Predictive Monocular Odometry Using Propagation-based Tracking

DISSERTATION

zur Erlangung des Doktorgrades
der Naturwissenschaften

vorgelegt beim Fachbereich Informatik und Mathematik
der Johann Wolfgang Goethe-Universität
in Frankfurt am Main

von

Nolang Fanani

aus Kudus (Indonesien)

Frankfurt 2018
(D 30)

vom Fachbereich Informatik und Mathematik
der Johann Wolfgang Goethe-Universität
als Dissertation angenommen.

Dekan:	Prof. Dr. Andreas Bernig
Gutachter:	Prof. Dr. Rudolf Mester Prof. Dr. Matthias Kaschube
Datum der Disputation:	18.10.2018

(Printed and/or published with the support of the German Academic Exchange Service)

Abstract

The technology of advanced driver assistance systems (ADAS) has rapidly developed in the last few decades. The current level of assistance provided by the ADAS technology significantly makes driving much safer by using the developed driver protection systems such as automatic obstacle avoidance and automatic emergency braking. With the use of ADAS, driving not only becomes safer but also easier as ADAS can take over some routine tasks from the driver, e.g. by using ADAS features of automatic lane keeping and automatic parking. With the continuous advancement of the ADAS technology, fully autonomous cars are predicted to be a reality in the near future.

One of the most important tasks in autonomous driving is to accurately localize the ego-car and continuously track its position. The module which performs this task, namely odometry, can be built using different kinds of sensors: camera, LIDAR, GPS, etc. This dissertation covers the topic of visual odometry using a camera. While stereo visual odometry frameworks are widely used and dominating the KITTI odometry benchmark (Geiger, Lenz and Urtasun 2012), the accuracy and performance of monocular visual odometry is much less explored.

In this dissertation, a new monocular visual odometry framework is proposed, namely Predictive Monocular Odometry (PMO). PMO employs the prediction-and-correction mechanism in different steps of its implementation. PMO falls into the category of sparse methods. It detects and chooses keypoints from images and tracks them on the subsequence frames. The relative pose between two consecutive frames is first pre-estimated using the pitch-yaw-roll estimation based on the far-field view (Barnada, Conrad, Bradler, Ochs and Mester 2015) and the statistical motion prediction based on the vehicle motion model (Bradler, Wiegand and Mester 2015). The correction and optimization of the relative pose estimates are carried out by minimizing the photometric error of the keypoints matches using the joint epipolar tracking method (Bradler, Ochs, Fanani and Mester 2017).

The monocular absolute scale is estimated by employing a new approach to ground plane estimation. The camera height over ground is assumed to be known. The scale is first estimated using the propagation-based scale estimation. Both of the sparse matching and the dense matching of the ground features between two consecutive frames are then employed to refine the scale estimates. Additionally, street masks from a convolutional neural network (CNN) are also utilized to reject non-ground objects in the region of interest.

PMO also has a method to detect independently moving objects (IMO). This is important for visual odometry frameworks because the localization of the ego-car should be estimated only based on static objects. The IMO candidate masks are provided by a CNN. The case of crossing IMOs is handled by checking the epipolar consistency. The parallel-moving IMOs, which are epipolar conformant, are identified by checking the depth consistency against the depth maps from CNN.

In order to evaluate the accuracy of PMO, a full simulation on the KITTI odometry

dataset was performed. PMO achieved the best accuracy level among the published monocular frameworks when it was submitted to the KITTI odometry benchmark in July 2017. As of January 2018, it is still one of the leading monocular methods in the KITTI odometry benchmark.

It is important to note that PMO was developed without employing random sampling consensus (RANSAC) which arguably has been long considered as one of the irreplaceable components in a visual odometry framework. In this sense, PMO introduces a new style of visual odometry framework. PMO was also developed without a multi-frame bundle adjustment step. This reflects the high potential of PMO when such multi-frame optimization scheme is also taken into account.

Deutsche Zusammenfassung

Einleitung

Die Technologie der Fahrerassistenzsysteme (ADAS) hat sich in den letzten Jahrzehnten rasant entwickelt. Die gegenwärtige ADAS-Technologie erhöht erheblich die Sicherheit des Fahrens durch die Verwendung von Fahrerschutzsystemen, wie z.B. automatische Hindernisvermeidung und automatische Notbremsung. Über die reine Erhöhung der Sicherheit hinaus wird das Fahren durch die ADAS auch einfacher, da z.B. unter Benutzung der automatischen Spurhalte- und Parkfunktion einige Routine-Aufgaben des Fahrers übernommen werden können. Die kontinuierliche Weiterentwicklung der ADAS-Technologie lässt erwarten, dass in naher Zukunft vollständig autonome Fahrzeuge Realität werden.

Eine der wichtigsten Aufgaben im Bereich des autonomen Fahren ist es, das Ego-Fahrzeug exakt zu lokalisieren und seine Position kontinuierlich zu verfolgen. Dies wird vom Odometrie-Modul übernommen, welches wiederrum auf Basis verschiedener Sensoren konstruiert werden kann: Kamera, LIDAR, GPS, etc. Diese Dissertation behandelt hierbei die visuelle Odometrie auf Basis einer Kamera. Während häufig stereoskopische visuelle Odometrie-Frameworks angewandt werden und den KITTI-Odometrie-Benchmark (Geiger et al. 2012) dominieren, ist die Anwendbarkeit der monokularen visuellen Odometrie viel weniger erforscht.

Diese Dissertation umfasst den Vorschlag eines neuen monokularen und visuellen Odometrie-Frameworks mit der Bezeichnung Predictive Monocular Odometry (PMO). Das Framework verwendet den *prediction-correction mechanism* in verschiedenen Schritten seiner Implementierung. Die monokulare absolute Skala wird auf Basis der *Ground Plane Estimation* Methode ermittelt. Darüber hinaus verfügt das PMO über eine Methode zum Erkennen von sich *independently moving object* (IMO). Dies ist besonders wichtig für visuelle Odometrie-Frameworks, da die Position des Ego-Fahrzeug nur auf Basis von statischen Objekten geschätzt werden sollte. Des Weiteren ist es wichtig anzumerken, dass das PMO ohne den Einsatz von *Random-Sampling-Consensus* (RANSAC) entwickelt wurde, welche lange Zeit als eine der unersetzbaren Komponenten in einem visuellen Odometrie-Framework angesehen wurde. In diesem Sinne führt das PMO eine neue Art der visuellen Odometrie ein. Außerdem ist anzumerken, dass das PMO ohne den *Multi-frame Bundle Adjustment* Schritt entwickelt wurde. Dies spiegelt das hohe Potenzial von PMO wider, welches durch die zusätzliche Verwendung eines solchen Optimierungsschemas genutzt werden kann.

Verwandte Arbeiten

Das entwickelte Verfahren PMO weist im Vergleich zu anderen klassischen und modernen visuellen SLAM / Odometrie-Methoden sowohl einige Gemeinsamkeiten als auch Neuerungen auf. In diesem Abschnitt wird das PMO mit den anderen Methoden verglichen.

PMO gehört als *sparse* Methode zu der Gruppe von Methoden, die auch PTAM (Klein and Murray 2007) und ORB-SLAM (Mur-Artal, Montiel and Tardós 2015) enthält. Anstatt jedoch wie in PTAM und ORB-SLAM nur Eckpunkte zu verwenden, enthalten die *Keypoints* von PMO auch *Edgepoints* von GETT (Piccini, Persson, Nordberg, Felsberg and Mester 2014), welche durch die Integration der *Matching Predictability* entlang der Epipolarlinie sorgfältig ausgewählt werden.

Die Verbesserung der Pose-Schätzung wird normalerweise in fast allen VO/SLAM-Frameworks auf Basis von *Bundle Adjustment* durchgeführt, wobei der Fokus auf der Minimierung der Multi-View-Reprojektionsfehler liegt. Demgegenüber verbessert das PMO sowohl die Pose-Schätzung als auch die Keypoint-Korrespondenzen gleichzeitig, indem das gemeinsame epipolare *Tracking* (Bradler 2016) verwendet wird, welches die photometrischen Fehler minimiert.

Andere Frameworks wie z. B. ORB-SLAM, FTMVO (Mirabdollah and Mertsching 2015) und MVOCE verlassen sich auf RANSAC um die Keypoint-Ausreißer abzulehnen. Das Framework PMO zeigt jedoch, dass eine robuste visuelle Odometrie ohne RANSAC entwickelt werden kann und zwar indem gerade die *Keypoint Matches* sorgfältig ausgewählt werden, die der photometrischen Konsistenz und der epipolaren Abhängigkeit entsprechen.

Die monokulare Skala wird im PMO ähnlich zu dem MLM-SFM (Song and Chandraker 2014), FTMVO und MVOCE Verfahren mit Hilfe einer Ground Plane Estimation-Methode und der Annahme geschätzt, dass die Kamerahöhe über Grund bekannt ist. Für die im Bodenbereich gefundenen Unregelmäßigkeiten muss entsprechend korrigiert werden, um eine robuste Skalenschätzung zu erhalten. Um dies umzusetzen, verwendet das PMO segmentierte Straßenbereiche aus einer CNN-Klassifizierung, mit welcher nur Bodenbereiche für die Skalenschätzung verarbeitet werden.

Um eine robuste Lokalisierung zu erhalten müssen darüber hinaus IMOs identifiziert werden, sodass diese aus dem Schritt der *Pose*-Schätzung ausgeschlossen werden können. Das PMO führt hierbei eine neue Methode ein, um IMOs mit Hilfe der von (van den Brand, Ochs and Mester 2016) vorgeschlagenen semantischen Klassifizierung von Autos auf Basis eines CNNs zu erkennen. Mit dieser Methode werden die Autolabels analysiert und anschließend in statische Objekte und IMOs getrennt.

Keypoint-freie anfängliche relative Pose-Schätzung

Eine der Hauptaufgaben in der visuellen Odometrie Bildverarbeitungssystems besteht darin, kontinuierlich die relative Pose zwischen zwei aufeinanderfolgenden Frames zu schätzen. Ein spezifisches Merkmal meines Ansatzes ist es, dass ich zum Zeitpunkt der Aufnahme von Frame $n+1$ verschiedene Informationen über die nächste relative Pose kombiniere und zwar ohne dabei Keypoints zu verwenden. Im Gegenteil dazu verwende ich die anfängliche Schätzung der relativen Pose, um die Berechnung der neuen Keypoint-Positionen zu stabilisieren.

Um die relative Drehung zwischen zwei Frames zu schätzen, berechne ich entsprechend

dem Vorschlag von (Barnada et al. 2015) die Neigungs-Gier-Rollwinkel zwischen den beiden Frames. Die Schätzung der relativen Bewegung basiert auf einer Erweiterung eines einstufigen relativen Pose-Prädiktors (Bradler et al. 2015), welcher die Vorhersage relativer Pose-Informationen basierend auf der vorherigen relativen Pose-Information liefert. Dieser Pose-Prädiktor liefert auch in den seltenen Fällen die Rotationsschätzung, wenn der Neigungs-Gier-Roll-Schätzer keine liefert.

Generierung neuer Keypoints in sparse abgedeckten Gebieten

Wenn das Auto vorwärts fährt, werden neue Objekte sichtbar und folglich entsteht eine neue Menge von Keypoints, die verfolgt werden können. Daher werden während des Verfolgungsprozesses in der Regel zwei Mengen an Keypoints vorhanden sein: (1) alte Keypoints, die bereits verfolgt wurden und (2) neue Keypoints, die auf Basis des letzten Frames neu erkannt wurden.

Mein Keypoint-Generator wird derart kontinuierlich auf jeden Frame angewendet, sodass neue Keypoints auf den Teilen des Frames erzeugt werden, an denen die Dichte der alten Keypoints niedrig ist. Auf diese Weise wird versucht, auf jedem Frame eine gute Verteilung der Keypoints zu gewährleisten. Neue Keypoints werden basierend auf dem GFTT-Score (Good-Feature-to-Track) erkannt und ausgewählt, sodass wie von (Shi and Tomasi 1994) vorgeschlagen, entsprechende *Cornerpoints* gefunden werden. Des Weiteren werden *Edgepoints* basierend auf einer entsprechenden Methode von (Piccini et al. 2014) ausgewählt. Dies bedeutet, dass die Keypoints im Rahmen meines Frameworks eine Kombination aus Cornerpoints und Edgepoints darstellen.

Prior-basiertes Matching neuer Keypoints

Der Umgang mit neuen Keypoints ist schwieriger als der Umgang mit den alten Keypoints, welche zuvor schon verfolgt wurden. Der Grund dafür ist, dass jeder alte Keypoint mit 3D-Positionsinformationen aus seinem vorherigen Verfolgungsergebnis verknüpft ist; neuen Keypoints aber jedoch keine Tiefeninformationen zugeordnet sind. Um die differentielle Suche nach der entsprechenden Keypoint-Position optimal zu initialisieren, verwende ich eine von (Fanani, Barnada and Mester 2015) vorgeschlagene Methode, die auf der Berechnung der Bewegungsverteilung in einem betrachteten Bereich basiert und hierfür eine erweiterte Phasenkorrelationsmethode anwendet (Ochs, Bradler and Mester 2015).

Das prior-basierte Matching wird speziell auf die neuen Keypoints angewendet, um das erste Matching jedes Keypoints bereitzustellen. Mit anderen Worten, dieses Matching dient als eine Initialisierungsmethode, bevor die Keypoints durch die übertragungsbasierte Verfolgung verarbeitet werden.

Es ist wichtig zu beachten, dass die Matches der neuen Keypoints nur als Kandidaten für Keypoints-Korrespondenzen angesehen werden. Nur Keypoints, welche in

mindestens drei aufeinanderfolgenden Frames erfolgreich verfolgt wurden, werden anschließend als glaubwürdig betrachtet. Dies stellt sicher, das Keypoints basierend auf groben Fehl-Matchings und IMOs weitgehend aus der Menge der Keypoint-Trajektorien ausgeschlossen sind.

Pose- und Korrespondenz-Verbesserung durch gemeinsames epipolares Tracking

Die anfänglichen Schätzungen der relativen Pose und der Keypoint-Korrespondenzen zwischen zwei Frames können verbessert werden, indem die gegenseitige Einschränkung ausgenutzt wird. Auf der einen Seite ergibt sich aus einer relativen Pose eine epipolare Beschränkung bezüglich der Keypoint-Korrespondenzen. Auf der anderen Seite induziert eine Menge gegebener Keypoint-Korrespondenzen eine relative Pose.

Das PMO benutzt einen neuartigen Ansatz namens Epipolar-Tracking, welcher gleichzeitig ein differenzielles ('Lucas-Kanade') epipolar-geführtes photometrisches Matching auf alle Keypoints anwendet, wie schon von (Bradler 2016) vorgeschlagen und in (Bradler et al. 2017) erweitert. Durch das gemeinsame epipolare Tracking erhalte ich sowohl einen optimalen Satz von Keypoint-Korrespondenzen, als auch eine verbesserte relative Pose, welche beide gemeinsam der epipolaren Beziehung entsprechen.

Skalenschätzung

Monokulare visuelle Odometrie erfordert die Beachtung des *Scale Ambiguity Problems*, bzw. erfordert es, die geschätzten unskalierten Posen in korrekt skalierte Posen zu transformieren. Während die Skalierung von Frame zu Frame generell möglich ist, ist sie jedoch sehr anfällig für den *Scale Drift Effekt*. Im Kontrast hierzu, kombiniert das vorgeschlagene Skalenschätzungs-Schema die Hinweise sowohl von der *dense*, als auch von der *sparse* Ground Plane Estimation, wodurch die vorgeschlagene Methode robust gegenüber der unterschiedlichen Verteilung der Bodenstruktur ist.

Triangulation

Wenn eine Keypoint-Korrespondenz zwischen den Frames $n-1$ und n gegeben ist, kann der entsprechende 3D-Punkt durch Triangulation bestimmt werden. Das PMO nimmt hierbei an, dass die epipolare Geometrie durch die Punkt-Korrespondenz erfüllt wird, diese also der geschätzten relativen Pose entspricht. Das Framework bestimmt den 3D-Punkt im Kamerakoordinatensystem, indem der Schnittpunkt als auch die jeweiligen Bildpunkte von den Strahlen berechnet werden, die sich zum Zeitpunkt $n-1$ und n in den Projektionszentren der Kameras befinden. Es ist wichtig zu beachten, dass die Triangulation in PMO auf dem Matching von Keypoints basiert, welche in der Art eingeschränkt sind, dass sie garantiert der epipolaren Geometrie entsprechen.

Erkennung von sich unabhängig bewegenden Objekten (IMOs)

Ein visuelles Odometrie-Framework sollte in der Lage sein IMOs zu erkennen, da die Lokalisierungsschätzungen nur basierend auf der Annahme einer statischen Welt (Übereinstimmung mit der epipolaren Relation) optimiert werden sollten. Ich schlage hierfür eine Methode zur Erkennung von IMOs unter Verwendung von epipolaren Konsistenzprüfungen und Tiefenüberprüfungen vor. Ich verwende hierbei die *IMO Candidate Patches*, die auf Basis eines CNN-basierten Klassifikators generiert wurden, welcher von unserer VSI-Gruppe erstellt und in (van den Brand et al. 2016) veröffentlicht wurde. Die erhaltenen Bewegungslabels (IMO / statisch) werden dann über die Zeit unter Verwendung der Kombination von Bewegungshinweisen und erscheinungsbasierten Informationen der IMO Candidate Patches erhalten.

Experimente und Ergebnisse

Um die Genauigkeit von dem PMO zu bewerten, wurde eine vollständige Simulation des KITTI-Odometrie-Datensatzes durchgeführt. Das PMO erreichte hierbei zum Zeitpunkt der Evaluation (Juli 2017) die beste Genauigkeit unter den veröffentlichten monokularen Frameworks. Zum Zeitpunkt des Einreichens meiner Dissertation (Januar 2018) ist es immer noch eine der führenden monokularen Methoden im KITTI-Odometrie-Benchmark.

Zusammenfassung und Ausblick

In dieser Dissertation wurde das PMO als eine neuartige monokulare visuelle Odometrie vorgestellt. Das vorgeschlagene Framework wurde für Automobilanwendungen mit einem monokularen Aufbau und speziell für eine auf dem Ego-Fahrzeug angebrachte vorausschauende Kamera entwickelt.

Das Hauptziel des vorgeschlagenen Frameworks ist es, die Lokalisierung des Ego-Fahrzeug durch Kumulierung der relativen Positionsschätzung von Frame zu Frame über die Zeit hinweg zu ermöglichen. Zusätzlich kann die 3D-Weltkarte in einer sparse 3D-Punktform als Nebenprodukt von PMO erhalten werden. Im Gegensatz zu den anderen Mainstream-Frameworks, wurde PMO ohne RANSAC und ohne den Multi-frame Bundle Adjustment Schritt erstellt.

Das vorgeschlagene PMO-Framework wurde ausführlich auf Basis des KITTI-Datensatzes getestet, welcher aus tausenden von Bildern besteht, die von einer an einem fahrenden Auto befestigten Kamera aufgenommen wurden. Die Ergebnisse zeigen, dass das PMO eines der führenden monokularen Frameworks im KITTI-Benchmark ist.

Das PMO kann in mehreren Aspekten noch weiter verbessert werden. Die folgenden potenziellen Verbesserungen für das PMO werden als zukünftige Forschung vorgeschlagen: Keyframe-Mechanismus, Robustheit gegen Beleuchtungsänderungen und photometrische Kalibrierung.

Publications

This dissertation summarizes the PhD work that I performed between October 2014 and March 2018. Most of the work has been submitted and reported in a number of scientific publications listed below.

Journal paper:

- **Nolang Fanani**, Alina Stürck, Matthias Ochs, Henry Bradler, and Rudolf Mester. "Predictive monocular odometry (PMO): What is possible without RANSAC and multiframe bundle adjustment?" Image and Vision Computing (2017).

Conference papers:

- **Nolang Fanani**, Marc Barnada, and Rudolf Mester. "Motion priors estimation for robust matching initialization in automotive applications." International Symposium on Visual Computing. Springer International Publishing, 2015.
- **Nolang Fanani**, Matthias Ochs, Henry Bradler, and Rudolf Mester. "Keypoint trajectory estimation using propagation based tracking." IEEE Intelligent Vehicles Symposium (IV), 2016.
- **Nolang Fanani**, Rudolf Mester. "Propagation based tracking with uncertainty measurement in automotive applications." IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI), 2016.
- **Nolang Fanani**, Alina Stürck, Marc Barnada, and Rudolf Mester. "Multimodal Scale Estimation for Monocular Visual Odometry." IEEE Intelligent Vehicles Symposium (IV), 2017.
- Henry Bradler, Matthias Ochs, **Nolang Fanani**, and Rudolf Mester. "Joint Epipolar Tracking (JET): Simultaneous optimization of epipolar geometry and feature correspondences."IEEE Winter Conference on Applications of Computer Vision (WACV), 2017.
- **Nolang Fanani** and Rudolf Mester. "The precision of triangulation in monocular visual odometry". IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI), 2018.
- **Nolang Fanani**, Matthias Ochs, Alina Stürck, and Rudolf Mester. "CNN-based multi-frame IMO detection from a monocular camera." IEEE Intelligent Vehicles Symposium (IV), 2018.
- **Nolang Fanani**, Matthias Ochs, and Rudolf Mester. "Detecting parallel-moving objects in the monocular case employing CNN depth maps". European Conference on Computer Vision (ECCV) Workshop - Geometry Meets Deep Learning, 2018.

Acknowledgement

There are many people and parties who have been contributing to the completion of my PhD work and also to the writing of this dissertation. I might not be able to name every single of them on this page, as I believe that there are also some invisible hands. I truly appreciate all the help and I am sending my utmost gratitude to all of them.

This PhD journey could only be started when Prof. Dr. Rudolf Mester opened the door for me to be a part of his team in the VSI lab of Goethe University Frankfurt. I really thank him for the opportunity and the continuous support that he has been giving since day one. Rudolf does not only pour me with the necessary academic guidance but also keeps me on track by periodically providing useful advice for my personal improvement. His efforts to warmly welcome me and my family when we were new in Frankfurt are also very much appreciated.

I would also like to thank Prof. Dr. Matthias Kaschube for his willingness to be the reviewer of my dissertation. He has also assisted me multiple times for some scholarship-related issues which are really crucial to keep my PhD work going forward. He has always been helpful and available whenever I seek help.

My PhD stay in the VSI lab has been enjoyable because of the comfortable environment with pleasant team members. I would especially deliver my gratitude to the co-authors of my papers: Matthias Ochs, Henry Bradler, Alina Stürck, Marc Barnada. It has also been a pleasure to be in the same lab with other students: Svitlana, Jonas, Lucas, Daniel, Patrick, etc. It has been enjoyable to spend time together with them. I would also like to thank Mikael Persson from Linköping University, Sweden, who has provided significant help in a collaboration work, particularly during the early phase of my PhD.

I am so thankful to *Deutscher Akademischer Austauschdienst* (DAAD) for sponsoring my PhD. I especially appreciate how DAAD was willing to bend an extra mile with full understanding to assist my family when we had a mournful family matter at the beginning of our stay in Germany. I am also very fortunate to be a part of the DAAD-IGSP-2014 family, that have been relentlessly supporting each other for the success of our PhD journeys although living apart in different corners of Germany.

This dissertation is especially dedicated to my nucleus family. To my parents who have been very supportive throughout all my life: *Maturnuwun sanget bapak-ibu kagem sedaya doa kaliyan pengestunipun.* To the love of my life, Siska Premida Wardani, thousands of thanks won't be enough to you for always being on my side in up and down moments. For Tamara my little angle and Athan my sunshine, you are the sources of my motivation.

Last but not least, thanks to Allah, the most gracious, the most merciful, for the countless blessings. May this dissertation give benefits not only to me but hopefully also to others.

Contents

Chapter 1

Introduction

1.1 Motivation and thesis statement

According to many experts, autonomous vehicles are 'just around the corner' (Franke, Gavrila, Görzig, Lindner, Paetzold and Wöhler 1998)(Geiger et al. 2012). The availability of reliable and efficient vision systems is crucial to achieve this ambitious goal in automotive technology. Vision systems for complex tasks such as environment perception usually consist of a large number of interacting modules developed for specific purposes. It is not sufficient to optimize these modules individually; often it is a challenging task to jointly optimize a network or pipeline of such modules to robustly perform a particular task in a large variety of scenarios. The work presented in this dissertation specifically aims at automotive applications and driving scenes.

In today's traffic, human drivers largely rely on visual information: traffic signs, traffic lights and road markings are only some examples of crucial visual cues. This is why — though non-visual sensors, such as radar and LIDAR are also commonplace in advanced driver assistance systems (ADAS) — there is still a strong need to employ visual information for navigation. Substantial advances have been made during the recent ten years in the processing of visual data coming from stereo cameras. The obvious next step is to ask what is feasible with monocular systems — possibly as an intermediate step to multi-monocular surround view.

However, visual sensing in traffic scenarios is difficult. Algorithms that, for the first time, successfully solve a problem which has been notoriously nagging the scientific community often lead in its wake to a long lasting belief that such a soon-to-be classic approach is the only correct way to tackle the problem.

This dissertation aims to show that a system consisting of components that are, at least to a large degree, located off the current mainstream for visual odometry can lead to superior performance if the constraints of the regarded application area — here: autonomous driving and driver assistance — are considered properly. My method named *'predictive monocular odometry' (PMO)* provides one of the leading results among other

published monocular methods on the challenging KITTI benchmark (Geiger, Lenz, Stiller and Urtasun 2013) and even outperforms some stereo methods.

It is important to note the PMO results have been achieved without RANSAC and PMO is thus a contrast to the mainstream camera-based methods which heavily rely on iterative robustification. The results have also been achieved without multi-frame bundle adjustment (MFBA) which is a very useful tool in sparse VO/VSLAM to globally optimize egomotion estimates. This means that the results give an impression of the level of accuracy that is possible without MFBA. It also means that all results can probably still be further improved through such global optimization.

The following questions are to be answered by the work reported in this dissertation:

- While there are already several highly accurate stereo odometry frameworks, how accurate can a monocular visual odometry be?
- Does visual odometry always have to rely on the same set of commonly used components, specifically using RANSAC to deal with outliers?

1.2 Contribution of this thesis

This dissertation offers several contributions:

- Propagation-based tracking (PbT) is proposed as the backbone of PMO, while PMO is further enriched with other modules, i.e. scale estimation and independently moving object (IMO) detection, as shown in figure 1.1. PbT provides an accurate estimate of the unscaled egomotion, sparse 3D mapping of tracked keypoints as well as the 2D trajectory of tracked keypoints. Chapter 3 contains the detailed description of the proposed propagation-based tracking framework.
- In a novel keypoint matching initialization, the motion prior between the two images being compared is calculated. It will be shown that the proposed method significantly shifts the search space into the vicinity of the true match, as indicated by the measured similarity metrics. This contribution is discussed in chapter 5.
- The triangulation of keypoint correspondences on two different frames is deeply analyzed. Specifically, the sensitivity of the depth estimation error with respect to the matching error is computed, as presented in chapter 6.
- Additionally, a scale estimation framework combining ground plane estimation and CNN-based street segmentation is proposed. The accuracy of the scale estimate, when combined with the PbT framework, leads to the best results in the KITTI odometry benchmark among published monocular methods, as in July 2017. More details are given in chapter 8.
- Last but not least, a method to detect independently moving object (IMO) is proposed and presented in chapter 9.

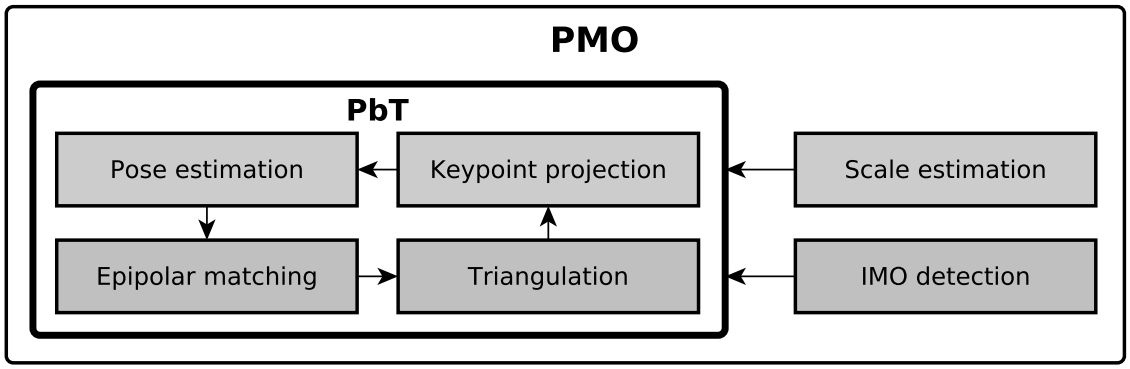

Figure 1.1: The core components of the PMO framework. PMO contains PbT as a backbone and is enriched by a scale estimation and a detection method for IMO.

1.3 Credits to the work of colleagues

The work presented in this PhD dissertation has been obtained during my stay in the Visual Sensorics and Information Processing (VSI) lab, Goethe University of Frankfurt. As a part of a big team in the VSI group, I built my visual odometry framework using supporting components which are the work of my colleagues in the VSI lab. These supporting components have been reported in the following publications:

- Keypoint-free pitch-yaw-roll (PYR) estimation by Marc Barnada et al (Barnada et al. 2015)
- Joint epipolar tracking (JET) and the pose predictor from a dynamical motion model by Henry Bradler et al (Bradler et al. 2015) (Bradler 2016)
- Phase correlation motion estimator by Matthias Ochs et al (Ochs et al. 2015)
- Diverse CNN implementations for depth images, street segmentation, vehicle detection and vehicle instance segmentation by Matthias Ochs (Ochs 2017) and Jan van den Brand (van den Brand et al. 2016). I only use the resulting CNN masks for PMO implementation.

1.4 Organization of the dissertation

This dissertation is organized in ten chapters. Chapter 1 briefly presents the overview of the dissertation and the main contributions are listed. Chapter 2 discusses the scientific foundation of monocular visual odometry and the state-of-the-art methods. Chapter 3 presents an overview of the proposed Predictive Monocular Odometry (PMO) framework.

A statistical analysis of keypoint matching is discussed in chapter 4. Chapter 5 discusses a keypoint matching method based on motion priors. Chapter 5 is mainly based on the

paper published in (Fanani et al. 2015).

In chapter 6, the triangulation of keypoint correspondences is analyzed, adopted from (Fanani and Mester 2018). The pose estimation used in PMO is discussed in chapter 7. Chapter 7 is mainly adopted from our paper published in (Fanani, Ochs, Bradler and Mester 2016).

Chapter 8 presents the monocular scale estimation method which is based on (Fanani, Stuerck, Barnada and Mester 2017). Chapter 9 covers the proposed method to detect independently moving object (IMO), which was written based on published papers (Fanani, Ochs, Stürck and Mester 2018) and (Fanani, Ochs and Mester 2018).

Last but not least, chapter 10 summarizes this dissertation and discusses the potential future work beyond the scope of this dissertation.

Chapter 2

Fundamentals of visual odometry and the state of the art

Visual odometry is the process of estimating the position and the orientation of a single or multiple cameras from a sequence of images. In most cases, the individual motion of the camera between two consecutive frames is determined by analyzing the 2D image motion of a set of image primitives: points, lines, or in rare cases also regions. Most important and dominant in the literature is the analysis of corresponding points. If a certain minimum number of corresponding points (or lines) is known, the relative 3D motion of the camera between the two frames can be estimated.

The 3D structure surrounding the camera can usually be provided as by-product information by a visual odometry system. Hence, visual odometry can be a part of a Structure-from-Motion (SfM) framework, in which both the 3D structure and the camera pose are to be estimated.

A projective camera, similar to the human eye, is able to capture the visual information of the world. However, the camera can only obtain information in the form of 2D data, which is the projection of the world to the image plane. Combining information from more than one views can give much more information than just a set of 2D data from only one view, which subsequently can be utilized for other purposes, e.g. the computation of the 3D world map and the estimation of the camera position.

This chapter presents the fundamentals of visual odometry which are useful to understand the content of this dissertation. The chosen coordinate system and the parametrization of camera pose are also described. In the end of this chapter, I present the state of the art of monocular visual odometry frameworks.

2.1 Motion parametrization

Let us describe the relative pose between any two different views at times $n-1$ and n with the rotation matrix ${}^{n-1}\mathbf{R}_n$ and the translation vector ${}^{n-1}\boldsymbol{t}_n$. These relative pose

parameters describes the camera pose at time $n-1$ with respect to the camera pose at time n based on the following convention for the 3D coordinate transformation

$$\boldsymbol{p}_n = {}^{n-1}\mathbf{R}_n \cdot \boldsymbol{p}_{n-1} + {}^{n-1}\boldsymbol{t}_n \tag{2.1}$$

where $\boldsymbol{p}$ is the 3D vector representing an arbitrary fixed static point in the world. It is often more comfortable to rewrite equation 2.1 to homogeneous form where $\tilde{\boldsymbol{p}} = \begin{pmatrix} \boldsymbol{p} & 1 \end{pmatrix}^T$,

$$\tilde{\boldsymbol{p}}_n = \begin{pmatrix} {}^{n-1}\mathbf{R}_n & {}^{n-1}\boldsymbol{t}_n \\ \mathbf{0} & 1 \end{pmatrix} \cdot \tilde{\boldsymbol{p}}_{n-1} \tag{2.2}$$

The relative pose information can be parametrized by six parameters as visualized in

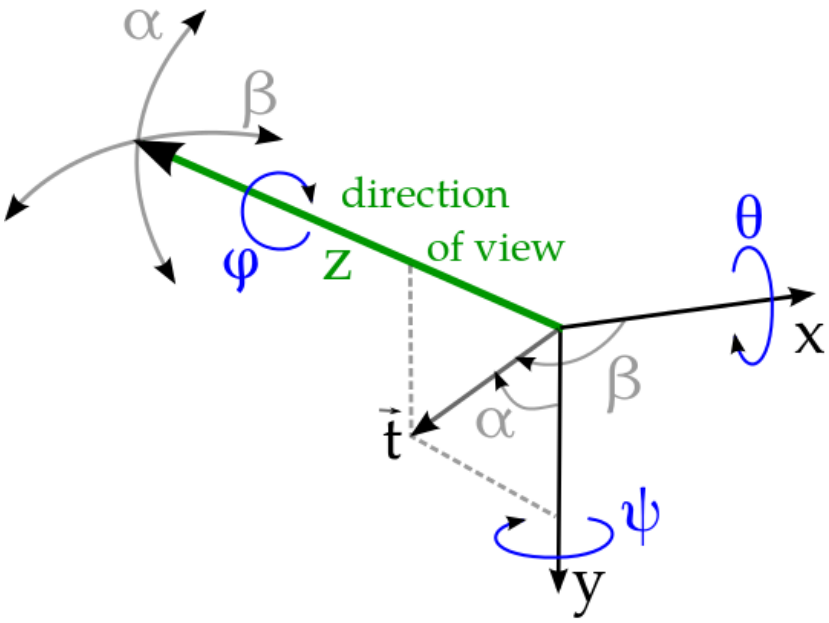

Figure 2.1: The parametrization for the camera egomotion, showing the camera coordinate system, the three rotation angles, and the translation vector $\boldsymbol{t}$ which is represented in spherical coordinates by angles α and β, and a length s.

figure 2.1 and proposed in (Bradler et al. 2015). This parametrization can easily be converted into a rotation matrix ${}^{n-1}\mathbf{R}_n$ and a translation vector ${}^{n-1}\boldsymbol{t}_n$ (see sections 2.1.1 and 2.1.2). Let $\boldsymbol{z}$ be a motion state vector constructed by a set of motion parameters yielding a relative pose transformation between two consecutive frames.

$$\boldsymbol{z}_n = [\theta, \psi, \varphi, v, \alpha, \beta] \tag{2.3}$$

It consists of three rotation parameters (θ, ψ, φ) and three translation parameters (v, α, β).

This particular representation of rotation is chosen instead of other rotation representations such as quaternions, because it is well suited to the case of motion which is dominantly in the direction of the z-axis (see figure 2.1). This means that the pitch, yaw, and roll angles are counted in such a way that their changes about the nominal motion is about 0. This allows for simple linearization of the motion equations.

2.1.1 Parametrization of rotations

Rotation is a geometric transformation in 3D space that leaves the length of any 3D vector unchanged. The rotation is represented by a rotation matrix $\mathbf{R}$ with the following characteristics:

- it is an orthogonal matrix: the transpose is also the inverse.

$$\mathbf{R}^{-1} = \mathbf{R}^T \tag{2.4}$$

- the determinant is 1.

$$|\mathbf{R}| = 1 \tag{2.5}$$

I chose to represent rotations by three rotation angles θ, ψ, φ when looking in viewing direction which respectively correspond to the pitch, yaw, roll angles.

- θ is positive when a pitch motion occurs upwards.
- ψ is positive when a yaw motion occurs to the right.
- φ is positive when a roll motion occurs in clockwise direction, when looking in the direction of the positive $z-$axis.

By convention, the rotation matrix can be composed by a concatenation of pitch, yaw, and roll motions:

$$^{n-1}\mathbf{R}_n = \mathbf{R}_z(\phi) \cdot \mathbf{R}_y(\psi) \cdot \mathbf{R}_x(\theta) \tag{2.6}$$

$$\mathbf{R}_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix} \tag{2.7}$$

$$\mathbf{R}_y(\psi) = \begin{pmatrix} \cos\psi & 0 & \sin\psi \\ 0 & 1 & 0 \\ -\sin\psi & 0 & \cos\psi \end{pmatrix} \tag{2.8}$$

$$\mathbf{R}_z(\phi) = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.9}$$

2.1.2 Translation parameters

The relative translation vector is represented by three translation parameters: translation length/scale s and two polar angles α, β which determine the direction of the translation vector (see figure 2.1). The translation length can be transformed into a velocity metric depending on the frame rate. β is the polar angle relative to the x-axis, while α is the azimuth angle measured relative to the y-axis in the y-z-plane. A perfect straight forward motion results in $\alpha = -\pi/2$ and $\beta = \pi/2$.

The translation vector can be represented using the following equation,

$$ {}^{n-1}\boldsymbol{t}_n = s \cdot \begin{pmatrix} \cos\beta \\ \sin\beta\cos\alpha \\ \sin\beta\sin\alpha \end{pmatrix} \tag{2.10} $$

2.2 Camera modeling

2.2.1 Camera: monocular vs. stereo

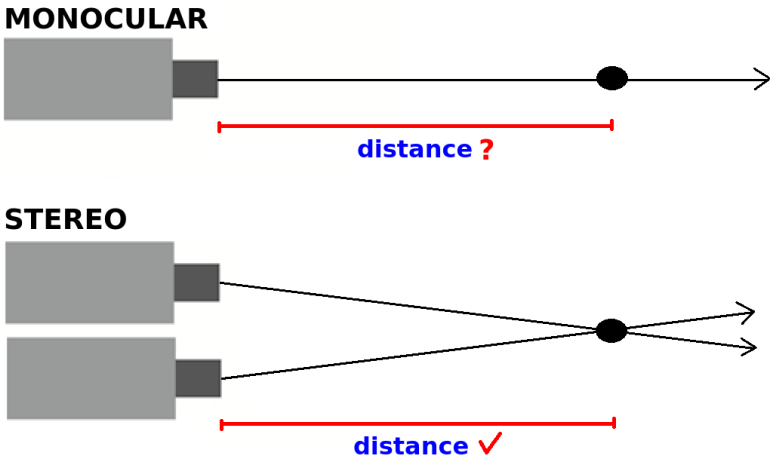

Figure 2.2: Illustration of monocular camera and stereo cameras when observing an object. Unlike a monocular setup, a stereo setup is able to estimate the distance of the object.

A camera is regarded as the natural choice of car sensor because it resembles the use of human eyes in driving. Unlike laser/radar sensors which only provide depth information, a camera can capture the rich visual information, both as grayscale or color images. This visual information is very important and it can be exploited for further steps in ADAS, such as for image segmentation and object tracking. Unlike LIDAR, cameras are also appealing to car industry because of the low price.

However, the raw data from a camera is only in a 2D form, i.e. an image. There is no depth information in one single image, in contrast to the laser/radar-based sensors which instantly also provide the depth information. More efforts are required in order to transform the data from images into the indispensable 3D world structure. For the stereo camera setup, the distance of an object can be estimated if the object is observed on both cameras, as illustrated in figure 2.2. Estimating the object distance is much more challenging in the monocular setup. Monocular frameworks usually need to combine observations from multiple time steps in order to estimate the distance of an object. However, without the knowledge of the inter-frame absolute distance, monocular frameworks inherently suffer scale ambiguity problem.

2.2.1.1 Monocular scale ambiguity problem

The scale ambiguity problem in monocular frameworks can be illustrated by figure 2.3. For a monocular camera, it cannot be decided on the basis of the mere visual information whether it moves in a world of actual scale (e.g. houses being about 10 meters tall) or a miniature world. Monocular visual odometry requires the ability to deal with the scale ambiguity problem, or equivalently to transform the estimated unscaled poses into correctly scaled poses. This problem will be dealt with in chapter 8.

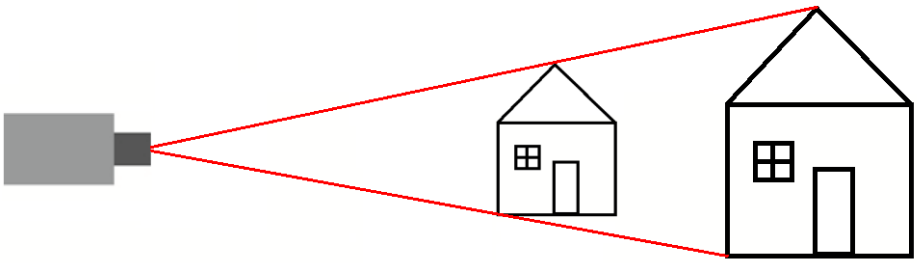

Figure 2.3: Illustration of the monocular scale ambiguity problem.

2.2.2 Coordinate frame

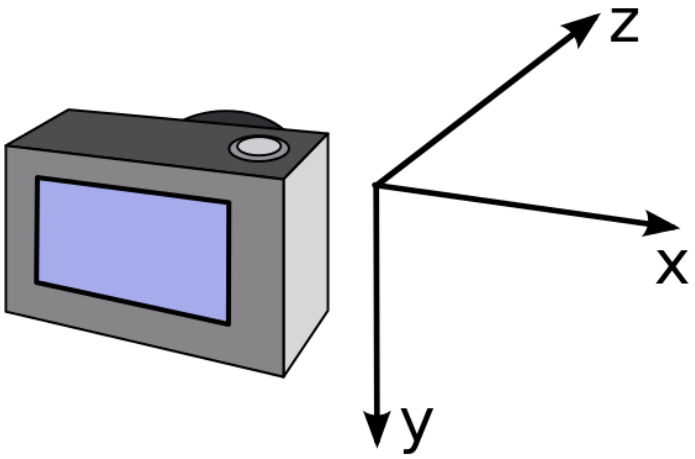

Figure 2.4: Convention for the orientation of the camera coordinate system.

The 3D camera coordinate system, as illustrated in figure 2.4, is defined as a right-handed system using the following conventions,

- x-axis pointing to the right direction
- y-axis pointing downwards

- z-axis pointing forwards to the view direction

The forward-looking camera is firmly attached to the vehicle, for example as illustrated in figure 2.5.

Figure 2.5: An example of camera placements in a vehicle. It can be attached inside the vehicle as shown in this picture (taken from (Wikimedia 2014)) or attached on top of the vehicle as in the KITTI camera setup (Geiger et al. 2013). In the proposed PMO framework, only the data from one camera (monocular) is used.

2.2.3 Mathematical camera model

The proposed PMO framework uses a pinhole camera model. Let us regard the relation between the 3D world coordinate (X, Y, Z) and the corresponding image coordinate (x, y).

Let the relative pose between the world coordinate system and the camera coordinate system be given by a rotation matrix ${}^{w}\mathbf{R}_c$ and a translation vector ${}^{w}\boldsymbol{t}_c$, such that the 3D world coordinate $\boldsymbol{p}_w$ can be transformed into the camera coordinate $\boldsymbol{p}_c$ using the following equation,

$$\boldsymbol{p}_c = \begin{pmatrix} {}^{w}\mathbf{R}_c & {}^{w}\boldsymbol{t}_c \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{p}_w \\ 1 \end{pmatrix} = \begin{pmatrix} {}^{w}\mathbf{R}_c & {}^{w}\boldsymbol{t}_c \end{pmatrix} \cdot \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} . \tag{2.11}$$

Let $\mathbf{K}$ be the matrix of intrinsic camera parameters which contains the information of focal lengths (f_x, f_y) and the principal point (c_x, c_y).

$$\mathbf{K} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2.12}$$

The following relation holds,

$$d \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathbf{K} \cdot \boldsymbol{p}_c = \mathbf{K} \cdot \begin{pmatrix} {}^w\mathbf{R}_c & {}^w\boldsymbol{t}_c \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{p}_w \\ 1 \end{pmatrix} = \mathbf{K} \cdot \begin{pmatrix} {}^w\mathbf{R}_c & {}^w\boldsymbol{t}_c \end{pmatrix} \cdot \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (2.13)$$

where d is a factor which depends on the regarded 3D point coordinate.

From equation 2.13, we can define now a projection operation $\pi(\boldsymbol{p}_w, c)$ which computes the projection of 3D world point $\boldsymbol{p}_w$ to the camera frame-c.

$$\pi(\boldsymbol{p}_w, c) = \begin{pmatrix} x \\ y \end{pmatrix} \quad (2.14)$$

2.2.4 Epipolar geometry

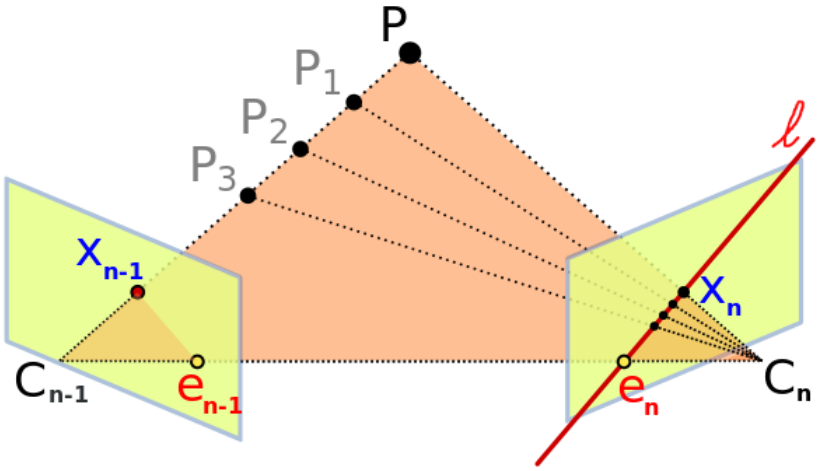

Figure 2.6: The epipolar geometry between frame $n-1$ and frame n. A 3D world P, observed in frame $n-1$ at point $\boldsymbol{x}_{n-1}$, must be observed at a point along the epipolar line ℓ in frame n.

Epipolar geometry defines the geometrical relation between two views, as illustrated in figure 2.6. Let us regard frame $n-1$ and frame n, with the camera centers denoted respectively as C_{n-1} and C_n. A 3D point P in the world is observed in frame $n-1$ at a 3D coordinate $\boldsymbol{y}_{n-1}$ and the corresponding pixel coordinate $\boldsymbol{x}_{n-1}$. If the relative pose between frame $n-1$ and frame n is perfectly known, we can determine the epipolar line ℓ in frame n such that the projection of the world point P to frame n must lie on this epipolar line. This property is particularly beneficial for keypoint matching and tracking in order to decrease the search space and to converge faster to the correct matches (see chapter 5).

Figure 2.6 shows that the projections of the 3D world points P,P_1,P_2,P_3 onto frame n are all found along the epipolar line ℓ. The points $\boldsymbol{e}_{n-1}$ and $\boldsymbol{e}_n$ are the epipoles which

are defined as the intersection points of the line $\overline{C_{n-1}C_n}$ and frame $n-1$ and frame n, respectively. The triangular plane defined by the points P, C_{n-1}, C_n is called the epipolar plane.

The epipolar structure between two views is usually represented by the essential matrix $\mathbf{E}$. The epipolar relation can be written as,

$$\boldsymbol{y}_{n-1}^T \cdot \mathbf{E} \cdot \boldsymbol{y}_n = 0 \tag{2.15}$$

The essential matrix $\mathbf{E}$ is given by

$$\mathbf{E} = \left[{}^{n-1}\boldsymbol{t}_n \right]_\times \cdot {}^{n-1}\mathbf{R}_n. \tag{2.16}$$

where $[\boldsymbol{t}]_\times$ is defined for $\boldsymbol{t} = (t_1, t_2, t_3)^T$ as follows:

$$[\boldsymbol{t}]_\times = \begin{pmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{pmatrix} \tag{2.17}$$

I refer to (Hartley and Zisserman 2003, p. 239) for further details of epipolar geometry.

2.3 Keypoint matching and tracking as basic 2D motion analysis operations

2.3.1 Keypoints

A *keypoint* is a geometrical entity, which is a point on the image plane, most often also a point on the image grid, thus represented by two real-valued or integer numbers. Keypoints are points which are special in a certain way; they should be chosen such that they can be easily found and associated with each other in several related images showing the same scene. Each keypoint is associated with the surroundings of that keypoint. As the gray value or color value of a single pixel is not very informative, it is commonplace to regard the surrounding pixel values of a keypoint. The local surrounding of the keypoint is an image patch of limited size (usually $N \times N$ pixels). The image patch content is represented by its pixel values. The patch is usually also associated with a mask of the same size. In the context of *feature based or descriptor based matching*, the patch is converted into a *feature vector* or *descriptor*. The reason for computing such an abstracted representation is that this representation can emphasize those characteristics which are important for the matching process, and make the representation more invariant against irrelevant changes of the pixel values, e.g. by illumination changes, or by variations of scale.

Keypoints are usually chosen during a keypoint detection step for their distinguishable characteristic that is important for the success of keypoint tracking. Keypoint detection

is the process to find salient points from an image, with the main importance of selecting the keypoints is to make sure that the keypoints can most likely be re-identified when the keypoints are seen in other images.

There are numerous methods to detect salient keypoints on an image. Some of them come also with an associated method for computing a descriptor. I briefly discuss some of them: Förstner keypoint detector (Förstner 1986), Harris corner detector (Harris and Stephens 1988), GFTT (Shi and Tomasi 1994), SIFT (Lowe 2004), SURF (Bay, Tuytelaars and Van Gool 2006), and FAST (Rosten and Drummond 2006). For a deeper comparison of different keypoint detectors, I refer to a survey paper by Li et al (Li 2017).

- **Förstner keypoint detector**
  An in-depth analysis of keypoint detection using local gradients has been performed by Förstner (Förstner 1986), and then extended in (Förstner and Gülch 1987). Förstner & Gülch (Förstner and Gülch 1987) proposed a keypoint detector with potential localization improvement. The proposed algorithm finds a good way to decide where to place the keypoint in a patch, not necessarily at the centre of the patch.

- **Harris corner detector**
  The Harris corner detector (Harris and Stephens 1988) is one of the first corner detection methods that is still commonly used in computer vision algorithms to detect corner keypoints in an image. Harris extended the work done by Moravec (Moravec 1981) where the use of interest points or keypoints was first introduced. The Harris corner detector introduces a corner score to identify a corner point which is computed based on the trace and the determinant of the structure tensor matrix (see section 4.2.1). The Harris corner points have the property of being rotation-invariant.

- **Good Features to Track (GFTT)**
  Shi and Tomasi proposed the *good features to track (GFTT)* method (Shi and Tomasi 1994), also known as Shi-Tomasi feature detector, to detect corner and edge points in an image. While Harris and Stephens (Harris and Stephens 1988) proposed a function to compute the corner score based on the eigenvalues of the structure tensor, the GFTT simply looks into the minimum eigenvalue of the structure tensor matrix in order to choose keypoints. The corner and edge points are classified based on the ratio between the two eigenvalues of the structure tensor. The Shi-Tomasi keypoints have the property of being rotation-invariant.

- **SIFT detector**
  The Scale Invariant Feature Transform (SIFT) algorithm (Lowe 2004) detects keypoints of an image and creates the corresponding keypoint descriptors which have the property of being scale-invariant. The SIFT features are defined as the maxima and minima of the result of the difference of a Gaussian averaging filter applied to a pyramid of differently scaled images. Besides the rotation and scale-invariant properties, the SIFT features also have the property of being affine-invariant.

- **SURF detector**
  The 'Speeded Up Robust Features' (SURF) detector (Bay et al. 2006) is a faster version of SIFT, with relatively similar properties. The difference is that SURF uses the Hessian matrix computed at every pixel location to choose the keypoints.
- **FAST detector**
  Rosten and Drummond proposed the 'Features from Accelerated Segment Test' (FAST) algorithm to detect corner points in an image (Rosten and Drummond 2006). In order to be chosen as a FAST point, a point must have a sufficient number of surrounding pixels whose pixel intensities are significantly higher or lower than the considered point. The surrounding pixels are chosen based on the distance to the considered point, hence they form a circular form. As the name suggests, the FAST algorithm is very fast as it directly checks the pixel intensity values and hence it is suitable for real-time application. On top of being rotation-invariant, the FAST feature also has the property of being scale-invariant.

The choice of the keypoint detection method directly affects the resulting set of chosen keypoints in an image. As the chosen keypoints serve as the representation of an image, it is crucial to choose the suitable keypoint detection method. Once keypoints are selected from an image, there are a number of alternatives to represent them as descriptors/features before running matching and tracking keypoints to other images. One of the simplest descriptors is to build a keypoint patch consisting of the color/gray-value information on the keypoint position and its surrounding pixels. This descriptor is suitable if we choose matching error metrics based on pixel color/gray-value information, such as the sum of squared difference (SSD).

2.3.2 Matching and tracking

Keypoint *matching* and *tracking*[1] are two related tasks to locate the position of keypoints on one or more other images. Keypoint matching aims to find the keypoint correspondences between two images, while keypoint tracking is chaining matches from one initial image across a sequence of images.

Matching of keypoints to find a pair of point-to-point correspondence between two images is equivalent to finding the *displacement vector* that connects two corresponding keypoints in two images. A displacement vector is defined as a vector which shows how the position of a keypoint changes from its original position in one image to a new position in another image. A set of displacement vectors for different keypoints between two images is often called the *optical flow* between the images. Depending on the density of the displacement vectors, we can have sparse, semi-dense, or dense optical flow.

Determining the optical flow can be carried out in local methods or global methods. The difference is whether the energy term, also known as the loss function, is defined on the global image array, or only on patches, or segments. Horn and Schunck (Horn

[1] In the literature, the imprecise term *feature* tracking is often found.

and Schunck 1981) proposed a method to estimate the optical flow between two images using a global constraint of smoothness. As a result, dense optical flow is obtained. The method is known to over-smooth the resulting optical flow field. Around at the same time, the Lucas-Kanade method (Lucas, Kanade et al. 1981) was proposed. On the contrary to the Horn-Schunck global method, Lucas-Kanade introduce a local differential method for optical flow estimation. The Lucas-Kanade method is more robust in terms of the over-smoothing problem than Horn-Schunck, but generates less dense optical flows. Bruhn et al (Bruhn, Weickert and Schnörr 2005) compared and proposed a way to combine both local and global methods.

Based on the survey by Zitova et al (Zitova and Flusser 2003), there are two main approaches of keypoint matching: feature-based methods and photometric patch matching methods [2]. Feature-based tracking, or more precisely *descriptor-based matching*, compares the selected keypoint descriptors between two images, and finds the best match between the descriptors on two images. Popular methods employ e.g. SIFT (Lowe 1999), SURF (Bay et al. 2006) and ORB (Rublee, Rabaud, Konolige and Bradski 2011). This means that the mentioned methods do not only define a keypoint detector, but also a descriptor for the local image patch around the keypoint. Feature-based matching is especially suitable when the relative pose between the two images is unknown, hence the true correspondence can be virtually everywhere in the visual field. Due to the large (virtual) search area of feature/descriptor-based matching, there is a large risk of ending up with false matches: outliers. If there is a parametric model for the true motion field, iterative re-sampling methods such as RANSAC (Fischler and Bolles 1981) are usually employed to separate the outliers from the 'inlier set' of correct matches (see section 2.4.2).

For photometric-based matching, there are a number of methods to measure the dissimilarity level between two keypoint patches. Here I discuss four of them: the sum of squared differences, the sum of absolute differences, the normalized crossed correlation, and the zero-mean normalized crossed correlation. I refer to (Aschwanden and Guggenbuhl 1992) for a deeper comparison of the error metrics for photometric matching.

Let $s_1(\boldsymbol{x}_i)$ and $s_2(\boldsymbol{x}_i)$ be the two patches being compared at a set of coordinates $\boldsymbol{x}_i$.

- Sum of squared difference (SSD)
  The SSD error metric is usually used when the feature descriptor is a block of image pixels. The dissimilarity level between two features is measured as the sum of the squared difference between the pixel intensity.

$$SSD = \sum_i (s_1(\boldsymbol{x}_i) - s_2(\boldsymbol{x}_i))^2 \tag{2.18}$$

- Sum of absolute difference (SAD)
  The SAD error metric is similar to SSD. Instead of summing the squared difference, the SAD error metric is measured as the sum of the absolute difference between

[2] The author calls the latter 'area-based', but I consider this term less precise.

the pixel intensity.

$$SAD = \sum_i |s_1(\boldsymbol{x}_i) - s_2(\boldsymbol{x}_i)| \tag{2.19}$$

- Normalized cross correlation (NCC)
  Unlike SSD and SAD which are dissimilarity metrics, NCC is a similarity metric. NCC is known for its ability to handle brightness changes. Brightness change is compensated during the normalization step when calculating the NCC value.

$$NCC = \frac{\sum_i s_1(\boldsymbol{x}_i) \cdot s_2(\boldsymbol{x}_i)}{\sqrt{\sum_i s_1^2(\boldsymbol{x}_i)}\sqrt{\sum_i s_2^2(\boldsymbol{x}_i)}} \tag{2.20}$$

- Zero-mean normalized cross correlation (ZNCC)
  ZNCC is a modification of NCC. The pixel intensity mean of each patch is first subtracted from the pixel values.

$$ZNCC = \frac{\sum_i (s_1(\boldsymbol{x}_i) - \bar{s_1}) \cdot (s_2(\boldsymbol{x}_i) - \bar{s_2})}{\sqrt{\sum_i (s_1(\boldsymbol{x}_i) - \bar{s_1})^2}\sqrt{\sum_i (s_2(\boldsymbol{x}_i) - \bar{s_2})^2}} \tag{2.21}$$

  where $\bar{s}$ is the mean value from all $s\boldsymbol{x}_i$.

