despite these specularities, PPSS performed well, providing a good reconstruction of the whole scene. LIBVISO2 was only able to properly estimate the camera motion between the first two positions, as there is significant overlap. It diverged in the last position due to the large difference in scale, originating incorrect point matches.

The last example is a 19-stereo pair sequence of an indoor scene acquired with

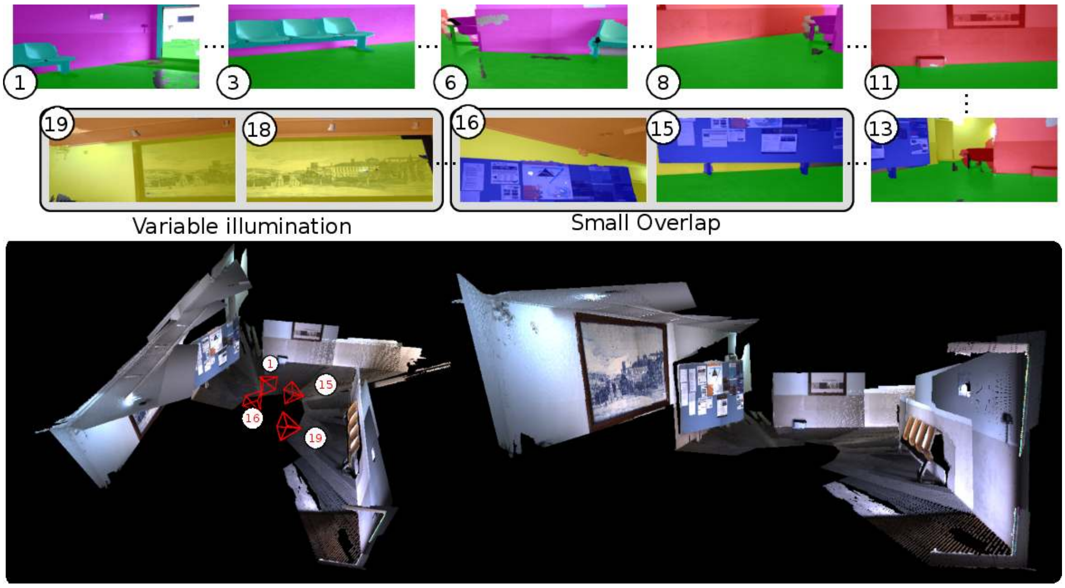
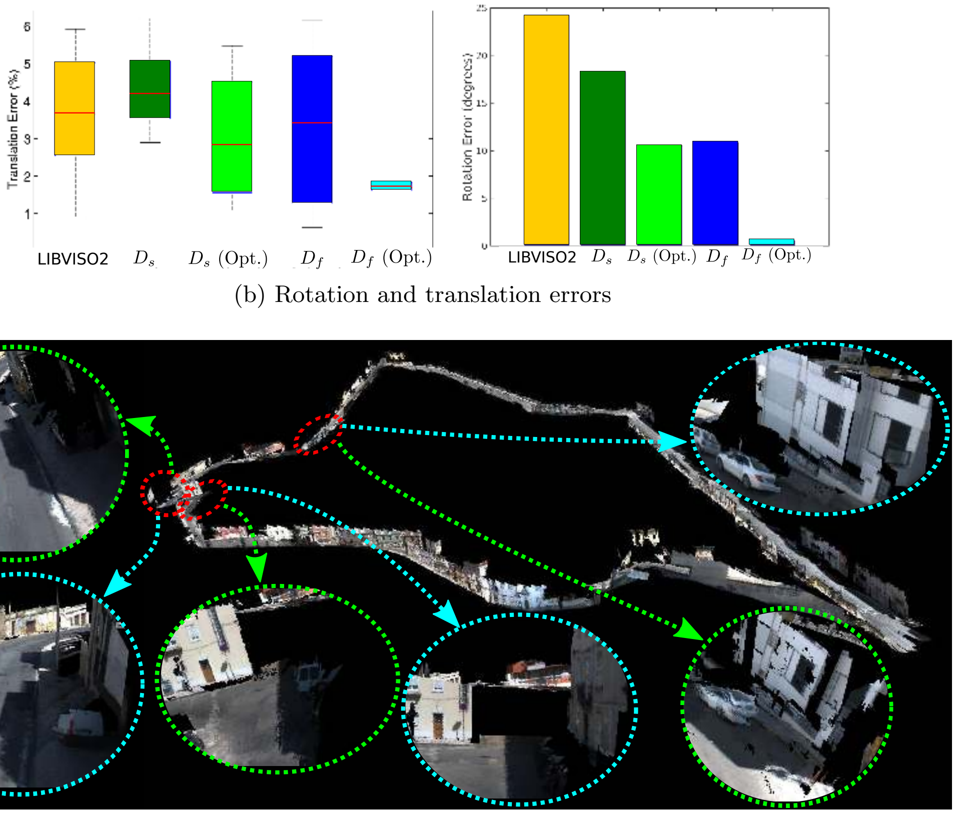


Figure 3.17: A challenging sequence of 19 stereo pairs presenting variable illumination and very low texture conditions was acquired, originating a partial reconstruction of a room. Since the camera motion is mainly composed of a rotation component, only 4 camera positions are shown in the top view of the reconstruction. LIBVISO2 failed to provide an estimation in most cases, so camera poses are not included. It is shown that our algorithm works in different illumination conditions (frames 18 and 19), small overlap (frames 15 and 16) and low texture (white walls and ceiling). The top row includes sampled images of the sequence indicating the order in which they were acquired and processed.

camera S_1, that illustrates that PPSS is adequate for reconstructing a full indoor room with a relatively small number of images. As shown in Figure 3.17, this is a challenging sequence presenting variable illumination, very low textured surfaces and small overlap between consecutive images. Our algorithm successfully recovered all the relative camera poses, yielding an accurate and almost complete reconstruction of the room. The reconstruction shows that the ceiling plane is not perfectly recovered. However, a good approximation is obtained, which is impressive given the lack of texture. Due to the difficult conditions of this dataset, LIBVISO2 was only able to compute 6 out of the 18 motions, with an average error of 24 cm in translation and 4.3° in rotation when compared to our solution.

Modality	Method	No. Ims
LIBVISO2	Point based [44]	1370
D_s	Plane based	380
D_s (Opt.)	D_s w/ final PEaRL	380
D_f	Plane based	1370
D_f (Opt.)	D_f w/ final PEaRL	1370

(a) Different modalities of methods and datasets, with the corresponding number of stereo frames (column *No. Ims)*



(b) Rotation and translation errors

(c) 3D model

Figure 3.18: (a) Description of the different modalities of methods and datasets used, and (b) corresponding rotation and translation loop closing errors. Small rotation errors yield small standard deviations in the translation errors. Since the translation error depends on the rotation, the median value in the translation errors distribution (red mark) should be considered when assessing the quality of the motion estimation. (c) Full 3D reconstruction using modalities D_f (Opt.) and D_s (Opt.). Selected areas are shown in greater detail. Videos of the 3D reconstructions obtained for D_s (Opt.) and D_f (Opt.) can be accessed in `https://youtu.be/wrBaV701Q7Q` and `https://youtu.be/IhELZ3-wPU0`, respectively.

3.3.5.3 Experiment in 3D urban modelling from street viewpoint

This experiment consists in the camera motion estimation and reconstruction of a 1370-frame sequence acquired with stereo camera S_2. The sequence corresponds to a 1100-meter loop-closing path of the city of Coimbra. This is a very challenging sequence not only because it was acquired with a forward-looking camera - making the building facades highly slanted - but also because it is a very curvy path in a hilly area of Coimbra, as can be seen in Figure 3.18c. The presence of dense vegetation in some areas and moving vehicles and pedestrians further hampers the camera motion estimation and reconstruction processes.

Note on the metrics used to quantify accuracy

Due to the absence of ground truth, the quality of the motion estimation is measured quantitatively by computing the loop closing error in the following way. Let T_i be the estimated camera motion between positions i and $i+1$. For a sequence of F frames, the loop closing error is computed as the relative rotation and translation between the 4×4 identity matrix I_4 and the transformation

$$\mathsf{T}_e = \left(\prod_{i=1}^{F-1} \mathsf{T}_i \right) \mathsf{T}_F, \tag{3.11}$$

where T_F is the pose between position F and position 1. Since matrix multiplication does not have the commutative property, different errors will be obtained if we consider different starting points. Note that varying the starting point translates into a cycling permutation of the product defined in Equation 3.11. Although the norm of the translation component in T_e varies with the starting point, the angle of rotation θ associated with the rotation component R_e does not. From Rodrigues' rotation formula, θ is computed by

$$\theta = \cos^{-1} \left(\frac{1}{2} \left(tr(\mathsf{R}_e) - 1 \right) \right), \tag{3.12}$$

which, as can be seen, only depends on the trace tr of matrix R_e. Since the trace has the cyclic property, meaning that it is invariant under cyclic permutations, the trace of transformation T_e, and consequently of its rotation component R_e, does not vary with the starting point. This explains why the rotation error θ is invariant under different starting points. Thus, it suffices to present information about the translation error as a function of the starting position.

Results

Our algorithm was tested not only using the complete 1370-frame sequence, referred to as D_f, but also using a subsequence D_s which is the result of sampling D_f by considering every fourth frame. This sampling is equivalent to acquiring the sequence with a frame rate of about 2fps, or driving 4 times faster. The complete sequence D_f was acquired at an average speed of 21.7km/h, which means that directly acquiring D_s could have been done by driving at about 78km/h. Refer to the table in Figure 3.18a for more details on the different modalities. Results obtained with LIBVISO2 for sequence D_s are not presented because it diverged in many cases and was not even able to provide an estimation in almost 10% of the relative motions due to the lack of sufficient point correspondences.

Figure 3.18b shows the translation error distributions and rotation errors obtained with our method when using sequences D_f and D_s, before and after the final optimization step, and with LIBVISO2 (which used sequence D_f). Considering the sampled sequence D_s, the camera moved an average of 2.9m and rotated an average of 3.4° between consecutive poses. However, in over 10% of the relative poses the displacement was larger than 3.9m and the rotation was over 8.5°. It can be seen that even without the final optimization step, PPSS outperforms LIBVISO2 in the estimation of the rotation component. The optimization step, which was performed with a sliding window of 6 frames, significantly improved the results. When using the complete sequence D_f, a very small rotation error was obtained, leading to translation errors with a small standard deviation. Remark that despite providing a smaller minimum translation error, LIBVISO2 performs worse than our method as the median error is considerably larger. Since the translation value is affected by errors in the rotation, a large rotation error may lead to small translation errors, depending on the starting position, not meaning that the overall result is better. The superior results of our method can be explained by the fact that this particular sequence is very challenging, presenting many curves and significant variations in height. Due to the low acquisition frame rate, consecutive stereo pairs sometimes present small overlap, preventing LIBVISO2 from extracting enough point correspondences to accurately estimate the camera motion.

In Figure 3.18c a top view of the full 3D model is given, with some areas in a different perspective. Each area is shown for both the sampled (D_s) and the complete (D_f) sequence with optimization, enclosed in a light green and cyan ellipse, respectively. These detailed views show that reducing the number of frames has some negative effects on the quality of both the reconstruction and camera motion estimation. Firstly, it can be seen that the moving vehicle was included in the reconstruction when using D_s but not when using D_f. This can be explained by the fact that it was detected in several frames of D_f, and, since the frontal plane is moving, it is not shared across frames, being discarded. The second example shows that a large plane is missing from the reconstruction with D_s. The reason is that this plane is far from the camera, not being observed from a proper viewpoint in D_s. The last example corresponds to a very challenging part of the path consisting of only faraway planes in a sharp turn. Using a sampled dataset hampers the odometry process, as can be seen by the misalignment of the building and car planes. In conclusion, reconstruction results obtained with D_s are slightly inferior not only due to the decreased quality of the camera motion estimation but also because less details are recovered. However, such a small dataset still provided good results, showing that a plane-based approach can be very superior to point-based ones, being useful in applications that require working with small amounts of data.

3.4 Conclusions

In this chapter we first propose to solve the relative pose estimation problem from plane-primitives by (i) establishing plane correspondences across cameras (RGB-D or passive stereo), and (ii) determining the motion whenever the available planes do not fully constrain the problem [123]. The first issue is efficiently solved by matching triplets of planes using the angles between their normals. Concerning the second issue, it is shown that the undetermined situations can be overcome by either using 2 planes and 1 image point correspondence, or 1 plane and 3 image point correspondences We derive closed-form minimal solutions for these cases and apply them in a hierarchical RANSAC that estimates the relative pose using point matches only when strictly necessary.

This novel relative pose estimation scheme is used in two SfM pipelines that take as input an image sequence either acquired by an RGB-D camera (Section 3.2) or a stereo camera (Section 3.3) and output the camera motion and the 3D planes in the scene. Experimental results show that the proposed pipelines outperform competing methods in cases of low or repetitive texture, variable illumination, high surface slant, dynamic foregrounds and wide baseline.

Chapter 4

Theory and Practice of SfM from Affine Correspondences

An AC, denoted in this chapter by $(\mathbf{x}, \mathbf{y}, \mathsf{A})$, consists in a PC across views plus a 2×2 affine transformation A that maps image points in the neighbourhood of $\mathbf{x}$ into image points around $\mathbf{y}$ (see Figure 4.1). Since an affine map describes well the warp undergone by a local image patch while the camera moves, the concept of AC is broadly used for tracking and/or matching points across views [9, 82], or estimating similarity models as in [94], where local transformations are exploited for geometrical alignment. However, and despite ACs being often readily available, mainstream methods for relative pose estimation, such as [47, 48, 119], use as only input the PCs $(\mathbf{x}, \mathbf{y})$, completely disregarding the information about local affine maps.

This fact has been noticed by previous authors that conducted seminal research in jointly using PCs and affine maps for SfM. Perdoch *et al.* [92] and Riggi *et al.* [106] proposed to use ACs for generating additional PCs and estimate the epipolar geometry. However, these new matches are mere approximations that do not necessarily correspond to correct PCs. Koser [68] studied the relationships between ACs and homographies and advanced a single point method to compute the relative pose between a plane and a camera [69]. Recently, Bentolila and Francos proved that 1 AC puts 3 constraints on the fundamental matrix [12]. Despite these seminal works, neither the theory relating ACs with multi-view geometry is fully understood, nor exist practical algorithms such that ACs can become an effective alternative to PCs in SfM pipelines.

The most obvious benefit in using ACs is that models can be estimated from fewer

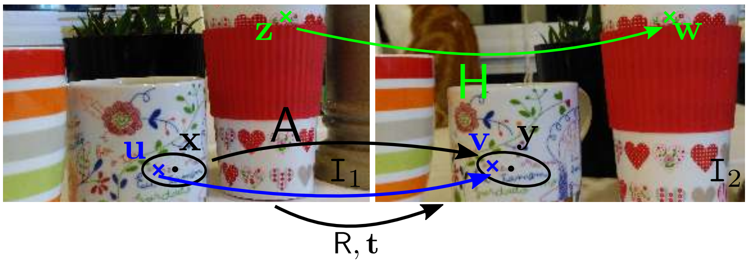


Figure 4.1: Images $\mathtt{I}_1$ and $\mathtt{I}_2$ are provided by two cameras related by $\mathsf{R}, \mathbf{t}$ that observe the same scene. They are used for extracting point $(\mathbf{z}, \mathbf{w})$ and affine $(\mathbf{x}, \mathbf{y}, \mathsf{A})$ correspondences. In this configuration, since the tangent plane to the surface in points $\mathbf{x}$ and $\mathbf{z}$ is the same, both the point match and the AC are compatible with homography H.

correspondences. Thus, and since SfM pipelines invariably comprise an iterative step of sample-and-test (*e.g.* RANSAC [33]), robustness and complexity can dramatically improve by reducing the number of possible combinations, as it happened in the past with the introduction of minimal solvers [88]. These advantages can be specially useful for applications with high combinatorics as it often arises in problems of multi-model fitting or single-model fitting with high percentages of outliers. Examples for the former are applications in plane detection [98] or multibody SfM [112], and for the latter the case of SfM in scenes dominated by deformable surfaces [73].

This chapter starts by deriving both new and known relations for characterizing the family of homographies compatible with an AC in a unified, systematic way (Section 4.1). From the obtained relations, constraints on the epipolar geometry and the homography are derived in Sections 4.2 and 4.3, respectively, leading to new algorithms for recovering the camera's relative pose and for segmenting planes in the scene. In Section 4.4 we make use of the new plane segmentation approach and propose a complete vSLAM pipeline that relies on plane features, named πMatch.

4.1 Geometric relations between ACs and homographies

In this section, theoretical results on the relationship between ACs and homographies are presented. We start by reviewing background concepts and afterwards show how an AC constrains the homography.

Consider the setup of Figure 4.1 where two cameras, related by a rotation R and a translation $\mathbf{t}$, observe a scene, originating images $\mathtt{I}_1$ and $\mathtt{I}_2$. Let $(\mathbf{x}, \mathbf{y}, \mathsf{A})$ be an AC such that the patches surrounding $\mathbf{x}$ and $\mathbf{y}$ are related by a non-singular 2×2

matrix A, with

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^\mathsf{T}, \mathbf{y} = \begin{bmatrix} y_1 & y_2 \end{bmatrix}^\mathsf{T}, \mathsf{A} = \begin{bmatrix} a_1 & a_3 \\ a_2 & a_4 \end{bmatrix}. \tag{4.1}$$

For a point correspondence $(\mathbf{u}, \mathbf{v})$ in the patch, it comes that

$$\mathbf{v} = \mathsf{A}\mathbf{u} + (\mathbf{y} - \mathsf{A}\mathbf{x}). \tag{4.2}$$

Let us also assume that the patches are related by an homography

$$\mathsf{H} \sim \begin{bmatrix} h_1 & h_4 & h_7 \\ h_2 & h_5 & h_8 \\ h_3 & h_6 & h_9 \end{bmatrix}, \tag{4.3}$$

such that in non-homogeneous coordinates

$$\mathbf{v} = \mathbf{f}(\mathbf{u}) = \delta_{\mathbf{u}}^{-1} \begin{bmatrix} h_1u_1{+}h_4u_2{+}h_7 & h_2u_1{+}h_5u_2{+}h_8 \end{bmatrix}^\mathsf{T}, \tag{4.4}$$

where $\delta_{\mathbf{p}} = h_3p_1 + h_6p_2 + h_9$. As first proposed by Koser *et al.* [67, 69], approximating Equation 4.4 using the first-order Taylor expansion around $\mathbf{x}$ yields $\mathbf{v} = \mathbf{f}(\mathbf{x}) + \mathsf{J}_\mathbf{f}(\mathbf{x})(\mathbf{u} - \mathbf{x})$, where $\mathsf{J}_\mathbf{f}$ is the Jacobian of $\mathbf{f}$. Knowing that $\mathbf{f}(\mathbf{x}) = \mathbf{y}$, the expression can be written as

$$\mathbf{v} = \mathsf{J}_\mathbf{f}(\mathbf{x})\mathbf{u} + (\mathbf{y} - \mathsf{J}_\mathbf{f}(\mathbf{x})\mathbf{x}). \tag{4.5}$$

Relating Equations 4.2 and 4.5, it can be seen that $\mathsf{A} = \mathsf{J}_\mathbf{f}(\mathbf{x})$, meaning that the affine transformation A is the Jacobian of the homography defined in point $\mathbf{x}$.

4.1.1 2-parameter family of homographies

The result introduced by Koser *et al.* [67, 69] allows the homography to be defined as a function of the AC. From $\mathbf{y} = \mathbf{f}(\mathbf{x})$, two constraints on the parameters of the homography are obtained. This allows (h_7, h_8) to be written as

$$\begin{bmatrix} h_7 \\ h_8 \end{bmatrix} = \begin{bmatrix} (h_3 - h_1)x_1 + (h_6 - h_4)x_2 + h_9 \\ (h_3 - h_2)x_1 + (h_6 - h_5)x_2 + h_9 \end{bmatrix}. \tag{4.6}$$

Replacing this result in Equation 4.4, the Jacobian becomes

$$\mathsf{J}_{\mathbf{f}}(\mathbf{x}) = \delta_{\mathbf{x}}^{-1} \left[\begin{bmatrix} h_1 & h_4 \\ h_2 & h_5 \end{bmatrix} - \mathbf{y} \begin{bmatrix} h_3 & h_6 \end{bmatrix} \right], \tag{4.7}$$

which, since $\mathsf{J}_{\mathbf{f}}(\mathbf{x}) = \mathsf{A}$, yields that

$$\begin{bmatrix} h_1 & h_4 \\ h_2 & h_5 \end{bmatrix} = \delta_{\mathbf{x}} \mathsf{A} + \mathbf{y} \begin{bmatrix} h_3 & h_6 \end{bmatrix}. \tag{4.8}$$

Replacing the results of Equations 4.6 and 4.8 in the homography 4.3, it comes that the AC $(\mathbf{x}, \mathbf{y}, \mathsf{A})$ induces a two-parameter family of homographies H defined as

$$\mathsf{H} \sim \delta_{\mathbf{x}} \begin{bmatrix} \mathsf{A} & \mathbf{y} - \mathsf{A}\mathbf{x} \\ \mathbf{0} & 1 \end{bmatrix} + \begin{bmatrix} \mathbf{y} \\ 1 \end{bmatrix} \begin{bmatrix} h_3 & h_6 & h_9 - \delta_{\mathbf{x}} \end{bmatrix}. \tag{4.9}$$

Note that this is a two-parameter family since, although there are 3 unknowns, there are only 2 DoFs because H is defined up to scale. Several authors [12, 67, 69] suggest to make $h_9 = 1$ in order to fix the scale factor, which has the drawback of not avoiding singular configurations. H is non-singular whenever A is full rank and $\delta_{\mathbf{x}} \neq 0$. In order to assure that H is always full rank, we introduced the following change of parameters:

$$\begin{bmatrix} g_3 \\ g_6 \\ g_9 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x_1 & x_2 & 1 \end{bmatrix} \begin{bmatrix} h_3 \\ h_6 \\ h_9 \end{bmatrix}, \tag{4.10}$$

which leads to

$$\mathsf{H} \sim g_9 \begin{bmatrix} \mathsf{A} & \mathbf{y} - \mathsf{A}\mathbf{x} \\ \mathbf{0} & 1 \end{bmatrix} + \begin{bmatrix} \mathbf{y} \\ 1 \end{bmatrix} \begin{bmatrix} g_3 & g_6 & -x_1 g_3 - x_2 g_6 \end{bmatrix}. \tag{4.11}$$

Making $g_9 = 1$ fixes the scale factor and avoids singular configurations, yielding

$$\mathsf{H}(\mathbf{g}; \mathbf{x}, \mathbf{y}, \mathsf{A}) = \begin{bmatrix} \mathsf{A} + \mathbf{y}\mathbf{g}^{\mathsf{T}} & \mathbf{y} - (\mathsf{A} + \mathbf{y}\mathbf{g}^{\mathsf{T}})\mathbf{x} \\ \mathbf{g}^{\mathsf{T}} & 1 - \mathbf{g}^{\mathsf{T}}\mathbf{x} \end{bmatrix}, \tag{4.12}$$

with $\mathbf{g} = \begin{bmatrix} g_3 & g_6 \end{bmatrix}^{\mathsf{T}}$.