In the proposed PMO framework, SSD is chosen as the error metric to be minimized. Unmatched keypoints are no longer considered for future tracking and thus rejected based on the SSD, as discussed in section 7.6 about outlier rejection.

In the PMO framework, I focus on using the photometric approach for keypoint matching. The photometric-based matching can be further classified into two types: *block matching* and *differential matching*. Both approaches aim at minimizing the photometric difference between the two compared keypoints or keypoint patches. The matching process can be made at a limited set of discrete positions (as in block matching) or in the local vicinity of a given displacement (as in differential matching) These fundamental approaches are briefly discussed in the following sections.

2.3.3 Block matching

The block matching approach aims at matching keypoints which are represented by a block of pixels, by comparing all possible matching possibilities within the search range, as illustrated in figure 2.7. The size of the block and the search range need to be carefully set to make sure that the keypoint is well represented by the block and the matching process is computationally not too expensive. When computing the similarity (or dissimilarity) metric, a mask with the same size of the block can be applied to each block, e.g. a Gaussian[3] mask. A loss function Q with a mask w based on the photometric SSD between two compared patches y and z is to be minimized by the

[3]The term 'Gaussian' should not be taken too literally; in fact in all implementation a spatially truncated and sampled discrete version of a Gaussian function is used which is parametrized by its 'standard deviation' σ and the truncation window size.

sought displacement vector $\boldsymbol{v}$. The pixel position on the referenced patch is denoted by $\boldsymbol{x}_i$.

$$Q \overset{def}{=} \sum_i w(\boldsymbol{x}_i) \cdot (z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i + \boldsymbol{v}))^2 \quad \longrightarrow \min \tag{2.22}$$

The shape of the mask determines the weight distribution for the pixels in the block. The block matching can also be performed on some defined search ranges. This applies for instance in epipolar matching (see chapter 5) or in a search range which is not centered on the position of the reference keypoint.

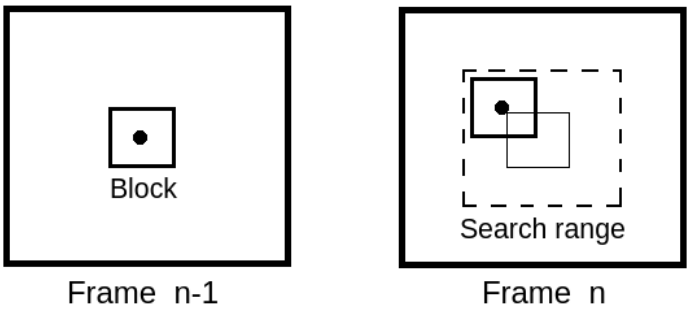

Figure 2.7: The illustration of block matching from frame $n-1$ to frame n. The best match is searched within the search range.

The range of the possible obtained displacement vector of block matching is in any case limited by the search range but can be as large as that. I refer to section 4.1 in this thesis, as well as (Sangi 2013) and (Mester and Hötter 1995) for the precision analysis of block matching.

2.3.4 Differential matching

Differential matching is often implemented in the spirit of the Lucas-Kanade approach (Baker and Matthews 2004). As in block matching, a loss function Q based on the photometric SSD between two compared patches y and z is defined as in equation 2.22.

The difference as compared to the block matching is that the differential matching tries to find the displacement vector $\boldsymbol{v}$ in small steps, not on the grid. Starting from an initial assumption of the displacement vector $\boldsymbol{v}_{prev}$ (which can also be $\mathbf{0}$), the derivative of the loss function w.r.t. a change of the displacement vector is computed. The scheme to find $\boldsymbol{v}$ continues iteratively until it converges.

$$\frac{\partial Q}{\partial \boldsymbol{v}} = 0 \quad \longrightarrow \quad \Delta\boldsymbol{v} = \arg\min_{\boldsymbol{v}} Q(\boldsymbol{v}) \tag{2.23}$$

$$\boldsymbol{v} = \boldsymbol{v}_{prev} + \Delta\boldsymbol{v} \tag{2.24}$$

The capture range of differential matching depends on the 'roughness' of the input signal. It can be increased by applying a low-pass filter to the regarded two images. This can also be done in a pyramidal approach, thus starting with a sub-sampled version of the image followed by gradual refinements on higher resolution images. However, the fine details of the image can be overlooked when applying the low-pass filter or the pyramidal approach. I refer to section 4.2 for a deeper analysis of the differential matching.

2.4 Two-view relative pose estimation from N given point correspondences

If a certain minimum number of keypoint correspondences between two frames is known, the relative 3D motion of the camera can be estimated. Based on the available correspondences in 2D coordinates or 3D coordinates, camera motion estimation approaches can be categorized into three groups:

- Relative pose estimation using two sets of corresponding image points (2D-to-2D optimization)
  This category compares only the pixel information between two images. The motion is estimated through the optimization of the essential matrix which sets the constraint of the feature position in two images.

  Longuet-Higgins (Longuet-Higgins 1981) proposed a method to compute the three-dimensional structure of a scene from two images with already known point-to-point correspondences. He proposed the 8-point method to estimate the pose between two image which requires at least 8 point correspondences to solve the estimation problem. The 8-point method was later significantly improved in terms of its numerical/statistical stability by Hartley (Hartley 1997). This 8-point method was then widely used by many researchers until Nister et al (Nistér, Naroditsky and Bergen 2004) introduced a 5-point method that becomes a standard tool until now, thanks to its robustness against outliers in the context of a RANSAC algorithm. This method is still widely used until now, especially at the beginning of visual odometry framework when only 2D pixel position of the feature points are available from the matching process.

  The optimized $\mathbf{E}$ matrix then still needs to be decomposed into rotation and translation information, unless we directly parametrize the relative pose as rotation matrix $\mathbf{R}$ and translation vector $\boldsymbol{t}$. The decomposition usually checks that the triangulated keypoints will lie in front of both camera views, with the assumption that the keypoints are rigid point set.

- Relative pose estimation using two sets of corresponding world points (3D-to-3D optimization)
  Motion estimation using 3D-to-3D optimization uses two sets of triangulated 3D keypoints from two different time step. The optimization is done by minimizing

the distance error between two corresponding 3D keypoint position.

Theoretically, three 3D-to-3D non-collinear keypoint correspondences are sufficient to solve the optimization problem (Scaramuzza and Fraundorfer 2011). No matrix decomposition is needed because the motion parameters are directly optimized.

- Relative pose estimation using two sets of corresponding world points and image points (3D-to-2D optimization)
  The 3D motion of camera can also be estimated from a set of world points that are computed up to frame n, and a set of image points in frame $n+1$. Motion estimation using 3D-to-2D optimization is getting more widely used than the 3D-to-3D case, especially since the work by Nister et al (Nistér et al. 2004).

  Nister et al (Nistér et al. 2004) reported that the 3D-to-2D approach generally has less error than the 3D-to-3D approach because the triangulated 3D position is naturally more uncertain than 2D image coordinate.

  The formulation of the motion estimation is done by minimizing the image reprojection error of the 3D point into the 2D image pixel. This problem is also called *perspective from n points (PnP)* problem. Six 3D-2D point correspondences are sufficient to estimate the relative pose between two frames (Hartley and Zisserman 2003, p. 296).

2.4.1 Bundle adjustment

Bundle adjustment is an approach to optimize a loss function which ties together measurements from typically more than 2 frames and tries to find the parameters of a model which explains as good as possible the ensemble of all these measurements. Bundle adjustment has been widely utilized to optimize the accuracy of the camera pose estimation in multi-view scenarios, including the 3D reconstruction of objects from a collection of photos, as well as its use in visual odometry, both in the monocular as well as the stereo cases. It is a technique that had been applied in photogrammetry for decades already when it became used in computer vision in the 1990s.

In the particular context of geometric computer vision, as illustrated in figure 2.8, bundle adjustment is a method to obtain an improved estimate of the relative pose of M cameras *and* the 2D projections or 3D position of N observed points by minimizing a loss function into which all the measured data and all the unknowns (relative camera poses, 3D coordinates of the points) are integrated.

Bundle adjustment ensures geometric consistency by minimizing a loss function Q which computes the sum of the reprojection errors (Triggs, McLauchlan, Hartley and Fitzgibbon 1999).

The reprojection error is defined as the difference between the initially estimated tracking position $\boldsymbol{m}$ and the projected position $\hat{\boldsymbol{m}}$ from the 3D world coordinate, as also illustrated in figure 2.8. In the case of perfect tracking results, the difference should be

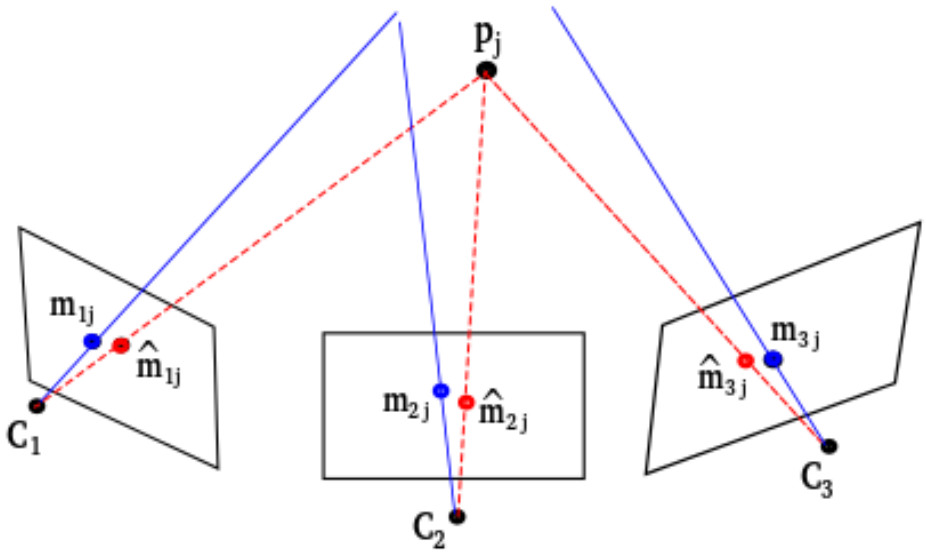

Figure 2.8: The illustration of bundle adjustment mechanism by minimizing the reprojection errors. A 3D point $\boldsymbol{p}_j$ is the estimated world point from a number of rays which might not exactly intersect at one point. The *reprojection error* of keypoint-j in frame-i is given by the distance in frame-i between the initially measured position $\boldsymbol{m}_{ij}$ (marked as blue) and the reprojected position $\hat{\boldsymbol{m}}_{ij}$ (marked as red) from the 3D world coordinate.

zero. The reprojection error for keypoint-j in frame-i, denoted by ε_{ij}, is given by

$$\varepsilon_{ij} = d(\boldsymbol{m}_{ij}, \hat{\boldsymbol{m}}_{ij}) = d(\boldsymbol{m}_{ij}, \pi(\boldsymbol{p}_j, i)) \tag{2.25}$$

where $d(\boldsymbol{a}, \boldsymbol{b})$ describes the Euclidean distance between point $\boldsymbol{a}$ and point $\boldsymbol{b}$, and $\pi(\boldsymbol{p}_j, i)$ refers to the projection operation of 3D world point $\boldsymbol{p}_j$ to frame-i, as described in equations 2.13 and 2.14.

The widely used loss function Q of the bundle adjustment, which has to be minimized, computes the sum of the square of the reprojection errors as follows

$$Q = \sum_{i=1}^{M}\sum_{j=1}^{N} \varepsilon_{ij}^2 = \sum_{i=1}^{M}\sum_{j=1}^{N} d^2(\boldsymbol{m}_{ij}, \pi(\boldsymbol{p}_j, i)) \tag{2.26}$$

In general, we can use some robust function $\rho(\boldsymbol{m}_{ij}, \pi(\boldsymbol{p}_j, i))$ as the loss function of the bundle adjustment.

$$Q = \sum_{i=1}^{M}\sum_{j=1}^{N} \rho(\boldsymbol{m}_{ij}, \pi(\boldsymbol{p}_j, i)) \tag{2.27}$$

Instead of minimizing the reprojection errors, a new method named *joint epipolar tracking (JET)* was proposed by Bradler (Bradler 2016) (Bradler et al. 2017) which minimizes the photometric errors and simultaneously optimizes both the correspondences

and pose estimates. The PMO framework employs an updated JET implementation as published in (Bradler et al. 2017). Similar to JET, Lucey et al (Alismail, Browning and Lucey 2016) also proposed a modification to the classical bundle adjustment approach by minimizing the photometric errors instead of the reprojection errors. However, the method proposed in (Alismail et al. 2016) requires the prior availability of the tracked world point positions, while JET works directly on image coordinates.

2.4.2 Random Sample Consensus (RANSAC)

RANSAC is a method that iteratively estimates parameters of a mathematical model from a set of observed data that contains outliers. Since it is unknown which part of the observations belong to the outliers, RANSAC randomly chooses a part of the observations as the basis to estimate the sought model parameters and then RANSAC checks whether the estimated parameters are consistent to the rest of the observations. The above steps are iteratively executed until the estimated parameters are consistent with the highest majority of the observations.

In the context of visual odometry, RANSAC is often employed to handle the presence of wrong correspondences between two frames when estimating the relative pose between the two frames. However, RANSAC has several major drawbacks, such as the high computational time and the difficulty in tuning the RANSAC thresholds. In contrast to the popular belief, I show in my proposed PMO framework that a robust monocular visual odometry framework can be built without RANSAC.

2.5 State of the art in monocular SLAM/odometry

Moravec (Moravec 1980) laid one of the first pipelines of visual odometry. One of the first major applications of visual odometry was to support NASA exploration mission to Mars (Maimone, Cheng and Matthies 2007). As the surface of Mars can be assumed to be irregular-terrain shape, a wheel odometry could not work satisfactorily to provide the self-navigation of the explorer robot. A visual odometry system was then developed to accommodate the explorer robot in any terrain-shape environment. A majority of Moravec's pipeline is still relevant to the current state of the art in visual odometry. Although the research to estimate pose from sequential frames had been ongoing for years, the term visual odometry started to be widely used at least since the work by Nister et al (Nistér et al. 2004).

In this section, I discuss major monocular visual odometry/SLAM frameworks that have been proposed during the recent 10 years. Table 2.1 shows and compares ten different frameworks, including the proposed PMO framework. I provide a brief explanation for each framework. For deeper comparison and analysis of different monocular visual odometry/SLAM frameworks, I refer to survey papers such as (Younes, Asmar and Shammas 2016) and (Taketomi, Uchiyama and Ikeda 2017).

Table 2.1: Characteristics of the state of the art monocular visual SLAM/odometry frameworks

Method	**Year**	**Dense/ sparse**	**Direct/ indirect**	**Keypoint detector**	**Descriptor**	**Epipolar matching**
PTAM	2007	sparse	indirect	FAST	patch	yes
DTAM	2011	dense	direct	n.a.	n.a.	n.a.
SVO	2014	sparse	hybrid	FAST	patch	no
LSD-SLAM	2014	dense	direct	n.a.	n.a.	n.a.
MLM-SFM	2014	sparse	indirect	FAST+GFTT	ORB	yes
ORB-SLAM	2015	sparse	indirect	FAST	ORB	yes
FTMVO	2015	sparse	indirect	GFTT	patch	yes
DSO	2016	sparse	direct	gradient-threshold	patch	yes
PMO	2017	sparse	indirect	GFTT+GETT	patch	yes
MVOCE	2017	sparse	indirect	GFTT	patch	yes

Table 2.2: Comparison of PMO with other monocular visual SLAM/odometry frameworks.

Method	**Pre-estimation of pose**	**Bundle adjustment**	**RANSAC**	**KITTI: Rot.Err.**	**KITTI: Trans.Err.**
PTAM	constant velocity	yes	yes	-	-
DTAM	constant velocity	n.a.	n.a.	-	-
SVO	last pose	yes	yes	-	-
LSD-SLAM	last pose	n.a.	n.a.	-	-
MLM-SFM	constant velocity	yes	yes	0.0057	2.54
ORB-SLAM	constant velocity	yes	yes	-	-
FTMVO	last pose	no	yes	0.0049	2.24
DSO	last pose	yes	no	-	-
PMO	pyr+smp	no	no	0.0051	2.05
MVOCE	last pose	yes	yes	0.0031	1.29

2.5.1 PTAM: Parallel Tracking and Mapping, 2007

Parallel Tracking and Mapping (PTAM) (Klein and Murray 2007) aims at estimating camera pose in an unknown scene using a hand-held camera. The paper argued that tracking keypoints and mapping the keypoint tracks into 3D world points should better be treated as two independent processes executed in parallel threads in order to effectively run in real time. In that way, the tracking process can be executed without the burden of updating the map. At the same time, the mapping process does not need to be updated at every frame. Considering the fact that there is a high similarity between two consecutive frames, the mapping process could be updated only on keyframes. This would allow mapping process to employ more accurate method up to the allowed time gap between two consecutive key frames.

The tracking task is implemented in two stages: first tracking a small number of keypoints on the coarse image and followed by camera pose refinement, and then subsequently tracking a larger number of keypoints on the original image. The camera pose is first initialized from a motion model and then refined after each stage of tracking task. The new keypoint correspondences are initialized with an epipolar search. RANSAC is performed to reject outliers when estimating the essential matrix between two keyframes and refinement over multiple frames is performed using bundle adjustment.

2.5.2 DTAM: Dense Tracking and Mapping, 2011

DTAM (Newcombe, Lovegrove and Davison 2011) aims to provide dense and real-time tracking and mapping using a monocular hand-held camera. Instead of calculating the optical flow and employing feature tracking, it uses a direct method by exploiting the whole image information. DTAM starts a new style of visual odometry/SLAM which inspires other dense frameworks, like later in LSD-SLAM (Engel, Schöps and Cremers 2014). The real-time capability is largely due to the continuous improvement of the GPU processor technology. The mapping method relies on the global energy minimization framework where the energy consists of the photometric error and spatial regularization term. The tracking method employs an initial estimated rotational pose, in a way similar to the proposed pitch-yaw-roll estimator (see section 7.3.1). A full pose refinement is done using Lucas-Kanade like optimization (see section 4.2.2). DTAM shows promising results, especially in augmented reality application.

2.5.3 SVO: Semi-direct monocular visual odometry, 2014

SVO (Forster, Pizzoli and Scaramuzza 2014) was proposed as a semi-direct framework because the tracking is done by keypoint-based matching but the mapping is based on the direct method. SVO takes the inspiration from PTAM by following the parallel processes of the matching and the tracking.

The initial pose estimation of SVO is based on the sparse photometric error minimization.

The matches are first searched along the epipolar line. The position of keypoint correspondences is then optimized using local optimization to find a better match by violating epipolar geometry. An outlier detection mechanism based on the probabilistic depth estimation of each keypoint is employed before approving a new 3D world point during mapping.

2.5.4 LSD-SLAM: Large-scale direct monocular SLAM, 2014

LSD-SLAM (Engel et al. 2014) was proposed as a full direct framework, similar to DTAM. The LSD-SLAM framework is based on a feature-less (direct) approach which directly minimizes the photometric errors when matching two frames to estimate the relative pose. LSD-SLAM is able to build large-scale and consistent maps in real-time. However, it is harder to remove outliers in a direct method. The scale estimation in the monocular LSD-SLAM is based on the correlation between the scene depth and the tracking accuracy, hence it is prone to the scale drift effect in a long sequence. This might well explain why only the stereo version of LSD-SLAM is available in the KITTI odometry benchmark.

2.5.5 ORB-SLAM, 2015

ORB-SLAM (Mur-Artal et al. 2015) was proposed as a feature-based (sparse) monocular SLAM framework for both indoor and outdoor environments. As the name suggests, the keypoints are represented by the ORB descriptor (Rublee et al. 2011). The concept of ORB-SLAM basically follows PTAM (Klein and Murray 2007) which splits the matching and the mapping processes. ORB-SLAM adds another thread for loop closing mechanism thus splitting the overall processes into three threads. The same set of ORB features are used in tracking, mapping, relocalization, and loop closing. The points and keyframes of the reconstruction are selected based on a survival of the fittest strategy. The ORB-SLAM source code is made public and it has been widely utilized in a variety of projects.

2.5.6 DSO: Direct sparse odometry, 2016

DSO (Engel, Koltun and Cremers 2016) has been proposed by the same group which developed LSD-SLAM (Engel et al. 2014). While LSD-SLAM is a dense and direct approach, DSO is unique because it offers a new style as the first sparse and direct visual odometry method. DSO combines a fully direct approach (minimizing a photometric error) with consistent, joint optimization of all model parameters, including geometry and camera motion.

DSO is able to sample all pixels in an image as long as it has sufficient intensity gradient, including edges or smooth intensity variations on relatively plain texture. As a result, DSO can produce a denser mapping than other sparse methods, such as PTAM

and ORB-SLAM. A full photometric calibration, accounting for exposure time, lens vignetting, and non-linear response functions is proposed.

2.5.7 MLM-SFM: High Accuracy Monocular SFM and Scale Correction for Autonomous Driving, 2014

MLM-SFM (Song and Chandraker 2014) was proposed as a visual odometry framework with the focus on the traffic scenarios from a monocular camera attached on an ego-car. Similar to PMO, the steady-state architecture is split into three parts: pose module, bundle adjustment module and epipolar update. The global bundle adjustment is employed for the last five keyframes.

MLM-SFM assumes that the camera height over ground is already known. The scale drift is handled by estimating the monocular scale from multiple cues, i.e. sparse and dense ground features. The confidence levels from all cues are calculated on each frame, resulting in a per-frame estimation based on the visual data observation. MLM-SFM achieves real-time performance with an average computation of 30 fps.

2.5.8 FTMVO: Fast Techniques for Monocular Visual Odometry, 2015

FTMVO (Mirabdollah and Mertsching 2015) was proposed as a monocular visual odometry specifically designed for the driving scenario. The tracked keypoints are classified based on their depth into far keypoints and close keypoints. The far keypoints are subjects to epipolar geometry constraints while the close keypoints have projective constraints. RANSAC is employed in FTMVO when estimating the essential matrix during pose estimation step.

The monocular scale is estimated using the sparse tracked ground points. The keypoints on the ground are detected and matched to other frames as proposed in libviso framework (Geiger, Ziegler and Stiller 2011). However, the method presented in FTMVO estimates only the corrected scale and does not estimate the ground plane by assuming that the camera is always parallel to the ground plane. Furthermore, it assumes that the majority of keypoints on the bottom half of the camera view are ground points. This assumption can easily be invalid when there is a significant presence of obstacles such as other cars, vegetation or fences.

2.5.9 MVOCE: Monocular Visual Odometry with Cyclic Estimation, 2017

MVOCE (Pereira, Ilha, Luft, Negreiros and Susin 2017) was proposed as a cyclic keypoint tracking method built to provide a monocular visual odometry in the driving scenario. The cyclic bundle adjustment implementation is based on three consecutive frames

to estimate the egomotion of the ego-vehicle. The perspective image transformation is employed to reject matching outliers due to similarity along the epipolar line. The monocular scale estimation is based on a carefully selected ground point after passing multi-attribute checks.

This paper was published after I reported my results in IV 2017 (Fanani, Stuerck, Barnada and Mester 2017), IMAVIS 2017 (Fanani, Stürck, Ochs, Bradler and Mester 2017) and is currently (December 2017) the only published monocular method that ranks better in KITTI than my approach. Repetitive structures along the epipolar line, which commonly cause wrong matches, are handled by projecting the previous frame to the current frame with a set of different distances. If the matches are not consistent, they are classified as dubious matches and discarded.

Chapter 3

Predictive Monocular Odometry (PMO): An overview

3.1 Introduction to PMO

Visual odometry (VO) and SLAM are core tasks in an area where a certain set of standard mainstream techniques have evolved throughout the recent 2 to 3 decades of intensive research: complex keypoint detectors and keypoint descriptors, descriptor-based matching, RANSAC-supported relative-pose estimation and multi-frame bundle adjustment are just some dominant key terms when discussing state-of-the-art VO/SLAM approaches. Switching from geometry-based optimization to 'direct methods' is only one recent example of the paradigm shifts that occur from time to time.

This chapter presents the overview of the main components of the proposed PMO framework. In contrast to the traditions in building VO/SLAM frameworks, PMO is developed without employing descriptor-based matching and RANSAC. Furthermore, PMO is able to achieve good results even without employing multi-frame bundle adjustment method. The description of PMO is based on the combination of work previously published in (Fanani et al. 2016), (Fanani, Stuerck, Barnada and Mester 2017), and (Fanani, Stürck, Ochs, Bradler and Mester 2017). Figure 3.1 visualizes some components of the approach, and presents an example result of the estimated course of the ego-car in the KITTI odometry sequence.

3.2 Approach overview

The overall structure of the proposed method is shown in fig. 3.2. The propagation based tracking method iteratively processes keypoint correspondence from two consecutive frames $n-1$ and frame n, and it yields an output of the predicted keypoint position in frame $n+1$. The tracking of new keypoints is initialized as a two-frame epipolar guided matching process as proposed in (Fanani et al. 2015). The initial estimated pose between

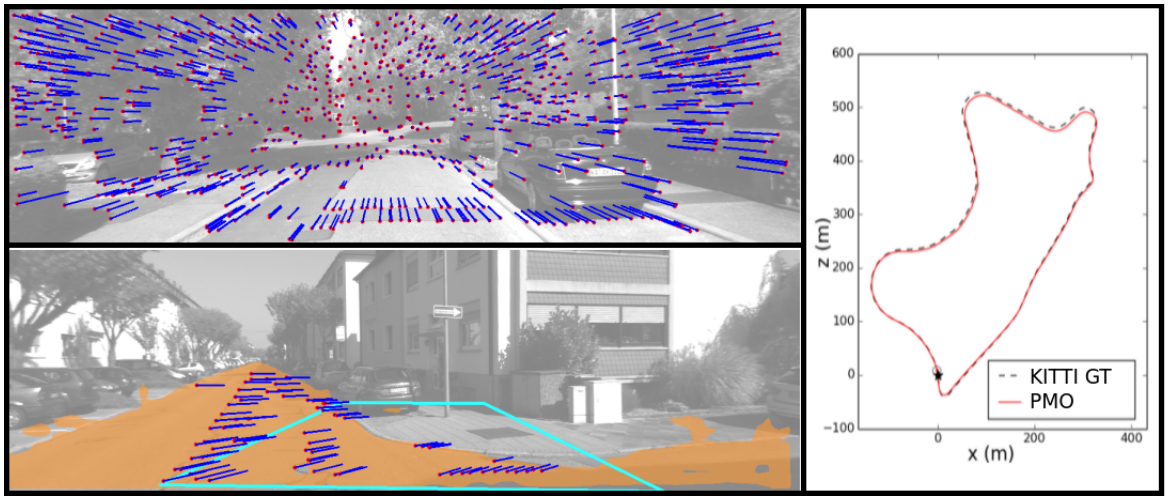

Figure 3.1: *Top-left*: optical flow vectors needed to perform unscaled VO; *bottom-left*: ground plane estimation utilizing CNN-based street masks to solve monocular scale ambiguity; *right*: reconstructed trajectory course on the KITTI dataset against the ground truth data. *Best viewed in color.*

two consecutive frames is computed using a combination of rotation estimation and statistical pose prediction as described in detail in section 3.2.1. Building on the initial estimated pose and the known camera calibration, both the keypoint correspondences and the relative pose between two frames are simultaneously refined using a method denoted as 'joint epipolar tracking' (JET) developed in (Bradler 2016) and (Bradler et al. 2017) which is summarized in section 3.2.4.

Keypoints are always tracked guided on the epipolar structure given by the estimated pose. The propagation based tracking method is designed to track keypoints in the static environment and to reject keypoints on independently moving objects, as implied by the nature of the epipolar geometry. Furthermore, an SSD check is always performed on a new keypoint correspondence in order to accept only matches with high photometric similarity level and to reject outliers. The pseudocode of the propagation based tracking method is listed in algorithm 1 (page 29) and each step is further discussed in Sections 3.2.1 – 3.2.8.

3.2.1 Keypoint-free initial relative pose estimation

One of the main tasks of an advanced automotive vision system is to continuously estimate the relative pose between two subsequent frames. A specific feature of my approach is that I combine different information about the next relative pose when frame $n+1$ has been captured, but in a first step *without* using keypoints. I use the *initial relative pose estimate* to stabilize and robustify the computation of the new corresponding keypoint positions.

To estimate the frame-to-frame relative rotation, I calculate the pitch-yaw-roll angles between two frames as proposed by Barnada *et al* (Barnada et al. 2015). The estimation

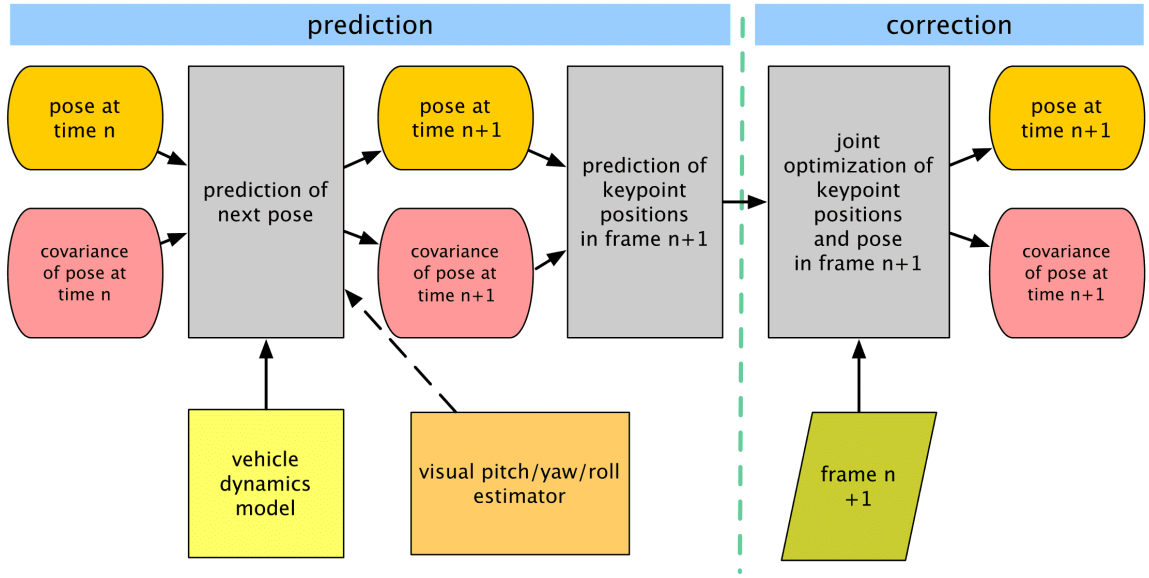

Figure 3.2: Block diagram of a generalized propagation scheme including the main elements of PMO.

Algorithm 1 Propagation based tracking algorithm

procedure PROPAGATION BASED TRACKING
Initialization :
Initial pose estimation between frame $n-1$ and n
Generation of new keypoints in sparsely covered areas at frame $n-1$
Prior based matching of new keypoints
Joint epipolar tracking between frame $n-1$ and n
Speed estimation
Keypoint triangulation
Main loop :
    while next frame $n+1$ is available **do**
        Initial pose estimation between frame n and $n+1$ (Sec. 3.2.1)
        Keypoint propagation to frame $n+1$ (Sec. 3.2.7)
        Joint epipolar tracking between frame n and $n+1$ (Sec. 3.2.4)
        Keypoint trajectory update
        Generate new keypoints in sparsely covered areas in frame n
        Prior based matching of new keypoints (Sec. 3.2.3)
        Speed estimation (Sec. 3.2.5)
        Keypoint triangulation (Sec. 3.2.6)
    end while
end procedure

of the relative translation is obtained by applying an extension of work by Bradler *et al* (Bradler et al. 2015) which proposes a one-step relative pose predictor, hence providing the prediction of relative pose information based on the previous relative pose information. This pose predictor also provides the rotation estimation on rare cases when the pitch-yaw-roll estimator fails to provide rotation estimation.

In contrast to PTAM which employs efficient second order minimization (Benhimane and Malis 2004) to the blurred version of the images to predict the rotation, the proposed rotation estimator in PMO (Barnada et al. 2015) uses the original image and looks only at the far field. The far field windows can be made adaptive to the depth structure when the depth map is available. Furthermore, the rotation estimator in PMO is based on the phase correlation which is a particularly robust motion estimation method.

3.2.2 Generation of new keypoints in sparsely covered areas

As the car moves forward, new objects will become visible and consequently, a new set of keypoint candidates appears to track. Therefore, most of the time, there are two sets of keypoints during the tracking process: (1) old keypoints, which have already been tracked previously, and (2) new keypoints which are detected newly for the first time. It is important to note that the generation of new keypoints is only performed when the car is not in a static mode (see section 7.5).

New keypoints are detected and chosen based on the good-feature-to-track (GFTT) score to find corner points as proposed by (Shi and Tomasi 1994). On top of that, edgel points are also chosen based on a method from (Piccini et al. 2014). Thus the PMO keypoints are a combination of corner points and edgel points. The keypoint generator is continuously applied on each frame in such a way that new keypoints are generated on those parts of the image where the density of old keypoints is low. In this way, I always try to maintain a good distribution of keypoints on each frame.

As indicated in algorithm 1, the new keypoints are detected in frame n only after the final set of the correspondences between frame n and frame $n+1$ are determined and also after the relative pose estimate between the two frames is optimized using JET (see section 3.2.4). The matches of the newly detected keypoints are searched in frame $n+1$ based on the prior-based epipolar matching method (see section 3.2.3).

3.2.3 Prior-based matching of new keypoints

Dealing with new keypoints is more challenging than dealing with the old keypoints which have been tracked before. This is because every old keypoint is already associated with 3D position information from its previous tracking result, but new keypoints have no associated depth information. In order to optimally initialize the differential search for the corresponding keypoint position, I employ a method proposed in (Fanani et al. 2015) which is based on computing the *distribution of motion* in a regarded area by an extended phase correlation method (Ochs et al. 2015).

The prior-based matching method is specifically applied to the new keypoints to provide the first match of each keypoint. In other words, it serves as an initialization method before the keypoints are processed by the propagation based tracking.

It is important to note that the matches associated with the new keypoints are considered only as candidates for keypoints correspondences. Only keypoints which have been successfully tracked on at least three consecutive frames are considered as credible. This ensures that keypoints on independently moving objects (IMOs) or gross mismatches are largely excluded from the set of keypoint trajectories.

3.2.4 Pose and correspondence refinement by joint epipolar tracking

The initial estimates of the relative pose and the keypoint correspondences between two frames can further be refined by exploiting the fact that they are mutually constrained by each other. On one hand, a given relative pose imposes an epipolar constraint on the keypoint correspondences. On the other hand, a set of given keypoint correspondences induces a relative pose.

The PMO framework uses a novel approach named joint epipolar tracking which simultaneously applies differential ('Lucas-Kanade' style) epipolar-guided photometric matching to all keypoints as proposed in (Bradler 2016) and extended in (Bradler et al. 2017). By applying joint epipolar tracking, PMO obtains both an optimum set of keypoint correspondences (in terms of the SSD) as well as an improved relative pose, jointly complying with the epipolar relation.

3.2.5 Scale estimation

As mentioned in section 2.2.1, monocular visual odometry requires the ability to deal with the scale ambiguity problem, or equivalently to transform the estimated unscaled poses into correctly scaled poses. While *propagating* the scale from frame to frame is possible, it is very prone to the scale drift effect. In contrast, the proposed scale estimation scheme presented in chapter 8 combines cues from both dense and sparse ground plane *estimation* which makes the proposed method robust towards varying availability and distribution of trackable ground structure.

3.2.6 Triangulation

Given a keypoint correspondence between frame $n-1$ and n, the corresponding 3D point can be determined by triangulation, as analyzed in detail in chapter 6. I assume that the point correspondence fulfills the epipolar geometry, which means that it complies with the estimated relative pose. The 3D point in the camera coordinate system is determined by computing the intersection of the rays which pass through the centers

of projection of the cameras at time $n-1$ and n, and their respective image points. This triangulation concept is discussed in detail in chapter 6. It is important to note that the triangulation in PMO is based on keypoint matches that are guaranteed and constrained to comply with the epipolar geometry.

3.2.7 Keypoint projection

Upon obtaining the 3D position of the keypoint through triangulation of keypoint correspondence at frames $n-1$ and n, we can predict the position of the keypoint at frame $n+1$ by propagating the 3D coordinate of the keypoint into the image plane of frame $n+1$. This step is discussed in section 6.5. The predicted keypoint position in frame $n+1$ is further refined by using the joint epipolar tracker, which simultaneously applies epipolar Lucas-Kanade matching.

3.2.8 Independently moving objects (IMO) detection

A visual odometry framework should be able to detect moving objects because the localization estimates should be optimized only based on the static world assumption (conformity with the epipolar relation). For this, I propose a method for detecting IMOs using epipolar consistency checks and depth verifications, as presented in chapter 9. I utilize the IMO candidate patches generated using a CNN-based classifier, which has been developed in our group and has been published in (van den Brand et al. 2016). In principle, each well-performing vehicle instance segmentation method can be used to provide the IMO candidate patches. The obtained motion labels (IMO/static) from the method proposed in chapter 9 are then propagated over time using the combination of motion cues and appearance-based information of the IMO candidate patches.

3.3 Comparison of the present method (PMO) against classical and state-of-the-art methods

The presented PMO framework shares some similarities but also offers novelties as compared to other classical and state-of-the-art visual SLAM/odometry methods. This section compares PMO against other relevant methods.

PMO is categorized as a sparse method and therefore belongs to the same group with the methods of PTAM (Klein and Murray 2007) and ORB-SLAM (Mur-Artal et al. 2015). Instead of using only corner points like in PTAM and ORB-SLAM, PMO also uses edge points from GETT (Piccini et al. 2014) which are carefully chosen by incorporating the matching predictability along the epipolar line. This enables PMO to have a denser keypoint distribution (e.g. for mapping purpose) and it also gives PMO an extra robustness especially when the scene significantly contains line structures instead of corner structures.

When processing a new incoming frame, many state of the art methods use the estimated relative pose from the last frame to provide the pre-estimation of the pose of the new frame, such as in LSD-SLAM (Engel et al. 2014), DSO (Engel et al. 2016), and MVOCE (Pereira et al. 2017). Some other frameworks apply a constant velocity assumption for the pre-estimate of the pose. PMO presents a novel approach for the pre-estimation of the pose by combining the pitch-yaw-roll estimation (Barnada et al. 2015) and the statistical motion predictor (Bradler et al. 2015). This novel approach makes PMO capable of anticipating abrupt pose changes, for instances due to bumpy roads or a sudden turn.

The refinement of the pose estimate is usually carried out in almost all VO/SLAM frameworks by employing bundle adjustment, with a focus of minimizing the multi-view reprojection errors. In this regard, PMO proposes to *simultaneously* refine both the pose estimate *and* the keypoint correspondences using joint epipolar tracking (Bradler 2016) which minimizes the photometric errors.

Other frameworks, such as ORB-SLAM, FTMVO (Mirabdollah and Mertsching 2015), and MVOCE, rely on RANSAC to reject the keypoint outliers. Here, PMO shows that a robust visual odometry can be built without RANSAC by carefully choosing the keypoint matches which are conformant to the photometric consistency and the epipolar constraint.

The monocular scale is estimated in PMO using ground plane estimation with the assumption that the camera height over ground is known, which is a similar approach to MLM-SFM (Song and Chandraker 2014), FTMVO, and MVOCE. The irregularities found on the ground area need to be properly handled to have a robust scale estimation. Here, PMO employs segmented street areas by CNN classification to make sure that only ground areas are processed for the scale estimation.

In order to have a robust localization, IMOs need to be identified so that they can be excluded during the pose estimation step. PMO proposes a novel method to detect IMOs with the help of semantic car classification from a CNN; the CNN-based method for *car instance segmentation* had been presented by other member of our group in (van den Brand et al. 2016). Within PMO, the car instances segmented by (van den Brand et al. 2016) (or any other suitable instance segmentation methods) are analyzed and subsequently classified into static objects and IMOs.

As a result, PMO ranked best in the KITTI odometry benchmark among the other published monocular frameworks in the period of April-September 2017. Only recently during the writing of this dissertation, a new method named MVOCE (Pereira et al. 2017) (see section 2.5.9) took over the leading position from PMO.

3.4 List of PMO parameters

Table 3.1: List of PMO parameters.

Parameter	Value	Description	Page
τ_λ	25	Minimum eigenvalue of structure tensor.	54
$\tau_{r\lambda}$	0.15	Minimum eigenvalue ratio of structure tensor.	54
$\tau_{\theta e}$	30°	Minimum angle of GFTT epipolar score.	55
$\tau_{\alpha t}$	0.05°	Minimum triangulation angle.	71
z_{min}	0 m	Minimum accepted keypoint depth from camera.	83
z_{max}	200 m	Maximum accepted keypoint depth from camera.	83
h_{max}	3 m	Maximum accepted keypoint height below camera.	83
r_{3d}	0.4	Maximum deviation ratio from the mean.	84
τ_θ	0.008 rad	Maximum relative pitch angle.	93
τ_ψ	0.01 rad	Maximum relative yaw angle.	93
τ_φ	0.008 rad	Maximum relative roll angle.	93
τ_{sc}	1.0 m/s	Maximum translation scale in a static mode	96
τ_a	5°/s	Maximum rotation rate in a static mode	96
τ_{tr}	1.0	Maximum trace of covariance for 3D consistency check	99
τ_{hg}	0.2 m	Maximum keypoint height over ground for sGPE	116
τ_{ps}	0.99	Threshold for p-level.	134
τ_h	80%	Threshold for keypoint ratio in IMO patch.	135
τ_i	25%	Threshold for static backward check.	136
τ_{mc}	3	Minimum correspondences for depth verification.	140
τ_v	0.3	Maximum depth difference for parallel IMO.	140
τ_{rm}	0.4	Maximum ratio of moving keypoints	140
τ_{xy}	0.1	Maximum shift of static object.	141
$\tau_{dm,IMO}$	0.05	Threshold for IMO relative depth difference.	141
$\tau_{dm,static}$	0.01	Threshold for static relative depth difference.	141

Chapter 4

Statistical analysis of keypoint matching

This chapter discusses the statistical analysis of keypoint matching based on minimizing the photometric error between image patches. Two different approaches are discussed: discrete matching (block matching) and differential matching.

As PMO, the method presented in this thesis, uses photometric matching, when analyzing whether two patches match in the PMO implementation, I look into the photometric similarity/dissimilarity metric between these two patches. From the choices of photometric similarity/dissimilarity metrics presented in section 2.3.2, PMO specifically uses the sum of squared difference (SSD) of the pixel intensity between the two patches. Section 4.1.4 on testing the SSD gives us information whether a hypothetical match is credible. This applies both to discrete (block) matching, as well as to differential (Lucas-Kanade style) matching. Section 4.2.3 which regards the covariance of the motion error provides us with information about the spatial precision of a hypothetical match in the case of differential matching. We need both informations. If a match is not credible, its spatial precision is irrelevant. If the spatial precision of a match is too low, the match is worthless.

4.1 Statistics of block matching

4.1.1 Image model and noise

The images to be matched, s_n and s_{n+1} are modeled as being shifted copies of the same unknown image s_0 under the constant brightness assumption, plus two unknown independent realizations of noise n_1 and n_2 on top of s_0. We regard s_n at location $\boldsymbol{x}_1$ and s_{n+1} at location $\boldsymbol{x}_2$, as shown in figure 4.1, where the displacement vector is given by $\boldsymbol{g}$:

$$\boldsymbol{g} = \boldsymbol{x}_1 - \boldsymbol{x}_2 \tag{4.1}$$

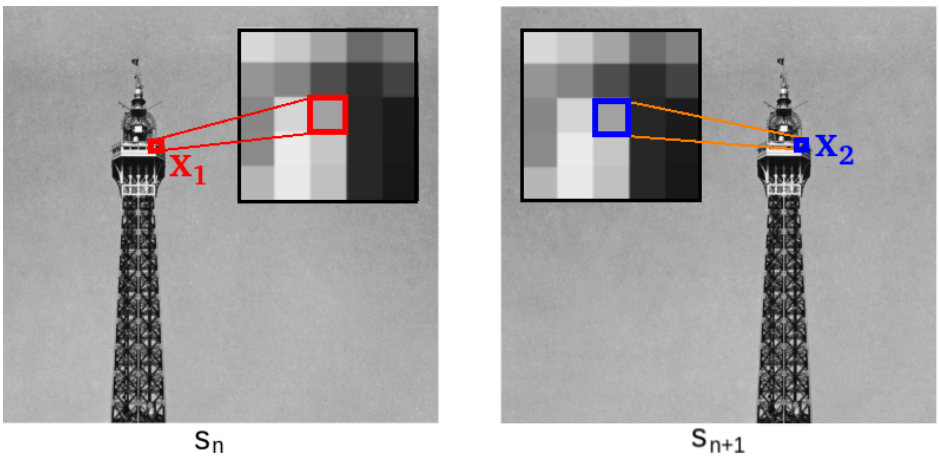

Figure 4.1: The illustration of matching pixels between $s_n(\boldsymbol{x}_1)$ and $s_{n+1}(\boldsymbol{x}_2)$.

$$s_n(\boldsymbol{x}_1) = s_0(\boldsymbol{x}_1) + n_1(\boldsymbol{x}_1) \tag{4.2}$$
$$s_{n+1}(\boldsymbol{x}_2) = s_0(\boldsymbol{x}_1) + n_2(\boldsymbol{x}_2) \tag{4.3}$$

The noise is assumed to be zero-mean white noise and independent and identically distributed (*i.i.d.*), meaning that all random variables have the same probability distribution and are mutually independent. Let σ_n^2 be the variance of the noise random variables n_1 and n_2. The following expected value and variance are valid for all pixel location $\boldsymbol{x}$.

$$\mathsf{E}\left[n_1(\boldsymbol{x})\right] = \mathsf{E}\left[n_2(\boldsymbol{x})\right] = 0 \tag{4.4}$$
$$\mathsf{Var}\left[n_1(\boldsymbol{x})\right] = \mathsf{Var}\left[n_2(\boldsymbol{x})\right] = \sigma_n^2 \tag{4.5}$$

4.1.2 Analysis of matching a single pixel

I analyze first the statistical model for matching a single pixel, and will subsequently extend this towards the statistical analysis of matching a whole set of pixels in section 4.1.3. Let us regard the pixel difference d for the case of a perfect match, that is: if the pixel at location $\boldsymbol{x}_1$ truly matches the pixel at location $\boldsymbol{x}_2$, which due to the noise, does *not* imply that their pixel values are identical.

$$d(\boldsymbol{x}) = s_n(\boldsymbol{x}_1) - s_{n+1}(\boldsymbol{x}_2) \tag{4.6}$$
$$= n_1(\boldsymbol{x}_1) - n_2(\boldsymbol{x}_2) \tag{4.7}$$

with the expectation value

$$\mathsf{E}\left[d(\boldsymbol{x})\right] = \mathsf{E}\left[n_1(\boldsymbol{x}_1)\right] - \mathsf{E}\left[n_2(\boldsymbol{x}_2)\right] = 0 \tag{4.8}$$

because the noise is independent and identically distributed. The variance of d, denoted as σ_d^2, is given by

$$\begin{aligned}
\sigma_d^2 &= \mathsf{Var}\left[d(\boldsymbol{x})\right] \\
&= \mathsf{E}\left[\left(d(\boldsymbol{x}) - \mathsf{E}\left[d(\boldsymbol{x})\right]\right)^2\right] \\
&= \mathsf{E}\left[d^2(\boldsymbol{x})\right] \\
&= \mathsf{E}\left[\left(n_1(\boldsymbol{x}_1) - n_2(\boldsymbol{x}_2)\right)^2\right] \\
&= \mathsf{E}\left[n_1^2(\boldsymbol{x}_1)\right] - 2\,\mathsf{E}\left[n_1(\boldsymbol{x}_1)\,n_2(\boldsymbol{x}_2)\right] + \mathsf{E}\left[n_2^2(\boldsymbol{x}_2)\right] \\
&= \mathsf{E}\left[n_1^2(\boldsymbol{x}_1)\right] - 2\,\mathsf{E}\left[n_1(\boldsymbol{x}_1)\right]\,\mathsf{E}\left[n_2(\boldsymbol{x}_2)\right] + \mathsf{E}\left[n_2^2(\boldsymbol{x}_2)\right] \\
&= \mathsf{E}\left[n_1^2(\boldsymbol{x}_1)\right] + \mathsf{E}\left[n_2^2(\boldsymbol{x}_2)\right] \\
&= \mathsf{Var}\left[n_1(\boldsymbol{x}_1)\right] + \mathsf{Var}\left[n_2(\boldsymbol{x}_2)\right] \\
&= 2\sigma_n^2
\end{aligned} \tag{4.9}$$

which indicates that the variance of the *pixel-to-pixel* difference σ_d^2 is twice of the variance of the mere pixel noise σ_n^2.

It is important to note that up to this point, the derivation is valid for any form of the pdf of the noise, Gaussian or not.

As long as only 1st and 2nd order statistics of the image and noise processes are known, only statements about the 1st and 2nd order statistics of *derived* processes, such as the image difference noise, can be made. However, if the *distributions* of the input processes are known (or assumed), conclusions on some distribution aspects of the derived processes can also be made.

Now let us assume that the noise is Gaussian distributed. Let $p_n(n)$ be the distribution of the pixel noise

$$p_n(n) = \frac{1}{\sqrt{2\pi\sigma_n^2}} \cdot \exp\left[-\frac{n^2}{2\sigma_n^2}\right] \tag{4.10}$$

With that assumption, then d is also Gaussian distributed because d is the difference between two Gaussian-distributed noise. Let $p_d(d)$ be the distribution of the pixel noise and pixel image difference

$$p_d(d) = \frac{1}{\sqrt{2\pi\sigma_d^2}} \cdot \exp\left[-\frac{d^2}{2\sigma_d^2}\right] \tag{4.11}$$

where $\sigma_d^2 = 2\,\sigma_n^2$ as shown in equation 4.9. The discussion for other noise distributions, such as Laplacian distribution, is presented in (Aach, Kaup and Mester 1993).

4.1.2.1 Hypothesis test for a single pixel

Given two pixels with pixel intensity difference d, we would like to test whether, given the variance of the image noise σ_n^2, this is a credible match. For this purpose we design a statistical test. The resulting test is of course trivial, but it serves to explain the principle subsequently used for testing the matching of a *block* of pixels.

I define a null hypothesis H_0 that supposes that the pair of pixels, whose intensity difference is d, is a match. The alternative hypothesis is denoted by H_1 which supposes that the pixel pair is not a match. For the null hypothesis, that is: the case of the match being true, conclusions on the distribution of different test statistics can be drawn; for the case of the counter-hypothesis H_1 (no match), this is much more difficult, if possible at all. A *test statistic* s is a scalar number computed on the available data which becomes the basis for making the decision to test the hypothesis. We relate this value of s to the distribution of this test statistic under the two competing hypotheses H_0 and H_1.

Let us assume now that we set the pixel difference d as the test statistic. It is important to note that choosing the pixel difference d is just *one* possible choice as a test statistic s.

$$s \stackrel{def}{=} d \tag{4.12}$$

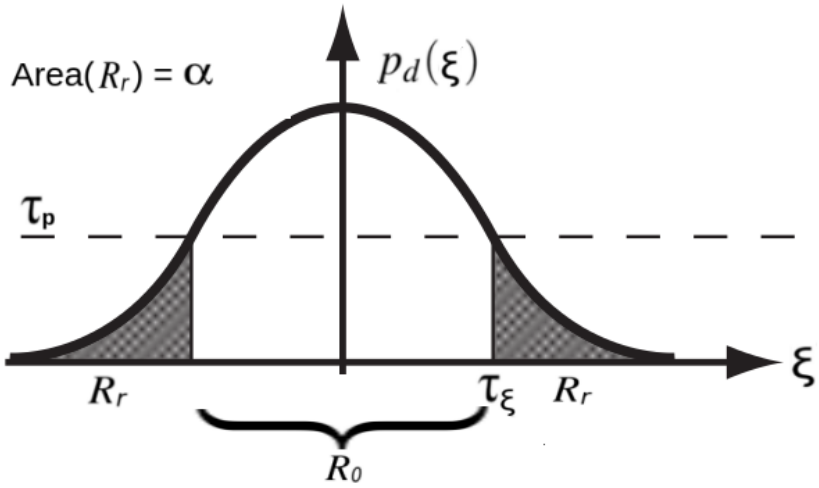

Figure 4.2: The determination of the threshold in a hypothesis test.

By looking into the probability distribution function (pdf) $p_d(\xi)$, the possible value range for the test statistic can be subdivided in two regions: a region of acceptance R_0 and a region of rejection R_r. Each subdivision of the complete value range gives a probability that a random variable from that distribution falls into region R_0 (acceptance) and a complementary probability that the random variable falls into region R_r (rejection), under the null hypothesis.

The standard situation in statistical testing is that there are *two* distributions of the test statistic s, one for the case that H_0 is true, the other for the case that H_1 is true. We do not have this situation here; we can only specify the distribution of the test statistic for the hypothesis H_0 (true match). We obtain a *one-sided test*, based only on the distribution of the chosen test statistic d for the case that H_0 is true.

The pdf of d shown in figure 4.2 is only an example. Now we look not only for d, but for the general test statistic s. All the following examples show distributions for the case of the null hypothesis H_0 being true. In the general case of a test statistic s,

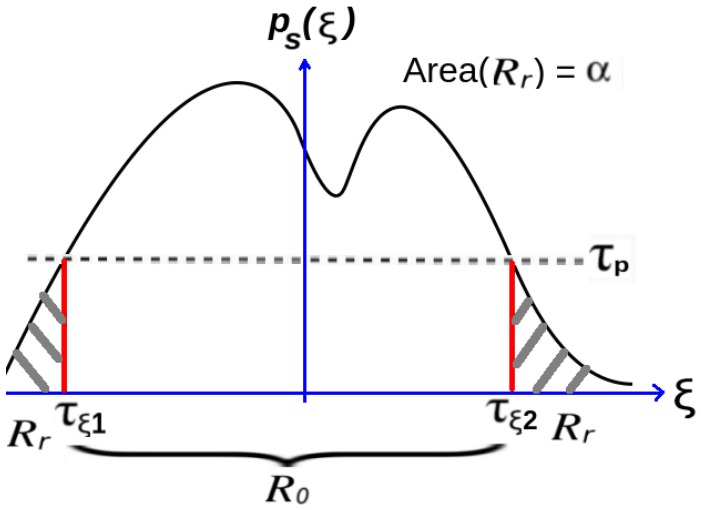

Figure 4.3: The determination of the threshold in a hypothesis test of a given distribution.

for example as shown in figure 4.3, we use a dummy variable ξ[1] as the name of the horizontal axis.

The Neyman-Pearson criterion (Neyman and Pearson 1928), (Papoulis 1990, p. 370) tells us to choose the region R_r in such a way that

- the integral of the distribution integrated over R_r is a user-specified value α (often also the symbol p is used, "p-value"), also denoted as the 'significance level'.

$$\int_{R_r} p_s(\xi \mid H_0)\, d\xi \stackrel{!}{=} \alpha \tag{4.13}$$

  This value α is the probability that hypothesis H_0 is rejected although it is actually true.

- in the case of a significance test (one sided test), the region should be chosen such that all the values which have a value of the probability density below a threshold τ_p are summarized in R_r. This leads to the determination of the threshold τ_p which is chosen such that the criterion in equation 4.13 is fulfilled.

If the test statistic s falls into the region R_0, then the null hypothesis H_0 is accepted, and vice versa.

$$p_s(\xi|H_0) \underset{H_1}{\overset{H_0}{\gtrless}} \tau_p \tag{4.14}$$

By setting τ_p, the thresholds on the ξ-axis, denoted as τ_ξ, are also automatically selected which define the borders of the regions R_0 and R_1. The number of thresholds τ_ξ depends on the pdf and the τ_p value. For the illustration given in figure 4.3, we have two thresholds $\tau_{\xi 1}$ and $\tau_{\xi 2}$. The test statistic s is to be tested against the τ_ξ thresholds.

[1] ξ is used as the dummy variable when writing and drawing a pdf.

For the case of analyzing the matching of *single* pixels, under the assumption that H_0 is true, figure 4.4 shows the pdf of d (left image), in comparison to the pdf of $|d|$ (middle image) and d^2 (right image). In principle, each of them can be chosen as a test statistic.

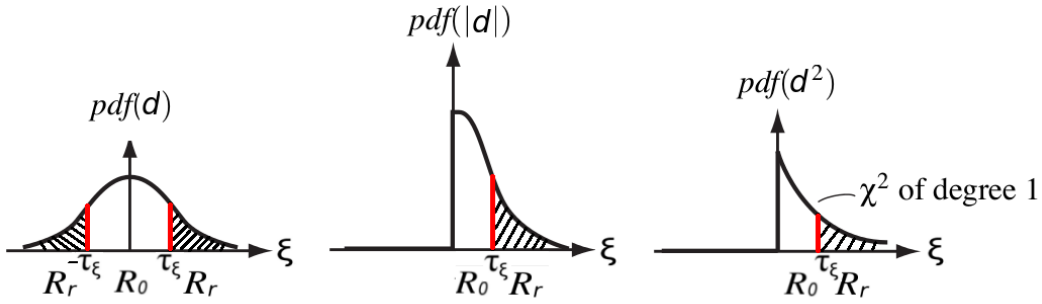

Figure 4.4: The probability density functions for d, $|d|$, and d^2 under the H_0 hypothesis, with the illustration of both acceptance (R_0) and rejection (R_r) regions for significance tests.