In case H is a perspectivity, there are 4 solutions that can be determined by solving two second-order equations in $\mathbf{g}$ that force the first and second columns of H to have the same norm and be orthogonal. Note that this generalizes the result by

Koser and Koch [69] that requires the origin of plane coordinates to be coincident with $\mathbf{x}$.

4.2 Epipolar geometry using ACs

It has recently been shown by Bentolila and Francos [12, 13] that one AC yields 3 linear constraints on the terms of the fundamental matrix F. Their method follows a sequence of steps including: (i) coordinate shifting so that the origin is the center of an AC; (ii) estimation of the epipole location $\mathbf{e}_p$ by intersecting 3 conics computed from the 3 ACs; (iii) estimation of two PCs through line intersection for finding an homography H; (iv) computing $\mathsf{F} = [\mathbf{e}_p]_\times \mathsf{H}$. This method does not make direct use of the linear constraints since they were derived only for the AC centred in the origin, requiring many small steps to achieve an estimation of F. Also, it is not clear how it could be adapted to the calibrated case, for the estimation of the essential matrix E.

We propose a new formulation for the estimation of the epipolar geometry from ACs by deriving the linear constraints in the original coordinate system. The new method does not require any of the steps proposed in [12], being much more straightforward and easier to implement. Moreover, it can be applied both to the uncalibrated and calibrated cases. This section concludes with an algorithm for estimating the essential matrix from 2 ACs that is extensively tested against the 5-point method [119] in real sequences. This work provides for the first time convincing experimental evidence that ACs are a viable alternative to PCs for visual odometry and can be highly advantageous in the presence of many outliers as it happens in scenes with multiple moving objects and/or deformable surfaces.

It is known that an homography compatible with a fundamental or an essential matrix verifies the condition $\mathsf{H}^\mathsf{T}\mathsf{T} + \mathsf{T}^\mathsf{T}\mathsf{H} = \mathsf{0}$, for $\mathsf{T} = \mathsf{F}, \mathsf{E}$, respectively. We will derive the constraints for the case of E, with

$$\mathsf{E} = \begin{bmatrix} e_1 & e_4 & e_7 \\ e_2 & e_5 & e_8 \\ e_3 & e_6 & e_9 \end{bmatrix}. \tag{4.13}$$

Consider the 2-parameter family of homographies induced by an AC $(\mathbf{x}, \mathbf{y}, \mathsf{A})$ in Equation 4.12. The matrix equation $\mathsf{H}^\mathsf{T}\mathsf{E} + \mathsf{E}^\mathsf{T}\mathsf{H} = \mathsf{0}$ yields 9 equations, 6 of which are linearly independent and can be written as

$$\underbrace{\begin{bmatrix} q_1 & q_2 & g_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & q_3 & q_4 & g_6 & 0 & 0 & 0 \\ q_3 & q_4 & g_6 & q_1 & q_2 & g_3 & 0 & 0 & 0 \\ q_5 & q_6 & \gamma & 0 & 0 & 0 & q_1 & q_2 & g_3 \\ 0 & 0 & 0 & q_5 & q_6 & \gamma & q_3 & q_4 & g_6 \\ 0 & 0 & 0 & 0 & 0 & 0 & q_5 & q_6 & \gamma \end{bmatrix}}_{\mathsf{N}} \mathbf{e} = \mathbf{0}, \tag{4.14}$$

where γ depends on the unknown $\mathbf{g}$, $\gamma{=}1{-}g_3x_1{-}g_6x_2$, $q_i, i = 1, \ldots, 6$ are defined as $q_1{=}a_1{+}g_3y_1$, $q_2{=}a_2{+}g_3y_2$, $q_3{=}a_3{+}g_6y_1$, $q_4{=}a_4{+}g_6y_2$, $q_5{=}y_1\gamma{-}a_1x_1{-}a_3x_2$ and $q_6{=}y_2\gamma{-}a_2x_1{-}a_4x_2$, and $\mathbf{e}$ is the vectorization of E by columns. Due to the sparse nature of matrix N, it is possible to combine the 6 equations in order to eliminate the two unknowns g_3, g_6. Right-multiplying the system of Equation 4.14 by the following matrix C

$$\mathsf{C}{=}\begin{bmatrix} x_1^2 & x_2^2 & x_1x_2 & x_1 & x_2 & 1 \\ -g_6x_1^2 & -x_2(g_6x_2{-}2) & -x_1(g_6x_1{-}1) & -g_6x_1 & 1{-}g_6x_2 & -g_6 \\ -x_1(g_3x_1{-}2) & -g_3x_2^2 & -x_2(g_3x_1{-}1) & 1{-}g_3x_1 & -g_3x_2 & -g_3 \end{bmatrix} \tag{4.15}$$

yields three equations that only depend on the terms of the AC $(\mathbf{x}, \mathbf{y}, \mathsf{A})$:

$$\begin{bmatrix} x_1y_1 & x_1y_2 & x_1 & x_2y_1 & x_2y_2 & x_2 & y_1 & y_2 & 1 \\ a_3x_1 & a_4x_1 & 0 & y_1{+}a_3x_2 & y_2{+}a_4x_2 & 1 & a_3 & a_4 & 0 \\ y_1{+}a_1x_1 & y_2{+}a_2x_1 & 1 & a_1x_2 & a_2x_2 & 0 & a_1 & a_2 & 0 \end{bmatrix} \mathbf{e}{=}\mathbf{0} \tag{4.16}$$

It can be seen that, as expected, the first equation corresponds to the point match. It is also important to note that the simplicity of matrix N is due to the change of variables that was performed when representing the homography (Equation 4.10). Moreover, solving for $\mathbf{g}$ using the first two equations in system 4.14 and substituting in the third, yields, after some algebraic manipulation, $\det(\mathsf{E}) = 0$.

All the derivations obtained up to this point are valid both for the essential and fundamental matrices. Thus, since one AC provides three linear equations in the form of Equation 4.16, the 7-DoFs matrix F can be determined from 3 ACs and the 5-DoFs matrix E from a minimum of 2 ACs. A total of 9 and 6 equations are obtained in the uncalibrated and calibrated cases, respectively, meaning that either the 8-point or the 7-point solvers [47] can be used in the former case and the 6-

point or 5-point solvers [119] in the latter. Section 4.2.1 presents experiments on the estimation of the essential matrix using 5 PCs and 2 ACs, in both rigid and non-rigid scenarios.

Note that Equation 4.14 can be interpreted the opposite way: suppose we know the epipolar geometry, E or F, and an AC, and wish to find the homography H compatible with both. Rewriting Equation 4.14 for isolating the unknown $\mathbf{g}$, a non-homogeneous system of 6 equations linear in the terms of $\mathbf{g}$ is obtained. Solving for $\mathbf{g}$ and substituting in Equation 4.12, allows for H to be fully determined.

4.2.1 Experiments

In order to assess the validity of using ACs extracted from any kind of scene, an experiment was conducted where the epipolar geometry is estimated in sequences with planes, without planes, and hybrid. Moreover, motivated by the fact that our proposed 2-AC algorithm significantly reduces the combinatorics of the essential matrix estimation problem when compared to the state-of-the-art 5-point algorithm, we evaluated the performance of both methods in the presence of outliers and/or few input matches. Such conditions may occur in scenarios with very low textured surfaces where it is difficult to extract features and/or much deformation caused by the movement of pedestrians, vegetation in the wind, *etc.*. Thus, in the first experiment we injected outliers and decreased the size of the data set, while in the second two real sequences dominated by large deformations were considered.

In all experiments, affine covariant features are extracted with the Hessian Laplace detector [83, 132] using the VLFeat library [136]. The affine part of the ACs, A, is refined by minimizing the photo-geometric error for increasing the estimation accuracy. The maximum number of iterations in the optimization was set to 10 in order to assure fast computation. Point normalization *a la Hartley* is always used before any linear estimation.

A total of 6 equations in the form of Equation 4.16 are obtained from 2 ACs. Our proposed method selects 5 out of the 6 equations for generating up to 10 solutions for E using the solver [119] and the remaining - that must be one corresponding to a point match - for selecting the best solution using the reprojection error. The 5-point algorithm used is the one proposed in [119]. As a final step for both methods, an iterative refinement was performed with the inlier matches.

The first experiment reports results on 3 sequences from [120], where the first only contains planes, the second contains both planar and non-planar objects and the third does not have any planar surfaces (Figure 4.2). The size of the data set was

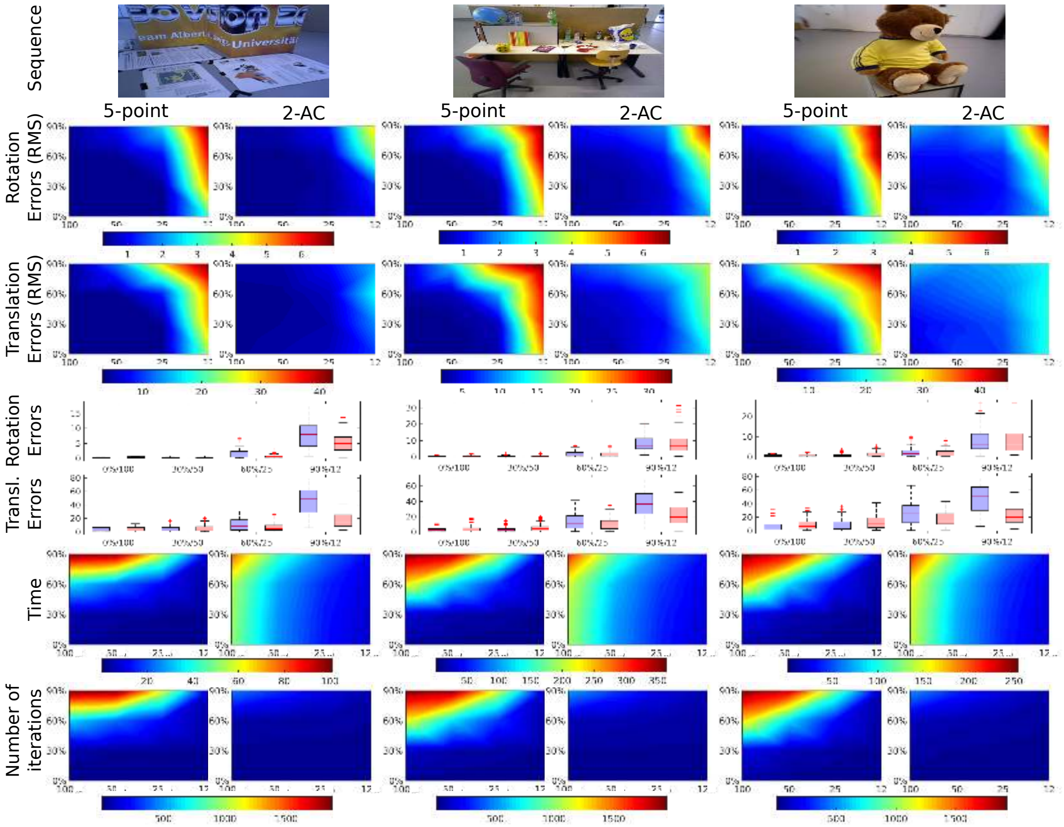


Figure 4.2: Experiment on the estimation of E for 3 sequences of the dataset from [120]. The number of input correspondences was reduced from 100 to 50, 25 and 12, and outliers were injected so that they constituted 0%, 30%, 60%, and 90% of the input set, originating 16 different configurations. For each sequence, color plots of the RMS rotation and translation errors are included - where the colors between the 16 error values were obtained by interpolation, for visualization purposes -, as well as boxplots corresponding to the diagonal configurations for both the 5-point (blue) and the proposed 2-AC (red) algorithms. Computational times for the complete sequences and average number of iterations of MSAC per image pair are shown.

reduced from 100 correspondences down to 12 by randomly sampling data points from the original set, and outliers were injected by adding noise sampled from a uniform distribution of mean 0 and standard deviation 5 pixels. In Figure 4.2, results are shown for our proposed method and for the 5-point algorithm using as input the PCs from the extracted ACs. However, we also performed tests by using as input Scale-Invariant Feature Transform (SIFT) features and 6 PCs extracted from the 2 ACs, as proposed in [12]. These results are not shown because they compare unfavourably to the 5-point and 2-AC algorithms, respectively. For the first case,

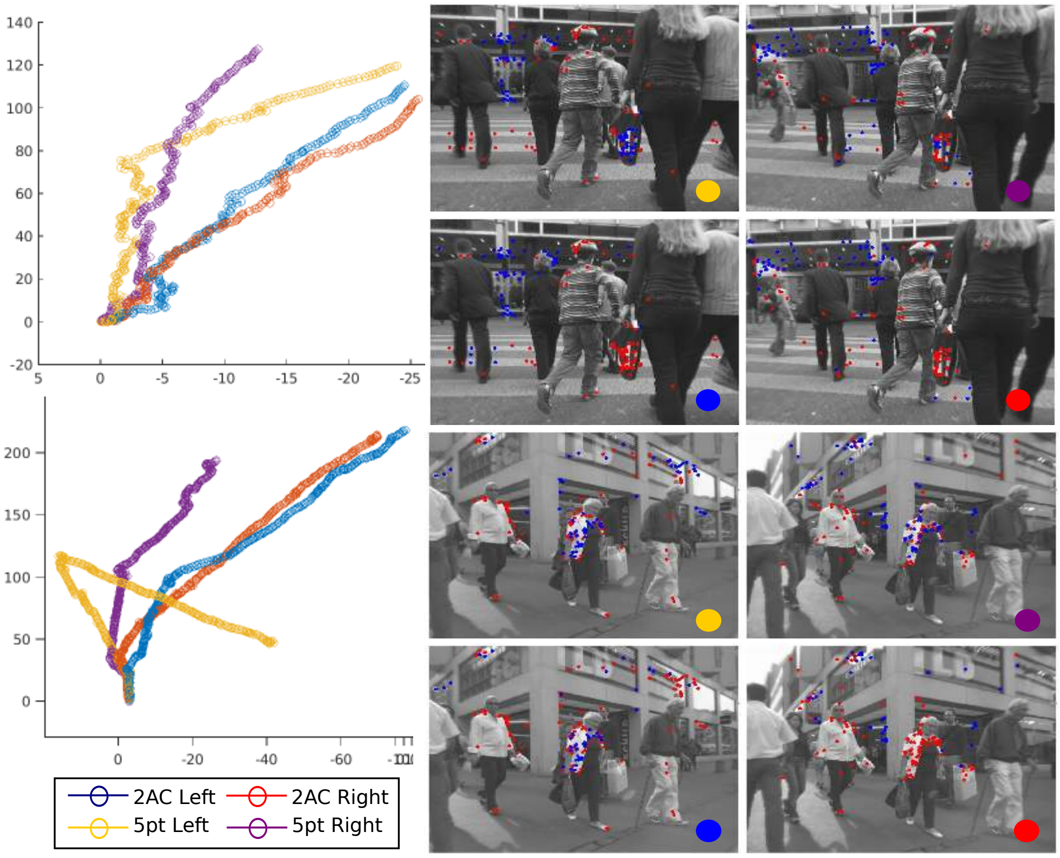


Figure 4.3: Experiment on the estimation of the essential matrix for a 220-image sequence (trajectories on the left) and a 600-image sequence (trajectories on the right) from the dataset presented in [32]. The estimation was performed for the left and right channels of the stereo pairs independently. The trajectories recovered for each channel, for the proposed 2-AC algorithm and the 5-point algorithm are identified with colors. For each sequence, an example showing the inlier (blue) and outlier (red) points is given. The stereo pair channel and used method are identified with a coloured circle.

using SIFT features provides a decrease in computational time of about 20% but originates a decrease in accuracy of approximately 9% in translation and 25% in rotation. When extracting points from the ACs, besides this extra overhead, the average error also increased (3.8% in translation and 19.6% in rotation). From Figure 4.2 it can be seen that the 5-point algorithm is slightly superior when working with large input sets and low percentages of outliers, being, however, similar to the proposed 2-AC method for the first sequence. This implies that the quality of the ACs in this sequence is higher. Another important observation is that our method is significantly more resilient to outliers and small data sets. As an example, the RMS error in translation never exceeds 25° while the 5-point algorithm frequently reaches errors over 60°. Concerning the number of MSAC iterations, it becomes clear that the large decrease in the size of the minimum set of correspondences from 5 to 2 significantly favours a robust estimator. In relation to computational time, the discrepancy is not so evident due to the overhead of AC refinement. However, for large data sets with high percentages of outliers, the 2-AC method still is computationally more efficient.

The last experiment shown in Figure 4.3 compares the performance of the 5-point algorithm with the proposed 2-AC method in two sequences dominated by strong deformations [32]. They were acquired with calibrated stereo cameras, allowing the estimation of E individually for each channel. Results show that the trajectories obtained with our method are much more similar and smoother than the ones obtained with the 5-point algorithm, suggesting the superiority of the 2-AC approach. This can be confirmed by the examples that show how the 5-point method tends to select as inliers matches that belong to pedestrians or moving objects. From the known extrinsic calibration of the stereo cameras, it is possible to compute the estimation error between left and right channels, for each image. It was observed that similar error distributions were obtained for both methods, with the exception that the proposed method yields less outliers in the distributions. For the first sequence, using 2 ACs and 5 PCs originated 6 and 22 outliers, respectively. For the second, our method originated large errors in only 4 frames, while the 5-point algorithm failed 14 times, of which 3 were rotation errors over 30°. It is important to note that small between-channels errors are obtained whenever a method incorrectly chooses inliers in the same moving objects. The examples in Figure 4.3 show that this happens often for the 5-point method, leading to trajectories that are not smooth.

The experiments reported in this section confirm that using 2 ACs as opposed to 5 PCs in the estimation of E brings important benefits when dealing with scenarios where it is difficult to extract valid correspondences.

4.3 Homographies using ACs

This section derives, for the first time, the constraints that must be verified by a PC or an AC to be compatible with the 2-parameter family of homographies associated with an initial AC. These constraints have a clear geometric interpretation and can be used as a metric for segmenting correspondences according to planes present in the scene. Comparative experiments show the benefits of this direct metric with respect to sophisticated global multi-model fitting approaches [71] using the 4-point algorithm for homography estimation [47].

4.3.1 Using PCs to constrain the homography

The two-parameter family of homographies in Equation 4.12 can be further constrained by using PCs. Consider an additional match $(\mathbf{z}, \mathbf{w})$, as depicted in Figure 4.1, which is used to determine the homography $\mathsf{H}(\mathbf{g}; \mathbf{x}, \mathbf{y}, \mathsf{A})$ by estimating $\mathbf{g}$. Although at first $(\mathbf{z}, \mathbf{w})$ may appear to provide two constraints in H that should suffice to uniquely determine $\mathbf{g}$, this is not the case as proven next.

Assume that $(\mathbf{z}, \mathbf{w})$ is compatible with H such that

$$k \begin{bmatrix} \mathbf{w}^\mathsf{T} & 1 \end{bmatrix}^\mathsf{T} = \mathsf{H}(\mathbf{g}; \mathbf{x}, \mathbf{y}, \mathsf{A}) \begin{bmatrix} \mathbf{z}^\mathsf{T} & 1 \end{bmatrix}^\mathsf{T}, \tag{4.17}$$

with k being a scale factor. From Equation 4.12, it comes in a straightforward manner that $k = \mathbf{g}^\mathsf{T}(\mathbf{z} - \mathbf{x}) + 1$, which, by replacing in Equation 4.17, yields

$$(\mathbf{w} - \mathbf{y})(\mathbf{z} - \mathbf{x})^\mathsf{T}\mathbf{g} = \mathsf{A}(\mathbf{z} - \mathbf{x}) - (\mathbf{w} - \mathbf{y}). \tag{4.18}$$

Two important facts arise from Equation 4.18. The first is that since $(\mathbf{w} - \mathbf{y})(\mathbf{z} - \mathbf{x})^\mathsf{T}$ is a rank-1 matrix, it can be concluded that a point match $(\mathbf{z}, \mathbf{w})$ only puts one constraint in $\mathbf{g}$. Thus, two point correspondences are required to fully constrain $\mathsf{H}(\mathbf{g}; \mathbf{x}, \mathbf{y}, \mathsf{A})$. The second fact is that the span S of matrix $(\mathbf{w} - \mathbf{y})(\mathbf{z} - \mathbf{x})^\mathsf{T}$ is 1-dimensional, with $\mathsf{S} = \lambda(\mathbf{w} - \mathbf{y}), \forall \lambda \in \mathbb{R}$, which means that H is compatible with $(\mathbf{z}, \mathbf{w})$ iff $\mathsf{A}(\mathbf{z} - \mathbf{x}) - (\mathbf{w} - \mathbf{y}) \in \mathsf{S}$. In other words, $(\mathbf{z}, \mathbf{w})$ is compatible with the homography H iff the following holds

$$(\mathbf{w} - \mathbf{y})^\mathsf{T}\mathsf{P}\mathsf{A}(\mathbf{z} - \mathbf{x}){=}0, \text{ with } \mathsf{P}{=}\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{4.19}$$

Note that the geometric meaning of Equation 4.19 is that vectors $\mathbf{c}_1 = \mathsf{A}(\mathbf{z} - \mathbf{x})$ and $\mathbf{c}_2 = \begin{bmatrix} w_2 - y_2 & -(w_1 - y_1) \end{bmatrix}^\mathsf{T}$ must be orthogonal. This finding allows using the angle between $\mathbf{c}_1$ and $\mathbf{c}_2$ as an error metric for checking compatibility between a PC and an homography induced by an AC. PCs not verifying Equation 4.19 cannot lie in a plane that contains the 3D point that gives rise to the AC. Although the condition is necessary but not sufficient to assure coplanarity, it can be used as an error metric for plane segmentation or tracking tasks. Section 4.3.3.1 validates the practical usefulness of the metric in a planar segmentation experiment.

4.3.2 Estimating the homography using ACs

Instead of using PCs, an additional AC can be applied to further constrain the homography $\mathsf{H}(\mathbf{g};\mathbf{x},\mathbf{y},\mathsf{A})$. Let $(\mathbf{z},\mathbf{w},\mathsf{B})$ be an extra AC that lies in the same plane as $(\mathbf{x},\mathbf{y},\mathsf{A})$, or that corresponds to the same plane tangent to the surface in the point of correspondence. This implies that there must be a choice of parameters $\mathbf{g},\mathbf{m}$ such that $\mathsf{H}(\mathbf{g};\mathbf{x},\mathbf{y},\mathsf{A}) = k\mathsf{H}(\mathbf{m};\mathbf{z},\mathbf{w},\mathsf{B})$, where k is a scale factor.

Considering the homography as represented in Equation 4.12, the relations $k = 1 + \mathbf{g}^\mathsf{T}(\mathbf{z}-\mathbf{x})$ and $\mathbf{g}=k\mathbf{m}$ are obtained. Replacing in Equation 4.12, and making $\mathsf{M}=\left((\mathbf{z}-\mathbf{x})^\mathsf{T}\mathbf{g}\right)\mathsf{B}$, it comes

$$k\mathsf{H}(\mathbf{m};\mathsf{B},\mathbf{z},\mathbf{w})=\begin{bmatrix}\mathsf{B}+\mathsf{M}+\mathbf{w}\mathbf{g}^\mathsf{T} & \mathbf{w}-\mathsf{B}\mathbf{z}-\mathsf{M}\mathbf{z}-\mathbf{g}^\mathsf{T}\mathbf{x}\mathbf{w}\\ \mathbf{g}^\mathsf{T} & 1-\mathbf{g}^\mathsf{T}\mathbf{x}\end{bmatrix}. \tag{4.20}$$

Since it is known that $\mathsf{H}(\mathbf{g};\mathbf{x},\mathbf{y},\mathsf{A})=k\mathsf{H}(\mathbf{m};\mathbf{z},\mathbf{w},\mathsf{B})$, the following system of six equations is obtained

$$\begin{array}{lcl}\mathsf{A}-\mathsf{B}-(\mathbf{w}-\mathbf{y})\mathbf{g}^\mathsf{T}-\mathsf{M} & = & 0\\ \mathbf{y}-\mathsf{A}\mathbf{x}-(\mathbf{w}-\mathsf{B}\mathbf{z})+(\mathbf{w}-\mathbf{y})\mathbf{x}^\mathsf{T}\mathbf{g}+\mathsf{M}\mathbf{z} & = & \mathbf{0}\end{array}. \tag{4.21}$$

This allows the computation of the 2 unknown terms of the homography, $\mathbf{g}$, from linear least squares. Therefore, replacing in Equation 4.12 or 4.20, H becomes fully determined.

As previously observed, each AC yields 6 constraints on the parameters of the homography H. Thus, two distinct ACs yield 12 restrictions, allowing 4 constraints to be written in the terms of $(\mathbf{x},\mathbf{y},\mathsf{A})$ and $(\mathbf{z},\mathbf{w},\mathsf{B})$ because the homography has only 8 DoFs. Using the first matrix equation of system 4.21 to substitute M and $(\mathbf{w}-\mathbf{y})\mathbf{g}^\mathsf{T}$ in the second, it yields two conditions as the one in Equation 4.19 in the terms of A and B, respectively. This is expected since, by construction, the point match of one AC is compatible with the homography induced by the other. The remaining two constraints can be obtained by determining $\mathbf{g}$ from *e.g.* the first two equations in the first matrix equation and replacing the solution in the last two. After some algebraic manipulation, this procedure yields the last matrix equation in System 4.22. Thus, the 4 conditions for $(\mathbf{x},\mathbf{y},\mathsf{A})$ and $(\mathbf{z},\mathbf{w},\mathsf{B})$ to be compatible with the same homography are

$$\begin{gathered}(\mathbf{w}-\mathbf{y})^{\mathsf{T}}\mathsf{P}\mathsf{A}(\mathbf{z}-\mathbf{x})=0\\(\mathbf{w}-\mathbf{y})^{\mathsf{T}}\mathsf{P}\mathsf{B}(\mathbf{z}-\mathbf{x})=0\\\underbrace{\begin{bmatrix}s{+}a_2b_3{-}a_3b_2 & -(a_1b_3{-}a_3b_1)\\a_2b_4{-}a_4b_2 & s{-}(a_1b_4{-}a_4b_1)\end{bmatrix}}_{\substack{\mathsf{L}\\s=\frac{[-a_2+b_2\;\;a_1-b_1](\mathbf{w}-\mathbf{y})-(a_1b_2-a_2b_1)(x_1-z_1)}{(x_2-z_2)}}}(\mathbf{w}{-}\mathbf{y})=\mathbf{0},\ \text{with}\ .\end{gathered}\tag{4.22}$$

Note that, as reasoned for Equation 4.19, the last matrix constraint means that both vectors $\mathbf{l}_1^{\mathsf{T}}$ and $\mathbf{l}_2^{\mathsf{T}}$, corresponding to the first and second rows of the 2×2 matrix L, respectively, must be orthogonal to $(\mathbf{w}{-}\mathbf{y})$. Thus, in this case, there are 4 different angles that can be combined to provide an error metric of compatibility between two ACs and an homography. Since more information is being included, this metric should be more robust than the one computed solely from PCs. Experiments in Section 4.3.3.1 on planar segmentation confirm this hypothesis.

4.3.3 Experiments in homography estimation

This section reports two experiments for testing the validity and usefulness of the theoretical results on homographies. The first one consists in performing planar segmentation on images that contain between 3 and 5 planes, both by formulating the problem as an UFL[1] problem that can be solved using message passing [71] and by using the novel error metrics proposed in Sections 4.3.1 and 4.3.2. The second experiment assesses the accuracy of homography estimation using 1 AC plus 2 PCs and 2 ACs and compares it with the 4-point linear algorithm applied in an MSAC-framework [130].

4.3.3.1 Segmentation of PCs and ACs

This experiment consists in the planar segmentation of all visible planes in 30 randomly sampled pairs of images from a sequence of the RGB-D dataset [120]. Ground truth segmentation was obtained using the method proposed in [125]. For each plane in an image pair, a reference AC was chosen by selecting the one that yielded the smallest photo-consistency error from the set of ACs belonging to that plane. Since multiple planes are being segmented simultaneously, this task is a multi-model fit-

[1] A global formulation (UFL) is much more effective for multi-modal fitting than greedy generalizations of RANSAC such as [147].

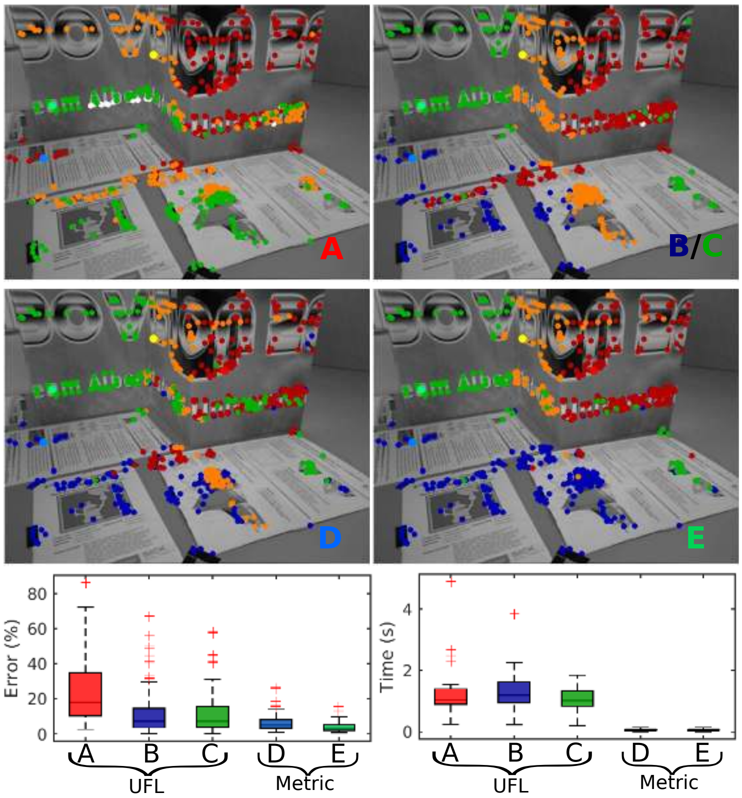


Figure 4.4: Experiment on planar segmentation using 5 different methods: A: 4 points UFL, B: 1 AC + 2 points UFL, C: 2 AC UFL, D: 1 AC + 2 Metric, E: 2 AC Metric. Segmentation errors and computation times are shown for each method. Planes are identified with colors and the reference AC is depicted with a lighter shade of the same color.

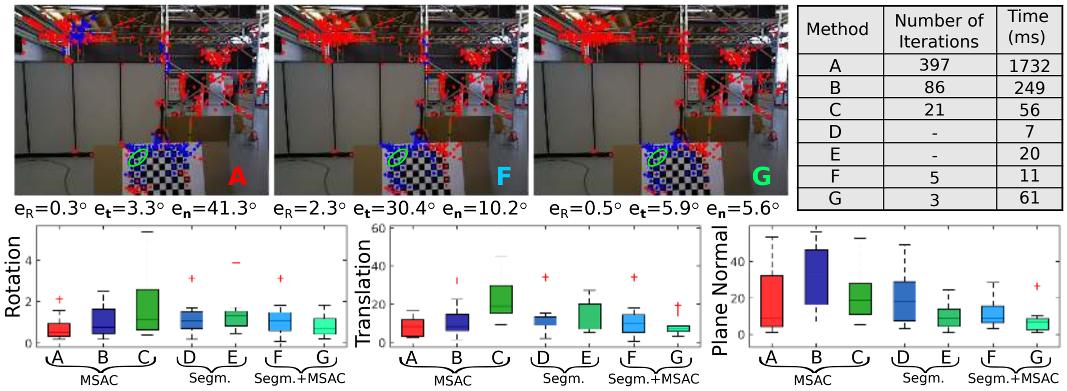
ting problem that can be cast as an UFL problem [71]. The goal is to assign each correspondence (client) to an homography hypothesis (facility), while simultaneously using as few hypotheses as possible. Our data cost matrix is created using the symmetric transfer error (STE) [47], and homography hypotheses are generated using the three minimal sets of 4 PCs, 1 AC plus 2 points (Section 4.3.1) and 2 ACs (Section 4.3.2). These three methods are referred to as A, B and C in Figure 4.4, respectively. Each homography hypothesis is generated by using the reference AC plus 3 PCs, 2 PCs or 1 AC randomly selected, for methods A, B and C, respectively. For each plane, 50 hypotheses were generated from this procedure, yielding a total of 50 × no. of planes + 1 labels per image pair due to the inclusion of the discard label.

An alternative way of performing planar segmentation is by using the constraints derived in Sections 4.3.1 and 4.3.2 that must be verified for an homography induced by an AC to be compatible with a point match and another AC, respectively (methods D and E in Figure 4.4). In this case, explicit estimation of the homography is not required as only the constraint is used. For the case of PCs, it consists in two vectors $\mathbf{a}$ and $\mathbf{b}$ that must be orthogonal. Thus, for each point match the error is simply $e = 90 - \angle(\mathbf{a}, \mathbf{b})$. When working with ACs, a stronger error metric may be used. In this case, 4 angular constraints were derived, yielding 4 errors e which are combined by taking their weighted mean that accounts for the quality of the ACs, computed from photo-consistency. In both cases, the constraints are computed between the reference AC and all remaining correspondences. Labelling is performed by assigning correspondences that yield errors below a pre-defined threshold ($1°$ in our experiments) to the plane and if a correspondence is assigned more that one label, the one that yielded the smallest error is chosen.

Segmentation errors and computation times are shown for each method in Figure 4.4. The first conclusion is that the proposed error metrics effectively segment all the planes in the scene, being much more accurate and faster than the more sophisticated UFL approach. Moreover, when formulating the segmentation task as a UFL problem, it can be seen that it is significantly better to use either the 1 AC plus 2 PCs or the 2 ACs minimal solutions than the 4-point algorithm. This is expected for two main reasons. The first is that the former solutions require less correspondences to be selected, increasing the probability of all correspondences being in the same plane. The second is that as an AC imposes 6 restrictions on the homography as opposed to 2 for a point match in method A, it is more likely that correct solutions will be chosen for methods B and C.

4.3.3.2 Structure-from-Motion

The homography associated to the checkerboard plane in Figure 4.5 is estimated for 12 different image pairs from dataset [120]. For each one, a reference AC on the plane was chosen as described in the previous experiment. The homography H is estimated using 7 different methods, 5 of which rely on the robust estimator MSAC [130] for selecting the inlier correspondences. Methods A, B and C in Figure 4.5 consist on using the reference AC and randomly selecting the rest of the required correspondences for estimating H from 4 points [47], 1 AC plus 2 points and 2 ACs, respectively. Methods F and G correspond to methods B and C with a prior planar segmentation using PCs and ACs, respectively, with the proposed metric. Finally, methods D and E are the simplest since they perform planar segmentation and



Method	Number of Iterations	Time (ms)
A	397	1732
B	86	249
C	21	56
D	-	7
E	-	20
F	5	11
G	3	61

Figure 4.5: Results of homography estimation for a 12-image dataset from [120] using 7 different methods: A: 4 points MSAC, B: 1 AC + 2 points MSAC, C: 2 AC MSAC, D: 1 AC + 2 points Planar segmentation, E: 2 AC Planar segmentation, F: 1 AC + 2 points Planar segmentation & MSAC, G: 2 AC Planar segmentation & MSAC. Coloured boxplots were used for better visualization, where the 4-point method corresponds to red and the methods using 1 AC + 2 points and 2 ACs are shown in shades of blue and green, respectively. Rotation, translation and plane normal errors are given in degrees. For the last two, the error is the angle between the estimated and ground truth vectors. The table shows the average number of iterations and computation time of a pair of images. Inlier and outlier points and the reference AC are shown in blue, red and green.

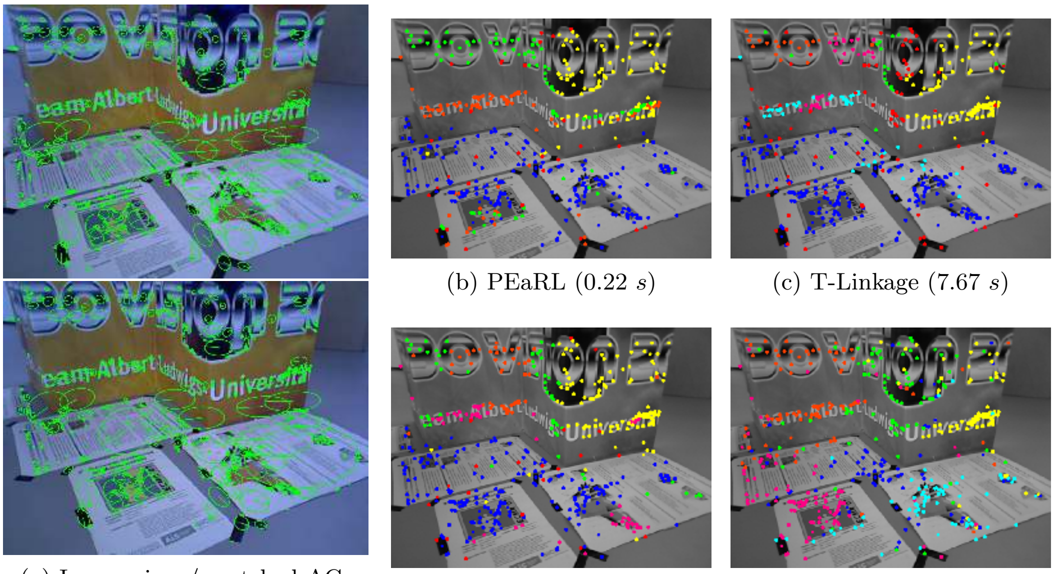
estimate H from the inlier correspondences.

The estimated homography is decomposed into a rotation, a translation known up to a scale factor and the plane normal. The test images contain ground truth rotation and translation. The ground truth plane equation is computed using the method presented in [125]. Presenting information on the quality of the plane normal estimation is relevant since, as observed in Figure 4.5, there are sets of matches that may provide homographies which originate small errors in rotation and translation but estimate the plane normal very poorly. This is very evident for method A both due to the combinatorics of the problem and because the reference AC only puts two constraints on H in this case, meaning that there are homographies that do not correspond to real scene planes that may originate minima in the cost function of MSAC.

The results in Figure 4.5 also show that method E performs better than method D, which is coherent with the results from the previous experiment. When combining the segmentation with an MSAC, a significant increase in the accuracy is obtained, being this the best choice of methods for the task of estimating H. Finally, the table shows that as the minimum required number of matches decreases, less iterations of MSAC are used, occurring an almost immediate convergence when applying a prior segmentation step.

4.4 πMatch: monocular vSLAM and PPR using fast plane correspondences

Monocular vSLAM is the process of estimating the camera position and orientation while building 3D maps of the environment, from a single camera. Although there has been intensive research on this topic, current methods still face several challenges and difficulties, including (i) presence of outliers, (ii) dynamic foregrounds and pure rotation of the camera, (iii) large baselines, (iv) scale drift, (v) density of 3D reconstruction, and (vi) computational efficiency. Nowadays, existing methods for monocular vSLAM follow two distinct approaches: feature extraction and direct image alignment. Each paradigm is effective in solving some of these challenges but, to the best of our knowledge, there is no monocular vSLAM algorithm that is able to tackle all these issues. While feature-based methods work on top of extracted features and are usually robust to outliers by applying RANSAC-based schemes [44], direct methods perform whole image alignment and cannot handle outliers [23, 30, 86]. Moreover, the former work with wide baselines and provide sparse reconstructions, as opposed to the latter that require small baselines, which is typically accomplished by high frame rates that tend to limit image resolution, and provide dense scene models. All feature-based methods [24, 65] perform tracking and mapping as separate tasks. This greatly reduces the complexity of the problem, allowing them to work in real-time. On the other hand, direct methods such as [86, 95] compute dense depth maps using variational approaches which are computationally expensive and require powerful GPUs to achieve real-time performance. Only recently, direct methods that estimate semi-dense depth maps have been proposed [23, 30, 31], allowing real-time operation on a CPU. Most feature-based monocular methods assume there is significant camera translation and that the scene is mainly rigid for applying epipolar geometry. However, there might be situations where this does not hold and a scheme to robustly estimate the camera motion is desirable. Both direct and non-direct methods perform poorly in the presence multiple motions and tend to drift in scale. While there is no explicit solution for the first problem in the state-of-the-art, the last issue is typically solved using prior information such as the height of the camera [44, 117] or the existence of loop



(a) Img. pair w/ matched ACs

(b) PEaRL (0.22 s)

(c) T-Linkage (7.67 s)

(d) PEaRL (4.36 s)

(e) Affinity Prop. (0.20 s)

Figure 4.6: Plane segmentation problem solved using different methods: (b) PEaRL [58] with 300 homography hypotheses, (c) T-linkage [75] with 300 hypotheses, (d) PEaRL with 5000 hypotheses, and (e) affinity propagation [36]. The computational times of PEaRL and T-linkage hamper real-time performance. On the contrary, affinity propagation is fast and is able to detect all the planes present in the image. Red points correspond to outliers.

closures for performing global optimization [30].

Performing PPR in monocular sequences has never been much explored due to the difficulties in detecting planes without knowing the camera motion. One possibility would be to use an hypothesize-and-test framework, such as RANSAC, to fit homographies, but this lacks robustness and is time consuming [58]. Other greedy methods such as J-linkage [129] or its continuous relaxation T-linkage [75] could also be used but they still suffer from low computational efficiency (Figure 4.6c). An alternative would be to use discrete optimization to replace greedy methods by a global scheme such as PEaRL [58] but, although there are improvements, the results are still not satisfactory (Figs 4.6b and 4.6d).

In the previous section, we defined an error metric that allows to quickly segment ACs in planes, without the need to generate homography hypotheses as in hypothesize-and-test approaches. In this section, we build on this result and propose a complete vSLAM pipeline that relies on plane features, named πMatch. ACs are extracted and quickly clustered into coplanar regions using affinity propagation [36] (Figure 4.6e) based on the new metric. For each plane cluster, a fast, robust scheme estimates the corresponding homography, which is decomposed into two solutions for rotation R and translation $\mathbf{t}$ (Section 4.4.1.1). The obtained motion hypotheses are used as input to a PEaRL formulation that merges close motions and decides about multiple motion situations (e.g. dynamic foreground, pure rotation of camera, *etc.*) (Section 4.4.1.2). Given the refined camera motion, the initial plane hypotheses are also merged and refined in a PEaRL framework, and, as an option, used as input to a standard MRF formulation [5, 40] for dense pixel labeling and subsequent PPR (Section 4.4.1.3). This two-view pipeline is applied to each image pair, providing camera motion estimations up to scale. As a final step, we use a fast scheme for scale estimation based on the minimization of the reprojection error that benefits from the high accuracy in the estimation of R and $\mathbf{t}$. This is followed by a discrete optimization step for improving the final PPR of the scene (Section 4.4.3). πMatch makes considerable advances in handling the aforementioned difficulties, being advantageous with respect to the state-of-the-art methods (Table 4.1).

4.4.1 Two-view SfM and PPR using πMatch

We propose πMatch, an SfM framework that is able to automatically recover the camera motion and a PPR of the scene from a monocular sequence. For each image pair, ACs are extracted and used for computing the error metric of compatibility between two ACs and an homography proposed in Section 4.3.2. These measures of similarity between pairs of ACs are used for segmenting planes by Affinity Propagation (AP) [36]. A robust MSAC scheme [130] is then applied to each cluster for filtering out outliers. This step provides a plane segmentation and a set of motion hypotheses, from which the ones present in the image pair are selected in a PEaRL [58] framework. The dominant one, which is assumed to be the camera motion, is identified and refined. Another PEaRL step is applied for plane merging and refinement, and a final standard MRF [5, 40] can be used for dense pixel-labeling. Figure 4.7 shows the sequence of steps of the proposed pipeline. The next subsections detail each building block using the image pair of Figure 4.6 as an illustrative example.

4.4.1.1 Generation of motion hypotheses

For each pair of ACs, we compute the error metric proposed in Section 4.3.2 by taking the average of the errors obtained for each condition, which are the values of the

Table 4.1: Advantages of the proposed method πMatch over existing feature-based and direct methods.