If we specifically look into d^2, the distribution of d^2 leads to a scaled χ^2 (chi-square) distribution with one degree of freedom since it only depends on one variable.

As the χ^2 distribution is important in the following, I insert a brief summary on this family of distributions here. Figure 4.5 shows the χ^2 distribution for different degrees of freedom.

The sum of the squared values of N *i.i.d.* zero-mean Gaussian random variables x_i with unit variance has a χ^2 distribution with N degrees of freedom, commonly written as $\chi^2(N)$.

$$y = \sum_{i=1}^{N} x_i^2 \quad \rightarrow \quad y \textit{ is distributed as } \chi^2(N) \tag{4.15}$$

The degree of freedom of such χ^2 distribution is determined by the number of independent variables affecting y. The expected value and the variance of a χ^2 random variable y with N degrees of freedom are given by

$$\mathsf{E}[y] = N \tag{4.16}$$

$$\mathsf{Var}[y] = 2\ N \tag{4.17}$$

To be compatible with the χ^2 distribution, we need to normalize d^2 to make the variance σ_d^2 equals to one. Thus $\frac{d^2}{\sigma_d^2}$ is chi square distributed with one degree of freedom (see figure 4.5 for an illustration of that particular distribution). Now we have a definition for the test statistic s.

$$s \stackrel{def}{=} \frac{d^2}{\sigma_d^2} \sim \chi_1^2 \tag{4.18}$$

We discuss more about the χ^2 distribution in the next section when we match two patches instead of matching two pixels.

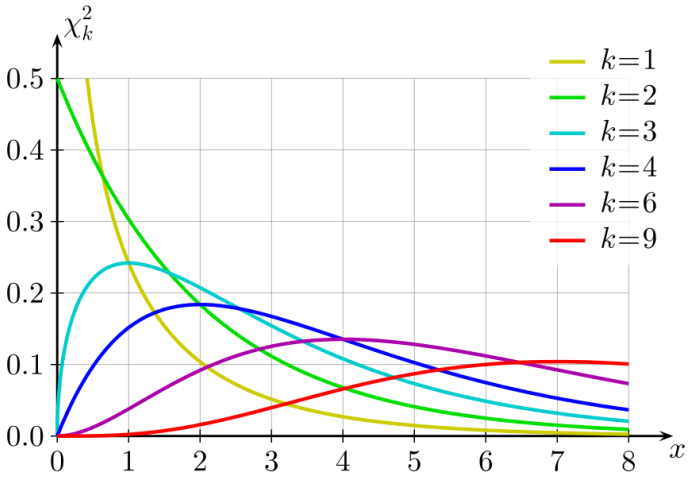

Figure 4.5: Chi-square distribution for different degrees k of freedom, taken from (Wikimedia 2017)

4.1.3 Analysis of matching patches of multiple pixels

After analyzing the statistical model for matching a single pixel, now we extend this towards the statistical analysis of matching a whole set of pixels. We define $\boldsymbol{d}$ as a vector of vectorized pixel difference $d(\boldsymbol{x})$ in a patch, as shown in figure 4.6.

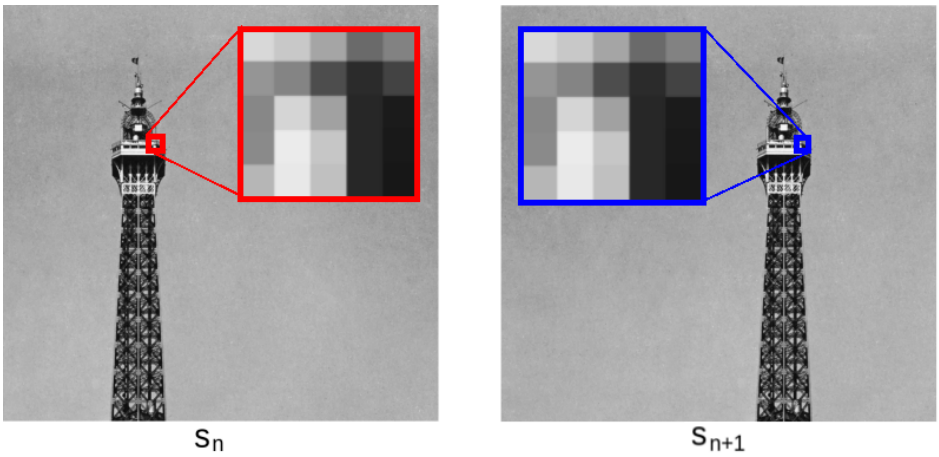

Figure 4.6: The illustration of matching patches in s_n (marked in red) and s_{n+1} (marked in blue).

$$\boldsymbol{d} \stackrel{def}{=} (d_1, d_2, ..., d_N) \tag{4.19}$$

In order to get an idea of what a suitable test statistic to be applied on the random vector $\boldsymbol{d}$ might be, we look into the distribution of the image difference $\boldsymbol{d}$ under the

Gaussian assumption.

$$\begin{aligned} p(\boldsymbol{d} \mid H_0) &= \prod_{i=1}^{N} p(d_i \mid H_0) \\ &= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma_d^2}} \exp\left[-\frac{d_i^2}{2\sigma_d^2}\right] \\ &= \left(\frac{1}{\sqrt{2\pi\sigma_d^2}}\right) \prod_{i=1}^{N} \exp\left[-\frac{d_i^2}{2\sigma_d^2}\right] \\ &= \left(\frac{1}{\sqrt{2\pi\sigma_d^2}}\right) \exp\left[-\sum_{i=1}^{N} \frac{d_i^2}{2\sigma_d^2}\right] \\ &= K \cdot \exp\left[-\sum_{i=1}^{N} \frac{d_i^2}{2\sigma_d^2}\right] \\ &= K \cdot \exp\left[-\frac{1}{2\sigma_d^2} \sum_{i=1}^{N} d_i^2\right] \end{aligned} \tag{4.20}$$

I define q as the sum of squared difference (SSD) between the two matched patches.

$$q \stackrel{def}{=} \sum_{i=1}^{N} d_i^2 \tag{4.21}$$

.

We see that the joint pdf $p(\boldsymbol{d} \mid H_0)$ is a function of the entity q. By knowing the value of q, we can disregard the individual data values d_i. (see equation 4.20), which implies that the SSD is a *sufficient statistic*[2]. As soon as we know the SSD, for testing the null hypothesis under the assumption that the image difference $\boldsymbol{d}$ is distributed according to a Gaussian distribution, we do not need any longer the individual values of the N random variables from which the SSD is computed.

Now we investigate the distribution of the chosen test statistic q under the null hypothesis. The expected value of q is given by,

$$\mathsf{E}[q] = \mathsf{E}\left[\sum_{i=1}^{N} d_i^2\right] = \sum_{i=1}^{N} \mathsf{E}\left[d_i^2\right] = \sum_{i=1}^{N} (\mathsf{E}[d_i])^2 + \mathsf{Var}[d_i] = \sum_{i=1}^{N} \mathsf{Var}[d_i] = N\,\sigma_d^2 \tag{4.22}$$

Next, we define a test statistic s as the normalized SSD by the variances σ_d^2, which appears in equation 4.20,

$$s \stackrel{def}{=} \sum_{i=1}^{N} \frac{d_i^2}{\sigma_d^2} = \frac{q}{\sigma_d^2} \tag{4.23}$$

This test statistic is well chosen for the Gaussian case, but useful (though suboptimal) also for other realistic distributions of the difference noise.

[2]I refer to (Papoulis 1990, p. 312) for the definition of sufficient statistics.

If we assume the pixel noise to be Gaussian, and thus the pixel difference under the H_0 hypothesis also to be Gaussian, then the SSD is the sum of squares of *i.i.d.* Gaussian random variables.

In addition, the test statistic s is a sum of squares of random variables which all of them have unit variance. By definition, this leads to a χ^2 distribution with N degrees of freedom.

4.1.3.1 Analysis of matching patches with Chi-Square distribution

Since we chose a normalized[3] SSD as the test statistic s (see equation (4.23)), s also has a χ^2 distribution if the pixel gray value difference is Gaussian distributed. If we have the distribution of the SSDs from experiments, and then we observe a certain value of a SSD for a regarded patch pair, we can look this value in the known distribution of SSDs. We can set up now a statistical test, similar to the procedure presented in section 4.1.2.1. It leads to the determination of a threshold τ_p for the p-value $p_s(\xi)$ (see figure 4.7),

$$p_s(\xi) \underset{H_1}{\overset{H_0}{\gtrless}} \tau_p \tag{4.24}$$

However, it is not a good idea and sub-optimal to convert each test statistic into a p-value (which involves going over the χ^2 distribution) and setting a threshold on the p-value (even though this strange approach is sometimes found in the literature). This approach might be acceptable only for one experiment, or few, not for testing a high number of match hypothesis. Hence, the analysis on the χ^2 distribution happens offline and only once.

As figure 4.7 clearly shows, the threshold τ_p results from the requirement that the false alarm rate is α. This results in a second threshold τ_ξ in the horizontal axis, which defines the border of the regions R_0 and R_1.

$$\int_{\tau_\xi}^{\infty} p_s(\xi \mid H_0)\, d\xi \overset{!}{=} \alpha \tag{4.25}$$

The test statistic s is checked against τ_ξ to determine whether it belongs to the regions R_0 or R_1 which leads to its hypothesis acceptance or rejection.

The SSD threshold τ_q is obtained using the relation between the SSD q and the test statistic s (see equation 4.23),

$$\tau_q = \sigma_d^2 \cdot \tau_\xi = 2\sigma_n^2 \cdot \tau_\xi \tag{4.26}$$

where σ_n^2 is the variance of the image noise.

[3]Precisely, not the SSD is normalized, but the individual elements from which the SSD is built are normalized.

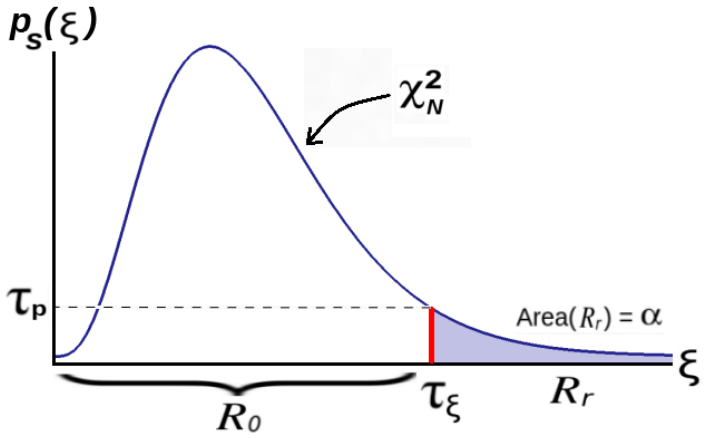

Figure 4.7: An example of setting a threshold of the test statistic in a χ^2 distribution with N degree of freedom.

4.1.4 Dependency of the SSD threshold value on the noise variance and the window size

Let us assume that we have empirically found a good value for the SSD threshold τ_q based on α level of significance for a given total pixel of the keypoint patch N_1, and a given noise variance of the images σ_1^2. Now let us assume that the window size is changed to N_2. The new value for the SSD threshold, denoted as τ_q', can be computed based on $\chi^2_{N_2}$ distribution due to the change of the number of pixels.

As shown in equation 4.25, we need to find the new threshold value τ_ξ' which corresponds to α level of significance on $p_s(\xi \mid H_0)$ which is χ^2 distribution of N_2 degrees of freedom.

$$\int_{\tau_\xi'}^{\infty} p_s(\xi \mid H_0)\, d\xi \stackrel{!}{=} \alpha \tag{4.27}$$

According to equation 4.26, it follows that

$$\frac{\tau_q'}{\sigma_1^2} = \tau_\xi' \quad \rightarrow \quad \tau_q' = \sigma_1^2\, \tau_\xi' \tag{4.28}$$

Let us assume that we have a given window size, a given weight function, and a good SSD threshold value τ_q. Assume now that the noise variance of the images changes to $\sigma_2^2 = k \cdot \sigma_1^2$. The mask size and the weight function remain unchanged. The new SSD threshold value τ_q' can be computed by the following,

$$\frac{\tau_q}{\sigma_1^2} = \frac{\tau_q'}{\sigma_2^2} \quad \rightarrow \quad \tau_q' = \frac{\sigma_2^2}{\sigma_1^2}\, \tau_q = k \cdot \tau_q \tag{4.29}$$

4.1.5 Estimation of the image noise variance

The noise variance of the image is directly related to the threshold for the accepted SSD. If we know the noise variance, we can then use this information to set up the

SSD threshold. Vice versa, as mentioned earlier, the noise variance of the image can empirically be estimated if we have many accepted SSDs from matching experiments, which is the case for PbT.

Let the accepted SSDs from matching experiments be q_i for $i = 1, 2, .., M$ and the mean accepted SSD is $\bar{q}$,

$$\bar{q} = \frac{\sum_{i=1}^{M} q_i}{M} \tag{4.30}$$

Based on equations 4.9 and 4.22, the expected value of the SSD is equal to,

$$\mathsf{E}[q] = N\,\sigma_d^2 = 2\,N\,\sigma_n^2 \tag{4.31}$$

where N is the number of pixels in the patch matching.

It follows that the image noise variance σ_n^2 can be empirically estimated by,

$$\sigma_n^2 = \frac{\bar{q}}{2\,N} \tag{4.32}$$

In the case when the accepted SSD information is not available from matching experiments, the noise variance of the image should be estimated using some methods of image quality measurement. If we assume that the image has a zero mean Gaussian noise, we can employ a fast noise variance estimation as proposed in (Immerkaer 1996).

4.2 Statistics of differential matching

Now I move from the discrete block matching to explain the differential matching. I assume that the pixel values are now samples from a continuous 2D signal. I begin by introducing the structure tensor of a local image patch.

4.2.1 The structure tensor of a two-dimensional image patch

Let $y(\boldsymbol{x})$ be a patch of N image pixels. The spatial gradient vectors $\boldsymbol{g}$ of y are assumed[4] to be known.

$$\boldsymbol{g}_i \stackrel{def}{=} \left(\frac{\partial y}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}_i} \right) = (g_1(\boldsymbol{x}_i),\ g_2(\boldsymbol{x}_i))^T = \left(\frac{\partial y}{\partial x_1}, \frac{\partial y}{\partial x_2} \right)^T \tag{4.33}$$

We define matrix $\mathbf{G}$,

$$\mathbf{G} \stackrel{def}{=} \begin{pmatrix} \boldsymbol{g}_1, & \boldsymbol{g}_2, & \ldots & \boldsymbol{g}_N \end{pmatrix} \tag{4.34}$$

[4]This assumption is a general source of loss of realism in the model, as the gradients need to be estimated from discrete data.

We define $\mathbf{S}$ as the structure tensor of the patch y with a weighting window w, as presented in (Jähne 1993, p. 147).

$$\mathbf{S} = \begin{pmatrix} s_{11} & s_{12} \\ s_{12} & s_{22} \end{pmatrix} \stackrel{def}{=} \begin{pmatrix} \sum_i w(\boldsymbol{x}_i) \cdot g_1^2(\boldsymbol{x}_i) & \sum_i w(\boldsymbol{x}_i) \cdot g_1(\boldsymbol{x}_i) g_2(\boldsymbol{x}_i) \\ \sum_i w(\boldsymbol{x}_i) \cdot g_1(\boldsymbol{x}_i) g_2(\boldsymbol{x}_i) & \sum_i w(\boldsymbol{x}_i) \cdot g_2^2(\boldsymbol{x}_i) \end{pmatrix} \tag{4.35}$$

Thus, we also have

$$\mathbf{S} = \mathbf{G} \cdot \mathsf{diag}\,\{w\} \cdot \mathbf{G}^T \tag{4.36}$$

The structure tensor will appear and play important role in the next section about the KLT matching.

4.2.2 Basic KLT matching: Optical flow with brightness constancy assumption

We regard now the basic Kanade-Lucas-Tomasi (KLT) keypoint matching. Now let z and y be two images with coordinates given by $\boldsymbol{x}$. We fix a set of points $\{\boldsymbol{x}_i\}$ in image $z(\boldsymbol{x})$ and try to determine *one* vector $\hat{\boldsymbol{v}}$ such that the sum of squared difference between $z(\boldsymbol{x}_i)$ and $y(\boldsymbol{x}_i + \hat{\boldsymbol{v}})$ is minimum. This loss function to be minimized is denoted as Q. Since this problem cannot be solved directly in closed form, it has to be performed iteratively, using local Taylor approximations. This chain of iterative steps to find a 'good' estimate $\hat{\boldsymbol{v}}$ is equivalent to finding a series of displacements $\boldsymbol{v}_j, j = 1, 2, \ldots$ that converge towards that displacement $\hat{\boldsymbol{v}}$ such that

$$Q \stackrel{def}{=} \sum_i \left(z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i + \hat{\boldsymbol{v}})\right)^2 \quad \longrightarrow \min \tag{4.37}$$

We derive the particular observation equation of optical flow, which is a form of the brightness constancy equation. The terms that appear in this equation are the spatial derivatives in x and y direction at a regarded point, and pixel differences between the regarded images. Like this is done in the overwhelming majority of papers, I assume that the derivatives are known error-free, but the pixel values contain zero-mean i.i.d. noise. This is an approximation of the true relations, and the analysis of optical flow considering also noisy derivative values leads into the area of 'total least squares' estimation (Mester and Mühlich 2001) (Mester 2014). It is common, and the core idea of most differential motion estimation methods, to tackle this minimization problem by the Gauss-Newton approach. For that purpose, a first order Taylor approximation of the difference term $z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i + \hat{\boldsymbol{v}})$ is computed:

$$\begin{aligned} z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i + \hat{\boldsymbol{v}}) &= z(\boldsymbol{x}_i) - \left(y(\boldsymbol{x}_i) + \hat{\boldsymbol{v}}^T \cdot \left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} \right) + \text{h.o.t.} && (4.38) \\ &= (z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)) - \hat{\boldsymbol{v}}^T \cdot \left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} + \text{h.o.t.} && (4.39) \end{aligned}$$

The fundamental assumption in KLT matching is that the values of z and y and the gradients can be computed even for discrete image data. Since the difference term is

linearized while the targeted loss function is the sum of *squared* differences, this is an approximation to the 'true' optimization of the loss function Q defined in (4.37).

$$Q \cong \sum_i \left((z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)) - \hat{\boldsymbol{v}}^T \cdot \left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} \right)^2 \quad \longrightarrow \min \tag{4.40}$$

We obtain:

$$\begin{aligned} Q \;\cong\; & \sum_i \Big((z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i))^T \cdot (z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)) \Big) \\ & - \sum_i \left(2\,(z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)) \cdot \hat{\boldsymbol{v}}^T \cdot \left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} \right) \\ & + \sum_i \left(\hat{\boldsymbol{v}}^T \cdot \left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} \cdot \left(\left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} \right)^T \cdot \hat{\boldsymbol{v}} \right) \quad \longrightarrow \min \end{aligned} \tag{4.41}$$

This can be written as

$$\begin{aligned} Q \;\cong\; & \sum_i (z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i))^2 \\ & - \sum_i \left(2\,(z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)) \cdot \left(\left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} \right)^T \right) \cdot \hat{\boldsymbol{v}} \\ & + \sum_i \left(\hat{\boldsymbol{v}}^T \cdot \left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} \cdot \left(\left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} \right)^T \cdot \hat{\boldsymbol{v}} \right) \quad \longrightarrow \min \end{aligned} \tag{4.42}$$

or even shorter as:

$$\begin{aligned} Q \;\cong\; & \sum_i (z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i))^2 \\ & - 2 \left(\sum_i (z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)) \cdot \left(\left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} \right)^T \right) \cdot \hat{\boldsymbol{v}} \\ & + \hat{\boldsymbol{v}}^T \cdot \sum_i \left(\left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} \cdot \left(\left.\frac{\partial y}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_i} \right)^T \right) \cdot \hat{\boldsymbol{v}} \quad \longrightarrow \min \end{aligned} \tag{4.43}$$

Thus we obtain

$$\begin{aligned} Q \;\cong\; & \sum_i (z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i))^2 \\ & - 2 \left(\sum_i (z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)) \cdot \boldsymbol{g}_i^T \right) \cdot \hat{\boldsymbol{v}} \\ & + \hat{\boldsymbol{v}}^T \cdot \sum_i \left(\boldsymbol{g}_i \cdot \boldsymbol{g}_i^T \right) \cdot \hat{\boldsymbol{v}} \quad \longrightarrow \min \end{aligned} \tag{4.44}$$

Hence, we can express the above equation as,

$$
\begin{aligned}
Q(\boldsymbol{v}) &= \boldsymbol{v}^T \cdot \mathbf{A} \cdot \boldsymbol{v} - 2\,\boldsymbol{b}^T \cdot \boldsymbol{v} + c && (4.45)\\
\text{with} \quad \mathbf{A} &= \sum_i \left(\boldsymbol{g}_i \cdot \boldsymbol{g}_i^T\right) = \mathbf{G}\,\mathbf{G}^T = \mathbf{S} && (4.46)\\
\boldsymbol{b} &= \sum_i \left(z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)\right) \cdot \boldsymbol{g}_i^T = \sum_i d_i \cdot \boldsymbol{g}_i^T = \mathbf{G} \cdot \boldsymbol{d} && (4.47)\\
c &= \sum_i \left(z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)\right)^2 && (4.48)
\end{aligned}
$$

where

$$d_i = d(\boldsymbol{x}_i) = z(\boldsymbol{x}_i) - y(\boldsymbol{x}_i) \tag{4.49}$$

The first derivative of the loss function Q is calculated to find displacement vector $\boldsymbol{v}$ which minimizes the loss function Q.

$$\frac{dQ}{d\boldsymbol{v}} = \left(\mathbf{A}^T + \mathbf{A}\right) \cdot \boldsymbol{v} - 2\boldsymbol{b} = \mathbf{0} \tag{4.50}$$

Since $\mathbf{A}$ is a symmetric matrix,

$$
\begin{aligned}
2\,\mathbf{A}\boldsymbol{v} - 2\,\boldsymbol{b} &= \mathbf{0} && (4.51)\\
\Rightarrow \quad \mathbf{A} \cdot \boldsymbol{v} &= \boldsymbol{b} && (4.52)
\end{aligned}
$$

This is the *normal equation for the optical flow problem.* Since matrix $\mathbf{A}$ is equal to $\mathbf{S}$ (see equation 4.46) and vector $\boldsymbol{b}$ is known, the problem to find vector $\boldsymbol{v}$ now becomes two equations with two unknowns problem which can be solved in a straightforward way.

Since $\mathbf{S}$ is a 2×2 matrix, this is no longer an overdetermined system. The solution depends whether $\mathbf{S}$ is invertible, which is assumed to be the case in my derivation below. A non-invertible $\mathbf{S}$ can occur for instance when the image is totally flat and structureless.

$$
\begin{aligned}
\hat{\boldsymbol{v}} &= \mathbf{S}^{-1} \cdot \boldsymbol{b} && (4.53)\\
&= \mathbf{S}^{-1} \cdot \mathbf{G} \cdot \boldsymbol{d} && (4.54)\\
&= (\mathbf{G} \cdot \mathbf{G}^T)^{-1} \cdot \mathbf{G} \cdot \boldsymbol{d} && (4.55)
\end{aligned}
$$

Now we compute the covariance of the motion error. This is a coarse approximation under the assumption that only the vector $\boldsymbol{d}$ is afflicted by pixel noise, whereas the structure tensor is not affected by pixel noise. This assumption is general practice in the motion estimation literature although it is actually not realistic. For a more detailed

analysis, I refer to (Mester 2014).

$$\begin{aligned}
\mathsf{Cov}\left[\hat{\boldsymbol{v}} - \boldsymbol{v}\right] &= \mathsf{Cov}\left[\mathbf{S}^{-1} \cdot \mathbf{G} \cdot \boldsymbol{d}\right] && (4.56)\\
&= (\mathbf{S}^{-1} \cdot \mathbf{G}) \cdot \mathsf{Cov}\left[\boldsymbol{d}\right] \cdot (\mathbf{S}^{-1} \cdot \mathbf{G})^T && (4.57)\\
&= (\mathbf{S}^{-1} \cdot \mathbf{G}) \cdot \mathsf{Cov}\left[\boldsymbol{d}\right] \cdot \mathbf{G}^T \cdot \mathbf{S}^{-T} && (4.58)\\
&= (\mathbf{S}^{-1} \cdot \mathbf{G}) \cdot \sigma_d^2 \cdot \mathbf{I}_N \cdot \mathbf{G}^T \cdot \mathbf{S}^{-T} && (4.59)\\
&= \sigma_d^2 \cdot \mathbf{S}^{-1} \cdot \mathbf{G} \cdot \mathbf{G}^T \cdot \mathbf{S}^{-T} && (4.60)\\
&= \sigma_d^2 \cdot \mathbf{S}^{-1}\mathbf{S} \cdot \mathbf{S}^{-T} && (4.61)\\
&= \sigma_d^2 \cdot \mathbf{S}^{-T} && (4.62)\\
&= \sigma_d^2 \cdot \mathbf{S}^{-1} && (4.63)\\
&= 2\,\sigma_n^2 \cdot \mathbf{S}^{-1} && (4.64)
\end{aligned}$$

where σ_n^2 is the variance of the pixel noise.

4.2.3 Characterizing the covariance matrix by its eigenvalues and its trace

Now we analyze the eigenvalues of the structure tensor $\mathbf{S}$, denoted by λ_1 and λ_2. We assume that none of them is really equal to zero, but the smallest one may be very small. It follows that the resulting covariance matrix $\mathsf{Cov}\left[\hat{\boldsymbol{v}} - \boldsymbol{v}\right]$ (see equation 4.64) has eigenvalues of $\dfrac{2\sigma_n^2}{\lambda_1}$ and $\dfrac{2\sigma_n^2}{\lambda_2}$.

The mean squared motion vector error is given by the trace of the covariance matrix. The eigenvalues of the covariance matrix serve as the precision measurement for the matching result. Specifically, we look into the sum of the eigenvalues of the covariance matrix which is equivalent to the trace of the covariance matrix.

$$\mathsf{E}\left[\sum_i (\hat{\boldsymbol{v}}_i - \boldsymbol{v}_i)^2\right] = \mathrm{Tr}(\mathsf{Cov}\left[\hat{\boldsymbol{v}}\right]) = 2\sigma_n^2\left(\frac{1}{\lambda_1} + \frac{1}{\lambda_2}\right) \qquad (4.65)$$

Since in PMO we use both corner points and edgel points, the smaller eigenvalue λ_2 can be very small close to zero. Furthermore, σ_n is a constant. Thus, the matching precision is graded based only on the bigger eigenvalue λ_1. Hence, we can define a threshold τ_λ to make the final decision on the matching precision.

$$\mathrm{Tr}(\mathsf{Cov}\left[\hat{\boldsymbol{v}} - \boldsymbol{v}\right]) \underset{accept}{\overset{reject}{\gtrless}} \tau_E \quad \rightarrow \quad \lambda_1 \underset{reject}{\overset{accept}{\gtrless}} \tau_\lambda \qquad (4.66)$$

If we have a set of matching results, we can look into the statistics of verified matchings in order to set τ_λ.

4.3 Matching implementation in PMO

The proposed PMO framework employs patch matching, with a patch size of 15×15 pixels. In the context of the Chi-Square distribution discussed in section 4.1.3.1, it corresponds to 225 degrees of freedom. Each patch is subjected to a 2D Gaussian ($\sigma = 7.5$) weighting mask.

The regarded images in PMO are 8-bit grayscale images, thus having gray values in the range of 0 and 255. Threshold values mentioned in this section are based on this 8-bit grayscale image assumption.

From the matching experiments, a set of accepted matchings was obtained along with the corresponding SSD values. As presented in equation 4.26 and equation 4.32, the image noise variance $\hat{\sigma}_n^2$ is empirically estimated at 5.54. For the significance level α of the hypothesis test at 0.05, it corresponds to an SSD threshold $\hat{\tau}_q$ of 16004.

The similar procedure was carried out for the analysis of the eigenvalues of the structure tensor. The threshold τ_λ (see equation 4.66) was empirically estimated at 25.

4.4 Concluding remarks

This chapter has discussed the statistical analysis of keypoint matching. First, the statistics of block matching was presented. The cases of matching single pixel as well as matching patch of multiple pixels were discussed. The analysis of the matching SSD has also been presented, and later utilized in section 5.5 as implemented in the proposed PMO framework.

The second part of this chapter is about the statistics of differential matching. The loss function representing the photometric error was introduced. The displacement vector which minimizes the loss function is obtained via iterative process involving the structure tensor of the image. The analysis on the quality of the estimated displacement vector has also been presented. The differential matching will be employed and discussed further in chapter 5 as implemented in the PMO framework.

Chapter 5

Epipolar-constrained matching and motion priors

5.1 Introduction to epipolar matching

The typical situation in many monocular SLAM scenarios is that a large percentage of the image area can be associated with a static scene and the other moving objects are only covering a small percentage of the image area, typically in the order of 5-20%. This means that there is a dominating epipolar relation, which can be used to impose a constraint on the motion field from 2 degrees of freedom per pixel in an unconstrained (and highly unrealistic) *general motion* scenario to 1 degree of freedom (= depth) in 80-95% in a typical SLAM scenario. If this epipolar relation would be known in beforehand, the risk of incurring false matches ('outliers') could be very significantly reduced.

Furthermore, in an automotive scenario (different from the hand-held camera case) the relative ego-motion from image to image varies only very smoothly and can be very well predicted. This means that it makes sense to predict the ego-motion when the next frame comes in and estimate the motion using a *soft* epipolar constraint. These facts have been observed before in (Yamaguchi, McAllester and Urtasun 2013) (see section 5.2), but we exploit them in a different way.

The imposed epipolar constraint gives meaningful information to a tracker on where to search for matches, as illustrated in figure 5.1. A differential tracker, such as the Kanade-Lucas-Tomasi (KLT) tracker (Lucas et al. 1981, Tomasi and Kanade 1991, Shi and Tomasi 1994) can be made more efficient by following the epipolar constraint. Such extensions of the KLT were proposed in (Ochoa and Belongie 2006, Trummer, Munkelt and Denzler 2009b, Piccini et al. 2014) where epipolar geometry is exploited on the classical KLT optimization to enable tracking not only of corner points but also edge pixels (edgels) (Birchfield and Pundlik 2008, Piccini et al. 2014). We build on the work of Piccini et al. (Piccini et al. 2014) and address an open problem from that paper, namely, how to get the tracking started. As soon as we have matched a point once

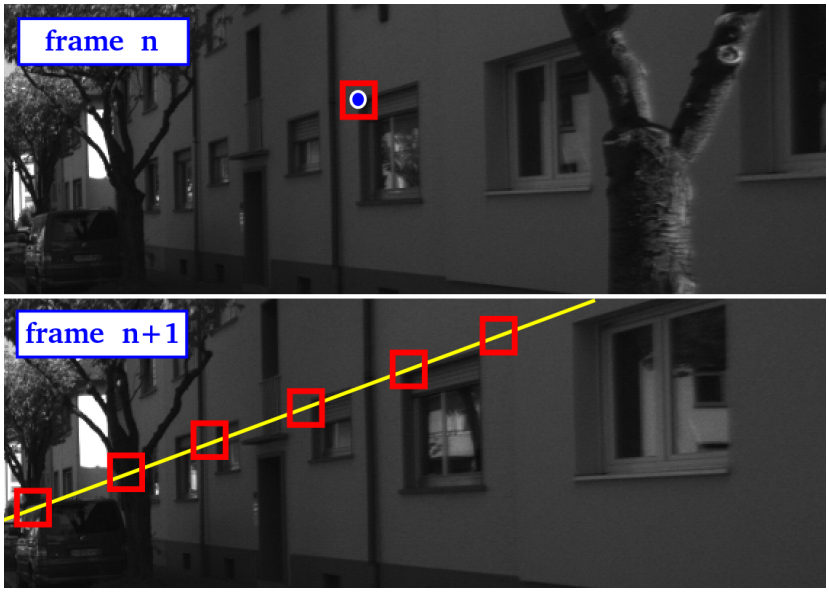

Figure 5.1: The illustration of the epipolar matching principle. A selected keypoint (a corner of a window) in frame n, represented by a patch (red), is matched to frame $n+1$. Because of the epipolar geometry, the correct match has to be on the epipolar line (yellow), hence decreasing the search space into only a one-dimensional search along the epipolar line.

and the epipolar relation is known, we have an indication of its 3D position and can propagate this information, making the transition from *matching* to *tracking*.

However, a differential tracker, such as KLT, has a limited convergence range, requiring the initial guess of the match being within a certain range around the true correspondence (Sutton, Orteu and Schreier 2009, p.84). This means that for a successful matching, in particular in the case of fast 2D motion, we need initialization information. An initialization for the KLT tracker usually comes as a displacement vector which only applies for single pixels and not for patches. Inside a patch, there can be multiple motion vectors, in the extreme case, one individual motion vector for each pixel. For all patches of pixels there will always be a *set* of true motion vectors. As long as the set of motion vectors obey a parametric model (affine motion model, etc.), this can be expressed by a set of parameters, but in case of multiple motions in a patch, this is not sufficient.

The fact that the pixel-wise shift (or motion) between the two images is far from uniform leads to our idea to compute the distribution of motion between any two images or between two patches, which we denote as a *motion prior*. As we will see later, this can be done by an *area based method*, not requiring individual keypoints to be detected and tracked. In our approach, we describe motion priors, as *motion distributions* parametrized by a mean vector and a 2×2 covariance matrix.

In this chapter, first, we propose to compute motion priors to provide the distribution of motion between two images. Then, by utilizing the information of motion priors, i.e. having better information about the motion distribution between two images, we aim

to optimally initialize the KLT tracker for matching purpose.

PMO generates motion priors by using a novel variant of the phase correlation method, and uses these priors in an epipolar-geometry guided matching scheme. Phase correlation, which is a classic method for image alignment and image registration, is employed to estimate the relative displacement between two image patches from two consecutive frames of the video stream. Here, we apply a recently proposed enhanced variant of phase correlation (Ochs et al. 2015). The particular advantage of using that method is that it outputs not just a displacement vector, but distribution of vectors, or moments of a distribution. Therefore this method is able to consider multiple motions inside a patch and thus it can handle the important issue of motion discontinuities which often occurs for example in the case of occlusion.

The work presented in this chapter provides several contributions:

- The motion of a patch is described by a motion distribution, which can represent the uncertainty introduced by multiple motions.
- Given a set of keypoints, we calculate the motion prior for each keypoint to provide an accurate initialization condition for the tracker.
- We run KLT-style differential matching in integer-grid steps first before going for sub-pixel precision on the last step to prevent expensive computation.
- The epipolar search is restricted on a segment of the epipolar line corresponding to positive depth rather than a full line search.

5.2 Related work on epipolar matching

In an automotive environment where a vehicle exhibits only moderate variations in pitch and roll, and only slowly changing translation speed and direction, further constrained by the known couplings between rotational and translational motion, frame-to-frame motion (3D pose change) can be predicted very well (Bradler et al. 2015). Thus the epipolar structure of all motion vectors which belong to points on the static environment can be predicted and exploited for stabilizing the computation of point matches. The epipolar-guided matching is powerful in a way that the search space of the keypoint matching decreases from two-dimensional search into 1-dimension search along the epipolar line.

Trummer et al, in a series of papers (Trummer, Denzler and Munkelt 2008, Trummer et al. 2009b, Trummer, Munkelt and Denzler 2009a, Trummer, Munkelt and Denzler 2010), proposed a framework to extend the Lucas-Kanade implementation by Baker et al (Baker and Matthews 2004). The first paper (Trummer et al. 2008) proposes KLT tracking with epipolar constraint by modifying the warp function presented in (Baker and Matthews 2004). It briefly introduces soft epipolar constraint by having two weighting parameters for displacement vector along epiline and perpendicular to it respectively. The second paper (Trummer et al. 2009b) extended the previous paper

with an explicit solution of soft constraint weighting parameters. It incorporates the weighting parameters inside the optimization process, thus alternately updates the warping parameters and epipolar weight in the iteration. The third paper (Trummer et al. 2009a) exploited the depth information by triangulating keypoint matches. It checks whether 3D position of a keypoint is consistent throughout the frames. The last paper (Trummer et al. 2010) utilized Guided-KLT (GKLT) to propose a method to automatically navigate the camera to follow a set of tracked keypoint.

Yamaguchi et al (Yamaguchi et al. 2013) presented an epipolar-constrained matching inspired by semi global block matching (SGM) by imposing the epipolar constraint. The approach employs bottom-up grouping algorithm for joint segmentation and flow estimation. It also adopts a slanted plane stereo model. When this method was firstly introduced, it achieved much lower error than the existing best epipolar flow algorithm in the KITTI flow benchmark. Mohamed et al (Mohamed, Mirabdollah and Mertsching 2015) also proposed pyramidal differential optical flow estimation under the monocular epipolar line constraint.

5.3 Generation of new keypoints

New keypoints in the image are detected and chosen in the PMO framework based on the good-feature-to-track (GFTT) score to find corner points as proposed by (Shi and Tomasi 1994). On top of that, I also choose edge-pixel (edgel) points based on a approach from (Piccini et al. 2014) which is an extended version of the GFTT method. Thus the keypoints in PMO are a combination of corner points and edgel points.

The GFTT approach uses the structure tensor $\mathbf{S}$ at a given position $\boldsymbol{x} = (x, y)$ which has been discussed earlier in section 4.2.1:

$$\mathbf{S} = \begin{pmatrix} s_{11} & s_{12} \\ s_{12} & s_{22} \end{pmatrix} \overset{def}{=} \begin{pmatrix} \sum_i w(\boldsymbol{x}_i) \cdot g_1^2(\boldsymbol{x}_i) & \sum_i w(\boldsymbol{x}_i) \cdot g_1(\boldsymbol{x}_i) g_2(\boldsymbol{x}_i) \\ \sum_i w(\boldsymbol{x}_i) \cdot g_1(\boldsymbol{x}_i) g_2(\boldsymbol{x}_i) & \sum_i w(\boldsymbol{x}_i) \cdot g_2^2(\boldsymbol{x}_i) \end{pmatrix} \tag{5.1}$$

where w is a weighting window and gradients g_i are defined as

$$\boldsymbol{g}_i \overset{def}{=} \left(\frac{\partial y}{\partial \boldsymbol{x}} \bigg|_{\boldsymbol{x}_i} \right) = (g_1(\boldsymbol{x}_i),\ g_2(\boldsymbol{x}_i))^T = \left(\frac{\partial y}{\partial x_1}, \frac{\partial y}{\partial x_2} \right)^T \tag{5.2}$$

The weighting window w in PMO is set as a 2D Gaussian ($\sigma = 7.5$) weighting mask. The image gradient is estimated by employing a Scharr-operator (Scharr 2000) G_x and G_y for the derivative in x and y directions respectively.

$$G_x = \frac{1}{32} \begin{pmatrix} 3 & 0 & -3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{pmatrix} \quad ; \quad G_y = \frac{1}{32} \begin{pmatrix} 3 & 10 & 3 \\ 0 & 0 & 0 \\ -3 & -10 & -3 \end{pmatrix} \tag{5.3}$$

The structure tensor $\mathbf{S}(\boldsymbol{x}) \in \mathbb{R}^{2\times 2}$ can be decomposed into the eigenvalues (λ_1, λ_2) and eigenvectors $(\boldsymbol{v}_1, \boldsymbol{v}_2)$ elements:

$$\mathbf{S}(\boldsymbol{x}) = \lambda_1 \boldsymbol{v}_1 \boldsymbol{v}_1^T + \lambda_2 \boldsymbol{v}_2 \boldsymbol{v}_2^T, \tag{5.4}$$

where $\lambda_1 \geq \lambda_2 \geq 0$ and $\boldsymbol{v}_1 \perp \boldsymbol{v}_2$.

I denote τ_λ as the threshold of the biggest eigenvalue and $\tau_{r\lambda}$ as the ratio between the eigenvalues (λ_2/λ_1). By analyzing the eigenvalues of the structure tensor at each pixel position, we can then classify each pixel into one of the three cases:

- $\lambda_1 > \tau_\lambda$ and $\dfrac{\lambda_2}{\lambda_1} > \tau_{r\lambda}$:
  The neighborhood does not show a dominant direction of the gradients and therefore it can be described as a two-dimensional/corner-like structure.
- $\lambda_1 > \tau_\lambda$ and $\dfrac{\lambda_2}{\lambda_1} \leq \tau_{r\lambda}$:
  All gradients in the observed neighborhood are approximately aligned and point into the same direction. The neighborhood can be described as a one-dimensional/edge-like structure.
- $\lambda_1 \leq \tau_\lambda$:
  The gradients in the neighborhood are of negligible magnitude. No noteworthy structure is present.

5.3.1 Check the epipolar compatibility of each keypoint

The implementation of PMO relies on the epipolar-constrained keypoint tracking. Since the edgels are also considered for keypoint tracking, it is important that there is sufficient angle between the epipolar line and the line structure corresponding to the edgel. If the angle is too small, the aperture problem can arise which leads to an uncertain keypoint matching.

Figure 5.2: The illustration of the angle between the epipolar line and the line structure corresponding to an edge point.

The direction of the edge corresponding to an edgel structure is given by the eigenvector corresponding to the smallest eigenvalue, denoted by $\boldsymbol{v}_2$. Let $\boldsymbol{l}$ be the vector representing the direction of the corresponding epipolar line. The angle difference between $\boldsymbol{v}_2$ and $\boldsymbol{l}$, as illustrated in figure 5.2 denoted as θ_e, can be computed as follows:

$$\theta_e = \arccos\left(\frac{\boldsymbol{l}^T \boldsymbol{v}_2}{\|\boldsymbol{l}\| \cdot \|\boldsymbol{v}_2\|}\right) \tag{5.5}$$

The minimum value for θ_e to accept the keypoints is set at $\tau_{\theta e}$.

5.3.2 Bucketing for uniform keypoint distribution

A bucketing mechanism is employed to control the placement of the chosen keypoints. The keypoints should not cluster too densely since a dense cluster of keypoints at almost the same position does not provide much more information than just a few keypoints reasonably placed in the same area. On the other hand, the keypoints should not be positioned too sparsely since in such a sparsely covered region I will not obtain sufficient information to be exploited.

The bucketing scheme works as follows:

1. The image is subdivided in a grid of square *cells* of dimension $c \times c$.
2. Let $f(\boldsymbol{x})$ be a scalar function with $f(\boldsymbol{x}) \geq 0$ which expresses the suitability of pixel position $\boldsymbol{x}$ to serve as a keypoint, i.e. the eigenvalue λ_1 as defined in equation 5.4. Compute $f(\boldsymbol{x})$ for the complete image.
3. Find the maximum of $f(\boldsymbol{x})$ for each cell and store the corresponding location $\boldsymbol{x}$ as a candidate. If the maximum value of the cell is lower than f_{min}, the candidate is rejected.
4. For each candidate position the distance with respect to the 8 neighbouring cells candidate is checked, if the distance is smaller than $c/2$, the candidate with the lowest value will be rejected.

5.4 Epipolar KLT keypoint matching

I regard now a keypoint matching method based on the KLT optimization with an epipolar constraint. Based on the classical KLT tracker method (see section 4.2.2), the matching optimization problem can be written as

$$Q(\boldsymbol{v}) = \boldsymbol{v}^T \cdot \mathbf{A} \cdot \boldsymbol{v} + \boldsymbol{b}^T \cdot \boldsymbol{v} + c \tag{5.6}$$

where $\boldsymbol{v} = (v_x, v_y)^T$ is the sought displacement vector, $\mathbf{A}$ is a symmetric 2×2 matrix built from the outer product of the gradient vectors in a patch, $\boldsymbol{b}$ is a 2×1 vector, and c is a scalar. The explicit formulas for the elements $\mathbf{A}$, $\boldsymbol{b}$, c have been presented earlier in section 4.2.2.

For the proposed epipolar constraint tracker, I add the epipolar constraint to the quadratic optimization function loss function Eq. 5.6. This problem is solved like in Eq. 5.12 with a Lagrange multiplier α_1, thus yielding Eq. 5.7 which leads to a closed form solution of $\boldsymbol{v}$.

$$\begin{pmatrix} \mathbf{A} & \mathbf{F}' \cdot \boldsymbol{x}_h \\ (\mathbf{F}' \cdot \boldsymbol{x}_h)^T & 0 \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{v} \\ \alpha_1 \end{pmatrix} = \begin{pmatrix} -\boldsymbol{b} \\ -(\boldsymbol{x}_h^T \cdot \mathbf{F} \cdot \boldsymbol{x}_h) \end{pmatrix} \tag{5.7}$$

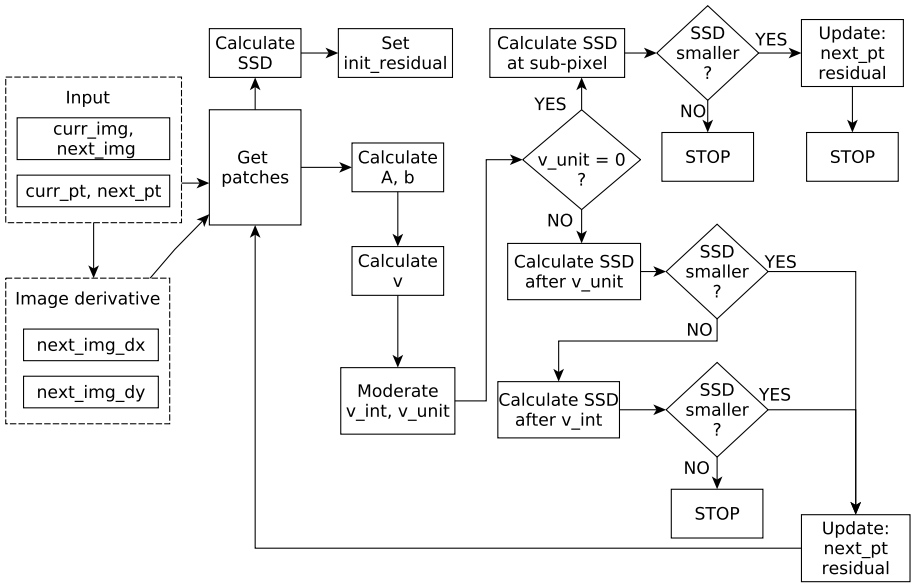

Figure 5.3: Block diagram for epipolar KLT matching

I propose a new approach to iteratively optimize the constrained KLT problem. Due to the small range where the differential optimization problem is valid, we moderate the real value displacement vector ($\boldsymbol{v}$) from the KLT constrained optimization iteration to a single integer pixel step ($\boldsymbol{v}_{unit}$) vertically and/or horizontally until it converges. After the integer-valued displacement vectors are obtained, the final real-valued sub-pixel refinement is used. I also take into account the rounded displacement vector ($\boldsymbol{v}_{int}$) whenever $\boldsymbol{v}_{unit}$ leads to a patch with higher MSD. The complete procedure of our epipolar KLT matching is illustrated in Figure 5.3.

5.5 Keypoint matching based on motion prior

In automotive scenarios, the tracking of keypoints is often difficult since long and even ultra-long displacements may occur due to the high speed and rotations rates of the vehicle. Typical multi-resolution approaches help only partially, since reducing the resolution mixes different motions and makes small moving elements disappear (Brox, Bregler and Malik 2009). Phase correlation can systematically deal with multiple motion, as shown in (Ochs et al. 2015). We consider this particular implementation of the phase correlation and show how to exploit its advantageous features.

The flowchart of the proposed method is depicted in Figure 5.4. We estimate the frame-to-frame motion distribution on a subdivision of each incoming gray scale image into cells using pyramidal phase correlation as described in Section 5.5.1. These per-cell motion distributions are subsequently employed to find the keypoint correspondences between the two frames by imposing an epipolar constraint.

The incremental movement of a keypoint after motion prior estimation and during epipolar KLT matching are shown in Figure 5.4(c). From a starting position A, the

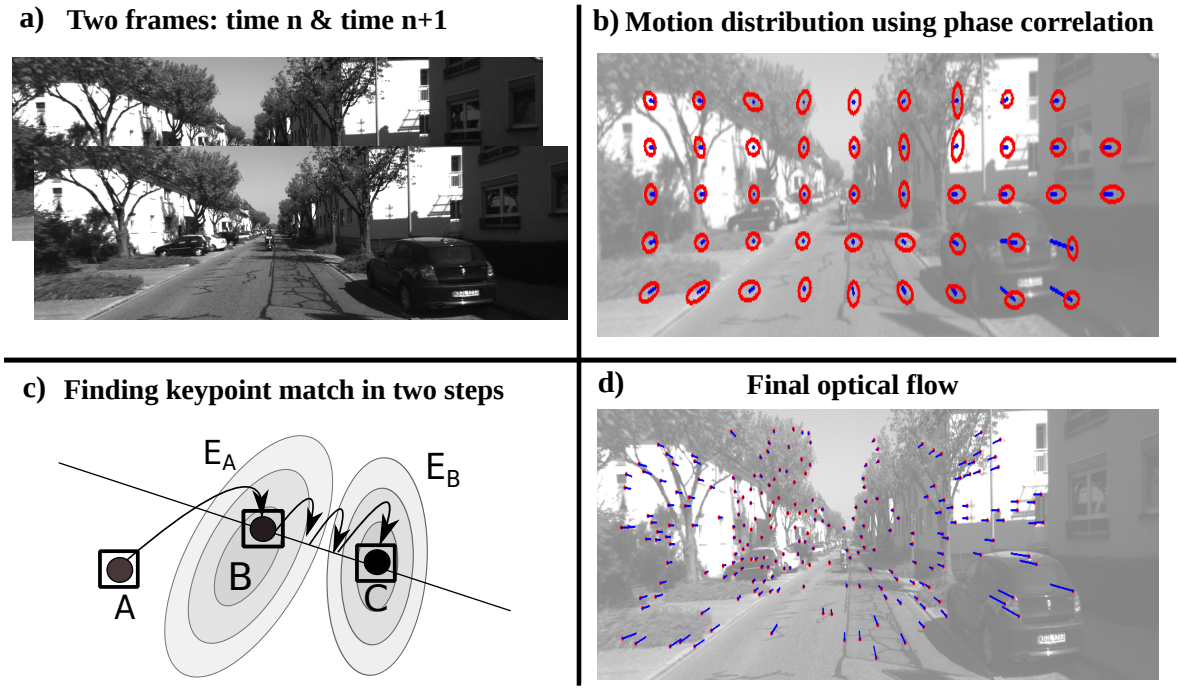

Figure 5.4: Overall view: (a) a pair of frames as input, (b) motion distribution using pyramidal phase correlation, (c) keypoints matching in two steps, (d) final optical flow

keypoint moves to point B through a matching initialization method. Point B is the solution of the matching initialization problem using optimization method between the motion prior covariance (ellipse E_A) with the epipolar line constraint. The epipolar KLT matching method subsequently uses matching initialization result and it will iteratively bring the keypoint from point B to point C which is the optimized position of the *mean squared difference* (MSD) ellipse E_B with the epipolar line constraint as discussed in Section 5.5.2.3.

5.5.1 Computing the motion prior distribution using phase correlation

The motion prior distribution is computed as proposed in (Fanani et al. 2015). We use a two-level pyramid for each image by having the original size image, I_{ORI}, and a sub-sampled image (I_{SUB}) which has been down-sampled by a factor of 2.

We apply the phase correlation method to analyze the distribution of spatial shift between two consecutive frames in a sequence of images, namely $I(n)$ and $I(n+1)$, as proposed in (Ochs et al. 2015). Prior to the phase correlation step, we calculate the average global 2D displacement between $I(n)$ and $I(n+1)$ using the method as proposed in (Barnada et al. 2015) as a preliminary shift information. We use two image patches respectively from $I(n)$ and $I(n+1)$ referred as *phase correlation (PhC)* cells as the input of the phase correlation method. In case of using the KITTI image sequence (376×1241 pixel resolution), the size of all the cells is 64×128 pixels, since we have

mostly horizontal motions in the scene. The cells are distributed partially overlapping each other covering all the image area as can be observed in Figure 5.5.

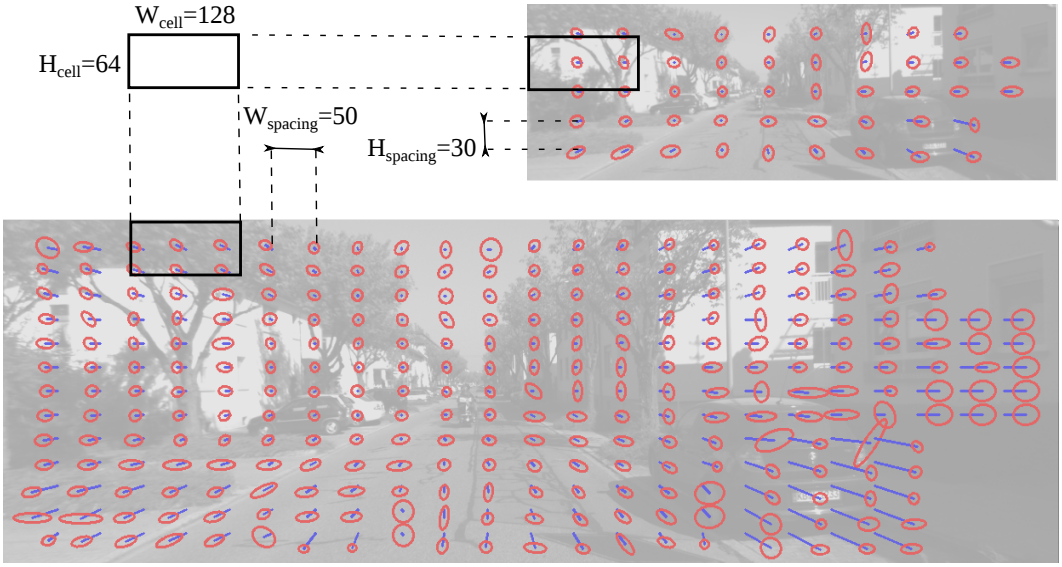

Figure 5.5: The size of the phase correlation cells is the same for both images as well as the spacing between cells. Due to the initial shift propagation, the spacing (or location) of the cells might vary from the top layer to the bottom layer.

The output of phase correlation for I_{SUB} is propagated to the bottom layer of the pyramid (original size, I_{ORI}) obtaining the final shift after adding all the inherited shifts.

5.5.1.1 Validity checks

We employ the latest improvement of the phase correlation method using multiple checks as proposed in (Ochs et al. 2015). The output of each phase correlation step is a displacement array containing all the estimated shifts between the two image patches. There are several cases when phase correlation correctly detects and reports that it cannot provide any output, in particular when there is not enough structure inside phase correlation (PhC) cells, e.g. for bright sky and plain road surfaces.

The advantages of using PhC methods as proposed in (Ochs et al. 2015) are:

- It provides a distribution of motion (mean and covariance).
- It can detect multiple motions.
- It is insensitive to affine photometric transforms.
- The method as implemented according to (Ochs et al. 2015) has built-in tests that allow us to detect failure situations.

5.5.1.2 Uncertainty representation using covariance ellipse

The phase correlation method gives us the displacement array which could approximatively be regarded as a probability mass for each of the displacements. We can compute the mean displacement vector and the spatial uncertainty expressed by the covariance matrix of that distribution as proposed in (Mester and Hötter 1995).

5.5.2 Keypoints matching

The proposed method for determining the correspondence of keypoints is described in Figure 5.4.

5.5.2.1 Relative pose estimation

The relative pose is composed by a rotation matrix and a translation vector in the 3D space. We estimate the relative pose parameters as proposed in chapter 7. On the rare cases when there are no existing tracked keypoints, e.g. in the beginning of a sequence, we use the keypoint-free pose estimation as described in section 7.3.

5.5.2.2 Keypoints matching initialization

Each selected keypoint will take motion prior distribution from the nearest PhC cell. Each keypoint is associated with a 15×15 patch centred at the keypoint location; these patches are used as the input of our matching method. We estimate the matching initialization for all the selected keypoints on the image by combining the distribution of PhC prior motion for each keypoint and the epipolar constraint from the camera intrinsic and extrinsic (=relative pose) parameters.

Although the case of a completely static vehicle will make the epipolar constraint invalid during that static time, it is actually not a problem for our proposed method since the keypoints will stay at the same place in such situations, hence a correct match is found instantly at the beginning of the search.

Given the covariance ellipse for each PhC cell and given also the epipolar constraint for each keypoint, we solve the constrained optimization problem using Lagrange multipliers. We approximate the given PhC displacement array by a Gaussian distribution which has the same mean vector and the same covariance matrix. This means that the approximated motion prior distribution has the form

$$f(v) = \frac{1}{\sqrt{2\pi\hat{\mathbf{C}}_d}} \exp\left(-\frac{1}{2}(\boldsymbol{v}-\boldsymbol{m})^T \cdot \hat{\mathbf{C}}_d^{-1} \cdot (\boldsymbol{v}-\boldsymbol{m})\right) \tag{5.8}$$

where $\boldsymbol{m}$ and $\hat{\mathbf{C}}_d$ are the mean vector and the covariance matrix and $\boldsymbol{v}$ is the sought matching initialization displacement vector.

We regularize the obtained covariance matrix $\hat{\mathbf{C}}_d$ by enforcing the variance in each spatial direction to be at least equal to σ^2_{min}:

$$\tilde{\mathbf{C}}_d = \hat{\mathbf{C}}_d + \sigma^2_{min} \cdot \mathbf{I}_2 \tag{5.9}$$

The next step is to find the value of the displacement vector $\boldsymbol{v}$ that maximizes the probability density given that $\boldsymbol{v}$ lies on the epipolar line, which is equivalent to finding a point on the epipolar line which has minimum Mahalanobis distance to the center of the covariance ellipse. The solution can be obtained by solving the following optimization problem

$$(\boldsymbol{v} - \boldsymbol{m})^T \cdot \tilde{\mathbf{C}}_d^{-1} \cdot (\boldsymbol{v} - \boldsymbol{m}) \quad \rightarrow \quad \min \tag{5.10}$$

under the epipolar constraint

$$\boldsymbol{y}_h^T \cdot \mathbf{F} \cdot \boldsymbol{x}_h \quad = \quad 0 \tag{5.11}$$

where $\mathbf{F}$ is the *fundamental matrix*, $\boldsymbol{x}_h$ and $\boldsymbol{y}_h$ are the homogeneous version of the vector $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively and $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{v}$. Let us also define $\mathbf{F}'$ be a truncated $\mathbf{F}$ matrix which consists only of its first two rows.