	Feature-based	Direct	πMatch
Robust to outliers	+	-	+
Dynamic foreground/pure rotation	-	-	+
Wide baselines	+	-	+
Scale drift problem	*Camera height*	*Loop closure*	*No priors*
Model density	-	+	+
Computational efficiency	*Real-time*	*Parallelizable*	*Near real-time*

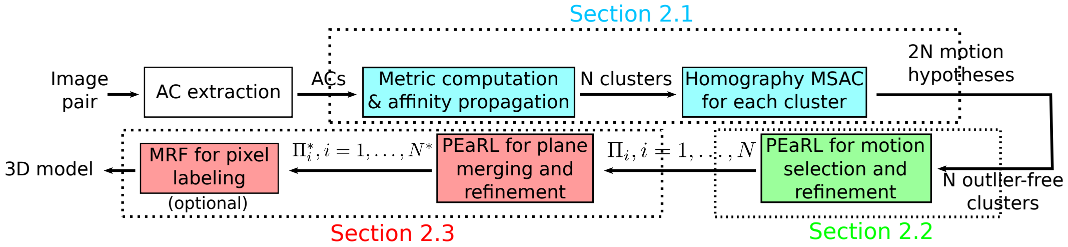
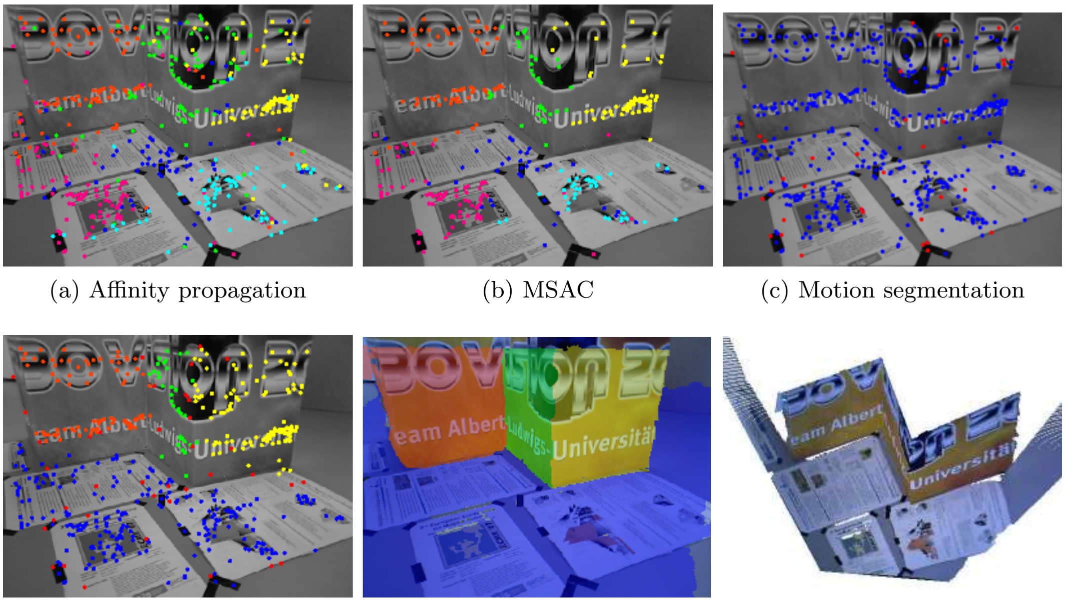

Section 2.1
Image pair
AC extraction
ACs
Metric computation & affinity propagation
N clusters
Homography MSAC for each cluster
2N motion hypotheses
N outlier-free clusters
PEaRL for motion selection and refinement
$\Pi_i, i = 1, \ldots, N$
PEaRL for plane merging and refinement
$\Pi_i^*, i = 1, \ldots, N^*$
MRF for pixel labeling
(optional)
3D model
Section 2.3
Section 2.2

Figure 4.7: Pipeline of the proposed method πMatch for two-view SfM and PPR. ACs are used for segmenting the scene into planes and providing motion hypotheses. The existing motions are selected in a PEaRL framework, and the dominant one is identified and refined. Plane hypotheses are generated from the clustering result and the refined motion, and PEaRL is again used for plane segmentation and refinement. A standard MRF scheme can be used for dense pixel-labeling.

expression on the left-hand side of each equation in the system of equations 4.22. For C ACs, this results in a $C \times C$ matrix of similarities between pairs of ACs, which is fed to an AP method [36] for clustering the ACs into scene planes. Since all data points are assigned to a cluster, the obtained segmentation contains outliers. Moreover, there are cases in which AP tends to oversegment, providing several clusters that correspond to the same scene plane. This is shown in Figure 4.8a, where the ground plane is segmented into 3 different clusters and there are data points incorrectly labeled.

In order to filter out outliers, each cluster is used as input to a MSAC framework for homography estimation. We consider the minimal set of 2 ACs for generating homography hypotheses as proposed in Section 4.3.2, which provides a speed-up of approximately 3× when compared using a 4-point minimal set of point correspondences. Also, since each MSAC is performed for each cluster independently, they can run in parallel, significantly speeding up the process.

The output of this MSAC step is a set of homographies and corresponding outlier-



(a) Affinity propagation

(b) MSAC

(c) Motion segmentation

(d) Plane merging

(e) MRF and 3D model

Figure 4.8: Results for the image pair in Figure 4.6a after each step of the pipeline. (a) AP tends to oversegment and does not identify outliers. (b) A robust scheme is required for filtering each cluster. (c) The best motion hypothesis is selected and (d) plane segmentation is performed, where the original plane hypotheses are merged. (e) πMatch provides an accurate 3D model of the scene from only two views. Colors across images identify planes. Outliers are shown in red.

free clusters (Figure 4.8b). Decomposing each homography yields two solutions for the camera rotation R, translation $\mathbf{t}$, and plane $\mathbf{n}$, up to scale [74].

4.4.1.2 PEaRL for motion selection

Cases in which the camera motion is a pure rotation must be correctly identified since nor the scene planes neither the scale of translation can be recovered, and schemes must be devised to overcome this problem (Section 4.4.3). The previous step of the pipeline outputs N outlier-free clusters and $2N$ motion hypotheses. Firstly, the motions that correspond to pure rotations are identified. This is done by considering the corresponding homography H and computing the distance between the identity matrix I and the matrix $\mathsf{H}\mathsf{H}^\mathsf{T}$. We opted to use metric Φ_4 proposed in [57] for computing this distance as it is the most computationally efficient. Homographies for which this distance lies below a pre-defined threshold are decomposed and only the rotation component is considered by setting $\mathbf{t} = \mathbf{0}$.

There may be more than one motion present in the image due to moving objects in the foreground. In case these objects are planar, they will be identified by the plane segmentation step of the pipeline. Thus, a scheme to decide which planes correspond to rigid structures is required. We propose to solve this problem by selecting the motions present in the image in a PEaRL framework, and afterwards identifying the camera motion. The motion selection task can be cast as a labeling problem where the nodes of the graph are the point correspondences $\mathbf{p}$, to which a label $l_{\mathbf{p}}$ must be assigned. The label set $\mathcal{L} = \{\{\mathcal{R}_0, \mathcal{T}_0\}, l_\emptyset\}$ consists of the set of motion hypotheses $\{\mathcal{R}_0, \mathcal{T}_0\}$ and the discard label $l_\emptyset$. This labeling problem is solved by minimizing an energy function E defined by

$$E(\mathbf{l}) = \underbrace{\sum_{\mathbf{p}} D_{\mathbf{p}}(l_{\mathbf{p}})}_{\text{Data term}} + \underbrace{\lambda_S \sum_{(\mathbf{p},\mathbf{q}) \in \mathcal{N}} w_{\mathbf{pq}} \delta(l_{\mathbf{p}} \neq l_{\mathbf{q}})}_{\text{Smoothness term}} + \underbrace{\lambda_L |\mathcal{L}_{\mathbf{l}}|}_{\text{Label term}} , \tag{4.23}$$

where λ_S and λ_L are weighting constants, $\mathbf{l}$ is the labeling being analysed, $\mathcal{N}$ is the neighbourhood of $\mathbf{p}$, weights $w_{\mathbf{pq}}$ set penalties for each pair of neighbouring data points $\mathbf{p}, \mathbf{q}$, and δ is 1 whenever the condition inside the parentheses is satisfied and 0 otherwise. The data term $D_{\mathbf{p}}$ is defined as the STE if the label corresponds to a pure rotation and the Sampson distance [47] otherwise. Two nodes $\mathbf{p}$ and $\mathbf{q}$ are neighbours if they belong to the same cluster from the set of clusters provided by the MSAC step (Section 4.4.1.1). We set $w_{\mathbf{pq}} = 1$, meaning that an equal penalty is set to all neighbours. This definition of neighbourhood forces points belonging to the same scene plane to be assigned the same motion label. Finally, the label term forces the algorithm to use as few motion hypotheses as possible. Due to the small size of the label set (typically 8-14 motion hypotheses), this discrete optimization step is very fast. Figure 4.8c shows that the algorithm selected only one motion and some points were assigned the discard label $l_\emptyset$. If more than one motion is chosen, the one to which more clusters are associated is selected as the camera motion. In case this is satisfied by more than one hypothesis, the one to which more points were assigned is considered. The camera motion is finally refined with the selected inliers in a standard bundle adjustment with point correspondences.

4.4.1.3 Plane refinement and PPR

Having the refined camera motion, the final step of the two-view pipeline is to merge and refine the initial plane hypotheses obtained from the AP step. This can only be done if the camera motion is not a pure rotation. Otherwise, the algorithm stops.

For each cluster associated to the camera motion, a plane hypothesis is generated by reconstructing its points and finding the 3D plane that best fits the point cloud by linear least squares. From the set of plane hypotheses $\mathcal{P}_0$, the objective is to find the minimum number of planes that best describes the scene. Similarly to Section 4.4.1.2, this task can be cast as a labeling problem where the goal is to assign each point $\mathbf{p}$ to a label from the label set $\mathcal{P} = \{\mathcal{P}_0, l_\emptyset\}$. Again, this is solved by minimizing an energy function E defined as in Equation 4.23, where the data cost is the STE obtained for the homographies computed using the refined camera motion and the plane hypotheses. In this case, our set of neighbours $(\mathbf{p}, \mathbf{q}) \in \mathcal{N}$ is determined by a Delaunay triangulation of points to account for possible small errors in the initial plane segmentation. The weights $w_{\mathbf{pq}}$ are defined as the inverse distance between points $\mathbf{p}$ and $\mathbf{q}$ because closer points are more likely to belong to the same plane. Figure 4.8d shows that the three initial plane hypotheses belonging to the ground plane were correctly merged into one plane, allowing its proper estimation. Also, some incorrectly labeled points in the MSAC stage (Figure 4.8b) were now assigned the correct label. Each selected plane is then refined in an optimization scheme that minimizes the STE. Since each plane is refined independently, this procedure can be performed in parallel, providing a significant speed-up. As a final step, a dense pixel labeling can be obtained using a standard MRF formulation [5,40]. Figure 4.8e shows that the proposed method is able to provide an accurate and visually pleasing dense PPR from only two views.

4.4.2 Two-view experimental results

In this section, we apply the proposed two-view pipeline to 4 different example scenarios (Figure 4.9) and show the obtained results after each step. The first three examples were selected from the KITTI dataset [42,43] and illustrate cases of normal motion, dominant dynamic foreground caused by a large vehicle moving, and static camera. The last example is the situation of a moving camera observing multiple planar motions, and was selected from the Hopkins dataset [131].

In all experiments, affine covariant features were extracted with the Difference of Gaussians operator using the VLFeat library [136]. We limit the number of extracted ACs to approximately 500 for computational efficiency. We used the publicly available implementations of AP [35] and graph cut optimization [137] for the PEaRL steps. The estimations of the relative rotation R and translation $\mathbf{t}$ up to scale are compared with the ground truth R_{GT} and $\mathbf{t}_{GT}$. The error in rotation (e_{R}) is quantified by the angular magnitude of the residual rotation $\mathsf{R}^{\mathsf{T}}\mathsf{R}_{GT}$ and the

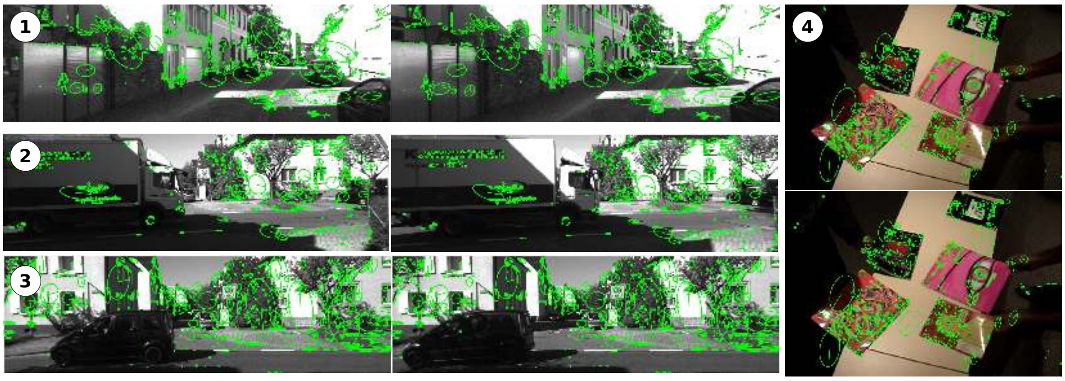
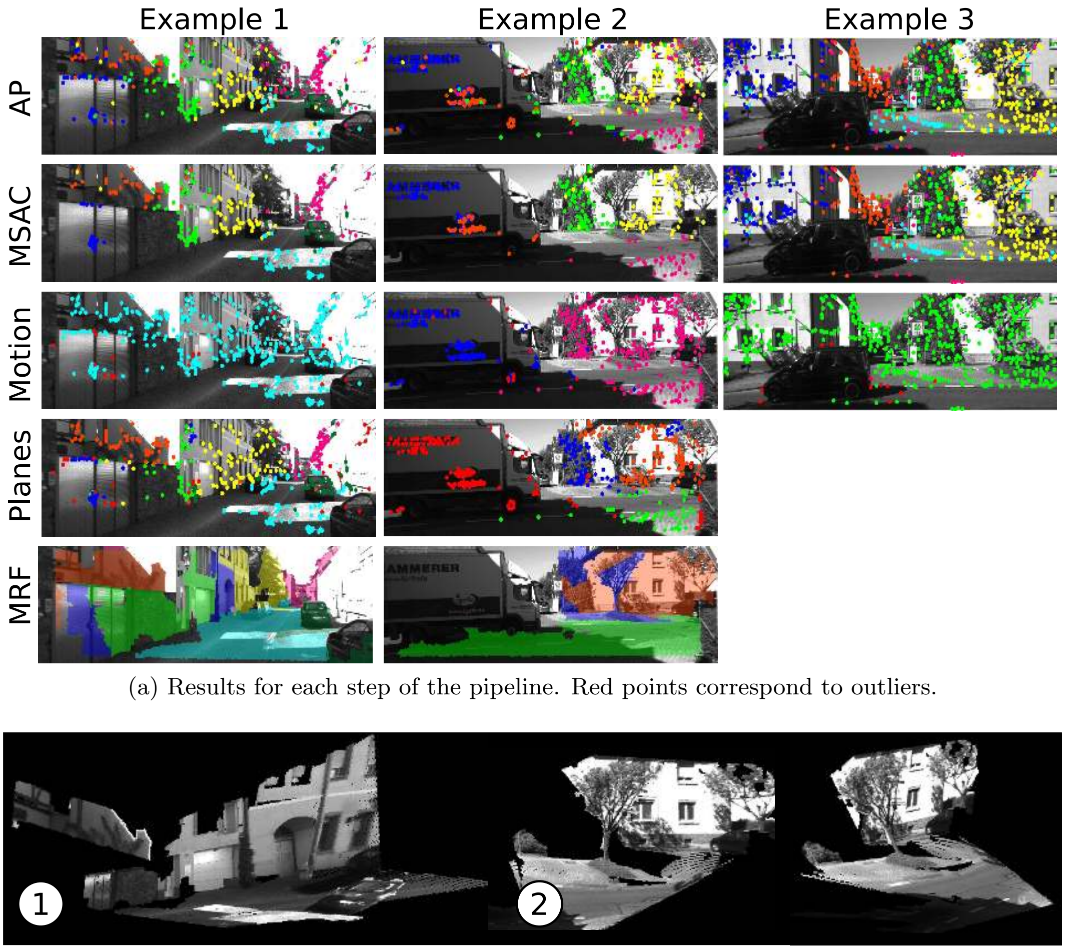


Figure 4.9: Image pairs with the extracted ACs for 4 different scenarios: 1 - normal motion, 2 - dominant dynamic foreground, 3 - static camera/pure rotation, and 4 - four motions besides the camera motion. Examples 1 to 3 are from the KITTI dataset [42,43] and Example 4 is from the Hopkins dataset [131]. πMatch is applied to each scenario and the results are shown in Figures 4.10 and 4.11.

error in translation $e_{\mathbf{t}}$ is defined as the angle between vectors $\mathbf{t}$ and $\mathbf{t}_{GT}$. In all experiments, red points correspond to outliers.

Figure 4.10 shows the outcome of each step of the pipeline for the KITTI image pairs. The first example corresponds to the most common scenario of a moving camera and static scene. AP initially segmented the scene into 7 clusters which were then merged into 6 clusters corresponding to different scene planes. Not only the larger planes corresponding to the ground and building façade were recovered, but also the smaller orange plane was accurately estimated, as shown in the 3D model of Figure 4.10b. Moreover, the camera motion was accurately estimated: $e_{\mathsf{R}} = 10e{-}3^\circ, e_{\mathbf{t}} = 1.2^\circ$.

Example 2 illustrates the case of dominant dynamic foreground, where AP detects 5 different clusters, 2 of which correspond to the moving vehicle. The PEaRL step described in Section 4.4.1.2 correctly detects two motions in the image, where the one to which more clusters are associated is selected as the camera motion. After refinement with the inliers (magenta points in the third row) the rotation and translation errors are $e_{\mathsf{R}} = 14e{-}3^\circ$ and $e_{\mathbf{t}} = 0.98^\circ$, respectively. 3 planes are then segmented in the image, with the remaining points being labeled as outliers. A final 3D model of the scene is shown in Figure 4.10b, where it can be seen that even the faraway plane corresponding to building façade is accurately estimated. VISO2-Mono uses a scheme for detecting if the camera motion is too small, providing the identity matrix as the result for the camera motion in those cases. For this image pair, although the true translation has a norm of $||\mathbf{t}_{GT}|| = 46.4\ cm$, VISO2-



(a) Results for each step of the pipeline. Red points correspond to outliers.

(b) Final 3D model

Figure 4.10: (a) Results obtained after each step of the proposed pipeline for the first 3 scenarios in Figure 4.9. Since scenario 3 corresponds to a static camera, the planes cannot be estimated and thus the last two steps are not performed. (b) PPR obtained for scenarios 1 and 2.

Mono identified this case as small motion and did not provide an estimation. By increasing the threshold, we forced VISO2-Mono to estimate the camera motion and observed that it selected many points on the moving vehicle as inliers, providing a poor estimation of the camera motion: $e_{\mathsf{R}} = 1.99°$, $e_{\mathbf{t}} = 77.9°$ and $||\mathbf{t}|| = 11.1\ m$.

The third example corresponds to the case of static camera. In fact, there is a residual rotation which allows the scene to be correctly segmented into planes and the camera rotation to be accurately estimated ($e_{\mathsf{R}} = 40e{-}4°$). However, since the translation component is negligible, it is not possible to estimate the scene planes.

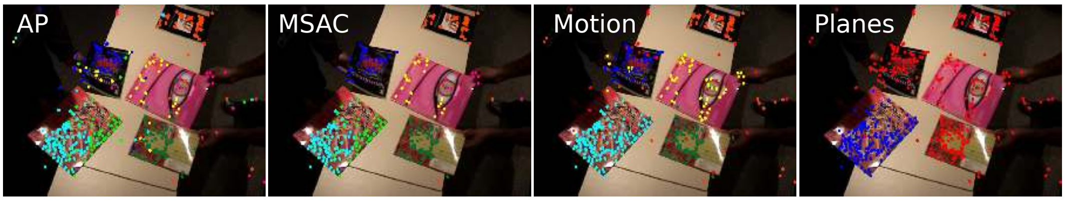


Figure 4.11: Results obtained for the scenario of presence of multiple motions in Figure 4.9. The dominant motion is selected as the one which has the largest number of associated clusters, which, in this case, does not correspond to the camera motion. This leads to the segmentation of only one plane in the last step of the pipeline, and all remaining correspondences being assigned as outliers (red).

Table 4.2: Computational times on an Intel Core i7 3.4 GHz

AC extraction & matching	Metric computation & affinity prop.	Homography MSAC	Motion segment.	Plane merging	Total
0.21 s	0.20 s	0.08 s	0.07 s	0.09 s	0.65 s

By forcing VISO2-Mono to provide an estimation for the camera motion, poor results were obtained: $e_{\mathsf{R}} = 0.03°$, $e_{\mathbf{t}} = 26.0°$ and $||\mathbf{t}|| = 69.3\ cm$.

Figure 4.11 shows example 4 that consists in a moving camera observing 4 different planar motions. It can be seen that 7 clusters were initially segmented, and 5 different motions were correctly detected by the PEaRL framework described in Section 4.4.1.2. Since the larger plane was initially segmented into two clusters, its motion is incorrectly identified as the camera motion and only this plane is segmented in the final step. In this case, the rigid structure has little image support, so a more sophisticated scheme for identifying the camera motion is required. A possibility would be to used temporal consistency as proposed in [73].

Table 4.2 shows the computational times of each step of the proposed pipeline. Except for the C++ implementation of the graph cut optimization [137], the rest of the algorithm is implemented in Matlab. We believe that a C++ implementation of the whole algorithm would allow it to reach a frame rate of $5 - 10\ fps$.

4.4.3 vSLAM pipeline

In this section we describe our proposed method πMatch that takes as input a sequence of images and outputs the camera motions and a PPR of the scene. We presented in Section 4.4.1 a two-view pipeline that takes as input a pair of images and outputs the camera motion, with the translation estimated up to scale, along with the PPR of the scene. In order to be able to work with image sequences, the relative scale of translation between motions must be estimated.

For every two consecutive motions $(\mathsf{R}_i, \mathbf{t}_i)$ and $(\mathsf{R}_{i+1}, \mathbf{t}_{i+1})$, where $(\mathsf{R}_i, \mathbf{t}_i)$ is the motion between frames i and $i{+}1$, the scale of translation s_i^{i+1} is estimated by fixing the norm of $\mathbf{t}_i$ and computing the new translation vector $s_i^{i+1}\mathbf{t}_{i+1}$. We consider point tracks between frames i and $i{+}2$ and start by reconstructing the 3D points in frames i and $i{+}1$ using motions $(\mathsf{R}_i, \mathbf{t}_i)$ and $(\mathsf{R}_{i+1}, \mathbf{t}_{i+1})$, respectively. We consider as inliers the 3D points whose reprojection error lies below a pre-defined threshold. We observed that the accurate estimation of the rotation and direction of translation allows a good selection of inliers. The two sets of reconstructed 3D points $\mathbf{X}_i$ and $\mathbf{X}_{i+1}$ correspond to the same scene points represented in different reference frames. Thus, using motion $(\mathsf{R}_i, \mathbf{t}_i)$, $\mathbf{X}_i$ can be represented in reference frame $i{+}1$, and scale s_i^{i+1} is initialized by taking the median of the element-wise ratio $\frac{\mathbf{X}_i'}{\mathbf{X}_{i+1}}$, where $\mathbf{X}_i' = \mathsf{R}_i\mathbf{X}_i + \mathbf{t}_i$. Scale s_i^{i+1} is then refined by minimizing the maximum reprojection error of the 3D points $\mathbf{X}_i'$ in frames $i{+}1$ and $i{+}2$, computed using motion $(\mathsf{R}_{i+1}, s_i^{i+1}\mathbf{t}_{i+1})$:

$$s_i^{i+1*} = \min_{s_i^{i+1}} \sum_k \left(\max(d_k^{i+1}, d_k^{i+2})\right)^2, \tag{4.24}$$

where d_k^i is the reprojection error of point k in frame i. Due to the good selection of inliers, this procedure provides accurate results. Also, since we only optimize one parameter, the computational time of this refinement step is very low (approximately 18 ms in our experiments). For images in which the camera motion is a pure rotation, the scale is not estimated. When the camera resumes the movement, the scale is determined using the new motion and the previous one which was not a pure rotation. This scheme allows the relative scale information to be kept through the whole sequence.

The last step of the pipeline concerns the refinement of the piecewise planar structure by selecting the best planes across multiple frames. This is an adaptation of the discrete optimization step proposed in Section 3.3 for stereo sequences, where we propose to refine the camera motion and the PPR in a PEaRL framework by considering multiple stereo pairs simultaneously. In this case, for the sake of computational efficiency and since both the camera motion and the planes have already been refined, we propose to include a final discrete optimization step in a sliding window approach for improving the overall PPR. As explained in Section 3.3, optimizing over multiple frames allows the backpropagation of planes, significantly improving the accuracy and visual perception of the 3D model. Figure 4.12 depicts

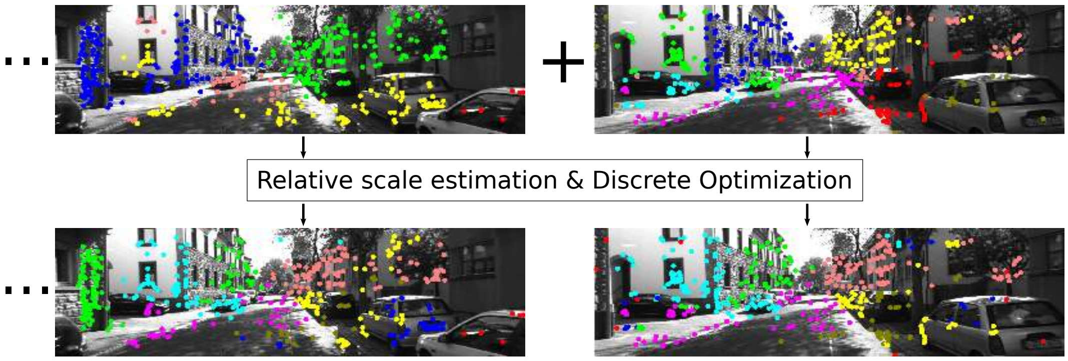


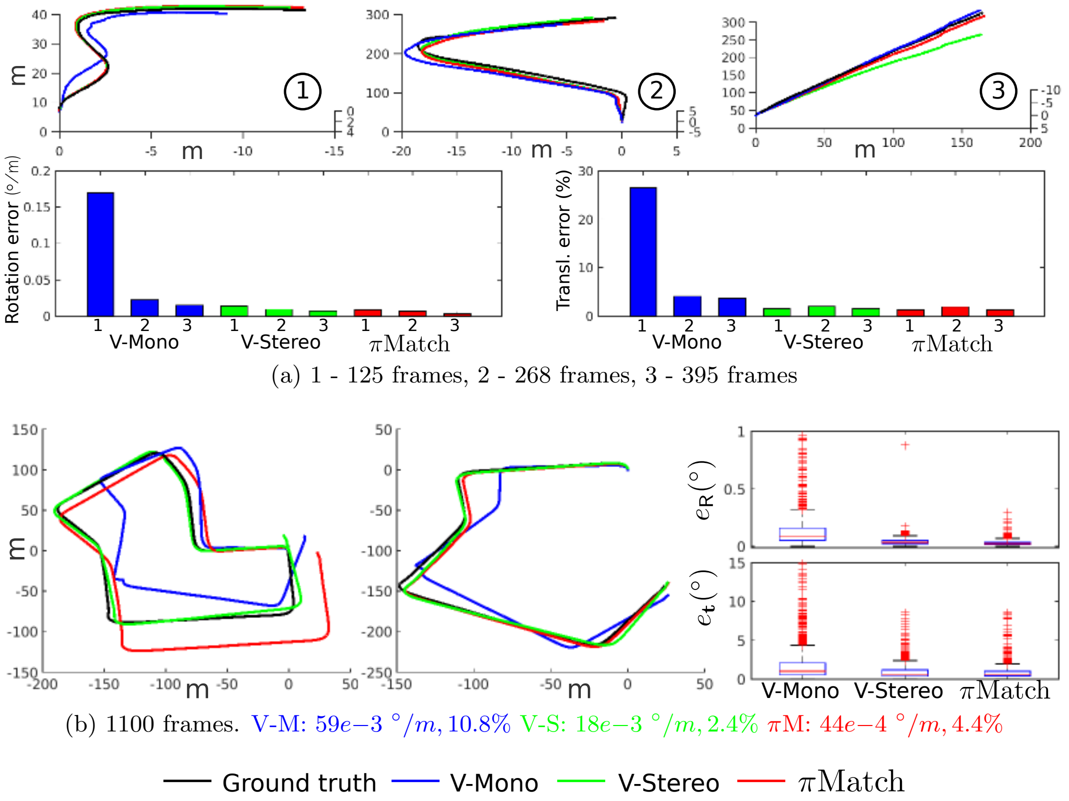
Figure 4.12: The two-view pipeline described in Section 4.4.1 is a applied to each new image pair and its scale is estimated. This allows to select the best planes across multiple views in a PEaRL framework. An important advantage is the back-propagation of planes: the fronto-parallel plane corresponding to the building façade (cyan in the output images) is correctly detected in the incoming image and back-propagated to previous images. Colors identify planes in the output images. Red identifies outlier points.

this advantage, where it can be seen that the fronto-parallel plane of the building façade is detected in the new image and backpropagated to the previous one, providing a much more realistic 3D model.

We formulate this discrete optimization as a labeling problem where the goal is to minimize an objective function E defined as

$$E(\mathbf{l}) = \underbrace{\sum_i \sum_{\mathbf{p}^i} D_{\mathbf{p}^i}(l_{\mathbf{p}^i})}_{\text{Data term}} + \underbrace{\lambda_{S'} \sum_i \sum_{(\mathbf{p}^i,\mathbf{q}^i)\in\mathcal{N}'} w_{\mathbf{p}^i\mathbf{q}^i}\,\delta(l_{\mathbf{p}^i} \neq l_{\mathbf{q}^i})}_{\text{Smoothness term}} + \underbrace{\lambda_{L'}|\mathcal{L}_{\mathbf{l}}|}_{\text{Label term}}\ , \tag{4.25}$$

where the label set is the union of the planes detected in each image pair i separately ($\mathcal{L} = \{\bigcup_i \Pi^i, l_\emptyset\}$), the nodes $\mathbf{p}^i$ are the point correspondences in all images i and $\lambda_{S'}$ and $\lambda_{L'}$ are weighting constants. We use the refined motions $\mathsf{R}_i, \mathbf{t}_i, s_i^{i+1}$ to represent the planes in the label set in all reference frames i and compute the STE for defining the data cost $D_{\mathbf{p}^i}$. The neighbourhood $\mathcal{N}'$ is defined by Delaunay triangulation of the points in each image i. We also define as neighbours the points $\mathbf{p}^i$ and $\mathbf{q}^i$ that correspond to the same point track, and set the weight $w_{\mathbf{p}^i\mathbf{q}^i}$ to a large value in this case. This forces points belonging to the same track to be assigned the same label across frames. The remaining weights $w_{\mathbf{p}^i\mathbf{q}^i}$ are inversely proportional to the distance between $\mathbf{p}^i$ and $\mathbf{q}^i$. In our experiments, for a sliding window of 5 frames (4 camera motions) this optimization took around 50 ms.



(a) 1 - 125 frames, 2 - 268 frames, 3 - 395 frames

(b) 1100 frames. V-M: $59e{-}3\ °/m, 10.8\%$ V-S: $18e{-}3\ °/m, 2.4\%$ πM: $44e{-}4\ °/m, 4.4\%$

Figure 4.13: Results obtained on 4 sequences of the KITTI dataset [42, 43] using the monocular method VISO2-Mono [44], the stereo method VISO2-Stereo [44], and our proposed monocular method πMatch. The bar plots and caption (b) show the average rotation and translation errors computed using the metric proposed in [43]. The boxplots show the distribution of rotation (e_{R}) and translation ($e_{\mathbf{t}}$) errors computed for each image pair.

4.4.4 Large-scale experiments

This section reports experiments on 4 sequences of the KITTI dataset [42, 43] performed with the monocular method VISO2-Mono [44], the stereo method VISO2-Stereo [44], and our proposed method πMatch. Figure 4.13 shows the results obtained for the 3 methods, with the errors being quantified using the error metric described in Section 4.4.2 and the metric proposed in [44].

The first observation is that, when compared to the other monocular method VISO2-Mono, our method is far more superior in the estimation of rotation and translation. Regarding the scale estimation, while VISO2-Mono uses information about the height of the camera, πMatch does not make any prior assumptions and

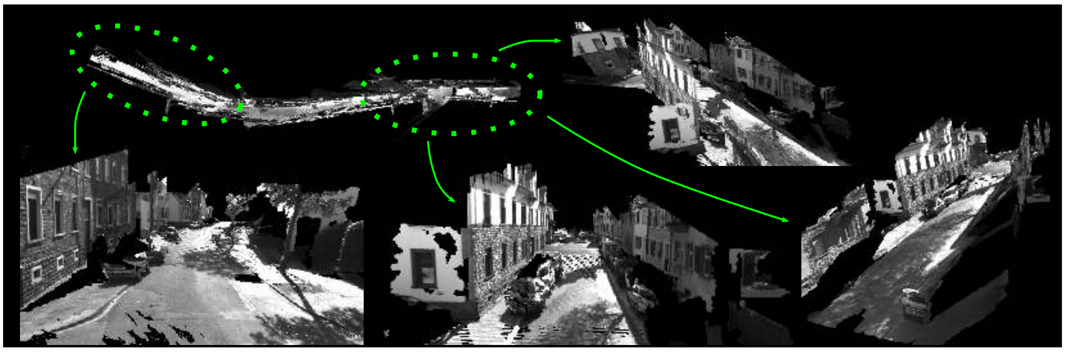


Figure 4.14: PPR of the 268-frame sequence in Figure 4.13a. A proper alignment of the individual PPRs of each image pair is observed, confirming the good quality of the scale estimation step. Some areas are shown in greater detail.

still significantly outperforms this method. Moreover, another important observation is that for the 3 shortest sequences, πMatch also manages to outperform the stereo method VISO2-Stereo, being particularly more accurate in the estimation of the rotation. This demonstrates the effectiveness of our proposed motion hypotheses generation and selection scheme.

Regarding the 1100-frame sequence, the trajectory makes it evident that VISO2-Stereo outperforms our method. However, from the boxplots showing the individual rotation e_{R} and translation $e_{\mathbf{t}}$ errors, it can be seen that πMatch provides more accurate estimations, leading to the conclusion that the reason for the overall inferior performance of our method is some inaccuracy in the scale estimation. We observed that the estimation of the scale is frequently very accurate, and only fails in few cases. Due to the propagation of error, one poorly estimated scale will influence all subsequent ones, which does not happen in stereo methods. In order to illustrate this fact, we show in Figure 4.13b the trajectories for the same sequence after removing the first 300 frames, where it can be seen that the πMatch outperforms VISO2-Stereo. For this sub-trajectory, VISO2-Stereo provided an error of $23e{-}3\ {}^\circ/m$ in rotation and 2.55% in translation while our method was more accurate: $50e{-}4\ {}^\circ/m$ in rotation and 1.96% in translation.

In Figure 4.14, the PPR obtained for the 268-frame sequence demonstrates not only the accuracy of our method but also the importance of the last discrete optimization step, where the best planes across multiple frames (5 in this case) are selected. Since we are simply concatenating the individual PPRs for each image pair, the final 3D model would be visually significantly worse if this optimization stage had not been used. This experiment shows that πMatch performs accurate vSLAM and dense PPR from monocular sequences, significantly outperforming the monocular method VISO2-Mono, and also being superior in the estimation of rotation to the state-of-the-art stereo system VISO2-Stereo.

4.5 Conclusions

We investigate the use of ACs in homography and essential matrix estimation and show that both can be accomplished from as few as 2 ACs, benefiting hypothesize-and-test schemes. The geometric insights provided two new error metrics that proved to be very useful in the clustering of points in planes. Also, we derive, for the first time, the general linear constraints for an AC to be compatible with an epipolar geometry. A particular case for these constraints was determined in [12, 13] that is only valid when the coordinate system is centred in the AC. The proposed approaches are successfully applied in planar segmentation and homography estimation tasks, as well as in conventional SfM. In the latter case, our method compares very favourably with the state-of-the-art 5-point algorithm in the presence of high percentages of outliers and/or small input data sets.

In addition, we describe the first feature-based pipeline for vSLAM and dense PPR from a monocular sequence. It works by extracting ACs and employing a new error metric for detecting scene planes. These planes are used for generating motion hypotheses that allow not only the accurate estimation of the camera motion, but also of other motions present in the image, in a PEaRL framework. The refined camera motion and initial plane hypotheses are used in another PEaRL scheme, yielding good PPRs of the scene from two views. The extension to longer sequences is done by estimating the scale between every two consecutive image pairs, and a final discrete optimization step allows the exchange of planes between frames, providing improved PPR results. The final experiment shows that scale drift may occur due to a few poor estimations of the scale of translation. As future work, we intend to devise a method to overcome this problem by making use of the detected planes. The idea is that since planes are more constant over time than points, using plane correspondences across frames could significantly reduce the scale drift. The total execution time of πMatch mainly implemented in Matlab is approximately 0.72 s. We will implement a C++ version of the pipeline, which we expect to run in about $5 - 10$ fps.

Chapter 5

Point Cloud Registration using Normals

3D Registration is the process of finding the rigid transformations (rotation and translation) that best align two or more point clouds such that their overlapping areas match as well as possible. 3D Point cloud registration is a well studied problem in the computer vision literature due to its important applications in object modelling, detection and recognition, tracking, and SLAM.

The oldest and best established algorithm for solving the point cloud registration problem is ICP [14]. It works iteratively by alternating between finding the points in the target 3D model that are closest to the source point cloud, estimating a new pose by minimizing a cost function, and re-aligning the two point clouds using the new pose estimation, until convergence. Since it relies in iterative optimization, if the initialization is poor, it might converge to local minima.

Many authors have worked in improving the resilience of ICP to outliers and missing data, often observed in 3D scans. Examples include the work by Mitra *et al.* [84] where local quadratic approximations of the squared distance function are used. Also, Bouaziz *et al.* presented SparseICP [16] that formulates the registration problem using sparsity-induced norms. More recently, an efficient Sparce ICP method [77] was devised by including a Simulated Annealing search. Similarly to the standard ICP, all of these methods require good initialization of the rigid transformation. Other works, such as Go-ICP [142,143], have tried to avoid the initialization issue by assuring global convergence. Go-ICP is based on a branch-and-bound scheme that searches the entire 3D motion space, with a local ICP included for speed-up. Unfortunately, this requires large overlap between the point clouds, which does not always happen due to occlusions or acquisitions in very different viewpoints, and scales poorly, becoming prohibitive for large datasets.

In order to avoid the requirements of good initialization and large overlap, some authors have suggested to find sparse correspondences between point clouds and use them in an hypothesize-and-test framework, like RANSAC, to perform an initial alignment that is subsequently refined by ICP. Such approaches handle completely misaligned point clouds, do not require the point clouds to have large overlap, and are inherently robust to outliers. The difficulty is in establishing plausible correspondences between point clouds. It is well known that a rigid transformation that aligns two point clouds can be computed in a closed form manner from 3 correct 3D point correspondences [55]. Thus, a naive approach could consist in selecting 3 random points in one point cloud and searching for all possible triplets in the other. This search would have complexity of $\mathcal{O}(N^3)$, with N being the number of points in the second model, easily becoming prohibitive. This chapter proposes a method to address this issue, which we call 2-Point-Normal Sets (2PNS).

An approach inspired by what is commonly done in 2D images is to find salient regions/landmarks in the point clouds, characterize them by a descriptor for performing association and establish alignment hypotheses for each match in a RANSAC-like framework or using voting-based pose estimation [22, 27, 138]. Such approaches fail mainly because point clouds are often smooth and/or noisy, which hampers finding repeatable saliences that can be matched.

Aiger *et al.* [1] proposed a framework, dubbed 4-Point Congruent Sets (4PCS), that, given 4 coplanar points in one point cloud, it enables to exhaustively search for correspondences in the other point cloud with a complexity of $\mathcal{O}(N^2)$. The key idea is to use the relations between 4 coplanar points to define affine invariants that are preserved by a rigid displacement. These invariants are used to speed up search, enabling to decrease complexity by an order of growth when compared with the naive approach. More recently, Mellado *et al.* proposed the Super4PCS algorithm [79] that decreases complexity from quadratic to linear time by using a number of improvements in the search stage. 4PCS and Super4PCS have inspired the Super Generalized 4PCS [85] and the works by Theiler *et al.* [127, 128]. The former removes the coplanarity constraint and considers general 4-point bases that lead to speed-ups up to 6.5× when compared to Super4PCS. The latter extract keypoints on LiDAR point clouds and apply the 4PCS algorithm for robust registration.

In this chapter, we advance the Super4PCS framework by using not 4 points to establish an alignment hypothesis but 2 points and its normals. The normals to points in a point cloud are relatively inexpensive to compute [7,64] and have already been used in the context of registration mainly to define local descriptors [22,27,113, 138]. We show that it is possible to compute the rigid transformation R, t from 2 points and 1 normal and that matching 2 points+normals can be significantly more efficient and robust than searching for sets of 4 points forming a congruent base. The results show that we can obtain speed-ups up to two orders of magnitude in noise-free datasets and up to 5.2× in Kinect scans with respect to Super4PCS, while improving robustness and alignment accuracy.

Since our proposed method is based on the Super4PCS algorithm, we start by revisiting the 4PCS algorithm and its improvements in Section 5.1. Our new method is presented in Section 5.2 that includes a detailed description of the differences w.r.t. Super4PCS for all the steps. Section 5.3 reports two sets of experiments that confirm the efficiency of our proposed method. A comparison with the original algorithm Super4PCS is provided in all cases. This method is chosen as baseline of comparison, instead of the recent developments that led to the Super Generalized 4PCS algorithm [85], mainly because Super4PCS is a more consolidated framework that, unlike Super Generalized 4PCS, has public implementations available. Moreover the method herein proposed has, similarly to Super Generalized 4PCS, the advantage of not requiring planar points (it only uses two points so planarity issues do not apply) and experiments show that it provides speed ups over Super4PCS that are significantly above the ones claimed in Super Generalized 4PCS. Section 5.4 concludes this chapter with some future research directions.

5.1 Review of Super4PCS [79]

Global 3D point cloud registration works by finding putative point correspondences between point clouds that enable to establish alignment hypotheses. These hypotheses are ranked according to some metric and the transformation T with the highest score is then refined by ICP. Since a rigid transformation can be estimated from a minimum of 3 points [55], a naive approach would try to perform exhaustive search of triplets with a runtime complexity of $\mathcal{O}(N^3)$, N being the size of the point cloud. The algorithms 4PCS [1] and later Super4PCS [79] work with sets of 4 points, instead of 3, to make the search easier.

Let $\mathcal{P}$ and $\mathcal{Q}$ be the source and target point clouds, respectively, to be registered. The goal of 4PCS and Super4PCS is to find the transformation T that best aligns them by solving the Largest Common Pointset (LCP) problem: maximize the cardinality of a transformed subset of $\mathcal{P}$ according to the property that every point in that subset is within a predefined distance to $\mathcal{Q}$. These methods make no assumption about the starting poses of the point clouds and are able to handle situations of small overlap. Also, they favour the use of points in the cloud that are far apart in order to increase resilience to noise and outliers.

5.1.1 The original 4PCS algorithm

The 4PCS algorithm solves the global 3D registration problem by using coplanar sets of 4 points, rather than the minimum sets of 3 points, allowing the employment of a technique that efficiently matches pairs of affine invariant ratios in 3D. The approach works by selecting a base of 4 coplanar points in the source point cloud $\mathcal{P}$ and finding all the 4-point sets in the target point cloud $\mathcal{Q}$ that are approximately congruent with the base, *i.e.* related by rigid transformations. For each potential 4-point set from $\mathcal{Q}$, the aligning transformation T is computed and the best transformation according to the LCP score is retained. This process is repeated in a RANSAC [33] scheme until a good solution is found, or a maximum number of iterations is reached.

The first step of each RANSAC iteration is the selection of a random base of 4 coplanar points from $\mathcal{P}$. The algorithm starts by randomly selecting 3 points from $\mathcal{P}$ that yield a wide triangle. This is done by testing a set of randomly chosen triangles and retaining the widest one. However, due to the possible small overlap between the source and target point clouds, the size of the triangle must be below a certain threshold. The fourth point in the quadrilateral is selected as one that is close to be planar to the other 3 but still not too close to them. This is done by testing all the points in $\mathcal{P}$ and picking the one that best fits the criteria. Thus, the runtime complexity of this stage is $\mathcal{O}(S)$, where S is the number of points in $\mathcal{P}$.

Let $\mathcal{B} = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$ in Figure 5.1 be the randomly selected coplanar base in source $\mathcal{P}$, such that $\mathbf{AB}$ intersects $\mathbf{CD}$ in the intermediate point $\mathbf{E}$. The key idea explored by the authors is that the ratios

$$r_1 = \frac{||\mathbf{A} - \mathbf{E}||}{||\mathbf{A} - \mathbf{B}||} \text{ and } r_2 = \frac{||\mathbf{C} - \mathbf{E}||}{||\mathbf{C} - \mathbf{D}||} \tag{5.1}$$

remain invariant under affine transformations, and hence under rigid motion. Since distances are also preserved under rigid transformations, these 4 invariants ($r_1, r_2, d_1 = ||\mathbf{A} - \mathbf{B}||, d_2 = ||\mathbf{C} - \mathbf{D}||$) are used to constrain the search for congruent 4-point sets in target $\mathcal{Q}$. The algorithm starts by extracting all pairs of points at distance d_1 or

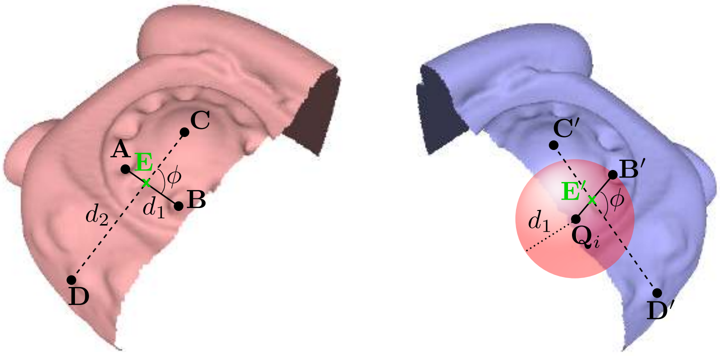


Figure 5.1: A source $\mathcal{P}$ (pink) and a target $\mathcal{Q}$ (blue) point clouds are shown with the corresponding 4-point sets used by 4PCS and Super4PCS. Super4PCS finds all pairs at distance d_1 in the target point cloud by centring a sphere of radius d_1 in all points and selecting the points that intersect with the point cloud.

d_2 from $\mathcal{Q}$, which is done in $\mathcal{O}(N^2)$ time, where N is the cardinality of $\mathcal{Q}$. For each extracted pair $(\mathbf{Q}_1, \mathbf{Q}_2) \in \mathcal{Q}$ with distance d_1 or d_2, the intermediate points $\mathbf{E}_1$ and $\mathbf{E}_2$ are computed:

$$\mathbf{E}_1 = \mathbf{Q}_1 + r_1(\mathbf{Q}_2 - \mathbf{Q}_1) \text{ and } \mathbf{E}_2 = \mathbf{Q}_1 + r_2(\mathbf{Q}_2 - \mathbf{Q}_1). \tag{5.2}$$

The authors explore the fact that two pairs that have their $\mathbf{E}_1$ and $\mathbf{E}_2$ coincident form a 4-point base that is related with $\mathcal{B}$ by an affine transformation. Thus, the intermediate points obtained from pairs at distance d_1 are used for building an approximate range tree structure (RS) in $\mathbb{R}^3$, which takes $\mathcal{O}(M \log M)$ to be built, where M is the number of pairs, and has a query time of $\mathcal{O}(\log M + K)$, K being the number of points to be reported. The intermediate points that arise from pairs at distance d_2 are used as queries in this tree, yielding K 4-point sets from the retrieved pairs. Since affine invariants were used, the set of extracted bases contains sets which are not related by a rigid transformation. These are then removed in a verification step that takes $\mathcal{O}(K)$ time, by using the angle ϕ between the line segments (Figure 5.1), which is preserved under rigid transformations.