Using a Lagrange multiplier λ_1, we obtain the following equation system:

$$\begin{pmatrix} \tilde{\mathbf{C}}_d^{-1} & \mathbf{F}' \cdot \boldsymbol{x}_h \\ (\mathbf{F}' \cdot \boldsymbol{x}_h)^T & 0 \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{v} \\ \lambda_1 \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{C}}_d^{-1} \cdot \boldsymbol{m} \\ -(\boldsymbol{x}_h^T \cdot \mathbf{F} \cdot \boldsymbol{x}_h) \end{pmatrix} \tag{5.12}$$

Considering the symmetric nature of the 3×3 pre-multiplying matrix, the displacement vector $\boldsymbol{v}$ can be expressed in a closed form solution.

Soft constraint optimization The relative pose between two image frames can only be an estimate in this early phase of processing (recall that we are still in the process of finding the set of keypoint matches). Hence it is certainly not 100% accurate. We propose a soft epipolar constraint as a further improvement of the keypoint matching initialization. The optimization step will allow the keypoint to be slightly off of the epipolar line, instead of forcing them to lie exactly on the epipolar line.

We recall the loss function in Eq. 5.10 as $Q_1(\boldsymbol{v})$ and the imposed epipolar soft constraint $Q_2(\boldsymbol{v})$ is added to complete the loss function with a well-defined balancing factor λ_2 empirically set at 0.3.

$$Q(\boldsymbol{v}) = Q_1(\boldsymbol{v}) + \lambda_2 Q_2(\boldsymbol{v}) \tag{5.13}$$

where $Q_2(\boldsymbol{v})$ is a function of the squared distance to an epipolar line and it is similar to Eq. 5.11 multiplied by a normalizing factor.

$$Q(\boldsymbol{v}) = (\boldsymbol{v} - \boldsymbol{m})^T \tilde{\mathbf{C}}_d^{-1} (\boldsymbol{v} - \boldsymbol{m}) + \lambda_2 \left(\frac{1}{\|\mathbf{F}' \cdot \boldsymbol{x}_h\|} \boldsymbol{y}_h^T \cdot \mathbf{F} \cdot \boldsymbol{x}_h \right)^2 \tag{5.14}$$

In order to minimize $Q(\boldsymbol{v})$, we calculate the first derivative with respect to $\boldsymbol{v}$ and equal it to zero. This way, the matching initialization for each keypoint is given by the displacement vector $\boldsymbol{v}$ as follows

$$\boldsymbol{v} = \left[\tilde{\mathbf{C}}_d^{-1} + \frac{\lambda_2}{\|z\|^2}\boldsymbol{z}\boldsymbol{z}^T\right]^{-1}\left[\tilde{\mathbf{C}}_d^{-1}\cdot\boldsymbol{m} - \frac{\lambda_2(\boldsymbol{x}_h^T\cdot\mathbf{F}\cdot\boldsymbol{x}_h)}{\|z\|^2}\boldsymbol{z}\right] \tag{5.15}$$

where $\boldsymbol{z} = \mathbf{F}'\cdot\boldsymbol{x}_h$.

5.5.2.3 Refinement using the epipolar KLT matching

In order to convert the provisional estimate into a final estimate, we employ a matching method based on the KLT optimization with an epipolar constraint. The procedure of the epipolar-constrained KLT matching has already been discussed in section 5.4.

Soft constraint optimization Similar to the soft constraint imposed in Eq. 5.13, we can derive a closed form solution of $\boldsymbol{v}$ with a balancing factor α_2 empirically set at 0.7.

$$\boldsymbol{v} = -0.5\left[\mathbf{A} + \frac{\alpha_2}{\|\boldsymbol{z}\|^2}\boldsymbol{z}\boldsymbol{z}^T\right]^{-1}\left[\boldsymbol{b} + 2\frac{\alpha_2}{\|\boldsymbol{z}\|^2}(\boldsymbol{x}_h^T\cdot\mathbf{F}\cdot\boldsymbol{x}_h)\boldsymbol{z}\right] \tag{5.16}$$

5.5.2.4 Mismatch detection

The final matching position based on the resulting displacement vector will be assessed further by multiple steps of the mismatch detection procedure. The mismatch detection module is used twice: after the motion prior estimation and after the epipolar KLT matching. Its role is to identify and eliminate mismatches. Each mismatch detection module consists of two checks: depth projection check and MSD check.

Depth projection check: Given the camera intrinsic matrix ($\mathbf{K}$) and the extrinsic parameters (${}^n\mathbf{R}_{n+1}, {}^n\boldsymbol{t}_{n+1}$), if the scale of ${}^n\boldsymbol{t}_{n+1}$ is provided by an odometry sensor or by estimating the ground plane, we can calculate the pixel position in $I(n+1)$ when the pixel position in $I(n)$ and the depth of the 3D point (d) are known. It is calculated as

$$\tilde{\boldsymbol{x}}_{n+1} = \mathbf{K}\cdot{}^n\mathbf{R}_{n+1}\cdot\mathbf{K}^{-1}\cdot\tilde{\boldsymbol{x}}_n + \mathbf{K}\cdot{}^n\boldsymbol{t}_{n+1}/d \tag{5.17}$$

where $\tilde{\boldsymbol{x}}$ is in homogeneous coordinates. Assuming that all the keypoints are further than one meter from the camera, we can determine a segment of the epipolar line on which the match of the keypoint should lie on the $I(n+1)$.

Hence any motion estimate which brings the keypoint to a position outside of the desired epipolar line segment can be regarded as a bad result and thus discarded.

MSD check: The MSD between two image patches A and B with size $p \times q$ is calculated as

$$MSD = \frac{\sum_{i,j}(A(i,j) - B(i,j))^2}{p \cdot q} \tag{5.18}$$

The MSD check ensures that the new MSD value after shifting is lower than the old MSD, which indicates that a better match has been found. We only retain the motion vectors which bring the keypoint patch into a new patch with a lower MSD value.

$$MSD_{new} \underset{\text{reject}}{\overset{\text{accept}}{\lessgtr}} MSD_{old} \tag{5.19}$$

5.6 Experiments on prior-based epipolar matching

In order to evaluate our method, we use KITTI image sequences from the *odometry training set*, which are taken from a moving car in an open traffic environment. In addition, we use the estimated translation for the previous frame and its keypoints locations kindly provided by the authors of (Persson, Piccini, Felsberg and Mester 2015) in order to test our method. The system proposed by Persson et al. (Persson et al. 2015) computes a high quality estimate of egomotion as well as very precise trajectories of the tracked keypoints. The keypoint correspondences have been extensively filtered by RANSAC and computed from stereo images of the KITTI odometry benchmark. It is to date one of the best vision-only approaches to odometry on that benchmark, even though they are restricted to the integer pixel grid. We use the keypoints provided by FAST detector after being filtered internally and we estimate the initial match for each of these keypoints into the next frame using the method described above.

We tested our method on 11 different sequences of the KITTI dataset, comprising of 23,157 image frames and millions of keypoints. The analysis of the matching between our keypoint matches and corresponding matches from (Persson et al. 2015) is performed regarding two values: Euclidean match distance and MSD of a patch.

5.6.1 Motion priors

We analyzed the MSD value of patches after applying the estimated motion prior. Out of all keypoints matches in a frame, we can calculate the average MSD for each frame, as presented in Figure 5.6 and Figure 5.7, where the comparison of the MSD value after each step can be observed. It shows gradual decrease of MSD values: from the starting position to a new position after motion priors, and finally to the final position after epipolar matching. It can be easily noticed that most of the time the information of motion prior is so crucial that it leads the patch very close to the optimal matching position, as illustrated by the closeness of the MSD value after the motion prior and the final MSD value after epipolar matching.

5.6.2 Epipolar matching

The motion prior estimation is followed by epipolar matching step in order to further find a better match for each of the keypoints patches. Figure 5.6 and 5.7 compares the MSD between our results and the results using method (Persson et al. 2015).

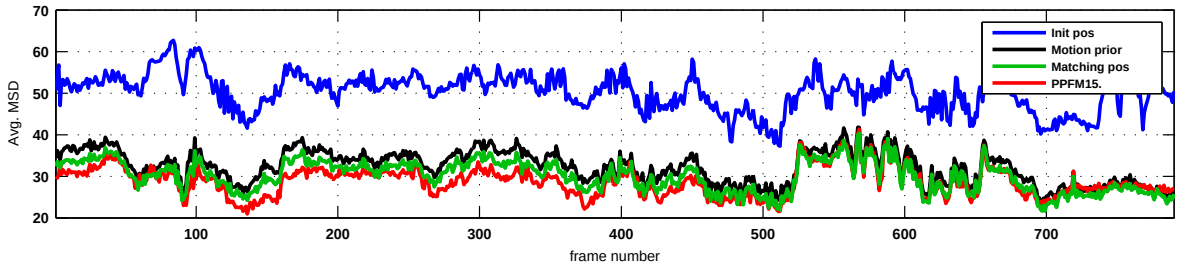

Figure 5.6: MSD comparison of keypoint patches after different stages of the proposed method: initial position (blue), after motion priors (black), final matching (green), and MSD using method from (Persson et al. 2015) (red) on KITTI sequence No.3

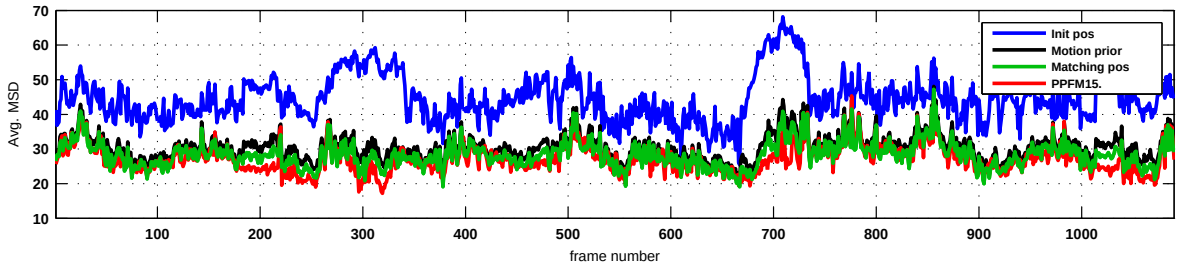

Figure 5.7: MSD comparison of keypoint patches after different stages of the proposed method: initial position (blue), after motion priors (black), final matching (green), and MSD using method from (Persson et al. 2015) (red) on KITTI sequence No.6

Table 5.1 shows the overall results of our method on the KITTI dataset. Based on the keypoints that passed the mismatch detection, our matching method has a lower MSD on more than 50% of keypoints than method (Persson et al. 2015) which heavily relies on the keypoints depth information, unlike our proposed method.

The distance values between our keypoint match and the results from (Persson et al. 2015) for each KITTI sequence are shown in Table 5.1. The overall distance mean is about 3 pixels, showing a high degree of closeness to the matches of (Persson et al. 2015).

Our mismatch detection method is capable of identifying strong false match candidates. Table 5.1 shows that the keypoints rejection rate is between 14.45%-68.45%. There are several factors causing the rejection of keypoints. Most rejection cases happen when the phase correlation method cannot provide reliable motion prior information, typically due to insufficient structure in the image, very long jumps of keypoints, and repetitive/periodic structures in the image. As a result, the keypoints move to the wrong direction and they are rejected either due to high distance to epipolar line segment or high MSD value.

Higher image frame rate or lower vehicle speed increase the quality of our results since the motion between consecutive frames is smaller and the cases of keypoint with a very

	frames	keypoints						
seq	total	total	rejection		with better MSD		distance	
			#points	percentage	#points	percentage	mean	std
0	4,537	1,626,656	535,295	32.91%	511,549	46.87%	2.66	1.21
1	1,097	225,059	102,074	45.35%	72,615	59.04%	5.01	3.07
2	4,657	1,894,521	1,028,222	54.27%	449,335	51.87%	3.22	2.11
3	797	430,102	62,140	14.45%	198,847	54.04%	2.37	0.70
4	267	89,669	61,381	68.45%	16,833	59.51%	3.56	1.99
5	2,757	1,060,460	359,691	33.92%	351,031	50.09%	2.84	1.38
6	1,097	348,949	195,367	55.99%	81,019	52.75%	3.12	1.53
7	1,097	474,541	106,592	22.46%	213,505	58.03%	2.56	3.75
8	4,067	1,723,007	574,475	33.34%	600,295	52.27%	2.75	1.16
9	1,587	590,631	299,963	50.79%	150,833	51.89%	2.93	1.55
10	1,197	462,561	126,667	27.38%	176,362	52.51%	3.26	1.51

Table 5.1: **The statistics of keypoints rejection, number of keypoints with lower MSD on KITTI sequences and keypoints match distance *w.r.t.* (Persson et al. 2015).**

long jump can be avoided as it is happening in Figure 5.6 for frames 550 to 800. This work can be extended by working on multiple frames instead of only matching between two frames. The 3D position of the keypoints can be extracted and utilized to get a better matching results, especially on long jumps and repetitive areas such as tiles and fence which cannot be solved by using only differential matching approaches.

5.7 Concluding remarks

In this chapter, it has been demonstrated that a good initialization through *motion prior* estimation significantly helps keypoints matching towards the final correct matches. In particular, it is beneficial to employ pyramidal phase correlation to obtain motion probability distributions on every part of the image frames. In the next chapters, we extend from two-frame matching into multi-frame matching, thus obtaining keypoints depth and 3D position estimation in order to have a better initialization for the keypoints matcher and the tracker.

Chapter 6

Triangulation

6.1 Introduction to triangulation

Triangulation is the step to compute the 3D position of a keypoint from multi-view observations. In the case of monocular frameworks, it is usually a triangulation between matches observed from two different times, namely here frame $n-1$ and frame n. Hartley et al (Hartley and Sturm 1997) presented the classic method to determine the best triangulated 3D position from a pair of corresponding points on the image plane. In the general case that the viewing rays from frame $n-1$ and frame n do not exactly intersect due to some errors (e.g. pose estimation error or matching error), they proposed a method to choose the best triangulated 3D position which minimized the reprojection errors on both frames. Another alternative approach is to choose the triangulated 3D point as the middle point along the shortest gap between the two viewing rays.

One can also assume that the keypoint position at one frame as a fixed point while refining the keypoint position at another frame so that the viewing rays perfectly intersect at the triangulated point. In my approach, I always restrict the keypoint position on the epipolar line. By doing so, we are sure that the viewing rays from frame $n-1$ and frame n meet during triangulation. The triangulation precision is definitely dependent on the precision of the keypoint 2D pixel positions, i.e. matching accuracy. Hence, I analyze the triangulation error due to the matching error, similar to the uncertainty propagation approach proposed by Di lLo et al ((Di Leo, Liguori and Paolillo 2011), (Di Leo and Paolillo 2011)).

The contributions of this chapter are the following:

- I derive the triangulation equation for the case of epipolar-constrained keypoint correspondence.
- I compute the triangulation error due to the matching error.

In this chapter, first I will derive the triangulation procedure to find the triangulated 3D point given a pair of correspondence which are conformant to the epipolar structure.

Then, I analyze the effect of the matching error to the triangulation error. Finally, the projection of the triangulated 3D point to a third view (frame $n+1$) is presented.

6.2 Triangulation from a pair of corresponding keypoints

Given a keypoint correspondence between frame $n-1$ and n, the corresponding 3D point can be determined by triangulation. It is important to note that the keypoint triangulation is only performed when the car is not in a static mode (see section 7.5).

I assume here that the point correspondence fulfills the epipolar geometry, that is: complies with the given relative pose. Figure 6.1 shows the triangulation of the keypoint correspondence $\boldsymbol{x}_{n-1}$ and $\boldsymbol{x}_n$, leading to a 3D point $\boldsymbol{p}_n$.

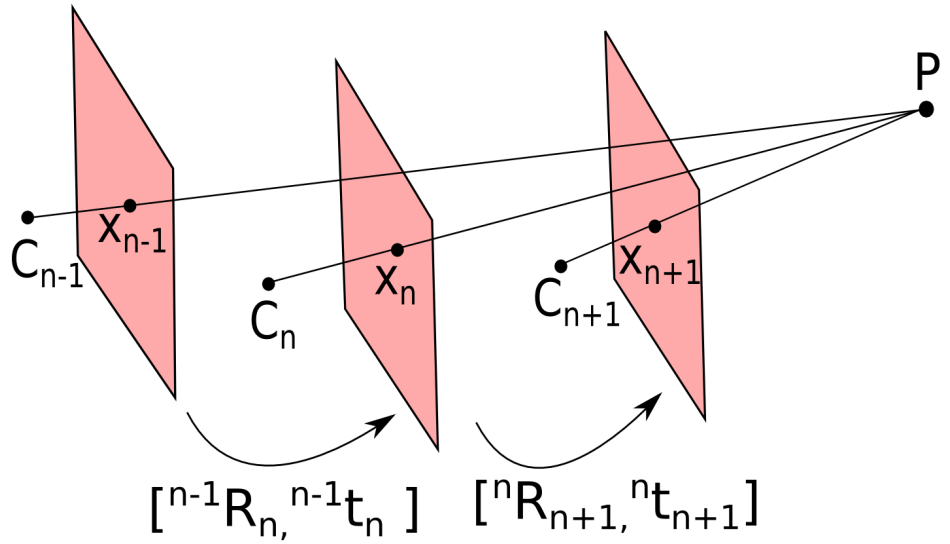

Figure 6.1: Three-view geometry: frame $n-1$, frame n, and frame $n+1$.

Let us determine the 3D point $\boldsymbol{p}_n$ in the camera coordinate system at time n by computing the intersection of the rays which pass through the centers of projection of the cameras at time $n-1$ and n, and the image points $\boldsymbol{x}_{n-1}$ and $\boldsymbol{x}_n$. Given the camera matrix $\mathbf{K}$, the normalized coordinate representation of the image points is written as $\hat{\boldsymbol{x}}_{n-1}$ and $\hat{\boldsymbol{x}}_n$.

$$\hat{\boldsymbol{x}} = \mathbf{K}^{-1} \cdot \begin{pmatrix} \boldsymbol{x} \\ 1 \end{pmatrix} \tag{6.1}$$

In many approaches, the matching is performed disregarding the epipolar relation (in particular when it is not known beforehand). Then, the points do not perfectly fulfill the epipolar relation, the projected viewing rays from the two cameras do not intersect, and triangulation yields only an estimate of the precise 3D position. In order to avoid

this problem, I force the keypoints in frame n to exactly lie on the epipolar line and refine each keypoint correspondence up to sub-pixel precision, as explained in section 5.5.2.

Given such a pair of epipolar-consistent keypoints, the following steps are performed to triangulate the 3D point $\boldsymbol{p}_n$:

1. First, the point $\hat{\boldsymbol{x}}_{n-1}$ measured in camera view $n-1$ is transformed into the camera coordinate system at time n, using the relative pose parameters ${}^{n-1}\mathbf{R}_n$ and ${}^{n-1}\boldsymbol{t}_n$:
$$\tilde{\boldsymbol{x}}_{n-1} = {}^{n-1}\mathbf{R}_n \cdot \hat{\boldsymbol{x}}_{n-1} + {}^{n-1}\boldsymbol{t}_n. \tag{6.2}$$
The centers of projection of the cameras $\boldsymbol{c}_{n-1}$ and $\boldsymbol{c}_n$ at time $n-1$ and n are expressed as well in the camera coordinate system of frame n:
$$\boldsymbol{c}_n = \mathbf{0} \tag{6.3}$$
$$\boldsymbol{c}_{n-1} = {}^{n-1}\boldsymbol{t}_n. \tag{6.4}$$

2. Let the direction of the viewing rays be expressed by the vectors $\boldsymbol{v}$ and $\boldsymbol{w}$.
$$\boldsymbol{v} = \tilde{\boldsymbol{x}}_{n-1} - \boldsymbol{c}_{n-1} \tag{6.5}$$
$$\boldsymbol{w} = \hat{\boldsymbol{x}}_n - \boldsymbol{c}_n \tag{6.6}$$
This can be rewritten as
$$\boldsymbol{v} = {}^{n-1}\mathbf{R}_n \cdot \mathbf{K}^{-1} \cdot \begin{bmatrix} \boldsymbol{x}_{n-1} \\ 1 \end{bmatrix}, \tag{6.7}$$
$$\boldsymbol{w} = \hat{\boldsymbol{x}}_n = \mathbf{K}^{-1} \cdot \begin{bmatrix} \boldsymbol{x}_n \\ 1 \end{bmatrix}. \tag{6.8}$$
Since the two viewing rays intersect at point $\boldsymbol{p}_n$, the following conditions must hold, where α and β are scale factors:
$$\boldsymbol{p}_n = \boldsymbol{c}_{n-1} + \alpha \cdot \boldsymbol{v} = {}^{n-1}\boldsymbol{t}_n + \alpha \cdot \boldsymbol{v} \tag{6.9}$$
$$\boldsymbol{p}_n = \boldsymbol{c}_n + \beta \cdot \boldsymbol{w} = \beta \cdot \boldsymbol{w} \tag{6.10}$$
$$\Rightarrow \beta \cdot \boldsymbol{w} = {}^{n-1}\boldsymbol{t}_n + \alpha \cdot \boldsymbol{v}. \tag{6.11}$$
Obviously, the three vectors ($\boldsymbol{w}$, ${}^{n-1}\boldsymbol{t}_n$, $\boldsymbol{v}$) above must lie in one plane in 3D, otherwise there is no solution to this equation. Since I restrict the keypoint match to be always along the epipolar lines, it is guaranteed that the three vectors are coplanar.

3. By post-multiplication of $\boldsymbol{v}$, I obtain
$$\beta \cdot \boldsymbol{w} \times \boldsymbol{v} = {}^{n-1}\boldsymbol{t}_n \times \boldsymbol{v} + a \cdot \boldsymbol{v} \times \boldsymbol{v} \tag{6.12}$$
$$\beta \cdot (\boldsymbol{w} \times \boldsymbol{v}) = {}^{n-1}\boldsymbol{t}_n \times \boldsymbol{v} \tag{6.13}$$

$$\beta \cdot (\boldsymbol{w} \times \boldsymbol{v})^T \cdot (\boldsymbol{w} \times \boldsymbol{v}) = (\boldsymbol{w} \times \boldsymbol{v})^T \cdot ({}^{n-1}\boldsymbol{t}_n \times \boldsymbol{v}) \tag{6.14}$$
$$\beta = \frac{(\boldsymbol{w} \times \boldsymbol{v})^T \cdot ({}^{n-1}\boldsymbol{t}_n \times \boldsymbol{v})}{\|\boldsymbol{w} \times \boldsymbol{v}\|^2}. \tag{6.15}$$

4. Combining eq. 6.10 and eq. 6.15, the 3D coordinate vector $\boldsymbol{p}_n$ can be computed:

$$\boldsymbol{p}_n = \frac{(\boldsymbol{w} \times \boldsymbol{v})^T \cdot ({}^{n-1}\boldsymbol{t}_n \times \boldsymbol{v})}{\|\boldsymbol{w} \times \boldsymbol{v}\|^2} \cdot \boldsymbol{w}. \tag{6.16}$$

In order to measure the precision of the computed triangulated points, we analyze the triangulation angle (see section 6.3) and the triangulation error based on the matching precision (see section 6.4).

6.3 Computing the triangulation angle

The triangulation step has provided us the 3D world coordinates from a set of correspondences from two image frames if the epipolar relation, that is $\mathbf{R}$ and $\boldsymbol{t}$ are known precisely.

Suppose that $\boldsymbol{x}_{n-1}$ and $\boldsymbol{x}_n$ are keypoint matches from frame $n-1$ and n in pixel coordinate. Suppose also that $\boldsymbol{p}_n$ is the resulting 3D world coordinate after triangulating $\boldsymbol{x}_{n-1}$ and $\boldsymbol{x}_n$.

The quality of the triangulated 3D world coordinate depends on a triangulation angle α_t (see figure 6.1). In general, the exact intersection point between two lines can be determined in higher precision when the two lines intersect with bigger intersection angle. For instance, we can compare the problem of determining the intersection point between two lines when they intersect with 90° and close to 0°. Hence, if the triangulation angle is big, then the triangulated world coordinate can be computed with high accuracy. On the other hand, when the triangulation angle is very small, or even close to zero, then the triangulation result has a high uncertainty.

We can compute the triangulation angle α_t by looking at the triangle $C_{n-1}PC_n$ as shown in figure 6.1 where C_{n-1} and C_n are respectively the camera center at frame $n-1$ and n.

Let ${}^{n-1}\boldsymbol{t}_n$ be the relative translation between frame $n-1$ and n, which also represents the coordinate of the camera center at frame $n-1$ in the camera coordinate system at frame n.

$$\boldsymbol{c}_n = \mathbf{0} \tag{6.17}$$

$$\boldsymbol{c}_{n-1} = {}^{n-1}\boldsymbol{t}_n \tag{6.18}$$

Having all the world coordinates of points C_{n-1}, C_n and P in the triangle $C_{n-1}PC_n$, we can now calculate the triangulation angle α_t ($= \angle C_{n-1}PC_n$)

$$\begin{aligned} \alpha_t &= \arccos\left(\frac{(\boldsymbol{p}_n - \boldsymbol{c}_n)^T \cdot (\boldsymbol{p}_n - \boldsymbol{c}_{n-1})}{\|\boldsymbol{p}_n - \boldsymbol{c}_n\| \, \|\boldsymbol{p}_n - \boldsymbol{c}_{n-1}\|}\right) \\ &= \arccos\left(\frac{\boldsymbol{p}_n^T \cdot (\boldsymbol{p}_n - {}^{n-1}\boldsymbol{t}_n)}{\|\boldsymbol{p}_n\| \, (\|\boldsymbol{p}_n - {}^{n-1}\boldsymbol{t}_n\|}\right) \end{aligned} \tag{6.19}$$

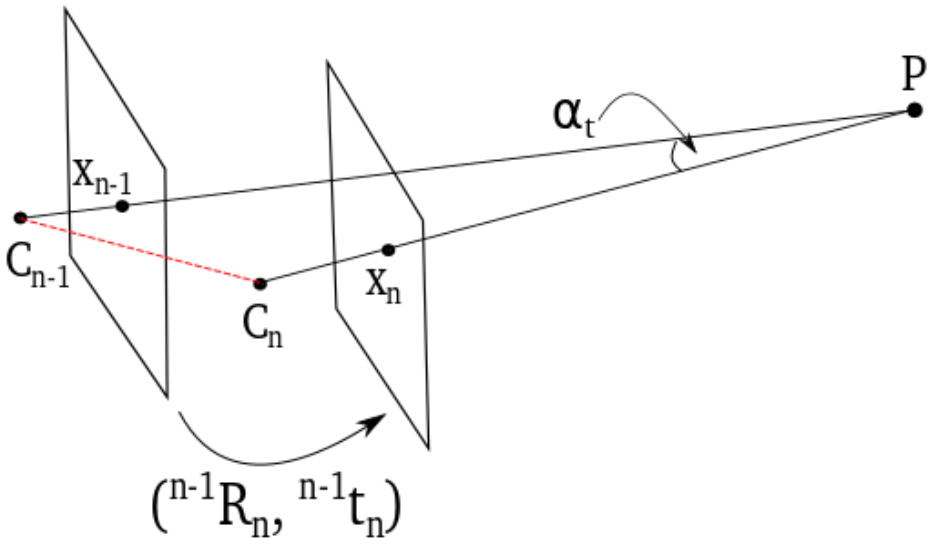

Figure 6.2: Definition of the triangulation angle $\angle C_{n-1}PC_n$.

Hence, I check on each keypoint match when performing the triangulation step, whether the triangulation angle is sufficiently big against a defined triangulation angle threshold $\tau_{\alpha t}$. This is an approximation to a more in-depth analysis of the triangulation error, as it is performed in section 6.4. Only 3D world coordinates from approved triangulation angles are considered for further processes, such as the 3D reconstruction and the scale estimation.

6.4 Analyzing the triangulation error caused by the limited matching precision

In the previous section, I assumed that the quality of the triangulation is dependent mainly on the triangulation angle. In the following sections, I will analyze the precision of triangulation in much more detail, arriving at a precise formula how the depth error is related to the correspondence error.

In the following, I assume that the relative pose (and thus also the epipolar relation) is known perfectly, but the individual keypoint matches are afflicted by correspondence errors.

6.4.1 Triangulation error analysis in the epipolar plane

In this section I analyze the triangulation error in the 2D case. Specifically, I investigate how the matching error affects the triangulation error. The triangulation is employed on matched keypoints which are constrained to be on their respective epipolar lines. In other words, both points are epipolar conformant under a given relative pose. The

triangulation problem is thus a triangulation problem on the epipolar plane, and I deal with it this way in this section. Figure 6.3 illustrates the triangulation of a correspondence match $\boldsymbol{x}_n$ and $\boldsymbol{x}_{n+1}$ on an epipolar plane. In this 2D case, the image plane degenerates into an image line. It is clear that $\boldsymbol{x}_n$ and $\boldsymbol{x}_{n+1}$ are located on these image lines. The image lines are also the epipolar lines.

Let $\boldsymbol{v}$ and $\boldsymbol{w}$ be the vectors from the camera centers at time n ending on $\boldsymbol{x}_n$ and from camera center at time $n+1$ ending at $\boldsymbol{x}_{n+1}$, respectively. I set the camera at time $n+1$ as the reference camera. This means, the camera coordinate frame of time $n+1$ is the used coordinate system in which all vectors are specified when a coordinate representation is needed.

Let $\boldsymbol{q}$ be the vector from the camera center at frame $n+1$ and perpendicular to the image line (=epipolar line) at frame $n+1$. This direction of vector $\boldsymbol{q}$ also serves as one of the coordinate axis, with another axis defined as vector $\boldsymbol{r}$ perpendicular to $\boldsymbol{q}$ (see figure 6.3). Vector $\boldsymbol{q}$ is defined to end at the epipolar line, while vector $\boldsymbol{r}$ can have arbitrary length although I assume here as a unit vector for simplicity.

The relative translation between the camera positions at time n and $n+1$ is represented by the vector $\boldsymbol{t}$. As illustrated in figure 6.3, the relative translation vector $\boldsymbol{t}$ is defined pointing from the camera center at time $n+1$ to time n because it describes how the world moves with respect to the camera. This is in accordance with the definition of the motion equations made earlier in section 2.2.2.

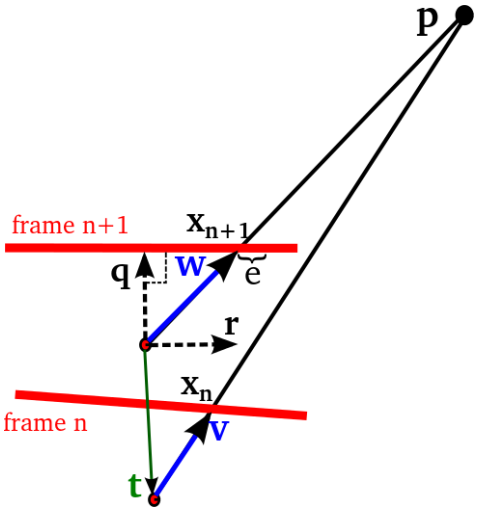

Figure 6.3: Keypoint triangulation on the epipolar plane.

We have by construction,

$$\boldsymbol{p} = \boldsymbol{t} + a \cdot \boldsymbol{v} = b \cdot \boldsymbol{w} \tag{6.20}$$

with some unknown coefficients a and b which I need to determine in order to perform the triangulation.

Let $\boldsymbol{v} = (v_1, v_2)^T$, $\boldsymbol{w} = (w_1, w_2)^T$, $\boldsymbol{t} = (t_1, t_2)^T$. Since I chose the coordinate frame of camera position $n+1$ as the reference frame, the first element of the vectors corresponds

to the direction of $\boldsymbol{q}$, while the second element is along the axis defined by $\boldsymbol{r}$. Then, the following equations hold,

$$t_1 + a \cdot v_1 = b \cdot w_1 \tag{6.21}$$

$$t_2 + a \cdot v_2 = b \cdot w_2 \tag{6.22}$$

Rearranged in a matrix form, I come to the following equation

$$\begin{pmatrix} v_1 & -w_1 \\ v_2 & -w_2 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} -t_1 \\ -t_2 \end{pmatrix}. \tag{6.23}$$

By solving the above equation, we can find the unknowns a and b:

$$a = \frac{t_2 w_1 - t_1 w_2}{w_2 v_1 - w_1 v_2} \qquad b = \frac{t_2 v_1 - t_1 v_2}{w_2 v_1 - w_1 v_2}. \tag{6.24}$$

Hence, the triangulated position $\boldsymbol{p}$ is given by

$$\boldsymbol{p} = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} = \begin{pmatrix} \dfrac{w_1(t_2 v_1 - t_1 v_2)}{w_2 v_1 - w_1 v_2} \\ \dfrac{w_2(t_2 v_1 - t_1 v_2)}{w_2 v_1 - w_1 v_2} \end{pmatrix} \tag{6.25}$$

Now, let us assume that there is a matching error e in finding the best match at frame $n+1$. Specifically, I define e as the error term along the epipolar line in selecting $\hat{\boldsymbol{x}}_{n+1}$ as the match at frame $n+1$, such that

$$\tilde{w}_2 = w_2 + e \tag{6.26}$$

This matching error leads to a noisy triangulated point $\tilde{\boldsymbol{p}}$,

$$\tilde{\boldsymbol{p}} = \begin{pmatrix} \tilde{p}_1 \\ \tilde{p}_2 \end{pmatrix} = \begin{pmatrix} \dfrac{w_1(t_2 v_1 - t_1 v_2)}{(w_2 + e) v_1 - w_1 v_2} \\ \dfrac{(w_2 + e)(t_2 v_1 - t_1 v_2)}{(w_2 + e) v_1 - w_1 v_2} \end{pmatrix} \tag{6.27}$$

The triangulation error ($\Delta\boldsymbol{p}$) is comprised of depth error (e_d) and lateral error (e_ℓ).

$$\Delta\boldsymbol{p} = \begin{pmatrix} e_d \\ e_\ell \end{pmatrix} = \begin{pmatrix} \tilde{p}_1 - p_1 \\ \tilde{p}_2 - p_2 \end{pmatrix} \tag{6.28}$$

The resulting depth error e_d of the triangulated point is given by

$$e_d = \tilde{p}_1 - p_1 = -\frac{e \cdot v_1 \cdot w_1 \cdot (t_2 v_1 - t_1 v_2)}{(w_2 v_1 - w_1 v_2)(e v_1 + w_2 v_1 - w_1 v_2)} \tag{6.29}$$

The resulting lateral error e_ℓ of the triangulated point is given by

$$e_\ell = \tilde{p}_2 - p_2 = -\frac{e \cdot v_2 \cdot w_1 \cdot (t_2 v_1 - t_1 v_2)}{(w_2 v_1 - w_1 v_2)(e v_1 + w_2 v_1 - w_1 v_2)} = \frac{v_2}{v_1} \cdot e_d = e_d \cdot \tan \angle(\boldsymbol{q}, \boldsymbol{v}) \tag{6.30}$$

The derivative of the depth error e_d w.r.t. the matching error e, is given by

$$\frac{\partial e_d}{\partial e} = \frac{e \cdot v_1^2 w_1 (t_2 v_1 - t_1 v_2)}{(w_2 v_1 - w_1 v_2)(e v_1 + w_2 v_1 - w_1 v_2)^2} - \frac{v_1 \cdot w_1 \cdot (t_2 v_1 - t_1 v_2)}{(w_2 v_1 - w_1 v_2)(e v_1 + w_2 v_1 - w_1 v_2)} \quad (6.31)$$

The first order Taylor series expansion of the triangulation error e_d is given by [1]

$$e_d = -\frac{v_1 \cdot w_1 \cdot (t_2 v_1 - t_1 v_2)}{(w_2 v_1 - w_1 v_2)^2} \cdot e + O(e)^2 \, . \quad (6.32)$$

For any two vectors $\boldsymbol{g} = (g_1, g_2)^T$ and $\boldsymbol{h} = (h_1, h_2)^T$ following the coordinate system defined by $\boldsymbol{q}$ and $\boldsymbol{r}$, the following relation holds:

$$h_2 g_1 - h_1 g_2 = \begin{pmatrix} g_1 & g_2 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \end{pmatrix} = \begin{pmatrix} g_1 & g_2 \end{pmatrix} \begin{pmatrix} h_2 \\ -h_1 \end{pmatrix} \quad (6.33)$$

Note that $\begin{pmatrix} h_2 \\ -h_1 \end{pmatrix}$ is equivalent with $\boldsymbol{h}$ rotated by 90° in anti-clockwise direction. Hence, we can express the above equation as a dot product with a modified angle due to the rotation.

$$h_2 g_1 - h_1 g_2 = \|\boldsymbol{g}\| \cdot \|\boldsymbol{h}\| \cdot \cos((\angle \boldsymbol{g}, \boldsymbol{h}) + 90°) = - \|\boldsymbol{g}\| \cdot \|\boldsymbol{h}\| \cdot \sin(\angle \boldsymbol{g}, \boldsymbol{h}) \quad (6.34)$$

where $\angle \boldsymbol{g}, \boldsymbol{h}$ represents the angle difference from $\boldsymbol{g}$ to $\boldsymbol{h}$ in anti-clockwise direction.

Thus, we can further modify equation 6.32,

$$e_d = \frac{v_1 \cdot w_1 \cdot \|\boldsymbol{t}\| \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\|\boldsymbol{v}\| \cdot (\|\boldsymbol{w}\| \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{w}))^2} \cdot e + O(e)^2 \quad (6.35)$$

$$e_d = \frac{v_1}{\|\boldsymbol{v}\|} \frac{w_1}{\|\boldsymbol{w}\|} \frac{\|\boldsymbol{t}\| \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\|\boldsymbol{w}\| \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} \cdot e + O(e)^2 \quad (6.36)$$

As I define v_1 and w_1 in the direction of $\boldsymbol{q}$, we can show that

$$\cos(\angle \boldsymbol{q}, \boldsymbol{v}) = \frac{v_1}{\|\boldsymbol{v}\|} \quad (6.37)$$

$$\cos(\angle \boldsymbol{q}, \boldsymbol{w}) = \frac{w_1}{\|\boldsymbol{w}\|} \quad (6.38)$$

which leads to

$$e_d = \frac{\cos(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \|\boldsymbol{t}\| \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\|\boldsymbol{w}\| \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} \cdot e + O(e)^2 \quad (6.39)$$

$$e_d = \frac{\|\boldsymbol{t}\| \cdot \cos(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\|\boldsymbol{w}\| \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} \cdot e + O(e)^2 \quad (6.40)$$

[1] Result derived using Mathematica

If the focal length in this 2D case is given by f_2, we can express $\|\boldsymbol{w}\|$ with the focal length and an angle.

$$\|\boldsymbol{w}\| = \frac{f_2}{\cos(\angle \boldsymbol{q}, \boldsymbol{w})} \tag{6.41}$$

Hence,

$$e_d = \frac{\|\boldsymbol{t}\| \cdot \cos(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\frac{f_2}{\cos(\angle \boldsymbol{q}, \boldsymbol{w})} \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} \cdot e + O(e)^2 \tag{6.42}$$

$$e_d = \frac{\|\boldsymbol{t}\| \cdot \cos(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{f_2 \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} \cdot e + O(e)^2 \tag{6.43}$$

The above equation is valid if $e_d, \|\boldsymbol{t}\|, f_2, e$ are all given in the same length unit, here assumed to be meters. As soon as I wish to specify any of these entities in e.g. pixels, and the others in meters, I need to introduce an explicit conversion factor between these units.

$$\frac{e_d}{[m]} = \frac{\|\boldsymbol{t}\|}{[m]} \frac{[m]}{f_2} \cdot \frac{\cos(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} \cdot \frac{e}{[m]} + O(e)^2 \tag{6.44}$$

With

$$\begin{aligned} \frac{f_2}{[m]} &= \frac{f_2}{[mm]} \cdot \frac{[mm]}{[m]} \\ \frac{e}{[m]} &= \frac{e}{[pixels]} \cdot \frac{[pixels]}{[mm]} \cdot \frac{[mm]}{[m]} \end{aligned}$$

we obtain

$$\frac{e_d}{[m]} = \frac{\|\boldsymbol{t}\|}{[m]} \frac{[mm]}{f_2} \cdot \frac{[m]}{[mm]} \cdot \frac{e}{[pixels]} \cdot \frac{[pixels]}{[mm]} \cdot \frac{[mm]}{[m]} \cdot \frac{\cos(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} + O(e)^2 \tag{6.45}$$

$$\frac{e_d}{[m]} = \frac{\|\boldsymbol{t}\|}{[m]} \frac{1}{\frac{f_2}{[mm]}} \cdot \frac{e}{[pixels]} \cdot \frac{[pixels]}{[mm]} \cdot \frac{\cos(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} + O(e)^2 \tag{6.46}$$

Let us now define δ as the number of pixels in one milimeter. The conversion factor appears on the above equation can be expressed in δ,

$$\frac{[pixels]}{[mm]} = \frac{1}{\delta} \tag{6.47}$$

so that equation 6.46 holds for e_d in meters, e in pixels, $\|\boldsymbol{t}\|$ in meters, and f_2 in milimeters.

$$e_d = \frac{\|\boldsymbol{t}\| \cdot \cos(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\delta \cdot f_2 \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} \cdot e + O(e)^2 \tag{6.48}$$

The lateral error e_ℓ can now also be derived,

$$e_\ell = e_d \cdot \tan(\angle \boldsymbol{q}, \boldsymbol{v}) = \frac{\|\boldsymbol{t}\| \cdot \sin(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\delta \cdot f_2 \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} \cdot e + O(e)^2 \tag{6.49}$$

6.4.2 Application of the precision analysis results obtained in the epipolar plane to the 3D entities

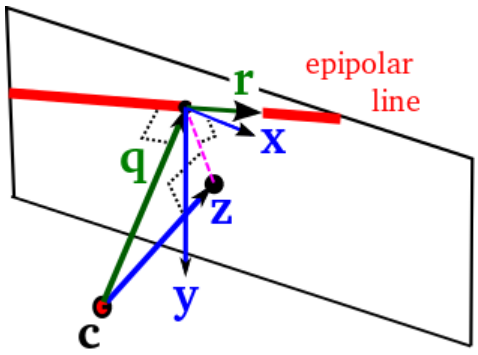

Figure 6.4: The rays to measure depth in 2D (green) and 3D (blue) cases. The triangulation depth for 2D case is in the direction of $\boldsymbol{q}$, while for 3D case the depth is in $\boldsymbol{z}-$direction.

Let the epipolar line in frame $n+1$ be defined by three parameters a, b, c as described below,

$$\boldsymbol{v}^T \cdot \mathbf{E} \cdot \boldsymbol{w} = \begin{pmatrix} a & b & c \end{pmatrix} \cdot \boldsymbol{w} = \begin{pmatrix} a & b & c \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0 \tag{6.50}$$

$$ax + by + c = 0\ . \tag{6.51}$$

As the vector $\boldsymbol{q}$ is defined from the camera center in frame $n+1$ and to the perpendicular intersection point to the epipolar line, we can find the intersection point given by the projection of the image center to the epipolar line.

$$\boldsymbol{q} = \begin{pmatrix} \dfrac{-ac}{a^2+b^2} \\ \dfrac{-bc}{a^2+b^2} \\ 1 \end{pmatrix} \tag{6.52}$$

Let us define $\boldsymbol{x}$ and $\boldsymbol{y}$ as unit vectors representing x-axis and y-axis. The positive x-axis points to the right while the positive y-axis points downwards, as illustrated in figure 6.4. Let us define $\boldsymbol{z}$ as the vector from the camera center of frame $n+1$ ending at the image center. Hence, $\boldsymbol{z}$ lies on the z-axis.

The triangulation error in the 3D case ($\Delta \boldsymbol{p}_3$) comprises three error components in the $x-$, $y-$, and $z-$ axis. The error values in the 3D case can be projected from the error values in the 2D case by considering the difference in the coordinate system.

$$\Delta \boldsymbol{p}_3 = \begin{pmatrix} e_x \\ e_y \\ e_z \end{pmatrix} = \begin{pmatrix} e_d \cdot \cos(\angle(\boldsymbol{q}, \boldsymbol{x})) + e_\ell \cdot \cos(\angle(\boldsymbol{r}, \boldsymbol{x})) \\ e_d \cdot \cos(\angle(\boldsymbol{q}, \boldsymbol{y})) + e_\ell \cdot \cos(\angle(\boldsymbol{r}, \boldsymbol{y})) \\ e_d \cdot \cos(\angle(\boldsymbol{q}, \boldsymbol{z})) + e_\ell \cdot \cos(\angle(\boldsymbol{r}, \boldsymbol{z})) \end{pmatrix} \tag{6.53}$$

$$\Delta \boldsymbol{p}_3 = \begin{pmatrix} \cos(\angle(\boldsymbol{q}, \boldsymbol{x})) & \cos(\angle(\boldsymbol{r}, \boldsymbol{x})) \\ \cos(\angle(\boldsymbol{q}, \boldsymbol{y})) & \cos(\angle(\boldsymbol{r}, \boldsymbol{y})) \\ \cos(\angle(\boldsymbol{q}, \boldsymbol{z})) & 0 \end{pmatrix} \cdot \begin{pmatrix} e_d \\ e_\ell \end{pmatrix} \tag{6.54}$$

$$\Delta \boldsymbol{p}_3 = \begin{pmatrix} \cos(\angle(\boldsymbol{q}, \boldsymbol{x})) & \cos(\angle(\boldsymbol{r}, \boldsymbol{x})) \\ \cos(\angle(\boldsymbol{q}, \boldsymbol{y})) & \cos(\angle(\boldsymbol{r}, \boldsymbol{y})) \\ \cos(\angle(\boldsymbol{q}, \boldsymbol{z})) & 0 \end{pmatrix} \cdot \begin{pmatrix} e_d \\ e_d \cdot \tan(\angle(\boldsymbol{q}, \boldsymbol{v})) \end{pmatrix} \tag{6.55}$$

The derivative of the triangulation error w.r.t. the matching error e is given by

$$\frac{\partial \boldsymbol{p}_3}{\partial e} = \begin{pmatrix} \frac{\partial e_x}{\partial e} \\ \frac{\partial e_y}{\partial e} \\ \frac{\partial e_z}{\partial e} \end{pmatrix} \tag{6.56}$$

$$\frac{\partial \boldsymbol{p}_3}{\partial e} = \begin{pmatrix} \cos(\angle(\boldsymbol{q}, \boldsymbol{x})) & \cos(\angle(\boldsymbol{r}, \boldsymbol{x})) \\ \cos(\angle(\boldsymbol{q}, \boldsymbol{y})) & \cos(\angle(\boldsymbol{r}, \boldsymbol{y})) \\ \cos(\angle(\boldsymbol{q}, \boldsymbol{z})) & 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{\partial e_d}{\partial e} \\ \frac{\partial e_\ell}{\partial e} \end{pmatrix} \tag{6.57}$$

$$\frac{\partial \boldsymbol{p}_3}{\partial e} = \begin{pmatrix} \cos(\angle(\boldsymbol{q}, \boldsymbol{x})) & \cos(\angle(\boldsymbol{r}, \boldsymbol{x})) \\ \cos(\angle(\boldsymbol{q}, \boldsymbol{y})) & \cos(\angle(\boldsymbol{r}, \boldsymbol{y})) \\ \cos(\angle(\boldsymbol{q}, \boldsymbol{z})) & 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{\|\boldsymbol{t}\| \cdot \cos(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\delta \cdot f_2 \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} + O(e) \\ \frac{\|\boldsymbol{t}\| \cdot \sin(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\delta \cdot f_2 \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} + O(e) \end{pmatrix} \tag{6.58}$$

The focal length in 3D case, denoted as f, is related to the focal length in 2D case by the following equation,

$$f = f_2 \cdot \cos(\angle(\boldsymbol{q}, \boldsymbol{z})) \quad \rightarrow \quad f_2 = \frac{f}{\cos(\angle(\boldsymbol{q}, \boldsymbol{z}))} \tag{6.59}$$

By ignoring the higher degree of the Taylor expansion $O(e)$, it follows,

$$\frac{\partial \boldsymbol{p}_3}{\partial e} = \begin{pmatrix} \cos(\angle(\boldsymbol{q}, \boldsymbol{x})) & \cos(\angle(\boldsymbol{r}, \boldsymbol{x})) \\ \cos(\angle(\boldsymbol{q}, \boldsymbol{y})) & \cos(\angle(\boldsymbol{r}, \boldsymbol{y})) \\ \cos(\angle(\boldsymbol{q}, \boldsymbol{z})) & 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{\|\boldsymbol{t}\| \cdot \cos(\angle(\boldsymbol{q}, \boldsymbol{z})) \cdot \cos(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\delta \cdot f \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} \\ \frac{\|\boldsymbol{t}\| \cdot \cos(\angle(\boldsymbol{q}, \boldsymbol{z})) \cdot \sin(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\delta \cdot f \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} \end{pmatrix} \tag{6.60}$$

6.4.3 Results of precision analysis: the sensitivity of depth estimation error w.r.t. the matching error

I show here the depth error e_z of the triangulated keypoints from a set of correspondences between two consecutive frames in the KITTI dataset. I use equation 6.60 by ignoring

the higher terms of the Taylor approximation. The previous analysis yields only the (first order) proportionality factor between a matching error in the image plane, and the resulting depth error in 3D. In order to visualize this factor in an intuitive way, I present the resulting depth error for an assumed matching error e of one pixel.

$$e_z = \frac{\|\boldsymbol{t}\| \cdot \cos^2(\angle(\boldsymbol{q}, \boldsymbol{z})) \cdot \cos(\angle\boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle\boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle\boldsymbol{v}, \boldsymbol{t})}{\delta \cdot f \cdot \sin^2(\angle\boldsymbol{v}, \boldsymbol{w})} \tag{6.61}$$

Figure 6.5 shows the depth error in color coded visualization. The different colors representing different depth ranges are described in table 6.1.

Color	**Depth error range**
Green	0-2 meters
Yellow	2-4 meters
Orange	4-6 meters
Red	more than 6 meters

Table 6.1: The color codes in fig. 6.5 representing different depth error ranges.

In general, high triangulation errors are observed for keypoints which are far away, denoted as distant keypoints, and keypoints which are in the direction of the translation vector, denoted as camera-collinear keypoints. In a straight camera motion, we can observe that the triangulation depth errors are distributed depending on the keypoint pixel position w.r.t. the image center. Keypoints which are close to the image center have higher triangulation errors in this straight motion because they are usually distant points and their 3D positions are in the direction where the car moves, as shown in figure 6.5 (top and middle images). However when the camera experiences strong rotation and translation, e.g. on a curve, the distant points are not necessarily in the center of the frame. Hence, the high triangulation errors are observed separately for distant keypoints and camera-collinear keypoints.

The depth error distributions for the above three examples presented in figure 6.5 are respectively shown in figure 6.6. The cumulative histograms are also shown so that the proportion of keypoints with depth error beyond the display depth range can be known.

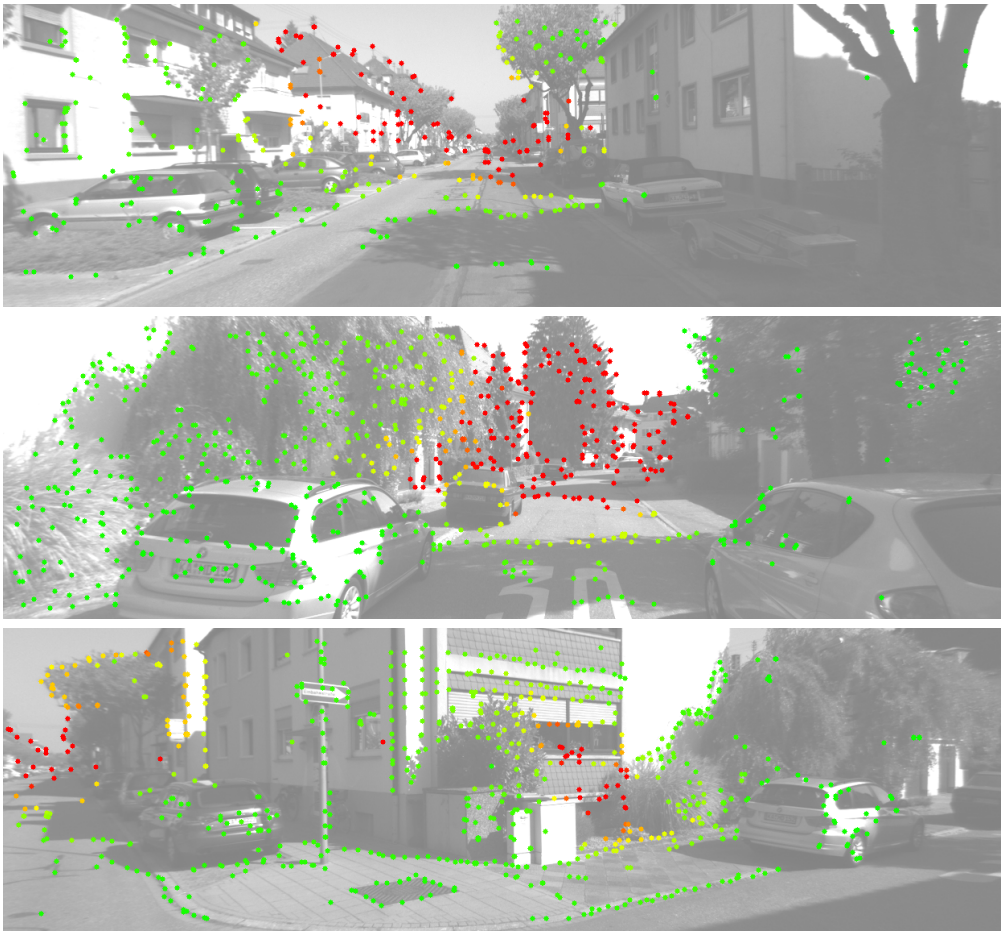

Figure 6.5: The sensitivity of the depth estimation error w.r.t. the matching error, showing low (green) to high (red) sensitivity levels, simulated on three selected image pairs from KITTI dataset (sequence 00, frames number 51-52, 106-107, and 122-123) representing both straight and turning motions.

$$\mathbf{E} = \mathbf{R} \cdot [\boldsymbol{t}]_x \tag{6.62}$$

$$\boldsymbol{v} = \mathbf{R} \cdot \mathbf{K}^{-1} \cdot \begin{pmatrix} \boldsymbol{x}_n \\ 1 \end{pmatrix} \tag{6.63}$$

$$\boldsymbol{w} = \mathbf{K}^{-1} \cdot \begin{pmatrix} \boldsymbol{x}_{n+1} \\ 1 \end{pmatrix} \tag{6.64}$$

$$\begin{pmatrix} a & b & c \end{pmatrix} = \left(\mathbf{K}^{-1} \cdot \begin{pmatrix} \boldsymbol{x}_n \\ 1 \end{pmatrix} \right)^T \cdot \mathbf{E} \tag{6.65}$$

$$\boldsymbol{q} = \begin{pmatrix} \frac{-ac}{a^2+b^2} \\ \frac{-bc}{a^2+b^2} \\ 1 \end{pmatrix} \tag{6.66}$$

$$e_z = \frac{\|\boldsymbol{t}\| \cdot \cos^2(\angle(\boldsymbol{q}, \boldsymbol{z})) \cdot \cos(\angle \boldsymbol{q}, \boldsymbol{v}) \cdot \cos^2(\angle \boldsymbol{q}, \boldsymbol{w}) \cdot \sin(\angle \boldsymbol{v}, \boldsymbol{t})}{\delta \cdot f \cdot \sin^2(\angle \boldsymbol{v}, \boldsymbol{w})} \tag{6.67}$$

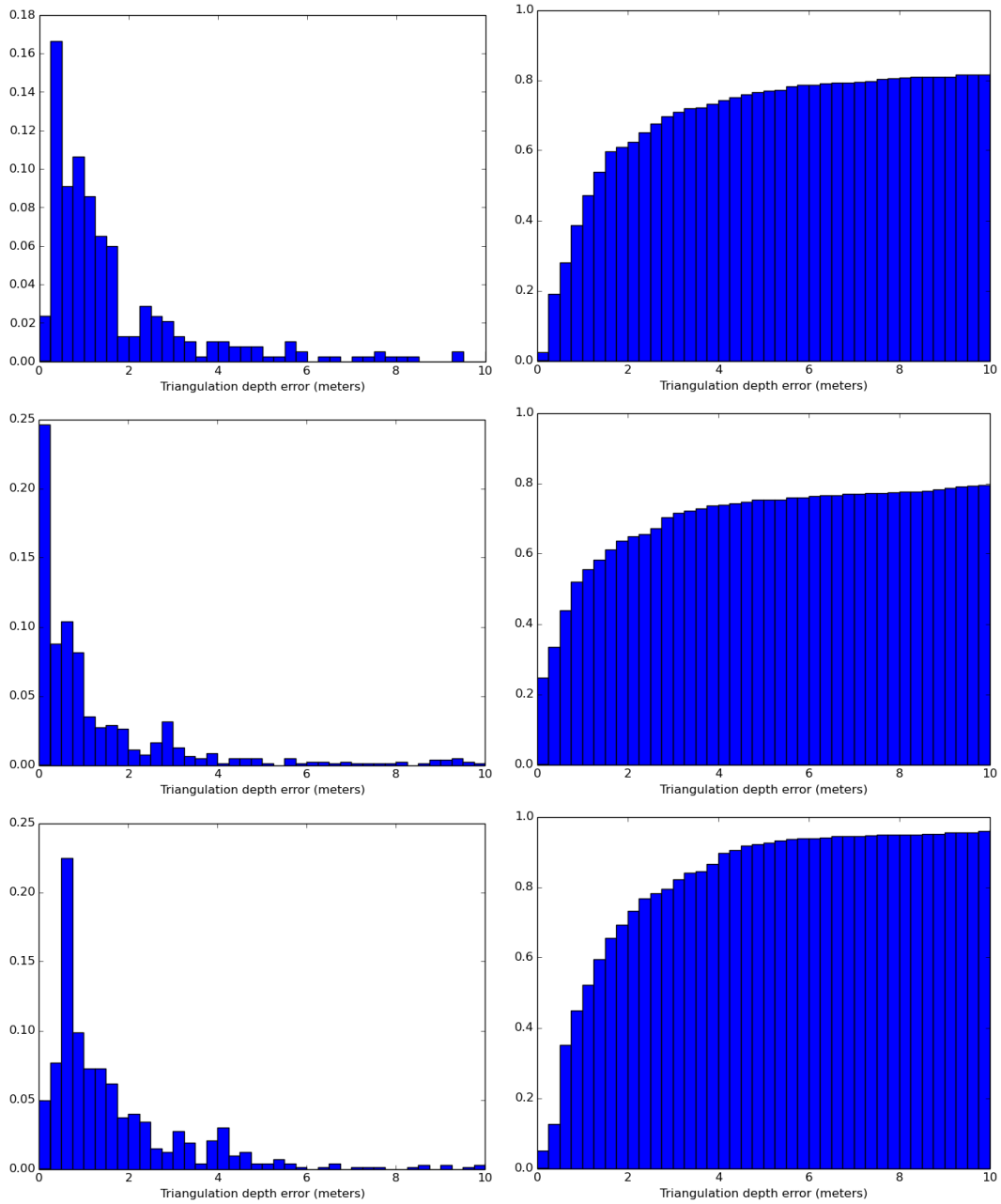

Figure 6.6: The histogram of the triangulation depth error distribution between two consecutive frames corresponding to figure 6.5, from simulations on KITTI sequence 00 frame number 51-52, 106-107, and 122-123. The cumulative histograms do not show the depth errors higher than 10 meters.