The authors claim that for large dense data sets, a heavy uniform sampling of the point clouds can be performed in order to use only a small fraction of the points for computing the alignment, and the full data set is then employed for the computation of the LCP. Experiments reported in the paper show that using as few as 5% of the data points is sufficient. It is the use of the LCP measure that makes the method resilient to such heavy uniform sampling.

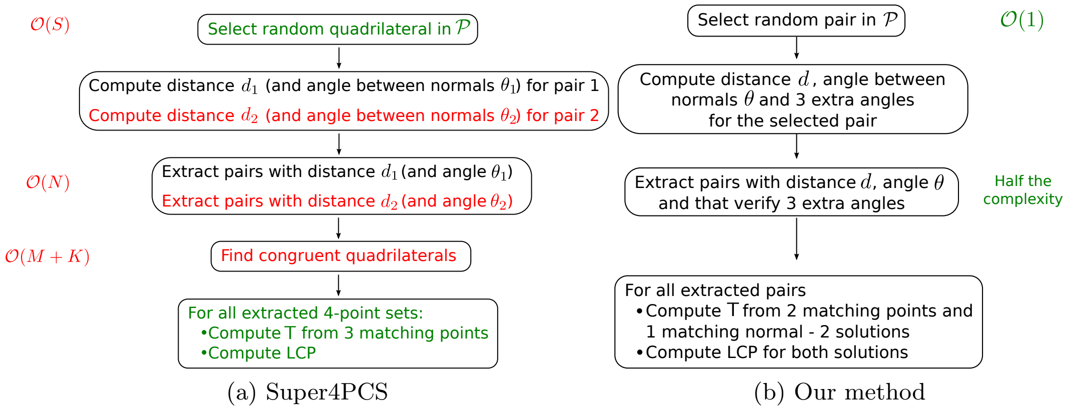
5.1.2 Super4PCS

Recently, Mellado *et al.* [79] proposed the Super4PCS algorithm that builds on 4PCS and decreases its complexity to $\mathcal{O}(N + M + K)$. This is done by solving the two main bottlenecks of 4PCS which are the pair extraction stage and the large number of reported congruent sets which leads to an expensive verification step to remove non-rigid invariant 4-point candidates.

Efficient pair extraction (in $\mathcal{O}(N)$ time) is achieved by finding the points close to the spheres centred in $\mathbf{Q}_i \in \mathcal{Q}$ and with radius $d_1 \pm \epsilon$ and $d_2 \pm \epsilon$, where ϵ is a tolerance considered due to the noise in the data. Figure 5.1 illustrates this procedure, depicting a sphere of radius d_1 centred in an arbitrary point $\mathbf{Q}_i$. The point cloud $\mathcal{Q}$ is organized in a 3D grid, subdivided recursively and the intersection between the set of spheres and the subdivided volumes is computed. Pairs are then built between the points lying in the intersected volumes and the sphere centres.

Concerning the second bottleneck, the authors' idea is to extract the exact set of congruent 4-point bases that only contains rigid-invariant candidates, avoiding the need to have a verification/filtering step. A quadrilateral is congruent to the base selected from $\mathcal{P}$ if it is composed of pairs with the correct length (d_1 and d_2), and if the angle ϕ between these pairs is similar to the angle formed by the two pairs in the base. This is accomplished as follows: each pair is represented by its intermediate point $\mathbf{E}$, computed as in Equation 5.1.1, and its orientation, and the pairs at distance d_1 are hashed by this position and orientation, with the directions being mapped to a spherical map. In the query stage, the position $\mathbf{E}$ is used to access cells in a regular grid, such that the retrieved points lie in the same cell as the query. Also, the corresponding spherical map is queried using a d_2 pair direction to find all pairs with angle ϕ w.r.t. the query direction. This is done by intersecting a cone of aperture 2ϕ around the query direction with the spherical map. This stage has runtime complexity $\mathcal{O}(M + K)$ due to the point retrieval and query of the spherical map steps.

Figure 5.2a shows the sequence of steps of a complete RANSAC iteration of Super4PCS. Since normals are often available for point clouds, or can be easily computed, they can be used to further prune the number of extracted pairs. Whenever this information is available, Super4PCS includes it by computing the angle θ_1 and θ_2 between the normals of pairs of points with distance d_1 and d_2, respectively, and filtering the extracted pairs such that θ_1 and θ_2 are similar to the corresponding angles between the normals of the base set extracted from $\mathcal{P}$. We will consider this version of Super4PCS with normals throughout this chapter.



$\mathcal{O}(S)$
Select random quadrilateral in $\mathcal{P}$
Compute distance d_1 (and angle between normals θ_1) for pair 1
Compute distance d_2 (and angle between normals θ_2) for pair 2
$\mathcal{O}(N)$
Extract pairs with distance d_1 (and angle θ_1)
Extract pairs with distance d_2 (and angle θ_2)
$\mathcal{O}(M+K)$
Find congruent quadrilaterals
For all extracted 4-point sets:
•Compute T from 3 matching points
•Compute LCP
(a) Super4PCS
Select random pair in $\mathcal{P}$
$\mathcal{O}(1)$
Compute distance d, angle between normals θ and 3 extra angles for the selected pair
Extract pairs with distance d, angle θ and that verify 3 extra angles
Half the complexity
For all extracted pairs
•Compute T from 2 matching points and 1 matching normal - 2 solutions
•Compute LCP for both solutions
(b) Our method

Figure 5.2: Sequence of steps of a complete RANSAC iteration of (a) Super4PCS and (b) our method. The stages that were modified are identified in green and the ones that were removed are shown in red.

5.2 2-Point-Normal Sets (2PNS)

In this section we present 2PNS, our global 3D registration method that builds on the Super4PCS method reviewed in the previous section. Motivated by the fact that normals to point clouds can be computed in a robust, inexpensive manner, we propose to use them to solve the alignment problem. It can be shown that the rigid transformation T can be computed from 2 points plus the normal at one of these points. Thus, instead of searching for quadrilaterals, we look just for pairs of points using their normals to decide if they are a plausible match. This dramatically reduces the combinatorics of the search leading to a simpler, substantially less complex algorithm, as shown in Figure 5.2b that presents the scheme of a complete RANSAC iteration of our method. The modified and removed steps w.r.t. Super4PCS are highlighted in green and red, respectively, in Figure 5.2a.

This section starts by describing how normals can be computed, presents the search algorithm, shows how to compute the alignment from 2 points and 1 normal, and finally makes a comparative analysis of complexity with respect to Super4PCS.

5.2.1 Computing normals

Point cloud normal estimation has been a well-studied problem due to its important applications in areas such as object detection, segmentation, surface fitting and registration [7, 60, 64]. According to a recent work that compares several approaches for surface normal estimation in point clouds, the most accurate method is

PlanePCA [60], an optimal total least squares solution. Let $\mathcal{S} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$, $\mathbf{X}_i \in \mathbb{R}^3$ be a point cloud and $\mathbf{X}_i$ the reference point whose normal $\mathbf{n}_i$ we wish to estimate. Also, let $\mathcal{P}_i = \{\mathbf{P}_{i1}, \mathbf{P}_{i2}, \ldots, \mathbf{P}_{ik}\}, \mathbf{P}_{ij} \in \mathcal{S}, \mathbf{P}_{ij} \neq \mathbf{X}_i$ be the k points in the neighbourhood of $\mathbf{X}_i$, which are commonly found using standard nearest neighbours searches with kd-trees. PlanePCA aims to find the direction of minimum variance in the neighbourhood $\mathsf{P}_i^+ = [\mathbf{X}_i, \mathbf{P}_{i1}, \mathbf{P}_{i2}, \ldots, \mathbf{P}_{ik}]^\mathsf{T}$, having an objective function defined by

$$F(\mathbf{n}_i) = ||[\mathsf{P}_i^+ - \bar{\mathsf{P}}_i^+]\mathbf{n}_i||_2, \tag{5.3}$$

$\bar{\mathsf{P}}_i^+$ being a matrix containing the mean vector of P_i^+ in every row. The singular vector of $[\mathsf{P}_i^+ - \bar{\mathsf{P}}_i^+]$ associated with the smallest singular value minimizes the function in Equation 5.3. Since this is equivalent to taking the principal component with the smallest variance after performing Principal Component Analysis (PCA) on P_i^+, this method is termed PlanePCA.

Since the direction of the normal is not recovered - its symmetric is also a solution - there are always two solutions for the unitary normal vector. This is important because normals are used in our proposed method in the estimation of the rotation R and symmetric vectors lead to different solutions, as shown next. A scheme for forcing coherent directions of all normals in the two point clouds being registered could be devised, for instance by computing each point cloud's centroid and making the normal vectors point in that direction. However, since we are dealing with cases of very small overlap, this could introduce errors and thus we opted to work with the two solutions for the normal vectors.

2PNS fails whenever normals are not properly estimated. This may occur if the point cloud is too sparse or strongly dominated by sharp edges and corners. However, both cases rarely occur, either in real or synthetic scenarios, with the point clouds being often sufficiently dense and containing many smooth regions.

5.2.2 The 2PNS search to obtain putative matches

At each RANSAC iteration, our method starts by extracting a random pair of points and the corresponding normals from the source point cloud $\mathcal{P}$. These entities are shown in Figure 5.3, where $\mathcal{P}$ is the pink point cloud, $(\mathbf{A}, \mathbf{B})$ is the pair and $\mathbf{n}_\mathbf{A}, \mathbf{n}_\mathbf{B}$ the two respective normals. Since we establish alignment hypotheses from 2 corresponding points and its normals, it is important that all the pairs extracted from the target point cloud $\mathcal{Q}$, shown in blue in Figure 5.3, are congruent with the base pair extracted from $\mathcal{P}$, *i.e.* can be aligned by a rigid transformation.

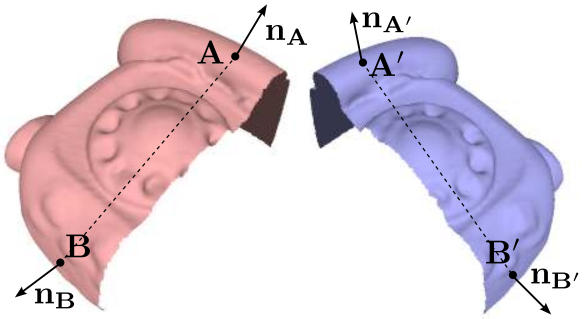


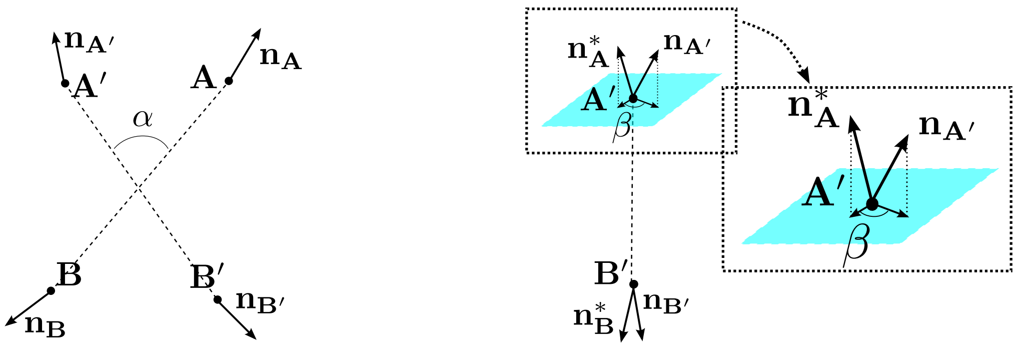
Figure 5.3: A source (pink) and a target (blue) point clouds are shown with the corresponding pairs of points and normals. These two pairs are the only information used for computing the rigid transformation between the two point clouds, as opposed to Super4PCS that requires the extraction of 4-point bases.

Super4PCS extracts pairs that have a distance $d = ||\mathbf{A} - \mathbf{B}||$ and an angle $\theta = \angle(\mathbf{n_A}, \mathbf{n_B})$. However, it is known that using only these two rigid invariants leads to a set of extracted pairs that may contain instances that can never be aligned by a rigid transformation. Thus, we propose to perform the search for pairs that verify not only the conditions for d and θ, but also for 3 extra angles, namely the angle of the first normal with the line segment joining the two points, the angle of the second normal with the line segment, and the angle between the two normals projected onto the plane orthogonal to line segment. Since angles are preserved under rigid transformations, these 5 invariants are used for extracting congruent pairs in the target point cloud $\mathcal{Q}$, using the method explained in Section 5.1.2. Due to the possible presence of noise and outliers in the data, a tolerance γ is considered, which can vary between 5° and 20°, depending on the quality of the input data.

5.2.3 Estimation of $\mathsf{R}, \mathbf{t}$

Let $(\mathbf{A}', \mathbf{B}')$, with normals $\mathbf{n_{A'}}$ and $\mathbf{n_{B'}}$, be a pair of points in the target point cloud $\mathcal{Q}$ congruent with the selected pair in the source $\mathcal{P}$, as shown in Figure 5.3. The rigid transformation is estimated by first centring the two pairs in the origin (using their centroids) and then computing the rotation R. This is done in two steps: first the rotation R_α that aligns the vectors $\mathbf{v}_1 = \mathbf{B} - \mathbf{A}$ and $\mathbf{v}_2 = \mathbf{B}' - \mathbf{A}'$ is estimated; then the rotation R_β that aligns the normal vectors is determined. The final rotation R comes from $\mathsf{R} = \mathsf{R}_\beta \mathsf{R}_\alpha$.

Figure 5.4a illustrates the estimation of R_α. This is a simple rotation to align two vectors, defined by the rotation axis $\omega_\alpha = \mathbf{v}_1 \times \mathbf{v}_2$ and the rotation angle $\alpha =$



(a) Estimation of α

(b) Estimation of β

Figure 5.4: The rotation R is computed in two steps, by first determining (a) the rotation R_α that aligns the two vectors connecting the 3D points and then (b) the rotation R_β that aligns the normal vectors.

$\cos^{-1}(\mathbf{v}_1 \cdot \mathbf{v}_2)$. R_α is determined using Rodrigues's formula. Let $\mathbf{n}^*_\mathbf{P} = \mathsf{R}_\alpha \mathbf{n}_P, \mathbf{P} = \mathbf{A}, \mathbf{B}$ be the rotated normal vectors as depicted in Figure 5.4b. R_β can be found by rotating an angle β around axis $\mathbf{v}_2$. In order to find β, we project the normal vectors $\mathbf{n}^*_\mathbf{A}$ and $\mathbf{n}_{\mathbf{A}'}$ onto the plane defined by the rotation axis $\mathbf{v}_2$ and compute the angle between the projected vectors. R_β has only one unknown parameter, so one normal suffices to fully determine the rotation (it could be done using $\mathbf{n}^*_\mathbf{B}$ and $\mathbf{n}_{\mathbf{B}'}$).

The solution for R_β corresponding to the other solution for the normal vector (the symmetric one) is computed from the angle $\beta' = \pi - \beta$ since the projected vector also becomes symmetric. A new solution for R is estimated, and thus this method always provides 2 solutions for the rigid transformation.

After knowing R, the translation $\mathbf{t}$ is estimated from R and the translation vectors used for centring the two pairs in the origin.

5.2.4 Comparison of complexity with Super4PCS

Since our proposed method is based on the Super4PCS algorithm, making use of one of its main strengths that is a fast scheme for extracting pairs of points according to a distance and an angle, in the present section we highlight the differences w.r.t. Super4PCS that yielded our much simpler and faster approach. The improvements are in avoiding extra computation while selecting a random base in $\mathcal{P}$, reducing in half the pair search and not requiring a congruent set extraction stage. On the down side comes the fact that for each pair there are two motion solutions that must be verified, whereas Super4PCS generates only one, but this overhead is largely compensated by the improvements above, as shown in the experimental section.

5.2.4.1 Selection of the base from the source point cloud

Both our and the Super4PCS algorithms start by extracting a base from the source point cloud $\mathcal{P}$. While the Super4PCS requires a 4-point coplanar base, our method simply extracts a random pair of points.

The Super4PCS's procedure of extracting a coplanar quadrilateral from $\mathcal{P}$ is described in Section 5.1.1. It runs in $\mathcal{O}(S)$ time because the algorithm goes through all the points in $\mathcal{P}$ to find the fourth point in the base. On the other hand, our method simply selects a random pair of points by testing a set of pairs and choosing the widest one. This distance has an upper bound to account for the overlap between the point clouds. Since we are not going through the whole point cloud, this stage is very fast and runs in $\mathcal{O}(1)$ time, being independent of the size of $\mathcal{P}$. Remark that since our base is a line, instead of a quadrilateral, we can choose a wider segment than the length of the quadrilateral sides and still find an overlapping area. Wider bases lead to smaller numbers of congruent sets, and thus the algorithm runs faster and more robustly [1].

5.2.4.2 Congruent set extraction

The modification to the pair extraction stage consists in performing it only once, instead of twice as in the Super4PCS algorithm, and by using 5 rigid invariants (a distance plus 4 angles), instead of 2, to constrain the search. This happens because we are working with a 2-point base, instead of a 4-point base that is decomposed into two pairs of points. It is important to notice that, while it is not required to include the normals of the points as input to the Super4PCS algorithm, when this information is available, the method uses it to further constrain the search. This is indicated with parentheses in Figure 5.2a.

In Super4PCS, the pair extraction stage runs in $\mathcal{O}(N)$ time, where N is the number of points in the target PC $\mathcal{Q}$. Since our method performs the pair extraction process only once, it has half the computational complexity of Super4PCS, running in half the time.

Besides extracting pairs in the target $\mathcal{Q}$, Super4PCS has a subsequent stage for finding congruent sets. This is necessary because generating candidate quadrilaterals based solely on the distance (and possibly the angle between normals) invariant would produce sets of congruent bases with many candidates not related by rigid transformations. This congruent set extraction stage has a runtime complexity of $\mathcal{O}(M + K)$ which is not included in our algorithm.

5.2.4.3 Estimation of R

The final modification performed to Super4PCS is the process of estimating the rigid rotation R. In the original algorithm, this is performed with Horn's method [55] using 3 out of the 4 points of the quadrilateral and provides a unique solution. In our case there are two solutions as explained in Section 5.2.3. For the same number K of congruent bases, this would lead to twice the computational complexity. However, due to the significant decrease in complexity in the other stages of the pipeline, the final runtime is still significantly lower. Also, since we are working with 2-point bases, wider bases can be used, yielding less candidates. Experiments in the next section show how our method is able to achieve similar results to Super4PCS while being approximately $3.6\times - 5.2\times$ faster when using Kinect scans and even achieving speed ups of two orders of magnitude in noise-free datasets.

5.3 Experiments

This section reports a set of experiments performed on several point clouds with different sizes, percentages of overlap and levels of noise. The performance of our proposed method is assessed in terms of speed and robustness, and compared with the state-of-the-art method Super4PCS [79] that served as a starting point for our algorithm. The first set of experiments was performed on the models shown in Figure 5.5 and we report both the alignments obtained for a search limited in time, which is relevant for applications with real-time requirements, and the best possible results when the execution time is not constrained. Since there is ground truth, we show a quantitative evaluation. The second set of experiments consisted in aligning several scans acquired with a Kinect camera, with both methods, and the qualitative results are shown. All these datasets were downloaded from [80].

We used the C++ source code for the Super4PCS algorithm available in [80] and made the necessary modifications to implement our method. For all point clouds, the normals to each point were computed using the PlanePCA algorithm described in Section 5.2.1 with a neighbourhood of 20 points. All tests were performed on a AMD Quad-Core Processor A6-3400M with a speed of 2.30 GHz and 6GB of RAM.

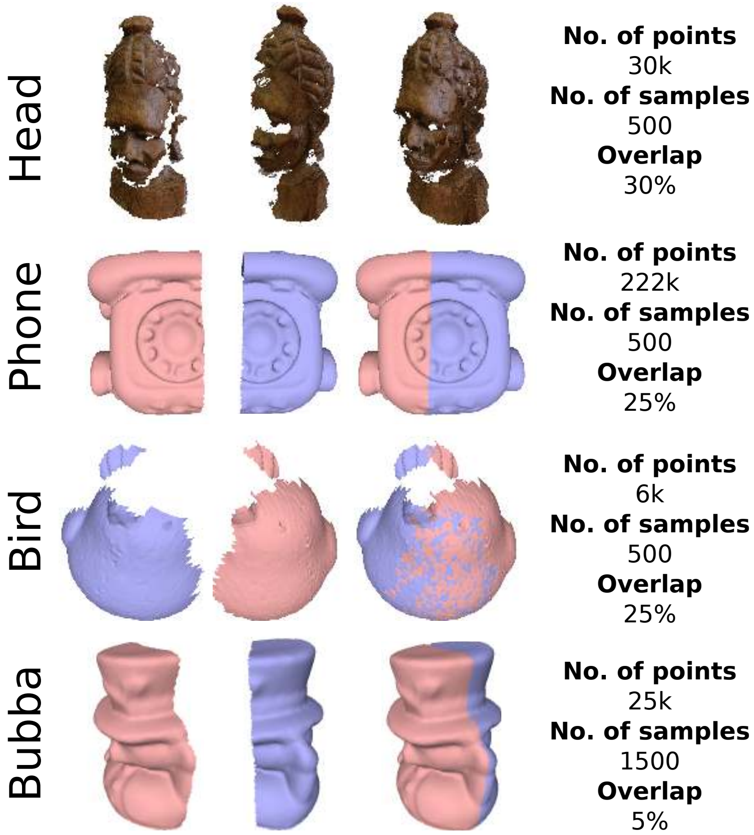
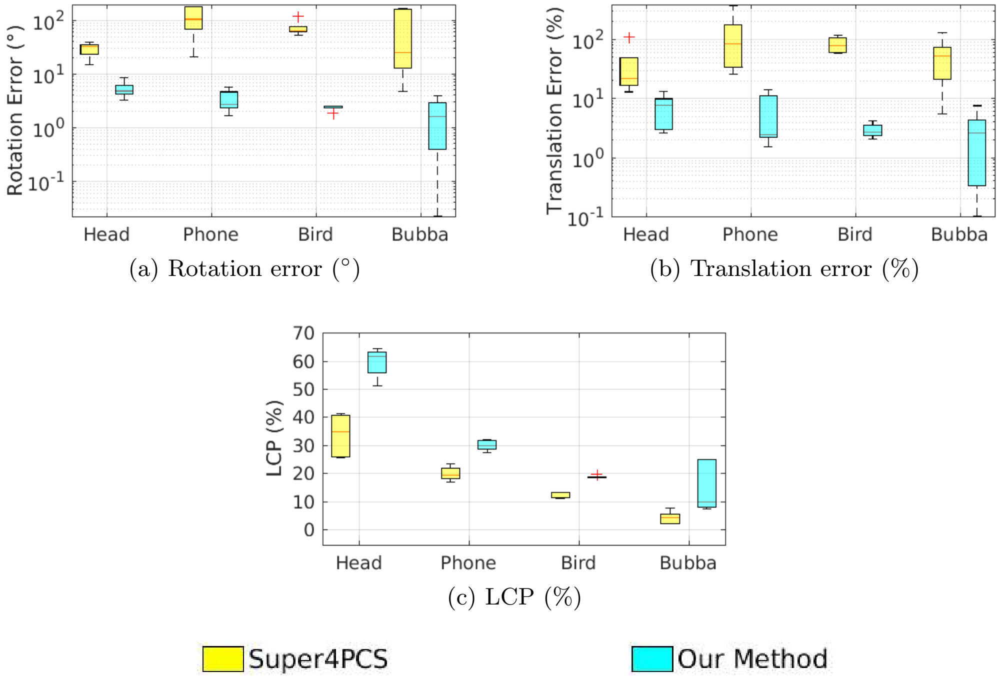


Figure 5.5: Different models with small overlap ($\leq$ 30%) used for testing our proposed method and the Super4PCS algorithm. All datasets were downloaded from [80] and contain ground truth.