6.4.4 Propagation of the matching error to the triangulation error

If the matching uncertainty is represented by the structure tensor at the final sought position, we can propagate this matching uncertainty to the triangulated 3D point. We analyze the precision of the depths obtained from the triangulation between keypoint correspondence $\boldsymbol{x}_n$ and $\boldsymbol{x}_{n+1}$. The triangulated point coordinates $\boldsymbol{p}$ are obviously deterministic functions of the 2D measurements we make in the image plane.

Let $\boldsymbol{v}$ and $\boldsymbol{w}$ be respectively the ray vector connecting the triangulated point $\boldsymbol{p}$ with the camera center at frame n and $n+1$.

As presented in equation 6.16, we can write down the triangulated point $\boldsymbol{p}$ in 3D as an explicit function of the camera relative pose, the camera calibration matrix, and the coordinates of the two matching points in the image plane.

$$\begin{aligned} \boldsymbol{p}\left(\mathbf{K},\mathbf{R},\boldsymbol{t},\boldsymbol{x}_n,\boldsymbol{x}_{n+1}\right) &= \left(p_1,p_2,p_3\right)^T \\ &= \frac{\left(\boldsymbol{w}\times\boldsymbol{v}\right)^T\cdot\left(\boldsymbol{t}\times\boldsymbol{v}\right)\cdot\boldsymbol{w}}{\left\|\boldsymbol{w}\times\boldsymbol{v}\right\|^2} \end{aligned} \tag{6.68}$$

where

$$\boldsymbol{v} = \mathbf{R}\cdot\mathbf{K}^{-1}\cdot\begin{pmatrix}\boldsymbol{x}_n\\1\end{pmatrix} \tag{6.69}$$

$$\boldsymbol{w} = \begin{pmatrix} w_1\\ w_2\\ 1\end{pmatrix} = \mathbf{K}^{-1}\cdot\begin{pmatrix}\boldsymbol{x}_{n+1}\\1\end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x\\ 0 & f_y & c_y\\ 0 & 0 & 1\end{pmatrix}^{-1}\cdot\begin{pmatrix}x\\y\\1\end{pmatrix} = \begin{pmatrix} f_x^{-1}(x-c_x)\\ f_y^{-1}(y-c_y)\\ 1\end{pmatrix} \tag{6.70}$$

First, we take the partial derivatives of the $\boldsymbol{w}$ with respect to the coordinates of the matching point $\boldsymbol{x}_{n+1}=(x,y)^T$, assuming that the intrinsic parameter, the egomotion and the coordinates of the first (reference) point $\boldsymbol{x}_n$ are perfectly known.

$$\frac{\partial\boldsymbol{w}}{\partial\boldsymbol{x}_{n+1}} = \begin{pmatrix}\frac{\partial\boldsymbol{w}}{\partial x} & \frac{\partial\boldsymbol{w}}{\partial y}\end{pmatrix} = \begin{pmatrix} f_x^{-1} & 0\\ 0 & f_y^{-1}\\ 0 & 0\end{pmatrix} \tag{6.71}$$

Then we can take the partial derivatives of the coordinate of the triangulated 3D point $\boldsymbol{p}$ with respect to the coordinates of the matching point $\boldsymbol{x}_{n+1}=(x,y)^T$,

$$\mathbf{J}_{\boldsymbol{p}} = \frac{\partial\boldsymbol{p}}{\partial\boldsymbol{x}_{n+1}} = \frac{\partial\boldsymbol{p}}{\partial\boldsymbol{w}}\cdot\frac{\partial\boldsymbol{w}}{\partial\boldsymbol{x}_{n+1}} = \frac{\partial\boldsymbol{p}}{\partial\boldsymbol{w}}\cdot\begin{pmatrix} f_x^{-1} & 0\\ 0 & f_y^{-1}\\ 0 & 0\end{pmatrix} = \begin{pmatrix} f_x^{-1}\cdot\frac{\partial\boldsymbol{p}}{\partial w_1} & f_y^{-1}\cdot\frac{\partial\boldsymbol{p}}{\partial w_2}\end{pmatrix} \tag{6.72}$$

The partial derivatives of $\dfrac{\partial\boldsymbol{p}}{\partial w_1}$ and $\dfrac{\partial\boldsymbol{p}}{\partial w_2}$ are obtained using Mathematica.

Let $\hat{\boldsymbol{x}}_{n+1}$ be the estimated coordinates of the matching point at frame $n+1$, with the spatial uncertainty given by a covariance matrix $\mathbf{C}_{\boldsymbol{x}}$. Now, we can compute the covariance of the triangulated depth as a function of relative pose ($\mathbf{R}$ and $\boldsymbol{t}$), pixel

position in frame n ($\boldsymbol{x}_n$), camera calibration matrix ($\mathbf{K}$), and the spatial uncertainty of the pixel position in frame $n+1$ ($\mathbf{C}_{\boldsymbol{x}}$).

$$\mathbf{C}_{\boldsymbol{p}} = \mathbf{J}_{\boldsymbol{p}} \cdot \mathbf{C}_{\boldsymbol{x}} \cdot \mathbf{J}_{\boldsymbol{p}}^T \tag{6.73}$$

6.5 Keypoint propagation

The propagation of a point observed in frame n and $n-1$ into the new frame $n+1$ then simply requires to apply the relative pose transform $n \mapsto n+1$ onto the 3D vector $\boldsymbol{p}_n$ (eq. 6.16) and back-project this 3D point into the image plane of frame $n+1$. Obviously, this will give a correct match only for points which are consistent with the used relative poses $n-1 \mapsto n \mapsto n+1$, thus image points located on independent moving objects can be excluded by checking photometric and epipolar consistencies, as explained in detail in chapter 9.

The prediction of the position of a keypoint is performed by propagating the 3D coordinate of the keypoint $\boldsymbol{p}$ into the image plane of frame $n+1$. Assuming that the 3D coordinate is given in the frame n coordinate system as $\boldsymbol{p}_n$, the predicted position of the keypoint in frame $n+1$ ($\boldsymbol{x}_{n+1}$) is back-projecting the 3D point as given by

$$\alpha \cdot \begin{bmatrix} \boldsymbol{x}_{n+1} \\ 1 \end{bmatrix} = \mathbf{K} \cdot \begin{bmatrix} {}^n\mathbf{R}_{n+1} & {}^n\boldsymbol{t}_{n+1} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{p}_n \\ 1 \end{bmatrix} \tag{6.74}$$

where α is a scalar factor, and $\boldsymbol{x}_{n+1}$ is obtained by dividing by the 3rd component of the vector on the left hand side.

Knowing the epipolar geometry between two frames makes the tracking process significantly easier. The match of a keypoint can only lie on an epipolar line which can be calculated from the previous keypoint position, relative pose and intrinsic camera information. This consequently decreases the search space of the keypoint tracking process from a two dimensional search to only one dimensional search along this epipolar line.

After the keypoint propagation using equation 6.74, the keypoint position is subsequently refined using a specific adaption of epipolar differential ('Lucas-Kanade') matching as described in (Fanani et al. 2015). Keypoint correspondences with a high sum of square difference (SSD) value are automatically rejected. The SSD threshold depends on the used patch size and the radially decreasing weight mask inside this patch. I also reject keypoints if a suitable match (in terms of the SSD) cannot be found in the vicinity (i.e. within 15 pixels) of the predicted position after propagation, as it strongly indicates wrong matches or keypoints on independent moving objects. An additional outlier rejection method identifies keypoints with weak structure indicated by their low GFTT (*good-feature-to-track*) scores as proposed in (Shi and Tomasi 1994).

6.6 Outlier rejection

One of the biggest challenges in keypoint tracking is to identify outliers from the tracking results. Outliers could significantly affect the accuracy of a tracking result, especially on a visual odometry framework which heavily relies on the tracking results to estimate the camera pose.

I employ a set of outlier detection schemes to effectively remove outliers from our result. The proposed outlier rejection schemes are based on varying criteria, such as photometric consistency and geometrical constraint.

6.6.1 Outlier rejection based on 3D position validity

A pair of keypoint match from two consecutive frames could be triangulated to find the 3D world position. Triangulation could lead to invalid 3D positions which give a strong hint that outliers are detected.

The validity of the 3D position of a keypoint is defined whether it fulfils all of the following requirements:

- The point must be in front of the camera (limited by z_{min})
- The point must have a reasonable distance, it may not be exceptionally too large (limited by z_{max})
- The point must be on or above the ground (limited by h_{max})

Keypoints which fail one of the above check will be regarded as outliers and are subsequently rejected.

6.6.2 Outlier rejection based on the 3D position consistency

On every pair of two consecutive frames, a set of 3D world coordinate of keypoints can be obtained using the triangulation method. As propagation-based tracking only track static keypoints, we can check the accuracy of the new tracks of the old keypoints by comparing the newly calculated 3D world position to the previous 3D world positions which have been calculated on previous frames.

I check only keypoints which have been tracked in at least four consecutive frames. I assume that the first four readings are accurate enough to be the basis of our stability check.

Let $\boldsymbol{p}_n = (x, y, z)$ be the 3D coordinate of a keypoint in frame-n. A new 3D keypoint position with coordinate $\boldsymbol{p}_{n+1}$ is checked against a group of previously calculated coordinates $\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3, ..., \boldsymbol{p}_n$, where n is the number of consecutive frames in which the keypoint has been tracked. All the coordinates $(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3, ..., \boldsymbol{p}_{n+1})$ are already transformed into the camera coordinate system at frame-$n+1$.

I calculate the mean and the standard deviation of the keypoint distribution $\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3, ..., \boldsymbol{p}_n$. Let $\bar{\boldsymbol{p}}_n = (\bar{x}, \bar{y}, \bar{z})$ be the mean and $\boldsymbol{\sigma_n} = (\sigma_x, \sigma_y, \sigma_z)$ be the standard deviation of such distribution.

It is important to note that the 3D world keypoints are always represented with respect to the latest camera-frame coordinate system. Hence, we can check the stability of the latest tracking results by comparing to the mean of the previously tracked positions. A new track is accepted if it fulfils all of the following conditions, where r_{3d} denotes the maximum deviation ratio from the mean:

$$|x_{n+1} - \bar{x}| \leq r_{3d} \times \bar{x} \tag{6.75}$$

$$|y_{n+1} - \bar{y}| \leq r_{3d} \times \bar{y} \tag{6.76}$$

$$|z_{n+1} - \bar{z}| \leq r_{3d} \times \bar{z} \tag{6.77}$$

6.7 Stixel representation

The 3D world points from the triangulation can also be represented by stixels (Badino, Franke and Pfeiffer 2009). A stixel displays the projection of a 3D world point to the ground plane and also provides the distance information of the world point based on the shown stixel color. An example of the stixel representation is presented in figure 6.7.

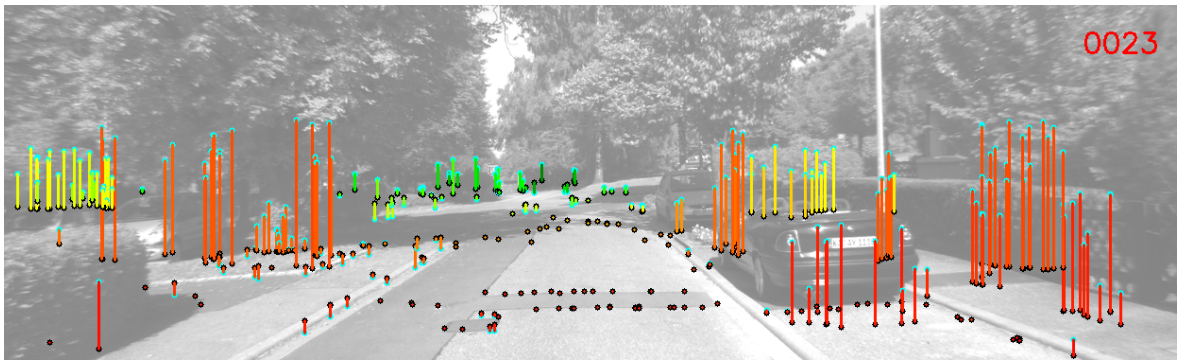

Figure 6.7: An example of the stixel representation for 3D world points with height below 2 m. The distance of the keypoint is represented by the stixel colors: red (near) to green (far).

6.7.1 Description of the ground plane

Suppose that the ground plane normal vector at a given frame is given by a unit vector $\boldsymbol{n} = [n_1, n_2, n_3]^T$ with $\|\boldsymbol{n}\| = 1$, as shown in figure 6.8. Suppose also that d is the distance from the camera center to the ground plane.

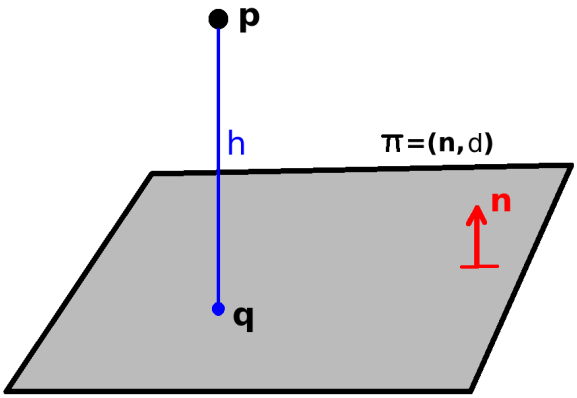

Figure 6.8: Projection of 3D world keypoint onto ground plane during stixel representation.

Hence, the ground plane can be expressed by $\boldsymbol{\pi} = [\boldsymbol{n}, d]^T = [n_1, n_2, n_3, d]^T$. where a 3D point $\boldsymbol{x}$ lies on the ground plane if it satisfies

$$d - \boldsymbol{n}^T \cdot \boldsymbol{x} = 0. \tag{6.78}$$

6.7.2 Computing the height of 3D points with respect to a ground plane

Given the ground plane equation $\boldsymbol{\pi} = [\boldsymbol{n}, d]^T$, we can compute the height of a world point from the ground plane. The height of a world point $\boldsymbol{p} = [x, y, z]^T$ with respect to a ground plane, denoted by h, is given by

$$h = d - \boldsymbol{n}^T \cdot \boldsymbol{p} \tag{6.79}$$

6.7.3 Finding the projection of a 3D point to a 3D plane

The projection of point P to the plane, expressed by vector $\boldsymbol{q} = (q_x, q_y, q_z)$, can be calculated as follows.

$$\boldsymbol{q} = \boldsymbol{p} - h \cdot \boldsymbol{n} \tag{6.80}$$

$$\boldsymbol{q} = \boldsymbol{p} - (d - \boldsymbol{n}^T \cdot \boldsymbol{p}) \cdot \boldsymbol{n} \tag{6.81}$$

This projection of point P to the ground plane is equivalent to the 3D world coordinate of the stixel foot.

6.7.4 Finding the 2D pixel coordinate of the stixel foot points

Having found the 3D coordinate of the stixel foot, I am then able to compute the 2D pixel coordinate. Let $\mathbf{K}$ be the intrinsic camera matrix.

The pixel coordinate of the stixel foot, denoted by $\boldsymbol{s}_f$, is given by

$$\begin{bmatrix} z \cdot \boldsymbol{s}_f \\ z \end{bmatrix} = \mathbf{K} \cdot \boldsymbol{q} \tag{6.82}$$

6.7.5 Transformation of the ground plane normal for frame-to-frame motion

The relative pose between frames $n-1$ and n can be expressed using relative rotation (${}^{n-1}\mathbf{R}_n$) and relative translation (${}^{n-1}\boldsymbol{t}_n$). Given a point $\boldsymbol{p}$ as 3D vector representing a fixed point in the world, the coordinate transformation is given by

$$\boldsymbol{p}_n = {}^{n-1}\mathbf{R}_n \cdot \boldsymbol{p}_{n-1} + {}^{n-1}\boldsymbol{t}_n. \tag{6.83}$$

Suppose that w have estimated the ground plane at frame n, denoted as $\boldsymbol{s}_n = [\boldsymbol{n}_n, d]^T$. The same ground plane can then be expressed in the coordinate system of frame $n+1$ using the relative rotation ${}^{n}\mathbf{R}_{n+1}$ and translation ${}^{n-1}\boldsymbol{t}_n$. The normal vector $\boldsymbol{n}_n$ as transformed to frame $n+1$ is given by

$$\boldsymbol{n}_{n+1} = \begin{bmatrix} {}^{n}\mathbf{R}_{n+1} & {}^{n}\boldsymbol{t}_{n+1} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{n}_n \\ 1 \end{bmatrix} - {}^{n}\boldsymbol{t}_{n+1} \tag{6.84}$$

6.8 Concluding remarks

In this chapter, the triangulation of keypoint matches from two different frames has been presented, under the assumption that the relative pose estimate between the frames was already available. The accuracy of the triangulated world point was analyzed based on the matching precision. Specifically, the sensitivity of depth estimation error with respect to the matching error has been investigated. A set of outlier rejection schemes has also been proposed to reject wrong keypoint correspondences.

Chapter 7

Scale-injected relative pose estimation

As mentioned earlier (see section 2.2.1), monocular visual odometry frameworks inherently suffer the scale ambiguity problem. That means, a monocular setup typically can only estimate the relative pose with an unknown scale.

In this chapter, we focus on the scale-injected pose estimation, where the scale estimates are assumed to be already available. In a real application on a car, the speed information will be available anyway as obtained from the wheel sensors. The scale estimation problem will be discussed in chapter 8 with the assumption that the camera height over ground is known. In this scale-injected pose estimation, the task is to estimate the frame-to-frame relative rotation and the translation direction. Hence, there are five parameters to be estimated: three rotation angles and two translation angles, as defined in section 2.2.2.

7.1 Introduction to pose estimation

The contributions of this chapter are the following:

- I combine a multi-window ego-rotation estimator (Barnada et al. 2015) and a pose predictor based on a dynamic vehicle model (Bradler et al. 2015) in order to obtain an initial estimate of the relative pose *before* any feature- or keypoint-based measurements are made on a new frame.
- Based on this initial pose prediction, I use an epipolar guided differential matching approach to propagate keypoints which could already be triangulated from at least one previous match.
- The propagated keypoints as well as the initial estimate of the new relative pose are subsequently jointly optimized which improves the quality of both kinds of data.

- Keypoints can usually be tracked across more than 10 frames; new keypoints which form the seed for new trajectories are dynamically created in sparsely filled image areas

With this scheme, I obtain very convincing estimates of the ego-motion of the vehicle as well as a sparse set of trajectories with only very few outliers, without using explicit robustification by RANSAC or similar methods.

7.2 Related work on pose estimation

Matching and tracking of keypoints or patches to find a set of point-to-point correspondences between two images has a very long history in computer vision, be it for stereo cameras (e.g. (Diebel, Reutersward, Thrun, Davis and Gupta 2004, Johnson, Goldberg, Cheng and Matthies 2008)) or for monocular ones. Some of the early major breakthroughs in SLAM using keypoint tracking with monocular cameras were Monocular SLAM (Davison 2003) and PTAM (Klein and Murray 2007).

As already discussed in detail in section 2.3, the methods for matching and tracking of keypoints can be structured in two classes: photometric matching, and descriptor-based matching. Photometric matching is either performed as a variant of block matching, allowing larger displacement vectors and a longer capture range, or implemented as a differential method in the spirit of the Lucas-Kanade approach (Baker and Matthews 2004) usually with a reduced convergence radius, therefore requiring a good initialization. An alternative to photometric matching of patches is feature based tracking, or more precisely: descriptor-based matching. Popular methods employ e.g. SIFT (Lowe 1999), SURF (Bay et al. 2006) and ORB (Rublee et al. 2011). Feature based tracking is especially suitable when the relative pose between the two images is unknown, hence the true correspondence can be virtually everywhere in the visual field. In a scenario where the motion between two images is small, or fast but well predictable, it is however advantageous to search for the match in the vicinity of the previous position, or (more versatile) to use predictive models for the motion (as I do in the proposed approach).

Due to the large (virtual) search area of feature/descriptor-based matching, there is a large risk of ending up with false matches: outliers. If there is a parametric model for the true motion field, iterative re-sampling methods such as RANSAC (Fischler and Bolles 1981) are usually employed to separate the outliers from the 'inlier set' of correct matches. I show in this chapter that, for the case of typical vehicle motion, such methods can be avoided.

In an automotive environment where a vehicle exhibits only moderate variations in pitch and roll, and only slowly changing translation speed and direction, further constrained by the known couplings between rotational and translational motion, frame-to-frame motion (3D pose change) can be predicted very well (Bradler et al. 2015). Thus the epipolar structures of all motion vectors which belong to points on the static

environment can be predicted and exploited for stabilizing the computation of point matches. (Trummer et al. 2008, Mohamed et al. 2015, Yamaguchi et al. 2013).

Research on pose estimation and visual odometry is a very active field; therefore, during the writing of the present thesis, other researchers partially took up ideas from the methods published by our work group; others proposed alternative concepts that compete with the (currently) good ranking of the PbT/PMO framework. Recently, the proposed propagation-based tracking concept (Fanani et al. 2016) which is reported in this dissertation was utilized for localization in (QU 2016). Some modifications were proposed by Qu Xiaozhi, such as predicting the frame-to-frame egomotion without phase correlation principles and extending the framework to the stereo setup.

Other approaches use three instead of only two consecutive frames for estimating the pose, such as in (Pereira et al. 2017). Pereira et al (Pereira et al. 2017) proposed a cyclic keypoint tracking where keypoints tracks are circularly checked between the three consecutive frames to estimate the egomotion, which heavily relies on RANSAC and bundle adjustment. Repetitive structures along the epipolar line, which commonly cause wrong matches, are handled by projecting the previous frame to the current frame with a set of different distances. If the matches are not consistent, they are classified as dubious matches and discarded.

7.3 Keypoint-free initial relative pose estimation

This section discusses the initial estimation of the relative pose even when the keypoint matching process has not been started. A specific feature of the proposed approach is that I combine different information about the next relative pose when frame $n+1$ has been captured *without* using keypoints for that purpose. On the contrary, I use this *initial relative pose estimate* to stabilize and robustify the computation of the new corresponding keypoint positions.

To estimate the frame-to-frame relative rotation, I calculate the pitch-yaw-roll (PYR) angles between two frames as proposed by Barnada *et al* (Barnada et al. 2015). The relative translation estimation is obtained by applying an extension of work by Bradler *et al* (Bradler et al. 2015) which proposes a one-step relative pose predictor, hence providing the prediction of relative pose information based on the previous relative pose information. This pose predictor also provides the rotation estimation on rare cases when the pitch-yaw-roll estimator fails to provide rotation estimation. The next subsections (7.3.1 – 7.3.2) explain these methods in more detail.

7.3.1 Rotation estimation based on the far field

It has been shown in (Barnada et al. 2015) that good estimates of the rotational egomotion of a vehicle can be achieved by analyzing the far field of a pair of images. Barnada et al (Barnada et al. 2015) employ an enhanced version (Ochs et al. 2015) of

phase correlation, an area-based 2D motion estimator, on multiple windows located in the far field of the image (see figure 7.1) for determining a 2-D in-plane translation for each selected window. From these N_{pyr} displacement vectors, all three rotation angles can be computed, assuming that the measurement areas actually lie in the far field . For further notes on determining the rotation from this set of displacement vectors, see (Eggert, Lorusso and Fisher 1997). As recommended in (Barnada et al. 2015), I choose $N_{pyr} = 5$.

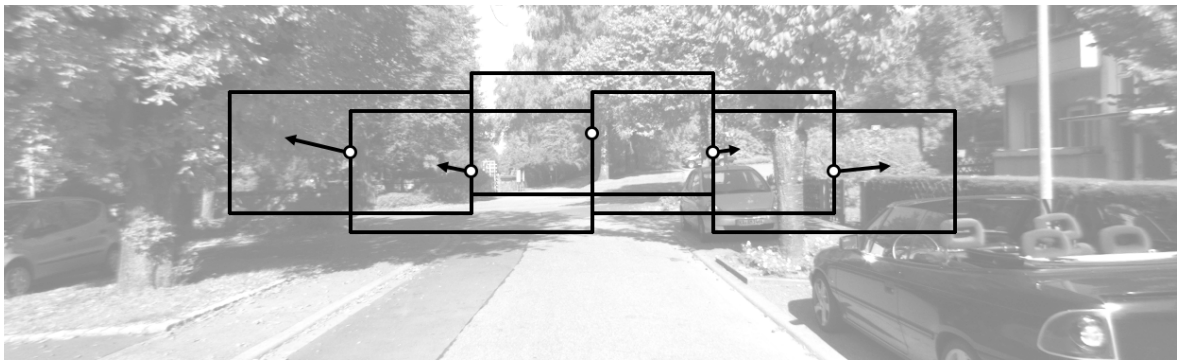

Figure 7.1: The estimation of pitch, yaw, and roll angle differences between two consecutive frames by computing the displacement vectors of five identical phase correlation windows on the far-field views. *The measurement windows are slightly vertically displaced for better visibility.*

The method also comes with several internal checks (as described in (Barnada et al. 2015)) to determine whether the obtained 2D displacement vector estimate is indeed reliable. I further add an additional check to detect invalid rotation estimates which could be caused by large independent moving objects (IMO) transversally crossing the field of view (e.g. see figure 7.2).

7.3.1.1 PYR angle validity check

A normal car usually has smooth motion and thus the change of rotation within a very short period is practically limited. It is restricted by several factors such as the wheel movement, the friction to the ground and the weight of the car. Using the PYR estimator, I can obtain the estimates of the relative rotation between two consecutive frames in a form of three angles: pitch, yaw, and roll.

Then, I analyze the rotation rates and rotation accelerations between consecutive frames from the KITTI training dataset. As shown later in a histogram in figure 7.5, we can notice from the KITTI ground truth data that the true rotation acceleration values are limited and close to zero. Hence, if a sudden change of PYR estimation results occurs, it is a strong indication that the estimates are doubtful.

The PYR estimates for KITTI sequence 07 are shown in figure 7.3, respectively for pitch, yaw, and roll angles. We can observe that there is a sudden change of PYR estimates in frame 630-650. The sudden change is caused by a big truck moving from

left to right direction, occupying much of the PYR measurement area, as shown in figure 7.2.

Figure 7.2: A failure case of PYR estimation due to a moving big truck in KITTI sequence 07.

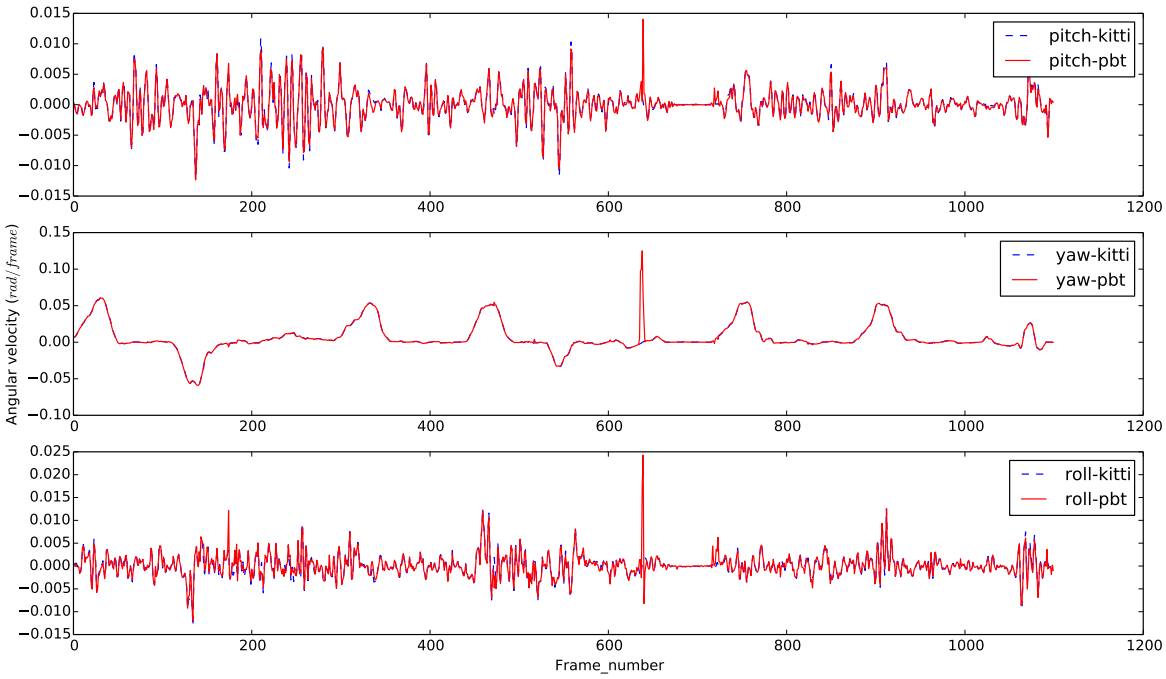

Figure 7.3: The estimates of pitch, yaw, and roll angle difference between two consecutive frames in KITTI sequence 07.

Each PYR result, namely $\Delta\theta[n], \Delta\psi[n], \Delta\varphi[n]$, is basically the angular difference between two consecutive frames $n-1$ and n. Given the frame-interval T, it can be converted into a rotation rate: $\dot{\theta}, \dot{\psi}, \dot{\varphi}$.

$$\dot{\theta}[n] = \frac{\Delta\theta[n]}{T} = \frac{\theta[n] - \theta[n-1]}{T} \tag{7.1}$$

$$\dot{\psi}[n] = \frac{\Delta\psi[n]}{T} = \frac{\psi[n] - \psi[n-1]}{T} \tag{7.2}$$

$$\dot{\varphi}[n] = \frac{\Delta\varphi[n]}{T} = \frac{\varphi[n] - \varphi[n-1]}{T} \tag{7.3}$$

As I want to detect a sudden change of PYR results to identify invalid PYR estimates, I am interested in examining the change of rotation rate, or equivalent to the rotation acceleration. The rotation accelerations for pitch, yaw, and roll angle, denoted as $\ddot{\theta}, \ddot{\psi}, \ddot{\varphi}$, are computed as follows.

$$\ddot{\theta}[n] = \frac{\dot{\theta}[n] - \dot{\theta}[n-1]}{T} \tag{7.4}$$

$$\ddot{\psi}[n] = \frac{\dot{\psi}[n] - \dot{\psi}[n-1]}{T} \tag{7.5}$$

$$\ddot{\varphi}[n] = \frac{\dot{\varphi}[n] - \dot{\varphi}[n-1]}{T} \tag{7.6}$$

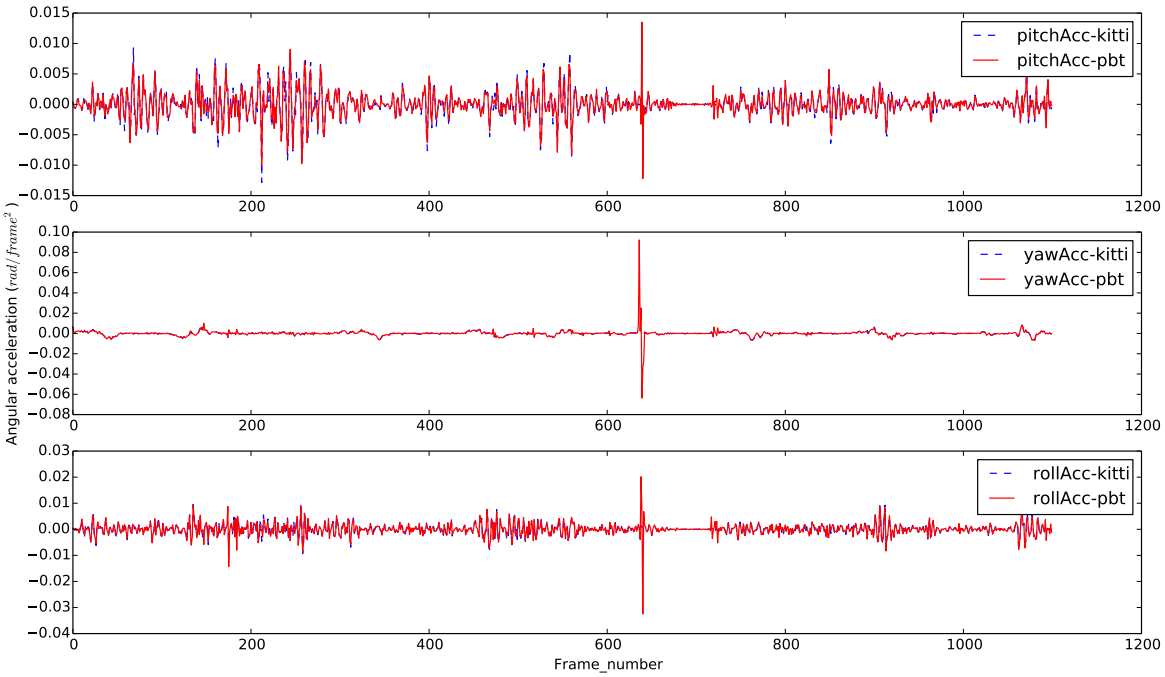

Figure 7.4: The estimated rotation acceleration using PYR estimator in KITTI sequence 07, showing very dubious peaks approximately at frame 630-640 when a big lorry is passing by.

The temporal course of rotation acceleration for pitch, yaw and roll angles in KITTI sequence 07 is shown in figure 7.4. We can now easily identify the occurrence of sudden change in PYR estimates. A proper threshold value needs to be defined to identify all dubious PYR estimates based on the rotation acceleration values.

Figure 7.5 shows the distribution of rotation acceleration for pitch, yaw, and roll angles from KITTI ground truth poses. The histogram distribution can be considered as a strong indication of valid range for rotation acceleration.

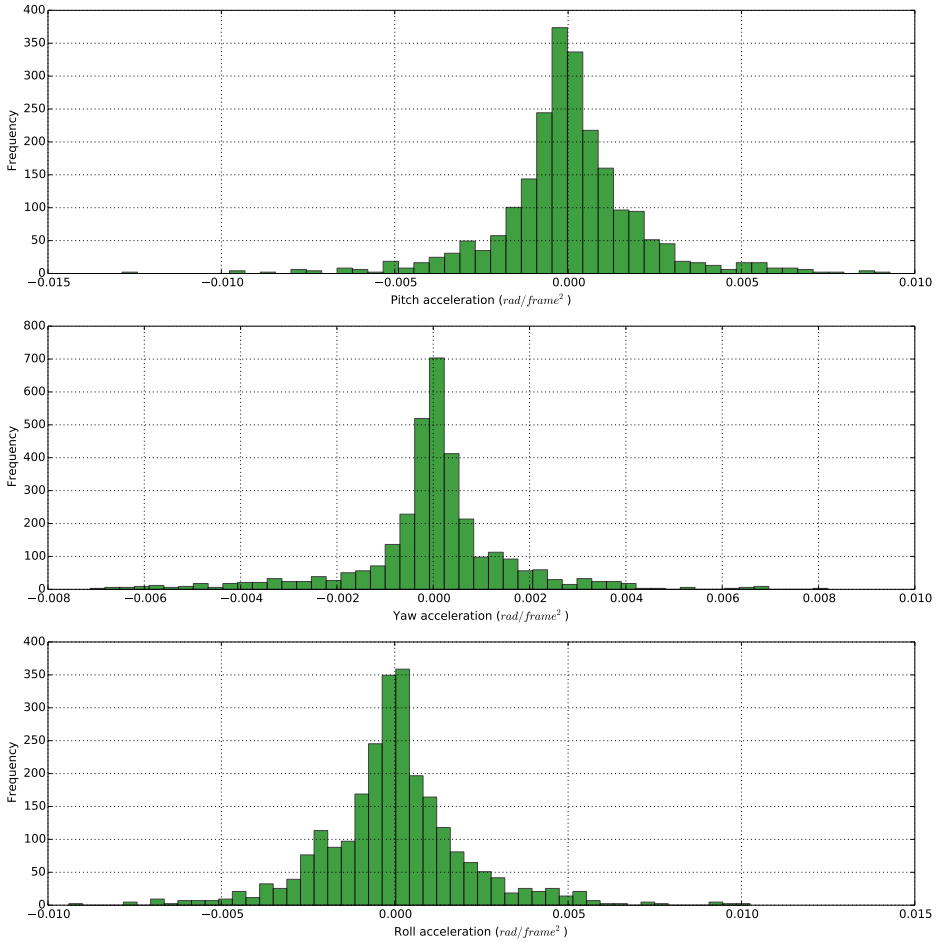

Figure 7.5: The histogram of true rotation acceleration in KITTI sequence 07.

Suppose that τ_θ, τ_ψ, τ_φ are respectively the thresholds for pitch-yaw-roll rotation acceleration. Based on the distribution of the rotation acceleration as shown in figure 7.5, the value of τ_θ, τ_ψ, τ_φ are defined. A pitch-yaw-roll angle estimate at frame n is checked for validity test using the all of the following criteria,

$$\ddot{\theta} \underset{\text{invalid}}{\overset{\text{valid}}{\lessgtr}} \tau_\theta \tag{7.7}$$

$$\ddot{\psi} \underset{\text{invalid}}{\overset{\text{valid}}{\lessgtr}} \tau_\psi \tag{7.8}$$

$$\ddot{\varphi} \underset{\text{invalid}}{\overset{\text{valid}}{\lessgtr}} \tau_\varphi \tag{7.9}$$

The proposed check is shown to be able to detect invalid pitch-yaw-roll estimates, such as observed in KITTI sequence-07 where a very big moving truck occupies the majority of the frame. Since the angle change is too steep as shown by its peak in rotation acceleration, it can correctly be identified as invalid result using the above check.

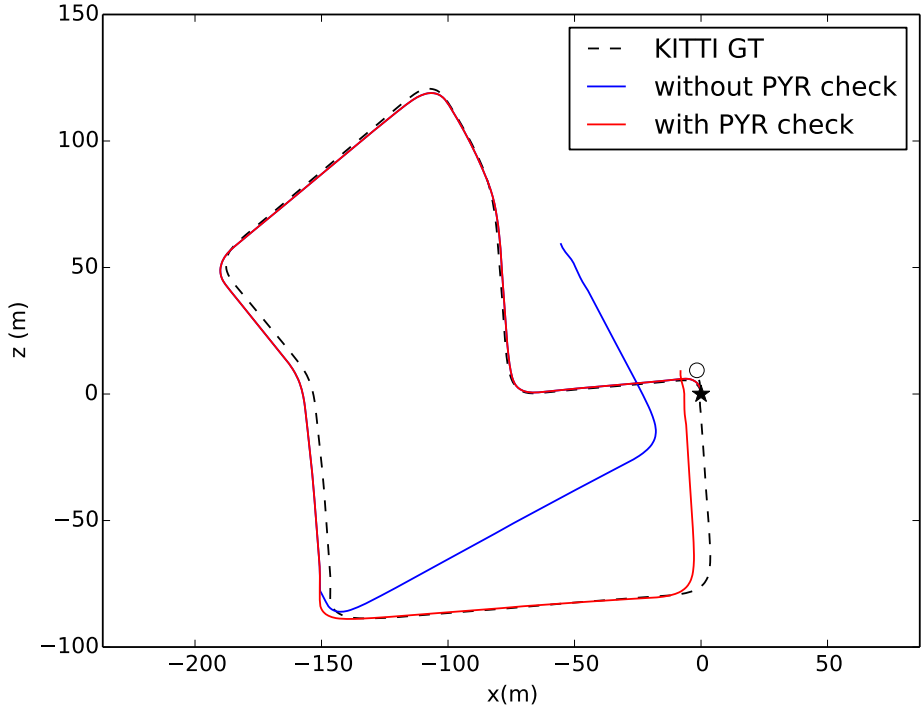

Figure 7.6: The corrected estimated course after PYR validity check in KITTI sequence 07.

The wrong rotation estimate and its corrected result can be observed in figure 7.6. This case is caused by an independent moving object occupying the PYR measurement area, hence violating the required validity condition of PYR estimation.

Whenever a PYR estimation result is deemed invalid, I estimate the relative rotation between two consecutive frames using statistical motion predictor (see section 7.3.2), which provides also the translation estimates.

7.3.2 Statistical motion predictor (SMP)

The egomotion of a car is highly constrained and steady due to its large mass making the motion inert and due to the restrictions of steering, which can be described well by a coordinated turn model. Thus the egomotion of a car can be predicted well based on the previous relative poses by employing a statistical model (Bradler et al. 2015). This predictor employs linear regression to predict the next motion parameter vector $\boldsymbol{z}$, given the previous three poses.

Let $\boldsymbol{z}$ be the pose vector at frame n (see equation 2.3) and let $\boldsymbol{y}$ be a stacked vector of the previous three poses. The best linear unbiased prediction of $\boldsymbol{z}$ given $\boldsymbol{y}$ (Bradler 2016) (Kay 1993, p. 20) is then given by

$$\hat{\boldsymbol{z}} = \mathsf{E}\left[\boldsymbol{z} \,|\, \boldsymbol{y}\right] = \mathbf{C}_{\boldsymbol{z},\boldsymbol{y}} \cdot \mathbf{C}_{\boldsymbol{y},\boldsymbol{y}}^{-1} \cdot (\boldsymbol{y} - \boldsymbol{m}_{\boldsymbol{y}}) + \boldsymbol{m}_{\boldsymbol{z}}. \tag{7.10}$$

where $\boldsymbol{m}$ denotes the mean and $\mathbf{C}_{\boldsymbol{z},\boldsymbol{y}}$ and $\mathbf{C}_{\boldsymbol{y},\boldsymbol{y}}$ are the cross and auto covariance matrix, respectively.

Not only does the prediction provide an initial translation estimate but it also provides a rotation estimate on the rare cases in which the pitch-yaw-roll-estimator does not provide a reliable rotation estimate. This way, a reasonable estimate of the next relative pose and the underlying epipolar geometry can always be obtained.

7.4 Pose and correspondence refinement by joint epipolar tracking

The initial estimates of the relative pose and the keypoint correspondences between two frames can further be refined by exploiting the fact that they are mutually constrained by each other. On one hand, a given relative pose imposes an epipolar constraint on the keypoint correspondences. On the other hand, a set of given keypoint correspondences induces a constraint on a relative pose.

I use a novel approach named joint epipolar tracking, which *simultaneously* applies differential ('Lucas-Kanade' style) epipolar-guided matching to all keypoints, as proposed in (Bradler 2016) and (Bradler et al. 2017). By applying the joint epipolar tracking, I obtain both an optimum set of keypoint correspondences (in terms of the SSD) as well as an improved relative pose, jointly complying with the epipolar relation.

The key observation in JET is that a given (or assumed) relative pose induces an individual constraint for each keypoint correspondence, and if the variation of the photometric matching term (e.g. SSD) can locally be expressed as a function of tiny variations of the corresponding keypoint position, this allows for the iterative joint optimization of all keypoint correspondences and the relative pose. Thus JET jointly minimizes the total SSD of all point correspondences while guaranteeing that they all obey the same epipolar geometry.

Let $\boldsymbol{x}$ be the keypoint position in image s_n and the sought displacement vector in image s_{n+1} is denoted by $\boldsymbol{v}$. Specifically, JET optimizes both the relative pose $\boldsymbol{z}$ and the set of point correspondences $\{\boldsymbol{x}_i \leftrightarrow \boldsymbol{x}_i + \boldsymbol{v}_i(\boldsymbol{z})\}$ simultaneously by minimizing the photometric loss between keypoint patches at image s_n and s_{n+1}.

$$\begin{pmatrix} \{\boldsymbol{v}_{i,\min}\} \\ \boldsymbol{z}_{\min} \end{pmatrix} = \underset{\{\boldsymbol{v}_i\},\boldsymbol{z}}{\arg\min} \sum_{i=1}^{N} \left\| s_{n+1}\left(\boldsymbol{x}_i + \boldsymbol{v}_i(\boldsymbol{z})\right) - s_n\left(\boldsymbol{x}_i\right) \right\|^2 \tag{7.11}$$

This is in contrast to the widely used minimization of the re-projection error (RPE), e.g. in (Song and Chandraker 2014) and (Pereira et al. 2017), that just employs a (sparse) set of keypoints extracted from the image and does not 'remember' the mere image intensities during the pose optimization process.

During the optimization of the relative pose $\boldsymbol{z}$, the statistical model of the ego-motion as presented in (Bradler et al. 2015) can be used to serve as prior knowledge and guide

the process. The optimization will then yield a relative pose whose epipolar geometry not only allows the image point correspondences to possess the smallest total SSD (accumulated across all point correspondences) but at the same time, it will be in agreement with the statistical properties of the motion (temporal and intra component correlations).

7.5 Detection when the ego-car is static

In a driving scene, sometimes the ego-car is in a static mode, for example when being halted at a traffic light, or in the middle of a traffic jam. It is important to be able to detect when the car is static, or moving very slowly, because some modules in the PMO framework should be only activated when the car is in a moving mode, i.e. the generation of new keypoints (see section 5.3) and the keypoint triangulation (see chapter 6).

We check whether the ego-car is in a static mode by looking into the estimated translation speed (scale) and the rotation rates. Specifically, we classify the ego-car as static if the estimated scale (see chapter 8) is below a threshold τ_{sc} and all of the estimated rotation angle rates ($\dot{\theta}$,$\dot{\psi}$,$\dot{\varphi}$) are below a threshold τ_a, simultaneously. The threshold values can be found in table 3.1.

7.6 Outlier rejection

One of the biggest challenges in the keypoint tracking is to identify outliers from the tracking results. Outliers could significantly affect the accuracy of a tracking result, especially on a visual odometry framework which heavily relies on the tracking results to estimate the camera pose.

I employ a set of outlier detection schemes to effectively remove outliers from the proposed result. The proposed outlier rejection schemes are based on different criteria, such as photometric consistency and geometrical constraints.

7.6.1 Outlier rejection based on the SSD value

The similarity measurement of a matching pair can be measured by its photometric error which can be expressed by the SSD between small patches surrounding the keypoint pair. Matching or tracking results with high SSD value (larger than a SSD threshold) are flagged as outliers and thus rejected. Suppose the SSD value of a patch is given by q and the SSD threshold is expressed by τ_q

$$q \underset{\text{outlier}}{\overset{\text{inlier}}{\lessgtr}} \tau_q \tag{7.12}$$

The detail of determining the SSD threshold is discussed in section 4. In the proposed framework, the SSD check is carried out during the epipolar-LK matching and joint epipolar tracking (JET).

7.6.2 Outlier rejection based on the GFTT score

The proposed photometric-based matching and tracking method is designed to track not only corner points but also edgel points. Those points are generated using the good-edgel-to-track (GETT) method (Piccini et al. 2014), which first computes the GFTT score (Shi and Tomasi 1994) of each pixel to determine if they are corner, edgel, or insignificant point.

It is obvious that I expect to have tracking results which are also either corner or edgel points. Hence, I always check the GFTT score of the proposed tracking results against the GFTT threshold as described in section 5.3, to make sure that all of them are corner or edgel points, and at the same time to identify outliers.

7.6.3 Outlier rejection based on the 3D-2D geometric consistency

While the photometric consistency are handled by the residual SSD check, I want to also ensure that the tracked keypoints satisfy the geometric consistency. For the purpose of detecting outliers, I analyze the 3D-2D geometric consistency error in three consecutive frames. Figure 7.7 illustrates the geometric relation between 3D-2D structure in three consecutive frames. I assume that I already have accurate relative pose information between each two consecutive frames: the frame $(n-1) \rightarrow n$ relative pose (${}^{n-1}\mathbf{R}_n$ and ${}^{n-1}\boldsymbol{t}_n$) and the frame $n \rightarrow (n+1)$ relative pose (${}^{n}\mathbf{R}_{n+1}$ and ${}^{n}\boldsymbol{t}_{n+1}$).

From the above pose information, we can then compute the relative pose between frame $(n-1) \rightarrow (n+1)$.

$$\begin{bmatrix} {}^{n-1}\mathbf{R}_{n+1} & {}^{n-1}\boldsymbol{t}_{n+1} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} {}^{n}\mathbf{R}_{n+1} & {}^{n}\boldsymbol{t}_{n+1} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} {}^{n-1}\mathbf{R}_{n} & {}^{n-1}\boldsymbol{t}_{n} \\ \mathbf{0} & 1 \end{bmatrix} \tag{7.13}$$

Suppose that $\boldsymbol{m}_{n-1}$, $\boldsymbol{m}_n$, $\boldsymbol{m}_{n+1}$ are the final estimates of the tracked keypoint pixel positions in frame $n-1$, n, $n+1$ respectively. I want to identify outliers within the tracked keypoints by checking the 3D-2D geometric consistency.

First, I compute the triangulated world points using the keypoints matches from any two frames out of the three consecutive frames. $\boldsymbol{p}_{12}$, $\boldsymbol{p}_{23}$, $\boldsymbol{p}_{13}$ denote the triangulated world coordinates respectively using keypoint matches from frame [$n-1$ and n], [n and $n+1$], [$n-1$ and $n+1$], as shown in figure 7.7.

Then, the world points $\boldsymbol{p}_{12}$, $\boldsymbol{p}_{23}$, $\boldsymbol{p}_{13}$ are reprojected respectively to frame $n+1$, $n-1$, n. The reprojected points are denoted as $\hat{\boldsymbol{m}}_{n-1}$, $\hat{\boldsymbol{m}}_n$, $\hat{\boldsymbol{m}}_{n+1}$, as shown in figure 7.7.

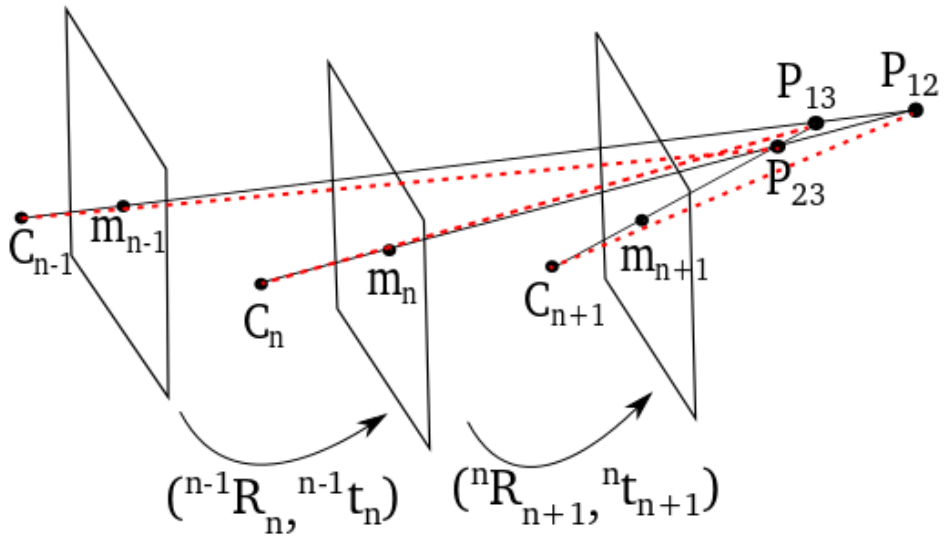

Figure 7.7: Three-view geometry: frame $n-1$, frame n, and frame $n+1$ with the triangulated world points and the back-projected rays (dashed-red lines).

7.6.3.1 Reprojection error check

The reprojection error, as already discussed in section 2.4.1, is defined as the difference between the estimated tracking position $\boldsymbol{m}$ and the projected position $\hat{\boldsymbol{m}}$ from the estimated 3D world coordinate. First of all, we check the reprojection error for each matched keypoint as defined in equation 2.25.

Buczko et al (Buczko and Willert 2016) showed that it is also important to examine beyond the magnitude of the reprojection error. They found that the variance of the reprojection error depends on the image coordinate and the depth of the keypoint, especially on the case of high-speed driving. However, the depth measurement is an error-prone problem and the resolution of the measured depth gets worse on higher distance. Hence, the normalized reprojection error, which is based on the length of the respective optical flow, is proposed as an additional filter to detect outliers.

Let ε and $\hat{\varepsilon}$ be the reprojection error and the normalized reprojection error, with thresholds given respectively by τ_ε and $\tau_{\hat{\varepsilon}}$.

$$\varepsilon \underset{\text{outlier-candidate}}{\overset{\text{inlier}}{\lessgtr}} \tau_\varepsilon \tag{7.14}$$

$$\hat{\varepsilon} \underset{\text{outlier-candidate}}{\overset{\text{inlier}}{\lessgtr}} \tau_{\hat{\varepsilon}} \tag{7.15}$$

Keypoints are regarded as outliers when they fail both checks above.

7.6.3.2 3D world coordinate consistency

Since the proposed framework is designed for static world points, the 3D world position should be consistent over time. As I have computed $\boldsymbol{p}_{12}$, $\boldsymbol{p}_{23}$, $\boldsymbol{p}_{13}$ as three estimates of the world coordinate using different combinations of matches from three consecutive frames, I want to make sure that these three estimates are consistent. It is important to note that only keypoints which pass triangulation angle test (see section 6.3) are checked for the 3D world coordinate consistency.

Let $\bar{\boldsymbol{p}}$ be the mean of the three points $\boldsymbol{p}_{12}$, $\boldsymbol{p}_{23}$, and $\boldsymbol{p}_{13}$

$$\bar{\boldsymbol{p}} = \frac{\boldsymbol{p}_{12} + \boldsymbol{p}_{23} + \boldsymbol{p}_{13}}{3} \tag{7.16}$$

Now, we look into the 3D components of each points. The 3D coordinates of $\bar{\boldsymbol{p}}$, $\boldsymbol{p}_{12}$, $\boldsymbol{p}_{23}$, and $\boldsymbol{p}_{13}$ are denoted by

$$\bar{\boldsymbol{p}} = (\bar{x}, \bar{y}, \bar{z})^T \tag{7.17}$$

$$\boldsymbol{p}_{12} = (x_{12}, y_{12}, z_{12})^T \tag{7.18}$$

$$\boldsymbol{p}_{13} = (x_{13}, y_{13}, z_{13})^T \tag{7.19}$$

$$\boldsymbol{p}_{23} = (x_{23}, y_{23}, z_{23})^T \tag{7.20}$$

The normalized world points w.r.t. the mean point are then computed respectively for $\boldsymbol{p}_{12}$, $\boldsymbol{p}_{23}$, and $\boldsymbol{p}_{13}$.

$$\hat{\boldsymbol{p}}_{12} = (\frac{x_{12}}{\bar{x}}, \frac{y_{12}}{\bar{y}}, \frac{z_{12}}{\bar{z}})^T \tag{7.21}$$

$$\hat{\boldsymbol{p}}_{13} = (\frac{x_{13}}{\bar{x}}, \frac{y_{13}}{\bar{y}}, \frac{z_{13}}{\bar{z}})^T \tag{7.22}$$

$$\hat{\boldsymbol{p}}_{23} = (\frac{x_{23}}{\bar{x}}, \frac{y_{23}}{\bar{y}}, \frac{z_{23}}{\bar{z}})^T \tag{7.23}$$

Finally, the 3D world coordinate inconsistency between $\boldsymbol{p}_{12}$, $\boldsymbol{p}_{23}$, $\boldsymbol{p}_{13}$ can be measured by the trace of the covariance matrix. If the trace is bigger than a defined threshold τ_{tr}, then the keypoint is considered as an outlier.

$$\mathrm{Tr}(\mathsf{Cov}\,[\hat{\boldsymbol{p}}_{12}, \hat{\boldsymbol{p}}_{13}, \hat{\boldsymbol{p}}_{23}]) \underset{\text{outlier}}{\overset{\text{inlier}}{\lessgtr}} \tau_{tr} \tag{7.24}$$

7.7 Course calculation

The relative pose between two consecutive frames $n-1$ and n is estimated in the form of relative rotation (${}^{n-1}\mathbf{R}_n$) and relative translation (${}^{n-1}\boldsymbol{t}_n$) as described in section 7.3 and 7.4. We can subsequently draw the real movement of the camera (and the car) by calculating the new camera position w.r.t. the starting camera position. In order to do so, I convert the relative pose into absolute pose w.r.t the starting camera position.

Let ${}^{a}\mathbf{T}_{b}$ be the transformation matrix to convert a homogeneous 3D coordinate $\hat{\boldsymbol{p}}$ from camera coordinate at frame a to frame b.

$$\hat{\boldsymbol{p}}_n = {}^{n-1}\mathbf{T}_n \cdot \hat{\boldsymbol{p}}_{n-1} \tag{7.25}$$

$$\hat{\boldsymbol{p}}_n = \begin{bmatrix} {}^{n-1}\mathbf{R}_n & {}^{n-1}\boldsymbol{t}_n \\ 0 & 1 \end{bmatrix} \cdot \hat{\boldsymbol{p}}_{n-1} \tag{7.26}$$

Let ${}^{n-1}\mathbf{R}_1$ and ${}^{n-1}\boldsymbol{t}_1$ be the already known absolute rotation and translation of the camera at frame $n-1$.

$$\hat{\boldsymbol{p}}_1 = {}^{n-1}\mathbf{T}_1 \cdot \hat{\boldsymbol{p}}_{n-1} \tag{7.27}$$

$$\hat{\boldsymbol{p}}_1 = \begin{bmatrix} {}^{n-1}\mathbf{R}_1 & {}^{n-1}\boldsymbol{t}_1 \\ 0 & 1 \end{bmatrix} \cdot \hat{\boldsymbol{p}}_{n-1} \tag{7.28}$$

The absolute pose at frame n can be calculated as follows.

$$\hat{\boldsymbol{p}}_1 = {}^{n}\mathbf{T}_1 \cdot \hat{\boldsymbol{p}}_n \tag{7.29}$$

$$\hat{\boldsymbol{p}}_1 = {}^{n-1}\mathbf{T}_1 \cdot {}^{n-1}\mathbf{T}_n^{-1} \cdot \hat{\boldsymbol{p}}_n \tag{7.30}$$

Hence, the absolute rotation and translation can be derived,

$${}^{n}\mathbf{R}_1 = - {}^{n-1}\mathbf{R}_1 \cdot ({}^{n-1}\mathbf{R}_n^{-1} \cdot {}^{n-1}\boldsymbol{t}_n) \tag{7.31}$$

$${}^{n}\boldsymbol{t}_1 = {}^{n-1}\mathbf{R}_1 \cdot {}^{n-1}\mathbf{R}_n^{-1} \tag{7.32}$$

Let $\boldsymbol{c}_n = (x, y, z)$ be the camera position in frame-n. The new camera position at frame n, $\boldsymbol{c}_n$, is simply given by ${}^{n}\boldsymbol{t}_1$. By performing the above computation repeatedly, the full movement/course of the camera can be computed.

7.8 Experiments on pose estimation

I evaluated the proposed framework on the KITTI training dataset (Geiger et al. 2012) with a total of 23201 frames. The training set consists of 11 sequences of driving scenarios. Additionally, a simulation was performed on the COnGRATS dataset which consists of synthetic driving sequences (Biedermann, Ochs and Mester 2015).

The root mean square error *RMSE* for an arbitrary angular rate entity x, measured in *rad/s* is calculated for each estimated pose parameter. Let r be the RMSE value of the measurement. I denote $\hat{x}_n$ as the estimated value at frame n, and x_n as the ground truth value at frame n. Let N be the number of frames in the sequence, and r_f be the camera frame rate expressed in frames/s. The value of r can be computed according to

$$r(\hat{x}) = \sqrt{\frac{\sum_{n=1}^{N} (\hat{x}_n - x_n)^2}{N}} \cdot r_f, \tag{7.33}$$

The mean absolute error (MAE), denoted as m, is calculated for each estimated pose parameter according to:

$$m(\hat{x}) = \frac{\sum_{n=1}^{N} |\hat{x}_n - x_n|}{N}, \tag{7.34}$$

Additionally, I also compute the normalized correlation coefficient ρ that measures the closeness between an estimate $\hat{x}$ and ground truth value x.

$$\rho = \frac{N \cdot (\sum x_n \cdot \hat{x}_n) - (\sum x_n) \cdot (\sum \hat{x}_n)}{\sqrt{(N \cdot \sum x_n^2 - (\sum x_n)^2) \cdot (N \cdot \sum \hat{x}_n^2 - (\sum \hat{x}_n)^2)}} \tag{7.35}$$

7.8.1 Rotation estimation

The estimated frame-to-frame relative rotation expressed as pitch θ, yaw ψ and roll φ angles is compared with the KITTI ground truth data in Table 7.1. These estimates are the final results after performing the keypoint-free initial pose estimation (see section 7.3) and the pose refinement using joint epipolar tracking (see section 7.4).

Table 7.1: **Results of estimated relative pose as compared to PYR2015 (Barnada et al. 2015):** *RMSE* (in *rad/s*) refers to the root mean square error in Eq. 7.33 and ρ is the normalized correlation coefficient in Eq. 7.35 *w.r.t.* KITTI ground truth. I compute the accuracy of each relative pose parameter.

Seq. no.	θ			ψ			φ			α	β
	RMSE PYR2015	**RMSE [ours]**	ρ **[ours]**	**RMSE PYR2015**	**RMSE [ours]**	ρ **[ours]**	**RMSE PYR2015**	**RMSE [ours]**	ρ **[ours]**	**RMSE [ours]**	**RMSE [ours]**
0	0.011	0.008	0.974	0.036	0.017	0.996	0.027	0.009	0.951	1.069	0.533
1	0.005	0.004	0.971	0.022	0.013	0.993	0.017	0.005	0.942	0.079	0.138
2	0.012	0.007	0.973	0.029	0.009	0.998	0.033	0.009	0.944	0.137	0.209
3	0.008	0.006	0.978	0.022	0.002	0.999	0.021	0.004	0.984	0.194	0.207
4	0.009	0.003	0.966	0.026	0.001	0.971	0.037	0.007	0.942	0.072	0.077
5	0.007	0.004	0.982	0.025	0.007	0.998	0.023	0.008	0.930	2.478	1.250
6	0.005	0.003	0.983	0.025	0.006	0.999	0.021	0.007	0.892	0.051	0.113
7	0.007	0.006	0.969	0.084	0.090	0.897	0.022	0.010	0.909	7.584	1.487
8	0.007	0.005	0.981	0.028	0.005	0.999	0.023	0.006	0.968	2.217	0.449
9	0.009	0.006	0.970	0.027	0.006	0.999	0.033	0.016	0.813	0.083	0.107
10	0.013	0.010	0.956	0.044	0.043	0.959	0.058	0.050	0.452	0.349	0.464

Table 7.1 shows the rotation estimation errors on all KITTI sequences. The rotation angle estimates are very close to the ground truth data, and the estimation errors are smaller than results from (Barnada et al. 2015). The estimation results on the KITTI sequence 3 and sequence 5 are given in fig. 7.8 and fig. 7.9 between the blue line (my result) and the black line (result from (Barnada et al. 2015)). The mean of the normalized correlation coefficient of the estimated pitch, yaw and roll angles from the proposed method are respectively 0.974, 0.983 and 0.885 in the tested 11 KITTI sequences.

The improvement of the rotation estimate is the effect of applying the pose predictor as described in section 3.2.1, especially when the pitch-yaw-roll estimation yields an 'indeterminate' result in specific cases. Additionally, the improvement also comes from

Table 7.2: **Results of estimated relative pose: alpha-beta angles are compared only when the car has speed at least 5 km/hour due to the GPS inaccuracy in low speed.** *MAE* refers to the mean absolute error in Eq. 7.34 *w.r.t.* KITTI ground truth. I compute the accuracy of each relative pose parameter.

KITTI	**MAE (rad/frame)**					**MAE (m/frame)**		
seq.	θ	ψ	φ	α	β	t_x	t_y	t_z
0	0.00048	0.00048	0.00051	0.00904	0.01552	0.01005	0.00678	0.00065
1	0.0003	0.00044	0.00036	0.00675	0.01135	0.02494	0.01485	0.00025
2	0.00045	0.00029	0.00048	0.00965	0.01121	0.01078	0.01006	0.00067
3	0.0004	0.00016	0.0003	0.00765	0.00926	0.00647	0.00483	0.00011
4	0.00026	0.00011	0.0003	0.00513	0.00542	0.00783	0.00767	0.00016
5	0.00034	0.00026	0.00042	0.00586	0.00726	0.00498	0.00417	0.00026
6	0.00027	0.00026	0.00034	0.00425	0.00683	0.00637	0.00447	0.00017
7	0.00031	0.00085	0.0004	0.00790	0.01350	0.00641	0.00464	0.00116
8	0.00037	0.00025	0.00039	0.02615	0.01747	0.01081	0.01234	0.00206
9	0.0004	0.00029	0.00058	0.00611	0.00611	0.00755	0.00625	0.001
10	0.0005	0.00058	0.00076	0.00871	0.01643	0.01034	0.00602	0.00176

the joint epipolar tracking through the optimization of the initial pose and the keypoint correspondences obtained from the initial propagation into frame $n+1$.

7.8.2 Translation angles estimation

The accuracy of the translation vector (direction only, without magnitude) is shown in Table 7.3.

Table 7.3: **Results of estimated relative translation vectors *w.r.t.* KITTI ground truth:** *RMSE* (in *meter/s*) refers to the root mean square error in Eq. 7.33 and ρ is the normalized correlation coefficient in Eq. 7.35.

KITTI	t_x		t_y		t_z	
Seq.	**RMSE**	ρ	**RMSE**	ρ	**RMSE**	ρ
0	0.184	0.617	0.089	0.666	0.036	0.999
1	0.297	0.001	0.174	0.592	0.005	1.0
2	0.165	0.419	0.141	0.442	0.036	0.999
3	0.088	0.725	0.067	0.609	0.002	1.0
4	0.113	0.293	0.110	0.054	0.002	1.0
5	0.067	0.870	0.061	0.815	0.012	0.999
6	0.081	0.843	0.051	0.924	0.005	1.0
7	0.091	0.830	0.064	0.811	0.051	0.999
8	0.154	0.667	0.299	0.193	0.126	0.999
9	0.102	0.718	0.080	0.639	0.058	0.999
10	0.344	0.199	0.094	0.606	0.114	0.999

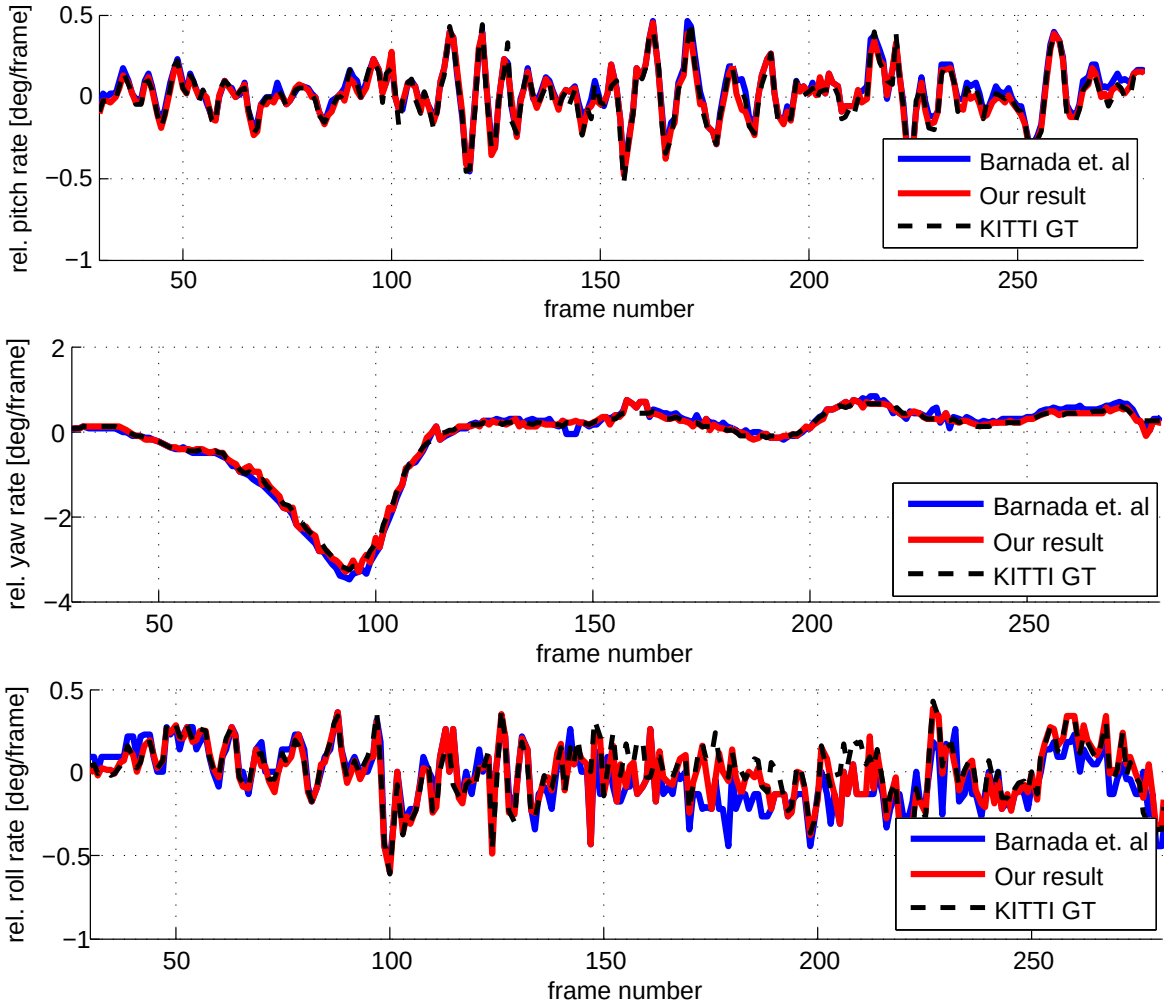

Figure 7.8: The relative rotation angles of the proposed pose estimation for KITTI dataset sequence 3 as compared with the ground truth and Barnada *et al* (Barnada et al. 2015). *Best viewed in color.* (From top: pitch, yaw, and roll)

7.8.3 Keypoint 2D trajectory

Another way to qualitatively evaluate the quality of the presented method is to compute the 2D trajectories of the tracked keypoints by accumulating the result of propagation based tracking in a sequence of images.

Some typical examples of resulting keypoint trajectories in the KITTI dataset are displayed in fig. 7.10 (*best viewed in color*). The 2D keypoint trajectory is shown for the last ten frames in blue lines, while the red dot indicates the most recent position of the keypoint. The top part of fig. 7.10 shows the case when the car moves forward, while the bottom figure shows the keypoint trajectories on a curve.

Fig. 7.11 shows an example of the keypoint trajectories on a synthetic dataset. The density of the keypoint tracking can be further increased by adjusting the minimum distance between two neighbouring keypoints in the proposed keypoint generator method.

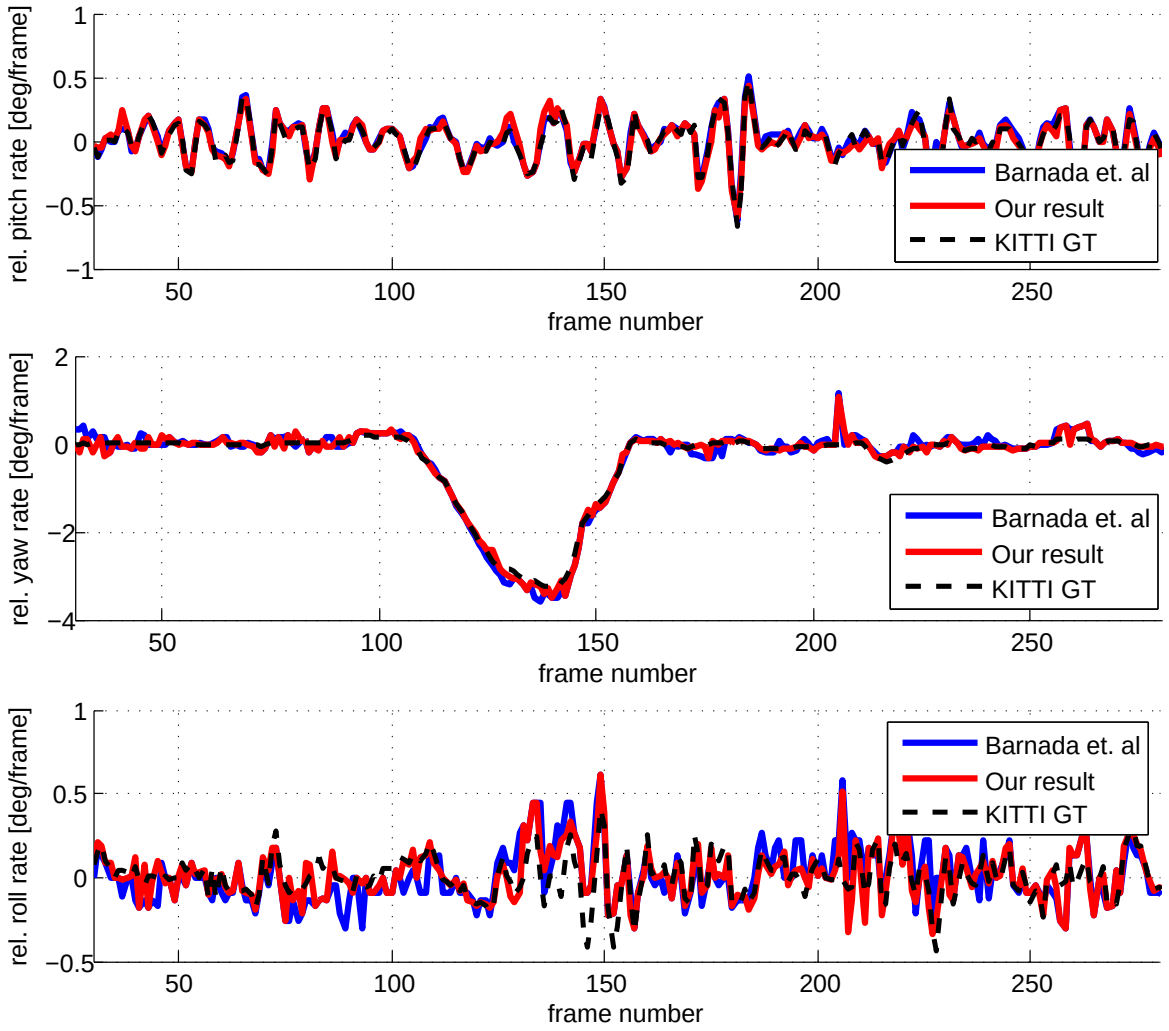

Figure 7.9: The relative rotation angles of the proposed pose estimation for KITTI dataset sequence 5 as compared with the ground truth and Barnada *et al* (Barnada et al. 2015). *Best viewed in color.* (From top: pitch, yaw, and roll)

The non-static keypoints are rejected during the process of the propagation based tracking. The rejection of a group of nearby keypoints is a strong hint that there is a moving object in that rejected area, as can be observed on fig. 7.12. The top image of fig. 7.12 shows that an 'independent moving object', a moving car, is excluded from the obtained keypoint trajectories. A similar case from the synthetic dataset is shown in fig. 7.12 (bottom) where a moving car on the front-left is seen surrounded by dense keypoint trajectories.

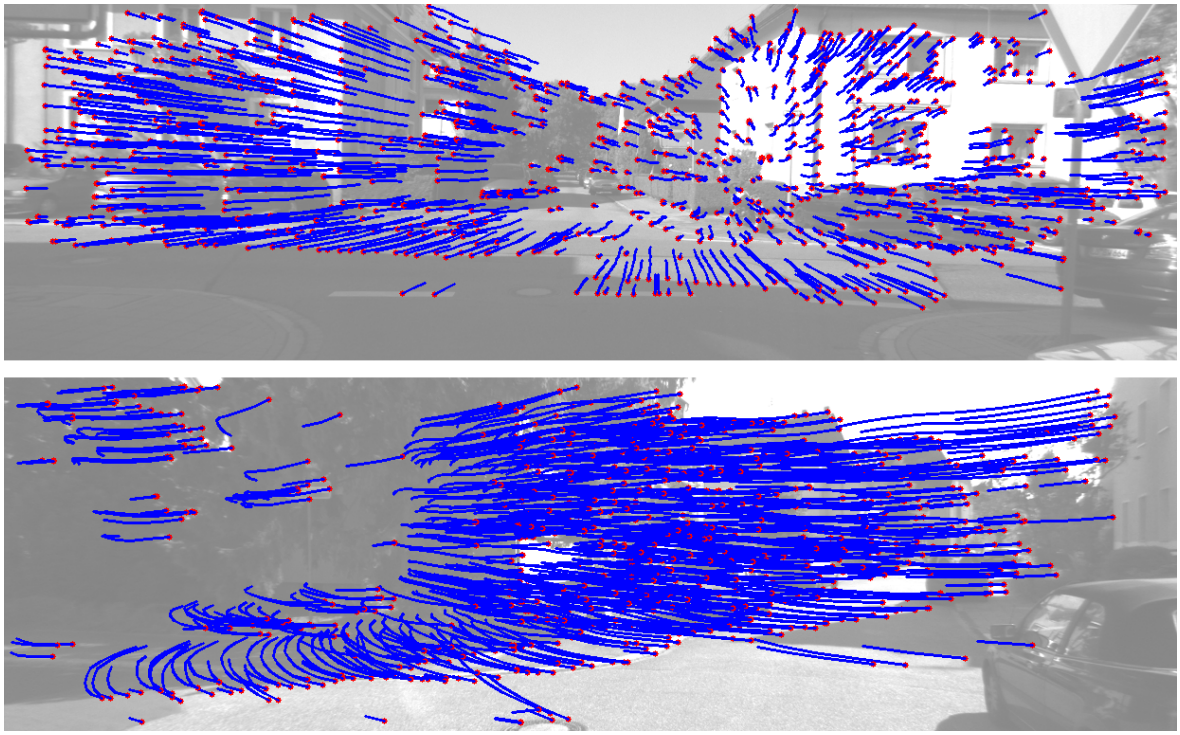

Figure 7.10: The keypoint trajectories are shown by blue lines for the last ten frames in the KITTI sequence: straight drive (*top*) and in a curve (*bottom*). *Best viewed in color.*

7.9 Concluding remarks

This chapter has presented a propagation based keypoint tracking method which computes the 2D trajectories of keypoint bundles and simultaneously determines the sequence of relative poses. I composed the system from a set of components such as a pitch-yaw-roll estimator, a pose predictor and a joint epipolar tracker, together yielding solid pose-and-trajectories estimates. I evaluated the system on the KITTI and synthetic COnGRATS driving sequences, observing a promising egomotion accuracy, for a monocular procedure without bundle adjustment. Our experimental results suggest that accurate and dense keypoint trajectories can be computed using our proposed method.

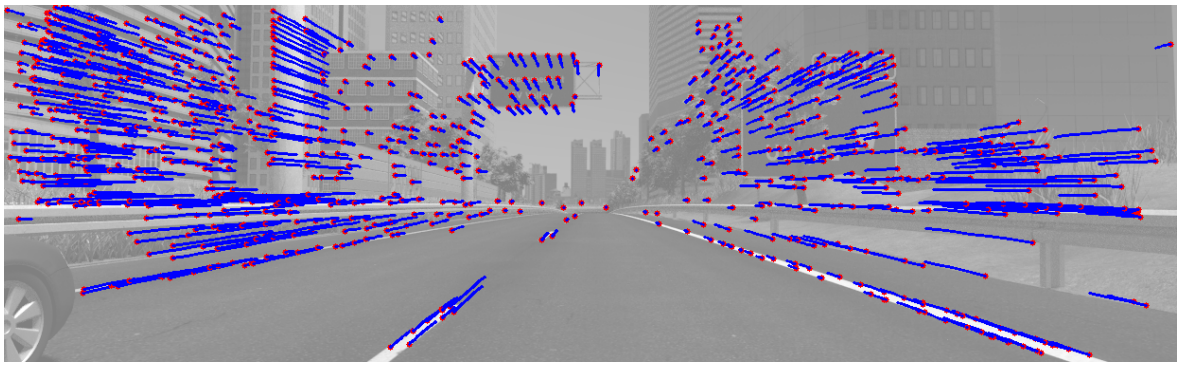

Figure 7.11: Keypoint trajectories are shown by blue lines for the last ten frames in the synthetic sequence. *Best viewed in color.*

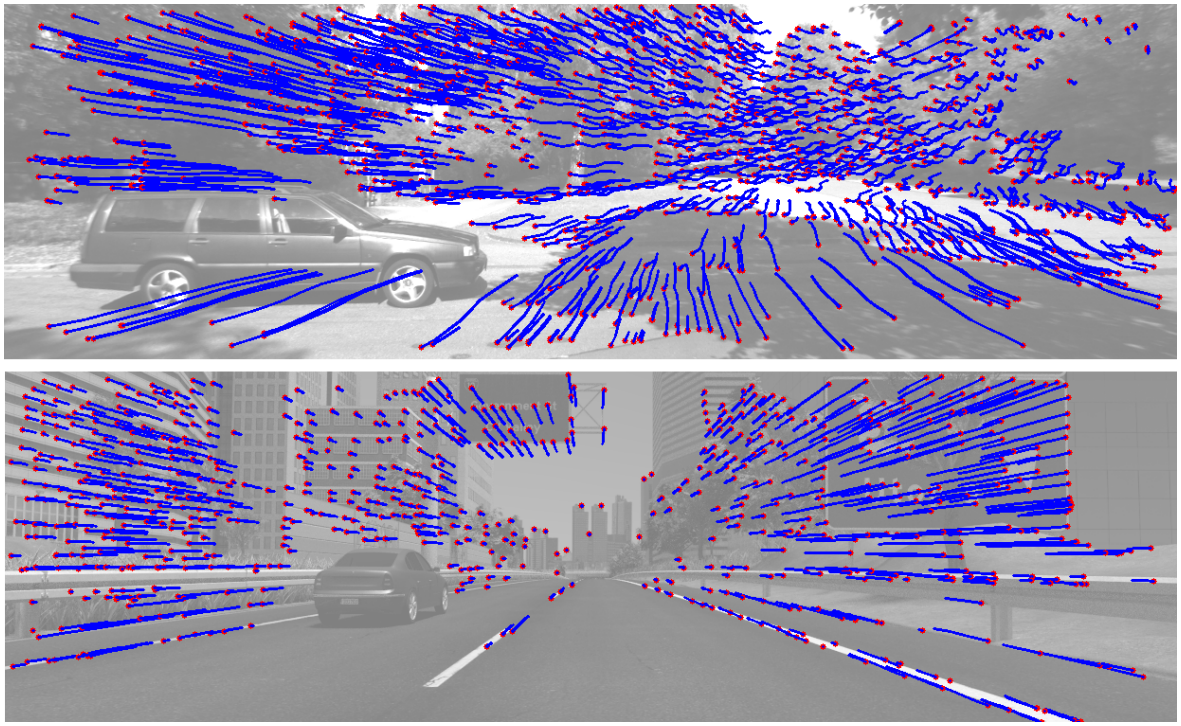

Figure 7.12: Moving object identification by rejected keypoints. The moving car is surrounded by tracked keypoints in the KITTI dataset (*top*) and in the synthetic dataset (*bottom*). *Best viewed in color.*

Chapter 8

Scale estimation

8.1 Introduction to monocular scale estimation

In an automotive driver assistance system, navigating with only one single camera is undoubtedly more challenging than combining information from two or more cameras. A stereo setup can easily extract the scaled egomotion and scaled depth information from the image stream. This is because in a stereo setup the length of the base line is known. Any triangle with known angles and one known side length can be drawn, whereas the size of a triangle with only the angles being known remains undetermined.

In contrast to the stereo setup, a monocular setup suffers from the scale ambiguity problem due to the unknown egomotion translation length between frames. Therefore, knowing the distance traveled between two points in time defines a triangulation triangle of known dimensions even for the monocular case.

We define *scale* as the translation length (e.g. in meters) between two consecutive frames as the camera moves and *relative scale* as the ratio between the scales from two consecutive frames, which is typically about 1.0.

The scale ambiguity problem in a monocular setup has been widely discussed in recent decades ((Nützi, Weiss, Scaramuzza and Siegwart 2011, Stein, Mano and Shashua 2000, Strasdat, Montiel and Davison 2010, Sturm 1997)), especially as a sub-system in visual odometry or visual SLAM systems. Without knowledge of the inter-frame distance, a monocular setup suffers from an accumulation of frame-to-frame scale errors, the so called scale-drift. This scale drift effect is inevitable in a visual odometry/SLAM system without any scale correction mechanism. It can be mitigated by bundling (joint optimization) of data from a large set of subsequent frames, but not eliminated completely, and updates of scale from time to time remain necessary. In order not to mix the effects of scale estimation and bundle adjustment, I abstain from using any bundle adjustment in this chapter, and use only frame-to-frame scale propagation and a set of elaborated scale estimation/correction methods for single frame pairs.

The majority of the early work on monocular scale correction focuses on estimating

the best homography matrix describing the ground plane on the image, such as in (Zhang and Hanson 1996) and (Arróspide, Salgado, Nieto and Mohedano 2010). While the scale can be derived from a homography matrix assuming that at least one true metric length is known, the decomposition of the homography matrix into rotation and translation is very sensitive to noise (Zhou, Dai and Li 2016). Furthermore, there is also an inherent ambiguity in the decomposition of homography matrix (Malis and Vargas 2007). Hence, in this work, I propose a monocular scale estimation with a direct motion parametrization, instead of using homography-based parametrization.

The road surface can be represented by sparse ground points or a dense ground plane. Recent methods which utilize triangulated sparse ground points to estimate the monocular scale can be found in (Mirabdollah and Mertsching 2015), (Song and Chandraker 2014) and (Zhou et al. 2016). However, these methods rely on the availability of localized and detectable ground features. On the other hand, dense area-to-area matching can be tracked back to the direct method in (Irani and Anandan 1999), which was recently also utilized in (Song and Chandraker 2014).

Both dense matching as well as sparse matching suffer when the ground area contains only insignificant structure. A comparison between such dense and sparse techniques has been provided by (Westerhoff, Lessmann, Meuter, Siegemund and Kummert 2016), underlining the above mentioned problems.

I propose a multimodal monocular scale estimation scheme which combines both dense and sparse methods and applies classifiers to detect scale outliers in the dense and sparse matching results to steer the fusion between multiple scale estimates. This extends the speed-injected pose estimation in chapter 7, which yields accurate unscaled pose estimates, but does not contain scale estimation. The proposed multimodal scale estimation method can also be used in other systems as well.

The contributions of this chapter are as follows:

- I build a two step prediction-correction scheme to estimate the monocular scale.
- I combine both dense matching and sparse matching ground plane estimation to provide a robust scale estimation.
- I estimate the monocular scale by optimizing the underlying motion parameters directly instead of the intermediate homography matrix.
- Outliers or wrong scale estimates are detected using classifiers which prevent the injection of unreliable data into the scale fusion .

8.2 Related work on monocular scale estimation

This section briefly summarizes the state of the art in monocular scale estimation. Previous work on visual-based monocular scale estimation can be grouped into two types: ground-based methods and non-ground-based methods.

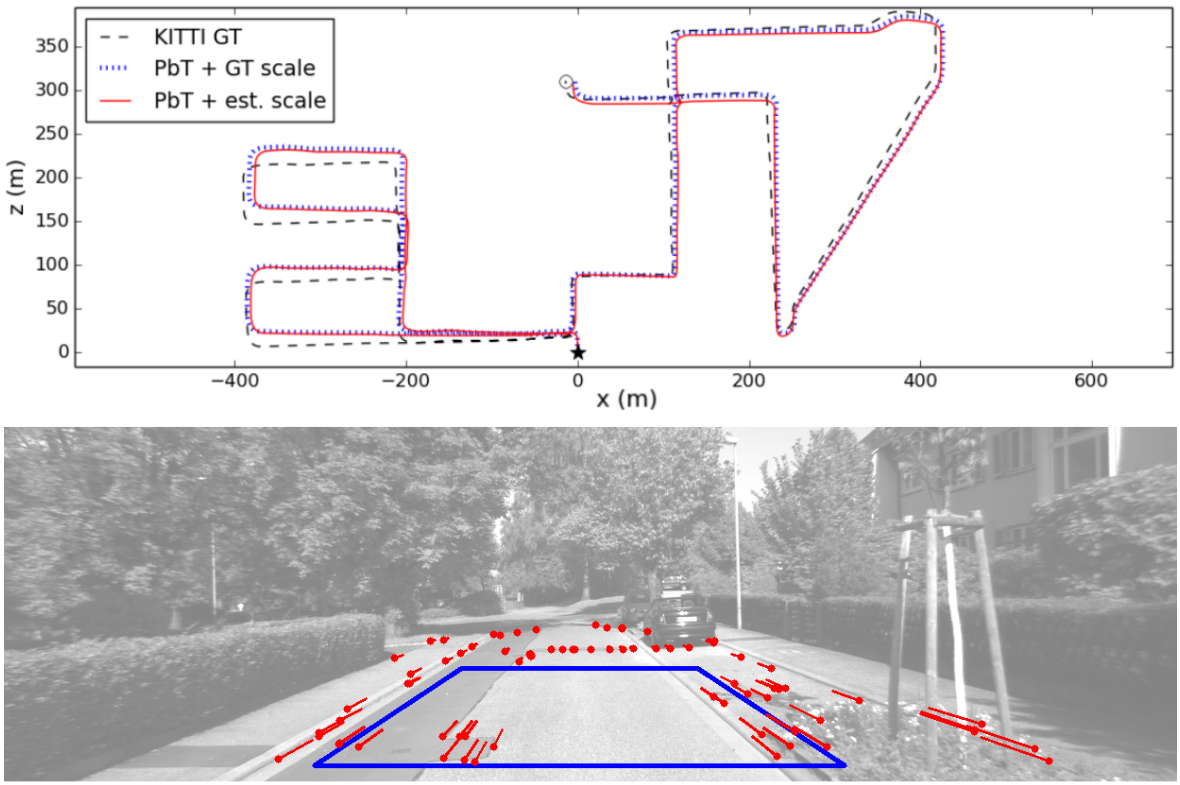

Figure 8.1: (Top) Exemplary results of different monocular visual odometry methods. The courses are shown a) for the scale-injected pose (dotted line), vs. b) using my scale estimation (solid line), and c) for comparison as the ground truth (dashed line). (Bottom) Illustration of the data used by the proposed monocular scale estimator: dense matching (blue RoI) and sparse matching (red optical flow vectors).

Ground-based methods typically rely on the assumption that the height of the camera over ground is known. This is also often combined with the assumption that the ground plane in front of the car is planar. In this scenario, ground plane estimation aims to find the relation between the ground region in two consecutive frames. A region of interest (RoI) is typically defined as a rectangle on the ground plane just in front of the car, visible as a trapezoid in the image as shown in figure 8.1(bottom).

The relation between the ground regions in two consecutive frames is commonly described by a homography matrix. The estimation of the homography matrix can be computed either using a set of sparse ground points or using a dense ground area. A set of sparse ground points can be utilized to form the best fitting ground plane like implemented in (Song and Chandraker 2014). However, the plane-fitting is very sensitive to outliers in the sparse ground points.

Another method to estimate the homography matrix is by dense matching between the ground region from two consecutive images. In comparison to matching based on sparse points, the dense plane matching is able to process richer information from all pixels inside the ground plane. However, any deviation from the ground plane assumption, e.g. by curbstones, parked cars, or other obstacles causes this method to fail. It is

more difficult to reject the non-ground objects in a dense method than in the sparse method where the keypoints already have an estimated height which can give strong hints whether the points belong to the ground plane.

Additionally, methods for estimating homography matrices can be divided into two more types where any combination with either dense or sparse estimation is possible. First, the matrix elements itself are to be estimated. Second, the underlying parameters are to be estimated directly. Due to the sensitivity of noise during the homography decomposition step, it is preferable to directly optimize the rotation and translation information instead of optimizing the homography matrix itself, as pointed out in (Zhou et al. 2016).

A combination of dense and sparse ground matching can also be employed. Song et al (Song, Chandraker and Guest 2013) proposed a monocular scale estimation method by looking at both dense matching and a sparse set of keypoints on the ground plane.

This work was then extended in (Song and Chandraker 2014) by also including object (non-ground-based) detection to estimate the ground plane and subsequently to obtain the scale estimation. The authors apply an adaptive per-frame covariance scheme for each estimate from the different cues during the data fusion. However, even by having multiple cues, the proposed method still cannot handle high-speed scenes with repeated textures and big occlusion as found in KITTI sequence 01 and 07.

Mirabdollah et al (Mirabdollah and Mertsching 2015) presented a method for monocular scale estimation based on sparse tracked ground points, using the feature descriptor as proposed in libviso (Geiger et al. 2011). Their monocular framework can rank as one of the top methods among the monocular based methods, even surpassing some of the state-of-the-art stereo methods. However, the method presented in (Mirabdollah and Mertsching 2015) does not consider the temporal variation of pitch and roll angles of the camera but they assume that the camera is always parallel to the ground plane.

Pereira et al (Pereira et al. 2017) use only one single ground keypoint to estimate the scale, while limiting the method by assuming that the car always has negligible pitch and roll angles. The camera height over ground is also assumed to be known. The chosen ground point is searched in the center bottom area of the image and should present a good image gradient along the epipolar line and its corresponding 3D position is expected to be close to the estimated ground.

Apart from the ground-based method, there are also other approaches to estimate monocular scale. The idea of employing the detection of non-ground objects and analyzing their size to estimate monocular scale was proposed by Song et al (Song and Chandraker 2014) and Frost et al (Frost, Murray et al. 2016). Obviously, this can only work for objects with *known* dimensions in the real world, and these methods are dependent on the precision by which these dimensions are known. Instead of relying on planarity assumption as in (Song and Chandraker 2014), Frost et al (Frost et al. 2016) only use object size as a comparison factor to estimate the scale. Hence, Frost et al (Frost et al. 2016) claims to be able to also handle aerial scenes. While other methods (e.g. Botterill et al (Botterill, Mills and Green 2013)) learn the object size online as the

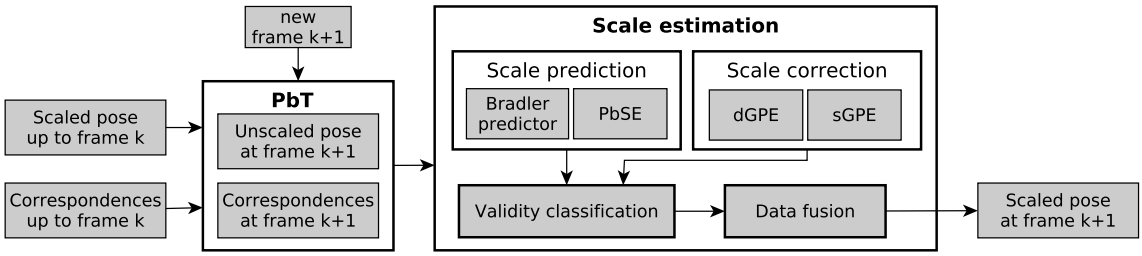

Figure 8.2: The flowchart of the monocular scale estimation. With the accurate unscaled pose has been provided by PbT framework, the ultimate goal is to estimate the scale so that the correctly scaled pose can be obtained.

basis for scale estimation when the same object is seen again on other occasions, Frost et al (Frost et al. 2016) uses a predefined size of the object class in order to prevent drift when estimating the object size online. Unfortunately, the pre-definition of some object sizes is not always reliable since objects of similar kind, such as cars, might have a significant size variation.

Scale estimation for monocular visual odometry using vanishing points has been proposed by Gräter et al (Gräter, Schwarze and Lauer 2015). The vanishing points are utilized to estimate the orientation of the ground plane, while the ground points from SfM are used to define the ground plane in a RANSAC-based style. The final scale is computed using a comparison to the a-priori information of the camera height over ground. It shows accurate results for urban scenes where buildings and rectangular-like shapes can be easily found. However, the approach assumes that there are sufficient line features to calculate the vanishing points. As a result, it cannot be applied for non-urban scenes, such as highway scenes.

Due to the challenging nature of estimating scale using only visual information, there have been efforts to combine camera and inertial measurement units (IMU). A loosely-coupled visual-inertial odometry was proposed by Weiss and Siegwart (Weiss and Siegwart 2011). In a more accurate but computationally more demanding version, tightly-coupled camera-IMU odometry was proposed by Leutenegger et al (Leutenegger, Lynen, Bosse, Siegwart and Furgale 2015). Recently, an extension of ORB-SLAM equipped with IMU was proposed in (Burusa 2017).

8.3 Approach of monocular scale estimation

For all further reasoning I use the camera coordinate system as reference where the z-axis is given by the viewing direction of the camera, the y-axis points towards the ground and the x-axis points to the right of the viewing direction.

The relative pose between two frames can be described using six parameters: three rotation parameters and three translation parameters. The angles θ, ψ and ϕ denote the rotation about the x-, y- and z-axis respectively in mathematically positive sense.

Additionally I choose to parametrize the translation in spherical coordinates by two angles α and β, and one length s. For notational purposes I combine all motion parameters in a vector $\boldsymbol{p} = (\alpha, \beta, s, \phi, \theta, \psi)^T$, as also used in (Bradler et al. 2015).

The absolute scale s is defined as the translation length between two consecutive frames. To visualize the workflow of my proposed algorithm figure 8.2 shows the flowchart of my monocular scale estimation framework. The scale estimation module consists of two steps: scale prediction and scale correction.

Our proposed method assumes that the unscaled pose is already available, i.e. five parameters of the relative pose are known (see chapter 7). I also assume that the correspondences at the new frame $n+1$ have been estimated and that they are conformant to the given epipolar geometry as this can be provided by using epipolar tracking. The world coordinates of the keypoints at the new frame $n+1$ are also assumed to be available (see chapter 6). Additionally, the camera height over ground d is assumed to be known.

8.3.1 Parametrization of the ground plane

Suppose that the ground plane normal vector at frame n is given by a unit vector $\hat{\boldsymbol{n}} = (\hat{n_1}, \hat{n_2}, \hat{n_3})^T$ in the camera coordinate system with $\|\hat{\boldsymbol{n}}\| = 1$. When the camera is in perfectly plane-parallel orientation, $\hat{\boldsymbol{n}} = (0, 1, 0)^T$. Suppose also that d is the distance from the camera center to the ground plane.

Hence, the ground plane can be expressed by $(\hat{\boldsymbol{n}}^T, d)^T = (\hat{n_1}, \hat{n_2}, \hat{n_3}, d)^T$, where a 3D point $\boldsymbol{p}$ lies on the ground plane if it satisfies

$$\hat{\boldsymbol{n}}^T \cdot \boldsymbol{p} - d = 0. \tag{8.1}$$

A homography matrix defines the transformation of points belonging to a plane from image points $\boldsymbol{x}$ at one frame to the corresponding image points at another frame. In this chapter, the transformation is described from frame $n+1$ to frame n.

$$\boldsymbol{x}_n = \mathbf{H} \cdot \boldsymbol{x}_{n+1} \tag{8.2}$$

Let $\mathbf{R}$ and $\boldsymbol{t}$ be the relative rotation matrix and the relative translation vector between the two frames. The homography matrix $\mathbf{H}$ is given by

$$\mathbf{H} = \mathbf{K} \left(\mathbf{R} - \frac{\boldsymbol{t}\hat{\boldsymbol{n}}^T}{d} \right) \mathbf{K}^{-1} \tag{8.3}$$

where $\mathbf{K}$ is the matrix of intrinsic camera parameters.

8.3.2 Scale prediction

I use two independent scale predictions, one from the *propagation-based scale estimator (PbSE)* which sometimes fails to provide prediction and the other one from the *statistical*

motion predictor (SMP) (Bradler et al. 2015) which is always able to provide a scale prediction and is employed only to fill the prediction gaps from PbSE data.

8.3.2.1 Statistical motion predictor (SMP)

The egomotion of a car is highly constrained and steady due to its large mass making the motion inert and due to the restrictions of steering, which can be described well by a coordinated turn model. Thus the egomotion of a car can be predicted well based on the previous relative poses by employing a statistical model (Bradler et al. 2015). This predictor employs linear regression to predict the next motion parameter vector $\boldsymbol{z}$, given the previous three poses.

Not only does the prediction provide an initial translation estimate but it also provides a rotation estimate on the rare cases in which the pitch-yaw-roll-estimator does not provide a reliable rotation estimate. This way, a reasonable estimate of the next relative pose and the underlying epipolar geometry can always be obtained. The detailed implementation of SMP in the proposed framework has been discussed earlier in section 7.3.2.

8.3.2.2 Propagation-based scale estimation (PbSE)

As discussed in chapter 7, the pose estimation method has already provided the unscaled pose (rotation matrix and translation unit vector) as well as the pixel and world position of the tracked points. Thus, there is one unknown parameter remaining, that is the absolute scale.

Given the above information, the problem of estimating the scale from one single tracked point is illustrated in figure 8.3. The world point $\boldsymbol{x}$ and the correspondences $\boldsymbol{m}_n$ and $\boldsymbol{m}_{n+1}$ are known. Additionally, the orientation of frame $n+1$ with respect to frame n was already fixed by the rotation matrix $\mathbf{R}$ and translation unit vector $\hat{\boldsymbol{t}}$. The problem of estimating the scale is equivalent to placing the camera center $\boldsymbol{c}_{n+1}$ along the red line so that line $\overline{\boldsymbol{x}\boldsymbol{c}_{n+1}}$ exactly crosses $\boldsymbol{m}_{n+1}$.

I only use keypoints with sufficiently large triangulation angles ($\alpha_t > 0.05°$) to estimate the scale in order to make sure that I have high quality triangulated points. It is important to note that the estimated scale using PbSE at frame $n+1$ depends on the scale at frame n. The reason behind this is that the world coordinate of the keypoint is previously computed using the scale on the previous time step. Hence, the final output of the PbSE is the relative scale $r_s^{(n)}$ which is defined as the ratio between the scales at two consecutive frames n and $n+1$

$$r_s^{(n)} = s^{(n+1)} / s^{(n)}. \tag{8.4}$$

Since the relative scale can be computed from each of the tracked keypoints, I choose the median value of the estimates from all tracked keypoints as the final PbSE relative

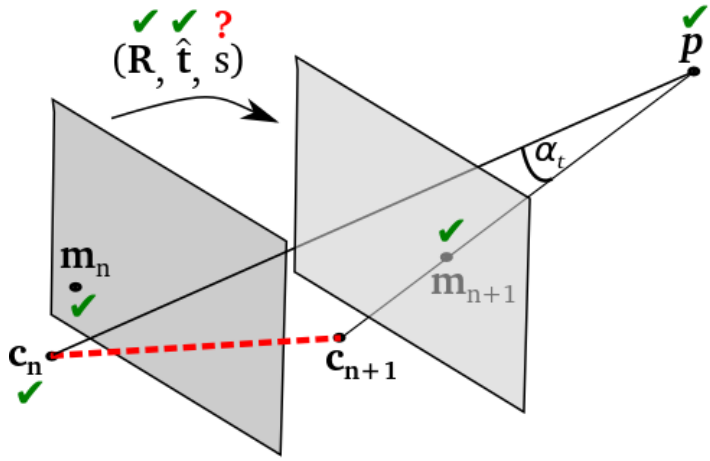

Figure 8.3: The illustration of solving the scale s using propagation-based scale estimation. The problem is equivalent to estimating the length of the red-dashed line segment $\overline{\boldsymbol{c}_n\boldsymbol{c}_{n+1}}$. α_t denotes the triangulation angle.

scale estimate. The tracked keypoints corresponding to significantly different relative scale estimates (more than three times the standard deviation) from the mean result are flagged as outliers.

8.3.3 Scale correction

Ground plane estimation from a single image sequence largely relies on the visible structure on the ground surface. The ground structure typically comes from the shadow of surrounding objects (buildings, trees, vehicles) as well as the structure of the road itself. However, there will not always be enough structure on the ground surface to start any approach of image alignment or keypoint matching. In such cases, it is beneficial to use the ground plane estimate from the previous frame.

Thus, I propose an image alignment algorithm to estimate the ground plane and the world scaling using knowledge from previous estimates. The proposed method is based on matching the ground plane at frame n to the frame $n+1$ via direct patch matching. An initial guess is based on an estimate of the relative pose of the vehicle from frame n to the frame $n+1$. It will then iteratively converge to the ground plane by minimizing the SSD of the presumably aligned patches while at the same time improving the estimate of the relative pose. Once I know the ground plane, the scaling of the world can be corrected by the already known height of the camera with respect to the ground plane.

The scale correction module consists of two methods for ground plane estimation (GPE): dense matching GPE and sparse matching GPE. The scale is initialized by the estimate from the scale prediction module by either the propagation-based tracking framework or – if no reliable value is available – by the statistical motion predictor (see section 7.3.2). The initial estimate of the ground plane is set at the normal vector $\tilde{\boldsymbol{n}}$ predicted

by propagating the previously estimated ground plane on the last frame.

8.3.3.1 Dense matching ground plane estimation (dGPE)

For dense matching, I first select all points $\{\boldsymbol{x}_i\}_{i=1,\dots,N}$, given in homogeneous coordinates, belonging to a measurement region in the current frame I_{n+1}. This region of interest is defined as a rectangular region (5 meters wide in x-direction and 8 meters long in z-direction) and it starts at 3.8 meters distance in front of the camera (as shown in Figure 8.1). I now aim to minimize the cost function Q with respect to the homography matrix $\mathbf{H}$:

$$Q(\mathbf{H}) = \sum_{i=1}^{N} \underbrace{\left(I_{n+1}(\boldsymbol{x}_i) - I_n(\mathbf{H}_{n+1} \cdot \boldsymbol{x}_i)\right)}_{=\, e(H_{n+1}, x_i)}{}^2. \tag{8.5}$$

But estimating the entries $h_{11}, \dots, h_{33}$ of the homography matrix itself will not directly result in the entities of interest, namely the motion parameters. Estimating those entries would mean that the homography matrix would have to be further decomposed.

Due to the sensitivity of noise during the homography decomposition step, I choose a different parametrization. The translation scale s is extracted out from the translation vector $\boldsymbol{t}$, and then the scale is estimated simultaneously with the unit vector $\hat{\boldsymbol{n}}$. Hence, the homography matrix (see equation 8.3) can be rewritten as

$$\mathbf{H} = \mathbf{K}\left(\mathbf{R} - \frac{\hat{\boldsymbol{t}} s \hat{\boldsymbol{n}}^T}{d}\right)\mathbf{K}^{-1} = \mathbf{K}\left(\mathbf{R} - \frac{\hat{\boldsymbol{t}} \boldsymbol{n}^T}{d}\right)\mathbf{K}^{-1} \tag{8.6}$$

where $\hat{\boldsymbol{t}}$ is the unit translation vector and $\boldsymbol{n} = (n_1, n_2, n_2)^T$ is the normal vector with the length of s.

By directly infusing the homography composition where the rotation matrix $\mathbf{R}$ as well as the translation unit vector $\hat{\boldsymbol{t}}$ are parametrized by a vector $\boldsymbol{p} = (\alpha, \beta, \phi, \theta, \psi)^T$, I obtain the following modified minimization:

$$\underbrace{\sum_{i=1}^{N} \left(I_{n+1}(\boldsymbol{x}_i) - I_n(\mathbf{H}_{n+1}(\boldsymbol{p}, \boldsymbol{n})(\boldsymbol{x}_i)\right)^2}_{Q(\boldsymbol{p}, \boldsymbol{n})} \quad \longrightarrow \min \tag{8.7}$$

Note that in this loss function the constraint that $\boldsymbol{n}$ is a unit vector was removed. This way the scale s will be easily extractable as the absolute value of the estimated normal vector, thus

$$s = \|\boldsymbol{n}\|. \tag{8.8}$$

Since this optimization problem cannot be solved directly in closed form, the SSD has to be minimized iteratively. This set-up is similar to the Lucas-Kanade algorithm and in the same way, I take advantage of the structure of the loss function to achieve a faster optimization process as has been discussed in (Baker and Matthews 2004).

The chain of iterative steps to find a good estimate is equivalent to finding a series of parameters $(\mathbf{R}_j, \hat{\boldsymbol{t}}_j, \boldsymbol{n}_j)_{j=1,2,\dots}$ that converge towards a matrix $\tilde{\mathbf{H}}$ such that the SSD converges to a minimum. To solve this minimization problem, an iterative Gauss-Newton algorithm is used. This requires the calculation of the Jacobian of $\mathbf{H}$ with respect to the optimization parameters, namely $\alpha, \beta, \phi, \theta, \psi, n_1, n_2, n_3$, which were derived using Mathematica software.

To numerically compute the optimization, the loss function is further modified by introducing a linearization of the image function I as I do not know directly in which way the image function itself changes when the normal vector $\boldsymbol{n}$ is modified. Thus, I use the first order term of its Taylor expansion.

Since I already have a good estimate of the unscaled pose and thus $\mathbf{R}$ and $\hat{\boldsymbol{t}}$ are available, I divide the optimization into two steps:

1. Optimization for the ground plane normal vector only $(\boldsymbol{n})$

$$\boldsymbol{n}_{min} = \arg\min_{\boldsymbol{n}} \underbrace{\sum_{i=1}^{N} \left(I_{n+1}(\boldsymbol{x}_i) - I_n(\mathbf{H}_{n+1}(\boldsymbol{p}, \boldsymbol{n})(\boldsymbol{x}_i)\right)^2}_{Q(\boldsymbol{p},\boldsymbol{n})} \tag{8.9}$$

2. Optimization for pose and ground plane $(\alpha, \beta, \phi, \theta, \psi, \boldsymbol{n})$

$$\begin{pmatrix} \boldsymbol{p}_{min}, \\ \boldsymbol{n}_{min} \end{pmatrix} = \arg\min_{\boldsymbol{p},\boldsymbol{n}} \underbrace{\sum_{i=1}^{N} \left(I_{n+1}(\boldsymbol{x}_i) - I_n(\mathbf{H}_{n+1}(\boldsymbol{p}, \boldsymbol{n})(\boldsymbol{x}_i)\right)^2}_{Q(\boldsymbol{p},\boldsymbol{n})} \longrightarrow \min \tag{8.10}$$

where the normal vector which was estimated in the first step is used as an initialization for the second step.

The final estimated scale from dGPE is then denoted as s_d.

$$s_d = \|\boldsymbol{n}_{min}\|. \tag{8.11}$$

8.3.3.2 Sparse ground plane estimation (sGPE)

Instead of using all pixels as in the dGPE case, the sparse ground plane estimation (sGPE) utilizes only a number of ground points to estimate the ground plane. The candidates for ground points are chosen among the tracked keypoints from the propagation-based tracking (PbT) framework.

There are two criteria for a tracked point to be selected as a ground point. First, the pixel coordinate of the point must be inside the sGPE RoI with the rectangular size of 10×26.2 meter starting at 3.8 meters distance in front of the camera, which is significantly larger than the dGPE RoI as can be seen in figure 8.1.

Second, the world coordinate of the point must not have height more than a threshold τ_{hg} to the predicted ground plane. Additionally, tracked points which were previously labelled as ground points remain in the set of ground points even though they have left the RoI as long as they are still tracked and visible in the image.

The best fitted ground plane is then selected based on the minimum squared distance from the chosen ground points. With the prior knowledge of the camera height, it is then straightforward to compute the estimated scale for scale correction. The final estimated scale from sGPE is then denoted as s_s.

8.3.4 Scale outlier rejection using classification

At this point in the processing chain, I have scale estimates from several methods: dGPE, sGPE, and PbSE. However, all of these estimates come with significant errors at times, which makes it essential to detect and reject those scale outliers, and in the extreme case bridge the gap until the next valid scale measurement by scale propagation. Errors typically occur when the assumption of a planar road surface being inside the RoI does not apply.

Instead of assigning fixed weights to the estimates from different methods throughout all frames, I need to be able to assess the quality of each individual scale estimate per frame. Thus, I need to analyze each estimation method and define features that can be utilized to determine the validity of each resulting scale estimate. These features characterize properties of the input data (e.g. existence of sufficient textural structure in the RoI image signal) and 'side products' of the estimation process, such as e.g. matching residuals.

For this purpose I employ classifiers that employ a set of hand-designed features to detect inliers and outliers among the scale estimates provided by each method (PbSE, dGPE, sGPE). The features for each method are selected and listed below.

- **Propagation-based scale estimation (PbSE)**:
  Three features, namely f_{A1}, f_{A2}, f_{A3}, are selected:
  1. the number of tracked keypoints,
  $$f_{A1} = n_A \tag{8.12}$$
  2. the standard deviation δ_r of the relative scale estimates r_s from all tracked keypoints,
  $$f_{A2} = \delta_r \tag{8.13}$$
  3. the percentage of inliers among tracked keypoints, where the detection of outliers has been explained in section 8.3.2.2.
  $$f_{A3} = \frac{n_{in}}{n_A} \tag{8.14}$$
  where n_{in} is the total number of points classified as inliers.

- **Dense ground plane estimation (dGPE)**:
  Suppose that g_i is the magnitude of the gradient vector $\boldsymbol{g}_i$ at pixel i

$$g_i = \|\boldsymbol{g}_i\|. \tag{8.15}$$

  Two features from dGPE, namely f_{B1} and f_{B2}, are chosen for the classification:

  1. the ratio between the sum of the magnitude g_i of each gradient inside the RoI and the sum of the squared difference (SSD) of the matched region denoted as q

$$q = \sum_{i \in RoI} \left(I_{n+1}(\boldsymbol{x}_i) - I_n(\mathbf{H} \cdot \boldsymbol{x}_i)\right)^2 \tag{8.16}$$

$$f_{B1} = \frac{\sum_{i \in RoI} g_i}{q}, \tag{8.17}$$

  2. the angular deviation (in degree) of the estimated ground plane normal vector $\boldsymbol{n}$ from the initial predicted normal vector $\tilde{\boldsymbol{n}}$

$$f_{B2} = \arccos\left(\frac{\boldsymbol{n}^T \cdot \tilde{\boldsymbol{n}}}{\|\boldsymbol{n}\| \, \|\tilde{\boldsymbol{n}}\|}\right) \tag{8.18}$$

- **Sparse ground plane estimation (sGPE)**:
  There are three features, namely f_{C1}, f_{C2}, f_{C3}, selected for sGPE:

  1. the number of the tracked ground keypoints,

$$f_{C1} = n_C \tag{8.19}$$

  2. the mean of the triangulation angles α_t of the tracked ground keypoints,

$$f_{C2} = \frac{\sum \alpha_t}{n_C} \tag{8.20}$$

  (3) the angular deviation (in degrees) of the estimated ground plane normal vector $\boldsymbol{n}$ from the predicted normal vector $\tilde{\boldsymbol{n}}$.

$$f_{C3} = \arccos\left(\frac{\boldsymbol{n}^T \cdot \hat{\boldsymbol{n}}}{\|\boldsymbol{n}\| \, \|\hat{\boldsymbol{n}}\|}\right) \tag{8.21}$$

I use the support vector machine module from the Scikit-Learn package (Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, Blondel, Prettenhofer, Weiss, Dubourg, Vanderplas, Passos, Cournapeau, Brucher, Perrot and Duchesnay 2011) to build classifiers using the above mentioned selected features. I specifically employ C-Support Vector Classification with a radial basis function (RBF) kernel and balanced class weight. I define two classes and label them as *inlier* and *outlier*. The training data are obtained by comparing the raw scale estimation results to the ground truth data, and subsequently I label them as *inlier* only when the estimation error is below 0.1 meter/frame or equivalent to ±3.6 km/h.