5.3.1 Quantitative evaluation

The first set of experiments consisted in performing the alignment of the 4 models in Figure 5.5 with both our method and Super4PCS, for 10 random initial positions of the point clouds. As explained in [1], since the LCP measure is being used as the metric for selecting the best alignment hypothesis, Super4PCS and 2PNS allow a very heavy sampling of the point clouds. The approximate size of the point clouds and the number of samples used are shown in Figure 5.5. We show quantitative results as the angular magnitude of the residual rotation between the ground truth and the estimated rotations, e_{R}, in degrees, and the norm of the difference between the ground truth and the obtained translation vectors, $e_{\mathbf{t}}$, in percentage. We started by analysing how well both methods perform when the maximum execution time is limited. We used a threshold of 1 s for the head and bird datasets, 2 s for the phone



(a) Rotation error (°)

(b) Translation error (%)

(c) LCP (%)

Figure 5.6: Results obtained with our and the Super4PCS methods for the 4 models in Figure 5.5 by limiting the maximum execution time as follows: Head - 1 s, Phone - 2 s, Bird - 1 s, Bubba - 3 s.

and 3 s for the bubba point set, as they have decreasing percentages of overlap.

Figure 5.6 shows the distributions of the best LCPs achieved by RANSAC in each run, and the rotation and translation errors for all models, without any ICP refinement. The first observation is that, in the same amount of time, our method significantly outperforms Super4PCS, providing much smaller rotation and translation errors and larger LCPs. More importantly, unlike Super4PCS, our method was able to provide acceptable results as can be seen in Figures 5.6a and 5.6b that the maximum median rotation and translation errors are below 5° and 5%, respectively. Also, it never diverged, with the maximum errors being $e_{\mathsf{R}} = 8.4°$ and $e_{\mathbf{t}} = 13.8\%$.

In Figure 5.7, we show examples of the alignments obtained with Super4PCS and our method in the experiment with limited time, for all models. The point clouds, in arbitrary initial positions, are accurately aligned when using our method, which significantly facilitates a subsequent step of ICP refinement. On the other hand, the alignment provided by Super4PCS is very poor, as initially shown in the boxplots of Figure 5.6, not being sufficiently accurate for a subsequent refinement

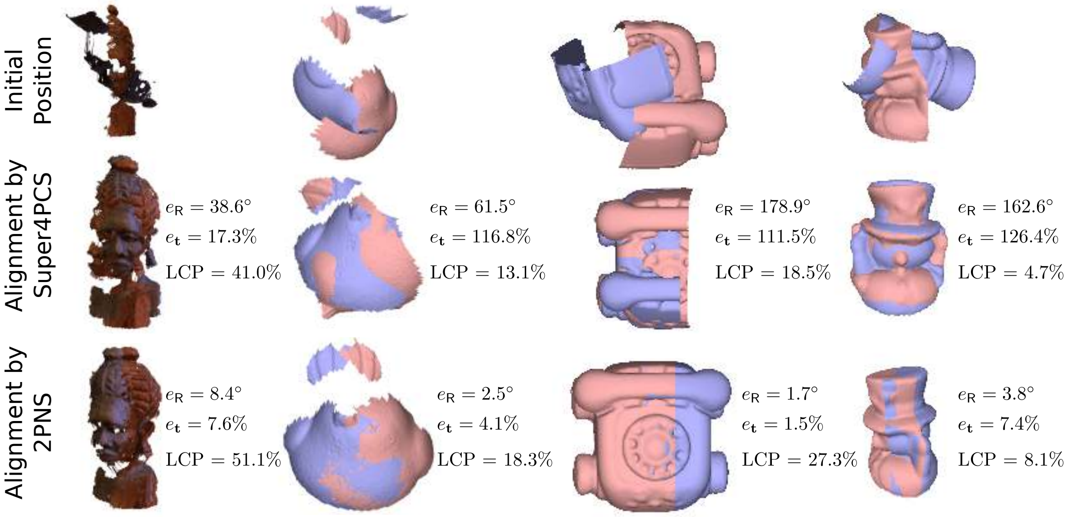
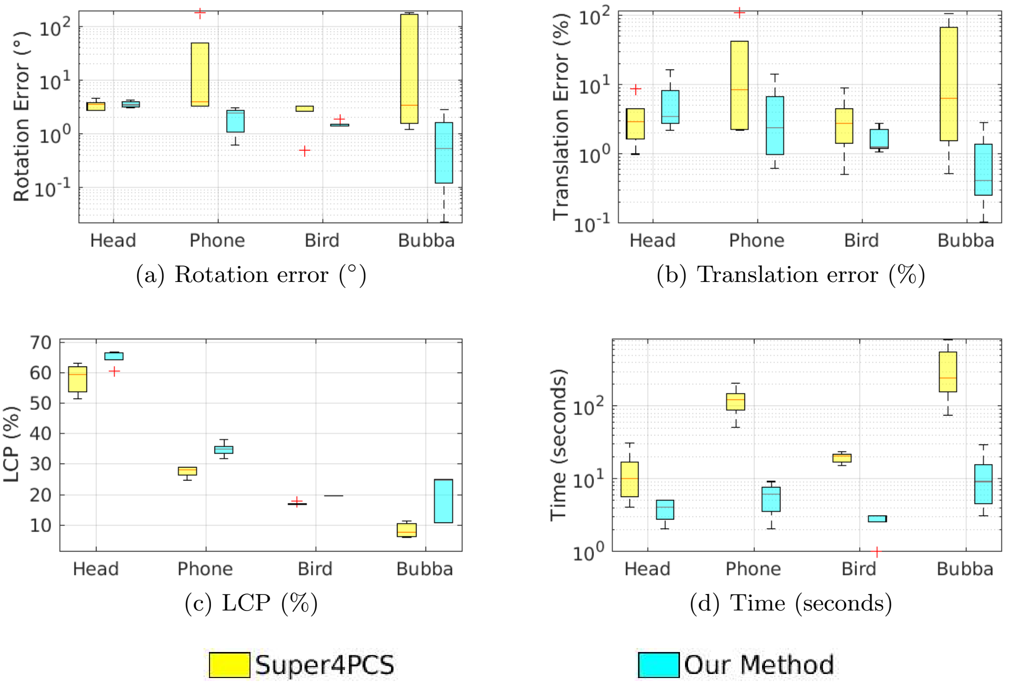


Figure 5.7: Alignment results and corresponding LCP and rotation and translation errors (without ICP) obtained with Super4PCS and our method 2PNS in the experiment with limited time.

to be applied. This experiment demonstrates that in applications that require fast processing, our method can be used as it very quickly provides sufficiently good results for a refinement step to converge to the global minimum.

In order to evaluate the best performance in terms of accuracy of alignment that both our and the Super4PCS methods are able to achieve, we removed the time limit and tested the methods with the same parameters as in the first part of this experiment. Results are shown in Figure 5.8 and include not only rotation and translation errors and the best LCP, but also the distribution of the execution times.

As expected, the accuracy of Super4PCS dramatically increased, reaching median errors of less than 4° and 10% in rotation and translation, respectively, as opposed to errors over 100° and 200% that were obtained in some cases of the first part of this experiment. The increase in the LCP is coherent with this decrease in the registration errors. However, despite running for over 15 minutes, Super4PCS still did not manage to converge to a good solution in a few initial poses of the phone and bubba datasets due to the very small overlap. As an example, the divergence cases for the bubba dataset corresponded to alignments similar to the one shown in Figure 5.7, due to the symmetric nature of the model that yields a local minimum. On the other hand, our method was able to achieve good solutions in all cases in less than 30 seconds for the bubba dataset and less than 10 seconds for the other ones. For the less problematic datasets head and bird, both methods performed well, with



(a) Rotation error (°)

(b) Translation error (%)

(c) LCP (%)

(d) Time (seconds)

Figure 5.8: Best possible results obtained with our and the Super4PCS methods using the same configuration parameters as the experiment corresponding to Figure 5.6, but without setting a maximum execution time threshold.

our method being able to find slightly better solutions (with larger LCPs) in about 30% and 13% of the time, respectively.

In order to estimate the speed ups achieved by our method, for the sake of fairness, we measured the time it requires to reach solutions as good as Super4PCS's, *i.e.* to achieve values of LCP equal to the ones obtained by Super4PCS. Table 5.1 shows these values, as well as the ones computed using the time corresponding to the best results that are shown in Figure 5.8. For the bubba and phone datasets, we were able to achieve speed-ups of 117× and 58×, respectively. The reason for these very high speed-ups is that these are synthetic noise-free datasets, and thus we can be very restrictive in selecting the threshold for extracting congruent pairs, leading to very few high-quality candidates for providing alignment hypotheses, and thus significantly decreasing computational time. For the bird and head datasets, our algorithm runs 19.2× and 7.8× faster, respectively, which is significantly better than the speed up reported for the Super Generalized 4PCS method [85] that is between 1.3× and 6.5×. Remark that the results provided in [85] are only for datasets with

Table 5.1: Speed-ups obtained by out method w.r.t. Super4PCS, for each model, by measuring the computational time to achieve both the best possible LCP, corresponding to the results in Figure 5.8, and the same LCP as Super4PCS.

	Head	Bird	Phone	Bubba
Best LCP	3.6×	7.4×	22.8×	30.8×
Same LCP as Super4PCS	7.8×	19.2×	58.1×	117×

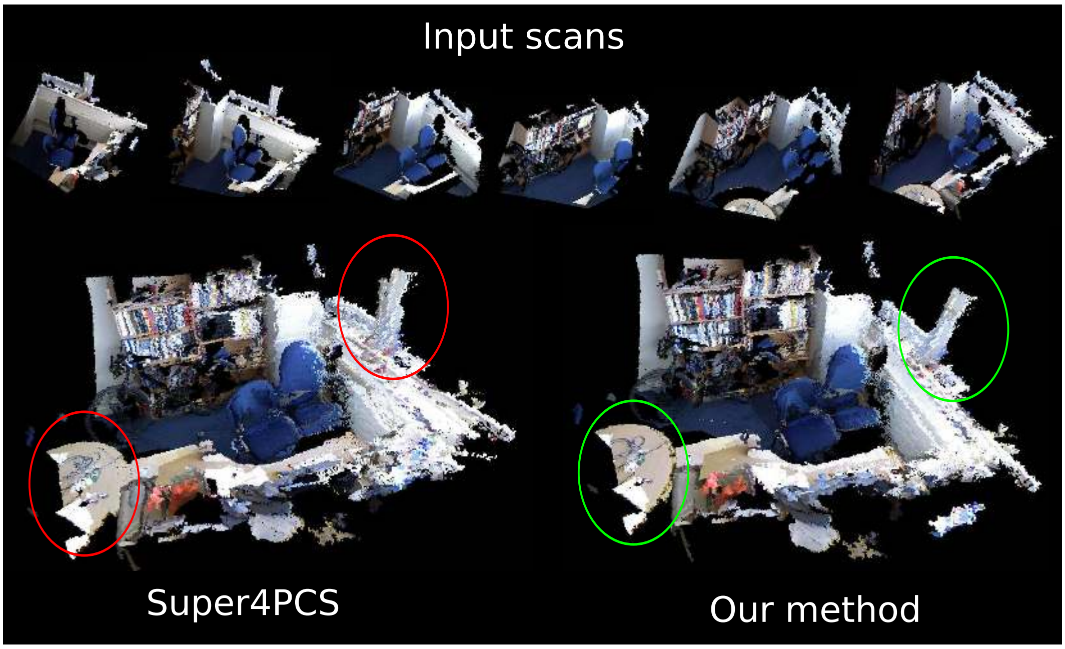
more than 30% of overlap, suggesting that the method is not able to perform well for smaller overlaps. Also, for some models shown in [85], Super4PCS outperforms Super Generalized 4PCS, whereas our method is always superior. This indirect comparison to Super Generalized 4PCS shows the superiority of our approach w.r.t. both state-of-the-art methods.

To conclude, this experiment demonstrates the importance of our new method as it shows that it is able to achieve similar or even better results than Super4PCS in a fraction of the time. Also, by limiting the execution time, while Super4PCS clearly fails, our method still performs well, with only a slight decrease in accuracy. This means that our method is very fast in finding a proper solution and increasing the execution time simply leads to more accuracy, which may not be crucial since the method performs coarse alignment and should be followed by a refinement step.

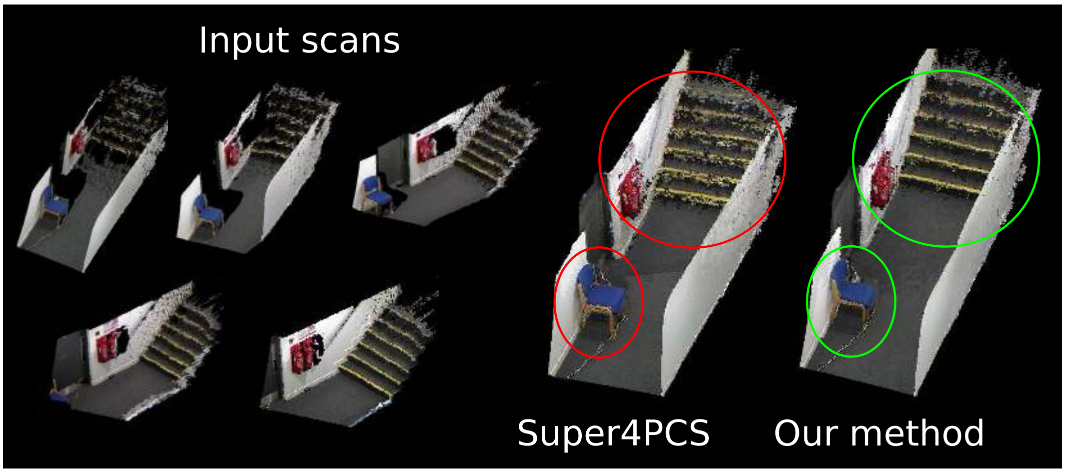
5.3.2 Experiments on Kinect scans

The second set of experiments consisted in registering several scans acquired by a Kinect of an office and a hall. Since there is no ground truth, we show the alignments obtained with Super4PCS and our method in Figure 5.9. For the sake of fairness, we used the same degree of sampling in both methods and did not limit the execution time. Also, all results are shown without any ICP refinement.

Overall, both methods were able to correctly align all the Kinect scans in the two examples of Figure 5.9, with our method being slightly more accurate, especially in the hall sequence (Figure 5.9b), as shown in the areas identified with ellipses. Regarding the office dataset in Figure 5.9a, while Super4PCS took approximately 36 s to align all 6 scans, our method was 3.6 times faster, requiring only 10 s. A better alignment can be observed in the table and window areas in the output provided by our algorithm. In the 5-scan hall example (Figure 5.9b), our method also provides an overall more accurate alignment not only in the stairs and the fire extinguisher areas but also in the other end of the point cloud, near the chair and the wire on the floor. Super4PCS and our method took 57 s and 11 s, respectively, to align the



(a) Registration of 6 Kinect scans acquired in an office



(b) Registration of 5 Kinect scans acquired in a hall

Figure 5.9: Registration results obtained using Super4PCS and our method on two datasets with 5 and 6 Kinect scans. The total execution times were (a) Super4PCS - 36s, our method - 10s, and (b) Super4PCS - 57s, our method - 11s. Registration is performed for every pair of consecutive input scans. The sequences of scans are sorted from left to right and top to bottom.

5 scans, corresponding to a speed up of 5.2×. The state-of-the-art algorithm Super Generalized 4PCS [85] reports a speed up of 4× w.r.t. Super4PCS in a sequence of

scans acquired with the Kinect. Again, our method is superior since we are able to perform 5.2× faster.

Although our method still performs significantly faster than Super4PCS on these real datasets, it can be noticed that the decrease in computational time is not as evident as in the experiments from Section 5.3.1. This is due to the fact that, unlike most of the models used in the first set of experiments, these Kinect scans are noisy and possibly contain outliers, providing a less accurate estimation of the normals. Thus, the angular threshold used for extracting congruent sets has to be relaxed, leading to more alignment hypotheses and hence a higher computational time. By achieving speed ups of up to 5.2×, this experiment confirms that including normals in the estimation of rigid transformations is very beneficial since very accurate results are obtained while significantly decreasing computational time. Since there are established algorithms for normal estimation, as long as the point cloud is sufficiently dense (as happens in the case of scans acquired with depth cameras), we believe there is no obvious reason for performing the registration using only points.

5.4 Conclusions

In this chapter we propose 2PNS, a new method that significantly advances the state-of-the-art in terms of global 3D registration of point clouds in arbitrary initial poses. Due to the important improvements provided by the recent algorithm Super4PCS [79] w.r.t. previous methods, our modifications to this method yielded a very fast approach that is able to register 3D point clouds in arbitrary positions and with small overlap. Experiments show that our method performs better than Super4PCS, providing more accurate alignments, in approximately 1/5 of the time when working with scans acquired by depth cameras that are contaminated by noise and outliers.

As future work, we intend to study until which extent working simultaneously with multiple bases randomly extracted from the source point cloud is beneficial. The idea is that, since there would be multiple hypotheses for the base, computational time would not be wasted in testing all the sets congruent to a base that is noisy or does not correspond to an overlapping area. We believe that this can provide a significant increase in computational efficiency.

Chapter 6

Conclusions

In computer vision, the problems of calibration, SfM and registration are commonly solved using the most fundamental primitive: keypoints. The main goal of this thesis is to fulfill these tasks using alternative primitives, such as planes, affine regions or normals, motivated by the various advantages this brings.

In Chapter 2, we propose to calibrate a camera-depth sensor pair using planes. This is accomplished through the registration of a set of corresponding planes that span the entire 3D space. We developed a new minimal solution for this problem that provides better initializations of the extrinsic calibration, when compared to competing methods, facilitating subsequent steps. We also proposed a new optimization step that prevents the calibration from drifting in scale, as well as a method for estimating a depth distortion model which significantly improves the calibration accuracy. All these contributions led to a fast and accurate calibration approach that outperforms the state-of-the-art methods when working with small datasets. The ability to perform accurate calibration of a camera-depth sensor pair from a small number of images motivated the development of a more general method that is able to calibrate heterogeneous sensor arrangements comprising cameras, LRFs and depth sensors in non-overlapping configurations. Since the extrinsic calibration in case of non-overlapping FoV is accomplished from mirror reflections, it is crucial that the conventional extrinsic calibration methods use a small number of object images, otherwise the total number of frames to be acquired becomes prohibitive. Our new calibration method is tested successfully in an application scenario for which there is no simple solution in the current state-of-the-art.

Chapter 3 also explores the use of planes in a different computer vision problem: SfM and 3D reconstruction. Plane-based registration is advantageous with respect to point-based registration because: (i) plane-primitives have a more global character, which helps avoiding local minima issues, (ii) scenes are often dominated by large planes, which allow correspondence between wide-baseline frames, (iii) plane primitives are typically in the static background, which improves odometry robustness to possible dynamic foreground, and (iv) the fact that the number of plane-features is much smaller than point-features favours faster correspondence and scalability under increasing image resolution. Motivated by all these advantages, we developed an hierarchical RANSAC-based scheme for camera motion estimation that favours the use of planes and only extracts point correspondences when the plane surface configuration is insufficient to determine motion with no ambiguity. This scheme is employed in two different pipelines that take as input a sequence of frames and output the camera motion and a reconstruction of the scene. While the first was developed for RGB-D cameras, where plane detection is relatively straightforward from the depth information, the second works with a calibrated stereo rig and plane detection is not trivial, requiring a more sophisticated approach such as [3]. Also, for refining the camera motion, the RGB-D pipeline uses simple optimization scheme based on minimizing the photo-geometric error. On the other hand, the stereo pipeline jointly refines the motion and the structure within a PEaRL framework [58] that alternates between discrete optimization to enforce coherent PPR across stereo frames, and continuous bundle adjustment to improve the accuracy of the results. The final dense labelling is accomplished by a new procedure that enforces global consistency, allowing cases of poor reconstruction and occlusions to be corrected. Both the RGB-D and the stereo pipelines have proven to be advantageous in cases of wide-baseline, dynamic foreground, weak texture when compared to state-of-the-art point-based approaches.

Despite the advances provided by these two pipelines, there is room for improvement. In future work, we intend to improve our RGB-D algorithm for dealing with partial plane occlusions by using a robust cost function in the optimization, and by including motion priors to increase the performance in small baseline situations. Regarding the stereo pipeline, since we are using a straightforward MATLAB implementation, its current runtime is prohibitive for real-time applications. Thus, we intend to develop a parallel version of the pipeline to be ran in the GPU (note that the initial PPR is computed for each stereo rig independently) in order to decrease computational time.

The new primitive that is explored in Chapter 4 is affine regions. Hypothesize-and-test frameworks for robustly determining the camera pose and scene structure

(SfM), either by homography or epipolar geometry estimation, have well established minimal solutions in the state-of-the-art. The importance of using minimal algorithms, as opposed to over-determined solutions, is to reduce the combinatorics of the hypothesize-and-test stage. Although the existing solutions have led to several successful systems, we argue in Chapter 4 that the combinatorics is still too high in certain applications and show that both homography and epipolar geometry estimation can be accomplished from as few as 2 ACs. The geometric insights led to approaches that are successfully applied in planar segmentation and homography estimation tasks, as well as in conventional SfM. Motivated by the good performance of our new plane segmentation scheme, we proposed the first feature-based pipeline for vSLAM and dense PPR from a monocular sequence based on ACs, named πMatch. It contains two consecutive PEaRL steps, where the first selects the motions present in the image and the second performs plane merging, yielding good PPRs of the scene from two views. Experiments show the superiority of our approach when compared to other competing monocular systems.

A large-scale experiment shows that scale drift may occur due to a few poor estimations of the scale of translation. To overcome this issue, we intend to use plane correspondences across frames since they are more constant over time than points. Also, we expect to achieve an execution rate of about $5 - 10$ fps after implementing the pipeline in C++.