I train the classifiers using the KITTI training dataset (Geiger et al. 2013) (sequence 00-10) which provides a vast collection of real driving scenarios. I employ a leave-one-out scheme when assigning the available sequences for learning and testing, e.g. I use data from sequence 00-09 to train the classifiers when testing sequence 10.

It is important to note that obvious scale outliers that can be identified earlier need not to be tested further as they have already been excluded earlier, leaving only reasonable scale estimates to be checked by the classifiers. Specifically, the scale estimation from dGPE is straightly flagged as outliers if the mean of the squared grayscale difference between the final matched pixels is more than 0.4. Scale estimation results are also rejected if the final scale estimate is more than equivalently 300 km/h. Additionally, the angular deviation of the estimated normal vector from its nominal value $(0, 1, 0)^T$ must not exceed 20° for both dGPE and sGPE.

8.3.5 Fusion and scale correction

Only scale estimates which pass the classifier check are considered for the final scale estimation. The estimates of the scale correction (see section 8.3.3) are taken into accounts only when both estimates from dGPE and sGPE agree to each other (maximum difference is set to 0.1 meter/frame). The mean value between both estimates from dGPE s_d and sGPE s_s is then chosen as the scale correction value $s_{corr} = \frac{s_d + s_s}{2}$.

To ensure scale consistency and to prevent that wrong scale corrections are taken into account, the scale correction is only activated when it is consistent with at least one more scale correction from two most adjacent scale correction candidates. The scale fusion is formulated as follows: the final scale is determined by propagating the predicted scale (see section 8.3.2) as long as a scale correction is not executed. Hence, given two consecutive frames n and $n+1$, the scale estimate at frame $n+1$ is computed as follows

$$s^{(k+1)} = \begin{cases} s_{corr}^{(k+1)}, & \text{if the correction is executed} \\ s^{(k)} \cdot r_s^{(k)}, & \text{otherwise} \end{cases} \tag{8.22}$$

8.3.6 Street pixel labeling using a CNN

To obtain a robust scale estimation which is not influenced by objects directly in front of the vehicle, our VSI group developed a method based on a CNN for detecting and segmenting the road area in each image. The CNN come from the work of a colleague at the VSI group and I only use the resulting CNN masks computed by the implementations. Specifically, I utilize the resulting street pixel label for the work reported in this chapter. The pixelwise labeling of the actual street area by the CNN yields more flexibility to use a wider RoI for the dense GPE as well as a larger set of keypoints for the sparse GPE, as non-ground pixels are reliably excluded. The intersection of the RoI of GPE and the street mask constitutes the final area to be processed for each method, as shown in figure 8.4. We can consider the scale estimate to

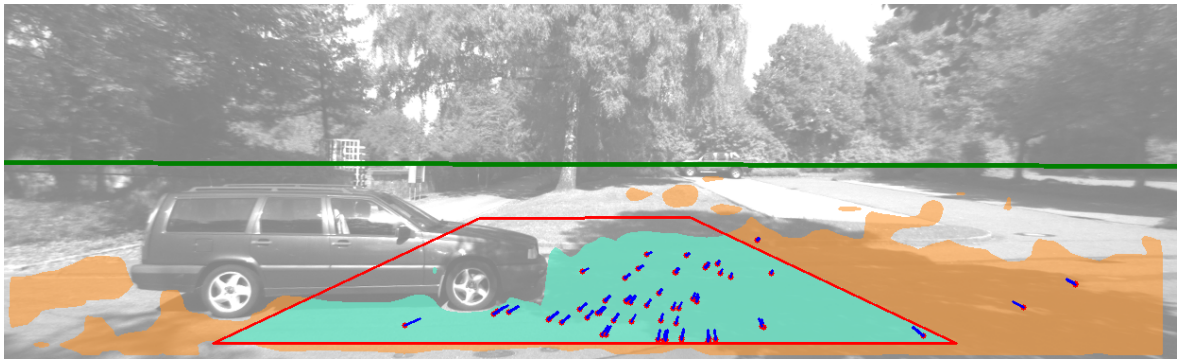

Figure 8.4: The initial RoI for dGPE is marked by the red (inner) rectangular area, while the RoI for sGPE is the complete region below the green horizon line. It is intersected with the blue (larger rectangular) street mask to make sure that the final RoI contains only street pixels. *Best viewed in color.*

be valid only when it fulfils certain checks. The scale estimate s must be smaller than it would be at a speed of τ_v and the ground plane normal vector must have less than a τ_a angle difference w.r.t. the vertical axis. Additionally, the mean square difference of pixel values in the final iteration of the dGPE must be less than τ_{ssd} when using 8-bit grayscale images.

8.4 Experiments on scale estimation

I test the proposed multimodal scale propagation/correction method on the KITTI dataset (Geiger et al. 2013). I specifically test on KITTI training sequences (Seq. 00-10) where the ground truth pose is available to allow a numeric evaluation of the final scale estimate precision. It is important to mention again that I utilize a leave-out-out scheme to train the classifiers, hence the classifiers are trained on all training sequences except the one being tested.

8.4.1 PbSE accuracy

I analyze the accuracy of the PbSE relative scale prediction. The PbSE relative scale is compared to the relative scale from the KITTI ground truth, as shown in figure 8.5, where relative scale describes the ratio between two consecutive scales as defined in equation 8.4.

We can observe that the relative scale obtained from PbSE is very accurate when compared to the ground truth data. As a result, the scale prediction alone, without the scale correction, is capable to follow the dynamics of the true scale change. This can be observed in figure 8.6 where the shape of the estimated scale after propagating the PbSE relative scale is similar to that of the ground truth scale. Even though the relative scale

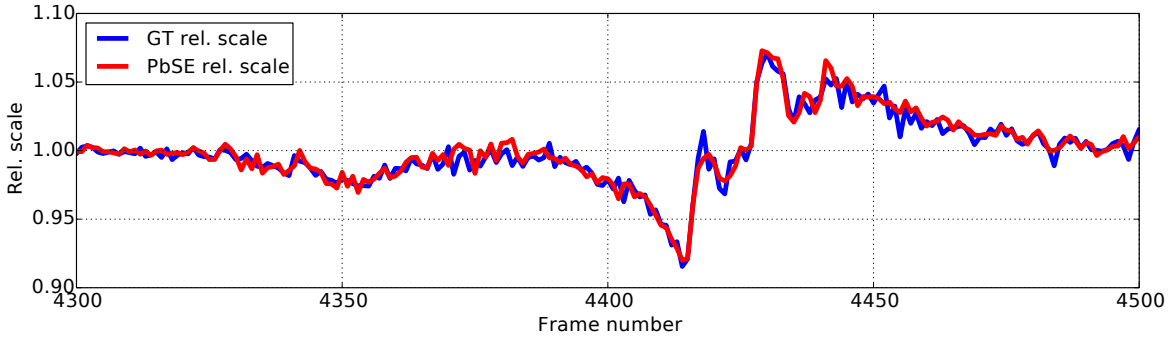

Figure 8.5: The relative scale estimates using PbSE (solid red) compared to the ground truth (dotted) on KITTI sequence-00. Relative scale is the ratio between the scales from two consecutive frames. *Best viewed in color.*

can be estimated by PbSE, the scale drift is still apparent without periodic corrections as shown in figure 8.6. That is why the scale needs to be periodically corrected.

8.4.2 Scale correction

The improvement of the estimated scale after the scale correction is shown in figure 8.6 in comparison to the ground truth data. The final estimated scale, which is composed by the available scale correction points while filling the gap using the propagated scale, show a high degree of agreement with the ground truth scale.

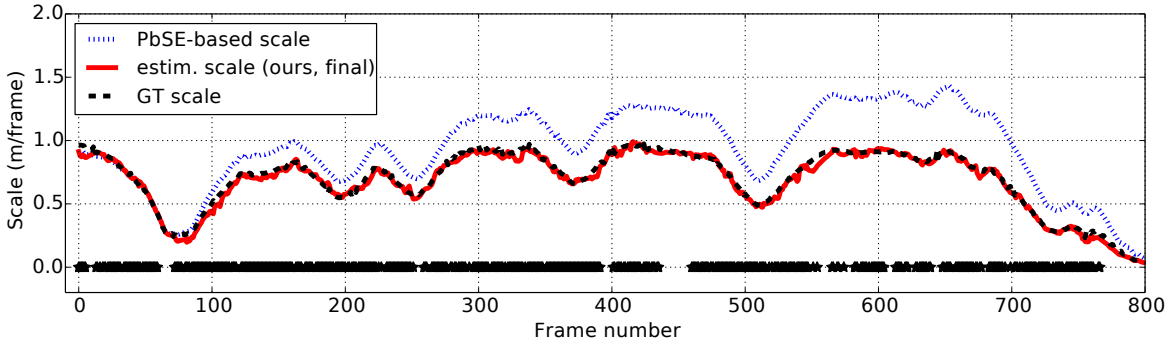

Figure 8.6: The scale estimates propagated using only PbSE (dotted line), plus scale corrections (solid line) as compared to the ground truth (dashed line) on KITTI sequence-03. The star symbol indicates the occasion when a scale correction is activated. *Best viewed in color.*

With the scale estimates from my proposed method, we can transform the unscaled pose from (Fanani et al. 2016) into a scaled pose. I plot the the estimated courses based on the estimated scaled pose in KITTI sequences as shown in figure 8.7.

I analyze the accuracy of my scale estimation result and compare it with state-of-the-art monocular scale estimation methods, specifically with MLM-SFM (Song, Chandraker

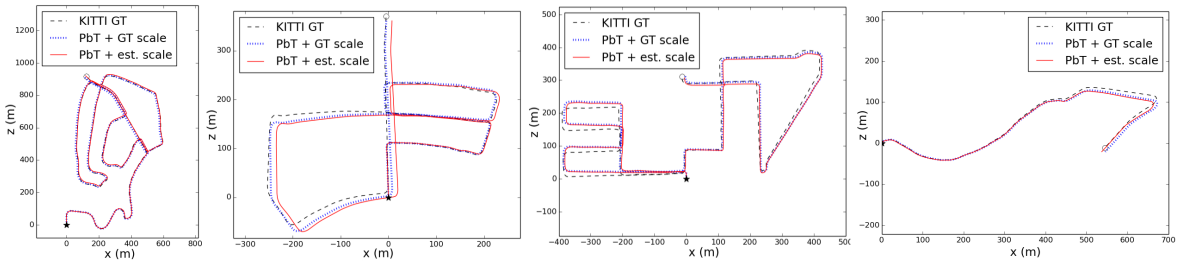

Figure 8.7: The estimated courses on KITTI sequence-02, sequence-05, sequence-08, and sequence-10 (*from left to right*). I compare my result (solid line) with the GT scale-injected result (dotted line) and KITTI ground truth (dashed line). *Best viewed in color.*

and Guest 2016) and FTMVO (Mirabdollah and Mertsching 2015).

Table 8.1 shows the comparison between my results and the results from MLM-SFM. MLM-SFM has a clear limitation as it cannot perform on 2 out of 11 KITTI training sequences due to repeated patterns and a big moving lorry. On the contrary, my proposed method is able to perform on all KITTI training sequences, including the challenging sequence-01 and sequence-07, as shown in figure 8.8. Additionally, my results are better on all 9 sequences which are available for comparison, as shown in table 8.1. The overall translation error of MLM-SFM is 2.03% while my translation error is significantly better at 1.54%.

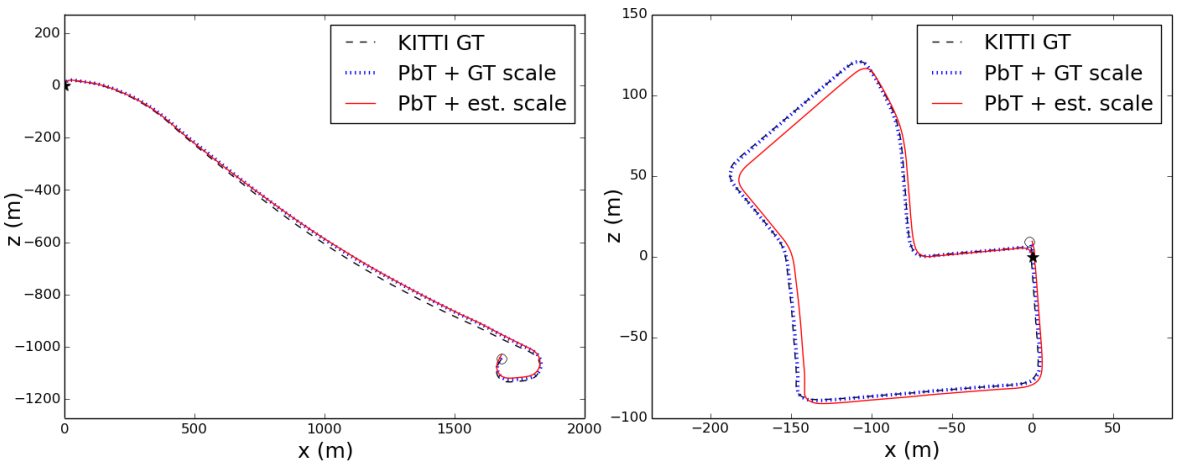

Figure 8.8: The estimated course on KITTI sequence-01 (left) and sequence-07 (right), where MLM-SFM fails to estimate. I compare my result (solid line) with the GT scale-injected result (dotted line) and KITTI ground truth (dashed line). *Best viewed in color.*

Table 8.1: Comparison of my results to MLM-SFM in terms of the translation error in KITTI dataset. Better results are written in bold numbers.

Seq	**Frames**	**Distance (m)**	**Translation error (%)**		
			PbT (Fanani et al. 2016) + GT scale	**PbT (Fanani et al. 2016) + our scale**	**MLM- SFM** (Song et al. 2016)
0	4541	3723.6	1.03	**1.65**	2.04
1	1101	2453.1	0.97	**1.92**	n.a.
2	4661	5067.0	0.89	**0.99**	1.50
3	801	560.9	0.82	**1.98**	3.37
4	271	393.6	0.50	**1.33**	1.43
5	2761	2205.6	0.95	**1.50**	2.19
6	1101	1232.9	1.12	**1.50**	2.09
7	1101	694.7	0.57	**2.37**	n.a.
8	4071	3222.8	1.22	**1.84**	2.37
9	1591	1705.1	0.94	**1.56**	1.76
10	1201	919.5	1.24	**1.83**	2.12
Avg			1.01	**1.54**	2.03

I also compare my results with FTMVO which previously held the best monocular-based visual odometry result in the KITTI benchmark (as of January 2017). Given the real camera position at frame n as $\boldsymbol{c}_n$ and the estimated camera position as $\hat{\boldsymbol{c}}_n$, FTMVO presents the scale accuracy level as the average distance error ϵ_c, which is defined by

$$\epsilon_c = \frac{1}{N_f} \sum_{n=0}^{N_f-1} \|\boldsymbol{c}_n - \hat{\boldsymbol{c}}_n\| \tag{8.23}$$

where N_f is the number of frames. The results from four KITTI training sequences are available in the FTMVO paper (Mirabdollah and Mertsching 2015). I obtain better results in 3 out of 4 available FTMVO results, as shown in table 8.2. Our overall distance error is 13.1 meters, significantly lower than FTMVO result at 21.3 meters.

Table 8.2: Comparison of my results to FTMVO in terms of the average distance error ϵ_c in KITTI dataset. Better results are written in bold numbers.

Seq		0	1	2	7	**Avg**
Avg. dist. error (m)	**FTMVO** (Mirabdollah and Mertsching 2015)	**10.4**	97.9	32.3	25.7	21.3
	Ours	18.7	**14.2**	**10.8**	**7.1**	**13.1**

8.5 Concluding remarks

I propose a method for monocular scale estimation to solve the scale ambiguity problem and the scale drift effect. I address the problem using a two-step mechanism: prediction and correction where I combine both dense and sparse ground plane estimation methods. The outliers of the scale estimates are detected using classifiers. Our experiments show that I achieve better results than the state-of-the-art methods in the challenging KITTI dataset. In the extension of this work, I aim to further improve the robustness of my method in more challenging scenes.

Chapter 9

Detection of Independently Moving Objects (IMO)

9.1 Introduction to IMO detection

This chapter presents a method for detecting independently moving objects (IMOs) from a monocular camera mounted on a moving car. While visual odometry relies on static environment to estimate the egomotion of the car, a typical traffic scene is usually a mixture of static environment (e.g. buildings, street, trees) and moving objects (e.g. cars, bicycles, pedestrians). Thus, it is important for a visual odometry to be able to detect and identify moving objects. One obvious benefit is to be able to exclude the moving objects from the egomotion estimation, while it also opens up more possibilities in other tasks such as for path navigation and collision avoidance.

As classical visual odometry methods rely strongly on the rigidity of the depicted environment through which a car moves, deviations from this rigidity (e.g. other cars, pedestrians, etc.) are detrimental to most classical methods of visual egomotion estimation or *Structure from Motion (SfM)*. Therefore, most of such methods have to be complemented by modifications that filter out measurements obtained from moving or non-rigid objects. I summarize such objects under the term *independently moving objects (IMOs)*.

As discussed in section 2.2.4, the frame-to-frame egomotion induces an *epipolar constraint* which all corresponding points in two images have to obey to. Points or areas which do not move conformant to the epipolar geometry are obviously candidates for belonging to independently moving objects. However, IMOs can also be *epipolar conformant*, when they move parallel to the camera motion (for a formal definition, see section 9.5.1).

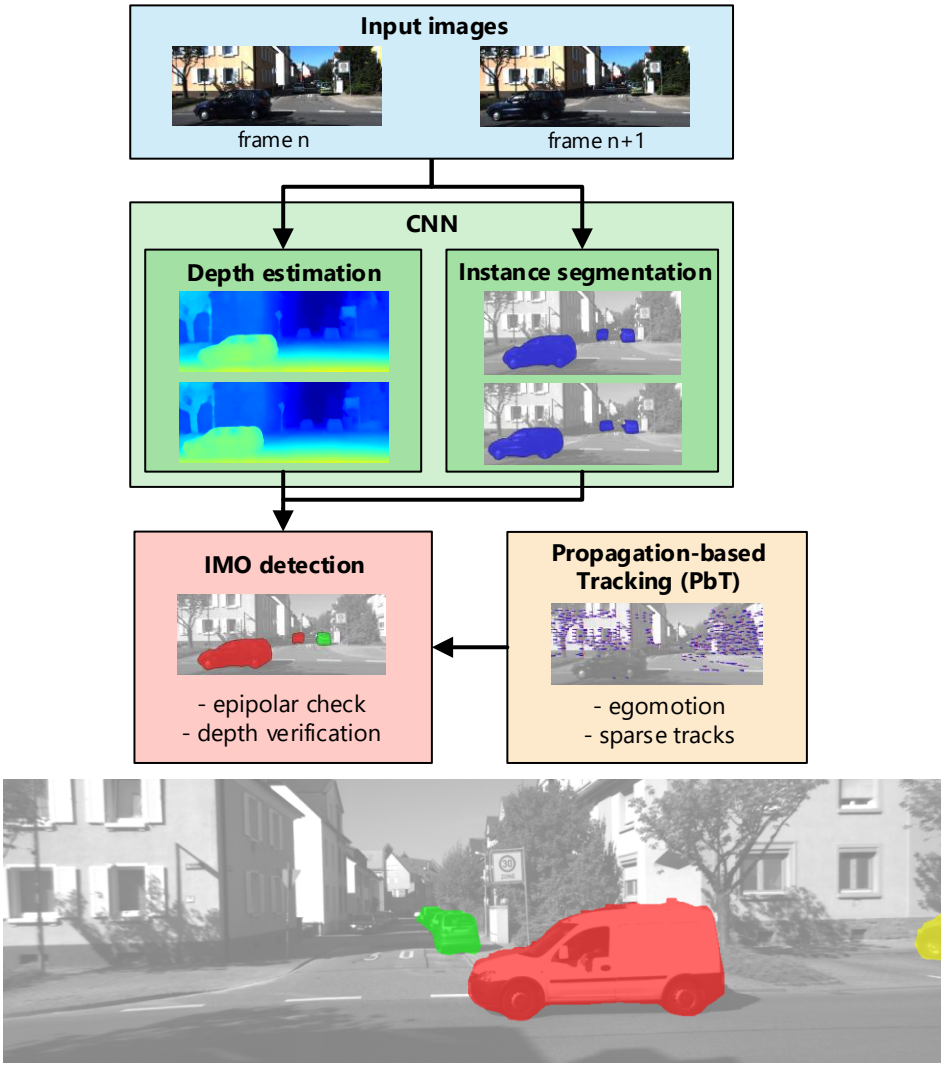

Figure 9.1: **Top**: The scheme of the the proposed method to identify IMOs. **Bottom**: An example of car classifications into static (green), IMO (red) and undetermined (yellow).

First, I propose a monocular IMO detection scheme which relies on multi-frame epipolar consistency checks. As a result, IMOs that are moving non-conformant to the epipolar structure (e.g. crossing IMOs) can be detected.

Then, this chapter also covers the more challenging problem of detecting even epipolar-conformant IMOs. I solve the inherently hard problem for the sparse monocular visual odometry: detecting moving objects which move parallel (or anti-parallel) to the camera motion, such as cars in the same or adjacent lanes, including oncoming traffic. Due to its epipolar consistency, a parallel-moving point visually appears exactly as an static point, but in a different (pseudo)distance.

The approach presented here is to employ two *convolutional neural networks (CNNs)*: one that provides a car instance segmentation (van den Brand et al. 2016), and another one that provides depth estimates for single monocular images (Ochs 2017). Those

CNNs come from the work of the VSI group and I only use the resulting CNN masks computed by the implementations. The car instance segmentation yields candidate patches that correspond to single vehicle instances, thus potential *independently moving objects (IMOs)*. In the presented scheme, these patches are subsequently associated with each other over time using a dynamic motion model and simple appearance descriptors. The presence of the depth map from the CNN makes it possible to compare distances obtained from geometric triangulation with such obtained from appearance, and thus supports the detection of epipolar-conformant IMOs also for the monocular case.

9.2 Related work on IMO detection

The detection of independently moving objects (IMOs) from visual sensors is a vitally important part of many computer vision systems. A traditional application can be found in visual surveillance, where the camera is static, but recently, object detection from moving cameras is becoming more influential.

In this work, I focus on the detection of moving cars from a vehicle mounted camera. This scenario is very different to others, for instance handheld cameras (such as (Kundu, Jawahar and Krishna 2010)) or general robot vision (such as (Jung and Sukhatme 2004)) because the vehicle motion is severely restricted.

In the development of other advanced driver assistance systems (ADAS), several approaches have been proposed. Many of those choose to work with additional information such as color images ((Brox, Bruhn and Weickert 2006), (Ošep, Mehner, Mathias and Leibe 2017)) or a stereo system (Wedel, Meißner, Rabe, Franke and Cremers 2009), (Lenz, Ziegler, Geiger and Roser 2011). In contrast to these approaches, I want to show that it is possible to reliably detect IMOs from a monocular camera only.

Previously published monocular algorithms on moving vehicle detection can be differentiated into two categories: appearance-based approaches (e.g. (Jung and Sukhatme 2004), (López-Rubio and López-Rubio 2015)) and motion-based approaches (e.g. (Yamaguchi, Kato and Ninomiya 2006), (Jazayeri, Cai, Zheng and Tuceryan 2011)).

I aim at providing a method that combines both approaches, in a way similar to (Ramirez, Ohn-Bar and Trivedi 2014), using the following cues to determine the presence of an independently moving object and to track it: a) the appearance of a car (in terms of a CNN-based car instance detector) as well as b) motion cues from sparse optical flow, considering the epipolar geometry. My approach shares some similarities with (Oliveira, Radwan, Burgard and Brox 2017) who use two separate CNNs to determine visual odometry and object localization and fuse their results to obtain object localizations. My method is also related to (Wulff, Sevilla-Lara and Black 2017) where CNNs are used to obtain a rigidity score for each object and this is combined with motion cues from optical flow. In contrast to their approach, I additionally use a series of hypothesis tests on keypoints lying on patches that have previously been identified as belonging to a vehicle in manner similar to (Wang, Thorpe, Thrun, Hebert and Durrant-Whyte 2007) to track the regarded vehicle through consecutive frames. Bai et al. (Bai, Luo, Kundu

and Urtasun 2016) estimate the dense optical flow fields from each IMO candidate using an approach similar to ours, by employing a CNN to provide the car region candidates. However, they focus only on obtaining the optical flow and do not identify whether the car patches are moving in 3D or not.

Crossing IMOs can be identified because the crossing motions induce inconsistency w.r.t. the epipolar geometry, as discussed in (Zhou, Frémont, Quost, Dai and Li 2017). However, parallel-moving IMOs are epipolar conformant. Klappstein et al. (Klappstein, Stein and Franke 2006) proposed a positive height and depth constraint but IMOs moving in opposite direction to the ego-car were only detected using a heuristic approach. Wong et al. (Wong, Siu, Jennings, Barnes and Fong 2015) utilized the size and contours of cars to detect parallel-moving IMOs.

9.3 Framework overview

The overall flow of the proposed method is presented in figure 9.1 (top image). I tackle the problem of IMO detection by classifying the IMOs into two categories: epipolar-conformant IMOs and non-epipolar-conformant IMOs. The keypoints found on the non-epipolar-conformant IMOs cannot be tracked by the PbT framework, because PbT restricts the matches to be along the epipolar lines. Not finding a photometric consistent match on or close to the epipolar line is thus the basis of labeling keypoints as *'cannot belong to static background'*. This fact serves as the basis of my strategy to detect non-epipolar-conformant IMOs. Failure to track a majority or even all keypoints on an IMO candidate indicates that the IMO candidate is highly likely an IMO.

An important principle of PbT is that the new relative pose for a new frame $n+1$ is predicted using the car ego-dynamics, and that this new pose is computed only on the basis of keypoints that have been tracked at least twice, that is: keypoints which already passed a stringent test of being belonging to the static environment. All keypoints, including the new ones generated in sparsely covered areas of a new frame, are tracked in an epipolar-guided manner as discussed more detailed in section 3.2.2.

In the present scheme, I detect IMO candidate patches for each new frame using the instance segmentation scheme of van den Brand et al. (van den Brand et al. 2016). This results in M new IMO candidate patches for each new frame $n+1$. The generation of the CNN-based IMO candidate patches is discussed in section 9.3.1.

All IMO candidate patches in image n are classified in one of the three states `static`, `IMO`, or `undetermined`. Keypoints that are located in IMO candidate patches are considered for pose estimation only if they have been classified as `static`.

When a new frame $n+1$ comes in, it is input to the CNN in order to determine the IMO candidate patches. At the same moment, I have a set of old tracked keypoints in previous frame n. Some of those are already confirmed as `static` as they have been tracked at least twice and have shown 3D consistency (see section 9.4.1). Others

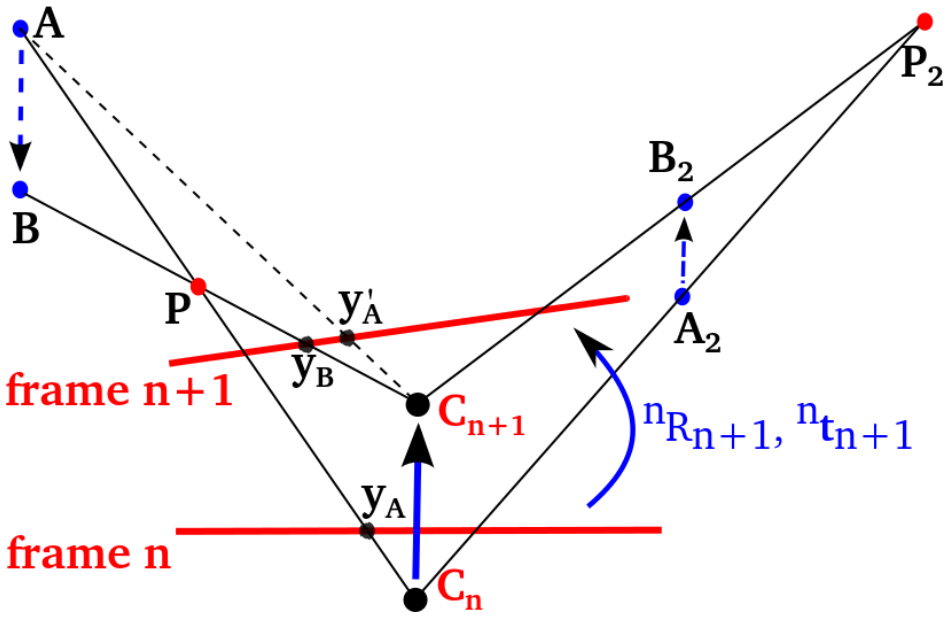

Figure 9.2: Bird-eye view. Parallel-moving point from A to B is visually identical to a static point P for the camera.

have been tracked just once (from frame $n-1$ to n) and are only candidates for being considered as static.

Subsequently, the relative pose between frame n and $n+1$ is determined from all keypoints which are considered to be static in frame n in the standard manner used in propagation based tracking. With the resulting relative pose $n \rightarrow n+1$ being computed, individual keypoints can then be checked for being conformant with the epipolar geometry. By accumulating the checks from all keypoints on a patch, this leads to the classification of that patch into `IMO`, `static`, or `undetermined`. This IMO detection procedure is described in detail in section 9.4.1 and section 9.4.2.

For each of the new IMO candidate patches in frame $n+1$, an association with the existing patches in frame n must be performed in order to propagate the motion state (`static`/`IMO`) over time. The association can be made *appearance-based*, that is: by comparing size and texture of the patches, or *tracking based*, that is: by checking matching residuals between keypoints inside of the patches. I determine the association between the IMO candidate patches by comparing the center of mass, the patch size (pixel count) and the gray value distribution. The inter-frame car patch association is discussed in detail in section 9.6.

Detecting epipolar-conformant IMOs, i.e. parallel-moving IMOs, is much more challenging. Monocular camera has an inherent limitation to identify objects moving parallel to the camera. Both static keypoints and parallel-moving keypoints can be tracked using epipolar-style PbT and they look exactly the same by the monocular camera as illustrated in figure 9.2. This means, a keypoint correspondence from a parallel-moving IMO could lead also to an ambiguous static point.

A CNN is employed to provide depth map estimates. With the depth map in hand, we can now detect epipolar-conformant IMOs using a depth verification scheme, consisting of the following two steps:

- Comparison of the depth information between triangulated depth by PbT and CNN depth map on the tracked keypoints observed on IMO candidates.
- Comparison of the relative depth difference extracted from two time-consecutive CNN depth maps of IMO candidates and the egomotion estimates from PbT.

In the present scheme, IMO candidate patches are detected for each new frame using the instance segmentation approach of van den Brand et al. (van den Brand et al. 2016). The generation of the CNN-based IMO candidate patches is discussed in section 9.3.1. In principle, any instance segmentation procedure which returns candidates patches for moving objects that provides reasonable quality can be used for this purpose, because my approach is not only limited to vehicles.

9.3.1 CNN-based IMO candidate patches

A CNN-based method for instance-level vehicle segmentation has been developed by our group, and presented in (van den Brand et al. 2016). I utilize the resulting IMO candidate patches obtained from the CNN-based instance-level vehicle segmentation based on the *'deep contours'* approach. The CNN has been trained to label cars using the Cityscapes dataset for training (Cordts, Omran, Ramos, Rehfeld, Enzweiler, Benenson, Franke, Roth and Schiele 2016). The output of the CNN are 5 layers: one for the car labels and four layers for the left, right, top, and bottom edges of cars. Initially, the car label image marks every pixel belonging to a car; therefore overlapping cars will be part of the same patch. By removing all edges from the car label image, the overlapping car instances are separated from one another. To successfully separate the instances, the contour of a car created by the edges must be closed and the edges are dilated to prevent small gaps. The car instance image is then created by labeling each independent patch as car instance.

9.3.2 CNN-based single-view depth estimation

CNNs have also been employed by our group to generate depth images from a given monocular image (Ochs 2017). For generating the appearance based dense depth maps, a CNN takes as input an RGB image and estimates the inverse depth $\rho(\boldsymbol{x}) = d(\boldsymbol{x})^{-1} \in [0, 1]$ for each pixel location $\boldsymbol{x} \in \Omega$. I utilize the resulting depth images in this work. The CNN-based depth images serve as the pseudo ground truth data during the depth verification step when testing the IMO candidates. They are particularly useful to identify parallel-moving IMOs.

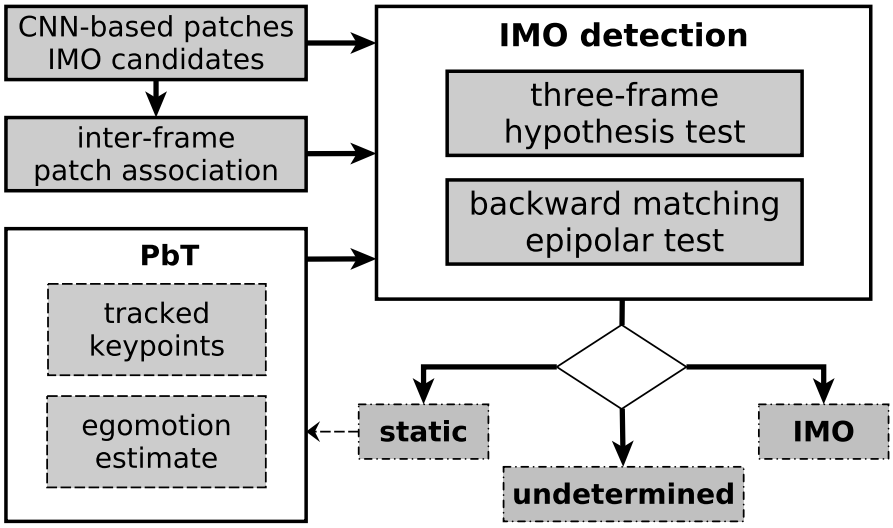

Figure 9.3: The scheme of the proposed IMO detection for non-epipolar-conformant case.

9.4 Detection of non-epipolar-conformant IMOs

The method presented here is based on three main steps: identifying any IMO candidates in the field of view, determining whether an object is moving or static, and finally tracking the movement of each object. As mentioned earlier, I combine cues from the appearance of the object itself with cues about object movement obtained from visual odometry. In the current presentation, I constrain myself to the detection and tracking of cars, although of course many classes of IMOs such as pedestrians, cyclists, animals, etc. could be detected and processed similarly.

I employ an IMO detection scheme which combines a *three-frame hypothesis test* (see section 9.4.1) and a *backward epipolar matching test* (see section 9.4.2) that decide for each keypoint inside the IMO candidate patch whether it is compliant to a static point in the environment through which the ego-car (and thus the camera) is moving. The results from the backward epipolar matching test are obtained instantly at the most recent frame while the results from the three-frame hypothesis test are only available two frames later. Hence, the three-frame hypothesis test serves as a correction step whenever there is a difference between the results from the two tests.

9.4.1 Three-frame hypothesis test

In some situations, particularly if objects are far away or if they are only moving slowly with respect to the static environment, it is difficult to determine that they move relative to the static scene from the observation of only two frames. Thus, I perform an IMO test using three consecutive frames.

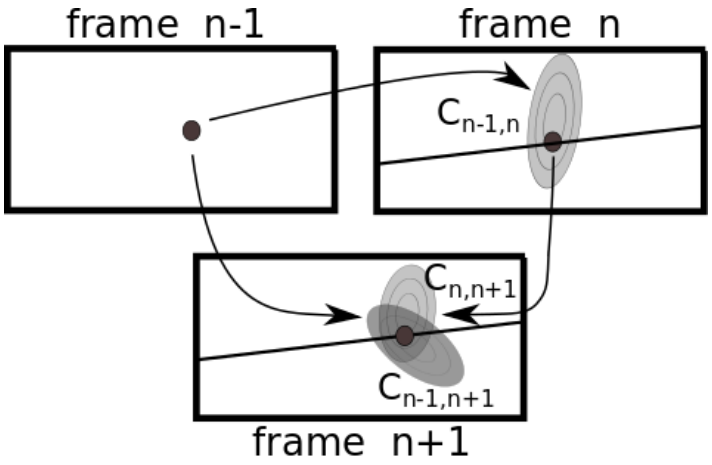

Figure 9.4: Illustration of IMO detection using three-frame hypothesis test. The epipolar constraints are shown by epipolar lines in frame n and $n+1$. The covariance matrix $\mathbf{C}$ is represented by an ellipse.

I begin the analysis of the found vehicle patches with the null hypothesis H_0 that supposes that a keypoint is static, i.e. that my two measurements matching with and without an epipolar constraint will yield the same result. I test this null hypothesis using a test statistic s which I compute from the observations, i.e. the Mahalanobis distance of the results of constrained and unconstrained matching, using the covariance matrix of the unconstrained matching. The matching model predicts which distribution the test statistic s should have if the hypothesis H_0 is true.

The test statistic computed from the measured data is an indicator number that should have a very different distribution for the two hypotheses that are taken into consideration. For instance, the test statistic s should be small if hypothesis H_0 is true, and large, if it is not true. Then I reject H_0 if I find that s takes a large value.

9.4.1.1 Testing single trajectories for belonging to an IMO

Let us regard testing a single trajectory obtained from using the propagation-based tracking method (Fanani et al. 2016). I will consider a joint test for multiple trajectories associated with the same IMO candidate patch later in section 9.4.1.5.

We have to discriminate two cases:

1. the trajectory is epipolar consistent by construction, e.g. since it is the result of epipolar matching and tracking (Fanani et al. 2016).
2. the trajectory is not constrained to the epipolar geometry induced by the sequence of poses associated with the trajectory.

The second variant is much harder to implement since I would need general long displacement matching to track general IMO motion. Hence, I focus on the hypothesis of static keypoints and consider a keypoint as belonging to an IMO whenever the

hypothesis is rejected.

9.4.1.2 Finding the best matches with and without epipolar constraint

A basic task appearing over and over again in the presented method is to match a keypoint $\boldsymbol{m}_{n-1}$ from frame $n-1$ to frame n. This matching is performed by a photometric comparison of the contents of the corresponding keypoint patches of size 15×15 pixels at times n and $n+1$, considering the weight mask $\mathcal{W}$ (see section 3.2.2). One of these patches is kept fixed (reference patch), whereas the position of the other one is varied in order to find an optimum match. This matching can be performed with or without consideration of the given epipolar relation induced by the relative pose between frame $n-1 \rightarrow n$. Let us denote the best matches with and without epipolar constraint as $\boldsymbol{m}_n$ and $\tilde{\boldsymbol{m}}_n$, respectively.

The classical Lucas-Kanade (LK) matching optimization problem can be written as a sequence of local quadratic optimization steps, where in each step the weighted sum of squared differences (WSSD) Q measured for patch (and keypoint) positions $\boldsymbol{x}_1, \boldsymbol{x}_2$ is minimized by small shifts $\boldsymbol{\Delta}_v$ from an initial position $\boldsymbol{x}_2 \mapsto \boldsymbol{x}_2 + \boldsymbol{\Delta}_v$.

$$Q(\boldsymbol{\Delta}_v) = \boldsymbol{\Delta}_v^T \cdot \mathbf{A} \cdot \boldsymbol{\Delta}_v + \boldsymbol{b}^T \cdot \boldsymbol{\Delta}_v + c \tag{9.1}$$

$\mathbf{A}$ is a symmetric 2×2 matrix built from the outer product of the gradient vectors around the moving patch, $\boldsymbol{b}$ is a 2×1 vector from the multiplication of gradient vectors and photometric error, and c is the sum of squared pixelwise photometric errors inside the patch. The final estimate of the displacement vector $\boldsymbol{v}$ after N iterations is given by

$$\boldsymbol{v} = \boldsymbol{x}_2(0) - \boldsymbol{x}_1 + \sum_{i=1}^{N} \boldsymbol{\Delta}_v(i), \tag{9.2}$$

For tracking under an epipolar constraint, I add the epipolar constraint

$$\boldsymbol{x}_{2,h}^T \cdot \mathbf{F} \cdot \boldsymbol{x}_{1,h} = 0 \tag{9.3}$$

to the quadratic optimization function Eq. 9.1, where $\boldsymbol{x}_{i,h} = [\boldsymbol{x}_i^T, 1]^T$. This problem is solved with a Lagrange multiplier α, thus yielding Eq. 9.4 which leads to a closed form solution of $\boldsymbol{\Delta}_v$.

$$\begin{pmatrix} \mathbf{A} & \mathbf{F}' \cdot \boldsymbol{x}_{1,h} \\ (\mathbf{F}' \cdot \boldsymbol{x}_{1,h})^T & 0 \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{\Delta}_v \\ \alpha \end{pmatrix} = \begin{pmatrix} -\boldsymbol{b} \\ -(\boldsymbol{x}_{1,h}^T \cdot \mathbf{F} \cdot \boldsymbol{x}_{1,h}) \end{pmatrix} \tag{9.4}$$

where $\mathbf{F}'$ corresponds to the first two rows of the $\mathbf{F}$ matrix.

9.4.1.3 Obtaining the spatial covariance matrix of a keypoint match

The advantage of employing photometric matching is that the determination of a covariance matrix expressing the spatial uncertainty of the match in the final position of the sliding patch can be immediately obtained from the linear equation system (9.1).

Let $\mathbf{C}_{m_n}$ be the covariance matrix expressing the uncertainty level of choosing $\boldsymbol{m}_n$ as the final match. The covariance of the final match $\mathbf{C}_{m_n}$ is given by matrix $\mathbf{A}$ from the last iteration of the LK matching optimization problem (see equation 9.1).

As a result, we have *two* entities expressing the (un)certainty of the given match:

- the final matching residual Q
- the curvature of the residual function, expressed by the covariance matrix

This information can be used in different ways:

- the final matching residual Q should be below a predetermined threshold
- the uncertainty of the position, as expressed by the covariance matrix, should be low enough

I perform an epipolar constrained search first, and allow an unconstrained post-optimization, starting from the found optimum on the epipolar line. The two resulting positions are then compared:

- if the matching residual q_1 of the constrained position is significantly higher than the residual q_2 of the pseudo-unconstrained[1] search, this indicates that the true match probably does not lie on the epipolar line.
- If the unconstrained match position is farther away from the constrained match position than suggested by the spatial covariance matrix, this is considered as an indicator that the true match is not on the epipolar line.

9.4.1.4 Computing the test statistic

Let us define a test statistic s which reflects how well the keypoint match in frame n follows the epipolar constraint. Let $\boldsymbol{w} = (w_x, w_y)$ be the distance vector between $\boldsymbol{m}_n$ and $\tilde{\boldsymbol{m}}_n$, where $\boldsymbol{m}_n$ and $\tilde{\boldsymbol{m}}_n$ denote the best matches with and without epipolar constraint.

$$\boldsymbol{w}_n = \tilde{\boldsymbol{m}}_n - \boldsymbol{m}_n \tag{9.5}$$

The metric s I use for measuring the distance between $\tilde{\boldsymbol{m}}_n$ and $\boldsymbol{m}_n$ is the Mahalanobis distance

$$s = \boldsymbol{w}_n^T \cdot \mathbf{C}_{m_n}^{-1} \cdot \boldsymbol{w}_n \tag{9.6}$$

As the inverse covariance matrix in this 2nd order form decorrelates and 'whitens' the elements of the random vector, the test statistic s is known to be χ^2-distributed, and we can define a threshold for s above which the difference between $\tilde{\boldsymbol{m}}_n$ and $\boldsymbol{m}_n$ is considered to be too high. The value of the test statistic s can alternatively be transformed into a 'p-level' p_n using the underlying χ^2 distribution.

[1] I denote this optimization as *pseudo-unconstrained* since it is not a true wide area search, but only a search in the vicinity of the constrained match position.

Since I assume all keypoints are IMO candidates in the beginning, a keypoint is only considered as a static object if the Mahalanobis s does not exceed a threshold τ_{ps} between the three consecutive frames. We can now decide whether the IMO candidate points are actually static, that is when they pass all of the following three checks:

$$s_{n-2,n-1} \underset{\text{static}}{\overset{\text{IMO}}{\gtrless}} \tau_{ps} \tag{9.7}$$

$$s_{n-1,n} \underset{\text{static}}{\overset{\text{IMO}}{\gtrless}} \tau_{ps} \tag{9.8}$$

$$s_{n-2,n} \underset{\text{static}}{\overset{\text{IMO}}{\gtrless}} \tau_{ps} \tag{9.9}$$

9.4.1.5 Testing sets of trajectories in the same IMO candidate patch

Let n_0 and n_1 be the number of keypoints on an IMO candidate patch which support the null hypothesis H_0 (= non-IMO) or the hypothesis $H_1 \rightarrow IMO$. The IMO candidate patch is classified as static if $n_0/(n_0 + n_1) > \tau_h$ and as IMO if $n_1/(n_0 + n_1) > \tau_h$.

9.4.2 Backward epipolar matching check

On top of the three frame hypothesis test from section 9.4.1, I also employ a backwards epipolar matching check to identify IMOs. There are several benefits in doing the epipolar matching check in backwards mode:

- I already have the IMO/static label of cars in the previous frame
- There is no delay in identifying IMOs since no future information is required.
- In a typical forward ego-motion, most keypoints in frame $n + 1$ are also visible in frame n, but not the other way around due to objects leaving the camera view from frame n to $n + 1$. Hence the chance of finding matches is higher when I match keypoints from frame $n + 1$ to frame n (backward mode).

Let us assume that we have obtained all information up to frame n. When frame $n + 1$ comes with its instance-level IMO segmentation, then we analyze each IMO candidate patch at this frame $n + 1$.

First, I generate keypoints on the new found car patches in frame $n + 1$, using the keypoint definition described in section 3.2.2. Second, I try to find the matches of these newly generated keypoints in frame n under consideration of the epipolar constraint given by the relative pose computed by PbT, as illustrated by figure 9.5.

Let $k_{i,init}$ be the number of keypoints generated on an IMO candidate patch i in frame $n + 1$ and let $k_{i,matched}$ be the number of successfully matched keypoints among the

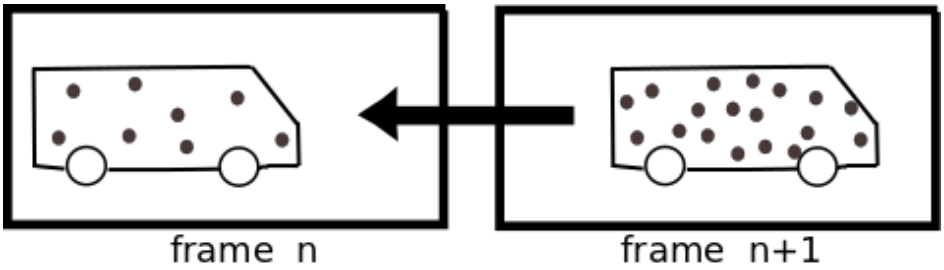

Figure 9.5: Backward epipolar matching from frame $n+1$ to frame n. The ratio of successfully found matches in frame n versus the number of initial keypoints in frame $n+1$ is computed.

keypoints in patch i. I define r_i as the ratio of successful epipolar matching of patch i

$$r_i = \frac{k_{i,matched}}{k_{i,init}} \tag{9.10}$$

A high ratio of the epipolar matching ratio r_i reflects the epipolar conformity of the car patch i, and thus indicates a static object. On the other hand, a low value of r_i gives a strong hint that the IMO candidate patch does not follow the epipolar structure. The decision is made by considering a patch as backwards epipolar consistent (non-IMO) if $r_i > \tau_i$ with an empirically chosen threshold τ_i.

An exemplary result from the KITTI training dataset sequence 07 is shown in figure 9.6. Keypoint matches are successfully found on the static car because all keypoints are conformant to the epipolar constraint. On the other hand, the epipolar matching cannot find keypoint matches for the moving car.

9.5 Detection of epipolar-conformant IMOs

In the monocular case, assuming rigidity for the complete set of points, epipolar-conformant point sets on moving objects will be assigned wrong distance values. Therefore, if we have some information about the depth of a candidate point set, we can design a test on conformity to the static background. The depth map which is needed as side information for this purpose is provided by CNN (Ochs 2017). I detect an epipolar-conformant IMO by showing that the depth of the IMO candidate car, as provided by the depth map, would not fit to the predicted depth calculated with the assumption that the car is static.

The triangulated depth of the target car, with the assumption that the car is static, can be provided by the PbT framework as long as there are some keypoint correspondences on the target car patch. However on some occasions, such as in a fast highway scene where long displacement occurs, no matched keypoint is available on the target car patch. Without tracked keypoints, no triangulation can be done, hence there is no depth prediction.

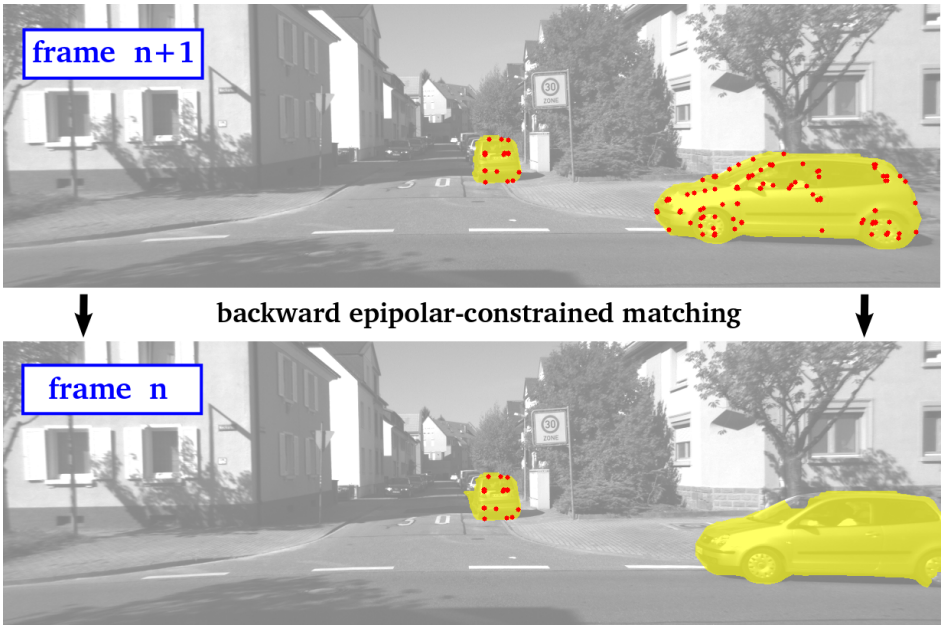

Figure 9.6: An exemplary result of the backward epipolar matching in the KITTI dataset sequence 07. Keypoints are first generated inside the car masks in frame $n+1$ (top image) before searching for the corresponding matches in frame n (bottom image) while imposing the epipolar constraint. The result shows that the keypoints on the left car found the corresponding matches in frame n which indicates that the car is static. On the other hand, there is no matched keypoint found on the right car in frame n which indicates that the car is an IMO.

In order to handle the case when there are no tracked keypoints on a car patch, the predicted depth of the car is obtained from the depth information on the previous frame, and then shifted by the estimated egomotion of the ego-car. I name the above two approaches as keypoint-based and keypoint-free depth verifications.

In this section, first I will prove that parallel-moving objects are consistent to the epipolar constraint. Second, I show that the speed ratio of a parallel-moving point w.r.t. the ego-car speed directly determines the depth of the triangulated ambiguous static point. Then I explain the keypoint-based and keypoint-free depth verifications to detect parallel-moving IMOs.

I assume that the egomotion estimates, more precisely: the relative pose between frames n and $n+1$, are already provided by PbT. I denote $\mathbf{R}$ and $\boldsymbol{t}$ as the relative rotation and relative translation to transform a fixed point $\boldsymbol{z}$ in the world of the camera coordinate system at frame n (CCS_n) to frame $n+1$ (CCS_{n+1}),

$$\boldsymbol{z}_{n+1} = \begin{pmatrix} {}^{n}\mathbf{R}_{n+1} & {}^{n}\boldsymbol{t}_{n+1} \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{z}_n \\ 1 \end{pmatrix} = {}^{n}\mathbf{R}_{n+1} \cdot \boldsymbol{z}_n + {}^{n}\boldsymbol{t}_{n+1} \tag{9.11}$$

9.5.1 Proof that a parallel-moving keypoint is epipolar-conformant

I refer to figure 9.2. Let $\boldsymbol{z}_{A(n)}$ be the 3D coordinate of a position A in CCS_n. The corresponding 3D coordinate in CCS_{n+1} is denoted by $\boldsymbol{z}_{A(n+1)}$ and is given by

$$\boldsymbol{z}_{A(n+1)} = {}^{n}\mathbf{T}_{n+1} \cdot \begin{pmatrix} \boldsymbol{z}_{A(n)} \\ 1 \end{pmatrix} = {}^{n}\mathbf{R}_{n+1} \cdot \boldsymbol{z}_{A(n)} + {}^{n}\boldsymbol{t}_{n+1}. \tag{9.12}$$

Let $\boldsymbol{y}_A$ and $\boldsymbol{y}'_A$ be respectively the normalized image coordinate of $\boldsymbol{z}_{A(n)}$ and $\boldsymbol{z}_{A(n+1)}$ such that,

$$\boldsymbol{z}_{A(n)} = d_{A(n)} \cdot \boldsymbol{y}_A \tag{9.13}$$

$$\boldsymbol{z}_{A(n+1)} = d_{A(n+1)} \cdot \boldsymbol{y}'_A \tag{9.14}$$

where d_A is the depth of position A from the camera center. If $\mathbf{E}$ is the essential matrix (see section 2.2.4) between the two frames, the epipolar relation can be written as,

$$\boldsymbol{y}'^{T}_A \cdot \mathbf{E} \cdot \boldsymbol{y}_A = 0 \tag{9.15}$$

In general, equation (9.15) applies to every static point. In other words, every static point is epipolar-conformant.

Now, let us consider a moving point which starts at position A at frame n to position B at frame $n+1$. It is important to note that the movement is parallel to the camera motion from frame n to frame $n+1$, as shown in figure 9.2. The new position at position B after the parallel motion, denoted as $\boldsymbol{z}_{B(n+1)}$, can be expressed as the old position at A plus a shift along the translation direction

$$\boldsymbol{z}_{B(n+1)} = \boldsymbol{z}_{A(n+1)} - v \cdot {}^{n}\boldsymbol{t}_{n+1}, \tag{9.16}$$

where v is a scale parameter describing the speed ratio of the point w.r.t. the ego-car speed. As the relative translation ${}^{n}\boldsymbol{t}_{n+1}$ defined in equation (9.11) actually describes how the world relatively moves w.r.t. the camera, I need the minus sign in front of v.

Let $\boldsymbol{y}_B$ be the normalized image coordinate of $\boldsymbol{z}_{B(n+1)}$ and d_B is the depth of position B such that the following applies

$$\boldsymbol{z}_{B(n+1)} = d_B \cdot \boldsymbol{y}_B. \tag{9.17}$$

Now, we can check the epipolar conformity of the moving point

$$\begin{aligned}
\boldsymbol{y}_B^T \cdot \mathbf{E} \cdot \boldsymbol{y}_A &= \frac{\boldsymbol{z}^T_{B(n+1)}}{d_B} \cdot \mathbf{E} \cdot \boldsymbol{y}_A \\
&= \frac{(\boldsymbol{z}_{A(n+1)} - v \cdot {}^{n}\boldsymbol{t}_{n+1})^T}{d_B} \cdot \mathbf{E} \cdot \boldsymbol{y}_A \\
&= \frac{(d_{A(n+1)} \cdot \boldsymbol{y}'_A - v \cdot {}^{n}\boldsymbol{t}_{n+1})^T}{d_B} \cdot \mathbf{E} \cdot \boldsymbol{y}_A \\
&= \frac{d_{A(n+1)}}{d_B} \underbrace{\cdot \boldsymbol{y}'^{T}_A \cdot \mathbf{E} \cdot \boldsymbol{y}_A}_{0} - \frac{v}{d_B} \cdot \underbrace{{}^{n}\boldsymbol{t}^T_{n+1} \cdot \mathbf{E}}_{\mathbf{0}} \cdot \boldsymbol{y}_A \\
&= 0.
\end{aligned} \tag{9.18}$$

Equation (9.18) shows that a parallel-moving keypoint also satisfies the epipolar constraint from frame n to frame $n+1$. That means, I have shown that a point moving parallel to the camera motion is epipolar-conformant.

9.5.2 Depth relation between parallel-moving points and ambiguous static points

As illustrated by figure 9.2, a keypoint correspondence $\boldsymbol{y}_A$ in frame n and $\boldsymbol{y}_B$ in frame $n+1$ can represent both a moving point from A to B, and an ambiguous triangulated static point P. Let $\boldsymbol{z}_P$ be the 3D coordinate at position P. I investigate the relation between the motion of the parallel-moving point (see equation (9.16)) and the position of the ambiguous triangulated static point $\boldsymbol{z}_P$.