Finally, Chapter 5 investigates the use of normals in 3D point cloud coarse registration. The proposed method builds on a recent point-based pipeline that uses smart indexing for extracting pairs of points in a point cloud [79]. Our method, that is significantly simplified due to the use of only pairs of points, instead of quadrilaterals as in [79], greatly benefits from this pair extraction scheme, being very fast and robust to small overlap. Experiments show that our method is as accurate as [79], taking at most 1/5 of the time. There are still possible improvements that can be applied to this method. As an example, we intend to analyse if it is advantageous to work with multiple pairs extracted from the source point cloud simultaneously, such that computational time is not spent in testing all the sets congruent to a base that is noisy or does not correspond to an overlapping area.

Bibliography

[1] Dror Aiger, Niloy J. Mitra, and Daniel Cohen-Or. 4-points congruent sets for robust pairwise surface registration. In *ACM SIGGRAPH 2008 Papers*, SIGGRAPH '08, pages 85:1–85:10, New York, NY, USA, 2008. ACM.

[2] P.F. Alcantarilla, C. Beall, and F. Dellaert. Large-scale dense 3d reconstruction from stereo imagery. In *5th Workshop on Planning, Perception and Navigation for Intelligent Vehicles (PPNIV13)*, 2013.

[3] M. Antunes and J. P. Barreto. Semi-dense piecewise planar stereo reconstruction using symstereo and pearl. In *3DimPVT*, 2012.

[4] Michel Antunes and J. P. Barreto. Symstereo: Stereo matching using induced symmetry. *International Journal of Computer Vision*, 2014.

[5] Michel Antunes, J. P. Barreto, and Urbano Nunes. Piecewise-planar reconstruction using two views. *Image and Vision Computing*, 46:47 – 63, 2016.

[6] E. Auvinet, J. Meunier, and F. Multon. Multiple depth cameras calibration and body volume reconstruction for gait analysis. In *Information Science, Signal Processing and their Applications (ISSPA), 2012 11th International Conference on*, July 2012.

[7] Hernn Badino, Daniel F. Huber, Yongwoon Park, and Takeo Kanade. Fast and accurate computation of surface normals from range images. In *ICRA*, pages 3084–3091. IEEE, 2011.

[8] Simon Baker, Ankur Datta, and Takeo Kanade. Parameterizing Homographies. Technical report, Robotics Institute-CMU, 2006.

[9] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *IJCV*, 56(3):221 – 255, March 2004.

[10] A Bartoli and P. Sturm. Multiple-view structure and motion from line correspondences. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 207–212 vol.1, Oct 2003.

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

[12] Jacob Bentolila and Joseph M. Francos. Conic epipolar constraints from affine correspondences. *Computer Vision and Image Understanding*, 122(0):105 – 114, 2014.

[13] Jacob Bentolila and JosephM. Francos. Homography and fundamental matrix estimation from region matches using an affine error metric. *Journal of Mathematical Imaging and Vision*, 49(2):481–491, 2014.

[14] P.J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 14(2):239–256, Feb 1992.

[15] Yunsu Bok, Dong-Geol Choi, P. Vasseur, and In So Kweon. Extrinsic calibration of non-overlapping camera-laser system using structured environment. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 436–443, Sept 2014.

[16] Sofien Bouaziz, Andrea Tagliasacchi, and Mark Pauly. Sparse iterative closest point. *Computer Graphics Forum (Symposium on Geometry Processing)*, 32(5):1–11, 2013.

[17] J. Y. Bouguet. Camera calibration toolbox, 2010.

[18] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE TPAMI*, (11), November 2001.

[19] C. Bregler, A Hertzmann, and H. Biermann. Recovering non-rigid 3d shape from image streams. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 690–696 vol.2, 2000.

[20] Nicholas Burrus. Kinect calibration, 2011. `http://nicolas.burrus.name/index.php/Research/KinectCalibration`.

[21] M. Chandraker, Jongwoo Lim, and D. Kriegman. Moving in stereo: Efficient structure and motion using lines. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1741–1748, Sept 2009.

[22] C. Choi, Y. Taguchi, O. Tuzel, M. Y. Liu, and S. Ramalingam. Voting-based pose estimation for robotic assembly using a 3d sensor. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1724–1731, May 2012.

[23] A. Concha and J. Civera. Dpptam: Dense piecewise planar tracking and mapping from a monocular sequence. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 5686–5693, Sept 2015.

[24] A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(6):1052–1067, June 2007.

[25] Andrew Delong, Anton Osokin, HossamN. Isack, and Yuri Boykov. Fast approximate energy minimization with label costs. *International Journal of Computer Vision*, 96(1):1–27, 2012.

[26] B. Douillard, D. Fox, F. Ramos, and H. Durrant-Whyte. Classification and semantic mapping of urban environments. In *Int. J. Rob.*, 2011.

[27] B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model globally, match locally: Efficient and robust 3d object recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 998–1005, June 2010.

[28] E. Dunn, B. Clipp, and J.-M. Frahm. A geometric solver for calibrated stereo egomotion. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1187–1194, Nov 2011.

[29] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard. An evaluation of the RGB-D SLAM system. In *IEEE-ICRA*, May 2012.

[30] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *eccv*, September 2014.

[31] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1449–1456, Dec 2013.

[32] A. Ess, B. Leibe, K. Schindler, and L. van Gool. A mobile vision system for robust multi-person tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*. IEEE Press, June 2008.

[33] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[34] Friedrich Fraundorfer, Petri Tanskanen, and Marc Pollefeys. A minimal case solution to the calibrated relative pose problem for the case of two known orientation angles. In *Computer Vision - ECCV 2010*, volume 6314 of *Lecture Notes in Computer Science*, pages 269–282, 2010.

[35] Brendan J. Frey. Affinity propagation. `http://www.psi.toronto.edu/index.php?q=affinity%20propagation`.

[36] Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315:972–976, 2007.

[37] Y. Furukawa, B. Curless, S.M. Seitz, and R. Szeliski. Manhattan-world stereo. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1422–1429, June 2009.

[38] Y. Furukawa, B. Curless, S.M. Seitz, and R. Szeliski. Reconstructing building interiors from images. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 80–87, Sept 2009.

[39] D. Gallup, J.-M. Frahm, P. Mordohai, and M. Pollefeys. Variable baseline/resolution stereo. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008.

[40] D. Gallup, J.-M. Frahm, and M. Pollefeys. Piecewise planar and non-planar stereo for urban scene reconstruction. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1418–1425, June 2010.

[41] Ravi Garg, Anastasios Roussos, and Lourdes Agapito. Dense variational reconstruction of non-rigid surfaces from monocular video. June 2013.

[42] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[43] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[44] Andreas Geiger, Julius Ziegler, and Christoph Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *Intelligent Vehicles Symposium (IV)*, 2011.

[45] W.E.L. Grimson and T. Lozano-Pérez. Model-based recognition and localization from sparse range or tactile data. *International Journal of Robotics Research*, 3(3):3–35, 1984.

[46] BertM. Haralick, Chung-Nan Lee, Karsten Ottenberg, and Michael Nlle. Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 1994.

[47] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision. 2nd edn.* Cambridge University Press, 2004.

[48] R.I. Hartley. In defense of the eight-point algorithm. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(6):580–593, Jun 1997.

[49] J. Heikkilä. Geometric camera calibration using circular control points. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(10), 2000.

[50] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using kinect-style depth cameras for dense 3D modeling of indoor environments. *International Journal of Robotics Research (IJRR)*, 31(5):647–663, April 2012.

[51] Peter Henry, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox. Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In *ISER*, volume 20, pages 22–25, 2010.

[52] Daniel Herrera, Juho Kannala, and Janne Heikkilä. Accurate and practical calibration of a depth and color camera pair. In *Proceedings of the 14th international conference on Computer analysis of images and patterns - Volume Part II*, CAIP'11, Berlin, Heidelberg, 2011.

[53] Daniel Herrera C., Juho Kannala, and Janne Heikkilä. Joint depth and color camera calibration with distortion correction. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(10), 2012.

[54] J. Hesch, A. Mourikis, and S. Roumeliotis. Determining the camera to robot-body transformation from planar mirror reflections. In *IROS*, 2008.

[55] Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am. A*, 4(4), Apr 1987.

[56] A. Howard. Real-time stereo visual odometry for autonomous ground vehicles. In *IEEE-IROS*, pages 3946–3952, 2008.

[57] Du Q. Huynh. Metrics for 3d rotations: Comparison and analysis. *J. Math. Imaging Vis.*, 35(2):155–164, October 2009.

[58] Hossam Isack and Yuri Boykov. Energy-based geometric multi-model fitting. *IJCV*, 2012.

[59] O. Javed, Z. Rasheed, O. Alatas, and M. Shah. Knight: a real time surveillance system for multiple and non-overlapping cameras. In *ICME*, 2013.

[60] K. Jordan and P. Mordohai. A quantitative evaluation of surface normal estimation in point clouds. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4220–4226, Sept 2014.

[61] Yun-Suk Kang and Yo-Sung Ho. High-quality multi-view depth generation using multiple color and depth cameras. In *Multimedia and Expo (ICME)*, 2010.

[62] T. Kazik, L. Kneip, J. Nikolic, M. Pollefeys, and R. Siegwart. Real-time 6d stereo visual odometry with non-overlapping fields of view. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1529–1536, June 2012.

[63] C. Kerl, J. Sturm, and D. Cremers. Robust odometry estimation for rgb-d cameras. In *IEEE-ICRA*, May 2013.

[64] K. Klasing, D. Althoff, D. Wollherr, and M. Buss. Comparison of surface normal estimation methods for range sensing applications. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 3206–3211, May 2009.

[65] Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[66] Kurt Konolige, Motilal Agrawal, Robert C. Bolles, Cregg Cowan, Martin Fischler, and Brian Gerkey. Outdoor mapping and navigation using stereo vision. In *ISER*, 2006.

[67] K. Koser, C. Beder, and R. Koch. Conjugate rotation: Parameterization and estimation from an affine feature correspondence. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008.

[68] Kevin Koser. *Geometric estimation with local affine frames and free-form surfaces.* PhD thesis, University of Kiel, 2009. http://d-nb.info/994782322.

[69] Kevin Koser and Reinhard Koch. Differential spatial resection - pose estimation using a single local image feature. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision - ECCV 2008*, volume 5305 of *Lecture Notes in Computer Science*, pages 312–325. Springer Berlin Heidelberg, 2008.

[70] R. Kumar, A. Ilie, J. M. Frahm, and M. Pollefeys. Simple calibration of non-overlapping cameras with a mirror. In *CVPR*, 2008.

[71] Nevena Lazic, Brendan J. Frey, and Parham Aarabi. Solving the uncapacitated facility location problem using message passing algorithms. *Journal of Machine Learning Research*, 2010.

[72] Xavier Llad, Alessio Del Bue, Arnau Oliver, Joaquim Salvi, and Lourdes de Agapito. Reconstruction of non-rigid 3d shapes from stereo-motion. *Pattern Recognition Letters*, 32(7):1020–1028, 2011.

[73] Miguel Lourenco, Danail Stoyanov, and Joao P. Barreto. Visual odometry in stereo endoscopy by using pearl to handle partial scene deformation. In *Augmented Environments for Computer-Assisted Interventions*, volume 8678 of *Lecture Notes in Computer Science*, pages 33–40. Springer International Publishing, 2014.

[74] Yi Ma, Stefano Soatto, Jana Kosecka, and S. Shankar Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models.* SpringerVerlag, 2003.

[75] Luca Magri and Andrea Fusiello. T-linkage: A continuous relaxation of j-linkage for multi-model fitting. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 0:3954–3961, 2014.

[76] G.L. Mariottini, S. Scheggi, F. Morbidi, and D. Prattichizzo. Planar mirrors for image-based robot localization and 3-d reconstruction. *Mechatronics, "Special Issue on Visual Servoing"*, 22(4):398 – 409, 2012.

[77] Pavlos Mavridis, Anthousis Andreadis, and Georgios Papaioannou. Efficient sparse {ICP}. *Computer Aided Geometric Design*, 35?36:16 – 26, 2015. Geometric Modeling and Processing 2015.

[78] C. Mei, S. Benhimane, E. Malis, and P. Rives. Efficient homography-based tracking and 3-d reconstruction for single-viewpoint sensors. *IEEE-TRO*, 24(6):1352–1364, Dec. 2008.

[79] Nicolas Mellado, Dror Aiger, and Niloy J. Mitra. Super 4pcs fast global point-cloud registration via smart indexing. *Computer Graphics Forum*, 33(5):205–215, 2014.

[80] Nicolas Mellado, Dror Aiger, and Niloy J. Mitra. Super 4pcs fast global pointcloud registration via smart indexing. `http://geometry.cs.ucl.ac.uk/projects/2014/super4PCS/`, 2014.

[81] B. Micusik and J. Kosecka. Piecewise planar city 3d modeling from street view panoramic sequences. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2906–2912, June 2009.

[82] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A Comparison of Affine Region Detectors. *Int. J. Comput. Vision*, 65:2005, 2005.

[83] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE-TPAMI, Trans. On Patt. Analysis and Mach. Intell.*, 27, October 2005.

[84] N. J. Mitra, N. Gelfand, H. Pottmann, and L. Guibas. Registration of point cloud data from a geometric optimization perspective. In *Symposium on Geometry Processing*, pages 23–31, 2004.

[85] M. Mohamad, M. T. Ahmed, D. Rappaport, and M. Greenspan. Super generalized 4pcs for 3d registration. In *3D Vision (3DV), 2015 International Conference on*, pages 598–606, Oct 2015.

[86] Richard A. Newcombe, Steven Lovegrove, and Andrew J. Davison. Dtam: Dense tracking and mapping in real-time. In Dimitris N. Metaxas, Long Quan, Alberto Sanfeliu, and Luc J. Van Gool, editors, *ICCV*, pages 2320–2327. IEEE, 2011.

[87] T. Nguyen and G. Reitmayr. Calibrating setups with a single-point laser range finder and a camera. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1801–1806, Nov 2013.

[88] D. Nister. An efficient solution to the five-point relative pose problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(6):756–770, June 2004.

[89] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–652–I–659 Vol.1, June 2004.

[90] F. Pagel. Calibration of non-overlapping cameras in vehicles. In *Intelligent Vehicles Symposium (IV)*, 2010.

[91] Hyun Soo Park, Takaaki Shiratori, Iain Matthews, and Yaser Sheikh. 3d reconstruction of a moving point from a series of 2d projections. In *European Conference on Computer Vision*, September 2010.

[92] M. Perd'och, J. Matas, and O. Chum. Epipolar geometry from two correspondences. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 4, pages 215–219, 2006.

[93] Roman Pflugfelder and Horst Bischof. Localization and trajectory reconstruction in surveillance cameras with nonoverlapping views. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[94] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. Ieee, 2007.

[95] M. Pizzoli, C. Forster, and D. Scaramuzza. Remode: Probabilistic, monocular dense reconstruction in real time. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2609–2616, May 2014.

[96] F. Pomerleau, S. Magnenat, F. Colas, M. Liu, and R. Siegwart. Tracking a depth camera: Parameter exploration for fast icp. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3824–3829, Sept 2011.

[97] C. Premebida, O. Ludwig, and U. Nunes. Lidar and vision-based pedestrian detection system. In *Journal of Field Robotics*, 2009.

[98] Carolina Raposo, Michel Antunes, and J. P. Barreto. Piecewise-planar stereoscan: Structure and motion from plane primitives. In *Computer Vision - ECCV 2014*, Lecture Notes in Computer Science. Springer International Publishing, 2014.

[99] Carolina Raposo, João P. Barreto, and Urbano Nunes. Fast and accurate calibration of a kinect sensor. In *3DV*, 2013.

[100] Carolina Raposo, João P. Barreto, and Urbano Nunes. Extrinsic calibration of multi-modal sensors with non-overlapping field-of-view. In *Machine Vision and Applications*, under revision.

[101] Carolina Raposo and Joao P. Barreto. πmatch: Monocular vslam and piecewise planar reconstruction using fast plane correspondences. In *European Conference on Computer Vision (ECCV)*, 2016.

[102] Carolina Raposo and Joao P. Barreto. Theory and pratice of structure-from-motion using affine correspondences. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[103] Carolina Raposo and Joao P. Barreto. Using 2 point+normal sets for fast registration of point clouds with small overlap. In *IEEE International Conference on Robotics and Automation (ICRA) 2017*, submitted.

[104] Carolina Raposo and Joao P. Barreto. Piecewise-planar stereoscan: Structure and motion from plane primitives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, under revision.

[105] Carolina Raposo, Miguel Lourenco, Michel Antunes, and Joao P. Barreto. Plane-based odometry using an rgb-d camera. In *British Machine Vision Conference*, pages 1–11, September 2013.

[106] F. Riggi, M. Toews, and T. Arbel. undamental matrix estimation via tip - transfer of invariant parameters. In *Proceedings of the 18th International Conference on Pattern Recognition*, 2006.

[107] Rui Rodrigues, João P. Barreto, and Urbano Nunes. Camera pose estimation using images of planar mirror reflections. In *ECCV*, 2010.

[108] A Roussos, C. Russell, R. Garg, and L. Agapito. Dense multibody motion estimation and reconstruction from a handheld camera. In *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*, pages 31–40, Nov 2012.

[109] D. Scaramuzza, A. Harati, and R. Siegwart. Extrinsic self calibration of a camera and a 3d laser range finder from natural scenes. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 2007.

[110] K. Schenk, A. Kolarow, M. Eisenbach, K. Debes, and H. Gross. Automatic calibration of multiple stationary laser range finders using trajectories. In *Advanced Video and Signal-Based Surveillance (AVSS), 2012 IEEE Ninth International Conference on*, Sept 2012.

[111] Grant Schindler, Panchapagesan Krishnamurthy, and Frank Dellaert. Line-based structure from motion for urban environments. In *3DPVT*, pages 846–853. IEEE Computer Society, 2006.

[112] K. Schindler and D. Suter. Two-view multibody structure-and-motion with outliers through model selection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(6):983–995, June 2006.

[113] J. Serafin and G. Grisetti. Nicp: Dense normal based point cloud registration. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 742–749, Sept 2015.

[114] S.N. Sinha, D. Steedly, and R. Szeliski. Piecewise planar stereo for image-based rendering. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1881–1888, Sept 2009.

[115] J. Smisek, M. Jancosek, and T. Pajdla. 3d with kinect. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, 2011.

[116] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Modeling the world from internet photo collections. *Int. J. Comput. Vision*, 80(2):189–210, November 2008.

[117] S. Song and M. Chandraker. Robust scale estimation in real-time monocular sfm for autonomous driving. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1566–1573, June 2014.

[118] F. Steinbruecker, J. Sturm, and D. Cremers. Real-time visual odometry from dense rgb-d images. In *IEEE-ICCV Workshop on Live Dense Reconstruction with Moving Cameras*, 2011.

[119] H. Stewénius, C. Engels, and D. Nistér. Recent developments on direct relative orientation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60:284–294, June 2006.

[120] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *IEEE-IROS*, Oct. 2012.

[121] P. Sturm and T. Bonfort. How to compute the pose of an object without a direct view? In *Lecture Notes in Computer Science*, 2006.

[122] P. F. Sturm and S. J. Maybank. On plane-based camera calibration: A general algorithm, singularities, applications. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 1, page 437 Vol. 1, 1999.

[123] Yuichi Taguchi, Yong-Dian Jian, Srikumar Ramalingam, and Chen Feng. Slam using both points and planes for hand-held 3d sensors. In *Proceedings of the 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, ISMAR '12, pages 321–322, Washington, DC, USA, 2012. IEEE Computer Society.

[124] Yuichi Taguchi, Yong-Dian Jian, Srikumar Ramalingam, and Chen Feng. Point-plane slam for hand-held 3d sensors. In *ICRA*, 2013.

[125] Camillo J. Taylor and Anthony Cowley. Parsing indoor scenes using rgb-d imagery. In *RSS*, July 2012.

[126] Jonathan Taylor, Allan D. Jepson, and Kiriakos N. Kutulakos. Non-rigid structure from locally-rigid motion. In *CVPR*, pages 2761–2768. IEEE, 2010.

[127] Pascal Willy Theiler, Jan Dirk Wegner, and Konrad Schindler. Keypoint-based 4-points congruent sets - automated marker-less registration of laser scans. {*ISPRS*} *Journal of Photogrammetry and Remote Sensing*, 96:149 – 163, 2014.

[128] Pascal Willy Theiler, Jan Dirk Wegner, and Konrad Schindler. Globally consistent registration of terrestrial laser scans via graph optimization. {*ISPRS*} *Journal of Photogrammetry and Remote Sensing*, 109:126 – 138, 2015.

[129] Roberto Toldo and Andrea Fusiello. Robust multiple structures estimation with j-linkage. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, ECCV '08, pages 537–547, Berlin, Heidelberg, 2008. Springer-Verlag.

[130] P.H.S. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78(1):138 – 156, 2000.

[131] R. Tron and R. Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, June 2007.

[132] Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors: a survey. *Found. Trends. Comput. Graph. Vis.*, 3, 2008.

[133] T. Ueshiba and F. Tomita. Plane-based calibration algorithm for multi-camera systems via factorization of homography matrices. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 966–973 vol.2, Oct 2003.

[134] A Varol, M. Salzmann, Engin Tola, and P. Fua. Template-free monocular reconstruction of deformable surfaces. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1811–1818, Sept 2009.

[135] Francisco Vasconcelos, João P. Barreto, and Urbano Nunes. A minimal solution for the extrinsic calibration of a camera and a laser-rangefinder. In *TPAMI*, 2012.

[136] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`, 2008.

[137] Olga Veksler and Andrew Delong. Multi-label optimization. `http://vision.csd.uwo.ca/code/`.

[138] E. Wahl, U. Hillenbrand, and G. Hirzinger. Surflet-pair-relation histograms: a statistical 3d-shape representation for rapid classification. In *3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on*, pages 474–481, Oct 2003.

[139] Tomás Werner and Andrew Zisserman. New techniques for automated architectural reconstruction from photographs. In *Proceedings of the 7th European Conference on Computer Vision-Part II*, ECCV '02, pages 541–555, London, UK, UK, 2002. Springer-Verlag.

[140] Andrew D. Wilson and Hrvoje Benko. Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, 2010.

[141] H. Yamazoe, H. Habe, I. Mitsugami, and Y. Yagi. Easy depth sensor calibration. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, 2012.

[142] J. Yang, H. Li, D. Campbell, and Y. Jia. Go-icp: A globally optimal solution to 3d icp point-set registration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP(99):1–1, 2016.

[143] J. Yang, H. Li, and Y. Jia. Go-icp: Solving 3d registration efficiently and globally optimally. In *2013 IEEE International Conference on Computer Vision*, pages 1457–1464, Dec 2013.

[144] Hongbin Zha, Huijing Zhao, Jinshi Cui, Xuan Song, and Xianghua Ying. Combining laser-scanning data and images for target tracking and scene modeling. In *Robotics Research*, volume 70 of *Springer Tracts in Advanced Robotics*, pages 573–587. 2011.

[145] Qilong Zhang and R. Pless. Extrinsic calibration of a camera and laser range finder (improves camera calibration). In *IROS*, 2004.

[146] Zhengyou Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, 1999.

[147] M. Zuliani, C. S. Kenney, and B. S. Manjunath. The multiransac algorithm and its application to detect planar homographies. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 3, pages III–153–6, Sept 2005.