I compute the intersection of two rays, one from the camera center at CCS_n crossing $\boldsymbol{z}_A$ and another one from the camera center at CCS_{n+1} crossing $\boldsymbol{z}_B$. I transform all coordinates into CCS_{n+1}, thus having the following two equations representing the rays:

$$\begin{aligned}
\boldsymbol{z}_P^{(1)} &= \mathbf{0} + \alpha \cdot (\boldsymbol{z}_{B(n+1)} - \mathbf{0}) \\
&= \alpha \cdot \boldsymbol{z}_{B(n+1)} \\
&= \alpha \cdot (\boldsymbol{z}_{A(n+1)} - v \cdot {}^{n}\boldsymbol{t}_{n+1}) \\
&= \alpha \cdot ({}^{n}\mathbf{R}_{n+1} \cdot \boldsymbol{z}_{A(n)} + {}^{n}\boldsymbol{t}_{n+1} - v \cdot {}^{n}\boldsymbol{t}_{n+1}) \\
&= \alpha(1-v) \cdot {}^{n}\boldsymbol{t}_{n+1} + \alpha \cdot ({}^{n}\mathbf{R}_{n+1} \cdot \boldsymbol{z}_{A(n)})
\end{aligned} \tag{9.19}$$

and

$$\begin{aligned}
\boldsymbol{z}_P^{(2)} &= {}^{n}\boldsymbol{t}_{n+1} + \beta \cdot (\boldsymbol{z}_{A(n+1)} - {}^{n}\boldsymbol{t}_{n+1}) \\
&= {}^{n}\boldsymbol{t}_{n+1} + \beta \cdot {}^{n}\mathbf{R}_{n+1} \cdot \boldsymbol{z}_{A(n)} + {}^{n}\boldsymbol{t}_{n+1} - {}^{n}\boldsymbol{t}_{n+1} \\
&= {}^{n}\boldsymbol{t}_{n+1} + \beta \cdot ({}^{n}\mathbf{R}_{n+1} \cdot \boldsymbol{z}_{A(n)})
\end{aligned} \tag{9.20}$$

By comparing equation (9.19) and (9.20), as long as ${}^{n}\mathbf{R}_{n+1} \cdot \boldsymbol{z}_{A(n)}$ is not a multiple of ${}^{n}\boldsymbol{t}_{n+1}$, I come to the conclusion that

$$\alpha(1-v) = 1 \quad \rightarrow \quad \alpha = \frac{1}{1-v} \tag{9.21}$$

applies. It is important to note that α is also the depth ratio between positions B and P (see equation (9.19)), denoted as d_B and d_P.

$$\frac{d_P}{d_B} = \frac{1}{1-v} \quad \rightarrow \quad (1-v)d_P = d_B \tag{9.22}$$

Hence, we can identify several cases of parallel-moving points based on the analysis of v:

- If the point moves on the opposite direction w.r.t. camera motion ($v < 0$), then the ambiguous static point is closer than the moving point.

$$v < 0 \quad \rightarrow \quad d_P < d_B \tag{9.23}$$

- If the point moves at the same direction w.r.t. camera motion with lower speed ($0 < v < 1$), then the ambiguous static point is farther than the moving point.

$$0 < v < 1 \quad \rightarrow \quad d_P > d_B \tag{9.24}$$

- If the point moves at the same direction w.r.t. camera motion with the same speed ($v = 1$), then the ambiguous static point is at infinity.

$$v = 1 \quad \rightarrow \quad d_P \rightarrow \infty \tag{9.25}$$

- If the point moves at the same direction w.r.t. camera motion with higher speed ($v > 1$), then the ambiguous static point $\boldsymbol{z}_P$ is found behind the camera.

9.5.3 Keypoint-based depth verification

Let $Q(n)$ and $Q(n+1)$ be two associated car patches corresponding to the same car from two consecutive frames n and $n+1$. I employ epipolar matching within $Q(n)$ and $Q(n+1)$ to obtain keypoint correspondences $\boldsymbol{x}_i(n)$ and $\boldsymbol{x}_i(n+1)$, for $i = 1, 2, .., m$. This approach is considered only when the number of correspondences is at least τ_{mc} (see table 3.1 for its value). Then, I triangulate the correspondences to obtain the 3D coordinates $\boldsymbol{z}_{Pi}$.

I compute the relative difference Δd_i between the triangulated depth d_{Pi} and the depth information from the CNN depth map d_{Bi}:

$$\Delta d_i = \frac{|d_{Bi} - d_{Pi}|}{d_{Pi}} = \frac{|(1-v)d_{Pi} - d_{Pi}|}{d_{Pi}} = |v|. \tag{9.26}$$

The keypoint $\boldsymbol{x}_i$ on the car patch Q is recognized as a moving point, if the relative depth difference exceeds τ_v (see table 3.1 for its value). Hence, τ_v also describes the maximum speed ratio w.r.t. the ego-car speed that can be detected as a moving point.

$$\Delta d_i > \tau_v \rightarrow \text{moving point} \tag{9.27}$$

Let m_i be the number of moving points found in patch Q. The car patch Q is identified as an IMO, if the ratio of moving keypoints exceeds an empirically set threshold τ_{rm} (see table 3.1 for its value):

$$\frac{m_i}{m} > \tau_{rm} \rightarrow \texttt{IMO}. \tag{9.28}$$

9.5.4 Keypoint-free depth verification

For keypoint-free depth comparison, I look into the car patches $Q(n)$ and $Q(n+1)$. Combining the 2D pixel position of the patches and the depth information from the CNN, each car patch can be represented by a single 3D point derived from the 2D center of mass of the patch and the median of the depth values.

The 2D center of masses of the patches $Q(n)$ and $Q(n+1)$ are given by $\boldsymbol{c}(n)$ and $\boldsymbol{c}(n+1)$, respectively. The median depth of patches $Q(n)$ and $Q(n+1)$ are denoted as $d(n)$ and $d(n+1)$. Hence, each patch can be represented by a 3D point $\boldsymbol{z}$ whose x and y positions are defined by the center of mass $\boldsymbol{c}$ and the z position is given by the median depth d.

$$\boldsymbol{z} = d \cdot \mathbf{K}^{-1} \cdot \begin{pmatrix} \boldsymbol{c} \\ 1 \end{pmatrix}, \tag{9.29}$$

where $\mathbf{K}$ is the intrinsic camera matrix.

Now, the patch $Q(n)$ and $Q(n+1)$ are represented by the 3D points $\boldsymbol{z}(n)$ and $\boldsymbol{z}(n+1)$. However, both 3D points are measured based on their respective camera coordinate systems (CCS). In order to compare them, I transform $\boldsymbol{z}(n)$ into CCS_{n+1},

$$\boldsymbol{z}(n \to n+1) = {}^{n}\mathbf{R}_{n+1} \cdot \boldsymbol{z}(n) + {}^{n}\boldsymbol{t}_{n+1}. \tag{9.30}$$

Now, we can calculate the absolute distance between the 3D points representing patches $Q(n)$ and $Q(n+1)$:

$$\Delta z = |\boldsymbol{z}(n+1) - \boldsymbol{z}(n \to n+1)|. \tag{9.31}$$

If both 3D points $\boldsymbol{z}(n \to n+1)$ and $\boldsymbol{z}(n+1)$ are similar, it indicates that the patch Q corresponds to a static car. However, if they significantly differ, I identify the car as an IMO.

As I deal with parallel-moving cars, the relative position of these cars change only in one axis corresponding to the depth value (z-axis in my setup), hence the depth consistency is the focus to analyze. The x and y components of $\boldsymbol{z}(n \to n+1)$ and $\boldsymbol{z}(n+1)$ are almost always the same. I set τ_{xy} as the maximum value for both x and y components of Δz to be classified as a static car.

Let $d_m(n)$ and $d_m(n+1)$ be the depth (z) components of $\boldsymbol{z}(n \to n+1)$ and $\boldsymbol{z}(n+1)$. I compute the relative depth difference r_{dm} by

$$r_{dm} = \frac{|d_m(n) - d_m(n+1)|}{\min\left(d_m(n), d_m(n+1)\right)}. \tag{9.32}$$

The car patch is categorized as an IMO, if the relative depth difference is more than $\tau_{dm,IMO}$ and as a static car, if it is less than $\tau_{dm,static}$.

9.6 Propagation of label information

I determine the number of patches $P(n)$ in image n and the set of labels $\ell_i(n)$. Each car patch is represented by a feature vector $\boldsymbol{f}_i(n)$ consisting of its center of mass $\boldsymbol{c}_i(n)$,

its size (pixel count) $s_i(n)$, its mean gray value $m_i(n)$, and its gray value standard deviation $\sigma_i(n)$. This information is collected for all patches by a single pass over the new label image generated by the CNN.

As the motion of those IMOs belonging to crossing traffic often exhibit very long 2D displacements in the image, they cannot be expected to be trackable by a differential LK-style motion tracker like the one which is applied in propagation based tracking (PbT). The association between 'old' patches in image n and new patches in image $n+1$ is performed by testing for each new patch the compatibility with each old patch.

The association is performed in a looping greedy manner (forward and backward) whenever car patches are observed in both image n and image $n+1$, subjecting each potential association between a patch j in image n represented by $\boldsymbol{f}_j(n)$, and a patch k in image $n+1$ represented by $\boldsymbol{f}_k(n+1)$. An association match between two car patches is accepted only when the pair of car patches reciprocally chooses each other as the best match.

I predict the position of the car patch j at frame $n+1$, represented by the predicted center of mass $\hat{\boldsymbol{c}}_j(n+1)$, using the information of past positions at up to three previous frames as presented in equation 9.34. I use the assumption of constant 2D acceleration to predict the pixel coordinate of the center of mass for each car patch.

$$\begin{aligned}&(\hat{\boldsymbol{c}}_j(n+1) - \boldsymbol{c}_j(n)) - (\boldsymbol{c}_j(n) - \boldsymbol{c}_j(n-1)) \\ &= (\boldsymbol{c}_j(n) - \boldsymbol{c}_j(n-1)) - (\boldsymbol{c}_j(n-1) - \boldsymbol{c}_j(n-2))\end{aligned} \tag{9.33}$$

which leads to

$$\hat{\boldsymbol{c}}_j(n+1) = 3\boldsymbol{c}_j(n) - 3\boldsymbol{c}_j(n-1) + \boldsymbol{c}_j(n-2) \tag{9.34}$$

A set of minimum requirements to accept patch matches between frame n and frame $n+1$ is defined using empirical thresholds τ_s, τ_m, τ_σ, and τ_c,

$$|s_j(n) - s_k(n+1)| < \tau_s \tag{9.35}$$
$$|m_j(n) - m_k(n+1)| < \tau_m \tag{9.36}$$
$$|\sigma_j(n) - \sigma_k(n+1)| < \tau_\sigma \tag{9.37}$$
$$\|\hat{\boldsymbol{c}}_j(n+1) - \boldsymbol{c}_k(n+1)\| < \tau_c \tag{9.38}$$

and subsequently confirming each 'surviving' association. Thus, there is no guarantee that this is the best association, but the procedure is fast. Obviously, some patches may be left unassociated, both in image n as well as in image $n+1$.

9.7 Experiment on IMO detection

I tested the proposed IMO detection method on the KITTI dataset (Geiger et al. 2013). First, I tested our method on the KITTI MoSeg dataset (Siam, Mahgoub, Zahran, Yogamani, Jagersand and El-Sallab 2017) which contains of 349 test images with

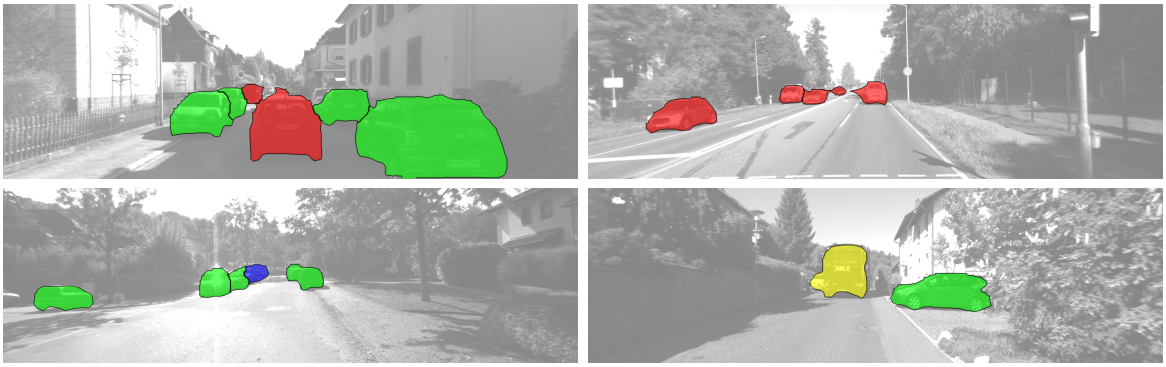

Figure 9.7: Examples from the new IMO candidates dataset. The colored overlay encoding is as follows: red ↔ IMO (class 0), green ↔ static (class 1), blue ↔ too far away (class 2) and yellow ↔ undetermined (class 3).

annotated motion labels. For evaluation purposes, I compare the results with MODNet (Siam et al. 2017) in terms of the precision of identifying static and moving objects.

Second, I tested the proposed method on the KITTI odometry dataset, which is much bigger than the KITTI MoSeg dataset. Unfortunately, KITTI does not provide for this dataset neither instance labels for each vehicle nor patches, which classifies vehicles into static or moving objects. Thus, a new dataset has been created to evaluate the proposed approach.

9.7.1 KITTI odometry dataset with motion labels

The proposed CNN from van den Brand et al. (van den Brand et al. 2016) was utilized to generate candidate labels for the vehicle instances. In the current state of the dataset, I have limited the detected objects to vehicles only. This can be further extended to other objects, like pedestrians or bicycle in future work.

Given these segmented candidate labels, one of the following class labels is manually assigned to each candidate patch in all images : 0 - background (non-vehicles), 1 - independently moving vehicle, 2 - static (non-moving) vehicle, 3 - far away vehicles (median distance greater than 50m) and 4 - undetermined. An IMO candidate is labeled as undetermined, if the patch does not show a vehicle or if the patch is stretched over more than one vehicles, which do not fall into same category, like static or IMO. Some examples of this dataset are shown in fig 9.7.

9.7.2 Evaluation of IMO detection

The parameters used for experiment are listed in table 3.1. As the CNN-based IMO candidate patches can reliably detect IMOs up to a distance of 50 meters, the proposed IMO detection is also evaluated for the same maximum distance.

Table 9.1: Accuracy of IMO detection on the KITTI MoSeg dataset.

Method	P **static**	P **moving**	P **average**
MODNet (Siam et al. 2017)	0.65	0.67	0.66
Ours	**0.74**	**0.84**	**0.79**

The performance of the IMO classification is expressed by precision P, recall R, specificity S, and accuracy A. They are computed based on four metric occurrences: *true positive* (t_p), *false positive* (f_p), *true negative* (t_n) and *false negative* (f_n).

$$P = \frac{t_p}{t_p + f_p}\ ; \quad R = \frac{t_p}{t_p + f_n}\ ; \quad S = \frac{t_n}{t_n + f_p}\ ; \quad A = \frac{t_p + t_n}{t_p + t_n + f_p + f_n} \tag{9.39}$$

Besides computing the accuracy of the results, I measure the decisiveness of the proposed method to give definite output (`IMO`/`static`) as compared to `undetermined`. I define the decisiveness level D as

$$D = \frac{n_{IMO} + n_{static}}{n_{IMO} + n_{static} + n_{undetermined}} \tag{9.40}$$

where n_{IMO}, n_{static}, and $n_{undetermined}$ are respectively the number of outputs as `IMO`, `static`, and `undetermined`.

9.7.2.1 Accuracy on the KITTI MoSeg dataset

Table 9.1 presents the precision of the IMO detection using the proposed method and using MODNet (Siam et al. 2017). The precision of the method proposed in this chapter is better on both identifying static cars and moving cars. The average precision of the proposed method is 0.79 as compared to 0.66 of MODNet. Figure 9.8 shows the exemplary results of the IMO detection using the proposed method and using MODNet.

9.7.2.2 Accuracy on the KITTI odometry dataset

The results of the proposed IMO detection on the KITTI odometry dataset are presented in table 9.2. The overall decisiveness level is 91%. That means, the undetermined outputs only happen in about 9% of the total car appearances and they mostly occur when the cars are first time observed in the scene. The recall rate, or the true positive rate, has an overall value of 87% which reflects the high accuracy of the IMO detection. The overall specificity rate, or the true negative rate, is 83%, while the overall accuracy is 84%.

Sequence 01 and sequence 04 are notably full of epipolar-conformant IMOs, both parallel and anti-parallel cases. The results in table 9.2 for both sequences indicate that the proposed IMO detection is able to identify almost all IMOs. Figure 9.9 (top and middle images) shows the IMO detection for KITTI sequence 01 and sequence 09. The

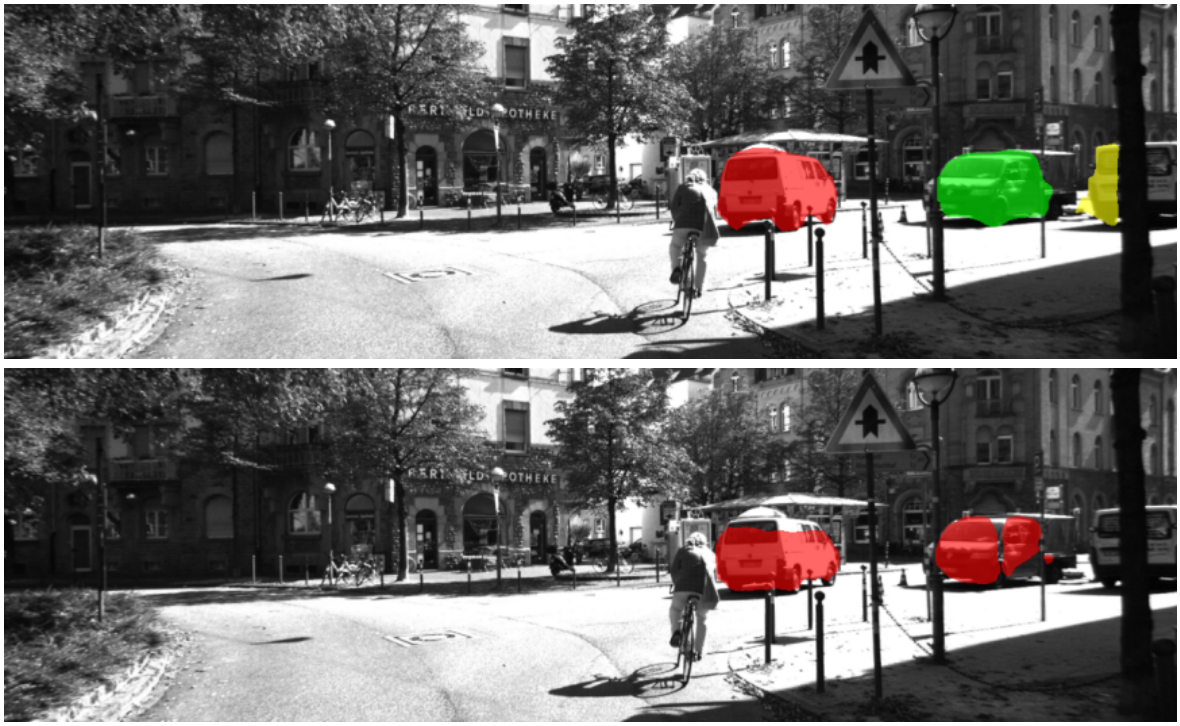

Figure 9.8: Exemplary results of the car classification into static and IMO labels on the KITTI MoSeg dataset: using the proposed method (**top**) and using MODNet (**bottom**). Red color represents IMO, green color represents static car, and yellow color represents undetermined. The comparison shows that the proposed method correctly identifies a static parked car while MODNet wrongly classifies it as an IMO.

parallel-moving cars are correctly detected and marked with red colors. The static cars are also correctly identified in green colors.

The accuracy level is directly influenced by the user-defined threshold τ_v (see equation (9.27)) that describes the maximum detectable speed ratio of the moving car w.r.t. the ego-car speed. The threshold τ_v should be low enough in order to be able to detect even slow moving objects, while at the same time it cannot be too low to anticipate measurement errors. If an IMO moves very slowly below τ_v, the proposed framework cannot identify it as a moving object, as happened in KITTI sequence 10, when a truck moves backward slowly (see the bottom image of figure 9.9). Similarly, if the error in determining triangulated 3D position is too high (e.g. from matching error or egomotion error), it could lead to false positive or false negative classifications.

9.8 Concluding remarks

This chapter has presented an IMO detection method using a monocular camera. The proposed method employs a CNN to provide IMO candidates and depth images. The non-epipolar-conformant IMOs are detected through checks against the multi-frame epipolar consistency. The epipolar-conformant IMOs are detected based on the depth

Table 9.2: Accuracy of IMO detection on the KITTI dataset.

Sequence	D	R	S	A
0	0.90	0.41	0.81	0.81
1	0.84	0.97	n.a.	0.97
2	0.90	0.71	0.82	0.82
3	0.89	1.00	0.86	0.91
4	0.96	1.00	1.00	1.00
5	0.90	0.95	0.86	0.86
6	0.86	1.00	0.86	0.86
7	0.93	0.87	0.89	0.89
8	0.93	0.61	0.81	0.81
9	0.92	0.76	0.90	0.90
10	0.95	0.68	0.97	0.92
Overall	0.91	0.87	0.83	0.84

consistency of each IMO candidate. The motion label (IMO/static) is propagated over time by establishing patch label association between two consecutive frames based on the cue combination of movement and appearance. An experiment on KITTI dataset verifies the accuracy of the proposed method.

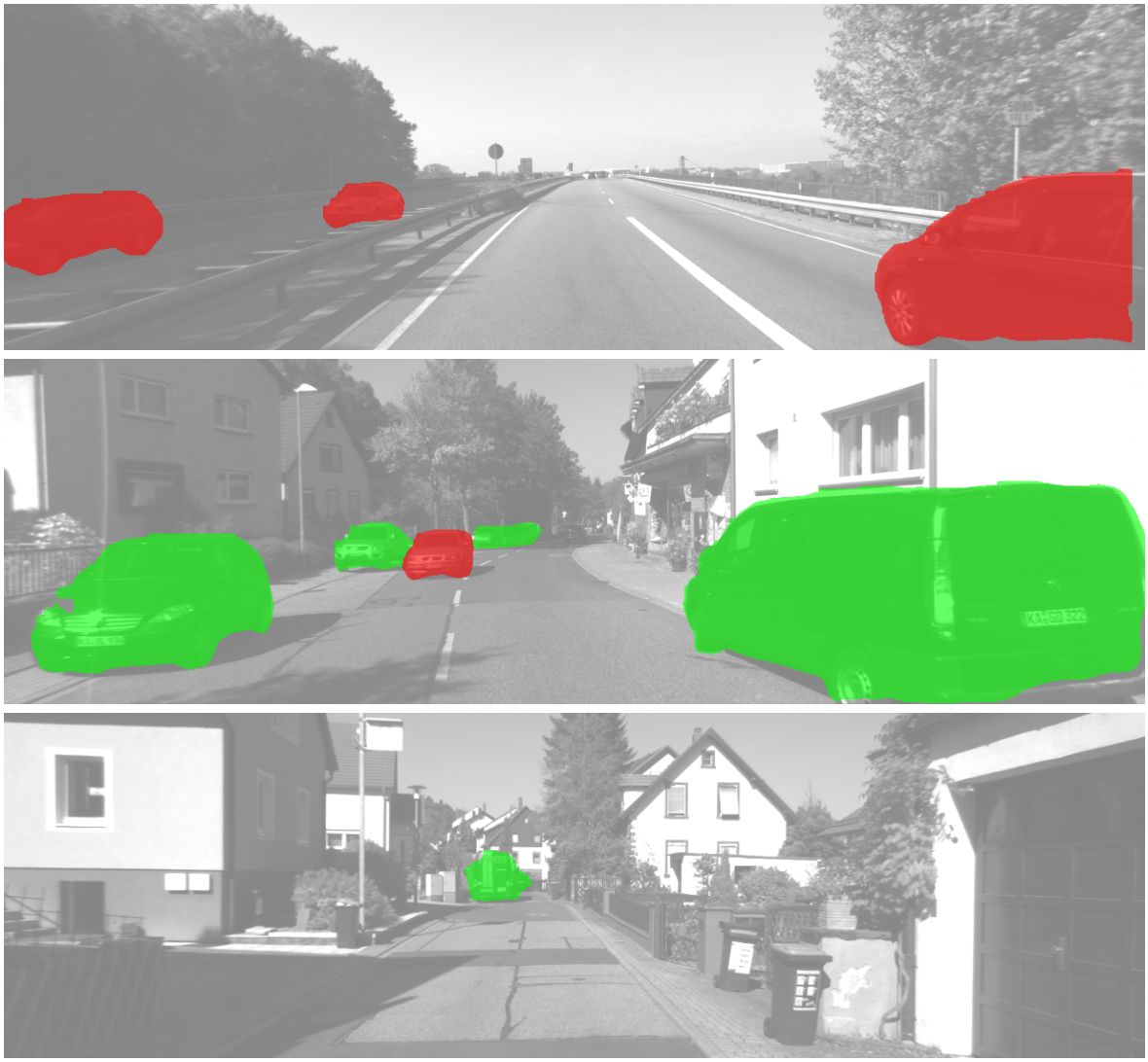

Figure 9.9: Exemplary results of the car classification into static and IMO labels on the KITTI sequence 01 (top), sequence 09 (middle), and sequence 10 (bottom). Red color represents IMO while green color represents static car.

Chapter 10

Summary and outlook

10.1 Summary

In this dissertation, Predictive Monocular Odometry (PMO) has been presented as a novel method for monocular visual odometry. The proposed framework has been developed for automotive applications in a monocular setup, specifically for a forward-looking camera attached on the ego-car.

The main goal of the proposed framework is to provide the localization of the ego-car by accumulating the frame-to-frame relative pose estimates over time. A 3D world map consisting of sparse 3D points can be obtained as a result of PMO. In contrast to the other mainstream frameworks, PMO has been built without using RANSAC and without multi-frame bundle adjustment.

The highlights of the PMO components can be summarized as follows. First, we regard the first two frames streamed from a camera: frame $n-1$ and frame n. An initialization bootstrapping step guarantees that the relative pose estimate between these first two frames is available. Furthermore, the initialization step also guarantees that the keypoint correspondences between frame $n-1$ and frame n as well as the corresponding 3D world positions are also available. Now, the main loop of PMO starts as soon as a new frame $n+1$ is processed.

The relative pose between frame n and frame $n+1$ is initially estimated using the keypoint-free pose estimator which combines the rotational pitch-yaw-roll estimation and the statistical motion prediction. An additional scheme to detect wrong rotation estimates due to moving objects has been proposed.

With the availability of the relative pose estimate between frame n and frame $n+1$, the 3D world position of the tracked keypoints can be projected onto frame $n+1$ to find the pixel coordinate of the keypoints in frame $n+1$. Both the relative pose estimate and the tracked keypoint pixel positions are subsequently simultaneously optimized using the joint epipolar tracking (JET).

In order to have a robust and long-term tracking capability, a new set of keypoints

in frame n are generated especially in the areas that lack tracked keypoints. The corresponding matches of these newly generated keypoints in frame $n+1$ are obtained using epipolar-constrained Lucas-Kanade matching. The statistics of the matching results are thoroughly analyzed.

All the keypoint correspondences are triangulated based on the optimized relative pose to obtain the 3D world position of the keypoints. A method to assess the triangulation precision for each keypoint correspondence is employed. The process is repeated every time a new frame arrives.

The scale ambiguity problem in monocular frameworks is solved using a multimodal scale estimation method which combines dense and sparse matching on the ground plane by assuming that the camera height over ground is known. A method to detect independently moving objects has also been proposed.

The proposed PMO framework has extensively been tested on the KITTI dataset which consists of thousands of images taken from a camera attached on a moving car. The results show that PMO is one of the leading monocular framework in the KITTI benchmark.

The work builds strongly on a good cooperation within the VSI work group: PMO exploits prior work by colleagues and joint work with colleagues, such as the robust phase correlation based motion estimator (Ochs et al. 2015), the statistical motion model and predictor (Bradler et al. 2015), the keypoint-free pitch-yaw-roll estimator (Barnada et al. 2015), the joint epipolar tracking (JET) method (Bradler et al. 2017), and CNNs developed for car instance segmentation (van den Brand et al. 2016), road surface labelling (Fanani, Stürck, Ochs, Bradler and Mester 2017), and monocular depth estimation (Ochs 2017). The present work was the first complete integration of these components into a large-scale system; the present thesis focuses on those components that were developed by the author and the integration into a complex scheme that finally achieved leading ranking positions in the KITTI benchmark.

10.2 Outlook

It has been shown that PMO provides an accurate monocular visual odometry. PMO can still be further improved and exploited on several aspects. The following potential improvements to PMO are proposed as future work:

- Adaptive phase-correlation windows for initial rotation estimation.
  The initial rotation estimate is obtained from the pitch-yaw-roll estimation which is based on far-view field analysis. In the presented scheme, five phase-correlation windows with uniform size are positioned on fixed positions in the middle of the image, assuming that they correspond to far-view field. However, this assumption is not always correct since the position of far-view fields on the image depends on the 3D world structure at each frame. Hence, a dynamic placement of the phase-correlation windows can be proposed in order to have a more robust initial

rotation estimation. The position of the far-view field on the image can be estimated from a depth image or using some other depth measurement methods.

- Keyframe mechanism.
  A keyframe mechanism can be introduced to PMO in order to make it more robust and computationally more efficient. The keyframe mechanism will select frames containing meaningful new information and skip frames with redundant contents.

- Robustness against illumination changes.
  It is well known that there is an illumination change problem even for images taken with the same camera. A method to handle the illumination changes can improve the robustness of PMO.

- Testing on other datasets.
  PMO has shown an impressive accuracy result for the KITTI dataset. It can be tested and applied to other datasets, both from real driving scenarios such as TorontoCity dataset (Wang, Bai, Mattyus, Chu, Luo, Yang, Liang, Cheverie, Fidler and Urtasun 2016) and synthetic datasets, such as Congrats driving dataset (Biedermann et al. 2015).

- Photometric calibration
  PMO assumes that the camera has been calibrated and takes the camera calibration information as one of the inputs to the PMO framework. In order to remove this assumption, an online photometric calibration can be applied as proposed in (Bergmann, Wang and Cremers 2017).

- Matching densification.
  PMO is categorized as a sparse method, where the matching density is much lower than in a dense method. The density can be increased using some densification methods such as employing an interpolated dense depth map as proposed in (Ochs, Bradler and Mester 2017).

- High-speed keypoint tracking.
  In high-speed scenarios, the tracked keypoints on the image result in longer-displacement optical flows than in a lower speed. As PMO employs differential matching to track the keypoints, the long displacements might be too large to handle. The condition could be worse if there are repetitive patterns along the epipolar line so that wrong correspondences could easily occur. Recently, Pereira et al (Pereira et al. 2017) proposed a method which projects a set of assumed keypoint depths to detect wrong matches due to repetitive patterns. It results in accurate odometry results in high-speed cases. This can be adopted to further improve the accuracy of PMO.

- IMO detection for vehicles and pedestrians.
  This dissertation proposes a method for detecting independently moving objects (IMO). However, the focus of the IMO detection is only on moving cars due to the unavailability of patches from moving objects other than cars. CNN can be employed to generate patches for other vehicles and also pedestrians. By doing so, it opens up the possibility of detecting all kinds of IMO in traffic scenes.

- Multi-frame optimization.
  PMO focuses on two-view visual odometry, meaning that the relative pose estimation is optimized only between two consecutive frames. While two-view approach offers simplicity, more accurate results can potentially be achieved when more than two frames are optimized simultaneously. An inevitable drawback of the multi-frame approach would be the higher computational time and power required to handle more data. A multi-frame optimization tool such as bundle adjustment can be applied to solve global optimization problem in estimating the relative pose between more than two frames.

- Multi-camera framework.
  The proposed PMO framework is intended for monocular setup. To build a complete visual ADAS system, multi-camera system might be required, for instance to have a full 360° coverage around the car. With some modifications, PMO can be employed in an integrated multi-camera setup by being the core framework for each camera. The data and results from all cameras can be combined to have more complete information than only from one single camera.

Bibliography

Aach, T., Kaup, A. and Mester, R.: 1993, Statistical model-based change detection in moving video, *Signal processing* **31**(2), 165–180.

Alismail, H., Browning, B. and Lucey, S.: 2016, Photometric bundle adjustment for vision-based SLAM, *arXiv preprint arXiv:1608.02026* .

Arróspide, J., Salgado, L., Nieto, M. and Mohedano, R.: 2010, Homography-based ground plane detection using a single on-board camera, *Intelligent Transport Systems, IET* **4**(2), 149–160.

Aschwanden, P. and Guggenbuhl, W.: 1992, Experimental results from a comparative study on correlation-type registration algorithms, *Robust computer vision* pp. 268–289.

Badino, H., Franke, U. and Pfeiffer, D.: 2009, The stixel world-a compact medium level representation of the 3D-world, *Pattern Recognition*, Springer, pp. 51–60.

Bai, M., Luo, W., Kundu, K. and Urtasun, R.: 2016, Exploiting semantic information and deep matching for optical flow, *European Conference on Computer Vision*, Springer, pp. 154–170.

Baker, S. and Matthews, I.: 2004, Lucas-Kanade 20 years on: A unifying framework, *International Journal of Computer Vision* **56**(3), 221–255.

Barnada, M., Conrad, C., Bradler, H., Ochs, M. and Mester, R.: 2015, Estimation of automotive pitch, yaw, and roll using enhanced phase correlation on multiple far-field windows, *Intelligent Vehicles Symposium Proceedings, 2015 IEEE*, IEEE.

Bay, H., Tuytelaars, T. and Van Gool, L.: 2006, SURF: Speeded up robust features, *Computer vision–ECCV 2006*, Springer, pp. 404–417.

Benhimane, S. and Malis, E.: 2004, Real-time image-based tracking of planes using efficient second-order minimization, *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, Vol. 1, IEEE, pp. 943–948.

Bergmann, P., Wang, R. and Cremers, D.: 2017, Online photometric calibration for auto exposure video for realtime visual odometry and SLAM, *arXiv preprint arXiv:1710.02081* .

Biedermann, D., Ochs, M. and Mester, R.: 2015, COnGRATS: Realistic simulation of traffic sequences for autonomous driving, *Image and Vision Computing New Zealand (IVCNZ)*, ACM.

Birchfield, S. T. and Pundlik, S. J.: 2008, Joint tracking of features and edges, *CVPR 2008.*, IEEE, pp. 1–6.

Botterill, T., Mills, S. and Green, R.: 2013, Correcting scale drift by object recognition in single-camera SLAM, *IEEE transactions on cybernetics* **43**(6), 1767–1780.

Bradler, H.: 2016, Verbundoptimierung von Bildpunktkorrespondenzen in Videosequenzen bei simultaner Optimierung der Epipolargeometrie, Masters Thesis (in German), VSI Lab, CS Dept, Goethe University, Frankfurt am Main.

Bradler, H., Ochs, M., Fanani, N. and Mester, R.: 2017, Joint Epipolar Tracking (JET): Simultaneous optimization of epipolar geometry and feature correspondences, *Sixth IEEE Workshop on Applications of Computer Vision (WACV)*, IEEE.

Bradler, H., Wiegand, B. and Mester, R.: 2015, The statistics of driving sequences – and what we can learn from them, *ICCV Workshop on Computer Vision for Road Scene Understanding and Autonomous Driving*.

Brox, T., Bregler, C. and Malik, J.: 2009, Large displacement optical flow, *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, IEEE, pp. 41–48.

Brox, T., Bruhn, A. and Weickert, J.: 2006, Variational motion segmentation with level sets, *Proc. ECCV 2006*, Springer, pp. 471–483.

Bruhn, A., Weickert, J. and Schnörr, C.: 2005, Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods, *International Journal of Computer Vision* **61**(3), 211–231.

Buczko, M. and Willert, V.: 2016, How to distinguish inliers from outliers in visual odometry for high-speed automotive applications, *Intelligent Vehicles Symposium (IV)*, IEEE, pp. 1–6.

Burusa, A. K.: 2017, Visual-inertial odometry for autonomous ground vehicles.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S. and Schiele, B.: 2016, The Cityscapes Dataset for Semantic Urban Scene Understanding, *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3213–3223.

Davison, A. J.: 2003, Real-time simultaneous localisation and mapping with a single camera, *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, IEEE, pp. 1403–1410.

Di Leo, G., Liguori, C. and Paolillo, A.: 2011, Covariance propagation for the uncertainty estimation in stereo vision, *IEEE Transactions on Instrumentation and Measurement* **60**(5), 1664–1673.

Di Leo, G. and Paolillo, A.: 2011, Uncertainty evaluation of camera model parameters, *Instrumentation and Measurement Technology Conference (I2MTC), 2011 IEEE*, IEEE, pp. 1–6.

Diebel, J., Reutersward, K., Thrun, S., Davis, J. and Gupta, R.: 2004, Simultaneous localization and mapping with active stereo vision, *IEEE International Conference on Intelligent Robots and Systems (IROS)*, Vol. 4, pp. 3436–3443.

Eggert, D. W., Lorusso, A. and Fisher, R. B.: 1997, Estimating 3-D rigid body transformations: A comparison of four major algorithms, *Machine Vision and Applications* **9**(5-6), 272–290.

Engel, J., Koltun, V. and Cremers, D.: 2016, Direct sparse odometry, *arXiv preprint arXiv:1607.02565* .

Engel, J., Schöps, T. and Cremers, D.: 2014, LSD-SLAM: Large-scale direct monocular SLAM, *Computer Vision–ECCV 2014*, Springer, pp. 834–849.

Fanani, N., Barnada, M. and Mester, R.: 2015, Motion priors estimation for robust matching initialization in automotive applications, *International Symposium on Visual Computing (ISVC)*.

Fanani, N. and Mester, R.: 2018, The precision of triangulation in monocular visual odometry, *IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, IEEE.

Fanani, N., Ochs, M., Bradler, H. and Mester, R.: 2016, Keypoint trajectory estimation using propagation based tracking, *IEEE Intelligent Vehicles Symposium (IV)*, IEEE, pp. 933–939.

Fanani, N., Ochs, M. and Mester, R.: 2018, Detecting parallel-moving objects in the monocular case employing CNN depth maps, *European Conference on Computer Vision*, Springer.

Fanani, N., Ochs, M., Stürck, A. and Mester, R.: 2018, CNN-based multi-frame IMO detection from a monocular camera, *IEEE Intelligent Vehicles Symposium (IV)*, IEEE.

Fanani, N., Stuerck, A., Barnada, M. and Mester, R.: 2017, Multimodal scale estimation for monocular visual odometry, *Intelligent Vehicles Symposium (IV)*, IEEE.

Fanani, N., Stürck, A., Ochs, M., Bradler, H. and Mester, R.: 2017, Predictive Monocular Odometry (PMO): What is possible without RANSAC and multiframe bundle adjustment?, *Image and Vision Computing* .

Fischler, M. A. and Bolles, R. C.: 1981, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, *Communications of the ACM* **24**(6), 381–395.

Forster, C., Pizzoli, M. and Scaramuzza, D.: 2014, SVO: Fast semi-direct monocular visual odometry, *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, IEEE, pp. 15–22.

Förstner, W.: 1986, A feature based correspondence algorithm for image matching, *International Archives of Photogrammetry and Remote Sensing* **26**(3), 150–166.

Förstner, W. and Gülch, E.: 1987, A fast operator for detection and precise location of distinct points, corners and centres of circular features, *Proc. ISPRS intercommission conference on fast processing of photogrammetric data*, pp. 281–305.

Franke, U., Gavrila, D., Görzig, S., Lindner, F., Paetzold, F. and Wöhler, C.: 1998, Autonomous driving goes downtown, *IEEE Intelligent Systems* **6**, 40–48.

Frost, D. P., Murray, D. W. et al.: 2016, Object-aware bundle adjustment for correcting monocular scale drift, *2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, pp. 4770–4776.

Geiger, A., Lenz, P., Stiller, C. and Urtasun, R.: 2013, Vision meets robotics: The KITTI dataset, *The International Journal of Robotics Research* p. 0278364913491297.

Geiger, A., Lenz, P. and Urtasun, R.: 2012, Are we ready for autonomous driving? The KITTI vision benchmark suite, *Conference on Computer Vision and PatternRecognition (CVPR)*.

Geiger, A., Ziegler, J. and Stiller, C.: 2011, Stereoscan: Dense 3D reconstruction in real-time, *Intelligent Vehicles Symposium (IV), 2011 IEEE*, IEEE, pp. 963–968.

Gräter, J., Schwarze, T. and Lauer, M.: 2015, Robust scale estimation for monocular visual odometry using structure from motion and vanishing points, *2015 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, pp. 475–480.

Harris, C. and Stephens, M.: 1988, A combined corner and edge detector., *Alvey vision conference*, Vol. 15, Citeseer, p. 50.

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

Hartley, R. I. and Sturm, P.: 1997, Triangulation, *Computer vision and image understanding* **68**(2), 146–157.

Hartley, R. and Zisserman, A.: 2003, *Multiple view geometry in computer vision*, Cambridge university press.

Horn, B. K. and Schunck, B. G.: 1981, Determining optical flow, *1981 Technical symposium east*, International Society for Optics and Photonics, pp. 319–331.

Immerkaer, J.: 1996, Fast noise variance estimation, *Computer vision and image understanding* **64**(2), 300–302.

Irani, M. and Anandan, P.: 1999, About direct methods, *Vision Algorithms: Theory and Practice*, Springer, pp. 267–277.

Jähne, B.: 1993, *Spatio-temporal image processing: theory and scientific applications*, Vol. 751, Springer Science & Business Media.

Jazayeri, A., Cai, H., Zheng, J. Y. and Tuceryan, M.: 2011, Vehicle detection and tracking in car video based on motion model, *IEEE Transactions on Intelligent Transportation Systems* **12**(2), 583–595.

Johnson, A. E., Goldberg, S. B., Cheng, Y. and Matthies, L. H.: 2008, Robust and efficient stereo feature tracking for visual odometry, *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 39–46.

Jung, B. and Sukhatme, G. S.: 2004, Detecting moving objects using a single camera on a mobile robot in an outdoor environment, *International Conference on Intelligent Autonomous Systems*, pp. 980–987.

Kay, S. M.: 1993, *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Klappstein, J., Stein, F. and Franke, U.: 2006, Monocular motion detection using spatial constraints in a unified manner, *Intelligent Vehicles Symposium, 2006 IEEE*, IEEE, pp. 261–267.

Klein, G. and Murray, D.: 2007, Parallel tracking and mapping for small ar workspaces, *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, IEEE, pp. 225–234.

Kundu, A., Jawahar, C. V. and Krishna, K. M.: 2010, Realtime moving object detection from a freely moving monocular camera, *2010 IEEE International Conference on Robotics and Biomimetics*, pp. 1635–1640.

Lenz, P., Ziegler, J., Geiger, A. and Roser, M.: 2011, Sparse scene flow segmentation for moving object detection in urban environments, *2011 IEEE Intelligent Vehicles Symposium (IV)*, pp. 926–932.

Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R. and Furgale, P.: 2015, Keyframe-based visual–inertial odometry using nonlinear optimization, *The International Journal of Robotics Research* **34**(3), 314–334.

Li, S.: 2017, A review of feature detection and match algorithms for localization and mapping, *IOP Conference Series: Materials Science and Engineering* **231**(1), 012003.

Longuet-Higgins, H.: 1981, A computer algorithm for reconstructing a scene from two projections.

López-Rubio, F. J. and López-Rubio, E.: 2015, Foreground detection for moving cameras with stochastic approximation, *Pattern Recognition Letters* **68**, 161 – 168.

Lowe, D. G.: 1999, Object recognition from local scale-invariant features, *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, Vol. 2, IEEE, pp. 1150–1157.

Lowe, D. G.: 2004, Distinctive image features from scale-invariant keypoints, *International journal of computer vision* **60**(2), 91–110.

Lucas, B. D., Kanade, T. et al.: 1981, An iterative image registration technique with an application to stereo vision., *IJCAI*, Vol. 81, pp. 674–679.

Maimone, M., Cheng, Y. and Matthies, L.: 2007, Two years of visual odometry on the mars exploration rovers, *Journal of Field Robotics* **24**(3), 169–186.

Malis, E. and Vargas, M.: 2007, *Deeper understanding of the homography decomposition for vision-based control*, PhD thesis, INRIA.

Mester, R.: 2014, Motion estimation revisited: an estimation-theoretic approach, *Image Analysis and Interpretation (SSIAI), 2014 IEEE Southwest Symposium on*, IEEE, pp. 113–116.

Mester, R. and Hötter, M.: 1995, Robust displacement vector estimation including a statistical error analysis, *International Conference on Image Processing and its Applications*, IET.

Mester, R. and Mühlich, M.: 2001, Improving motion and orientation estimation using an equilibrated total least squares approach, *Image Processing, 2001. Proceedings. 2001 International Conference on*, Vol. 2, IEEE, pp. 929–932.

Mirabdollah, M. H. and Mertsching, B.: 2015, Fast techniques for monocular visual odometry, *German Conference on Pattern Recognition*, Springer, pp. 297–307.

Mohamed, M. A., Mirabdollah, M. H. and Mertsching, B.: 2015, Differential optical flow estimation under monocular epipolar line constraint, *Computer Vision Systems*, Springer, pp. 354–363.

Moravec, H. P.: 1980, Obstacle avoidance and navigation in the real world by a seeing robot rover., *Technical report*, DTIC Document.

Moravec, H. P.: 1981, Rover visual obstacle avoidance., *IJCAI*, pp. 785–790.

Mur-Artal, R., Montiel, J. and Tardós, J. D.: 2015, ORB-SLAM: A versatile and accurate monocular SLAM system, *IEEE Transactions on Robotics* **31**(5), 1147–1163.

Newcombe, R. A., Lovegrove, S. J. and Davison, A. J.: 2011, DTAM: Dense tracking and mapping in real-time, *Computer Vision (ICCV), 2011 IEEE International Conference on*, IEEE, pp. 2320–2327.

Neyman, J. and Pearson, E. S.: 1928, On the use and interpretation of certain test criteria for purposes of statistical inference: Part i, *Biometrika* pp. 175–240.

Nistér, D., Naroditsky, O. and Bergen, J.: 2004, Visual odometry, *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, Vol. 1, IEEE, pp. I–652.

Nützi, G., Weiss, S., Scaramuzza, D. and Siegwart, R.: 2011, Fusion of IMU and vision for absolute scale estimation in monocular SLAM, *Journal of intelligent & robotic systems* **61**(1-4), 287–299.

Ochoa, B. and Belongie, S.: 2006, Covariance propagation for guided matching, *Proceedings of the Workshop on Statistical Methods in Multi-Image and Video Processing (SMVP)*.

Ochs, M.: 2017, Depth estimation from single image using a cnn, Unpublished technical report, VSI Lab, CS Dept, Goethe University, Frankfurt am Main.

Ochs, M., Bradler, H. and Mester, R.: 2015, Enhanced phase correlation for reliable and robust estimation of multiple motion distributions, *Pacific Rim Symposium on Image and Video Technology (PSIVT)*.

Ochs, M., Bradler, H. and Mester, R.: 2017, Learning rank reduced interpolation with principal component analysis, *arXiv preprint arXiv:1703.05061* .

Oliveira, G. L., Radwan, N., Burgard, W. and Brox, T.: 2017, Topometric localization with deep learning. ArXiv preprint.

Ošep, A., Mehner, W., Mathias, M. and Leibe, B.: 2017, Combined image- and world-space tracking in traffic scenes, *ICRA*.

Papoulis, A.: 1990, *Probability & statistics*, Vol. 2, Prentice-Hall Englewood Cliffs.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E.: 2011, Scikit-learn: Machine learning in Python, *Journal of Machine Learning Research* **12**, 2825–2830.

Pereira, F. I., Ilha, G., Luft, J., Negreiros, M. and Susin, A.: 2017, Monocular visual odometry with cyclic estimation, *Graphics, Patterns and Images (SIBGRAPI), 2017 30th SIBGRAPI Conference on*, IEEE, pp. 1–6.

Persson, M., Piccini, T., Felsberg, M. and Mester, R.: 2015, Robust stereo visual odometry from monocular techniques, *Intelligent Vehicles Symposium Proceedings, 2015 IEEE*, IEEE.

Piccini, T., Persson, M., Nordberg, K., Felsberg, M. and Mester, R.: 2014, Good edgels to track: Beating the aperture problem with epipolar geometry, *Computer Vision-ECCV 2014 Workshops*, Springer, pp. 652–664.

QU, X.: 2016, *Landmark based localization: detection, matching and update of landmark with uncertainty analysis*, PhD thesis, Université Paris-Est, Marne-la-Vallée, France.

Ramirez, A., Ohn-Bar, E. and Trivedi, M. M.: 2014, Go with the flow: Improving multi-view vehicle detection with motion cues, *2014 22nd International Conference on Pattern Recognition*, pp. 4140–4145.

Rosten, E. and Drummond, T.: 2006, Machine learning for high-speed corner detection, *Computer Vision–ECCV 2006*, Springer, pp. 430–443.

Rublee, E., Rabaud, V., Konolige, K. and Bradski, G.: 2011, ORB: An efficient alternative to SIFT or SURF, *Computer Vision (ICCV), 2011 IEEE International Conference on*, IEEE, pp. 2564–2571.

Sangi, P.: 2013, *Object motion estimation using block matching with uncertainty analysis*, PhD thesis, University of Oulu, Finland.

Scaramuzza, D. and Fraundorfer, F.: 2011, Visual odometry [tutorial], *IEEE Robot Autom Mag* **18**(4), 80–92.

Scharr, H.: 2000, *Optimal operators in digital image processing*, PhD thesis.

Shi, J. and Tomasi, C.: 1994, Good features to track, *Proceedings CVPR'94.*, IEEE, pp. 593–600.

Siam, M., Mahgoub, H., Zahran, M., Yogamani, S., Jagersand, M. and El-Sallab, A.: 2017, MODNet: Moving object detection network with motion and appearance for autonomous driving, *arXiv preprint arXiv:1709.04821* .

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

Song, S., Chandraker, M. and Guest, C. C.: 2013, Parallel, real-time monocular visual odometry, *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, IEEE, pp. 4698–4705.

Song, S., Chandraker, M. and Guest, C. C.: 2016, High accuracy monocular sfm and scale correction for autonomous driving, *IEEE transactions on pattern analysis and machine intelligence* **38**(4), 730–743.

Stein, G. P., Mano, O. and Shashua, A.: 2000, A robust method for computing vehicle ego-motion, *Intelligent Vehicles Symposium (IV)*, IEEE, pp. 362–368.

Strasdat, H., Montiel, J. and Davison, A. J.: 2010, Scale drift-aware large scale monocular SLAM, *Robotics: Science and Systems VI* .

Sturm, P.: 1997, Critical motion sequences for monocular self-calibration and uncalibrated euclidean reconstruction, *Computer Vision and Pattern Recognition*, IEEE, pp. 1100–1105.

Sutton, M. A., Orteu, J. J. and Schreier, H.: 2009, *Image correlation for shape, motion and deformation measurements: basic concepts, theory and applications*, Springer Science & Business Media.

Taketomi, T., Uchiyama, H. and Ikeda, S.: 2017, Visual SLAM algorithms: A survey from 2010 to 2016, *IPSJ Transactions on Computer Vision and Applications* **9**(1), 16.

Tomasi, C. and Kanade, T.: 1991, Detection and tracking of point features.

Triggs, B., McLauchlan, P. F., Hartley, R. I. and Fitzgibbon, A. W.: 1999, Bundle adjustment—a modern synthesis, *International workshop on vision algorithms*, Springer, pp. 298–372.

Trummer, M., Denzler, J. and Munkelt, C.: 2008, KLT tracking using intrinsic and extrinsic camera parameters in consideration of uncertainty., *VISAPP (2)*, pp. 346–351.

Trummer, M., Munkelt, C. and Denzler, J.: 2009a, Combined GKLT feature tracking and reconstruction for next best view planning, *Joint Pattern Recognition Symposium*, Springer, pp. 161–170.

Trummer, M., Munkelt, C. and Denzler, J.: 2009b, Extending GKLT tracking feature tracking for controlled environments with integrated uncertainty estimation, *Image Analysis*, Springer, pp. 460–469.

Trummer, M., Munkelt, C. and Denzler, J.: 2010, Online next-best-view planning for accuracy optimization using an extended e-criterion, *Pattern Recognition (ICPR), 2010 20th International Conference on*, IEEE, pp. 1642–1645.

van den Brand, J., Ochs, M. and Mester, R.: 2016, Instance-level segmentation of vehicles by deep contours, *Asian Conference on Computer Vision 2016 – Workshops*, Springer, pp. 477–492.

Wang, C.-C., Thorpe, C., Thrun, S., Hebert, M. and Durrant-Whyte, H.: 2007, Simultaneous localization, mapping and moving object tracking, *The International Journal of Robotics Research* **26**(9), 889–916.

Wang, S., Bai, M., Mattyus, G., Chu, H., Luo, W., Yang, B., Liang, J., Cheverie, J., Fidler, S. and Urtasun, R.: 2016, Torontocity: Seeing the world with a million eyes, *arXiv preprint arXiv:1612.00423* .

Wedel, A., Meißner, A., Rabe, C., Franke, U. and Cremers, D.: 2009, Detection and segmentation of independently moving objects from dense scene flow, *Energy minimization methods in computer vision and pattern recognition*, Springer, pp. 14–27.

Weiss, S. and Siegwart, R.: 2011, Real-time metric state estimation for modular vision-inertial systems, *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, IEEE, pp. 4531–4537.

Westerhoff, J., Lessmann, S., Meuter, M., Siegemund, J. and Kummert, A.: 2016, Development and comparison of homography based estimation techniques for camera to road surface orientation, *Intelligent Vehicles Symposium (IV)*, IEEE, pp. 1034–1040.

Wikimedia, C.: 2014, File:dashcams p1210477.jpg — wikimedia commons, the free media repository. [Online; accessed 20-February-2018].
URL: *https://commons.wikimedia.org/w/index.php?title=File:Dashcams_P1210477.JPG&oldid=126617054*

Wikimedia, C.: 2017, File:chi-square pdf.svg — wikimedia commons, the free media repository. [Online; accessed 22-February-2018].
URL: *https://commons.wikimedia.org/w/index.php?title=File:Chi-square_pdf.svg&oldid=269278845*

Wong, C.-C., Siu, W.-C., Jennings, P., Barnes, S. and Fong, B.: 2015, A smart moving vehicle detection system using motion vectors and generic line features, *IEEE Transactions on Consumer Electronics* **61**(3), 384–392.

Wulff, J., Sevilla-Lara, L. and Black, M. J.: 2017, Optical flow in mostly rigid scenes, *arXiv preprint arXiv:1705.01352* .

Yamaguchi, K., Kato, T. and Ninomiya, Y.: 2006, Vehicle ego-motion estimation and moving object detection using a monocular camera, *18th International Conference on Pattern Recognition (ICPR'06)*, Vol. 4, pp. 610–613.

Yamaguchi, K., McAllester, D. and Urtasun, R.: 2013, Robust monocular epipolar flow estimation, *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, IEEE, pp. 1862–1869.

Younes, G., Asmar, D. and Shammas, E.: 2016, A survey on non-filter-based monocular visual SLAM systems, *arXiv preprint arXiv:1607.00470* .

Zhang, Z. and Hanson, A. R.: 1996, 3D reconstruction based on homography mapping, *Proc. ARPA96* pp. 1007–1012.

Zhou, D., Dai, Y. and Li, H.: 2016, Reliable scale estimation and correction for monocular visual odometry, *Intelligent Vehicles Symposium (IV)*, IEEE, pp. 490–495.

Zhou, D., Frémont, V., Quost, B., Dai, Y. and Li, H.: 2017, Moving object detection and segmentation in urban environments from a moving platform, *Image and Vision Computing* .

Zitova, B. and Flusser, J.: 2003, Image registration methods: A survey, *Image and vision computing* **21**(11), 977–1000.

Curriculum Vitae

NOLANG FANANI

Education

- Ph.D. in Computer Science (2014 - 2018)
  Goethe University of Frankfurt am Main, Germany
  Dissertation: Predictive Monocular Odometry Using Propagation-based Tracking
  Supervisor: Prof. Dr-Ing. Rudolf Mester
- Master in Computer Vision and Robotics (2011 - 2013)
  European Erasmus Mundus Programme (VIBOT) Consortium: Universite de Bourgogne (France), Universitat de Girona (Spain), Heriot Watt University (United Kingdom)
- Bachelor in Electrical and Electronic Engineering (2005 - 2009)
  Nanyang Technological University, Singapore

Publications

- **Nolang Fanani**, Alina Stürck, Matthias Ochs, Henry Bradler, and Rudolf Mester. "Predictive monocular odometry (PMO): What is possible without RANSAC and multiframe bundle adjustment?" Image and Vision Computing (2017).
- **Nolang Fanani**, Marc Barnada, and Rudolf Mester. "Motion priors estimation for robust matching initialization in automotive applications." International Symposium on Visual Computing. Springer International Publishing, 2015.
- **Nolang Fanani**, Matthias Ochs, Henry Bradler, and Rudolf Mester. "Keypoint trajectory estimation using propagation based tracking." IEEE Intelligent Vehicles Symposium (IV), 2016.
- **Nolang Fanani**, Rudolf Mester. "Propagation based tracking with uncertainty measurement in automotive applications." IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI), 2016.
- **Nolang Fanani**, Alina Stürck, Marc Barnada, and Rudolf Mester. "Multimodal Scale Estimation for Monocular Visual Odometry." IEEE Intelligent Vehicles Symposium (IV), 2017.
- Henry Bradler, Matthias Ochs, **Nolang Fanani**, and Rudolf Mester. "Joint Epipolar Tracking (JET): Simultaneous optimization of epipolar geometry and feature correspondences."IEEE Winter Conference on Applications of Computer Vision (WACV), 2017.
- **Nolang Fanani** and Rudolf Mester. "The precision of triangulation in monocular visual odometry". IEEE Southwest Symposium on Image Analysis and

Interpretation (SSIAI), 2018.

- **Nolang Fanani**, Matthias Ochs, Alina Stürck, and Rudolf Mester. "CNN-based multi-frame IMO detection from a monocular camera." IEEE Intelligent Vehicles Symposium (IV), 2018.
- **Nolang Fanani**, Matthias Ochs, and Rudolf Mester. "Detecting parallel-moving objects in the monocular case employing CNN depth maps". European Conference on Computer Vision (ECCV) Workshop - Geometry Meets Deep Learning, 2018.

Awards

- Best Paper Award (Second Prize) - IEEE Intelligent Vehicle (IV) 2018
- DAAD Research PhD Scholarship (2014 - 2018)
- Erasmus Mundus Master Scholarship (2011 - 2013)
- Undergraduate Scholarship (2005 - 2009)
- Silver Award, NTU Student Union BP Mentoring, Singapore (2008)
- Bronze Award, Asian Pacific Mathematics Olympiad 2004